université de BORDEAUX

LaBRI

THÈSE

présentée à

L'UNIVERSITÉ DE BORDEAUX

École Doctorale de Mathématiques et Informatique de Bordeaux

par

Julien BENSMAIL

pour obtenir le grade de

DOCTEUR

SPÉCIALITÉ : **INFORMATIQUE**

Partitions et décompositions de graphes

Soutenue le 10 juin 2014 au Laboratoire Bordelais de Recherche en Informatique (LaBRI)

Après avis des rapporteurs :

Frédéric Havet	Directeur de Recherches CNRS à Sophia-Antipolis (France)
Ingo Schiermeyer	Professeur à l'Université de Freiberg (Allemagne)

Devant la commission d'examen composée de :

Examinateurs

Cyril Gavoille	Professeur à l'Université de Bordeaux (France)
Brett Stevens	Professeur à l'Université de Carleton (Canada)
Mariusz Woźniak	Professeur à l'Université AGH de Cracovie (Pologne)

Rapporteurs

Frédéric Havet	Directeur de Recherches CNRS à Sophia-Antipolis (France)
Ingo Schiermeyer	Professeur à l'Université de Freiberg (Allemagne)

Directeurs de thèse

Olivier Baudon	Maître de Conférences à l'Université de Bordeaux (France)
Éric Sopena	Professeur à l'Université de Bordeaux (France)

-2014-

Partitions and decompositions of graphs

Abstract:

This thesis is dedicated to the study of two families of graph partition problems.

First, we consider the problem of vertex-partitioning a graph into connected subgraphs. Namely, given p positive integers $n_1, n_2, ..., n_p$ summing up to the order of some graph G, can we partition $V(G)$ into p parts $V_1, V_2, ..., V_p$ so that each V_i induces a connected subgraph with order n_i? We then consider stronger questions. Namely, what if we want G to be partitionable whatever are p and $n_1, n_2, ..., n_p$? What if we also want specific vertices of G to belong to some specific subgraphs induced by the vertex-partition? What if we want the subgraphs induced by the vertex-partition to be more than connected? We consider all these questions regarding both the structural (are there structural properties ensuring that a specific vertex-partition necessarily exists?) and algorithmic (is it hard to deduce a specific vertex-partition?) points of view.

Then, we focus on the so-called 1-2-3 Conjecture, which asks whether every graph G admits a neighbour-sum-distinguishing 3-edge-weighting, i.e. a 3-edge-weighting by which all adjacent vertices of G get distinguished by their sums of incident weights. As a tool to deal with the 1-2-3 Conjecture, we notably introduce the notion of locally irregular edge-colouring, which is an edge-colouring in which every colour class induces a subgraph whose adjacent vertices have distinct degrees. The main point is that, in particular situations, a neighbour-sum-distinguishing edge-weighting of G can be deduced from a locally irregular edge-colouring of it. Our concerns in this context are mostly algorithmic (can we easily find a neighbour-sum-distinguishing edge-weighting or locally irregular edge-colouring using the least number of weights or colours?) and structural (what is the least number of colours in a locally irregular edge-colouring?). We also consider similar matters in the context of oriented graphs.

Keywords:

partition into connected subgraphs, (preassignable, on-line, recursively) arbitrarily partitionable graph, neighbour-distinguishing edge- and arc-weighting, locally irregular edge- and arc-colouring

Laboratoire Bordelais de Recherche en Informatique (LaBRI)
Université de Bordeaux
351, cours de la Libération
33405 Talence Cedex, France

Partitions et décompositions de graphes

Résumé :

Cette thèse est dédiée à l'étude de deux familles de problèmes de partition de graphe.

Nous considérons tout d'abord le problème de sommet-partitionner un graphe en sous-graphes connexes. Plus précisément, étant donnés p entiers positifs $n_1, n_2, ..., n_p$ dont la somme vaut l'ordre d'un graphe G, peut-on partitionner $V(G)$ en p parts $V_1, V_2, ..., V_p$ de sorte que chaque V_i induise un sous-graphe connexe d'ordre n_i ? Nous nous intéressons ensuite à des questions plus fortes. Que peut-on dire si l'on souhaite que G soit partitionnable de cette manière quels que soient p et $n_1, n_2, ..., n_p$? Si l'on souhaite que des sommets particuliers de G appartiennent à des sous-graphes particuliers de la partition ? Et si l'on souhaite que les sous-graphes induits soient plus que connexes ? Nous considérons toutes ces questions à la fois du point de vue structurel (sous quelles conditions structurelles une partition particulière existe-t-elle nécessairement ?) et algorithmique (est-il difficile de trouver une partition particulière ?).

Nous nous intéressons ensuite à la 1-2-3 Conjecture, qui demande si tout graphe G admet une 3-pondération voisin-somme-distinguante de ses arêtes, i.e. une 3-pondération par laquelle chaque sommet de G peut être distingué de ses voisins en comparant uniquement leur somme de poids incidents. Afin d'étudier la 1-2-3 Conjecture, nous introduisons notamment la notion de coloration localement irrégulière d'arêtes, qui est une coloration d'arêtes dont chaque classe de couleur induit un sous-graphe dans lequel les sommets adjacents sont de degrés différents. L'intérêt principal de cette coloration est que, dans certaines situations, une pondération d'arêtes voisin-somme-distinguante peut être déduite d'une coloration d'arêtes localement irrégulière. Nos préoccupations dans ce contexte sont principalement algorithmiques (est-il facile de trouver une pondération d'arêtes voisin-somme-distinguante ou une coloration d'arêtes localement irrégulière utilisant le plus petit nombre possible de poids ou couleurs ?) et structurelles (quel est le plus petit nombre de couleurs d'une coloration d'arêtes localement irrégulière ?). Nous considérons également ces questions dans le contexte des graphes orientés.

Mots-clefs :

partition en sous-graphes connexes, graphe (récursivement) arbitrairement partitionnable (k-préassignable, à la volée), coloration voisin-distinguante d'arêtes ou d'arcs, coloration localement irrégulière d'arêtes ou d'arcs

Laboratoire Bordelais de Recherche en Informatique (LaBRI)
Université de Bordeaux
351, cours de la Libération
33405 Talence Cedex, France

Remerciements

Bien que les remerciements constituent généralement le seul passage un peu amusant (et facile à comprendre) d'un manuscrit de thèse, on ne se rend pas compte, avant d'en écrire soi-même, de la délicatesse que cette tâche représente. D'une part car il faut mesurer le sens de chaque mot employé (au risque de pouvoir froisser des lecteurs) et surtout n'oublier personne, ce qui peut vite arriver tant le nombre de personnes que l'on peut être amené à côtoyer en trois ans est grand. Et d'autre part car écrire ses remerciements demande de se remémorer tout ce qui a pu se passer en trois ans, faisant au passage bien comprendre, si besoin en était, que la thèse est désormais bien terminée. Ceci étant dit, c'est donc à mon tour de me lancer, dans la joie et l'allégresse :) .

Il m'est inconcevable de ne pas adresser mes premiers remerciements à Olivier Baudon et Éric Sopena, mes deux directeurs de thèse, qui, grâce à leurs qualités complémentaires, m'ont permis, je le crois, d'effectuer cette thèse dans les meilleures conditions. Je souhaite à tout doctorant d'avoir, dans son entourage, quelqu'un ayant autant de recul qu'Olivier sur le monde de l'enseignement et de la recherche. Non seulement ses conseils et anecdotes sont (souvent) amusants (bien que certains se répètent un peu parfois !), mais ceux-ci sont surtout une source d'inspiration et de remise en question constante sur la condition d'enseignant-chercheur. Ce qui est très bien pour garder la tête sur les épaules. Quant à Éric, je reste surtout très impressioné par sa capacité à s'intéresser à (à peu près) n'importe quel problème de théorie des graphes (notamment), et par sa patience pour relire tout type de document à à peu près n'importe quelle heure du jour ou de la nuit (généralement très tard, ou très tôt selon certains). Je ne saurai jamais assez les remercier tous les deux de m'avoir autant appris lors de ces trois dernières années, et pour m'avoir permis d'effectuer cette thèse de manière aussi libre, en me permettant notamment de vagabonder sur des problématiques n'ayant généralement pas de rapport avec mon sujet d'origine.

Je remercie ensuite tous les membres de mon jury de thèse de s'être intéressés à mes travaux. Merci donc tout d'abord à Frédéric Havet et Ingo Schiermeyer pour avoir accepté de rapporter mon manuscrit de thèse, malgré sa longueur, et pour leurs commentaires et critiques constructifs sur celui-ci. Merci également à Cyril Gavoille pour avoir accepté de présider le jury (encore une fois, après celui de ma soutenance de Master), ainsi qu'à Brett Stevens et Mariusz Woźniak d'avoir été présents lors de la soutenance, ce qui m'a réellement touché compte tenu des liens que j'ai pu tisser avec eux ces trois dernières années. Je remercie également tout ce beau monde pour les questions intéressantes qu'ils m'ont posées lors de la séance de questions.

Toujours en rapport avec ma soutenance de thèse, je remercie également toutes les personnes présentes lors de celle-ci. Merci à tous mes collègues et/ou amis pour leur présence. Je suis très sensible au fait que certains de ceux-ci, de même que les membres de ma famille, aient parfois dû effectuer un long trajet pour pouvoir être là. Merci également aux quelques personnes qui n'ont pu être présentes mais ont néanmoins pris la peine de m'envoyer un mail pour me soutenir ou me féliciter.

Une thèse pouvant difficilement être menée à bien dans de mauvaises conditions, je m'estime assez priviligié d'avoir pu effectuer ma thèse au LaBRI, qui constitue un environnement très dynamique et stimulant au quotidien. Je tiens donc à avoir une pensée pour toutes les personnes (chercheurs, personnels administratifs, techniques...) qui contribuent à la bonne humeur et à l'ambiance bon enfant au quotidien.

Il me paraît légitime de remercier plus chaleureusement les personnes du LaBRI desquelles j'ai été le plus proche. À ce titre, il m'est impensable de ne pas mentionner celles et ceux qui ont effectué leur thèse en même temps que moi, car, quoi qu'on en dise, être dans la même galère forge forcément des liens :) . Au sein du thème Graphes et Applications, je pense notamment à Clément Charpentier (la personne derrière la notion avant-gardiste de "vertex crossing number"), Émilie

Diot (pour sa bonne humeur quotidienne), Florent Foucaud (que je remercie tout particulièrement pour le thème de son manuscrit de thèse, que le présent document utilise), Hervé Hocquard (qui m'a initié à la méthode dite du déchargement), Gabriel Renault (avec qui nous nous sommes bien amusés à Pise, malgré les problèmes de vols), Sagnik Sen (I do hope our quite ambitious paper on signed graphs will be published somewhere), et Petru Valicov (merci encore pour la super semaine à Lyon). Parmi les autres doctorants, je remercie notamment Vincent Autefage (mais où trouve-t-il autant de temps pour faire tout ça ?), Romaric Duvignau (on a passé une bonne semaine à Perpignan), Noël Gillet (il faudrait que l'on retourne faire un basket-ball un jour), Christian Glacet (parce qu'enseigner de l'Assembleur, forcément ça rapproche), et Ahmed Wade (idem, mais pour les Bases de Données). Et, bien évidemment, une ligne spéciale pour Cédric Teyton, avec qui j'ai passé le plus clair de mon temps ces trois dernières années. En recherche, je ne sais pas, mais en tout cas, en ping-pong, nous avons fait de belles choses :) . À vous tous, je vous souhaite une très grande carrière.

Enfin, toujours en ce qui concerne le LaBRI, je tiens également à remercier plus particulièrement certains membres permanents. Parmi les personnes que je n'ai pas encore mentionnées (et qui ne le seront pas plus bas), je retiendrai notamment celles ayant toujours témoigné de la sympathie à mon égard ou de l'intérêt pour mes travaux. Je pense en particulier à Nicolas Bonichon (qui a toujours le sourire), Pierre Castéran (toujours le premier arrivé au laboratoire), Olivier Delmas (toujours un bonjour dans les couloirs), Philippe Duchon (qu'on apprécie toujours avoir dans les publics de séminaire), Olivier Guibert (l'un des rares enseignants de l'IUT de Bordeaux se rappelant que j'en viens), František Kardoš (merci infiniment de m'avoir emmené en Slovaquie), Ralf Klasing (toujours une intention sympa), et Arnaud Pêcher (idem, toujours de bonne humeur). Également, je tiens à remercier de nouveau Brett Stevens pour notre collaboration tout au long de cette dernière année qu'il a passée au LaBRI. Il est vraiment très agréable de pouvoir travailler avec quelqu'un ayant autant de temps libre, et, bien que nous n'ayons pas toujours été très heureux dans nos résultats, j'ai beaucoup appris de toutes nos tentatives pour traiter certains problèmes compliqués.

En parallèle de mes recherches, j'ai également eu la chance de pouvoir enseigner à l'IUT informatique de Bordeaux, lieu très particulier pour moi puisque j'y ai démarré mes études universitaires. À ce titre, je ne remercierai jamais assez Isabelle Dutour pour m'avoir donné cette opportunité, ainsi que pour avoir toujours été bienveillante à mon égard. Depuis cette expérience, je tiens vraiment à conserver une composante "enseignement" dans mon plan de carrière. Sans nul doute cela resulte des bonnes conditions dans lesquelles se sont passées ces trois années d'enseignement. Il me semble alors naturel de remercier tous les enseignants de l'IUT, qui ont contribué à mon intégration au sein de l'équipe. Je tiens plus particulièrement à remercier Romain Bourqui, Arnaud Casteigts (qui a toujours eu un mot gentil à mon encontre), Patrick Félix, Colette Johnen et Nicholas Journet, avec qui j'ai eu l'occasion d'intervenir dans plusieurs matières.

L'une des choses que je retiendrai de mon expérience dans l'enseignement est que, à mon sens, un bon enseignant doit être proche de ses étudiants. Mais bien entendu, il faut que la connexion se fasse dans les deux sens pour qu'un cours se déroule dans les meilleures conditions. Je tiens donc à remercier ici tous les étudiants qui ont bien voulu jouer le jeu et ont eu une attitude positive lors de mes cours, les rendant (généralement) plus agréables. J'espère sincèrement que mes interventions leur auront servi d'une manière ou d'une autre, car s'il est difficile d'être un bon chercheur, il l'est encore plus d'être un bon enseignant. De la même manière, je tiens à remercier tous les étudiants que j'ai pu encadrer à l'occasion de projets. Je leur souhaite à tous une très bonne continuation, quelle que soit la direction qu'ils puissent choisir.

Une partie de la thèse se déroulant à l'extérieur, je tiens également à remercier toutes les personnes avec qui j'ai pu échanger sur des sujets divers et variés lors de conférences, séminaires, écoles... Je pense ici naturellement à Marthe Bonamy (qui devrait se faire payer par Hervé

pour les slides d'EuroComb 2013), Nicolas Gastineau (encore désolé de n'avoir pu être présent à ta soutenance de thèse), Ararat Harutyunyan (I do hope we will meet in Lyon soon), Sergey Kirgizov (merci encore d'être venu de Paris pour ma soutenance), et Reza Naserasr (thank you for inviting me to a nice working week in Thézac)

During the Ph.D. period, I also had the opportunity to closely collaborate with people from foreign universities. First, I would like to thank Mariusz Woźniak and his team at AGH University of Kraków, Poland, for inviting me twice. Thank you in particular to Rafał Kalinowski, Antoni Marczyk, Mateusz Nikodem, Monika Piśniak and Jakub Przybyło for their sympathy and for interesting research discussions during my two stays. My thanks again to Mateusz and his wife for bringing me to an interesting visit of the Kazimierz district. Second, I would also like to thank Roman Soták and his colleagues at Pavol Jozef Šafárik University of Košice, Slovakia, for receiving me once for an interesting collaboration about some conjecture by Vizing. I will always remember the nice basketball game Roman took me to, as well as the wonderful Imaginations festival. Anyway, thank you all guys for your sympathy and for all research work we have been interested into together.

Contents

Chapter 1
Introduction

This thesis is dedicated to the study of two graph partition problems. So that the context of this thesis is clear, we start by defining, in Section 1.1, what we consider to be a partition problem in graph theory. In particular, we recall what these problems are about and which questions are of interest regarding these. We then briefly present the two families of graph partition problems considered in this thesis.

The rest of this chapter is dedicated to the introduction of most of the materials which are necessary to understand our investigations. In Section 1.2, we give most of the general definitions (as well as some related results), terminology and notation which are used throughout this thesis. These cover general mathematics, and graph, probability and computational complexity theories. We end up this introductory chapter by listing some decision problems and their respective complexity in Section 1.3.

1.1 Context of the thesis

A graph partition problem is basically a problem asking whether the elements (generally either its vertices or its edges, or both of them) of a graph G can be partitioned into parts $P_1, P_2, ..., P_k$ in such a way that a set of partition constraints is not violated. We generally make the distinction between two kinds of such partition constraints:

Intra-constraints: constraints each of $P_1, P_2, ..., P_k$ must respect,

Inter-constraints: constraints every two distinct P_i and P_j must respect.

Many problems of practical interest (e.g. scheduling problems, allocation problems, etc.) can be expressed as graph partition problems, so studying a practical problem from the graph theory point of view can be a convenient way to solve it. Surely the investigations leading to the well-known Four-Colour Theorem are the most pertinent illustration of this statement. In the 1850's, Guthrie addressed the following question: given a map, is it always possible to colour its regions using at most four colours so that every two regions sharing a common border are not coloured with the same colour? This question, though its simple formulation, showed up to

be quite challenging and awaited more than one century to be answered in the affirmative by Appel and Haken in the 1970's as a graph partition problem [9]. More precisely, they showed that the vertex set $V(G)$ of every planar graph G can be partitioned into four parts V_1, V_2, V_3, V_4 satisfying the intra-constraint that no edge joins two vertices from a same part. This result implies the positiveness of Guthrie's question.

Since the above mentioned proof of the Four-Colour Theorem, graph partition problems, and more particularly graph colouring problems, have become an important field of research in combinatorics and algorithmic, with hundreds problems of various complexity being introduced and receiving ingrowing attention in the last decades. Every such partition problem is generally studied regarding several recurrent aspects, like e.g. the *algorithmic aspects* of finding a correct partition of a graph, the *structural properties* of partitionable graphs, or the *optimization concern*, i.e. what can be considered the best partition of a graph?

Two graph partition problems are investigated in this thesis, which we briefly present below. Refer to Chapters 2 and 7 for complete introductions dedicated to these problems.

- In Part I, we consider several variants of a graph partition problem where one basically aims at finding a partition of the vertex set of a graph G where each part induces a connected subgraph. We in particular focus, mainly in Chapter 3, on the notion of *arbitrarily partitionable graphs*, which are graphs that can be partitioned into arbitrarily many arbitrarily large connected subgraphs. In Chapter 4, we study the consequences of additionally requesting vertices to belong to such or such subgraphs of a graph partition. Replacing the original connectivity constraint by a stronger constraint, namely a recursive constraint, we finally study, in Chapter 5, those graphs that are partitionable into arbitrarily many arbitrarily large arbitrarily partitionable subgraphs. Our concerns regarding all these notions are mostly algorithmic (i.e. is it easy to find a partition of a given graph? or is it easy to recognize a non-partitionable graph?) and structural (i.e. what does a partitionable graph look like?).
- In Part II, we investigate two graph colouring and weighting problems (which are specific cases of partition problems) in which one aims at making the adjacent vertices of a graph G distinguishable regarding some parameters related to a colouring or weighting of the edges of G. In the first weighting problem, investigated in Chapter 8, two adjacent vertices are considered distinguished when the sums of their incident weights are different. In the second colouring problem, which is studied in Chapter 9, two adjacent vertices u and v are considered distinguished when their degrees in the subgraph induced by the colour assigned to uv are different. These two problems are also considered regarding oriented graphs in Chapters 10 and 11, respectively. All these notions are investigated notably with regards to the algorithmic (i.e. can we easily find a distinguishing weighting or colouring of a graph?) and the optimization (i.e. what is the least number of weights or colours used by a distinguishing weighting or colouring of a graph?) points of view.

1.2 Definitions, notation, terminology, and related results

Since a lot of materials from different fields are introduced herein (especially regarding graph theory), the reader may note that some of the terminology and notation we introduce overlap, which could be a source of confusion in specific contexts. So we will make sure to not use two conflicting terminologies or notation simultaneously throughout this thesis.

1.2.1 General mathematics

Informally, a *multiset* is a set where each element can appear more than once. Although a multiset M should be rigorously defined as a couple (A, f), where A is the set of elements appearing in

M and $f(a)$ is the number of occurrences of every member a of A in M, we herein privilege a set notation $M = \{n_1, n_2, ..., n_p\}$ for the sake of simplicity, where $f(a)$ of the n_i's are equal to a in M for every $a \in A$. A *partition* of an integer n is a multiset $\{n_1, n_2, ..., n_p\}$ summing up to n, i.e. we have $\sum_{i=1}^{p} n_i = n$.

Example 1.1. The multiset $\{1, 1, 1, 4, 5, 5\}$ forms a partition of 17.

The number of partitions of an integer n is known as the *partition number* of n, and is commonly denoted $p(n)$. The partition number function p has been attracting much interest in number theory, refer e.g. to the book [8] of Andrews dedicated to this topic. One important property of p is that it asymptotically grows as the exponential function, see e.g. the reference book [59] of Flajolet and Sedgewick wherein a proof of this statement is provided.

Theorem 1.2 ([59])**.** *The partition number $p(n)$ asymptotically tends to $\frac{1}{4n\sqrt{3}} \exp\left(\pi\sqrt{\frac{2n}{3}}\right)$.*

Let S be a set of mathematical objects. We say that S is *partitioned* into the k *parts* $S_1, S_2, ..., S_p$ if we have $s \in \bigcup_{i=1}^{p} S_i$ for every $s \in S$, and $S_i \cap S_j = \emptyset$ for every $i \neq j$. We most of the time denote the resulting *partition* of S either $(S_1, S_2, ..., S_p)$ or $S_1 \cup S_2 \cup ... \cup S_p$. A partition into two parts is sometimes called a *bipartition*.

1.2.2 Graph theory

For the sake of completeness, we herein present most of the graph theory notions and terminology which are used throughout this thesis. Please refer to the reference books by Bondy and Murty [36] or Diestel [48] for more information on any further detail we would have not presented herein.

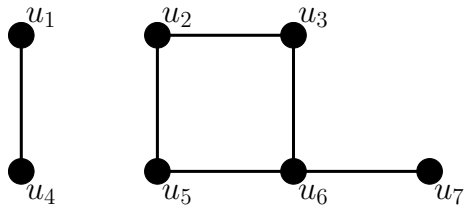
1.2.2.1 Different kinds of graphs

An *undirected (simple) graph* G is made up of a set $V(G)$ of *vertices*, along with a set $E(G)$ of pairs of vertices, called *edges*. Such a graph is sometimes denoted $G = (V(G), E(G))$. We refer to $|V(G)|$ and $|E(G)|$ as the *order* and the *size* of G, respectively. An edge $\{u, v\}$ of G is rather denoted uv, with u and v being the *ends* of uv. An edge and its ends are said to be *incident*, while two vertices u and v of G are said *adjacent* (or *neighbouring*) if $uv \in E(G)$, i.e. u and v are joined by an edge in G. Similarly, we say that two edges of G are *adjacent* if they share an end.

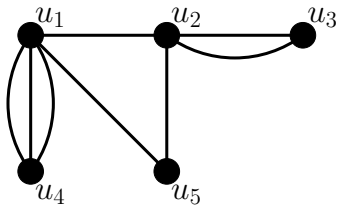
An undirected *multigraph* is an undirected graph in which two adjacent vertices may be joined by more than one edge. A multigraph G can hence be formally described as a set of vertices and a multiset of edges over these vertices (which we still denote $E(G)$ for convenience). Two edges of a multigraph with the same ends are said *parallel*.

Hypergraphs are a generalization of undirected simple graphs where each edge may connect more than just two vertices. Formally, a hypergraph H is made up of one vertex set $V(H)$ and one set $E(H)$ of *hyperedges*, which are subsets of $V(H)$. Clearly a hypergraph having all of its edges consisting in one or two elements is nothing but an undirected graph.

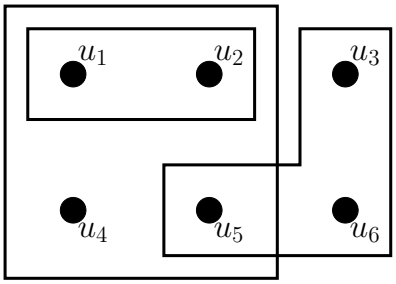
A *directed graph* D consists of vertices which can be joined by means of two kinds of edges, rather called *arcs* in this context (therefore, we denote $A(D)$ the arc set of D). First, every two vertices u and v of D can be joined by an antisymmetric arc, that is an edge which is assigned a direction. A *directed arc* from u to v is denoted $\overrightarrow{uv}$, where the over right arrow is used to make the direction of the antisymmetric arc joining u and v clear. In such a situation, the vertex u is called the *tail* of $\overrightarrow{uv}$, while v is called the *head* of $\overrightarrow{uv}$. We sometimes say that $\overrightarrow{uv}$ is *outgoing* from u, and *ingoing* to v. Second, u and v can be joined by a symmetric arc, which is similar to an edge in an undirected graph. Equivalently, such an arc can be seen as being directed both from u and v and from v to u.



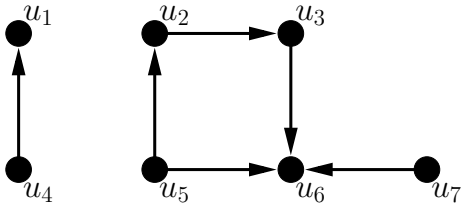
(a) An undirected graph with vertex set $\{u_1, u_2, u_3, u_4, u_5, u_6, u_7\}$ and edge set $\{u_1u_4, u_2u_3, u_2u_5, u_3u_6, u_5u_6, u_6u_7\}$.



(b) A multigraph with vertex set $\{u_1, u_2, u_3, u_4, u_5\}$ and edge multiset $\{u_1u_4,\ u_1u_4,\ u_1u_4,\ u_1u_2,\ u_1u_5,\ u_2u_5,\ u_2u_3,\ u_2u_3\}$.



(c) A hypergraph with vertex set $\{u_1, u_2, u_3, u_4, u_5, u_6\}$ and hyperedge set $\{\{u_1, u_2\}, \{u_1, u_2, u_4, u_5\}, \{u_3, u_5, u_6\}\}$.



(d) A directed graph with vertex set $\{u_1, u_2, u_3, u_4, u_5, u_6, u_7\}$ and arc set $\{\overrightarrow{u_4u_1}, \overrightarrow{u_2u_3}, \overrightarrow{u_5u_2}, \overrightarrow{u_3u_6}, \overrightarrow{u_5u_6}, \overrightarrow{u_7u_6}\}$.

Figure 1.1: Four examples of different kinds of graphs.

An *oriented graph* $\overrightarrow{G}$ is a directed graph with no symmetric arcs. Equivalently, every oriented graph can be obtained from an undirected graph G by *orienting* each edge uv of G either from u to v, or from v to u. From this point of view, we hence call $\overrightarrow{G}$ an *orientation* of G. Conversely, we call G the undirected graph *underlying* $\overrightarrow{G}$, and denote it $und(\overrightarrow{G})$.

Example 1.3. Figure 1.1 depicts examples of an undirected graph (Figure 1.1.a), a multigraph (Figure 1.1.b), a hypergraph (Figure 1.1.c) and a directed graph (Figure 1.1.d). The undirected graph from Figure 1.1.a underlies the directed graph from Figure 1.1.d, which is also an oriented graph since it has no symmetric arcs.

Though the term "graph" may refer to either of the above kinds of graphs, by a *graph* it should be throughout understood that we refer to a simple undirected graph. Every mention to any other introduced above kind of graphs is done with the use of the convenient qualifier. Besides, since undirected graphs and oriented graphs are the main concerns of this thesis, we mainly introduce materials related to these graphs in this section. So again we refer the reader to the above mentioned reference books for an analogue dedicated to another type of graphs of any definition, terminology or notation introduced herein (though most of these can be intuitively guessed).

1.2.2.2 Neighbourhoods and degrees of a vertex

Let G be a graph. In case two vertices u and v of G are not adjacent, we sometimes say that u and v are *independent*. The set of vertices adjacent to v in G form its *neighbourhood* and is denoted $N_G(v)$, or simply $N(v)$ when no ambiguity is possible. The *degree* of v, denoted $d_G(v)$ (or simply $d(v)$), is the number of its adjacent vertices in G, that is $|N(v)|$. A set of pairwise independent vertices is said *independent*. The *minimum* and *maximum degrees* of G correspond to the minimum and maximum, respectively, degree of a vertex in G. These parameters are denoted $\delta(G)$ and $\Delta(G)$, respectively. A vertex with degree k is sometimes called a *k-vertex*.

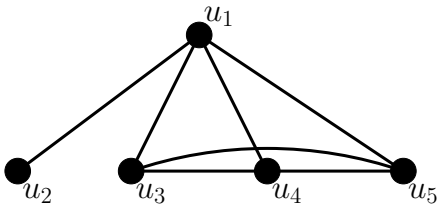


Figure 1.2: A graph with universal and hanging vertices.

A 1-vertex is sometimes referred to as an *hanging vertex*, while a $(|V(G)|-1)$-vertex is said *universal*. Assuming u and v are two adjacent vertices of G and v has degree k, we call v a *k-neighbour* of u.

Example 1.4. The vertex u_1 from the graph G in Figure 1.2 is universal, while u_2 is hanging. The neighbourhood of u_3 in G is $\{u_1, u_4, u_5\}$, and hence u_3 is a 3-vertex. The minimum degree of G is 1, which is the degree of u_2, while its maximum degree is 4, which is the degree of u_1.

Two vertices u and v of a directed graph D are considered adjacent if there is an arc (either antisymmetric or symmetric) involving both u and v. Every vertex v of D has two kinds of neighbours, namely its *inneighbours* and *outneighbours*. Formally, the sets $N_D^-(v)$ and $N_D^+(v)$, or simply $N^-(v)$ and $N^+(v)$ when it is clear from the context, of inneighbours and outneighbours, respectively, of v in D are defined as

$$N_D^-(v) = \{u \in V(D) : \overrightarrow{uv} \in A(D)\} \quad \text{and} \quad N_D^+(v) = \{u \in V(D) : \overrightarrow{vu} \in A(D)\},$$

respectively. Note that the definitions imply that if two vertices u and v are joined by a symmetric arc, then $v \in N^-(u)$ and $v \in N^+(v)$ (and similarly for u). The numbers of inneighbours and outneighbours of v in D, which correspond to $|N_D^-(v)|$ and $|N_D^+(v)|$, respectively, are called the *indegree* and *outdegree*, respectively, of v, and are denoted $d_D^-(v)$ and $d_D^+(v)$, respectively. Again, these parameters are simply denoted $d^-(v)$ and $d^+(v)$ when no confusion is possible. Similarly as for the undirected case, we denote $\delta^-(D)$ and $\Delta^-(D)$ (resp. $\delta^+(D)$ and $\Delta^+(D)$) the *minimum* and *maximum indegrees* (resp *outdegrees*) of D, i.e. the minimum and maximum indegrees (resp. outdegrees) of vertices of D.

Example 1.5. The set of inneighbours of u_6 in the oriented graph $\overrightarrow{G}$ depicted in Figure 1.1.d is $\{u_3, u_5, u_7\}$, while the set of outneighbours of u_5 is $\{u_2, u_6\}$. Both the indegree and outdegree of u_3 are equal to 1 in $\overrightarrow{G}$. We have $\delta^-(\overrightarrow{G}) = 0$ and $\Delta^-(\overrightarrow{G}) = 3$, which are the indegrees of u_4 and u_6, respectively. Besides, we have $\delta^+(\overrightarrow{G}) = 0$ and $\Delta^+(\overrightarrow{G}) = 2$, these values being the outdegrees of u_1 and u_5, respectively.

A graph G where all the vertices have the same degree, say k, is called *regular*. If for some reason we want to make the value of k explicit, then we say that G is *k-regular*. Since we have $\delta(G) = \Delta(G)$ whenever G is regular, it is understood that by the *degree* of G, we refer to the (same) degree of its vertices. We sometimes refer to a 3-regular graph as a *cubic* graph.

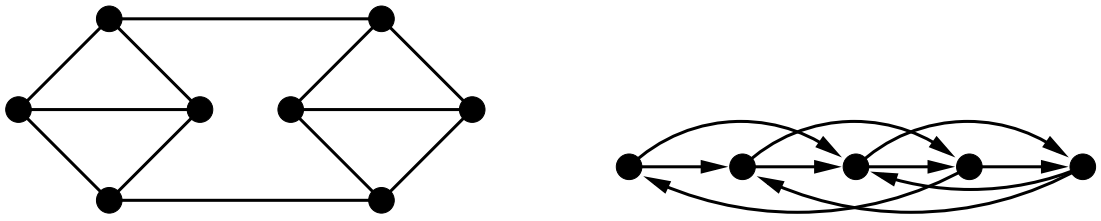
Regarding oriented graphs, we distinguish two notions of regularity. Namely, an oriented graph $\overrightarrow{G}$ in which all vertices have the same indegree, say k, is called *inregular*, or *k-inregular*. Similarly we say that $\overrightarrow{G}$ is *outregular* or *k-outregular* when all vertices of $\overrightarrow{G}$ have outdegree k.

A hypergraph H is said *k-uniform* (or more generally *uniform*) if each edge of H includes exactly k vertices. By definition, a 2-uniform hypergraph is actually a graph.

Example 1.6. In Figure 1.3.a is drawn a cubic graph, while Figure 1.3.b depicts a 2-outregular oriented graph. The oriented graph from Figure 1.3.b is not inregular since its two left-most vertices have indegree 1 and 2, respectively.

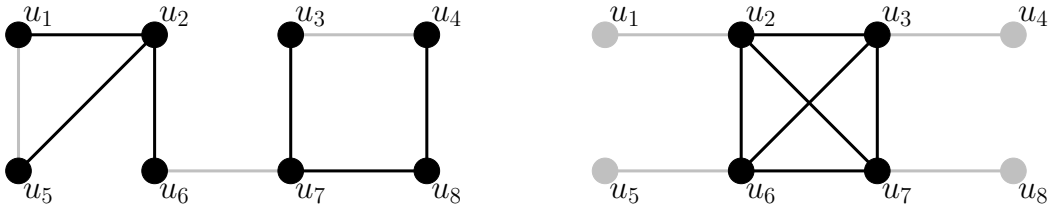
The *density* of a graph G corresponds to its ratio

$$\frac{2|E(G)|}{|V(G)|(|V(G)|-1)}.$$



(a) A 3-regular graph. (b) An outregular oriented graph.

Figure 1.3: A regular graph and an outregular oriented graph.



(a) A subgraph of a graph. (b) An induced subgraph of a graph.

Figure 1.4: Subgraphs (in black only) of two graphs (in black and grey).

Intuitively, this parameter measures how close is G to the graph with the same order and the maximum number of edges. By a *dense graph*, we generally refer to a graph with a lot of edges, i.e. with density close to 1. On the contrary, a *sparse graph* rather refers to a graph with few edges, i.e. with density close to 0. No notion of closeness has been ever unanimously adopted, so these notions of denseness and sparseness graphs are highly context sensitive.

1.2.2.3 Isomorphic graphs

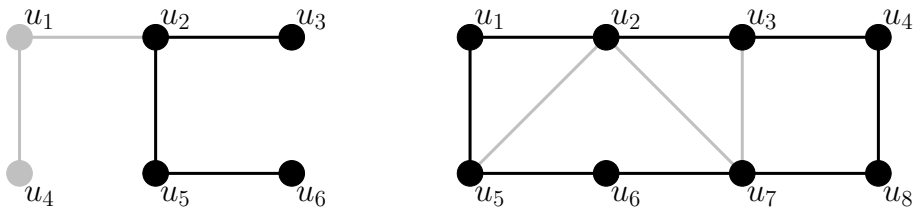
An *isomorphism* between two graphs G and H is a bijective mapping $f : V(G) \to V(H)$ preserving the adjacencies, that is uv is an edge of G if and only $f(u)f(v)$ is an edge of H. Two graphs are said *isomorphic* when there exists an isomorphism between them. By writing $G \simeq H$, we mean that G and H are isomorphic. Set differently, if G is isomorphic to H, then it means that we can obtain H from G by relabelling the vertices of G.

1.2.2.4 Subgraphs of a graph

Let G be a graph. A *subgraph* G' of G is a graph whose vertex set V' is a subset of $V(G)$ and whose edge set E' is a subset of $E(G)$ restricted to V' (i.e. for every edge $uv \in E'$, we have $u, v \in V'$). We say that G' is *trivial* if either $G' = G$ or G' is empty (i.e. $V' = \emptyset$). In case $V' = V$, we say that G' *spans* G. Conversely, we say that G is a *supergraph* of G'. If, for every two vertices u and v of G', we have $uv \in E'$ if and only if $uv \in E(G)$, then we say that G' is an *induced subgraph* of G. Note that for some vertex u of G, the values $d_G(u)$ and $d_{G'}(u)$ can be different. This justifies the above introduction of the notation d_G.

Let $S \subseteq V(G)$ be a subset of vertices. We denote by $G[S]$ the subgraph of G *induced* by S, that is the induced subgraph of G with vertex set S. Similarly, given a subset $F \subseteq E(G)$ of edges of G, the induced subgraph $G[F]$ of G is the graph whose edges are those in F and whose vertices are those of G which are incident to edges in F.

Example 1.7. The subgraph G'_1 of the graph G_1 depicted in Figure 1.4.a is a spanning subgraph of G_1. It is however not induced since u_1 and u_5 are vertices of G'_1, but u_1u_5 is an edge of G_1 which does not belong to G'_1. The subgraph G'_2 of the graph G_2 depicted in Figure 1.4.b is induced by $\{u_2, u_3, u_6, u_7\}$.



(a) A path of a graph. (b) A cycle of a graph.

Figure 1.5: A path and a cycle (in black only) of two graphs (in black and grey).

Let G' and G'' be two subgraphs of G. We say that G' and G'' are *vertex-disjoint* if $V(G') \cap V(G'') = \emptyset$, i.e. no vertex of G belongs both to G' and G''. Similarly, we say that G' and G'' are *edge-disjoint* if no edge of G belongs both to G' and G''. A *clique* refers to an induced subgraph G' of G in which every two vertices are adjacent, i.e. we have $uv \in E(G')$ for every two vertices $u, v \in V(G')$. The *clique number* of G, denoted $\omega(G)$, is the order of a largest clique of G. Given a graph H with $|V(H)| \leq |V(G)|$, we say that G is *H-free* if no induced subgraph of G is isomorphic to H. This notion extends to more than one pattern: given k graphs $H_1, H_2, ..., H_k$ with smaller order than G, we say that G is *$\{H_1, H_2, ..., H_k\}$-free* if no induced subgraph of G is isomorphic to one of $H_1, H_2, ..., H_k$.

Example 1.8. The subgraph G' induced by $\{u_2, u_3, u_6, u_7\}$ of the graph G drawn in Figure 1.4.b is a clique with order 4. Since it is the largest clique of G, we have $\omega(G) = 4$.

1.2.2.5 Paths and cycles

A *path* of a graph G is a sequence $(v_1, v_2, ..., v_k)$ of distinct[1] vertices such that $v_i v_{i+1}$ is an edge of G for every $i \in \{1, 2, ..., k-1\}$. For the sake of simplicity, such a path is denoted $v_1 v_2 ... v_k$. Since a path of G is nothing but a subgraph of G, the notion of *order* of $v_1 v_2 ... v_k$ directly makes sense. This parameter, which is equal to k, is denoted $|v_1 v_2 ... v_k|$. The *length* of $v_1 v_2 ... v_k$, denoted $\|v_1 v_2 ... v_k\|$, refers to its number of edges (which is $k-1$). A path with order 1 is called *trivial*. We call v_1 and v_k the *first* and *last vertices* of $v_1 v_2 ... v_k$, respectively, these two vertices being the *endvertices* of $v_1 v_2 ... v_k$. To clarify the endvertices of a path $v_1 v_2 ... v_k$ of G, we call $v_1 v_2 ... v_k$ a *$\{v_1, v_k\}$-path*. A vertex of a path which is not an endvertex is sometimes called an *inner* vertex. By the *endedges* of a path, we similarly refer to those at most two edges whose at least one end has degree 1.

A *cycle* of a graph G is a sequence $(v_1, v_2, ..., v_k)$ of distinct[1] vertices such that $v_1 v_2 ... v_k$ is a path of G and $v_k v_1 \in E(G)$. A cycle with consecutive vertices $v_1, v_2, ..., v_k$ is denoted $v_1 v_2 ... v_k v_1$ for short. The notions of *order* and *length* (and their associated terminologies) of $v_1 v_2 ... v_k v_1$ are defined analogously as for paths. Note that, contrary to paths, the order and the length of a cycle are always equal. A graph with only one induced cycle is said *unicyclic*. The *girth* of G is the length of its smallest cycles. Conversely, by the *circumference* of G we refer to the length of its longest cycles. These two values are equal to infinity when G has no induced cycles. A cycle with length 3 of G is sometimes called a *triangle*.

Example 1.9. The sequence (u_3, u_2, u_5, u_6) of vertices forms a path with order 4 and length 3 in the graph G_1 depicted in Figure 1.5.a, so $u_3 u_2 u_5 u_6$ is a $\{u_3, u_6\}$-path of G_1. The girth and circumference of G_1 are equal to infinity since G_1 has no induced cycles. The sequence $(u_1, u_2, u_3, u_4, u_8, u_7, u_6, u_5)$ forms a cycle with length 8 in the graph G_2 depicted in Figure 1.5.b, but G_2 is not unicyclic as e.g. $u_1 u_2 u_5 u_1$ is another cycle of G_2. The girth of G_2 is 3, which is the length of the triangle $u_1 u_2 u_5 u_1$, while its circumference is 8, which is the length of $u_1 u_2 u_3 u_4 u_8 u_7 u_6 u_5 u_1$.

[1] We only deal with *simple* paths or cycles throughout this thesis, that is paths or cycles with no repeated vertices (aside from the first and last vertices in the case of cycles).

The order of the longest paths of G is denoted $\varsigma(G)$. A path or cycle with order $|V(G)|$ of G is said *Hamiltonian*. A graph with an Hamiltonian path is said *traceable*, while an *Hamiltonian graph* refers to a graph with an Hamiltonian cycle. In case G has an Hamiltonian $\{u,v\}$-path for every $u \neq v \in V(G)$, we call G *Hamiltonian-connected*.

Example 1.10. The path $u_3u_2u_5u_6$ of the graph G_1 depicted in Figure 1.5.a is not Hamiltonian. Besides, it is easily seen that $\varsigma(G_1) = |V(G_1)| - 1 = |u_4u_1u_2u_5u_6|$, and hence that G_1 is not traceable. The cycle $u_1u_2u_3u_4u_8u_7u_6u_5u_1$ of the graph G_2 from Figure 1.5.b is Hamiltonian, and hence G_2 is an Hamiltonian graph. However G_2 is not Hamiltonian-connected since it does not admit any Hamiltonian $\{u_2,u_6\}$-path.

We now consider similar notions for directed graphs. A *directed path* of a directed graph D is a sequence $(v_1,v_2,...,v_k)$ of distinct vertices of D such that $\overrightarrow{v_iv_{i+1}}$ is an arc for every $i \in \{1,2,...,k-1\}$. Such a directed path is denoted $\overrightarrow{v_1v_2...v_k}$ and is sometimes called a *directed* (v_1,v_k)*-path*. The sequence $(v_1,v_2,...,v_k)$ of distinct vertices of D forms a *circuit* if $\overrightarrow{v_1v_2...v_k}$ is a directed path of D and $\overrightarrow{v_kv_1} \in A(D)$. Again, a circuit is denoted $\overrightarrow{v_1v_2...v_kv_1}$. In case an oriented graph $\overrightarrow{G}$ has no induced circuit (note that this condition does not imply that $und(\overrightarrow{G})$ has no induced cycles), we call $\overrightarrow{G}$ *acyclic*.

1.2.2.6 Modifying a graph

Given two graphs G and H, by $G+H$ we refer to the graph $(V(G) \cup V(H), E(G) \cup E(H))$. In case we just want to augment G with new vertices $v_1,v_2,...,v_k$ without having to define a new graph H as a graph consisting in k independent vertices, we write $G + \{v_1,v_2,...,v_k\}$ this operation for short. Assuming $\{u_1,v_1\},\{u_2,v_2\},...,\{u_k,v_k\}$ are pairs of independent vertices of G, we denote by $G + \{u_1v_1,u_2v_2,...,u_kv_k\}$ the graph $(V(G), E(G) \cup \{u_1v_1,u_2v_2,...,u_kv_k\})$.

We now introduce similar operations on G regarding the removal of its vertices or edges. Let $S \subseteq V(G)$ be a subset of vertices of G. By $G - S$, we refer to the subgraph

$$(V(G) \setminus S, E(G) \setminus \bigcup_{u \in S} \bigcup_{v \in N_G(u)} \{uv\})$$

obtained by removing the vertices in S and their incident edges from G. Similarly, given a subset $F \subseteq E(G)$ of edges of G, we denote $G - F$ the subgraph $(V(G), E(G) \setminus F)$ obtained by removing the edges in F from G.

Subdividing an edge uv of G means that we add a new vertex in between u and v. Formally, subdividing uv results in the graph

$$(G - \{uv\} + \{w\}) + \{uw, wv\},$$

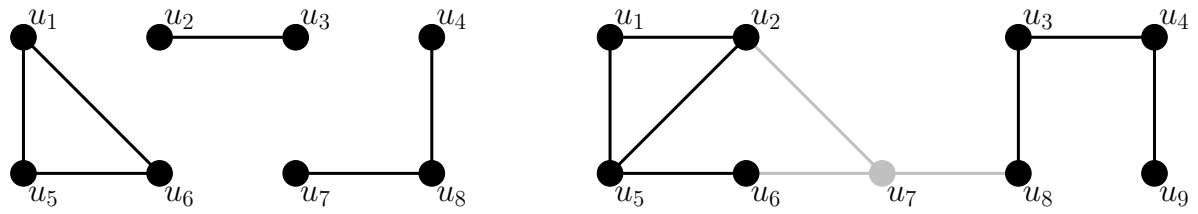
where w is a new vertex added to G. By *identifying* u and v in G, we mean that we "merge" u and v into a new vertex whose neighbourhood is made up of the neighbourhoods of both u and v. Formally set, identifying u and v can be seen as resulting in the graph

$$(G - \{v\}) + \bigcup_{w \in N_G(v)} \{uw\}.$$

By identifying more than two vertices $u_1,u_2,...,u_k$, we mean that we first identify u_1 and u_2, then identify the resulting vertex and u_3, then identify the resulting vertex and u_4, and so on.

1.2.2.7 Connectedness and connectivity

A graph G is said *connected* if there is a $\{u,v\}$-path for every two vertices u and v of G. A non-connected graph is said *disconnected*. By a *connected component* of G (or simply *component* for short), we refer to a connected subgraph of G which is maximal in terms of order. So equivalently a graph is connected if and only if it has only one component (i.e. the entire graph).



(a) A graph with three components.

(b) A graph (in black and grey) with a cut vertex (in grey only).

Figure 1.6: A disconnected graph and a connected graph.

Example 1.11. The graph G drawn in Figure 1.6.a has three components, induced by $\{u_1, u_5, u_6\}$, $\{u_2, u_3\}$, and $\{u_4, u_7, u_8\}$, respectively. This number of components implies that G is disconnected.

A *cutset* of G is a subset $S \subset V(G)$ of vertices such that $G - S$ is disconnected. A cutset with size k is sometimes called a *k-cutset*. In case $S = \{v\}$ is a 1-cutset of G, we call v a *cut vertex*. We say that G is *k-connected* for a positive integer $k \geq 1$ if G does not admit a cutset with size at most $k - 1$. Clearly 1-connectedness and classic connectedness are equivalent notions. In case G is connected, the *connectivity* of G is the positive integer $\kappa(G) \geq 1$ defined as

$$\kappa(G) = \max\{k : G \text{ is } k\text{-connected}\}.$$

Example 1.12. Removing the vertex u_7 from the graph G depicted in Figure 1.6.b results in two components, so u_7 is a cut vertex of G. This implies that G is not 2-connected, and hence that $\kappa(G) = 1$.

The connectedness of a graph G can be checked using *search algorithms*. Such algorithms consist in starting from a *root* vertex of G and then inductively traversing the edges joining discovered vertices and undiscovered vertices as long as we can, i.e. new vertices keep on being reached. Clearly G is connected if and only if all of its vertices have been reached once the algorithm is finished. Executing a search algorithm more than once (i.e. from an undiscovered root vertex remaining after the previous execution) is also a way for identifying all components of G. During the execution of a search algorithm, assuming we are currently exploring G from a vertex v, we call a *return edge* every edge joining v and an already discovered vertex.

The two common search algorithms are the *depth-first* and *breadth-first*. The main difference between them is the strategy for choosing the next vertex to exploit for pursuing the exploration. Roughly explained, the depth-first search algorithm privileges depth during the exploration of a graph, while the breadth-first search algorithm rather privileges a wide exploration. These two algorithms have running time[2] $\mathcal{O}(|V(G)| + |E(G)|)$, so checking connectedness can be done in linear time in the general case.

1.2.2.8 Distances and diameters

By the *distance* $\text{dist}(G, u, v)$ between two vertices u and v of a graph G, or simply $\text{dist}(u, v)$ when it is clear from the context, we refer to the length of a smallest $\{u, v\}$-path, that is

$$\text{dist}(u, v) = \min\{\|P\| : P \text{ is a } \{u, v\}\text{-path of } G\}.$$

In particular, note that $\text{dist}(u, v) = 1$ for every two adjacent vertices u and v of G, and that $\text{dist}(u, u) = 0$ since the trivial path has length 0. In case G is not connected and u and v belong to different components, by the definition we have $\text{dist}(u, v) = \infty$.

By $\text{diam}(G)$, we refer to the *diameter* of G defined as

[2]The notion of time complexity of an algorithm is introduced in upcoming Section 1.2.4.

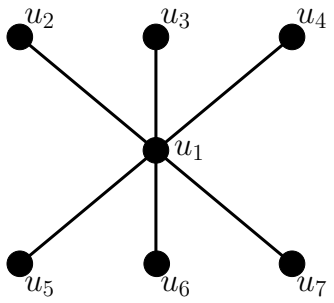


Figure 1.7: A graph with diameter 2.

$$\mathrm{diam}(G) = \max\{\mathrm{dist}(u,v) : u \text{ and } v \text{ are two vertices of } G\},$$

that is the distance between the two most distant vertices of G. The diameter is a parameter intuitively measuring how extent is a connected graph (note that a disconnected graph has infinite diameter).

Example 1.13. In the graph G from Figure 1.7, all two vertices are at distance either 1 (every pair of adjacent vertices) or 2 (e.g. u_2 and u_3). So we have $\mathrm{diam}(G) = 2$.

Two notions of distance in oriented graphs are defined, mainly because going from a vertex u to another vertex v in an oriented graph $\overrightarrow{G}$ can be easier than going from v to u as the orientation of the arcs has to be taken into account. The basic *distance* $\mathrm{dist}(\overrightarrow{G}, u, v)$ from u to v in $\overrightarrow{G}$ (or $\mathrm{dist}(u,v)$ for short) is defined as

$$\mathrm{dist}(u,v) = \min\{\|\overrightarrow{P}\| : \overrightarrow{P} \text{ is a directed } (u,v)\text{-path of } \overrightarrow{G}\}.$$

Now, if we are optimistic, then we can consider that u and v are close when one of $\mathrm{dist}(u,v)$ or $\mathrm{dist}(v,u)$ is small. This yields to the definition of the *weak distance* $\mathrm{dist}_w(u,v)$ from u to v in $\overrightarrow{G}$, which is

$$\mathrm{dist}_w(u,v) = \min\{\mathrm{dist}(u,v), \mathrm{dist}(v,u)\}.$$

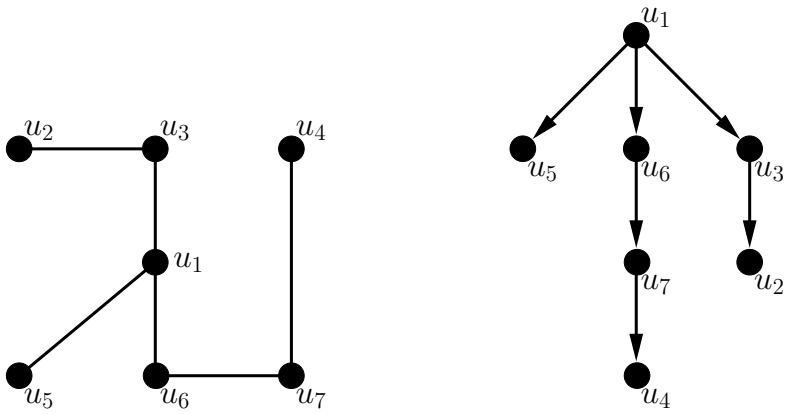
On the contrary, if we really want the notion of distance to be representative of the connexion between u and v in $\overrightarrow{G}$, then we come up with the notion of *strong distance* $\mathrm{dist}_s(u,v)$, which is defined as

$$\mathrm{dist}_s(u,v) = \max\{\mathrm{dist}(u,v), \mathrm{dist}(v,u)\}.$$

Using these two notions of oriented distance, two notions of oriented diameter then arise depending on whether we consider the weak or strong definition of distance.

1.2.2.9 Trees and rooted trees

A *tree* designates a connected graph with no cycle, while a disconnected graph whose all components are trees is called a *forest*. The vertices of a tree are rather called *nodes* in this context. A 1-node is commonly called a *leaf*, while every non-leaf node is called an *inner node*. By choosing a particular node r as the *root* of a tree T, one naturally defines an up-bottom orientation of T from its root to its leaves. The resulting *rooted tree* is denoted T_r. According to the orientation of T_r, a node u has at most one neighbour, denoted u^-, which is nearer from r than u. This node, if it exists, is referred to as the *father* of u in T_r. In contrast, the other neighbours of u, which are farther from r than u, are called the *children* of u in T_r. All nodes of T_r which are farther than u from r are called the *descendants* of u. Clearly, the root r has no father and the leaves of T_r have no children, and all non-root nodes of T_r are descendants of r. In the special case where u has only one child in T_r, we denote by u^+ this node.



(a) A tree T. (b) The rooted tree T_{u_1}.

Figure 1.8: A tree and one of its rooted trees.

Example 1.14. A tree T is depicted in Figure 1.8.a. The rooted tree T_{u_1}, obtained by rooting T at u_1, is depicted in Figure 1.8.b. The root u_1 has three children, namely u_5, u_6, and u_3. The father of u_7 is u_6, while the unique child of u_7 is u_4.

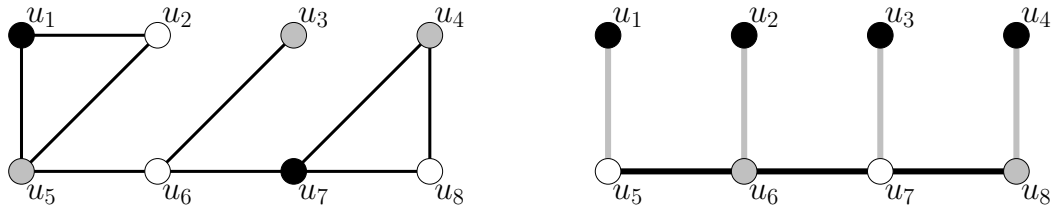
The *subtree of T_r rooted at u*, denoted $T_r[u]$, is the subgraph induced by u and the descendants of u in T_r. When dealing with such a subtree, we generally keep on using the same orientation as in T_r. In this context, the root of $T_r[u]$ is then u.

Now assuming u has $p \geq 0$ well-ordered children $v_1, v_2, ..., v_p$ in T_r, where v_i is the ith child of u for every $i \in \{1, 2, ..., p\}$, by the *ith subtree of T_r rooted at u*, denoted $T_r[u, i]$, we refer to the subtree of T_r induced by u, v_i, and the descendants of v_i in T_r. Note that, in such a subtree, the root u has only one child, which is v_i (so u^+ is defined, and $u^+ = v_i$). Besides, by identifying the roots of $T_r[u, 1], T_r[u, 2], ..., T_r[u, p]$ we obtain $T_r[u]$.

1.2.2.10 Weighting or colouring the elements of a graph

Let G be a graph. A *weighting* of some elements of G (e.g. its vertices and/or edges, etc.) is a mapping assigning a value from a given set S of values, called *weights*, to each of these elements. Most of the time, this set S is considered as being part of $\mathbb{R}$, so, unless specified, it is understood throughout that a weighting is an assignment of real values. In case the values assigned by the weighting are not meaningful in the sense that changing some weights do not impact on the problem we are interested in, we rather speak of a *colouring* of some elements of G, while a weight is rather called a *colour* in this context. For a colouring which is not a weighting, it is commonly considered that S is of the form $\{1, 2, ..., k\}$ for the sake of simplicity. So a colouring is also a weighting, but the converse does not necessarily hold.

The upcoming notions related to weightings naturally transpose to colourings. Three main kinds of graph weightings are investigated throughout this thesis. Namely, a weighting of G is called a *vertex-*, *edge-*, or *total-weighting* when this weighting concerns the vertices, edges, or both the vertices and edges of G, respectively. A maximum subset of elements of G being assigned a same weight by a weighting forms a *weight class*. Given a subset $S \subset \mathbb{R}$ of weights, an *S-weighting* w of G is a weighting assigning a value among S to each target element of G. In the special case where $S = \{1, 2, ..., k\}$, we call w a *k-weighting*. Regarding a total-weighting, it might be the case that the sets of values assigned to the vertices and to the edges are different, that is we assign e.g. a weight among $\{1, 2, ..., k\}$ to the vertices of G and among $\{1, 2, ..., \ell\}$ to the edges of G, with possibly $k \neq \ell$. We call such a weighting a *(k, ℓ)-total-weighting*. We refer to a (k, k)-total-weighting as a *k-total-weighting* for short.



(a) A 3-vertex-colouring of a graph. (b) A $(3,2)$-total-colouring of a graph.

Figure 1.9: Colourings of two graphs (each colour corresponds to a value among $\{1,2,3\}$).

Example 1.15. Figure 1.9.a depicts a 3-vertex-colouring of a graph, while Figure 1.9.b illustrates a $(3,2)$-total-colouring of a graph, where e.g. colours black, grey and white represent colours 1, 2, and 3, respectively, of these colourings.

Well-known particular kinds of graph colourings are *proper colourings*. A colouring c of G is said *proper* if every colour class of c is independent. Applying this concept to the introduced above notions, we get that a vertex-colouring is proper if every two adjacent vertices have distinct colours, an edge-colouring is proper if every two adjacent edges receive distinct colours, and a total-colouring is proper if no two adjacent vertices, no two adjacent edges, and no vertex and one of its incident edges are assigned the same colour.

Example 1.16. The 3-vertex-colouring represented in Figure 1.9.a is proper since no two adjacent vertices have the same colour. The $(3,2)$-total-colouring of the graph G depicted in Figure 1.9.b is not proper since e.g. the vertex u_6 and the edge u_2u_6 have the same colour, and similarly for the two edges u_5u_6 and u_6u_7. Restricted to its vertices, the total-colouring of G is however a proper 3-vertex-colouring.

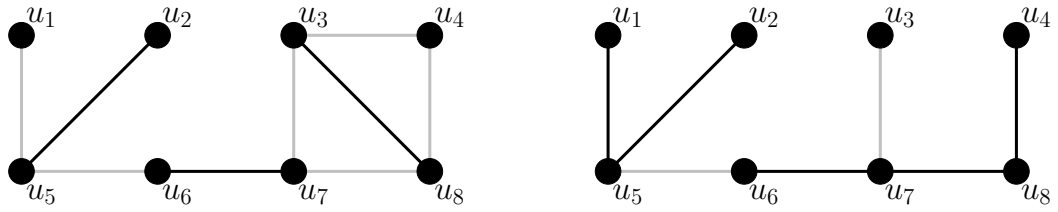
The least number of colours used by a proper vertex-colouring of G is called the *chromatic number* of G, denoted $\chi(G)$. Similarly are defined the *chromatic index* of G, denoted $\chi'(G)$, regarding proper edge-colourings of G, and the *total chromatic number* of G, denoted $\chi''(G)$, regarding proper ℓ-total-colourings of G. We say that G is *k-vertex-colourable* if G admits a proper k-vertex-colouring. The properties of being *k-edge-colourable* and *k-total-colourable* are defined analogously regarding proper edge-colouring and proper ℓ-total-colouring, respectively. Colouring parameters are generally related to other graph invariants, e.g. $\omega(G)$ for $\chi(G)$, or $\Delta(G)$ for $\chi'(G)$. One classic result regarding the chromatic number of graphs is the following upper bound on χ exhibited by Brooks [40].

Theorem 1.17 ([40])**.** *For every graph G, we have $\chi(G) \leq \Delta(G)+1$. Besides, we have $\chi(G) = \Delta(G)+1$ if and only if G is an odd length cycle or a complete graph*[3]*.*

In the classic colouring notions above, it is assumed that all coloured elements are assigned a colour from a same set S. Assuming now that each coloured element x is assigned a list $L(x)$ of possible colours, a *list colouring* of G is obtained by assigning a colour from $L(x)$ to each element x of G to colour. By a *k-list colouring*, it should be understood that every list $L(x)$ has size k.

The notion of list colouring can of course be combined with the notions of vertex-, edge- and total-colouring, as well as specific notions like the one of proper colouring. In particular, we say that G is *k-vertex-choosable* if we can obtain a proper k-list vertex-colouring of G no matter what are the lists of size k assigned to its vertices. Similarly are defined the notions of *k-edge-choosability* and *k-total-choosability*. The parameters χ, χ' and χ'' in turn extend to list colouring as well, though we rather speak of vertex-, edge- and total-*choice number*, respectively,

[3]Classes of graphs are introduced in upcoming Section 1.2.2.13.



(a) A matching of a graph. (b) A path cover of a graph.

Figure 1.10: A matching and a path cover (in black only) of two graphs (in black and grey).

in this context, which correspond to the least k such that a graph is k-vertex-, k-edge-, or k-total-choosable, respectively. These parameters are denoted ch, ch' and ch'', respectively.

We end up with some terminology associated with edge-colourings. Let c be a k-edge-colouring of G, and a be a colour used by c. An edge of G is said *a-coloured* when it is assigned colour a by c. The *a-subgraph* of G refers to the subgraph of G induced by all of its a-coloured edges. Now if v is a vertex of G, by the *a-degree* of v we refer to the degree of v in the a-subgraph. This parameter is sometimes denoted $d_{c,a}(v)$. Assuming c is an arc-colouring of a directed graph, the notions of *a-indegree* and *a-outdegree* of v by c are defined analogously. These parameters are denoted $d^-_{c,a}(v)$ and $d^+_{c,a}(v)$, respectively.

1.2.2.11 Covering the elements of a graph

A *matching* of a graph G is a subset $M \subseteq E(G)$ of edges such that no two edges of M share an end. A matching with size $\left\lfloor \frac{|V(G)|}{2} \right\rfloor$ is called *perfect* in case $|V(G)|$ is even, or *quasi-perfect* otherwise.

A *path cover* of G is a collection of vertex-disjoint paths covering all vertices of G, or, equivalently, a partition $V_1 \cup V_2 \cup ... \cup V_k$ of $V(G)$ such that $G[V_i]$ is traceable for every $i \in \{1, 2, ..., k\}$. The *path cover number* of G is defined as

$$\mu(G) = \min\{k : V(G) \text{ admits a partition } V_1 \cup V_2 \cup ... \cup V_k \text{ where each } G[V_i] \text{ is traceable}\},$$

i.e. the minimum size of a path cover of G.

Example 1.18. The matching M of the graph G_1 depicted in Figure 1.10.a is not perfect since u_1 and u_4 are not covered by edges of M. The path cover of the graph G_2 depicted in Figure 1.10.b is one of the smallest path covers of G_2 (it is easily seen that G_2 has no path covers made up of only two parts). Therefore, we have $\mu(G) = 3$.

The *arboricity* of G, denoted $a(G)$, is the smallest number of forests into which G can be edge-partitioned, that is

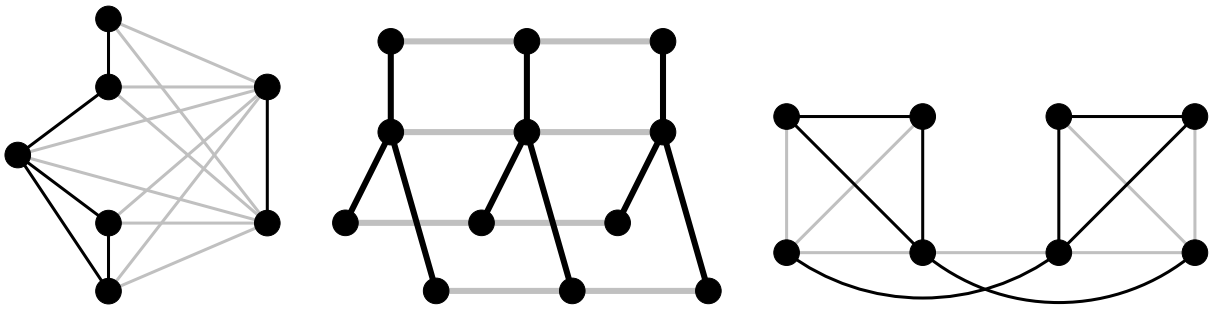
$$a(G) = \min\{k : E(G) \text{ admits a partition } E_1 \cup E_2 \cup ... \cup E_k \text{ where each } G[V_i] \text{ induces a forest}\}.$$

A *feedback vertex set* of G is a subset $S \subset V(G)$ of vertices such that $G - S$ is a forest, i.e. a set whose removal from G removes all cycles of G.

1.2.2.12 Graph operations

Let G and H be two graphs. The *disjoint union* of G and H is the graph $G + H$. The *complete join* of G and H, denoted $G \times H$, is the graph

$$(V(G) \cup V(H), E(G) \cup E(H) \cup (V(G) \times V(H)))$$



(a) The complete join (resulting edges are in grey) of two graphs (in black only).

(b) The Cartesian product of two graphs (edges in each shade of grey correspond to edges of one operand).

(c) The 2nd power (in grey and black) of a graph (in grey only).

Figure 1.11: A complete join and Cartesian product of two graphs, and a power of a graph.

obtained by first considering $G + H$ and then adding all possible edges between vertices which originally belonged to G and vertices which originally belonged to H.

The *Cartesian product* of G and H, denoted $G \square H$, is the graph with vertex set $V(G) \times V(H)$, and whose every two vertices (u_1, v_1) and (u_2, v_2) are linked by an edge if and only if either

- $u_1 = u_2$ and $v_1v_2 \in E(H)$, or
- $v_1 = v_2$ and $u_1u_2 \in E(G)$.

Intuitively, $G \square H$ is the graph obtained by considering $|V(H)|$ copies of G and connecting them accordingly to the structure of H (or conversely).

Let $k \geq 1$ be a positive integer. The *kth power* of G, commonly denoted G^k, is the graph with the same vertex set as G, and in which two vertices of G^k are linked by an edge if they are at distance at most k in G. Clearly G^1 is nothing but G.

Example 1.19. The complete join of two graphs is depicted in Figure 1.11.a, the Cartesian product of two graphs is illustrated in Figure 1.11.b, and the 2nd power of a graph is represented in Figure 1.11.c.

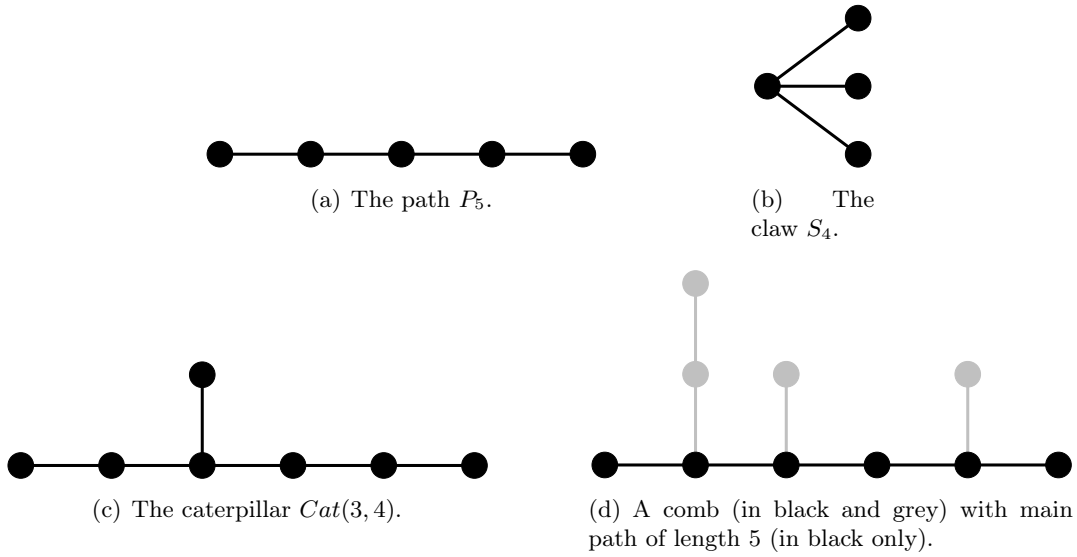
1.2.2.13 Classes of graphs

The *path* with order n is denoted P_n. The *star* of order n, denoted S_n, is the tree obtained by considering one central node r, called the *root* of S_n, and joining it to $n - 1$ independent nodes. The star S_4 is sometimes called the *claw*.

A *multipode* P is a tree made up of exactly one node r with degree at least 3. Equivalently P is obtained after several subdividings of the edges of $S_{d(r)+1}$, the star with order $d(r)+1$. We call r the *root* of P, while the $d(r)$ maximum node-disjoint paths of $P - \{r\}$ are called the *arms* of P. Assuming P has $d(r)$ arms with orders $a_1, a_2, ..., a_{d(r)} \geq 1$, respectively, we sometimes write $P_{d(r)}(a_1, a_2, ..., a_{d(r)})$ to directly refer to P, and call P a $d(r)$*-pode* to emphasize the number of its arms. We sometimes refer to a 3-pode as a *tripode*. Note that any multipode $P_k(a_1, a_2, ..., a_k)$ has order $1 + \sum_{i=1}^{k} a_i$.

A *caterpillar* refers to a tree made up of one main path to which every node either belongs or is at distance 1. Note that, in the very special case where only one node u does not belong to the main path, a caterpillar can be viewed as a tripode $P_3(1, a, b)$, where u is joined to v_{a+1} assuming the main path is $v_1v_2...v_{a+b+1}$. Such a caterpillar is sometimes denoted $Cat(a+1, b+1)$ for convenience (the order of such a caterpillar is $a + b + 2$).

A *comb* is a tree with maximum degree 3 whose all nodes with degree 3 are located along a same path. Every tripode is actually a comb with only one degree-3 node.



(a) The path P_5.

(b) The claw S_4.

(c) The caterpillar $Cat(3, 4)$.

(d) A comb (in black and grey) with main path of length 5 (in black only).

Figure 1.12: Examples of particular trees.

Example 1.20. A path, the claw, a caterpillar, and a comb are illustrated in Figure 1.12.a, .b, .c, and .d, respectively.

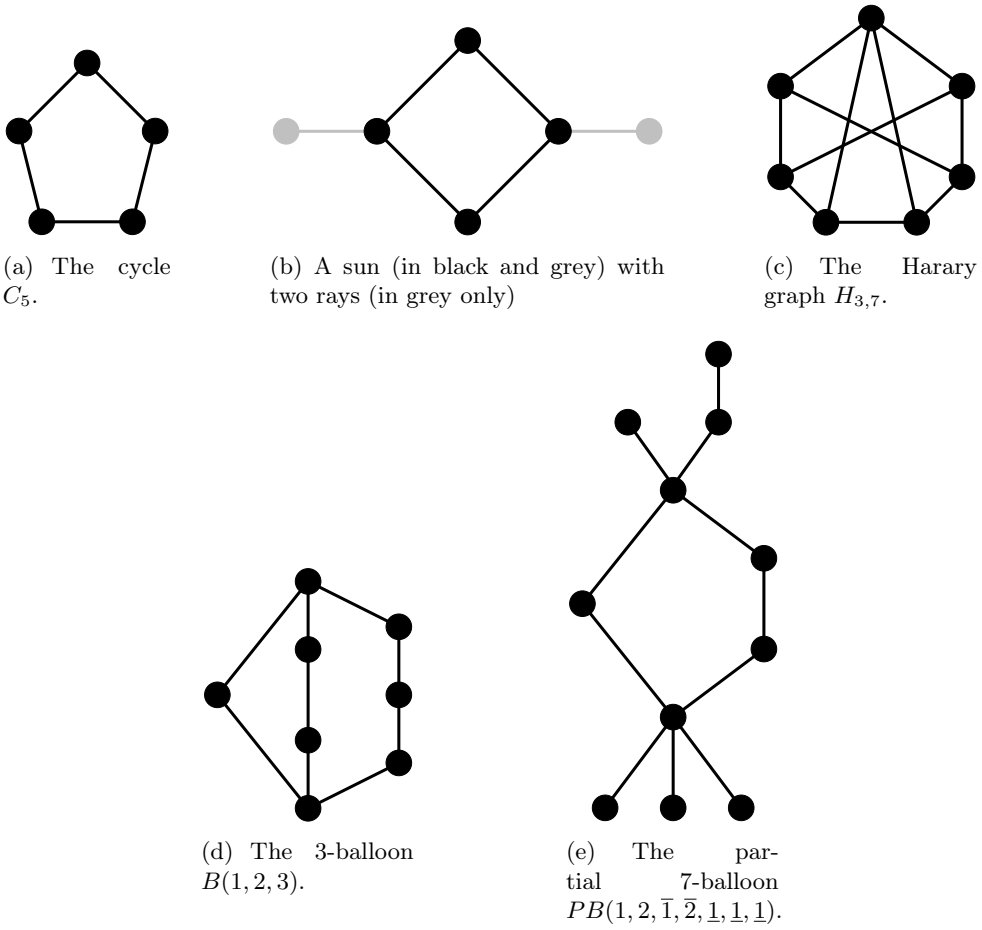
The *cycle* with order (and length) n is denoted C_n. A *sun* is a 1-connected unicyclic graph made up of one main cycle to which degree-1 vertices, called *rays*, are distinctly attached, i.e. in such a way that if u and v are two rays then $N(u) \neq N(v)$ (no two rays are attached to the same vertex of the main cycle).

Let $k \geq 1$ and $n \geq k$ be two integers. The k-connected *Harary graph* on n vertices, denoted by $H_{k,n}$, is the graph with vertex set $\{v_0, v_1, ..., v_{n-1}\}$ and the following edges:

- if $k = 2r$ is even, then two vertices v_i and v_j are linked if and only if $i - r \leq j \leq i + r$;
- if $k = 2r + 1$ is odd and n is even, then $H_{k,n}$ is obtained by joining v_i and $v_{i+\frac{n}{2}}$ in $H_{2r,n}$ for every $i \in \{0, 1, ..., \frac{n}{2} - 1\}$;
- if $k = 2r + 1$ and n are odd, then $H_{k,n}$ is obtained from $H_{2r,n}$ by first linking v_0 to both $v_{\lfloor \frac{n}{2} \rfloor}$ and $v_{\lceil \frac{n}{2} \rceil}$, and then each vertex v_i to $v_{i+\lceil \frac{n}{2} \rceil}$ for every $i \in \{1, 2, ..., \lfloor \frac{n}{2} \rfloor - 1\}$;

where the subscripts are taken modulo n. Note that in every graph $H_{k,k}$, every two vertices are joined by an edge. When k is odd, the neighbours of a vertex v_i of $H_{k,n}$ which are at distance strictly more than k from v_i in the underlying C_n are called the *antipodal neighbours* of v_i (there are at most two of them). In particular, the vertex v_i has two antipodal neighbours if and only if $i = 0$, and k and n are both odd. A *diagonal edge* of $H_{k,n}$ is an edge linking two antipodal neighbours of $H_{k,n}$. Harary graphs are known as a family of graphs with arbitrary order and connectivity, and the least number of edges regarding these two parameters.

A *balloon* is a 2-connected graph B obtained as follows. Consider a multigraph with order 2 whose two vertices r_1 and r_2 are joined by $k \geq 2$ parallel edges. Now subdivide each of these edges an arbitrarily number of times (but at least one) to get B. Equivalently B can be obtained by considering paths $P_{b_1}, P_{b_2}, ..., P_{b_k}$, with $3 \leq b_1 \leq b_2 \leq ... \leq b_k$, whose endvertices are denoted u_1 and v_1, u_2 and v_2, ..., and u_k and v_k, respectively, and identifying all the u_i's, and identifying all the v_i's. We call r_1 and r_2 the *roots* of B. By removing the vertices r_1 and r_2 from B, we get a forest of paths, called *branches* of B, with order $b_1 - 2, b_2 - 2, ..., b_k - 2$, respectively. An *even branch* designates a branch with even order, while an *odd branch* designates a branch with odd



(a) The cycle C_5.

(b) A sun (in black and grey) with two rays (in grey only)

(c) The Harary graph $H_{3,7}$.

(d) The 3-balloon $B(1,2,3)$.

(e) The partial 7-balloon $PB(1,2,\overline{1},\overline{2},\underline{1},\underline{1},\underline{1})$.

Figure 1.13: Examples of graphs with cyclic structure.

order. We sometimes call B a *k-balloon* and write $B = B(b_1 - 2, b_2 - 2, ..., b_k - 2)$ to clarify its number of branches and their lengths.

A *partial balloon* refers to a connected graph obtained by removing some edges incident with the roots of a balloon. More precisely, assuming $B = B(b_1, b_2, ..., b_{x+y+z})$ is a k-balloon, the *partial $(x+y+z)$-balloon* $B' = PB(b_1, b_2, ..., b_x, \overline{b_{x+1}}, \overline{b_{x+2}}, ..., \overline{b_{x+y}}, \underline{b_{x+y+1}}, \underline{b_{x+y+2}}, ..., \underline{b_{x+y+z}})$ is obtained by removing from B the edges incident to r_2 of the $(x+1)$th, $(x+2)$th, ..., $(x+y)$th branches of B, as well as the edges incident to r_1 of the $(x+y+1)$th, $(x+y+2)$th, ..., $(x+y+z)$th branches of B. The last $y+z$ branches of B', which are attached to only one of the two roots of B', are said *hanging*.

Example 1.21. A cycle, a sun, a Harary graph, a balloon, and a partial balloon are illustrated in Figure 1.13.a, .b, .c, .d, and .e, respectively.

A *(k,ℓ)-compound graph* is a graph obtained as follows. Let $G_1, G_2, ..., G_\ell$ be ℓ graphs such that for each G_i, with $i \in \{1, 2, ..., \ell\}$, there are k vertices $u_1^i, u_2^i, ..., u_k^i$ for which

$$G_1[\{u_1^1, u_2^1, ..., u_k^1\}] \simeq G_2[\{u_1^2, u_2^2, ..., u_k^2\}] \simeq ... \simeq G_\ell[\{u_1^\ell, u_2^\ell, ..., u_k^\ell\}].$$

Then, as $C_{k,\ell}(G_1, G_2, ..., G_\ell)$, we refer to the (k,ℓ)-compound graph obtained by considering the disjoint union $G_1 + G_2 + ... + G_\ell$ and identifying the vertices u_i^1, u_i^2, ..., u_i^ℓ for every $i \in \{1, 2, ..., k\}$. In other words, the graph $C_{k,\ell}(G_1, G_2, ..., G_\ell)$ is made up of ℓ components "glued" together along k of their vertices. The k vertices $u_1, u_2, ..., u_k$ resulting from the identifications are called the *roots* of $C_{k,\ell}(G_1, G_2, ..., G_\ell)$. Considering the components $G_1, G_2, ..., G_\ell$ independently, the *projection* of every root vertex u_i to the jth component G_j is denoted u_i^j.

Observation 1.22. *Every k-pode $P_k(a_1, a_2, ..., a_k)$ is a $(1, k)$-compound graph $C_{1,k}(P_{a_1+1}, P_{a_2+1}, ..., P_{a_k+1})$. Every k-balloon $B(b_1, b_2, ..., b_k)$ is a $(2, k)$-compound graph $C_{2,k}(P_{b_1+2}, P_{b_2+2}, ..., P_{b_k+2})$.*

A graph G is said to be *k-partite* (or simply *multipartite*) for a positive integer $k \geq 2$ if $V(G)$ admits a partition $V_1 \cup V_2 \cup ... \cup V_k$ such that V_i is a non-empty independent set for every $i \in \{1, 2, ..., k\}$. A 2-partite graph is rather called a *bipartite graph*. A bipartite graph with vertex set $A \cup B$ is said *balanced* if $|A| = |B|$, or *unbalanced* otherwise.

Observation 1.23. *Every k-colourable graph is k-partite.*

The *complete graph* on n vertices, denoted K_n, is the graph of order n in which every two vertices are adjacent. An oriented graph $\overrightarrow{T}$ whose underlying graph is complete is called a *tournament*. We say that $\overrightarrow{T}$ is *transitive* if we have $\overrightarrow{uw} \in A(\overrightarrow{T})$ for every two arcs $\overrightarrow{uv}$ and $\overrightarrow{vw}$ of $\overrightarrow{T}$.

For given $k \geq 2$ positive integers $p_1, p_2, ..., p_k$ with $1 \leq p_1 \leq p_2 \leq ... \leq p_k$, the *complete k-partite graph* $M_k(p_1, p_2, ..., p_k)$ is the k-partite graph whose vertex set admits a partition $V_1 \cup V_2 \cup ... \cup V_k$ such that V_i is an independent set on p_i vertices for every $i \in \{1, 2, ..., k\}$ and which has the maximum number of edges.

Observation 1.24. *Every star S_n is isomorphic to the complete bipartite graph $M_2(1, n-1)$.*

A *split graph* G is a graph whose vertex set $V(G)$ admits a partition $I \cup C$ such that I is an independent set and $G[C]$ is a complete graph. A notable split graph is the *net*, which is the graph obtained by considering $|I| = \{u_1, u_2, u_3\}$ and $|C| = \{v_1, v_2, v_3\}$, and then adding the edges u_1v_1, u_2v_2 and u_3v_3. Equivalently, the net can be obtained by replacing the root of the claw by a triangle.

A *planar graph* is a graph which can be drawn on the plane in such a way that no two edges cross. Such a drawing of a graph forms a *plane graph*. We say that a plane graph is *triangulated* if adding any edge to it results in a graph which is not planar.

Example 1.25. A $(3, 3)$-compound graph, a complete graph, a complete bipartite graph, a split graph, the net, and a plane triangulation are illustrated in Figure 1.14.a, .b, .c, .d, .e, and .f, respectively.

Cographs are a family of graphs described by the following rules:

Base case: K_1 is a cograph,

Union operation: if G and H are two cographs, then $G + H$ is a cograph,

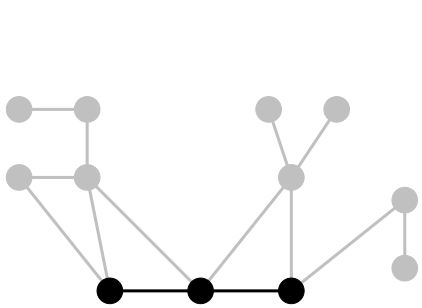
Join operation: if G and H are two cographs, then $G \times H$ is a cograph.

A cograph can be represented as a term involving the three symbols $\bullet$, $+$ and $\times$, where $\bullet$ intends to represent K_1. Equivalently, every cograph G admits a *cotree representation* T_r, i.e. can be expressed as a rooted tree where every inner node has exactly two children, whose leaves are the vertices of G, and where each inner node u symbolizes either the union or the join of the two cographs represented by the two subtrees rooted at the children of u.

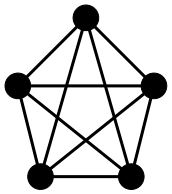
Series-parallel graphs are defined as follows. Each series-parallel graph G has two specific vertices $s(G)$ and $t(G)$ called *terminals*. Then:

Base case: P_2 is a series-parallel graph whose terminals are its two endvertices,

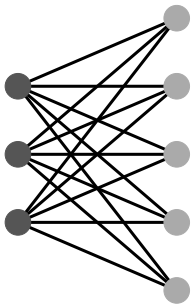
Series composition: if G and H are two series-parallel graphs, then the graph obtained by identifying $t(G)$ and $s(H)$ is a series-parallel graph with terminals $s(G)$ and $t(H)$,



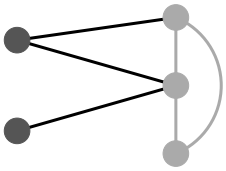
(a) A compound graph (in black and grey) with three roots (in black only) and three components (in grey only).



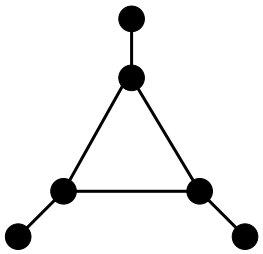
(b) The complete graph K_5.



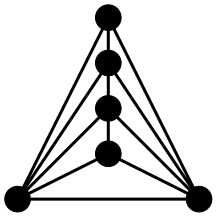
(c) The complete bipartite graph $M_2(3,5)$ with parts of size 3 and 5 (in two shades of grey).



(d) A split graph with independent part of size 2 and complete part of size 3 (in two shades of grey).



(e) The net.



(f) A plane triangulation.

Figure 1.14: Examples of locally dense graphs.

Parallel composition: if G and H are two series-parallel graphs, then the graph obtained by identifying $s(G)$ and $s(H)$ and identifying $t(G)$ and $t(H)$ is a series-parallel graph with terminals $s(G) = s(H)$ and $t(G) = t(H)$.

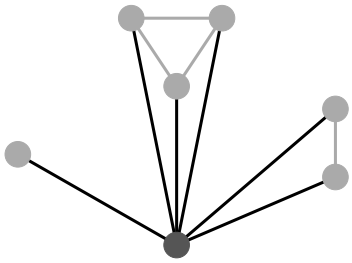
Observation 1.26. *Every path P_n is a series-parallel graph obtained from series compositions of $n-1$ copies of P_2. Every k-balloon is a series-parallel graph obtained from parallel compositions of k paths.*

One point for introducing cographs and series-parallel graphs is that these graphs are known to be fair regarding some notoriously hard problems, in the sense that some such problems can be handled easily when restricted to cographs or series-parallel graphs.

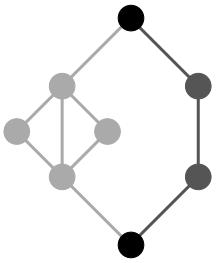
Example 1.27. A cograph and a series-parallel graph are illustrated in Figure 1.15.a, and .b, respectively.

We now introduce some common classes of graphs which may be obtained using the Cartesian product operation. For any two positive integers $a, b \geq 1$, the *grid* $G_{a,b}$ is $P_a \,\square\, P_b$. A grid $G_{2,a}$ is sometimes referred to as a *ladder*. One interesting class of Cartesian products of graphs is the one of *hypercubes*. Hypercubes are defined inductively. The smallest hypercube is $Q_1 \simeq K_2$. Then, for every $n \geq 2$ such that Q_{n-1} has been defined, the hypercube Q_n is $Q_{n-1} \,\square\, K_2$. Regarding a hypercube Q_n, we call n the *dimension* of Q_n.

Let $P_k = v_1v_2...v_k$ be the path of order k, and $i \in \{1, 2, ..., k\}$ be any index. In case we are



(a) A join of two cographs (in two shades of grey).



(b) A parallel composition of two series-parallel graphs (in two shades of grey).

Figure 1.15: Examples of inductively defined graphs.

dealing with a Cartesian product involving P_k, i.e. of the form $G \,\square\, P_k$, we refer to the subgraph

$$G^i = (G \,\square\, P_k)[\bigcup_{u \in V(G)} (u, v_i)]$$

as the *ith layer* of G in $G \,\square\, P_k$. For every vertex $u \in V(G)$, we refer to the vertex $u^i = (u, v_i)$ of $G \,\square\, P_k$ as the *ith layer* of u in $G \,\square\, P_k$. Similarly, the *ith layer* in $G \,\square\, P_k$ of a vertex subset $S \subseteq V(G)$ is the set S^i made up of the ith layers of all vertices in S.

1.2.3 Probabilistic tools

The *Erdős-Rényi model* refers to sets of random graphs obtained as follows [57]. Let $n \geq 1$ be a positive integer, and $p \in [0, 1]$ be a probability. The family $G(n, p)$ of graphs includes graphs with order n in which every two vertices are joined by an edge with probability p.

We now introduce two classical probabilistic tools, namely the *Lovász Local Lemma* and the *Chernoff Bound*. Roughly set, the Local Lemma states that if a large number of (generally) bad events are sufficiently independent and each has probability less than 1 to occur, then there is a positive probability that none of these bad events occurs. This tool has been widely used in the field of probability for proving the existence of mathematical objects. Refer to e.g. the reference books of Alon and Spencer [7] and Molloy and Reed [94] for more details on this topic.

Theorem 1.28 (Local Lemma - Symmetric case)**.** *Let $A_1, A_2, ..., A_n$ be events in an arbitrary probability space. Suppose that each event A_i is mutually independent of a set of all the other events A_j but at most D, and that $\Pr(A_i) \leq p$ for every $i \in \{1, 2, ..., n\}$. If*

$$e \cdot p \cdot (D + 1) \leq 1,$$

then $\Pr(\bigcap_{i=1}^{n} \overline{A_i}) > 0$.

Theorem 1.29 (Chernoff Bound)**.** *Let* $\mathrm{BIN}(n, p)$ *be the sum of n independent variables, each equal to 1 with probability p and 0 otherwise. Then, for any t with $0 \leq t \leq np$, we have*

$$\Pr(|\mathrm{BIN}(n, p) - np| > t) < 2e^{-\frac{t^2}{3np}}.$$

1.2.4 Computational complexity theory

More precise details on all explanations given throughout this section can be found e.g. in the book of Garey and Johnson [62]. The goal of computational complexity theory is basically to assess how hard it is to deal with a mathematical problem, hence defining classes of equivalently hard problems.

Decision problems

A *problem* Π is generally defined as a set of *inputs*, i.e. objects of possibly different natures, and a *question* regarding these inputs. Clearly two different problems Π and Π' may admit different kinds of answers to their question (e.g. integers, strings, lists, etc.), yielding a first classification of mathematical problems. We herein focus on the family of *decision problems*, which are those problems whose question is closed, i.e. can only be answered *yes* or *no*. Since decision problems are the only kind of problems considered within this manuscript, the term "decision" is voluntarily omitted throughout.

An application I of Π to specific values $A_1, A_2, ..., A_k$ of its inputs, where, again, the A_i's may be objects of quite different natures, is called an *instance* of Π, and is sometimes denoted $<A_1, A_2, ..., A_k>$. We say that I is *positive* if its answer regarding the question of Π (and the inputs of I) is *yes*, or *negative* otherwise. Since Π is assumed to be a decision problem, by *solving* I we mean that we determine correctly whether I is positive or negative. The correct answer to I, i.e. either *yes* or *no*, is called the *solution*. The *size* of I, denoted $|I|$, is generally defined as the number of bits necessary to describe the inputs of I.

Example 1.30. A typical decision problem is Prime Number, which asks: given an integer n, is n a prime number? The instance $<5>$ of this problem is positive, while the instance $<10>$ is negative.

Solving algorithms, and time and space complexities

The usual way for dealing with Π is to design a *solving algorithm* A for Π, that is a set of successive atomic tasks (or *instructions*) which eventually yield a solution when applied to any input instance I of Π. The efficiency of A can be evaluated regarding several criteria, though the more usual ones are the amounts of *time*, counted as a number of instructions, or *space*, counted as the number of working memory cells (or bits), A needs before eventually answering. These amounts are called the *time complexity* and the *space complexity*, respectively, of A.

The time and space complexities of A are related to the size of the input instance I it is asked to solve (in general A must indeed at least "read" all input parameters making up I before solving it). Therefore, each complexity of A is rather expressed as a function of the size of an instance of Π. To this end, we make use of the $\mathcal{O}$ notation. Formally, let $f(n)$ and $g(n)$ be two functions defined over a subset of the real numbers. We say that $f(n)$ *is* $\mathcal{O}(g(n))$ if there is a positive constant $c > 0$ and a real number n_0 for which $f(n) < c \cdot g(n)$ for every $n > n_0$. Roughly speaking, saying that $f(n)$ is $\mathcal{O}(g(n))$ means that $f(n)$ asymptotically behaves as $g(n)$, i.e. for large enough values of n. Since the $\mathcal{O}$ notation describes the asymptotic behaviour of $f(n)$, we generally do not make the constants nor the low-order terms appear since they are less significant than the highest order term of $f(n)$ of the growth of $f(n)$ (but they are actually caught by the "hidden" constant c). This simplification permits to lighten the notation by only keeping the term of interest to characterize the asymptotic growth of a function. We sometimes also make use of the o notation, where we say that $f(n)$ *is* $o(g(n))$ if for *every* positive constant $c > 0$, there exists a real number n_0 for which $f(n) < c \cdot g(n)$ for every $n > n_0$. So the o notation is stronger than the $\mathcal{O}$ one in the sense that every function which is $o(g(n))$ is also $\mathcal{O}(g(n))$, but the contrary does not necessarily hold.

Example 1.31. A naive solving algorithm for Prime Number is the *trivial division* method, which consists, regarding an instance $<n>$, in checking whether an integer $n' \in \{2, 3, ..., \sqrt{n}\}$ is a divisor of n. Assuming a division is performed in time $\mathcal{O}(1)$, the trivial division method is achieved in time $\mathcal{O}(\sqrt{n})$.

Not only we want to find a solving algorithm for Π, but we also want to design a solving algorithm with the best possible time and/or space complexity regarding an input size n. From this point of view, it is commonly accepted that a time or space complexity of a solving algorithm

is "good" if it is $\mathcal{O}(n^{\mathcal{O}(1)})$ (polynomial) and "bad" if it is $\mathcal{O}(2^{\mathcal{O}(n)})$ (exponential). By saying that an algorithm for Π is *polynomial-time* or *exponential-time*, it should be understood that its time complexity is $\mathcal{O}(|I|^{\mathcal{O}(1)})$ or $\mathcal{O}(2^{\mathcal{O}(|I|)})$, respectively, when applied on an instance I. Notions of *polynomial-space* and *exponential-space* algorithms for Π are defined in an analogous way. It is important to keep in mind that the above notions of "good" and "bad" algorithms are defined in an asymptotic context. Notably, a polynomial-time solving algorithm A for Π can be practically worst than an exponential-time solving algorithm A' for Π, notably when the constant behind the time-complexity of A is awful, when dealing with "small" instances of Π.

Common time-related complexity classes

Unless specified, the upcoming sections are dedicated to time complexity only. Based on how efficiently we can solve some problems, i.e. what is the time complexity of the "fastest" solving algorithms we can design for these, we obtain a classification of equivalent problems, where two problems are considered equivalent if they can be solved via algorithms with similar time complexities. One has to keep in mind that the hierarchy of complexity classes presented herein is not unique and that other problem classifications could be introduced based on different criteria. Besides, we only introduce those complexity classes which are relevant regarding the results of this manuscript.

A very first classification is based on whether a problem can be solved efficiently. On the one hand, the P complexity class gathers all problems which can be solved in polynomial time. On the other hand, all problems which can be solved in exponential time belong to the $\mathsf{EXPTIME}$ complexity class. Clearly $\mathsf{P} \subseteq \mathsf{EXPTIME}$.

Example 1.32. Since the trivial division algorithm runs in polynomial time on every instance of Prime Number, Prime Number is in P.

There is of course a big gap between the P and $\mathsf{EXPTIME}$ classes, and it would be quite awkward to consider that a problem which we cannot solve in polynomial time is nothing more refined than an $\mathsf{EXPTIME}$ problem. So several additional complexity classes in between P and $\mathsf{EXPTIME}$ have been introduced to characterize the time complexity of every problem $\Pi \in \mathsf{EXPTIME} \setminus \mathsf{P}$. Among these additional classes, NP and $\mathsf{co\text{-}NP}$ are of great interest for us. First, the NP class contains those problems for which we can check in polynomial-time whether an instance is positive. Basically Π is in NP if we can design a polynomial-time *checking algorithm* for the *yes*, i.e. an algorithm which can, provided an instance $I \in \Pi$ and additional inputs related to I, determine whether I is positive regarding these additional inputs. Conversely, the $\mathsf{co\text{-}NP}$ class includes those problems for which we can design a polynomial-time checking algorithm for the *no*. The additional inputs based on which a checking algorithm decides whether an instance is positive or negative is called a *certificate*. A checking algorithm aiming at deciding whether an instance is positive (resp. negative) is sometimes called *yes-checking* (resp. *no-checking*).

Example 1.33. A *yes*-checking algorithm for Prime Number would basically, regarding an instance $<n>$, take two integers p and q as input parameters (certificate), and check whether $n = pq$.

Clearly we have $\mathsf{P} \subseteq \mathsf{NP} \cap \mathsf{co\text{-}NP}$ since a solving algorithm can be seen as a checking algorithm requiring no certificate to answer. It is however still far from being clear whether $\mathsf{NP} \subseteq \mathsf{P}$. This yields to surely the major unsolved computer science question.

Question 1.34. *Do we have $\mathsf{P} = \mathsf{NP}$?*

Question 1.34, if true, would have drastic consequences on several fields of nowadays life, including mathematics, cryptology, computer science, economics, etc.. Unless $\mathsf{P} = \mathsf{NP}$, a problem $\Pi \in \mathsf{NP} \setminus \mathsf{P}$ is basically a problem which we do not know how to solve in polynomial time, but, given an instance $I \in \Pi$, we can "guess" a certificate (among an exponential set of candidates) and check whether I is positive regarding this guessed certificate.

Hardness and completeness

So that we introduce the next complexity notion, we first need to introduce the notion of *reduction*. A reduction from a problem Π to another problem Π' is a function $f : \Pi \to \Pi'$ such that

$$\text{an instance } I \in \Pi \text{ is positive} \Leftrightarrow f(I) \in \Pi' \text{ is positive.}$$

In case such a reduction from Π to Π' exists, we say that Π is *reducible* to Π', and write $\Pi \leq \Pi'$. Besides, we sometimes call $f(I)$ the *reduced instance* of I (resulting from f). Assuming that the reduction f can be described as a polynomial-time algorithm, the problem Π' can be intuitively regarded as "harder" than Π (and thus using the symbol $\leq$ makes sense). Indeed, if we had a polynomial-time solving algorithm for Π', then we would have one for Π too: given an instance $I \in \Pi$, we could just use the solving algorithm for Π' on $f(I)$. Since both the reduction and the solving algorithm would run under polynomial time, the whole process would be achieved within polynomial time too. We write $\Pi \leq_p \Pi'$ if Π is polynomial-time reducible to Π'. Of course the relations $\leq$ and $\leq_p$ are both transitive.

Combining the above complexity classes and this intuitive notion of "harder" problems leads us to the notion of *completeness*. We first need a few rough definitions beforehand. A *formula* F in *conjunctive normal form* is a conjunction of *clauses* $C_1, C_2, ..., C_m$ each consisting of a disjunction of *literals*, where a literal is either a *boolean variable* x_i or its *negation* $\overline{x_i}$ from a set of variables $\{x_1, x_2, ..., x_n\}$. A *(boolean) assignment* ϕ of the variables of F is a function $\{x_1, x_2, ..., x_n\} \to \{0, 1\}$, where each variable x_i is either set to *true* (if $\phi(x_i) = 1$) or *false* (otherwise, if $\phi(x_i) = 0$). This assignment ϕ naturally transposes to literals. In particular, the truth value assigned to $\overline{x_i}$ is the opposite of the one assigned to x_i. A clause of F is considered *satisfied* by ϕ if at least one of its literals is evaluated true by ϕ. Finally, we consider that F is *satisfied* by ϕ if ϕ satisfies all clauses of F. We say that F is *satisfiable* if there is a truth assignment to its variables making it satisfied.

Example 1.35. The conjunctive normal form formula $F = x_1 \wedge (x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_2} \vee x_3)$, involving three clauses and three variables, is satisfied if we set e.g. $\phi(x_1) = \phi(x_2) = \phi(x_3) = 1$.

We can now consider the following classic NP problem.

Satisfiability
Instance: A formula F in conjunctive normal form with clauses $C_1, C_2, ..., C_m$ over variables $x_1, x_2, ..., x_n$.
Question: Is F satisfiable?

An important theorem of Cook states that every instance of every problem in NP can be expressed, in polynomial-time, as an equivalent boolean formula in conjunctive normal form [44]. In other words, every problem in NP reduces to Satisfiability in polynomial time. Regarding the explanations above, Satisfiability can legitimately be considered as one of the hardest NP problems. Formally we say that Satisfiability is *complete* in NP, or *NP-complete* for short. Now note that if we have Satisfiability $\leq_p \Pi$ for another NP problem Π, then Π can also be regarded as one of the hardest problems of NP since every other NP problem Π' is polynomial-time reducible to it by transitivity, i.e. we have

$$\Pi' \leq_p \text{Satisfiability} \leq_p \Pi.$$

Polynomial-time reductions thus induce notions of equivalent and harder problems for a given complexity time-related class C, which are defined as follows. Assume C includes a base C-complete problem Π_c, i.e. $\Pi_c \in C$ and all other problems of C reduce to Π_c in polynomial

time. Then any other problem Π satisfying $\Pi_c \leq_p \Pi$ is "harder" than Π_c. We call Π *C-hard*. A C-hard problem which belongs to C is called *C-complete*. The fundamental difference between C-hard and C-complete problems is that finding a polynomial-time solving algorithm for a C-complete problem results in a polynomial-time algorithm for solving all problems in C, but does not imply that every C-hard problem is solvable in polynomial time (typically it does not imply anything regarding C-hard problems not in C).

Polynomial hierarchy

One alternative way to define the NP and co-NP classes is to make use of the notion of *oracle*. An oracle for a problem Π is a kind of "black box" which can instantaneously (i.e. with constant time) determine whether an instance of Π is positive or negative. Then the NP and co-NP classes can be defined as the sets of problems which admit a polynomial-time *yes*- or *no*-checking algorithm, respectively, making use of an oracle for a P problem.

Generalizing this alternative definition of NP and co-NP leads to the so-called *polynomial hierarchy*, which not only contains the classical NP and co-NP classes but also additional complexity classes containing problems with apparently higher time complexity. The polynomial hierarchy is made up of infinitely many levels, where each ith level consists in three complexity classes denoted Δ_i^p, Σ_i^p and Π_i^p. At the very base of the polynomial hierarchy, i.e. at level 0, we have

$$\Delta_0^p = \Sigma_0^p = \Pi_0^p = \mathsf{P}.$$

Now, for any $i \geq 1$ such that the $(i-1)$th level, i.e. the classes Δ_{i-1}^p, Σ_{i-1}^p and Π_{i-1}^p, has been defined, for a given problem Π, we have:

- $\Pi \in \Delta_i^p$ if Π admits a polynomial-time solving algorithm using a $\Sigma_{i-1}^p \cup \Pi_{i-1}^p$ oracle,
- $\Pi \in \Sigma_i^p$ if Π admits a polynomial-time *yes*-checking algorithm using a $\Sigma_{i-1}^p \cup \Pi_{i-1}^p$ oracle,
- $\Pi \in \Pi_i^p$ if Π admits a polynomial-time *no*-checking algorithm using a $\Sigma_{i-1}^p \cup \Pi_{i-1}^p$ oracle.

In particular, we have $\Delta_1^p = \mathsf{P}$, $\Sigma_1^p = \mathsf{NP}$, and $\Pi_1^p = \text{co-NP}$. The union of the complexity classes from all levels of the polynomial hierarchy is commonly denoted PH.

The above formal definitions of Σ_k^p and Π_k^p are not intuitive for the analysis of decision problems, but Σ_k^p and Π_k^p problems can generally be easily recognized by using the following observations. The fact that, for a problem $\Pi \in \mathsf{NP}$, we can efficiently check whether an instance I of Π is positive is because the question of Π is about the *existence* of an object related to the inputs of I. So basically we can guess one certificate for I, which may assess the positiveness of I. Conversely, the reason why we can easily check whether an instance of a co-NP problem is negative is because its question asks whether *all* some objects related to the inputs of I satisfy some properties. So we can hence show that I is negative by just exhibiting one counterexample certificate.

According to the above arguments and the relationship between two consecutive levels of the polynomial hierarchy, it can be easily seen that Σ_k^p gathers those problems whose question can be expressed as a formula involving an alternation of k quantifiers (either $\exists$, the existence, or $\forall$, the universality) starting from an $\exists$ symbol. Conversely, a problem in Π_k^p has its question being a formula involving k alternating quantifiers starting with a $\forall$ universal quantifier. So the polynomial hierarchy gathers those problems whose questions can be expressed as formulas involving a constant number of alternating quantifiers.

So that the notions of Σ_k^p- and Π_k^p-complete problems make sense, as explained previously we just need Σ_k^p or Π_k^p, respectively, problems to which all problems in Σ_k^p or Π_k^p, respectively,

reduce in polynomial time. Such problems were exhibited independently by Meyer and Stockmeyer [92] and Wrathall [120]. The archetypal Σ_k^p-complete problem is the following Σ_k^p variant of SATISFIABILITY.

$\exists\forall\exists\forall$... SATISFIABILITY
Instance: A formula F in conjunctive normal form with clauses $C_1, C_2, ..., C_m$ over variables $X = \{x_1, x_2, ..., x_n\}$, and a partition $X_1 \cup X_2 \cup ... \cup X_k$ of X.
Question: Is there a truth assignment to X_1 such that for all truth assignments to X_2 there exists a truth assignment to X_3 such that for all truth assignments to X_4... such that F is satisfied?

Analogously, the base Π_k^p-complete problem is the following Π_k^p variant of SATISFIABILITY.

$\forall\exists\forall\exists$... SATISFIABILITY
Instance: A formula F in conjunctive normal form with clauses $C_1, C_2, ..., C_m$ over variables $X = \{x_1, x_2, ..., x_n\}$, and a partition $X_1 \cup X_2 \cup ... \cup X_k$ of X.
Question: For all truth assignments to X_1 is there a truth assignment to X_2 such that for all truth assignments to X_3 there is a truth assignment to X_4... such that F is satisfied?

Parameterized time complexity

Parameterized complexity theory is a field of computational complexity theory which gained ingrowing attention since its recent introduction. The idea is roughly to consider parameters inherent to a (generally hard) problem Π, and to investigate whether these parameters are intimately related to the hardness of Π. These parameters can then reveal to not impact on the hardness of Π, i.e. instances of Π with small or large values of these parameters are as complicated to handle, or, on the contrary, to be related to the hardness of Π, i.e. instances of Π with small values of these parameters can be solved drastically easily than instances of Π having large values of these parameters. So two parameterized versions of Π can show up to have quite different time complexities, typically if one of these two versions is parameterized by a crucial parameter making Π difficult. More details on this topic can be found e.g. in the book of Downey and Fellows [51].

Formally, by a *parameter* k of Π, we refer to a function assigning a natural number to every instance of Π. We say that Π is *fixed-parameter tractable* (when parameterized by k) if every instance I of Π can be solved with time $\mathcal{O}(f(k) \cdot |I|^{\mathcal{O}(1)})$, where f is a computable function of k. So assuming the value of k associated to I is fixed, we basically get that I can be solved in time $\mathcal{O}(|I|^{\mathcal{O}(1)})$, where $f(k)$ is part of the constant hidden by the $\mathcal{O}$ notation, that is in polynomial time. The function f above can of course be replaced with a function of more than one parameter, so the definitions above can be naturally adapted to problems parameterized by several parameters.

Parameterized complexity can be considered as an extension of classic complexity for studying hard problems, notably NP-complete problems, in the following sense. By definition, an NP-complete problem Π should not admit a polynomial-time solving algorithm (unless $\mathsf{P} = \mathsf{NP}$). So the next interesting question is to determine whether Π is fixed-parameter tractable regarding some parameters, i.e. which parameters of Π make Π hard. Of course such parameters should not be easy to compute since otherwise we would get a polynomial-time solving algorithm for Π. But studying which aspects of a notoriously hard problem make it hard is generally a fascinating direction.

Space complexity

It might be the case that a problem Π is apparently so hard that it does not belong to the polynomial hierarchy at all, i.e. not only solving algorithms but also checking algorithms for

Π need more than polynomial time to achieve. In such a situation, it gets more convenient to estimate the complexity of Π in terms of space required to solve it.

In between PH and EXPTIME is then located the PSPACE class, which contains those problem which admit a polynomial-space solving algorithm. Clearly we have $\mathsf{PH} \subseteq \mathsf{PSPACE}$. Indeed, by definition, each problem Π in PH admits a polynomial-time checking algorithm. Since this checking algorithm runs in polynomial time, clearly the certificates it works on have polynomial size. So basically a polynomial-space solving algorithm for Π would generate all possible certificates using polynomial space, and call the checking algorithm on each of these (which uses polynomial space since it runs in polynomial time) to answer.

Similarly as for problems in PH, one straight way for recognizing typical PSPACE problems is that their questions can be expressed as formulas involving a polynomial number of quantifiers (either $\exists$ and $\forall$). The notion of PSPACE-complete problems is defined analogously as for the previous complexity classes we have introduced. The archetypal base PSPACE-complete problem is a PSPACE variant of Satisfiability called Quantified Boolean Formula, which is similar to the Σ_k^p and Π_k^p variants of Satisfiability, except that the number of quantifiers involved in its question is not constant. We refer the reader to the work of Stockmeyer and Meyer [114] for more details on Quantified Boolean Formula.

1.3 List of decision problems

We herein list some decision problems and point out some aspects of their time complexity.

1.3.1 Satisfiability-like problems

Satisfiability is known to remain NP-complete even under strong restrictions on the input formula, or on the conditions for a clause or the formula to be considered satisfied. Maybe one of the most known NP-complete restrictions of Satisfiability is 3-Satisfiability, which is the restriction where the formula is assumed to be a conjunction of 3-clauses.

Definition 1.36. For every $k \geq 1$, a *k-clause* refers to a disjunction of k literals.

3-Satisfiability
Instance: A formula F in conjunctive normal form with 3-clauses $C_1, C_2, ..., C_m$ over variables $x_1, x_2, ..., x_n$.
Question: Is F satisfiable?

3-Satisfiability is one of the twenty-one first problems shown to be NP-complete by Karp [82].

Theorem 1.37 ([82])**.** 3-Satisfiability *is NP-complete.*

Assuming that the clauses of a formula involving 3-clauses are satisfied when they have a certain number of true literals, we come up with the upcoming two restrictions of 3-Satisfiability.

Definition 1.38. Let F be a formula in conjunctive normal form, and ϕ be a truth assignment to the variables of F. We say that F is *1-in-3 satisfied* if every of its clauses has exactly one true literal by ϕ. We say that F is *not-all-equal satisfied* if every of its clauses has at least one true literal and at least one false literal by ϕ.

1-in-3 Satisfiability
Instance: A formula F in conjunctive normal form with 3-clauses $C_1, C_2, ..., C_m$ over variables $x_1, x_2, ..., x_n$.
Question: Is F 1-in-3 satisfiable?

Not-All-Equal 3-Satisfiability
Instance: A formula F in conjunctive normal form with 3-clauses $C_1, C_2, ..., C_m$ over variables $x_1, x_2, ..., x_n$.
Question: Is F not-all-equal satisfiable?

1-in-3 Satisfiability and Not-All-Equal 3-Satisfiability were both shown to be NP-complete by Schaefer's dichotomy theorem [106].

Theorem 1.39 ([106])**.** 1-in-3 Satisfiability *and* Not-All-Equal 3-Satisfiability *are NP-complete.*

So that we give the complexity status of restrictions of both 1-in-3 Satisfiability and Not-All-Equal 3-Satisfiability, we need to introduce further notions. We start by clarifying the notion of *monotone formula*.

Definition 1.40. A formula F in conjunctive normal form is *monotone* if none of its clauses involves a negated variable.

We also define what is a *planar formula*.

Definition 1.41. Let F be a formula in conjunctive normal form. By the *graph underlying* F, we refer to the bipartite graph G whose vertex set $C \cup L$ includes the following vertices:

- for every clause C_j of F, there is a vertex v_{C_j} in C,
- for every literal ℓ_i of F, there is a vertex v_{ℓ_i} in L,

and in which two vertices v_{C_j} and v_{ℓ_i} are joined if and only if $\ell_i \in C_j$.

Definition 1.42. A formula in conjunctive normal form is *planar* if its underlying graph is planar.

We can now introduce the following NP-complete restrictions of 3-Satisfiability, 1-in-3 Satisfiability, and Not-All-Equal 3-Satisfiability, where prefixing every of these problems with Monotone means that we further assume that the input formula F is monotone, while prefixing them with Planar means that F is also assumed planar.

Theorem 1.43. *The following problems are NP-complete:*

- Planar 3-Satisfiability *(Lichtenstein [87]),*
- Monotone 1-in-3 Satisfiability *(Schaefer [106]),*
- Planar 1-in-3 Satisfiability *(Dyer and Frieze [55]),*
- Planar Monotone 1-in-3 Satisfiability *(Laroche [85]),*
- Monotone Not-All-Equal 3-Satisfiability *(Schaefer [106]).*

Of course Monotone 3-Satisfiability is trivially in P since every of its instances is positive. It is further important mentioning that Planar Not-All-Equal 3-Satisfiability was surprisingly shown to be in P by Moret [96].

We now raise some easy remarks regarding instances of 3-Satisfiability, 1-in-3 Satisfiability and Not-All-Equal 3-Satisfiability.

Observation 1.44. *When dealing with a formula F being an instance of* 3-Satisfiability*,* 1-in-3 Satisfiability *or* Not-All-Equal 3-Satisfiability*, we can suppose that all literals over the variables of F appear in F.*

Proof. Assume a literal ℓ_i does not appear in F. Then just note that

$$F \wedge (\ell_i \vee \ell_i \vee \overline{\ell_i})$$

is a formula involving ℓ_i and equivalent to F in case F is an instance of 3-SATISFIABILITY or NOT-ALL-EQUAL 3-SATISFIABILITY, while

$$F \wedge (\ell_i \vee \overline{\ell_i} \vee x_{n+1}) \wedge (x_{n+1} \wedge x_{n+1} \vee \overline{x_{n+1}}),$$

where x_{n+1} is a new variable, is a formula involving ℓ_i and equivalent to F when F is an instance of 1-IN-3 SATISFIABILITY. ■

Repeating the modification from the proof of Observation 1.44 for all literals which do not appear in F, we get a formula equivalent to F but involving all possible literals over its variables. This procedure is achieved in polynomial time.

We now introduce the notion of *forced literal* before raising more observations.

Definition 1.45. Let F be a formula from an instance of a SATISFIABILITY-like problem. A literal ℓ_i of F is *forced* to true (resp. false) (by a clause C of F) if, regarding C, necessarily ℓ_i is set to true (resp. false) by every satisfying truth assignment satisfying F.

Observation 1.46. *If a clause C of an instance F of* 3-SATISFIABILITY *is of the form $(\ell_i \vee \ell_i \vee \ell_i)$, then ℓ_i is forced to true by C.*

Observation 1.47. *If a clause C of an instance F of* 1-IN-3 SATISFIABILITY *is of the form $(\ell_i \vee \ell_j \vee \ell_j)$ with $\ell_i \neq \ell_j$, then ℓ_i is forced to true and ℓ_j is forced to false by C.*

Observation 1.48. *If a clause of an instance F of* 1-IN-3 SATISFIABILITY *or* NOT-ALL-EQUAL 3-SATISFIABILITY *is of the form $(\ell_i \vee \ell_i \vee \ell_i)$, then F cannot be satisfied.*

Our reductions often make use of the following notation.

Notation 1.49. Let F be a formula from an instance of a SATISFIABILITY-like problem. For every clause C_j of F, we denote $m(C_j)$ the number of distinct literals in C_j. For every literal ℓ_i (resp. variable x_i in case the problem is MONOTONE) of F, we denote $n(\ell_i)$ (resp. $n(x_i)$) the number of distinct clauses which contain ℓ_i (resp. x_i).

As for higher levels of the polynomial hierarchy, all Σ_k^p and Π_k^p versions of 3-SATISFIABILITY, i.e. $\exists\forall\exists\forall$...3-SATISFIABILITY and $\forall\exists\forall\exists$...3-SATISFIABILITY with k alternating quantifiers being involved, were both shown to be complete in their respective class by Stockmeyer [113] and Wrathall [120], independently.

Theorem 1.50 ([113] and [120], independently)**.** *Every problem $\exists\forall\exists\forall$...3-SATISFIABILITY or $\forall\exists\forall\exists$...3-SATISFIABILITY involving exactly k alternating quantifiers is Σ_k^p- or Π_k^p-complete, respectively.*

In the same vein, we also introduce the Π_2^p-complete version of 1-IN-3 SATISFIABILITY.

$\forall\exists$ 1-IN-3 SATISFIABILITY
Instance: A formula F in conjunctive normal form with 3-clauses $C_1, C_2, ..., C_m$ over variables $X = \{x_1, x_2, ..., x_n\}$, and a bipartition $X_1 \cup X_2$ of X.
Question: For all truth assignments to X_1, is there a truth assignment to X_2 such that F is 1-in-3 satisfied?

As we did not find any proof of the Π_2^p-completeness of $\forall\exists$ 1-IN-3 SATISFIABILITY in the literature, we show it below by reduction from $\forall\exists$ 3-SATISFIABILITY. This reduction is nothing but a generalization of a common reduction from 3-SATISFIABILITY to 1-IN-3 SATISFIABILITY given by Schaefer in [106].

Lemma 1.51. ∀∃ 1-IN-3 SATISFIABILITY *is* Π_2^p*-complete.*

Proof. ∀∃ 1-IN-3 SATISFIABILITY is clearly in Π_2^p. One can indeed design an algorithm that takes, as inputs, F and a truth assignment ϕ_1 to the variables of X_1 for which there is no truth assignment ϕ_2 to the variables in X_2 making F evaluated in a 1-in-3 way. It just has to check that ϕ_2 does not exist by invoking an oracle dealing with 1-IN-3 SATISFIABILITY. Such a checking algorithm runs in polynomial time regarding the size of F.

We now show that ∀∃ 1-IN-3 SATISFIABILITY is Π_2^p-complete by reduction from ∀∃ 3-SATISFIABILITY. From a formula F in conjunctive formal form with 3-clauses $C_1, C_2, ..., C_m$ over variables $X = \{x_1, x_2, ..., x_n\}$ and a bipartition $X_1 \cup X_2$ of X, we construct another formula F' in conjunctive normal form involving 3-clauses over variables among a set X' admitting a bipartition $X'_1 \cup X'_2$ such that

for all truth assignments ϕ_1 to X_1, there exists a truth assignment ϕ_2 to X_2 making F satisfied

$$\Leftrightarrow$$

for all truth assignments ϕ'_1 to X'_1, there exists a truth assignment ϕ'_2 to X'_2 making F' 1-in-3 satisfied.

The reduction is as follows. First, for each clause $C_i = (\ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3})$ of F, add four clauses

$$(\overline{\ell_{i_1}} \vee a_i \vee b_i),\ (\overline{\ell_{i_2}} \vee c_i \vee d_i),\ (\overline{\ell_{i_3}} \vee e_i \vee f_i) \text{ and } (a_i \vee c_i \vee e_i)$$

to F', where a_i, b_i, c_i, d_i, e_i and f_i are six new variables associated with C_i. Finally, let $X'_1 = X_1$ and $X'_2 = X_2 \cup \bigcup_{i=1}^{m} \{a_i, b_i, c_i, d_i, e_i, f_i\}$. Note that F' has $4m$ clauses and may thus be obtained in polynomial time.

First suppose that for every truth assignment ϕ'_1 of X'_1 there exists a truth assignment ϕ'_2 of X'_2 for which F' is 1-in-3 satisfied. Consider every four clauses $(\overline{\ell_{i_1}} \vee a_i \vee b_i)$, $(\overline{\ell_{i_2}} \vee c_i \vee d_i)$, $(\overline{\ell_{i_3}} \vee e_i \vee f_i)$ and $(a_i \vee c_i \vee e_i)$ for every $i \in \{1, 2, ..., m\}$. Because every clause of F' has exactly one true literal by ϕ'_1 and ϕ'_2, it means that only one element in $\{a_i, c_i, e_i\}$ is evaluated true by ϕ'_2. Let us suppose that for such an i we have $\phi'_2(a_i) = 1$ and $\phi'_2(c_i) = \phi'_2(e_i) = 0$ without loss of generality. Thus, we have ℓ_{i_1} evaluated true by either ϕ'_1 or ϕ'_2. It follows that the following truth assignments ϕ_1 and ϕ_2 of X_1 and X_2, respectively,

- $\phi_1 = \phi'_1$,
- $\phi_2(x) = \phi'_2(x)$ for every $x \in X_2$,

make F satisfied. Conversely, suppose that for every truth assignment ϕ_1 of X_1 there is a truth assignment ϕ_2 of X_2 such that F has all its clauses satisfied by ϕ_1 and ϕ_2. We explain how to get a truth assignment ϕ'_2 to X'_2 so that F' is evaluated true in a 1-in-3 way under ϕ'_2 and the truth assignment $\phi'_1 = \phi_1$ to X'_1. First, let $\phi'_2(x) = \phi_2(x)$ for every $x \in X_2$. We then have to provide a truth assignment of a_i, b_i, c_i, d_i, e_i and f_i by ϕ'_2 for every $i \in \{1, 2, ..., m\}$. This assignment depends on the number of true literals in $C_i = (\ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3})$ via ϕ_1 and ϕ_2. Let $\phi_3 : X_1 \cup X_2 \to \{0, 1\}$ be the truth assignment of $X_1 \cup X_2$ deduced from ϕ_1 and ϕ_2 as follows:

- for every $x \in X_1$, set $\phi_3(x_i) = \phi_1(x_i)$;
- for every $x \in X_2$, set $\phi_3(x_i) = \phi_2(x_i)$.

Consider now that the images of the a_i's, b_i's, c_i's, d_i's, e_i's and f_i's by ϕ'_2 are the ones depicted in Table 1.16. It should then be clear that F' is evaluated true in a 1-in-3 way under ϕ'_1 and ϕ'_2. ■

$(\phi_3(\ell_{i_1}), \phi_3(\ell_{i_2}), \phi_3(\ell_{i_3}))$	$\phi'_2(a_i)$	$\phi'_2(b_i)$	$\phi'_2(c_i)$	$\phi'_2(d_i)$	$\phi'_2(e_i)$	$\phi'_2(f_i)$
$(1,0,0)$	1	0	0	0	0	0
$(0,1,0)$	0	0	1	0	0	0
$(0,0,1)$	0	0	0	0	1	0
$(1,1,0)$	1	0	0	1	0	0
$(1,0,1)$	1	0	0	0	0	1
$(0,1,1)$	0	0	1	0	0	1
$(1,1,1)$	1	0	0	1	0	1

Table 1.16: Truth assignment of the variables in $X_2 \setminus X'_2$ by ϕ'_2.

1.3.2 Graph problems

We kick off with the following well-known NP-complete problem, whose one first NP-completeness proof is due to Garey and Johnson [62].

Hamiltonian Path
Instance: A graph G.
Question: is G traceable?

Now consider the following hypergraph colouring problem.

2-Colouring of 3-Uniform Hypergraph
Instance: A 3-uniform hypergraph H.
Question: Is there a 2-vertex-colouring of H such that no edge has its three vertices assigned the same colour?

2-Colouring of 3-Uniform Hypergraph was shown to be NP-complete by Lovász [88].

Theorem 1.52 ([88])**.** 2-Colouring of 3-Uniform Hypergraph *is NP-complete.*

One worthy observation is that every instance of 2-Colouring of 3-Uniform Hypergraph can be regarded as an instance of Monotone Not-All-Equal 3-Satisfiability whose formula has all of its clauses having three distinct variables. We hence get the following.

Observation 1.53. Monotone Not-All-Equal 3-Satisfiability *remains NP-complete when we have* $m(C_j) = 3$ *for every clause* C_j *of* F.

1.3.3 Partition problems

One NP-complete partition problem which we use in further chapters is the following.

3-Partition
Instance: A set $A = \{a_1, a_2, ..., a_{3m}\}$ of $3m$ elements, a bound $B \in \mathbb{Z}^+$, and a size $s : A \to \mathbb{Z}^+$ such that $\frac{B}{4} < s(a) < \frac{B}{2}$ for every $a \in A$ and $\sum_{a \in A} s(a) = mB$.
Question: Can A be partitioned into m parts $A_1 \cup A_2 \cup ... \cup A_m$ such that we have $\sum_{a \in A_i} s(a) = B$ for every $i \in \{1, 2, ..., m\}$?

One important point of interest is that 3-Partition remains NP-complete when:

- the sizes by s are not small, i.e. at least 2,
- B and $\min\{s(a) : a \in A\}$ are larger than an integer constant $c \geq 1$, and
- either B or $\min\{s(a) : a \in A\}$ has given parity.

A proof for the above first item only was given by Dell'Amico and Martello in [47]. Our proof below is slightly different but also agree with the above two other items.

Observation 1.54. 3-PARTITION *remains NP-complete when restricted to an instance* $<A, B, s>$ *where* $s(a) > 1$ *holds for every element* $a \in A$*, where* B *and* $\min\{s(a) : a \in A\}$ *are larger than an integer constant* $c \geq 1$*, and either* B *or* $\min\{s(a) : a \in A\}$ *has given parity.*

Proof. It is easily seen that $<A, B', s'>$, where

- $B' = B + 3$,
- $s'(a) = s(a) + 1$ for every $a \in A$,

is another instance of 3-PARTITION (in particular $\frac{B}{4} < s(a) < \frac{B}{2}$ implies $\frac{B'}{4} < s'(a) < \frac{B'}{2}$ for every $a \in A$) which is equivalent to $<A, B, s>$. Besides, by successively performing this transformation an arbitrary number of times starting from $<A, B, s>$, we keep on getting new instances of 3-PARTITION which are equivalent to $<A, B, s>$. The claim then follows by performing this transformation the appropriated number of times. In particular, the first condition is met as soon as the transformation is performed at least once. The second condition is fulfilled by performing at least c such transformations. If the third condition is not already met, then just perform the transformation once again. ■

Part I

Partitioning graphs into connected subgraphs

Chapter 2
Introduction to Part I

In this introductory chapter, we give all the contents which are necessary to understand our results from Chapters 3, 4 and 5. We start by introducing and motivating the notion of arbitrarily partitionable graphs in Section 2.1. We then give formal definitions and terminology in Section 2.2 which are used throughout Part I. Using these, we detail, in Section 2.3, the most important past works related to our investigations. We eventually give an overview of our contributions in Section 2.4.

2.1 Motivations

Perhaps the most influential result regarding the investigations of Part I is the following one, due independently to Győri and Lovász in the 1970's (answering a question raised by Frank during a conference held at Aberdeen in 1975), though the problem of partitioning graphs into connected subgraphs surely gained attention earlier.

Theorem 2.1 ([89] and [66], independently)**.** *Let G be a k-connected graph, $v_1, v_2, ..., v_k$ be k distinct vertices of G, and $n_1, n_2, ..., n_k$ be k positive integers summing up to $|V(G)|$. Then there exists a partition $V_1 \cup V_2 \cup ... \cup V_k$ of $V(G)$ such that $G[V_i]$ is connected, has order n_i, and includes v_i for every $i \in \{1, 2, ..., k\}$.*

The notion of *arbitrarily partitionable graphs*, which is the main notion investigated in Part I, is related to Theorem 2.1 and was introduced by Barth, Baudon and Puech for studying the following practical problem [11]. Assume we own a connected network of resources we want to share among $p \geq 1$ users, where the ith user requests exactly $n_i \geq 1$ of our resources. For the sake of performance, we do not want the sharing to be done arbitrarily, but in such a way that the following two conditions are met:

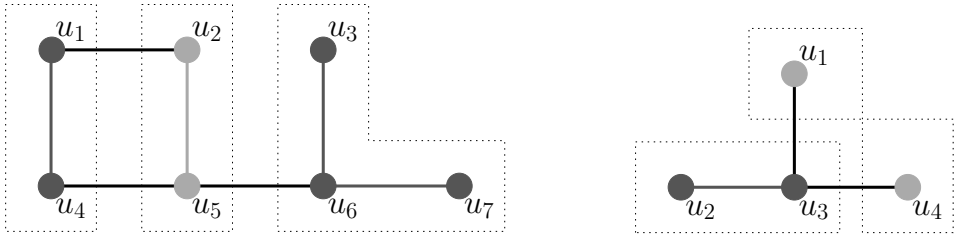
Condition 1: each resource is attributed to exactly one user,

Condition 2: two resources from a same subnetwork must be able to communicate within it.

Since each user requests a given number of resources, we also have to fulfil the following:

Condition 3: each of the p users gets the number of resources he requests.

We use graph theory to deal with this problem. Our network can be regarded as a graph G where each vertex of G corresponds to one resource of our network, two vertices u and v of G being joined by an edge if the associated two resources of our network are directly interconnected. Sharing our network in such a way that Conditions 1, 2 and 3 are satisfied is then similar to



(a) A partition $(\{u_1, u_4\}, \{u_2, u_5\}, \{u_3, u_6, u_7\})$ of the vertex set.

(b) A partition $(\{u_2, u_3\}, \{u_1, u_4\})$ of the vertex set.

Figure 2.1: Partitions of two graphs (in black and grey) into subgraphs (in shades of grey).

finding a partition $V_1 \cup V_2 \cup ... \cup V_p$ of $V(G)$ (Condition 1) such that $G[V_1], G[V_2], ..., G[V_p]$ are connected (Condition 2) and have order $n_1, n_2, ..., n_p$, respectively (Condition 3). So we come up with the following notion of *realization* of a sequence in a graph.

Definition 2.2. Let G be a graph, and $\pi = (n_1, n_2, ..., n_p)$ be a sequence of positive integers summing up to $|V(G)|$. We say that π is *realizable* in G if there is a *realization* of π in G, that is a partition $(V_1, V_2, ..., V_p)$ of $V(G)$ such that $G[V_i]$ is connected and has order n_i for every $i \in \{1, 2, ..., p\}$.

Example 2.3. The partition of the vertex set of the graph G_1 depicted in Figure 2.1.a is a realization of the sequence $(2, 2, 3)$ in G_1. On the contrary, the partition of the vertex set of the graph G_2 drawn in Figure 2.1.b is not a realization of $(2, 2)$ in G_2 since one of the two induced subgraphs (the one in light grey) is not connected.

Now assuming that neither the number of users requesting a part of our network nor the number of resources they each request are known, as the network owner we would like our network to be shareable at will. Regarding the graph theory point of view, we would then like the graph underlying our network to be *arbitrarily partitionable*.

Definition 2.4. A graph G is *arbitrarily partitionable* if all sequences of positive integers summing up to $|V(G)|$ are realizable in G.

Example 2.5. It can be checked that the graph in Figure 2.1.a is arbitrarily partitionable. The graph drawn in Figure 2.1.b is not arbitrarily partitionable since the sequence $(2, 2)$ is not realizable in it.

Regarding the above practical problem, studying arbitrarily partitionable graphs with respect to both the algorithmic (is it hard to find a realization of a sequence in a graph? or to recognize an arbitrarily partitionable graph?) and structural (what does an arbitrarily partitionable graph look like?) points of view is thus of interest. This practical concern apart, the property of being arbitrarily partitionable also turned out to be related to some classic notions of graph theory, especially with the properties of having a (possibly quasi-) perfect matching, which is a necessary condition for being arbitrarily partitionable, or of being traceable, which is a sufficient condition for being arbitrarily partitionable.

Though the notion of arbitrarily partitionable graphs is quite natural regarding the above practical problem, it could be refined to tackle the following two issues arising from our definitions.

Issue 1: When partitioned following a sequence, a graph is fully partitioned at once in the sense that *each* of its vertices is included into a part. From the network sharing point of view, this constraint is like waiting for every single resource of our network to be requested before

eventually satisfying the users. This is of course not satisfying since we would like to satisfy the users as soon as possible (immediately, ideally).

Issue 2: When a sequence is realized in a graph, the induced subgraphs must only meet the connectivity constraint. But according to our network analogy, it would be more convenient to make sure that the allocated subnetworks themselves have the property of being shareable at will. This would be useful if, for example, a user wants himself to share his subnetwork among several other users.

To deal with the above two issues, we need notions of graph partitions meeting additional constraints. We then come up with the following notions of *on-line* and *recursively arbitrarily partitionable graphs* which were introduced by Horňák, Tuza and Woźniak in [72], and Baudon, Gilbert and Woźniak in [24], respectively.

Definition 2.6. A graph G is *on-line arbitrarily partitionable* if either

- $G \simeq K_1$, or
- for every $\lambda \in \{1, 2, ..., |V(G)| - 1\}$, there is a bipartition $V_\lambda \cup V_{|V(G)|-\lambda}$ of $V(G)$ such that $G[V_\lambda]$ is connected and has order λ, and $G[V_{|V(G)|-\lambda}]$ is on-line arbitrarily partitionable and has order $|V(G)| - \lambda$.

Definition 2.7. A graph G is *recursively arbitrarily partitionable* if either

- $G \simeq K_1$, or
- for every sequence $\pi = (n_1, n_2, ..., n_p)$ of positive integers summing up to $|V(G)|$, there is a realization $(V_1, V_2, ..., V_p)$ of π in G such that $G[V_i]$ is recursively arbitrarily partitionable for every $i \in \{1, 2, ..., p\}$.

Then if, as the network owner, we want to tackle Issue 1 or 2, respectively, then we need the graph underlying our network to be on-line or recursively arbitrarily partitionable, respectively. This supports the study of both the algorithmic and structural properties of on-line or recursively arbitrarily partitionable graphs.

2.2 Definitions, terminology and notation

We first raise two important remarks concerning the terminology and notation introduced in Definition 2.2. It should be clear that what we call a "sequence of integers summing up to $|V(G)|$" in this definition is nothing but a multiset of integers performing a partition of $|V(G)|$. Since we often deal with the notions of partition of an integer and partition of a set simultaneously throughout Part I, to avoid any confusion we voluntarily refer to a partition of an integer as a *sequence*. Besides, we make use of a tuple notation for denoting a sequence of integers, e.g. $\pi = (n_1, n_2, ..., n_p)$, and a realization of π in a graph, e.g. $(V_1, V_2, ..., V_p)$, because of the straight relationship between these two objects (that is the ith part V_i has size n_i). The use of this notation also allows the introduction of well-defined operations involving sequences of integers or realizations of sequences in graphs.

Sequences of integers

Let $\pi = (n_1, n_2, ..., n_p)$ be a sequence of $p \geq 1$ positive integers. We say that π is an *n-sequence* if π forms a partition of n. The n-sequences $(1, 1, ..., 1)$ and (n) are said *trivial*. By the *size* of π, we refer to the number p of elements in π. This parameter is denoted $|\pi|$. We sometimes refer to the sum n of the elements in π as $\|\pi\|$. The *spectrum* of π, denoted $sp(\pi)$, is the set of values which appear in π. Again, the number of elements in $sp(\pi)$ is the *size* of $sp(\pi)$, denoted $|sp(\pi)|$.

Example 2.8. The spectrum of the 18-sequence $(1, 1, 2, 3, 3, 4, 4)$ is

$$sp((1, 1, 2, 3, 3, 4, 4)) = \{1, 2, 3, 4\}$$

and has size 4, while $(1, 1, 2, 3, 3, 4, 4)$ has size 7.

Let σ be a permutation of the set $\{1, 2, ..., p\}$, and $r \in \{1, 2, ..., p-1\}$ be an index. Then π can be *partitioned* into two sequences $\pi_1 = (n_{\sigma(1)}, n_{\sigma(2)}, ..., n_{\sigma(r)})$ and $\pi_2 = (n_{\sigma(r+1)}, n_{\sigma(r+2)}, ..., n_{\sigma(p)})$. The definition of a partition of π into more than two sequences is defined analogously. Writing $\pi = \pi_1 \cup \pi_2$ or $\pi_2 = \pi \setminus \pi_1$ should be clear whenever π_1 and π_2 form a partition of π, or a specific ordering of the operand sequences is assumed (e.g. increasing or decreasing order). In all other cases, the ordering of such resulting sequences will be explicitly given.

Realizations of a sequence

The previous remark also applies to realizations of sequences in graphs, so that we can write both

$$(U_1, U_2, ..., U_p) \cup (V_1, V_2, ..., V_{p'})$$

and

$$(V_1, V_2, ..., V_p) \setminus (V_{i_1}, V_{i_2}, ..., V_{i_p})$$

when dealing with disjoint subsets of vertices.

The decision problem related to the existence of an ordinary realization of a sequence in a graph reads as follows.

Realizable Sequence
Instance: a graph G and a $|V(G)|$-sequence π.
Question: is π realizable in G?

We now consider the recursive variants of the notion of arbitrarily partitionable graphs, i.e. on-line and recursively arbitrarily partitionable graphs. By an *on-line λ-partition* of a graph G for some $\lambda \in \{1, 2, ..., |V(G)| - 1\}$, we refer to a bipartition $V_\lambda \cup V_{|V(G)|-\lambda}$ of $V(G)$ such that $G[V_\lambda]$ is a connected graph on λ vertices and $G[V_{|V(G)|-\lambda}]$ is an on-line arbitrarily partitionable graph on $|V(G)| - \lambda$ vertices.

As for recursively arbitrarily partitionable graphs, a realization $(V_1, V_2, ..., V_p)$ of a $|V(G)|$-sequence π in G such that $G[V_i]$ is recursively arbitrarily partitionable for every $i \in \{1, 2, ..., p\}$ is called a *recursive realization*. We say that π is *recursively realizable* in G as soon as there exists a recursive realization of π in G.

Partitioning a graph in every possible manner

Let $k \geq 1$ be a positive integer. If all $|V(G)|$-sequences with size k are realizable in G, then G is said *arbitrarily k-partitionable*. So G is arbitrarily partitionable if G is arbitrarily k-partitionable for every $k \in \{1, 2, ..., |V(G)|\}$. Similarly as for Realizable Sequence, we introduce the decision problem associated with the notion of arbitrarily partitionable graphs.

Arbitrarily Partitionable Graph
Instance: a graph G.
Question: is G arbitrarily partitionable?

Regarding the notion of on-line partition introduced in the previous section, rephrased differently, we get that G is on-line arbitrarily partitionable if G is K_1 or there is an on-line λ-partition

of G for every $\lambda \in \{1, 2, ..., |V(G)| - 1\}$. Besides, the graph G is recursively arbitrarily partitionable if G is K_1 or all $|V(G)|$-sequences are recursively realizable in G.

In case G has one of the partition properties above, i.e. G is (possibly on-line or recursively) arbitrarily partitionable, but $G - \{e\}$ looses this property for every edge $e \in E(G)$, we say that G is *minimal* (with respect to this partition property).

Kernel of sequences

Let K be a set of n-sequences for a given value of $n \geq 1$. We say that K is *realizable* in a graph G with order n if all sequences of K are realizable in G. If G is arbitrarily partitionable if and only if K is realizable in G, then we call K a *kernel* for G. This notion of kernel of sequences extends to families of graphs: we say that K is a kernel for a family $\mathcal{F}$ of graphs with order n if and only if K is a kernel for every member of $\mathcal{F}$. A kernel of n-sequences is said *polynomial* if it has size $\mathcal{O}(n^{\mathcal{O}(1)})$.

Example 2.9. For every $n \geq 1$, by definition, the set

$$\{\pi : \|\pi\| = n\}$$

is a kernel for the set of all graphs with order n. It is however not polynomial since the number of n-sequences grows exponentially compared to n, recall Theorem 1.2.

2.3 Related work

The notion of arbitrarily partitionable graphs and its recursive variants have been considered regarding four main aspects. Firstly, the algorithmic aspect: given a graph G, is it easy to decide whether some $|V(G)|$-sequences are realizable in G? Secondly, the structural aspect: what is the typical structure of an (possibly on-line or recursively) arbitrarily partitionable graph? Thirdly, the structural properties of minimal arbitrarily partitionable graphs. Fourthly, the relationship between the properties of being arbitrarily partitionable and being Hamiltonian. We survey some results related to these four aspects in this section.

Algorithmic aspects of partitioning graphs into connected subgraphs

This section mainly deals with the algorithmic complexity of Realizable Sequence and Arbitrarily Partitionable Graph. One consequence of Theorem 2.1 is that every instance of Realizable Sequence involving a k-connected graph and a sequence with size k is positive. However, the two independent proofs of Theorem 2.1 by Győri and Lovász are not constructive, and hence do not provide an efficient way for deducing a realization of any sequence with size k in a k-connected graph. Polynomial-time algorithms for deducing such realizations have been proposed for small values of k and (sometimes) restricted families of k-connected graphs. Such algorithms were in particular proposed for $k = 2$ by Suzuki, Takahashi and Nishizeki [115], and for $k = 3$ by Miyano, Nishizeki, Takahashi and Uneo [93]. A polynomial-time algorithm for specific situations of the case $k = 4$ was also proposed by Nakano, Rahman and Nishizeki in [97]. Namely, their algorithm deduces a realization in case the input graph is a 4-connected planar graph and the preassigned vertices are located on a same face. A polynomial-time algorithm for the general case, i.e. for all values of k, was proposed by Ma and Ma in [90] but is unfortunately wrong, as first mentioned in the work of Nakano, Rahman and Nishizeki [97]. Examples of graphs on which the algorithm of Ma and Ma fails were explicitly exhibited by Hofer and Lambert [70].

Series of positive results regarding Realizable Sequence were also exhibited by Diwan in [49] and Diwan and Kurhekar in [50]. In particular, it was proved in [50] that plane triangulations are arbitrarily k-partitionable for every $k \leq 6$, this result being tight since there exist plane

triangulations which are not arbitrarily 7-partitionable. In [49] was considered the partitioning of k-connected graphs with maximum degree at most $k+1$ into at most $k+1$ connected parts, wherein it is shown that such partitions exist when $k = 2$. So in the same vein as above, these results yield yet other conditions under which some instances of Realizable Sequence are necessarily positive.

In general, that is when not restricted to k-connected graphs and sequences with size k, the Realizable Sequence problem is known to be NP-complete, even under the following restrictions.

Theorem 2.10. Realizable Sequence *is NP-complete, even when*

- $\pi = (3, 3, ..., 3)$ *(Dyer and Frieze [54]),*
- G *is a comb (Ravaux [104]),*
- G *is a split graph (Broesma, Kratsch and Woeginger [39]).*

One important remark to raise regarding Theorem 2.10 is that its first item is tight in the sense that Realizable Sequence is not NP-complete under the restrictions $sp(\pi) = \{k\}$ and $k < 3$. It is indeed clear that the $|V(G)|$-sequence $(1, 1, ..., 1)$ is realizable in every graph G, while looking for a realization of the $|V(G)|$-sequence $(2, 2, ..., 2)$ in G is actually similar to finding a perfect matching of G, which can be done in polynomial time by the so-called Blossom algorithm by Edmonds [56].

Theorem 2.11 ([56])**.** *A maximum matching of a graph G can be computed in $\mathcal{O}(|V(G)|^{\mathcal{O}(1)})$.*

The general NP-hardness of Realizable Sequence aside, parameters for which Realizable Sequence is fixed-parameter tractable were exhibited in [104] by Ravaux. In particular, Ravaux proved that Realizable Sequence can be handled in linear time when restricted to trees and parameterized by the size of π, and can be generally treated in linear time when parameterized by both the size of π and the number of vertices with degree at least 3 of G.

The complexity of the Arbitrarily Partitionable Graph problem is still not clear at the moment. As first pointed out by Barth and Fournier in [12], it is easily seen that Arbitrarily Partitionable Graph is a Π_2^p problem since its question can be reformulated as *"for every $|V(G)|$-sequence π, is there a realization of π in G?"*, which clearly catches the form of a Π_2^p question, and Realizable Sequence is in NP (as understood in Theorem 2.10). It has however not been proved yet whether Arbitrarily Partitionable Graph is Π_2^p-complete or not.

The membership of Arbitrarily Partitionable Graph to NP does not seem trivial as the number of n-sequences is asymptotically exponential in n, recall Theorem 1.2. Hence, a certificate for an instance $<G>$ of Arbitrarily Partitionable Graph should naively have exponential size since one would have to provide a concrete proof that all $|V(G)|$-sequences are realizable in G (unless we can overcome this, see below). Conversely, the Arbitrarily Partitionable Graph problem does not seem to be in co-NP. Basically one could point out a sequence π which is not realizable in G, but then checking in polynomial time whether π is indeed not realizable in G seems difficult as the number of potential realizations of π in G is also exponential in $|V(G)|$.

Arbitrarily Partitionable Graph can nevertheless be handled in polynomial time when restricted to particular classes of graphs, namely multipodes (as shown by Barth, Baudon and Puech [11], Barth and Fournier [12], and Ravaux [104]), and split graphs (as shown by Broesma, Kratsch and Woeginger [39]). The fact that Realizable Sequence is NP-complete when restricted to split graphs (Theorem 2.10) while Arbitrarily Partitionable Graph is in P when restricted to split graphs is intriguing but can be explained as follows. Let $\mathcal{S}(n)$ denote the set of split graphs with order n. The membership of Arbitrarily Partitionable Graph to

P when restricted to split graphs results from the fact that every set $\mathcal{S}(n)$ admits a polynomial kernel $K_{\mathcal{S}}(n)$ of sequences whose realizability in graphs of $\mathcal{S}(n)$ is easy to check. So basically Realizable Sequence is in P when restricted to graphs of $\mathcal{S}(n)$ and sequences of $K_{\mathcal{S}}(n)$, but NP-complete when restricted to graphs of $\mathcal{S}(n)$ and some sequences not in $K_{\mathcal{S}}(n)$.

We summarize these ideas as follows. By first exhibiting a polynomial kernel $K_{\mathcal{F}}(n)$ for a given family $\mathcal{F}(n)$ of graphs with order n, we directly get that Arbitrarily Partitionable Graph is in NP when restricted to graphs of $\mathcal{F}(n)$. By then proving that Realizable Sequence is in P when restricted to graphs of $\mathcal{F}(n)$ and sequences of $K_{\mathcal{F}(n)}$, we get that Arbitrarily Partitionable Graph is in P for graphs of $\mathcal{F}(n)$. These remarks are related to one of the most important algorithmic open questions related to arbitrarily partitionable graphs, which was independently raised by Barth and Fournier [12] and Broesma, Kratsch and Woeginger [39].

Conjecture 2.12 ([12] and [39], independently)**.** Arbitrarily Partitionable Graph *is in NP.*

Conjecture 2.12 could be basically tackled by showing that every graph admits a polynomial kernel. It is worth mentioning that it is still unknown whether there are classes of graphs for which Arbitrarily Partitionable Graph is NP-complete. To prove such a result, i.e. that Arbitrarily Partitionable Graph is NP-complete when restricted to graphs of $\mathcal{F}(n)$, it would be necessary to show that Realizable Sequence is NP-complete when restricted to graphs of $\mathcal{F}(n)$ and sequences from every polynomial kernel for $\mathcal{F}(n)$.

So that the reader gets an idea of what a polynomial kernel may look like, we recall the following polynomial kernels for $\mathcal{T}(n)$, the set of tripodes with order n, which were exhibited by Barth, Baudon and Puech [11] and Ravaux [104], and split graphs, which was exhibited by Broesma, Kratsch and Woeginger [39].

Theorem 2.13 ([11] and [104], respectively)**.** *For every $n \geq 1$, the sets*

$$K_{\mathcal{T}}(n) = \{\pi : \|\pi\| = n \text{ and } (\pi = (k, k, ..., k, r) \text{ or } \pi = (k, k, ..., k, k+1, k+1, ..., k+1, r))\}$$

and

$$K'_{\mathcal{T}}(n) = \{\pi : \|\pi\| = n \text{ and } |sp(\pi)| \leq 7\},$$

are polynomial kernels for $\mathcal{T}(n)$, where $k \leq n-1$ and $r < k$.

Theorem 2.14 ([39])**.** *For every $n \geq 1$, the set*

$$K_{\mathcal{S}}(n) = \{\pi : \|\pi\| = n \text{ and } (\pi = (1, 3, 3, ..., 3) \text{ or } sp(\pi) = \{2, 3\})\}$$

is a polynomial kernel for $\mathcal{S}(n)$.

Ravaux also considered the existence of polynomial kernels for general trees in [104, 105]. In particular, he proved the following.

Theorem 2.15 ([104, 105])**.** *The set*

$$\{\pi : \|\pi\| = n \text{ and } |sp(\pi)| < \alpha\}$$

is a polynomial kernel for trees with order n and diameter $n - \alpha$.

In this scope, Ravaux addressed the following conjecture in [104] (which was also formulated by Barth, Fournier and Ravaux in [13]).

Conjecture 2.16 ([13, 104])**.** *There is a polynomial kernel with size $\mathcal{O}(n^{\mathcal{O}(\alpha)})$ for combs with order n and α degree-3 nodes.*

Structural properties of arbitrarily partitionable graphs

Trees have been one of the most investigated families of graphs regarding the structure of arbitrarily partitionable graphs, mainly because every graph spanned by an arbitrarily partitionable tree is arbitrarily partitionable itself (see upcoming Observation 2.27), and because they have small connectivity. In this context, the most important results are about the maximum degree of arbitrarily partitionable trees. As a first result, it was proved by Horňák and Woźniak that arbitrarily partitionable trees have maximum degree at most 6 [73]. This result was then lowered to 4 by Barth and Fournier, this upper bound being shown to be tight [12].

Theorem 2.17 ([12]). *Arbitrarily partitionable trees have maximum degree at most* 4.

Additional results regarding the structure of arbitrarily partitionable trees were also exhibited by Barth and Fournier in [12]. At first, they pointed out that every 4-node in an arbitrarily partitionable tree is adjacent to a leaf. They also showed that the number of arbitrarily partitionable trees with maximum degree 3 or 4 is not finite. These results were later completed by Barth, Fournier and Ravaux in [13], wherein the shape of arbitrarily partitionable trees with maximum degree 3 is characterized.

Theorem 2.18 ([13]). *Every arbitrarily partitionable tree with maximum degree* 3 *is a comb.*

One important reason for investigating more constrained versions of the original definition of arbitrarily partitionable graphs is that every graph which is "more than arbitrarily partitionable" (i.e. on-line or recursively arbitrarily partitionable) is also arbitrarily partitionable. Actually the following hierarchy was proved by Baudon, Gilbert and Woźniak in [24], where $PM(n)$ denotes the sets of graphs with order n which admit a perfect matching, $AP(n)$ denotes the set of arbitrarily partitionable graphs with order n, $OLAP(n)$ and $RAP(n)$ respectively denote the sets of on-line and recursively arbitrarily partitionable graphs with order n, and $Tr(n)$ denotes the set of traceable graphs with order n.

Theorem 2.19 ([24]). *For every $n \geq 1$, we have*

$$PM(n) \supset AP(n) \supset OLAP(n) \supset RAP(n) \supset Tr(n).$$

Concrete characterizations of some families of arbitrarily partitionable caterpillars were exhibited by Cichacz, Görlich, Marczyk, Przybyło and Woźniak in [43]. On-line and recursively arbitrarily partitionable trees have a much more restricted structure than arbitrarily partitionable trees, as shown by Horňák, Tuza and Woźniak in [72], and Baudon, Gilbert and Woźniak in [24], respectively. We gather the two resulting characterizations within the next result.

Theorem 2.20 ([72, 24]). *A tree T is on-line or recursively arbitrarily partitionable if and only if T is either a path, the tripode $P_3(2,4,6)$, or one caterpillar $Cat(a,b)$ with a and b taking values in Table 2.2.a or Table 2.2.b, respectively.*

One motivation for studying arbitrarily partitionable multipodes is that some of their structural properties can be derived to structural properties of arbitrarily partitionable graphs which have a cut vertex. Assume indeed that a multipode $G = P_k(a_1, a_2, ..., a_k)$ with order n does not admit a realization of an n-sequence π. Then every other graph with order n made up of one cut vertex whose removal results in k components with order $a_1, a_2, ..., a_k$, respectively, cannot admit a realization of π since otherwise we could deduce a realization of π in G. This argument extends to the study of (possibly on-line or recursively) arbitrarily partitionable graphs which have a cutset with size k and have convenient local partition properties.

Observation 2.21. *Let $G = C_{k,\ell}(K_{n_1}, K_{n_2}, ..., K_{n_\ell})$ be a (k,ℓ)-compound graph where the ℓ components are complete. If a $|V(G)|$-sequence π is not realizable in G, then π is not realizable in every graph G' with a k-cutset whose removal results in ℓ components with order $n_1 - k, n_2 - k, ..., n_\ell - k$.*

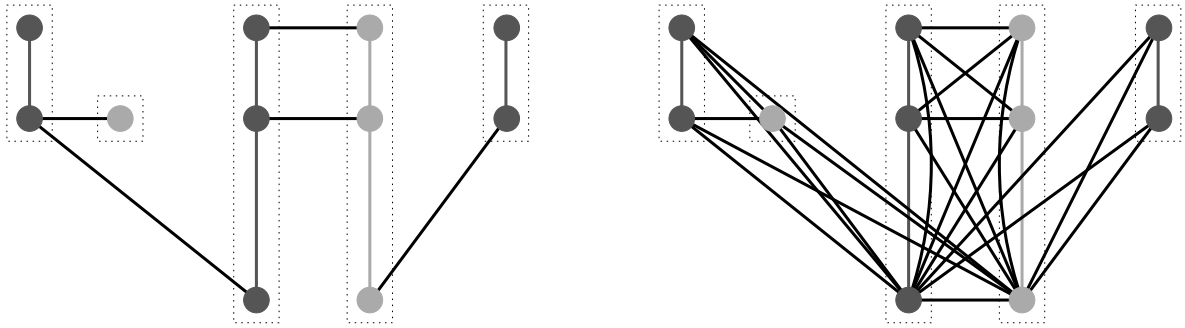
(a) $Cat(a, b)$ is on-line arbitrarily partitionable.

a	b
2, 4	$\equiv 1 \pmod 2$
3	$\equiv 1, 2 \pmod 3$
5	6, 7, 9, 11, 14, 19
6	$\equiv 1, 5 \pmod 6$
7	8, 9, 11, 13, 15
8	11, 19
9, 10	11
11	12

(b) $Cat(a, b)$ is recursively arbitrarily partitionable.

a	b
2, 4	$\equiv 1 \pmod 2$
3	$\equiv 1, 2 \pmod 3$
5	6, 7, 9, 11, 14, 19
6	7
7	8, 9, 11, 13, 15

Table 2.2: Values of a and b, with $b \geq a$, for which $Cat(a, b)$ is on-line (a) or recursively (b) arbitrarily partitionable.



(a) A realization of a sequence in a graph.

(b) The same realization in its maximum supergraph $C_{2,3}(K_5, K_6, K_4)$ with the same structure.

Figure 2.3: Illustration of Observation 2.21.

Observation 2.21 follows from the fact that G' is a spanning subgraph of G, so every realization of π in G' is also a realization of π in G.

Example 2.22. Observation 2.21 is illustrated in Figure 2.3. A realization of the sequence $(1, 2, 2, 3, 3)$ in the graph G from Figure 2.3.a also forms a realization of $(1, 2, 2, 3, 3)$ in the maximum supergraph of G with the same structure, see Figure 2.3.b. This supergraph is a compound graph where all components are complete.

Observation 2.21 was notably considered by Baudon, Foucaud, Przybyło and Woźniak in [22] (although stated differently), wherein properties of the components resulting from the removal of a cutset with size k from an arbitrarily partitionable graph are exhibited. In particular, they proved the following.

Theorem 2.23 ([22])**.** *Removing a cutset with size $k \geq 2$ from an arbitrarily partitionable graph can result in arbitrarily many components, but their orders follow an exponential growth. In particular, if these orders are $n_1, n_2, ..., n_\ell$, with $n_1 \leq n_2 \leq ... \leq n_\ell$, then we have*

$$n_i \geq \frac{1}{k} \cdot \sum_{j=1}^{i-1} n_j$$

for every $i \in \{2, 3, ..., \ell\}$.

Theorem 2.23 is also related to the following theorem by Tutte [117] which concerns the realizability of the sequence $(2, 2, ..., 2)$ only.

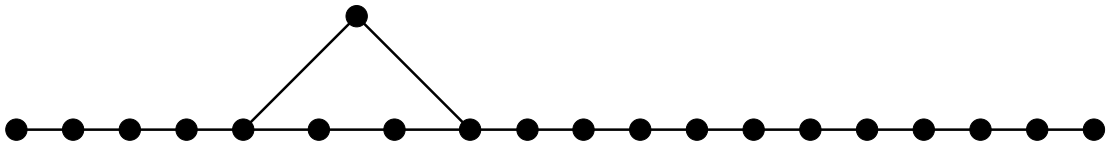


Figure 2.4: A minimal arbitrarily partitionable non-tree graph.

Theorem 2.24 ([117]). *A graph G has a perfect matching if and only if, for every subset $S \subset V(G)$, removing S from G results in at most $|S|$ components with odd order.*

This approach can also be adapted to deduce structural properties of on-line or recursively arbitrarily partitionable graphs which have a cutset with given size. In particular, since balloons form a family of compound graphs, the following can be deduced as an immediate corollary of a property of recursively arbitrarily partitionable balloons proved by Baudon, Gilbert and Woźniak in [24].

Corollary 2.25 ([24]). *Removing a 2-cutset from a recursively arbitrarily partitionable graph results in at most five components.*

Corollary 2.25 contrasts with Theorem 2.23 since removing a cutset with size 2 from an arbitrarily partitionable graph can result in arbitrarily many components.

Studies dedicated to the class of partitionable suns may also be found in the literature. In particular, we refer the reader to works of Kalinowski, Pilśniak, Woźniak and Zioło [75, 76] and of Baudon, Gilbert and Woźniak [23] who gave full characterizations of on-line and recursively, respectively, arbitrarily partitionable suns.

It is also worth mentioning that another stronger version of recursively arbitrarily partitionable graphs was considered by Baudon, Gilbert and Woźniak in [24]. This is defined as follows.

Definition 2.26. A graph G is *strongly recursively arbitrarily partitionable* if either

- $G \simeq K_1$, or
- for every $|V(G)|$-sequence $\pi = (n_1, n_2, ..., n_p)$ and every realization $(V_1, V_2, ..., V_p)$ of π in G, the graphs $G[V_1], G[V_2], ..., G[V_p]$ are strongly recursively arbitrarily partitionable.

Baudon, Gilbert and Woźniak proved that strongly recursively arbitrarily partitionable graphs are claw- and net-free [24], and hence traceable according to a result of Duffus, Gould and Jacobson [53].

Minimal arbitrarily partitionable graphs

Since adding an edge to a graph cannot decrease its connectivity, adding arbitrarily many edges to a yet arbitrarily partitionable graph yields another arbitrarily partitionable graph. Equivalently, every graph which is spanned by an arbitrarily partitionable graph is arbitrarily partitionable itself.

Observation 2.27. *Every graph spanned by a graph with some partition properties has the same partition properties.*

From this easy observation, and since every arbitrarily partitionable tree is minimal (with regards to the property of being arbitrarily partitionable), one could think that perhaps every arbitrarily partitionable graph is spanned by an arbitrarily partitionable tree. But this claim was shown to be false with several authors exhibiting minimal arbitrarily partitionable graphs which are not trees. For such counterexamples, we refer the reader to works of Baudon, Gilbert and Woźniak [24], and of Ravaux [104].

Example 2.28. The minimal arbitrarily partitionable non-tree graph exhibited by Ravaux in [104] is drawn in Figure 2.4.

Minimal arbitrarily partitionable graphs were mainly studied by Ravaux in [104], and Baudon, Przybyło and Woźniak in [25], though still little is known about these graphs so far. Ravaux mainly considered the maximum degree of minimal arbitrarily partitionable graphs and notably showed the following [104].

Theorem 2.29 ([104])**.** *For every minimal arbitrarily partitionable graph G with order $n \geq 4$, we have $\Delta(G) \leq n - 2$.*

Speaking of maximum degree, it is worth mentioning that there does not exist an absolute constant c such that minimal arbitrarily partitionable graphs have maximum degree at most c. This was raised by Baudon, Przybyło and Woźniak, who pointed out in [25] that we cannot remove too many edges from an arbitrarily partitionable k-balloon with large maximum degree (such balloons exist according to Theorem 2.23) without loosing the property of being arbitrarily partitionable.

Observation 2.30 ([25])**.** *Minimal arbitrarily partitionable graphs can have arbitrary large maximum degree.*

One feeling regarding minimal arbitrarily partitionable graphs is that they should be sparse since otherwise we could easily find edges which are "useless" for partitioning them. So the following conjecture was raised by Ravaux [104].

Conjecture 2.31 ([104])**.** *Every minimal arbitrarily partitionable graph with order n has size $\mathcal{O}(n)$.*

Because all known minimal arbitrarily partitionable graphs have connectivity 1 and the density of a graph increases with its connectivity, Ravaux legitimately addressed in [104] the following question related to Conjecture 2.31.

Question 2.32 ([104])**.** *Are there minimal arbitrarily partitionable graphs with arbitrary large connectivity?*

Towards Conjecture 2.31, Baudon, Przybyło and Woźniak focused on the density of minimal arbitrarily partitionable graphs. In [25], they exhibited a family of minimal arbitrarily partitionable graphs where every member with order n has size $\lceil \frac{31n}{30} \rceil$. These are the densest minimal arbitrarily partitionable graphs known so far. As other properties, it is also worth mentioning that these minimal arbitrarily partitionable graphs can have arbitrarily large girth and arbitrarily many vertex-disjoint cycles.

Theorem 2.33 ([25])**.** *Minimal arbitrarily partitionable graphs can have arbitrarily many vertex-disjoint cycles with arbitrarily large length.*

Arbitrarily partitionable graphs as "almost" Hamiltonian graphs

Every path is obviously arbitrarily partitionable. Following Observation 2.27, we then directly derive the following (which also follows from Theorem 2.19).

Observation 2.34. *Every traceable graph is arbitrarily partitionable.*

Since every traceable, and hence Hamiltonian, graph is arbitrarily partitionable, the property of being arbitrarily partitionable can be seen as a weakening of the property of being Hamiltonian. It is then natural to investigate how a classic sufficient condition for being Hamiltonian can be weakened to an analogous sufficient condition for being arbitrarily partitionable. Consider in particular the following parameter.

Definition 2.35. For every graph G and integer $k \geq 2$, the parameter $\sigma_k(G)$ is defined as

$$\sigma_k(G) = \min\{d(v_1) + d(v_2) + ... + d(v_k) : v_1, v_2, ..., v_k \text{ are non-adjacent vertices of } G\}.$$

The σ_k parameters have been widely considered in the literature to exhibit sufficient conditions for Hamiltonicity, refer for instance to [65] for a survey on this topic by Gould. Adapting such sufficient conditions to arbitrarily partitionable graphs was first considered by Marczyk, who proved in [91] that every graph G on $n \geq 8$ vertices satisfying $\sigma_2(G) \geq n - 3$ is arbitrarily partitionable if and only if G has a perfect matching (or quasi perfect matching if n is odd). Clearly this result is nothing but a weakening of the well-known Ore's result stating that every graph G on $n \geq 3$ vertices satisfying $\sigma_2(G) \geq n$ is Hamiltonian [99].

The result by Marczyk was later strengthened by Horňák, Marczyk, Schiermeyer and Woźniak in the following way [71]

Theorem 2.36 ([71])**.** *Every graph G on $n \geq 20$ vertices satisfying $\sigma_2(G) \geq n - 5$ is arbitrarily partitionable if and only if G has a perfect matching (or quasi perfect matching if n is odd).*

The relationship between arbitrarily partitionable graphs and σ_3 was considered by Brandt, who proved that every graph G on n vertices satisfying $\sigma_3(G) \geq n$ is arbitrarily partitionable if and only if G admits a perfect matching (or a quasi perfect matching if n is odd) [38]. Relating the property of being arbitrarily partitionable and σ_k parameters for $k \geq 4$ has not been done yet.

Another possible direction for weakening results on Hamiltonian graphs was briefly mentioned in [49], wherein Diwan raised the following.

Conjecture 2.37. *For every $k \geq 3$, all k-connected k-regular graphs are arbitrarily partitionable.*

The counterpart of Conjecture 2.37 for traceable and Hamiltonian graphs was disproved for $k = 3$ by Garey, Johnson and Tarjan, who proved that Hamiltonian Path is NP-complete when restricted to 3-connected 3-regular graphs [63]. Later on, Czumaj and Strothmann generalized this result to all $k \geq 3$ [45]. No results towards Conjecture 2.37 are known so far.

2.4 Contributions of Part I

Chapter 3: Arbitrarily partitionable graphs

Chapter 3 is dedicated to the study of the ordinary notion of arbitrarily partitionable graphs, that is in which no additional constraint is requested. Our concerns in this scope are mostly algorithmic and, hence, related to Realizable Sequence and Arbitrarily Partitionable Graph. Throughout Section 3.1, we exhibit new restrictions on both the sequence structure (see Section 3.1.1) and the graph structure (see Section 3.1.2) under which Realizable Sequence remains NP-complete. These restrictions are new in the sense that they are not caught by Theorem 2.10. As a side result, we also prove the tightness of Theorem 2.1 in Section 3.1.3.

We then consider the complexity of Arbitrarily Partitionable Graph. In Section 3.2, we discuss about the Π_2^p-completeness of Arbitrarily Partitionable Graph, and mainly show that a specific problem related to our concerns is Π_2^p-complete. In Section 3.3, we show that Arbitrarily Partitionable Graph is in NP for several classes of graphs, namely complete multipartite graphs (Section 3.3.1), graphs with about one half universal vertices (Section 3.3.2), and graphs made up of components with fair partition properties (Section 3.3.3). This is done by exhibiting new polynomial kernels for these families of graphs. These results support the prevailing feeling that Arbitrarily Partitionable Graph should be an NP problem, recall Conjecture 2.12.

Various results are exhibited in the rest of Chapter 3. Some results on minimal arbitrarily partitionable graphs are first gathered in Section 3.4, wherein are exhibited small minimal arbitrarily partitionable non-tree graphs (see Section 3.4.1) and an improvement of Theorem 2.29.

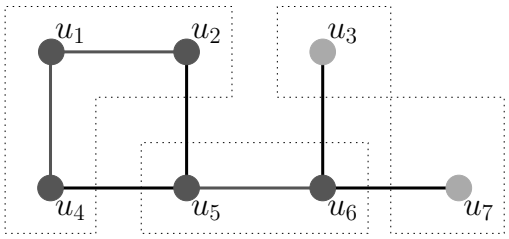


Figure 2.5: A graph which cannot be partitioned under specific vertex-membership constraints.

We then consider Cartesian product of arbitrarily partitionable graphs in Section 3.5. As a main result, we get that every Cartesian product $G \square H$, involving two arbitrarily partitionable graphs G and H, is arbitrarily partitionable whenever $|V(H)| \leq 4$.

Chapter 4: Preassignable arbitrarily partitionable graphs

Regarding the practical problem introduced in Section 2.1, one could consider the following stronger situation. Suppose that we still want to satisfy the p users requesting some resources of our network, but that k of these users are, for some reason, each allowed to additionally request one specific resource to belong to his attributed subnetwork.

In order to deal with this stronger problem using graph theory, we need the following definitions.

Definition 2.38. A *k-preassignation* of a graph G is a k-tuple $(v_1, v_2, ..., v_k)$ of k distinct vertices of G. The vertices $v_1, v_2, ..., v_k$ are then said to be *preassigned*.

Definition 2.39. Let G be a graph, $\pi = (n_1, n_2, ..., n_p)$ be a $|V(G)|$-sequence, and $P = (v_1, v_2, ..., v_k)$ be a k-preassignation of G with $k \leq p$. We say that π is *P-realizable* in G if there is a realization $(V_1, V_2, ..., V_p)$ of π in G satisfying

$$v_1 \in V_1, v_2 \in V_2, ..., v_k \in V_k.$$

Note that, in Definition 2.39, we have adopted the convention that the k preassigned vertices are intended to belong to the parts of the realization whose sizes are the k first values of the sequence. So assuming that the k privileged users' needs are $n_1, n_2, ..., n_k$ and the specific resources they request are those labelled $v_1, v_2, ..., v_k$, respectively, in the graph G underlying our network, we get that we can satisfy the augmented resource demand if and only if $(n_1, n_2, ..., n_p)$ is $(v_1, v_2, ..., v_k)$-realizable in G.

Example 2.40. The realization drawn on Figure 2.1.a of the sequence $(2, 2, 3)$ in a graph G is also a (u_6)-realization of $(3, 2, 2)$ in G since u_6 belongs to the part of size 3. However G does not admit any (u_6)-realization of $(2, 2, 3)$ since, for every connected part V_1 with size 2 including u_6, the subgraph $G - V_1$ does not admit any realization of $(2, 3)$, see Figure 2.5.

Assuming, as for the initial practical problem, that we do not know either how many users will ask for resources of our network or how many resources they will request, but we know the number of special users who will be allowed to each request a specific resource, as network managers we would like the graph modelling our network to be *preassignable arbitrarily partitionable*.

Definition 2.41. Let P be a k-preassignation of a graph G. We call G *arbitrarily P-partitionable* if, for every k'-preassignation P' of G with $P' \subseteq P$, every $|V(G)|$-sequence with size at least k' is P'-realizable in G. We say that G is *k-preassignable arbitrarily partitionable* if G is arbitrarily P-partitionable for every k-preassignation P of G.

The notion of graph partition satisfying vertex-assignation constraints is inspired by the vertex-membership requirement from Theorem 2.1. To the best of our knowledge, this notion

was first considered in the context of arbitrarily partitionable graphs by Diwan and Kurhekar in [50] under the name of *partitionable graphs with basis*, though they introduced this notion for a specific purpose and did not investigate elementary properties of k-preassignable arbitrarily partitionable graphs.

We focus on preassignable arbitrarily partitionable graphs in Chapter 4. In Section 4.1, we give very first results regarding these graphs. We then mainly consider the structural aspect. By studying several classes of graphs (including powers of cycles and Harary graphs) in Sections 4.2 and 4.3, we exhibit a tight lower bound on the size of k-preassignable arbitrarily partitionable graphs. More precisely, we show that, for every $k \geq 1$ and $n \geq k+1$, every k-preassignable arbitrarily partitionable graph on n vertices cannot have less than $\lceil \frac{n(k+1)}{2} \rceil$ edges, and that there exist k-preassignable arbitrarily partitionable graphs with order n and exactly this many edges.

Clearly, the property of being preassignable arbitrarily partitionable requires for a graph to be dense enough. Actually it turns out that most of the graphs we consider to prove the previously mentioned results have some convenient Hamiltonian properties making the checking process easier. In Section 4.4, we thus consider the relationship between the properties of being preassignable arbitrarily partitionable and Hamiltonian. As a main result, we get that there is no systematic Hamiltonian property appearing in a preassignable arbitrarily partitionable graph. For this purpose, we show that the longest path of a k-preassignable arbitrarily partitionable graph G can be arbitrarily smaller than $|V(G)|$.

As for arbitrarily partitionable graphs, we also consider, in Section 4.5, the question asking whether every Cartesian product involving a k-preassignable arbitrarily partitionable graph is k-preassignable arbitrarily partitionable itself. Our main result in this scope states that $G \,\square\, P_\ell$ is 1-preassignable arbitrarily partitionable whenever G is 1-preassignable arbitrarily partitionable.

Chapter 5: On-line and recursively arbitrarily partitionable graphs

Chapter 5 is dedicated to the study of on-line or recursively arbitrarily partitionable graphs. After having given some elementary properties of these graphs in introductory Section 5.1, we establish, in Section 5.2, the membership to PSPACE of the two decision problems below.

On-Line Arbitrarily Partitionable Graph
Instance: a graph G.
Question: is G on-line arbitrarily partitionable?

Recursively Arbitrarily Partitionable Graph
Instance: a graph G.
Question: is G recursively arbitrarily partitionable?

Chapter 5 is then dedicated to two main series of results regarding the structure of on-line and recursively arbitrarily partitionable graphs.

Our main objective, in Sections 5.3 and 5.4, is to make a first step towards an equivalent of Theorem 2.23 for on-line or recursively arbitrarily partitionable graphs, which could be highly different as suggested by Theorem 2.25. In this scope, we start, in Section 5.3, by showing that removing a k-cutset from a recursively arbitrarily partitionable graph cannot yield more than $4k-1$ components. This result is not tight, but confirms the fact that recursively arbitrarily partitionable graphs have a more constrained structure than arbitrarily partitionable graphs (arbitrarily many such components may result from the deletion when dealing with an arbitrarily partitionable graph). We then focus, in Section 5.4, on the orders of the components resulting from the removal of a 2-cutset from an on-line or recursively arbitrarily partitionable graph. As a main result, we obtain that some of these components must be small, i.e. with order upper-bounded by a constant.

We then investigate the relationship between the properties of being recursively arbitrarily partitionable and Hamiltonian, which empirically seemed to be somehow related. In Section 5.5, we exhibit two constructions providing recursively arbitrarily partitionable graphs whose longest paths are arbitrarily smaller than their orders, hence proving that recursively arbitrarily partitionable graphs (and hence on-line arbitrarily partitionable graphs, recall Theorem 2.19) can be, in some sense, arbitrarily not Hamiltonian.

Chapter 3
Arbitrarily partitionable graphs

This chapter is dedicated to the study of the classic notion of arbitrarily partitionable graphs. We mainly investigate both the positive and negative algorithmic aspects related to it. Regarding the negative aspects, we start, in Section 3.1, by exhibiting several conditions concerning the sequence π (Section 3.1.1) or the graph G (Section 3.1.2) under which Realizable Sequence remains NP-complete. In the same vein, we also show, in Section 3.1.3, that Theorem 2.1 is tight in the sense that deciding whether a k-connected graph is partitionable into at least $k+1$ connected subgraphs (with possibly some of them including preassigned vertices) is NP-complete in general. We finally discuss the Π_2^p-completeness of Arbitrarily Partitionable Graph in Section 3.2, wherein we show that graph partition problems can be Π_2^p-complete. As positive results, we exhibit, in Section 3.3, new polynomial kernel of sequences for several classes of graphs. These imply the membership to NP of several restrictions of Arbitrarily Partitionable Graph. As a side result, one of these kernels yields a polynomial-time algorithm for deciding whether a complete multipartite graph is arbitrarily partitionable, see Section 3.3.1.

The algorithmic point of view apart, we then investigate the structure of minimal arbitrarily partitionable graphs in Section 3.4. In this scope, we exhibit two small minimal arbitrarily partitionable non-tree graphs in Section 3.4.1, as well as a slight improvement of Theorem 2.29 in Section 3.4.2.

We finally address a conjecture concerning the Cartesian product of two arbitrarily partitionable graphs in Section 3.5, see Conjecture 3.44. As a first step towards this conjecture, our main result states that $G \,\square\, H$ is arbitrarily partitionable whenever G and H are arbitrarily partitionable and H has order at most 4.

Most of the results from Sections 3.1.1, 3.1.3, and 3.2 are about to be published [30]. Our results from Section 3.3 and Theorem 3.18 are part of an article submitted for publication [33]. The results presented in Section 3.5 are part of a published joint work with Baudon, Kalinowski, Marczyk, Przybyło and Woźniak [16].

3.1 On the NP-completeness of Realizable Sequence

We herein exhibit new conditions on π or G under which Realizable Sequence (or some derived problems) remains NP-complete. To that end, we introduce two polynomial reductions from known NP-complete problems to Realizable Sequence, and then modify these reductions so that desired properties on π or G appear in a reduced instance. Our first reduction in Section 3.1.1 is from 1-in-3 Satisfiability and is modified to deduce NP-complete restrictions of Realizable Sequence related to π. Our second reduction in Section 3.1.2 is from 3-Partition and is modified to obtain reduced instances of Realizable Sequence in which G has a specific structure. We finally investigate the tightness of Győri-Lovász Theorem (Theorem 2.1) in Section 3.1.3.

Recall that Realizable Sequence is already known to be NP-complete, see Theorem 2.10, and hence is in NP by definition. Given an instance $<G, \pi>$ of Realizable Sequence, one can basically provide a partition of $V(G)$ to an algorithm checking that the number of parts and their sizes agree with π, and that the induced subgraphs these parts induce are connected. Since checking the connectedness of a graph can be done in polynomial time, such a checking algorithm runs in polynomial time. As this statement depends neither on the structure of G nor on the elements in π, this implies the membership to NP of all restrictions of Realizable Sequence we consider throughout. Therefore we only focus on the NP-hardness of restricted versions of Realizable Sequence, but their membership to NP is understood. We also sometimes voluntarily omit to mention the fact that our reductions are performed in polynomial time, as this fact can be checked easily.

3.1.1 Restrictions on the sequence

3.1.1.1 Sequences with fixed size

We investigate the consequences on Realizable Sequence of fixing the size of π as a constant. Said differently we are interested in the following refined decision problem.

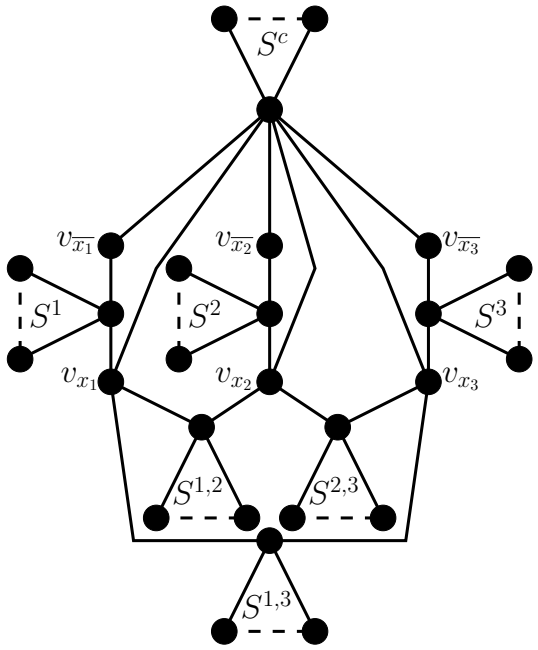
Realizable Size-k Sequence
Instance: A graph G and a $|V(G)|$-sequence π with size k.
Question: Is π realizable in G?

Since the $|V(G)|$-sequence $\pi = (|V(G)|)$ is realizable in G if and only if G is connected, the problem Realizable Size-1 Sequence is in P. We show below that Realizable Size-k Sequence is NP-complete for every $k \geq 2$, and hence that Realizable Sequence should not be fixed-parameter tractable when parameterized by $|\pi|$. This result is shown in two steps. We first show, in Theorem 3.1, that Realizable Size-2 Sequence is NP-complete by reduction from 1-in-3 Satisfiability. We then explain, in Theorem 3.2, how to modify our reduction from 1-in-3 Satisfiability to Realizable Size-2 Sequence so that we get a reduction from 1-in-3 Satisfiability to Realizable Size-k Sequence for any $k \geq 3$.

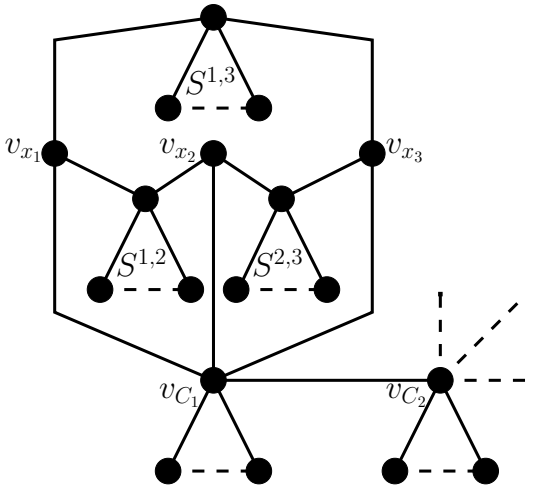
Theorem 3.1. Realizable Size-2 Sequence *is NP-complete.*

Proof. We prove that Realizable Size-2 Sequence is NP-hard by reduction from 1-in-3 Satisfiability. From a given formula F in conjunctive normal form whose clauses include three literals, we construct a graph G_F and a $|V(G_F)|$-sequence π_F with size 2 such that

$$F \text{ is 1-in-3 satisfiable}$$
$$\Leftrightarrow$$
$$\pi_F \text{ is realizable in } G_F.$$



(a) Resulting subgraph in the clause subgraph for a clause $(x_1 \vee x_2 \vee x_3)$ of F.



(b) Connection between the base and clause subgraphs for a clause $C_1 = (x_1 \vee x_2 \vee x_3)$ of F.

Figure 3.1: Construction of the reduced graph G_F.

Recall that we may suppose that all literals over the variables of F appear in F, refer to Observation 1.44. Recall further that F has $2n$ literals and m clauses.

The graph G_F is made up of two main vertex-disjoint subgraphs. The first one is the *clause subgraph*. Each literal ℓ_i of F is associated with a *literal vertex* v_{ℓ_i} in the clause subgraph. For each pair $\{\ell_i, \overline{\ell_i}\}$ of negated literals ℓ_i and $\overline{\ell_i}$ of F, we then link the literal vertices v_{ℓ_i} and $v_{\overline{\ell_i}}$ to the root vertex of a star S^i with n vertices of degree 1. Two literal vertices v_{ℓ_i} and v_{ℓ_j} such that $\ell_j \neq \overline{\ell_i}$ are similarly linked to the root vertex of a star $S^{i,j}$ with n vertices of degree 1 if they appear in a same clause of F. We finally add a *connecting star* S^c with n vertices of degree 1 to the clause subgraph of G_F and link its root to every literal vertex so that the clause subgraph is connected.

The construction so far is detailed in Figure 3.1.a. Let n_2 be the number of vertices of the clause subgraph. Then we have

$$n_2 \leq 2n + n(n+1) + 3m(n+1) + n + 1$$

since there are exactly $2n$ literals and n pairs of literals of the form $\{\ell_i, \overline{\ell_i}\}$ in F, all the clauses of F can have distinct literals, and the connecting star S^c has exactly n vertices of degree 1.

The second subgraph of G_F is the *base subgraph*. With each clause C_j in F we associate a *clause vertex* v_{C_j} in the base subgraph that is linked to $n_2 - n$ vertices of degree 1. For each $j \in \{1, 2, ..., m-1\}$, we finally add the edge $v_{C_j}v_{C_{j+1}}$ to $E(G_F)$ so that the clause vertices induce a path in G_F. If we denote by n_1 the number of vertices of the base subgraph of G_F, then we have

$$n_1 = m(n_2 - n + 1).$$

We end up the construction of G_F by adding the following edges between the base and clause subgraphs of G_F: for each clause $C_j = (\ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3})$ in F, we just add $v_{C_j}v_{\ell_{i_1}}$, $v_{C_j}v_{\ell_{i_2}}$, and $v_{C_j}v_{\ell_{i_3}}$ to $E(G_F)$. This connection is illustrated in Figure 3.1.b.

The number of vertices of G_F is $n_1 + n_2$, and hence the construction of G_F is performed in polynomial time regarding the size of F. Consider now the sequence $\pi_F = (n_1 + n, n_2 - n)$.

Since the two elements of π_F are strictly greater than 1, every part U from a realization R of π_F in G_F covering the root vertex of a star subgraph must also contain all the vertices of degree 1 attached to it. Indeed, if this were not the case, then the graph $G_F - U$ would contain at least two components and, thus, the part of R different from U could not induce a connected subgraph of G_F.

For this reason, observe that, because of all the induced stars S_{n_2-n+1} in the base subgraph of G_F, this subgraph must be covered by the part V_1 with size $n_1 + n$ of a realization (V_1, V_2) of π_F in G_F. Starting from this, we then have to add n additional vertices from the clause subgraph of G_F to V_1. For a similar reason as the one above, we can only pick up some literal vertices of G_F since picking up any other vertex would disconnect G_F into too many small components. According to our construction, we also cannot add to V_1 two literal vertices v_{ℓ_i} and v_{ℓ_j} such that ℓ_i and ℓ_j are negated literals, or appear in a same clause of F, since otherwise this would once again make the subgraph $G_F - V_1 = G_F[V_2]$ disconnected.

We can then deduce a 1-in-3 truth assignment of the variables of F from a realization $R = (V_1, V_2)$ of π_F in G_F and vice-versa. If R is a correct realization of π_F in G_F, then there are exactly n literal vertices $v_{\ell_{i_1}}, v_{\ell_{i_2}}, ..., v_{\ell_{i_n}}$ from the clause subgraph of G_F which belong to V_1. Since $G_F[V_2]$ is connected, setting the literals $\ell_{i_1}, \ell_{i_2}, ..., \ell_{i_n}$ true makes F evaluated true in a 1-in-3 way since no pair of these literals is a literal of F and its negation or appears in a same clause of F. Conversely, if F is 1-in-3 satisfiable, then let $\phi : \{\ell_1, \ell_2, ..., \ell_{2n}\} \to \{0, 1\}$ be a satisfying 1-in-3 truth assignment of its literals. Then observe that (V_1, V_2), where

- V_1 contains all the vertices from the base subgraph of G_F and every literal vertex v_{ℓ_i} from the clause subgraph of G_F such that $\phi(\ell_i) = 1$,
- $V_2 = V(G_F) \setminus V_1$,

is a correct realization of π_F in G_F according to the arguments above. ■

We now explain how to generalize the reduction from the proof of Theorem 3.1 so that we get a reduction from 1-in-3 Satisfiability to Realizable Size-k Sequence for any $k \geq 3$.

Theorem 3.2. *For every $k \geq 2$,* Realizable Size-k Sequence *is NP-complete.*

Proof. The proof that Realizable Size-k Sequence is NP-hard for a fixed value of $k \geq 3$ is based on our reduction from 1-in-3 Satisfiability to Realizable Size-2 Sequence (proof of Theorem 3.1). More precisely, we modify the instance resulting from the reduction, i.e. the graph G_F and the sequence π_F, so that $|\pi_F| = k$ and the arguments given in the proof of Theorem 3.1 are still correct and not altered by the modifications.

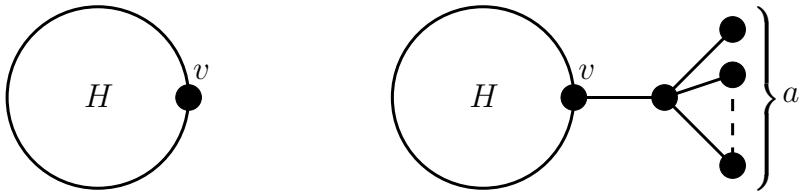
So that we introduce these modifications, we beforehand need to introduce the following graph construction.

Construction 3.3. Given a graph H, a vertex $v \in V(H)$, and an arbitrary integer $a \geq 3$, the (a, v)-*star augmentation* of H is the graph obtained as follows:

1. consider the disjoint union of H and the star S_a with $a - 1$ vertices of degree 1,
2. add an edge between v and the root of S_a.

This construction is illustrated in Figure 3.2. We first show that Realizable Size-3 Sequence is NP-hard by reduction from 1-in-3 Satisfiability before generalizing our arguments. From a formula F in conjunctive normal form involving 3-clauses, we construct a graph G_F and a $|V(G_F)|$-sequence $\pi_F = (n_1, n_2, n_3)$ with size 3 such that

$$F \text{ is 1-in-3 satisfiable}$$
$$\Leftrightarrow$$
$$\pi_F \text{ is realizable in } G_F.$$



(a) A graph H with a vertex v.

(b) An arbitrary (a, v)-star augmentation of H.

Figure 3.2: Illustration of the star augmentation operation.

By applying the reduction from 1-in-3 Satisfiability to Realizable Size-2 Sequence from the proof of Theorem 3.1, we get a graph G'_F and a $|V(G'_F)|$-sequence $\pi'_F = (n'_1, n'_2)$ with size 2 which is realizable in G'_F if and only if F is 1-in-3 satisfiable. Besides, recall that $n'_1, n'_2 \geq 2$. Now consider, as G_F, an (a, v)-star augmentation of G'_F where $a = n'_1 + n'_2 + 1$ and $v \in V(G'_F)$ is arbitrary, and set $\pi_F = (a, n'_1, n'_2)$. In a realization (U, V_1, V_2) of π_F in G_F, note that, because $n'_1, n'_2 \geq 2$, the star subgraph S_a of G_F resulting from the star augmentation must be covered entirely by the part U with size a since covering it with one of the other two parts would disconnect G_F into too many small components. Therefore, π_F is realizable in G_F if and only if π'_F is realizable in G'_F, and by transitivity we get that F is 1-in-3 satisfiable if and only if π_F is realizable in G_F.

One can repeat the previous transformation as many times as wanted until π_F has desired size k. At each step, we get another instance of Realizable Sequence which is equivalent to the previous one but whose sequence has one more element. Namely, from the instance F of 1-in-3 Satisfiability we first construct an equivalent instance of Realizable Size-2 Sequence using the reduction from the proof of Theorem 3.1. From this instance of Realizable Size-2 Sequence is then obtained an equivalent instance of Realizable Size-3 Sequence by performing a star augmentation. Using the same construction, we then get an equivalent instance of Realizable Size-4 Sequence. And so on. All these reduced instances are equivalent to F by transitivity. We thus get that Realizable Size-k Sequence is NP-hard for every $k \geq 3$. ■

3.1.1.2 Sequences with fixed spectrum size

For every $k \geq 2$, note that the elements of the sequence π_F resulting from our reduction from 1-in-3 Satisfiability to Realizable Size-k Sequence (proof of Theorem 3.2) all have distinct values since each of these successively obtained elements is expressed as a summation of the previously obtained elements. Hence not only $|\pi_F| = k$, but we also have $|sp(\pi_F)| = k$. Recall further that Realizable Sequence remains NP-complete when $|sp(\pi)| = 1$, see Theorem 2.10.

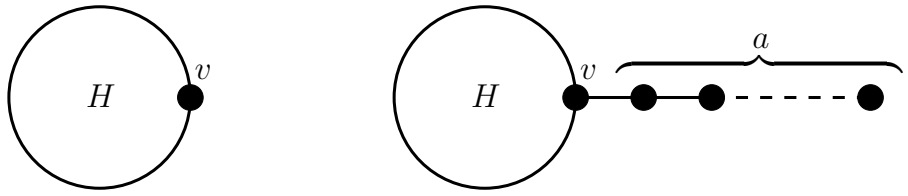
From these two facts, we directly get that Realizable Sequence remains NP-complete under the assumption that $|sp(\pi)| = k$ for every constant integer $k \geq 1$. Consequently Realizable Sequence should not be fixed-parameter tractable when parameterized by $|sp(\pi)|$.

Theorem 3.4. *For every $k \geq 1$,* Realizable Sequence *remains NP-complete when restricted to sequences with spectrum of size k.*

3.1.1.3 Sequences with fixed preassignation size

We now consider the hardness of realizing a sequence in a graph when a preassignation must be respected. We hence focus on the following refined version of Realizable Sequence.

Realizable Sequence with Preassignation
Instance: A graph G, a $|V(G)|$-sequence π, and a preassignation P of G.
Question: Is π P-realizable in G?



(a) A graph H with a vertex v.

(b) An arbitrary (a, v)-path augmentation of H.

Figure 3.3: Illustration of the path augmentation operation.

Similarly as for Realizable Sequence and Realizable Size-k Sequence, the following refinement of Realizable Sequence with Preassignation is more convenient for a sharper study of the restrictions we are interested in.

Realizable Size-k Sequence with k'-Preassignation
Instance: A graph G, a $|V(G)|$-sequence π with size k, and a k'-preassignation P of G.
Question: Is π P-realizable in G?

For every problem Realizable Size-k Sequence with k'-Preassignation, we have $k \geq k'$ by definition. Besides, Realizable Size-k Sequence with 0-Preassignation is equivalent to Realizable Size-k Sequence which was shown to be in P for $k = 1$, and NP-complete for every $k \geq 2$ in previous Section 3.1.1.1. Note further that an instance of Realizable Size-1 Sequence with 1-Preassignation is positive if and only if G is connected. Therefore Realizable Size-1 Sequence with 1-Preassignation is in P.

We now prove that the remaining problems, i.e. Realizable Size-k Sequence with k'-Preassignation with $k \geq 2$ and $1 \leq k' \leq k$, are all NP-complete. As a consequence, we directly get that Realizable Size-k Sequence with k'-Preassignation should not be fixed-parameter tractable when parameterized by both k and k'.

Theorem 3.5. *For every $k \geq 2$ and $k' \in \{0, 1, 2, ..., k\}$,* Realizable Size-$k$ Sequence with k'-Preassignation *is NP-complete.*

Proof. Note first that every polynomial-time checking algorithm for Realizable Sequence can be modified so that it still runs in polynomial time but also checks whether the input realization of π in G respects a preassignation. Therefore Realizable Sequence with Preassignation is in NP.

Let $k \geq 2$ and $k' \in \{0, 1, 2, ..., k\}$ be fixed. Recall that if $k' = 0$, then Realizable Size-k Sequence with k'-Preassignation is nothing but Realizable Size-k Sequence, which is NP-hard by Theorem 3.1. Suppose thus that $k' \geq 1$. We show that Realizable Size-k Sequence with k'-Preassignation is NP-hard by using our reduction from 1-in-3 Satisfiability to Realizable Size-k Sequence (proof of Theorem 3.2) and the following construction.

Construction 3.6. Given a graph H, a vertex $v \in V(H)$, and an arbitrary integer $a \geq 1$, the *(a, v)-path augmentation* of H is the graph obtained as follows:

1. consider the disjoint union of H and P_a, the path of order a,
2. add an edge between v and one endvertex of P_a.

This construction is depicted in Figure 3.3. First suppose that $k - k' \geq 2$. From F, start by constructing a graph G_F and a $|V(G_F)|$-sequence $\pi_F = (n_1, n_2, ..., n_{k-k'})$ with size $k - k'$ such that F is 1-in-3 satisfiable if and only if π_F is realizable in G_F. This graph G_F and this sequence π_F may be obtained using the reduction from the proof of Theorem 3.2 since $k - k' \geq 2$. Let us

now denote by G'_F the graph obtained from G_F by performing k' arbitrary path augmentations, e.g. one (a_1, v)-path augmentation, one (a_2, v)-path augmentation, etc., for an arbitrary vertex $v \in V(G_F)$ and arbitrary integers $a_1, a_2, ..., a_{k'} \geq 1$. Let $u_1, u_2, ..., u_{k'}$ denote the vertices with degree 1 of the resulting hanging paths, where u_i is the endvertex of the ith path augmentation. Finally, let $\pi'_F = (a_1, a_2, ..., a_{k'}, n_1, n_2, ..., n_{k-k'})$ and $P'_F = (u_1, u_2, ..., u_{k'})$ be a $|V(G'_F)|$-sequence with size k and a k'-preassignation of G'_F, respectively.

Since the first k' parts $U_1, U_2, ..., U_{k'}$ of a P'_F-realization of π'_F in G'_F must induce connected subgraphs of G'_F on $a_1, a_2, ..., a_{k'}$ vertices, respectively, including $u_1, u_2, ..., u_{k'}$, respectively, the only way for choosing the part U_i is to pick up every vertex resulting from the ith path augmentation for every $i \in \{1, 2, ..., k'\}$. Once these parts $U_1, U_2, ..., U_{k'}$ have been picked up, we still have to find a realization $(V_1, V_2, ..., V_{k-k'})$ of the remaining sequence $(n_1, n_2, ..., n_{k-k'}) = \pi_F$ in the remaining graph $G'_F - \bigcup_{i=1}^{k'} U_i = G_F$. Hence, π'_F is P'_F-realizable in G'_F if and only if π_F is realizable in G_F. By transitivity, we get that F is 1-in-3 satisfiable if and only if π'_F is P'_F-realizable in G'_F.

Note that this reduction does not work when $k - k' \in \{0, 1\}$ since Realizable Size-0 Sequence and Realizable Size-1 Sequence are not NP-hard. But recall that, in the reduction from 1-in-3 Satisfiability to Realizable Size-2 Sequence, some vertices from the base and clause subgraphs of G_F, respectively, have to be covered by the parts with size n_1 and n_2, respectively, of a realization of π_F in G_F. Thus, we could, without altering the equivalence, request up to two preassigned vertices, and directly get that Realizable Size-2 Sequence with 1-Preassignation and Realizable Size-2 Sequence with 2-Preassignation are NP-hard. By then performing the same reduction scheme as above but from one of these two problems, we get that Realizable Size-k Sequence with k'-Preassignation is also NP-hard when $k - k' \in \{0, 1\}$. ■

3.1.2 Restrictions on the graph

The cornerstone of the upcoming complexity results is the following straightforward reduction from 3-Partition to Realizable Sequence.

Proposition 3.7. 3-Partition *is polynomial-time reducible to* Realizable Sequence.

Proof. Consider an instance $<A, B, s>$ of 3-Partition. We produce an instance of Realizable Sequence, i.e. a graph G and a $|V(G)|$-sequence π, such that

$$<A, B, s> \text{ admits a solution}$$
$$\Leftrightarrow$$
$$\pi \text{ is realizable in } G.$$

As in the definition of 3-Partition, we here and further let $|A| = 3m$ and $A = \{a_1, a_2, ..., a_{3m}\}$. The graph G is a disconnected one only consisting in m vertex-disjoint arbitrarily partitionable components with order B (e.g. the complete graph K_B, the path P_B, etc.). In particular we have $|V(G)| = mB = \sum_{a \in A} s(a)$. Now consider, as π, the $|V(G)|$-sequence $(s(a_1), s(a_2), ..., s(a_{3m}))$.

The equivalence between the two instances is easy to visualize. Consider any part V_i with size $s(a_i)$ from a realization of π in G. Then V_i includes vertices from one component of G only since otherwise $G[V_i]$ would not be connected. So basically if a realization of π in G exists, then it means that each of the components of G is covered by three parts with size $s(a_{i_1})$, $s(a_{i_2})$ and $s(a_{i_3})$, and thus that $s(a_{i_1}) + s(a_{i_2}) + s(a_{i_3}) = B$. A solution to $<A, B, s>$ can then directly be deduced from a realization of π in G, and conversely by similar arguments. ■

Hence, if, from an instance $<A, B, s>$ of 3-Partition, we can produce a graph G and a $|V(G)|$-sequence π such that, in every realization of π in G, some parts are "forced" to include some vertices of G so that what remain are m vertex-disjoint arbitrarily partitionable

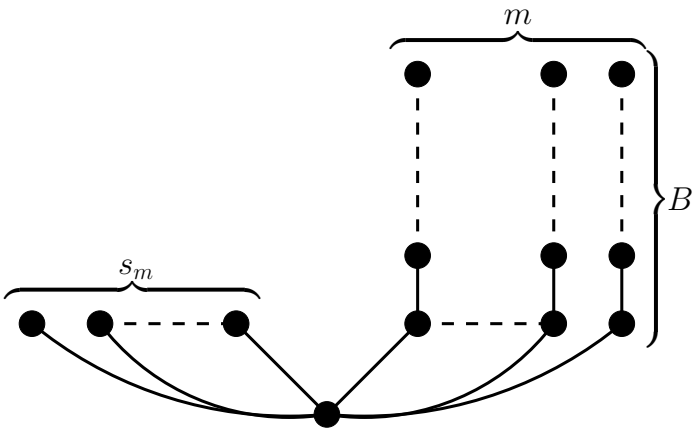


Figure 3.4: Structure of the reduced multipode.

(these actually only need to be arbitrarily 3-partitionable) components with order B and the sequence $(s(a_1), s(a_2), ..., s(a_{3m}))$, then, according to Proposition 3.7, by transitivity we get that the instance $<G, \pi>$ of Realizable Sequence is equivalent to the instance $<A, B, s>$ of 3-Partition.

3.1.2.1 Multipodes

As mentioned in Theorem 2.10, it was shown by Ravaux in [104] that Realizable Sequence is NP-complete when restricted to combs. The reduction designed by Ravaux to show this statement actually provides combs with α degree-3 nodes, where α is linear with the size of an instance of the reduced problem.

We show below that Realizable Sequence remains NP-complete when restricted to multipodes, i.e. to trees with only one node with degree at least 3. This contrasts with the fact that Arbitrarily Partitionable Graph is in P when restricted to multipodes, as mentioned in introductory Section 2.3.

Theorem 3.8. Realizable Sequence *is NP-complete when restricted to multipodes.*

Proof. The reduction is from 3-Partition. Given an instance $<A, B, s>$ of 3-Partition, we construct a multipode G and a $|V(G)|$-sequence π such that

$$<A, B, s> \text{ admits a solution}$$
$$\Leftrightarrow$$
$$\pi \text{ is realizable in } G.$$

Recall that we may suppose that $s(a) > 1$ holds for every $a \in A$ according to Observation 1.54. Let $s_m = \max\{s(a) : a \in A\}$, and consider $G = P_{s_m+m}(1, 1, ..., 1, B, B, ..., B)$ the multipode with s_m arms of order 1 and m arms of order B (see Figure 3.4). Now, as π, consider $\pi = (s_m + 1, s(a_1), s(a_2), ..., s(a_{3m}))$.

The keystone of the reduction is that, because no element of π is equal to 1, in every realization of π in G the part containing the root of G necessarily also contains all nodes from the arms with order 1. Since there are s_m arms with order 1 in G, the part containing the root must thus have size at least $s_m + 1$. So basically the part with size $s_m + 1$ of every realization of π in G must include the root of G as well as all the nodes from its s_m arms with order 1.

Once this part is picked, what remain are a forest of m paths P_B and the sequence $(s(a_1), s(a_2), ..., s(a_{3m}))$. Hence finding a realization of π in G is equivalent to the problem of finding a realization of $(s(a_1), s(a_2), ..., s(a_{3m}))$ in a forest of m paths P_B, while this problem is equivalent to solving $<A, B, s>$ according to the arguments in the proof of Proposition 3.7. The NP-hardness of Realizable Sequence when restricted to multipodes then follows by transitivity. ■

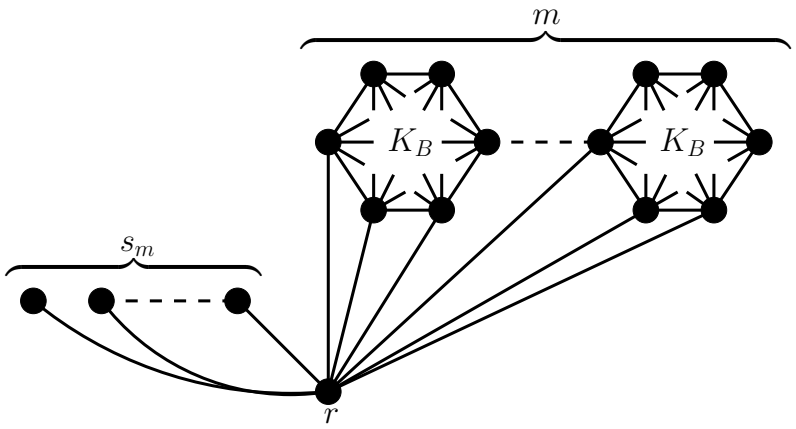


Figure 3.5: Structure of the reduced cograph.

3.1.2.2 Cographs

The question of the recognition of arbitrarily partitionable cographs was notably addressed by Broesma, Kratsch and Woeginger in [39]. Although we have no clue regarding the status of Arbitrarily Partitionable Graph when restricted to cographs, we herein show, as a first result, that Realizable Sequence remains NP-complete when restricted to cographs.

Note at first that the reduction described in the proof of Proposition 3.7 can be directly used to prove the NP-completeness of Realizable Sequence when restricted to non-connected cographs.

Proposition 3.9. Realizable Sequence *remains NP-complete when restricted to non-connected cographs.*

Proof. Just perform the reduction from 3-Partition to Realizable Sequence described in the proof of Proposition 3.7 so that each of the m components of the reduced graph G is a copy of K_B. Then the equivalence holds according to the arguments given in the proof of Proposition 3.7. Clearly G is a cograph since complete graphs are cographs defined as

$$K_n = \begin{cases} \bullet & \text{if } n = 1 \\ \bullet \times K_{n-1} & \text{otherwise,} \end{cases}$$

and G is nothing but a disjoint union of m copies of K_B. ■

Since studying arbitrarily partitionable graphs only makes sense in the context of connected graphs, we prove the analogue of Proposition 3.9 for connected cographs.

Theorem 3.10. Realizable Sequence *is NP-complete when restricted to connected cographs.*

Proof. The reduction is similar to the one for multipodes (proof of Theorem 3.8), with the exception that G differs a bit since a multipode is generally not a cograph. This time, the graph G is obtained as follows (see Figure 3.5). Start from a single vertex r. Now, add s_m copies of K_1 as well as m copies of K_B to the graph. Finally turn r into a universal vertex. The sequence π is obtained similarly as in the proof of Theorem 3.8.

The equivalence between the instance $<G, \pi>$ of Realizable Sequence and $<A, B, s>$ then follows from the same reasons, namely because all elements in π have value at least 2 and a lot of vertices of G only neighbour r. Besides G is a cograph whose one representation is

$$\bullet \times (\bullet + \bullet + ... + \bullet + K_B + K_B + ... + K_B),$$

where the terms "$\bullet$" and "K_B" are repeated s_m and m times, respectively, in the parentheses. ■

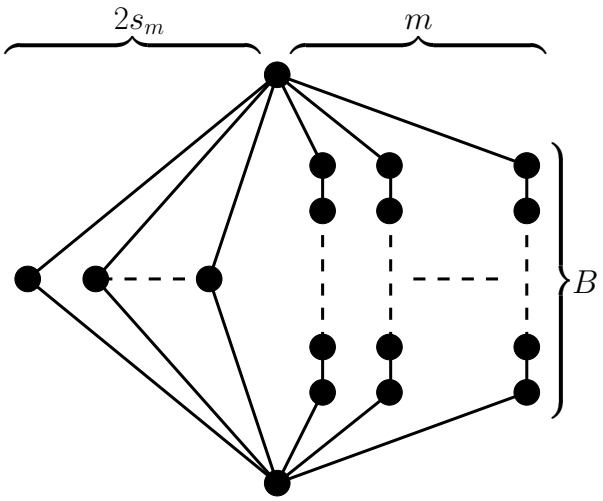


Figure 3.6: Structure of the reduced balloon.

3.1.2.3 Series-parallel and balloon graphs

We herein show that Realizable Sequence remains NP-complete when restricted to balloons, which, recall, are series-parallel graphs. This result is interesting for two reasons. First because the class of balloons turned out to be interesting regarding the structure of arbitrarily partitionable 2-connected graphs, as explained in Section 2.3. Second because series-parallel graphs shown up to have convenient properties when dealing with some notoriously hard problems. This NP-complete restriction hence confirms the hardness of Realizable Sequence.

Theorem 3.11. Realizable Sequence *is NP-complete when restricted to balloons.*

Proof. We use the same reduction scheme as in the proofs of Theorems 3.8 and 3.10, but modify the construction of G so that this graph is a balloon, and modify π accordingly so that its elements sum up to $|V(G)|$.

As G, just consider $G = B(1, 1, ..., 1, B, B, ..., B)$ the $(2s_m + m)$-balloon with $2s_m$ branches with order 1, and m branches with order B (see Figure 3.6). Now, as π, consider $\pi = (s_m + 1,$ $s_m + 1,$ $s(a_1),$ $s(a_2),$ ..., $s(a_{3m}))$.

Because every vertex from a branch with order 1 of G only neighbours the roots of G, it has to belong, in every realization of π in G, to a same part as one of the roots. Said differently, the at most two parts covering the roots of G also have to cover all of the vertices from the branches with order 1. Because there are $2s_m$ branches with order 1, these at most two parts must cover at least $2s_m + 2$ vertices. In view of the part sizes of π, in every realization of π in G we necessarily have to use the two parts with size $s_m + 1$ to cover all these vertices. Once these two parts have been picked, what remain are a forest of m paths P_B with order B and the sequence $(s(a_1), s(a_2), ..., s(a_{3m}))$. The equivalence between $<G, \pi>$ and $<A, B, s>$ then follows from the arguments in Proposition 3.7. ■

3.1.2.4 Graphs with given connectivity

As shown by Győri and Lovász, recall Theorem 2.1, realizations of small sequences, i.e. with size at most k, necessarily exist in k-connected graphs. Although k-connected graphs have minimum degree k, this property does not make these graphs dense enough to necessarily be arbitrarily partitionable (consider e.g. a k-connected complete bipartite graph with no perfect matching). Generalizing the hardness reduction from Theorem 3.11 for balloons, we actually prove that Realizable Sequence is NP-complete for graphs with fixed connectivity (or, equivalently, that Realizable Sequence should not be fixed-parameter tractable when parameterized by $\kappa(G)$).

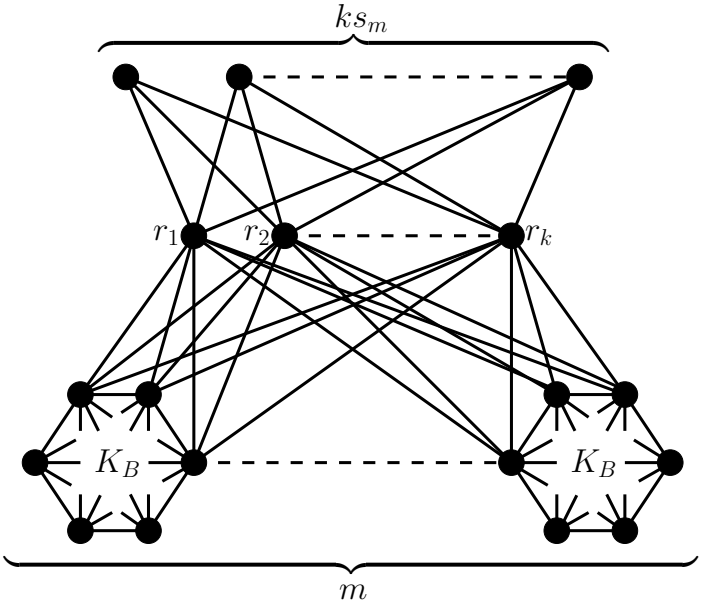


Figure 3.7: Structure of the reduced k-connected graph.

Theorem 3.12. *For every $k \geq 1$,* Realizable Sequence *is NP-complete when restricted to k-connected graphs.*

Proof. The reduction is again inspired by the reduction we used in the proofs of Theorem 3.8, 3.10 and 3.11. Let $k \geq 1$ be fixed, and construct G as follows (see Figure 3.7). Add k vertices $r_1, r_2, ..., r_k$ to G, as well as ks_m copies of K_1 and m copies of K_B. Finally, for every $i \in \{1, 2, ..., k\}$ add an edge between r_i and every vertex of $V(G) \setminus \{r_1, r_2, ..., r_k\}$. To obtain the instance $<G, \pi>$ of Realizable Sequence, just set

$$\pi = (s_m + 1, s_m + 1, ..., s_m + 1, s(a_1), s(a_2), ..., s(a_{3m})),$$

where the value $s_m + 1$ appears exactly k times at the beginning of π.

The graph G, which is clearly a k-connected graph with k-cutset $\{r_1, r_2, ..., r_k\}$, has to be thought of as a kind of balloon, where $r_1, r_2, ..., r_k$ would be its roots and the components isomorphic to K_1 or K_B its branches. According to the same arguments as in the proof of Theorem 3.11, the k parts with size $s_m + 1$ of a realization of π in G must each include one root vertex and s_m vertices from components isomorphic to K_1. What remain once these k parts have been picked are m vertex-disjoint components isomorphic to K_B, as well as the sequence $(s(a_1), s(a_2), ..., s(a_{3m}))$. The equivalence between $<A, B, s>$ and $<G, \pi>$ then follows from the arguments given in the proof of Proposition 3.7. ■

3.1.2.5 Regular graphs

We now focus on the NP-completeness of Realizable Sequence when restricted to regular graphs, which make up a class of graphs of interest regarding arbitrarily partitionable graphs, recall Conjecture 2.37. Regarding Conjecture 2.37, it would be interesting to prove that Realizable Sequence remains NP-complete when restricted to k-regular graphs for every fixed $k \geq 3$ (though it would not prove or disprove this conjecture).

As a first observation, note that, in the line of Proposition 3.9, the straight reduction from the proof of Proposition 3.7 can be directly modified to prove the following.

Proposition 3.13. *For every $k \geq 3$,* Realizable Sequence *is NP-complete when restricted to non-connected k-regular graphs.*

Proof. Let $k \geq 3$ be fixed. We use the reduction scheme from 3-Partition to Realizable Sequence we introduced in the proof of Proposition 3.7. Let $<A, B, s>$ be an instance of 3-Partition. We may suppose that $B \geq k+1$ and B is even, recall Observation 1.54. As the sequence π of the reduced instance of Realizable Sequence, just consider $\pi = (s(a_1), s(a_2), ..., s(a_{3m}))$. As its graph G, just consider

$$G = H_{k,B} + H_{k,B} + ... + H_{k,B},$$

the disjoint union of m copies of $H_{k,B}$, the k-connected Harary graph with order B.

Since B is even, by construction $H_{k,B}$, and hence G, is k-regular. Besides, it is easily seen that $H_{k,B}$ is Hamiltonian (and even traceable), and hence arbitrarily partitionable according to Observation 2.34. The arguments from the proof of Proposition 3.7 then directly imply that $<G, \pi>$ and $<A, B, s>$ are equivalent. ■

Since only connected graphs are concerned by Conjecture 2.37, studying whether an equivalent of Proposition 3.13 for connected graphs holds makes more sense. We prove below that such a result is true for every $k \geq 5$ odd, though we think our reduction could be adapted to some of the remaining values of k with some more efforts. But, although not complete, this result already ensures that Realizable Sequence should not be fixed-parameter tractable when parameterized by k when assumed G is a connected k-regular graph. Since a k-regular graph is a specific case of graph with maximum degree k, this result also implies the non-fixed parameter tractability of Realizable Sequence when parameterized by the maximum degree of G.

Theorem 3.14. *For every $k \geq 5$ odd,* Realizable Sequence *is NP-complete when restricted to connected k-regular graphs.*

Proof. The reduction is again based on the reduction scheme from the proof of Proposition 3.7 we used to prove the previous results. Let $k \geq 5$ odd, and consider an instance $<A, B, s>$ of 3-Partition. We produce an instance $<G, \pi>$ of Realizable Sequence which is equivalent to $<A, B, s>$ with the property that G is a connected k-regular graph. The value s_m related to $<A, B, s>$ is defined similarly as previously, and we analogously define $s_i = \min\{s(a) : a \in A\}$. Recall that we may suppose that $s_i - 1$ is even according to Observation 1.54.

The reduced graph G is made up of several of the upcoming gadgets. We make use of the notation introduced in Section 1.2.2.13 to deal with Harary graphs.

Construction 3.15. A (B, k)-*gadget* is a graph H obtained by starting with H being the k-connected Harary graph $H_{k,B}$ on B vertices, and then removing some edges from H. In case B is even (and hence H is k-regular so far), just remove the edge v_0v_1 from H. Now in case B is odd, remove the edges $v_0v_{\lfloor \frac{B}{2} \rfloor}$ and $v_0v_{\lceil \frac{B}{2} \rceil}$ from H. The (at most three) vertices of H which where incident to removed edges are called the *roots* of H.

Note that every (B, k)-gadget H is almost k-regular in the sense that either its roots have degree $k-1$ while all its non-root vertices have degree k (when B is even), or H has three roots whose one has degree $k-2$ while the other two have degree $k-1$, and the non-root vertices of H have degree k (when B is odd). Besides, it is easily seen that a (B, k)-gadget is traceable (and hence arbitrarily partitionable according to Observation 2.34) since Harary graphs are Hamiltonian.

Construction 3.16. A $(s_i - 1, k)$-*gadget* H is obtained similarly as a (B, k)-gadget, i.e. by removing specific edges from a Harary graph. Recall that $s_i - 1$ is even. First consider H as being the k-regular Harary graph H_{k,s_i-1} on $s_i - 1$ vertices, and just remove the edge v_0v_1 from H. Again we call v_0 and v_1 the *roots* of H.

Note again that a $(s_i - 1, k)$-gadget is a traceable graph and is almost k-regular in the sense used above. We finally need to introduce what we call a Y_k-*gadget*.

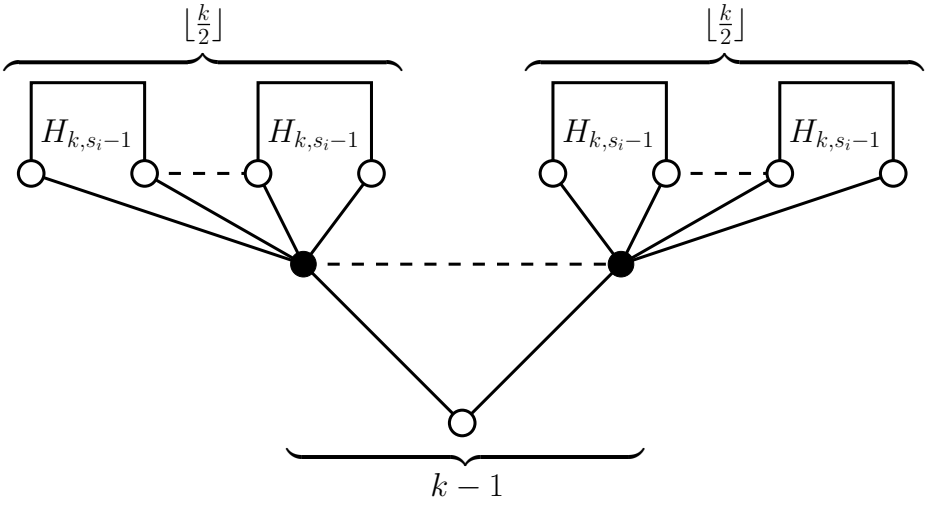


Figure 3.8: A Y_k-gadget. The bottommost white vertex is the root of the gadget. The upmost white vertices are the roots of $(s_i - 1, k)$-gadgets.

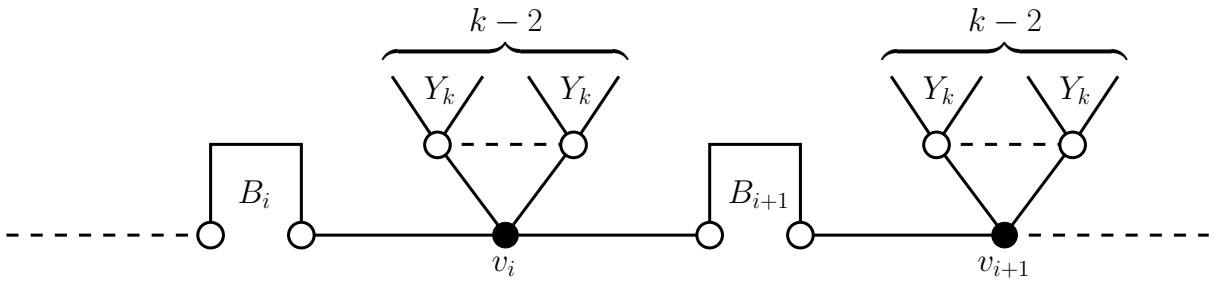


Figure 3.9: Illustration of the construction for B even. The white vertices are the roots of some gadgets.

Construction 3.17. A *Y_k-gadget* H is obtained by starting from the star S_k on k vertices. We refer to the vertex with degree $k-1$ of H as its *root*. Now, for each vertex v adjacent with the root of H (such a vertex has degree 1 so far), add edges joining v and the roots of $\lfloor \frac{k}{2} \rfloor$ new $(s_i - 1, k)$-gadgets.

The structure of a Y_k-gadget is depicted in Figure 3.8. By construction, note that all vertices of a Y_k-gadget H but its root (which has degree $k-1$) have degree k. The gadget H can be seen as a tree whose leaves are $(s_i - 1, k)$-gadgets.

We can now describe the graph G of the reduced instance of Realizable Sequence. First add m vertices $v_0, v_1, ..., v_{m-1}$ to G, as well as m (B, k)-gadgets $B_0, B_1, ..., B_{m-1}$. Next, proceed as follows, where the indices are taken modulo m.

- If B is even, then start by adding an edge between v_i and the root of $k-2$ new Y_k-gadgets for every $i \in \{0, 1, 2, ..., m-1\}$. Then, for every $i \in \{0, 1, ..., m-1\}$, assuming u_1 and u_2 are the roots of B_i, add the edges $v_i u_1$ and $u_2 v_{i+1}$ to G.
- If B is odd, then start by adding an edge between v_i and the root of $k-3$ new Y_k-gadgets for every $i \in \{0, 1, 2, ..., m-1\}$. Then, for every $i \in \{0, 1, ..., m-1\}$, assuming u_1, u_2 and u_3 are the roots of B_i, add $v_i u_1$, $u_2 v_{i+1}$ and $u_3 v_{i+1}$ to G.

The construction is illustrated in Figure 3.9. Note that G is k-regular, since only the roots of its gadgets did not have degree k, but we added edges incident to these vertices so that their degree is exactly k. Set $y = 1 + (k-2)|V(Y_k)|$ if B is even, and $y = 1 + (k-3)|V(Y_k)|$ otherwise, where Y_k denotes a Y_k-gadget (basically y is the order of the components obtained by removing

the (B,k)-gadgets from G). To obtain the instance $<G,\pi>$ of Realizable Sequence, finally consider

$$\pi = (y, y, ..., y, s(a_1), s(a_2), ..., s(a_{3m})),$$

where there are m occurrences of the element y at the beginning of π.

The equivalence between $<G,\pi>$ and $<A,B,s>$ follows from the same arguments as in the previous proofs. Consider any vertex u from one of the Y_k-gadgets of G which is joined to some (s_i-1,k)-gadgets. Since $k \geq 5$, there are at least two such gadgets attached to u. Because removing u from G leaves two "small" components with order $s_i - 1$ and all elements of π are strictly greater than $s_i - 1$, note that every part of a realization of π in G including u also has to cover all vertices from all (s_i-1,k)-gadgets attached to u. This part cannot be one part with size at most s_m since otherwise we would have

$$s_m \geq 2(s_i - 1) + 1,$$

but then, since $s_i > \frac{B}{4}$ by assumption, we would obtain

$$s_m > \frac{B}{2} - 1,$$

a contradiction to the initial assumptions.

So every such vertex u has to belong to one part with size y of the realization. Due to the connectivity of the Y_k-gadgets, which are 1-connected, every Y_k-gadget and the vertex attached to its root actually have to belong to a same part. What are left once these parts have been picked are m (B,k)-gadgets and the subsequence $(s(a_1), s(a_2), ..., s(a_{3m}))$. Since the (B,k)-gadgets are arbitrarily partitionable as pointed out above, the claimed equivalence then follows directly by the arguments given in the proof of Proposition 3.13. ■

3.1.2.6 Graphs with about one third universal vertices

Universal vertices are quite helpful for partitioning a graph into connected subgraphs since the presence of any such vertex in a subgraph implies its connectedness. But clearly a graph with universal vertices does not have to be arbitrarily partitionable (consider e.g. a star on at least four vertices).

In the next result, we show that Realizable Sequence remains NP-complete when restricted to graphs with up to one third universal vertices. This result motivates the upcoming study of the partitioning of graphs with between one third and one half universal vertices in Section 3.3.2. The reduction we use is not based on the previous reduction from 3-Partition we have been exploiting so far.

Theorem 3.18. Realizable Sequence *is NP-complete when restricted to graphs with up to one third universal vertices.*

Proof. Recall that Realizable Sequence remains NP-hard when restricted to instances with $\pi = (3, 3, ..., 3)$ (but with no conditions on the structure of G), see Theorem 2.10. We use this restriction of Realizable Sequence for the reduction. Namely, from a graph G on $3n$ vertices, we construct a graph G' with order $3n'$ $(n' > n)$ and up to one third universal vertices, and such that

$$\text{the } 3n\text{-sequence } (3, 3, ..., 3) \text{ is realizable in } G$$
$$\Leftrightarrow$$
$$\text{the } 3n'\text{-sequence } (3, 3, ..., 3) \text{ is realizable in } G'.$$

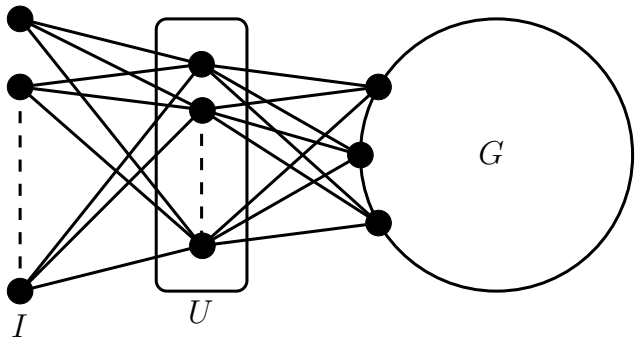


Figure 3.10: Structure of the reduced graph with about one third universal vertices. The vertices in U form a clique.

To obtain the graph G', proceed as follows. Let $k \geq 1$ be an arbitrary integer, and add $3k$ new vertices to G. Arbitrarily partition these $3k$ newly added vertices into two parts $I \cup U$ in such a way that $|I| = 2k$ and $|U| = k$. Finally turn the vertices in U into universal vertices. The graph G' is thus made of three vertex-disjoint subgraphs $G'[I]$, $G'[U]$ and $G'[V(G)]$, which are connected in such a way that all vertices from the "central" component $G'[U]$ neighbour all vertices of G', the vertices in I neighbour vertices in U only, and $G'[V(G)]$ is nothing but G (with all possible edges between vertices of U and $V(G)$). Clearly $|V(G')| = 3n' = 3(n + k)$. Refer to Figure 3.10 for an illustration of this construction.

Note that in every realization of the $3n'$-sequence $(3, 3, ..., 3)$ in G', every vertex from I has to belong to a same part as a vertex from U because of the structure of G'. Besides, every part containing a vertex from U can only cover up to two vertices in I. For these reasons, and since there are $2k$ vertices in I and k vertices in U by construction, note that, in every realization of the $3n'$-sequence $(3, 3, ..., 3)$ in G', exactly k parts necessarily consist in exactly one vertex from U and two vertices from I. Once these k parts are removed from G', what remains is G. Hence, the existence of a realization of the $3n'$-sequence $(3, 3, ..., 3)$ in G' only depends on the existence of a realization of the $3n$-sequence $(3, 3, ..., 3)$ in $G' - (I \cup U) = G$. The two instances of Realizable Sequence are thus equivalent.

It should be clear that the reduction above holds whatever is the value of k. In particular, note that the order of G gets irrelevant in front of the order of G', and thus that the number of universal vertices of G' tends to one third, as k grows to infinity. ■

3.1.3 On the tightness of Győri-Lovász Theorem

We now investigate the tightness of Theorem 2.1 in terms of connectivity and number of preassigned vertices. Theorem 2.1 implies that every instance of Realizable Size-k Sequence with k'-Preassignation involving a q-connected graph with $k' \leq k \leq q$ is positive. In the next result, we show that this easiness result is tight in the sense that partitioning a q-connected graphs into at least $q + 1$ connected parts is difficult as soon as at least $q + 1$ vertices are preassigned. Again, as a side result we directly get that Realizable Size-k Sequence with k'-Preassignation should not be fixed-parameter tractable when parameterized by k, k' and $\kappa(G)$.

Theorem 3.19. *For every $q \geq 1$, $k' \geq q+1$ and $k \geq k'$,* Realizable Size-k Sequence with k'-Preassignation *is NP-complete when restricted to q-connected graphs.*

Proof. First, because Realizable Size-k Sequence with k'-Preassignation is NP-complete for every $k \geq 2$ and our proof of this statement was obtained by reducing instances of 1-in-3 Satisfiability to instances of Realizable Sequence involving 1-connected graphs, see the proof of Theorem 3.5, the statement holds for $q = 1$.

We now prove the general case, i.e. $q \geq 2$, by reduction from 1-IN-3 SATISFIABILITY. Let $k' \geq q+1$ and $k \geq k'$ be fixed. Given a formula F being an instance of 1-IN-3 SATISFIABILITY, produce a graph G_F, a $|V(G_F)|$-sequence $\pi_F = (n_1, n_2, ..., n_{k-q+1})$, and a $(k'-q+1)$-preassignation $P_F = (u_1, u_2, ..., u_{k'-q+1})$ of G_F such that F is 1-in-3 satisfiable if and only if π_F is P_F-realizable in G_F. This reduced instance may be obtained by using the reduction given in the proof of Theorem 3.1 and the star and path augmentation constructions, recall Constructions 3.3 and 3.6, respectively. Now consider the following instance $<G'_F, \pi'_F, P'_F>$ of REALIZABLE SIZE-k SEQUENCE WITH k'-PREASSIGNATION.

- G'_F is obtained by successively adding $q-1$ universal vertices $v_1, v_2, ..., v_{q-1}$ to G_F.
- $\pi'_F = (1, 1, ..., 1, n_1, n_2, ..., n_{k-q+1})$ is a $|V(G'_F)|$-sequence with $q-1$ 1's.
- $P'_F = (v_1, v_2, ..., v_{q-1}, u_1, u_2, ..., u_{k'-q+1})$ is a k'-preassignation of G'_F.

Clearly, G'_F is q-connected since G_F is 1-connected, and π'_F and P'_F have size k and k', respectively. Besides, since preassigning a vertex to a part with size 1 is like removing it from the graph, what is left once the vertices $v_1, v_2, ..., v_{q-1}$ have been preassigned to parts with size 1 of a P'_F-realization of π'_F in G'_F is G_F, π_F and P_F. Therefore, π'_F is P'_F-realizable in G'_F if and only if π_F is P_F-realizable in G_F. By transitivity, we hence get that F is 1-in-3 satisfiable if and only if π'_F is P'_F-realizable in G'_F. ■

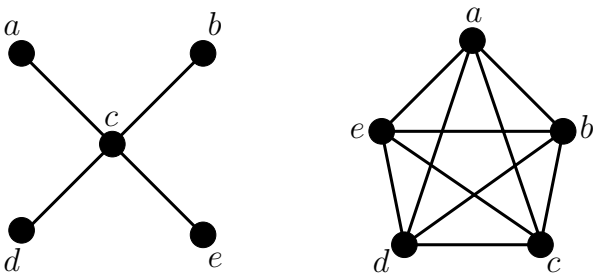
3.2 Relationship between Π_2^p and partition problems

As mentioned in Section 2.3, the membership of ARBITRARILY PARTITIONABLE GRAPH to Π_2^p follows directly from the membership of REALIZABLE SEQUENCE to NP. We however do not know whether ARBITRARILY PARTITIONABLE GRAPH is Π_2^p-complete. Indeed, most of known Π_2^p-complete problems are "completion" problems, i.e. problems of the form *"For every X, is there a Y such that...?"* with the two input objects X and Y being of the same nature (e.g. truth assignments, sets of vertices, etc.). Refer e.g. to the compendium [107] by Schaefer and Umans, wherein several Π_2^p-complete problems are listed, to have an illustration of this claim. So although the question of ARBITRARILY PARTITIONABLE GRAPH catches the form of a Π_2^p question, it is however different from the form of a Π_2^p-complete question. This is one reason why it seems difficult to design a reduction from one classic Π_2^p-complete problem to ARBITRARILY PARTITIONABLE GRAPH.

In order to show that graph partition problems are not "incompatible" with the notion of Π_2^p-complete problems, we introduce another related problem.

Definition 3.20. Let G be a graph and $\pi = (n_1, n_2, ..., n_p)$ be a $|V(G)|$-sequence. Given an $\ell \in \{1, 2, ..., p\}$, an *$n_\ell$-partition-level* for π and G is a set L_ℓ of subsets of $V(G)$ that induce connected subgraphs of G with order n_ℓ. An *$(n_1, n_2, ..., n_\ell)$-partition-hierarchy* L for π and G is a collection $L = (L_1, L_2, ..., L_\ell)$, where L_i is an n_i-partition-level for every $i \in \{1, 2, ..., \ell\}$, such that no subsets in L_i and L_j intersect for $i \neq j$. We finally say that π is *L-realizable* in G if, for every collection $(V_1, V_2, ..., V_\ell)$ of subsets from the partition-levels of L such that $V_1 \in L_1, V_2 \in L_2, ..., V_\ell \in L_\ell$, there exists a realization $(V_1, V_2, ..., V_\ell, V_{\ell+1}, V_{\ell+2}, ..., V_p)$ of π in G.

In other words, we are given partial realizations of π in G (or, more precisely, ways for picking the connected parts associated with the ℓ first elements of π), whose parts are dispatched into ℓ partition-levels, and we ask whether each of these partial realizations is extendible to a whole realization of π in G. A partition-hierarchy is actually a compact way to describe a large number of partial realizations.



(a) $K_{1,4}$. (b) K_5.

Figure 3.11: Labellings of the graphs $K_{1,4}$ and K_5.

Example 3.21. Consider the two graphs $K_{1,4}$ and K_5 of Figure 3.11. Let $\pi = (1, 1, 3)$ be a 5-sequence, let $L_1 = (\{a\}, \{c\})$ and $L_2 = (\{b\}, \{e\})$ be two 1-partition-levels for π and both $K_{1,4}$ and K_5, and $L = (L_1, L_2)$ be a $(1, 1)$-partition-hierarchy for π and both $K_{1,4}$ and K_5. Clearly π is not L-realizable in $K_{1,4}$ since $(\{c\}, \{b\}, V(K_{1,4}) \setminus \{c, b\})$ is not a realization of π in $K_{1,4}$. However, π is L-realizable in K_5 since $(\{a\}, \{b\}, V(K_5) \setminus \{a, b\})$, $(\{a\}, \{e\}, V(K_5) \setminus \{a, e\})$, $(\{c\}, \{b\}, V(K_5) \setminus \{c, b\})$ and $(\{c\}, \{e\}, V(K_5) \setminus \{c, e\})$ are realizations of π in K_5.

We now focus on the complexity of the problem associated with the definitions above.

Dynamic Realizable Sequence
Instance: A graph G, a $|V(G)|$-sequence $\pi = (n_1, n_2, ..., n_{p'}, n_{p'+1}, n_{p'+2}, ..., n_p)$ with size $p \geq p'$, and a $(n_1, n_2, ..., n_{p'})$-partition-hierarchy L for π and G.
Question: Is π L-realizable in G?

It is worth mentioning that Dynamic Realizable Sequence does not seem to be in co-NP (for the same reason as Realizable Sequence does not seem to belong to co-NP) or in NP since the number of partial realizations encoded by a partition-hierarchy may be exponential compared to the input size. Dynamic Realizable Sequence however has a typical Π_2^p-complete problem form, as proved in the next result.

Theorem 3.22. Dynamic Realizable Sequence *is* Π_2^p*-complete.*

Proof. Dynamic Realizable Sequence is clearly a Π_2^p problem. Given a combination of parts $(V_1, V_2, ..., V_{p'})$ from L, we can, using an oracle for a problem in NP $\cup$ co-NP, check in polynomial time whether these parts cannot be extended to get a realization of π in G. For this purpose, we just have to invoke an oracle for Realizable Sequence to make sure that the sequence $(n_{p'+1}, n_{p'+2}, ..., n_p)$ is indeed not realizable in $G - \bigcup_{i=1}^{p'} V_i$.

We now show that Dynamic Realizable Sequence is complete in Π_2^p by reduction from $\forall\exists$ 1-in-3 Satisfiability, which is Π_2^p-complete according to Lemma 1.51. Our reduction is nothing but a Π_2^p-complete version of the reduction from 1-in-3 Satisfiability to Realizable Sequence we gave in the proof of Theorem 3.1. Remember that, in this reduction, setting a literal of F to true is simulated, in the reduced instance $<G_F, \pi_F>$ of Realizable Sequence, by adding the corresponding literal vertex of G_F to the part with size $n_1 + n$ of a realization of π_F in G_F. We here want to keep this relationship between attributing a truth value to a literal of F and adding the associated literal vertex of G_F to one of the two parts of a realization of π_F in G_F. Given a truth assignment ϕ_1 to the variables in X_1, it means that we have to check whether the partial realization of π_F in G_F whose part with size $n_1 + n$ contains the literal vertices associated with the true literals via ϕ_1 is extendible to a realization of π_F in G_F. All these possible partial realizations (i.e. associated with all possible truth assignments to X_1) are encoded by a partition-hierarchy for π_F and G_F.

Set $X_1 = \{x_1, x_2, ..., x_{n'}\}$ and $X_2 = \{x_{n'+1}, x_{n'+2}, ..., x_n\}$. First of all, let G_F be the graph obtained from F using the reduction we gave in the proof of Theorem 3.1. Then, let

$$\pi_F = (1, 1, ..., 1, n_1 + n - n', n_2 - n)$$

be a $|V(G_F)|$-sequence with size $n' + 2$, let $L_i = \{\{v_{x_i}\}, \{v_{\overline{x_i}}\}\}$ be a 1-partition-level for π_F and G_F for every $x_i \in X_1$, and $L = (L_1, L_2, ..., L_{n'})$ be a $(1, 1, ..., 1)$-partition-hierarchy for π_F and G_F. With every truth assignment ϕ_1 to X_1 setting n' literals of F to true is then associated the combination of vertex-disjoint subsets $(V_1, V_2, ..., V_{n'})$ from L where $V_i = \{v_{x_i}\}$ if $\phi_1(x_i) = 1$ or $V_i = \{v_{\overline{x_i}}\}$ otherwise for every $i \in \{1, 2, ..., n'\}$. This association is clearly bijective.

Let us now suppose that for every truth assignment ϕ_1 to X_1 there exists a truth assignment ϕ_2 to X_2 such that F is 1-in-3 satisfied. Then the partition $(V_1, V_2, ..., V_{n'+2})$ of $V(G_F)$, where

- for every $i \in \{1, 2, ..., n'\}$, we have $V_i = \{v_{x_i}\}$ if $\phi_1(x_i) = 1$ or $V_i = \{v_{\overline{x_i}}\}$ otherwise,
- $V_{n'+1}$ contains all the vertices from the base subgraph of G_F and every literal vertex v_{ℓ_i} of the clause subgraph of G_F such that $\phi_2(\ell_i) = 1$,
- $V_{n'+2} = V(G_F) \setminus \bigcup_{i=1}^{n'+1} V_i$,

is a realization of π_F in G_F according to the arguments we gave in the proof of Theorem 3.1. Because of the bijection described above, it follows that every combination of parts from the 1-partition-levels of L can be extended to a realization of π_F in G_F.

Conversely, suppose that every combination $(V_1, V_2, ..., V_{n'})$ of subsets from the 1-partition-levels of L is extendible to a realization $(V_1, V_2, ..., V_{n'+2})$ of π_F in G_F. As explained above, the partition $(V_1, V_2, ..., V_{n'})$ is associated with a truth assignment ϕ_1 to X_1, while from the literal vertices contained in $V_{n'+1}$ we can deduce a truth assignment ϕ_2 to X_2 such that F is 1-in-3 satisfied by ϕ_1 and ϕ_2 (see the proof of Theorem 3.1). Then for every truth assignment to X_1, there exists a truth assignment to X_2 making F 1-in-3- satisfied, as claimed. ■

3.3 Three polynomial kernels of sequences

In this section, we exhibit polynomial kernels of sequences for three classes of graphs, namely complete multipartite graphs (Section 3.3.1), graphs with about one half universal vertices (Section 3.3.2), and graphs made up of partitionable components (Section 3.3.3).

3.3.1 Complete multipartite graphs

A lot of complete multipartite graphs are arbitrarily partitionable since they are traceable. However, all complete multipartite graphs are not arbitrarily partitionable. To be convinced of that statement, note that every graph $M_2(1, k)$ with $k \geq 3$ odd does not admit a perfect matching, and hence any realization of the $(1 + k)$-sequence $(2, 2, ..., 2)$.

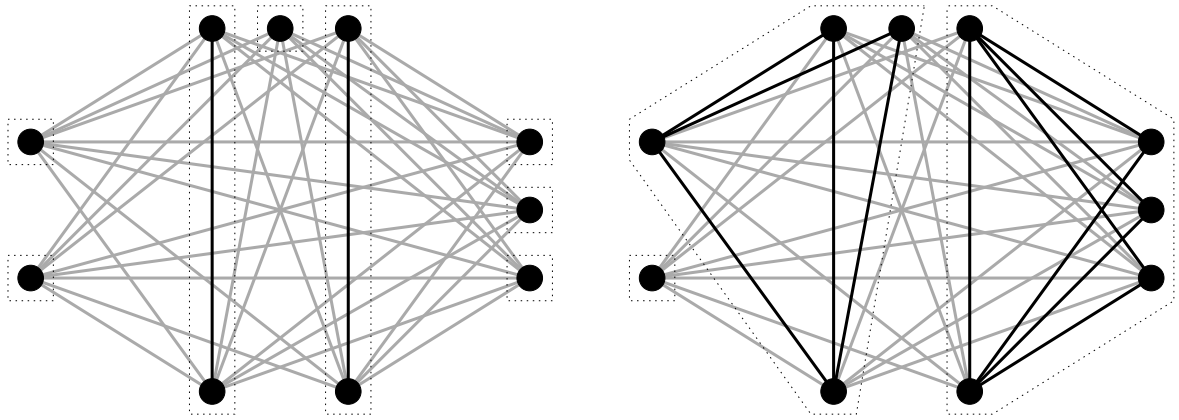
We herein prove the following result.

Theorem 3.23. Arbitrarily Partitionable Graph *is in P when restricted to complete multipartite graphs.*

So that we describe how we proceed to prove Theorem 3.23, we need to introduce a few terminology beforehand.

Notation 3.24. Let $k \geq 2$ and $n \geq k$ be two integers. We denote by $\mathcal{M}_k(n)$ the set of complete k-partite graphs with order n. We denote by $K_{\mathcal{M}_k}(n)$ the following set of n-sequences:

$$K_{\mathcal{M}_k}(n) = \{\pi : \|\pi\| = n \text{ and } sp(\pi) = \{1, 2\}\}.$$



(a) A realization of $(2, 2, 1, 1, 1, 1, 1, 1)$. (b) A realization of $(5, 4, 1)$.

Figure 3.12: Deduction of a realization of a sequence not in $K_{\mathcal{M}_k}(n)$ (b) in a complete multipartite graph (in black and grey) from a realization of a sequence in $K_{\mathcal{M}_k}(n)$ (a). The connected subgraphs induced by the realizations are depicted in black only.

Theorem 3.23 is proved as follows. First, we prove that $K_{\mathcal{M}_k}(n)$ is a (obviously polynomial) kernel for $\mathcal{M}_k(n)$. We then prove that Realizable Sequence is in P when restricted to graphs of $\mathcal{M}_k(n)$ and sequences of $K_{\mathcal{M}_k}(n)$. These two results directly imply Theorem 3.23.

The proof that $K_{\mathcal{M}_k}(n)$ is a kernel for $\mathcal{M}_k(n)$ relies on the following lemma.

Lemma 3.25. *Let $G = M_k(p_1, p_2, ..., p_k)$ be a complete k-partite graph with $k \geq 2$ and order $n \geq k$, and $\pi = (n_1, n_2, ..., n_p)$ be an n-sequence. If π is realizable in G, then every n-sequence $\pi' = \pi \setminus (n_i, n_{i_1}, n_{i_2}, ..., n_{i_{p'}}) \cup (n_i + n_{i_1} + n_{i_2} + ... + n_{i_{p'}})$, where $n_i \geq 2$ and $n_{i_1}, n_{i_2}, ..., n_{i_{p'}}$ are distinct arbitrary elements of $\pi \setminus (n_i)$, is realizable in G.*

Proof. Let $(V_1, V_2, ..., V_p)$ be a realization of π in G. Since $n_i \geq 2$, observe that $G[V_i \cup \{u\}]$ is connected for every vertex $u \notin V_i$ of G, and so is every subgraph $G[V_i \cup V_j]$ with $j \neq i$. It then follows directly that

$$(V_1, V_2, ..., V_{i-1}, V_i \cup V_{i_1}, V_{i+1}, V_{i+2}, ..., V_p) \setminus (V_{i_1})$$

is a realization of the n-sequence $(n_1, n_2, ..., n_{i-1}, n_i + n_{i_1}, n_{i+1}, n_{i+2}, ..., n_p) \setminus (n_{i_1})$ in G (see Figure 3.12). Repeating the same argument as many times as necessary, we eventually get a realization of π' in G. ■

Theorem 3.26. *For every $k \geq 2$ and $n \geq k$, the set $K_{\mathcal{M}_k}(n)$ is a kernel for $\mathcal{M}_k(n)$.*

Proof. Let $G \in \mathcal{M}_k(n)$ be a complete k-partite graph with order n. We show that G is arbitrarily partitionable if and only if $K_{\mathcal{M}_k}(n)$ is realizable in G. As the necessity follows from the definition of an arbitrarily partitionable graph, let us focus on the sufficiency. Assume $K_{\mathcal{M}_k}(n)$ is realizable in G, and that $\pi = (n_1, n_2, ..., n_p)$ is an n-sequence with $\pi \notin K_{\mathcal{M}_k}(n)$. We deduce an n-sequence $\pi' \in K_{\mathcal{M}_k}(n)$ whose realizability in G implies the realizability of π in G.

Since $\pi \notin K_{\mathcal{M}_k}(n)$, there are elements of π with value at least 3. For each such element n_i, replace n_i in π' with one occurrence of the element 2 and $n_i - 2$ occurrences of the element 1. Directly transfer every other element of π, i.e. with value at most 2, to π'. By construction, we have $\pi' \in K_{\mathcal{M}_k}(n)$. Now consider a realization of π' in G, which exists by assumption, and any element n_i of π which was split into one occurrence of 2 and several occurrences of 1 in π'. Then, according to Lemma 3.25, we can merge exactly one part with size 2 and $n_i - 2$ parts with size 1 of the realization of π' in G so that their union induces a connected subgraph of G with order n_i.

Repeating the same argument for every split element of π, we eventually get a realization of π in G. ■

Clearly, if the n-sequence $(2, 2, ..., 2)$ (or $(2, 2, ..., 2, 1)$ if n is odd) is realizable in a complete multipartite graph G with order n, then every other n-sequence with spectrum $\{1, 2\}$ is also realizable in G. So G is arbitrarily partitionable if and only if it has a matching with size $\lfloor \frac{|V(G)|}{2} \rfloor$. Since this can be checked in polynomial time, recall Theorem 2.11, we directly get that Arbitrarily Partitionable Graph is in P when restricted to complete multipartite graphs. This proves Theorem 3.23.

3.3.2 Graphs with about a half universal vertices

In this section we exhibit a polynomial kernel for graphs with about a half universal vertices. Recall that graphs having up to one third universal vertices do not have to be arbitrarily partitionable, see Theorem 3.18, while graphs with order n and at least $\lceil \frac{n-5}{2} \rceil$ universal vertices are arbitrarily partitionable according to Theorem 2.36. There hence should be a threshold t lying in between $\frac{n}{3}$ and $\frac{n}{2}$ such that every graph with order n and at least t universal vertices is necessarily arbitrarily partitionable.

Towards this question, we investigate the existence of a polynomial kernel for graphs with about one half universal vertices. In particular, as the main result of this section, we exhibit a polynomial kernel for graphs with order n and at least $\lceil \frac{n-\ln(n)-2}{2} \rceil$ universal vertices. Although such graphs have a lot of universal vertices, they do not have to be arbitrarily partitionable (consider e.g. the situation where all non-universal vertices form an independent set). So having such a polynomial kernel makes sense.

We start by raising the following easy remark.

Observation 3.27. *Let G be a graph with $k \geq 1$ universal vertices and order $n \geq k$. Then every n-sequence $\pi = (n_1, n_2, ..., n_p)$ with size $p \leq k$ is realizable in G.*

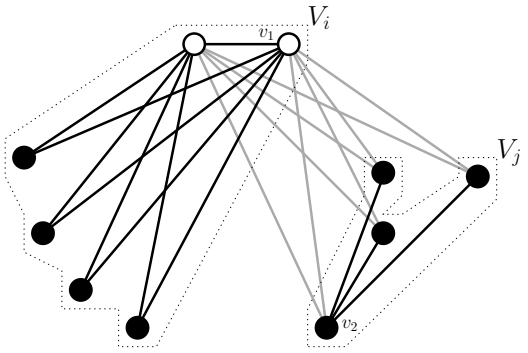
Proof. Let $u_1, u_2, ..., u_k$ be the universal vertices of G. Under the conditions of the claim, we deduce a realization $(V_1, V_2, ..., V_p)$ of π in G as follows. Start with $V_1 = \{u_1\}$, $V_2 = \{u_2\}$, ..., $V_p = \{u_p\}$. Now consider the parts $V_1, V_2, ..., V_p$ consecutively. If V_i already has size n_i, i.e. $n_i = 1$, then consider the next part. Otherwise, add $n_i - 1$ arbitrary vertices from $V(G) \setminus \bigcup_{j=1}^{p} V_j$ to V_i. Clearly V_i has size n_i. Besides, because $u_i \in V_i$ and u_i is a universal vertex of G, the subgraph induced by V_i is connected. ■

From now on, it is thus understood that all sequences have size at least $k + 1$ when dealing with a graph with k universal vertices. We now introduce a result dealing with the existence of a particular realization of every sequence which is realizable in a graph having universal vertices.

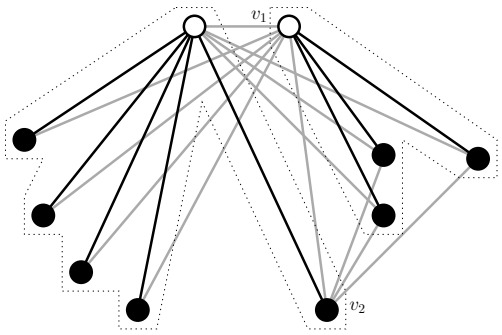
Lemma 3.28. *Let G be a graph with $k \geq 1$ universal vertices and order $n \geq k$, and $\pi = (n_1, n_2, ..., n_p)$ be an n-sequence with $n_1 \geq n_2 \geq ... \geq n_p$. If π is realizable in G, then there exists a realization $(V_1, V_2, ..., V_p)$ of π in G such that each of $V_1, V_2, ..., V_k$ contains one universal vertex.*

Proof. The claim means that if π is realizable in G, then there exists a particular realization of π in G such that each of the k biggest parts includes one universal vertex. Let $u_1, u_2, ..., u_k$ denote the universal vertices of G, and assume $(V_1, V_2, ..., V_p)$ is a realization of π in G. If $(V_1, V_2, ..., V_p)$ satisfies the conditions of the claim, then we are done. Otherwise, we obtain a satisfying realization in two steps.

We first modify the realization $(V_1, V_2, ..., V_p)$ so that each of $V_1, V_2, ..., V_k$ includes at most one universal vertex of G. Suppose there is a part V_i with $i \in \{1, 2, ..., k\}$ containing at least two universal vertices, while another part V_j with $j \in \{1, 2, ..., k\}$ and $j \neq i$ does not contain



(a) Two parts V_i and V_j inducing connected subgraphs. The part V_i includes two universal vertices.



(b) The universal vertex v_1 is moved from V_i to V_j. The two induced subgraphs are still connected and have the same orders.

Figure 3.13: Moving a universal vertex v_1 from a part inducing a connected subgraph with at least two universal vertices to another part. The original graph is in black and grey, its white vertices are universal, and the two subgraphs induced by the two parts are in black only.

any universal vertex. We may suppose that $i < j$ without loss of generality. We prove that we can exchange vertices between V_i and V_j in such a way that exactly one universal vertex is moved from V_i to V_j, and this without altering the sizes of V_i and V_j, nor the connectivity of the subgraphs of G they induce. Let $v_1 \in V_i$ and $v_2 \in V_j$ be arbitrary vertices of G such that v_1 is a universal vertex. Then note that $G[V_i \setminus \{v_1\}]$ remains connected since V_i includes at least two universal vertices. Besides, the subgraph $G[V_j \setminus \{v_2\} \cup \{v_1\}]$ is connected since v_1 is a universal vertex. It then follows that

$$(V_1 \text{ , } V_2 \text{ , ... , } V_{i-1} \text{ , } V_i \setminus \{v_1\} \cup \{v_2\} \text{ , } V_{i+1} \text{ , } V_{i+2} \text{ , ... , } V_{j-1} \text{ , } V_j \setminus \{v_2\} \cup \{v_1\} \text{ , } V_{j+1} \text{ , } V_{j+2} \text{ , ... , } V_p)$$

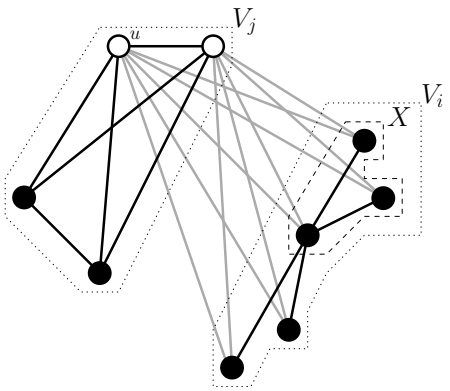
is another realization of π in G. Repeating the same argument as many times as necessary, we eventually get a realization of π in G in which each of the k biggest parts contains at most one universal vertex. This process is illustrated in Figure 3.13.

Suppose now that every part $V_1, V_2, ..., V_k$ of $(V_1, V_2, ..., V_p)$ includes at most one universal vertex. If each of $V_1, V_2, ..., V_k$ contains exactly one universal vertex, then the claim is proved. Otherwise, it means that a part different from $V_1, V_2, ..., V_k$ includes at least one universal vertex. Let V_j with $j \in \{k+1, k+2, ..., p\}$ be such a part whose set $U = \{u_1, u_2, ..., u_k\} \cap V_j$ of universal vertices is not empty, and let $u \in U$ be a universal vertex of V_j. By assumption there is another part V_i with $i \in \{1, 2, ..., k\}$ such that V_i does not include any universal vertex. We exchange vertices between V_i and V_j so that they still have size n_i and n_j, respectively, induce connected subgraphs of G, and u is the only universal vertex of V_j moved to V_i (so that we do not break the property that each of the k biggest parts includes at most one universal vertex).

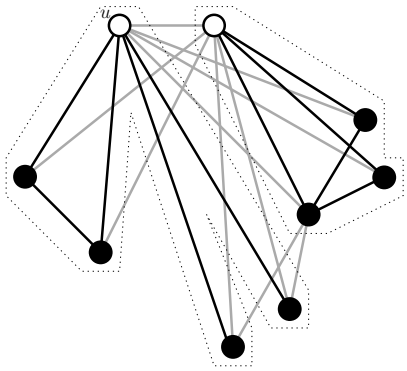
Recall that $n_i \geq n_j$. Then we can find a set $X \subseteq V_i$ such that $|X| = n_j - |U| + 1$ and $G[X]$ is connected, e.g. by applying a breadth-first search algorithm. Because u is a universal vertex, the subgraph of G induced by $V_i \setminus X \cup (V_j \setminus U) \cup \{u\}$ is connected. It then follows that

$$(V_1 \text{ , } V_2 \text{ , ... , } V_{i-1} \text{ , } V_i \setminus X \cup (V_j \setminus U) \cup \{u\} \text{ , } V_{i+1} \text{ , } V_{i+2} \text{ , ... , } V_{j-1} \text{ , } U \setminus \{u\} \cup X \text{ , } V_{j+1} \text{ , } V_{j+2} \text{ , ... , } V_p)$$

is a realization of π in G in which the part with size n_i now includes a universal vertex, while the part with size n_j has one less universal vertex (see Figure 3.14). Repeating the same arguments as many times as necessary, we eventually get a realization of π in G satisfying the conditions of the claim. ■



(a) Two parts V_i and V_j inducing connected subgraphs, with $|V_i| \geq |V_j|$. The part V_j includes two universal vertices. The subgraph induced by X is connected.



(b) The vertex u is moved from V_j to V_i, while X is moved from V_i to V_j. The two induced subgraphs are still connected and have the same orders.

Figure 3.14: Moving a universal vertex u from a part to a bigger part. The original graph is in black and grey, its white vertices are universal, and the two subgraphs induced by the two parts are in black only.

The kernel presented below relies on the following crucial lemma.

Lemma 3.29. *Let G be a graph with $k \geq 1$ universal vertices and order $n \geq k$, and $\pi = (n_1, n_2, ..., n_p)$ be an n-sequence with $n_1 \geq n_2 \geq ... \geq n_p$. If π is realizable in G, then every n-sequence $\pi' = \pi \setminus (n_i, n_{i_1}, n_{i_2}, ..., n_{i_{p'}}) \cup (n_i + n_{i_1} + n_{i_2} + ... + n_{i_{p'}})$, where $i \in \{1, 2, ..., k\}$ and $n_{i_1}, n_{i_2}, ..., n_{i_{p'}}$ are distinct arbitrary elements of $\pi \setminus (n_i)$, is realizable in G.*

Proof. The claim means that from a realization of π in G, we can deduce a realization in G of every n-sequence obtained from π by adding one big part size and additional part sizes. Since π is realizable in G by assumption, there exists a particular realization $(V_1, V_2, ..., V_p)$ of π in G such that each of V_1, V_2, ..., V_k includes one universal vertex, recall Lemma 3.28. There is thus one vertex in V_i neighbouring every other vertex of G, implying that the subgraph induced by $V_i \cup V_{i_1} \cup V_{i_2} \cup ... \cup V_{i_{p'}}$ is connected. It thus follows directly that

$$(V_1, V_2, ..., V_{i-1}, V_i \cup V_{i_1} \cup V_{i_2} \cup ... \cup V_{i_{p'}}, V_{i+1}, V_{i+2}, ..., V_p) \setminus (V_{i_1}, V_{i_2}, ..., V_{i_{p'}})$$

is a realization of

$$\pi' = (n_1, n_2, ..., n_{i-1}, n_i + n_{i_1} + n_{i_2} + ... + n_{i_{p'}}, n_{i+1}, n_{i+2}, ..., n_p) \setminus (n_{i_1}, n_{i_2}, ..., n_{i_{p'}})$$

in G. ■

So that we present the main result of this section, we first introduce the following notation.

Notation 3.30. Let $k \geq 1$ and $n \geq k$ be two integers. We denote by $\mathcal{U}_k(n)$ the set of graphs with k universal vertices and order n. We denote by $K_{\mathcal{U}_k}(n)$ the following set of n-sequences:

$$K_{\mathcal{U}_k}(n) = \{\pi : \|\pi\| = n \text{ and the greatest element value of } \pi \text{ appears at least } k+1 \text{ times}\}.$$

We prove below that $K_{\mathcal{U}_k}(n)$ is a kernel for $\mathcal{U}_k(n)$ no matter what are the values of k and n.

Theorem 3.31. *For every $k \geq 1$ and $n \geq k$, the set $K_{\mathcal{U}_k}(n)$ is a kernel for $\mathcal{U}_k(n)$.*

Proof. Let $G \in \mathcal{U}_k(n)$ be a graph with order n and k universal vertices with $k \geq 1$ and $n \geq k$ being fixed. We prove that G is arbitrarily partitionable if and only if $K_{\mathcal{U}_k}(n)$ is realizable in G. The necessary condition is obvious by the definition of an arbitrarily partitionable graph, so let us prove the sufficient condition.

Let us assume $K_{\mathcal{U}_k}(n)$ is realizable in G. We show that for every sequence $\pi \notin K_{\mathcal{U}_k}(n)$, we can deduce a sequence $\pi' \in K_{\mathcal{U}_k}(n)$ such that from a realization of π' in G we can obtain a realization of π in G. Assume $\pi = (n_1, n_2, ..., n_p)$ with $n_1 \geq n_2 \geq ... \geq n_p$. Since $\pi \notin K_{\mathcal{U}_k}(n)$, there are at most k occurrences of n_1 in π. The sequence π' is obtained from π by just replacing every element n_i with $i \in \{1, 2, ..., k\}$ of π by one occurrence of the element n_{k+1} and $n_i - n_{k+1}$ occurrences of the element 1, and transferring every other element n_i with $i \in \{k+1, k+2, ..., p\}$ to π' directly.

By construction, the biggest element of π' has value n_{k+1}, this element value appearing at least $k+1$ times in π'. The sequence π' thus belongs to $K_{\mathcal{U}_k}(n)$ and is realizable in G by assumption. Then Lemma 3.29 implies that from a realization of π' in G we can deduce a realization of π in G. More precisely, from a realization of π' in G obtained using Lemma 3.28, we can unify connected parts whose sizes result from the split of a single element of π in such a way that the resulting subgraphs are also connected. ■

Every kernel $K_{\mathcal{U}_k}(n)$ contains n-sequences consisting in one big element value n_1 appearing $\alpha \geq k+1$ times, and a partition of $n - \alpha n_1$ whose elements have value at most k. So that $K_{\mathcal{U}_k}(n)$ has polynomial size, we need the number of partitions of $n - \alpha n_1$ to be polynomial in n. This is asymptotically the case when $n - \alpha n_1$ is logarithmic in n, recall Theorem 1.2.

Corollary 3.32. *Let $k \geq 1$ and $n \geq k$ be two integers. If $k \geq \lceil \frac{n-\ln(n)-2}{2} \rceil$, then the kernel $K_{\mathcal{U}_k}(n)$ is polynomial.*

Proof. Regarding the terminology above, we have $\alpha \geq k+1$ and $n_1 \geq 2$. Since $n - \alpha n_1$ must be logarithmic in n, we want $n - \alpha n_1 \leq \ln(n)$, and hence $n - 2(k+1) \leq \ln(n)$. Solving the inequality, we end up with $k \geq \lceil \frac{n-\ln(n)-2}{2} \rceil$. For such a value of k, the size of $K_{\mathcal{U}_k}(n)$ is asymptotically $\mathcal{O}(n^3)$ since n_1 and α can take up to n values. ■

It is worth recalling that, in the proof of Corollary 3.32, the fact that $K_{\mathcal{U}_k}(n)$ has polynomial size mainly depends on the polynomiality of $n - \alpha n_1$. So the inequality $n - \alpha n_1 \leq \ln(n)$ can actually be replaced with $n - \alpha n_1 \leq \ln(n^q)$ for any fixed constant $q \geq 1$. For such a fixed value of q, we hence get a strengthening of Corollary 3.32, namely that $K_{\mathcal{U}_k}(n)$ is polynomial whenever $k \geq \lceil \frac{n-\ln(n^q)-2}{2} \rceil$ (though the size of $K_{\mathcal{U}_k}(n)$ increases to $\mathcal{O}(n^{q+2})$). But it is important noting that there is no constant threshold value of $q \leq n$ above which the amount $\lceil \frac{n-\ln(n^q)-2}{2} \rceil$ becomes upper-bounded by $\frac{n}{3}$.

3.3.3 Graphs made up of partitionable components

We herein focus on (k, ℓ)-compound graphs $G = C_{k,\ell}(G_1, G_2, ..., G_\ell)$ in which every component G_i is arbitrarily $(u_1^i, u_2^i, ..., u_k^i)$-partitionable for every $i \in \{1, 2, ..., \ell\}$, where $u_1, u_2, ..., u_k$ designate the roots of G. It is therefore understood that every compound graph G considered in this section has all of its components being arbitrarily P-partitionable, where P is the tuple of vertices glued to form G.

Although a (k, ℓ)-compound graph with the property above has strong local partition properties, i.e. it is made of ℓ (more than) arbitrarily partitionable components, it does not have to be arbitrarily partitionable. To be convinced of that statement, just note that a $(1, \ell)$-compound graph $C_{1,\ell}(P_1, P_2, ..., P_\ell)$, obtained by identifying one endvertex of $\ell \geq 5$ paths $P_1, P_2, ..., P_\ell$ with length at least 1, cannot be arbitrarily partitionable, recall Theorem 2.17.

We exhibit below a polynomial kernel of sequences for (k, ℓ)-compound graphs verifying $\ell \leq k$. The ideas used throughout are a rough generalization of the arguments used by Ravaux in [104] to prove that $K'_{\mathcal{T}}(n)$ is a kernel for tripodes. As in previous Section 3.3.2, we first prove some lemmas beforehand. An important remark to raise is that these lemmas are more general than needed for our purpose in the sense that they hold for all values of ℓ. So these lemmas could be reused in future works.

Observation 3.33. *Let $k \geq 1$ and $\ell \geq 1$ be two integers, and $G = C_{k,\ell}(G_1, G_2, ..., G_\ell)$ be a (k,ℓ)-compound graph with order $n \geq k$. Then every n-sequence $\pi = (n_1, n_2, ..., n_p)$ with size $p \leq k$ is realizable in G.*

Proof. Let $u_1, u_2, ..., u_k$ denote the root vertices of G. We start by picking some (possibly incomplete) connected parts of the realization in G_1. For this purpose, let

$$\pi_1 = (n_1^1, n_2^1, ..., n_p^1)$$

be an arbitrary partition of $|V(G_1)|$ into p parts such that $1 \leq n_i^1 \leq n_i$ for every $i \in \{1, 2, ..., p\}$. Since G_1 is arbitrarily $(u_1^1, u_2^1, ..., u_p^1)$-partitionable, there exists a $(u_1^1, u_2^1, ..., u_p^1)$-realization $(V_1^1, V_2^1, ..., V_p^1)$ of π_1 in G_1.

In order to get a realization of π in G, we now have to complete the part V_i^1 with $n_i' = n_i - n_i^1$ additional vertices for every $i \in \{1, 2, ..., p\}$. So, for every $i \in \{1, 2, ..., p\}$, let $(n_i^2, n_i^3, ..., n_i^\ell)$ be an arbitrary partition of $n_i' + \ell - 1$ such that $n_i^2, n_i^3, ..., n_i^\ell \geq 1$ and $n_1^j + n_2^j + ... + n_p^j = |V(G_j)| - (k-p)$ for every $j \in \{2, 3, ..., \ell\}$. Now define

$$\pi_i = (n_1^i, n_2^i, ..., n_p^i, 1, 1, ..., 1)$$

for every $i \in \{2, 3, ..., \ell\}$, where the value 1 appears $k - p$ times at the end of π_i. Since, for every $i \in \{2, 3, ..., \ell\}$, the component G_i is arbitrarily $(u_1^i, u_2^i, ..., u_k^i)$-partitionable and the sequence π_i is a $|V(G_i)|$-sequence, there exists a $(u_1^i, u_2^i, ..., u_k^i)$-realization $(V_1^i, V_2^i, ..., V_k^i)$ of π_i in G_i. Now observe that

$$(\bigcup_{i=1}^{\ell} V_1^i, \bigcup_{i=1}^{\ell} V_2^i, ..., \bigcup_{i=1}^{\ell} V_p^i)$$

is a realization of π in G. In particular, every resulting part $V_i^1 \cup V_i^2 \cup ... \cup V_i^\ell$ with $i \in \{1, 2, ..., p\}$ has size n_i and $G[V_i^1 \cup V_i^2 \cup ... \cup V_i^\ell]$ is connected since each of $G_1[V_i^1], G_2[V_i^2], ..., G_\ell[V_i^\ell]$ is connected and includes a "local copy" of the root vertex u_i. ■

From now on, it is thus understood that every considered sequence has sufficiently many elements, i.e. at least $k+1$ elements when dealing with a (k,ℓ)-compound graph. We now focus on the existence of a particular realization of every sequence which is realizable in a compound graph.

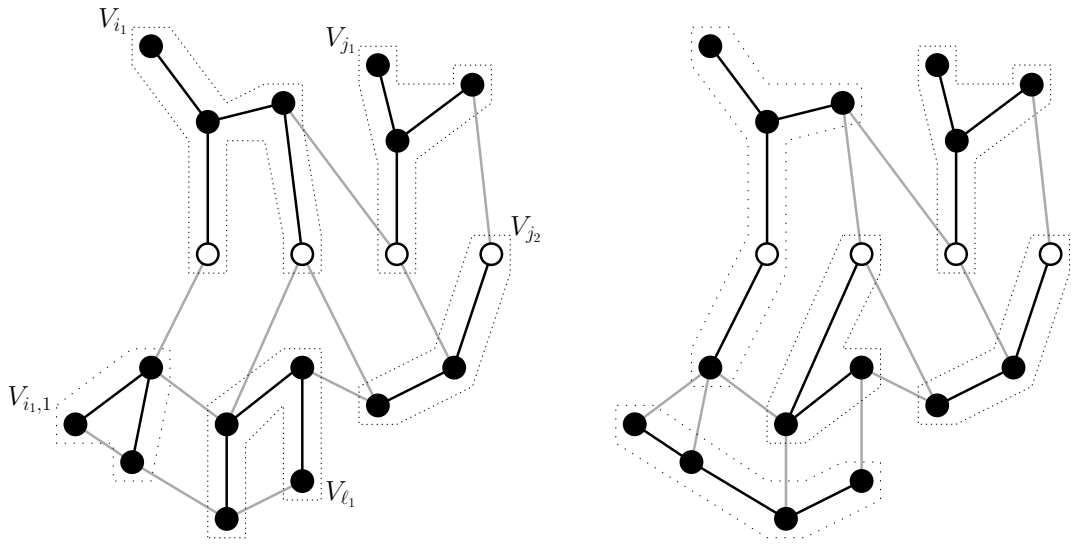
Lemma 3.34. *Let $k \geq 1$ and $\ell \geq 1$ be two integers, $G = C_{k,\ell}(G_1, G_2, ..., G_\ell)$ be a (k,ℓ)-compound graph with order $n \geq k$, and $\pi = (n_1, n_2, ..., n_p)$ be an n-sequence with $n_1 \geq n_2 \geq ... \geq n_p$. If π is realizable in G, then there exists a realization $(V_1, V_2, ..., V_p)$ of π in G such that each of $V_1, V_2, ..., V_k$ contains one root vertex.*

Proof. Let $R = \{u_1, u_2, ..., u_k\}$ denote the set of root vertices of G, and assume $(V_1, V_2, ..., V_p)$ is a realization of π in G. As in the proof of Lemma 3.28, we successively modify the realization $(V_1, V_2, ..., V_p)$ so that it eventually respects the conditions of the claim. If these conditions are already respected, then we are done. Otherwise, for each element n_i of π, let $n_i^1, n_i^2, ..., n_i^\ell$ be possibly null elements such that

$$n_i^j = |V_i \cap V(G_j)| \text{ for every } j \in \{1, 2, ..., \ell\}.$$

If we set $r_i = |V_i \cap R|$, then we have $n_i = (\sum_{j=1}^{\ell} n_i^j) - r_i(\ell - 1)$ for every $i \in \{1, 2, ..., p\}$. Besides, if $V_i \subseteq (V(G_j) \setminus \{u_1^j, u_2^j, ..., u_k^j\})$ for some $j \in \{1, 2, ..., \ell\}$, i.e. the part V_i only contains non-root vertices, then we have $n_i^j = n_i$ and $n_i^{j'} = 0$ for every $j' \neq j$. The original realization $(V_1, V_2, ..., V_p)$ of π in G can be then rewritten

$$(\bigcup_{i=1}^{\ell} V_1^i, \bigcup_{i=1}^{\ell} V_2^i, ..., \bigcup_{i=1}^{\ell} V_p^i),$$



(a) The part V_{i_1} includes two root vertices, while the part $V_{i_1,1}$ is included in the bottommost component.

(b) One root vertex from V_{i_1} is now included in the part with size $|V_{i_1,1}|$. The realization has locally changed (see notably the part with size $|V_{\ell_1}|$), but the inclusion of the parts in the two components is locally preserved.

Figure 3.15: Modifying a realization in a compound graph so that each part includes at most one root vertex. The original graph is in black and grey, its white vertices are its roots, and the two subgraphs induced by the parts are in black only.

where $(V_1^i, V_2^i, ..., V_p^i)$ is a realization of $(n_1^i, n_2^i, ..., n_p^i)$ in G_i for every $i \in \{1, 2, ..., \ell\}$. In particular, at most p of the subgraphs induced by the realization are connected because of the presence of some root vertices.

We start by modifying the realization so that $r_i \leq 1$ holds for every $i \in \{1, 2, ..., p\}$. If this condition is already met, then consider the next step. Otherwise, let $V_{i_1}, V_{i_2}, ..., V_{i_\alpha}$ be the parts of the realization including at least two root vertices, i.e. we have $r_{i_1}, r_{i_2}, ..., r_{i_\alpha} \geq 2$. For each such part V_i, let $V_{i,1}, V_{i,2}, ..., V_{i,r_i-1}$ be $r_i - 1$ distinct parts of the realization including no root vertex, i.e. $r_{i,1}, r_{i,2}, ..., r_{i,r_i-1} = 0$. These additional parts have to be chosen uniquely, i.e. the indices $(i_1, 1)$, $(i_1, 2)$, ... , $(i_1, r_{i_1} - 1)$, $(i_2, 1)$, $(i_2, 2)$, ... , $(i_2, r_{i_2} - 1)$, ... , $(i_\alpha, 1)$, $(i_\alpha, 2)$, ... , $(i_\alpha, r_{i_\alpha} - 1)$ must all be distinct. We modify the realization so that, for every $i \in \{i_1, i_2, ..., i_\alpha\}$, each of the parts $V_{i,1}, V_{i,2}, ..., V_{i,r_i-1}$ and V_i contains exactly one of the r_i root vertices which originally belonged to V_i (see Figure 3.15). For this purpose, we also need to take into account the parts of the original realization which include exactly one root vertex. These are denoted $V_{j_1}, V_{j_2}, ..., V_{j_\beta}$. By definition note that

$$(V_{i_1} \cup V_{i_2} \cup ... \cup V_{i_\alpha} \cup V_{j_1} \cup V_{j_2} \cup ... \cup V_{j_\beta}) \cap R = R,$$

and also that

$$\alpha + \beta + (r_{i_1} - 1) + (r_{i_2} - 1) + ... + (r_{i_\alpha} - 1) = k.$$

We finally denote by $\ell_1, \ell_2, ..., \ell_\gamma$ those $p - k$ indices not among $\{i_1, i_2, ..., i_\alpha\} \cup \{(i_1, 1), (i_1, 2), ..., (i_1, r_{i_1} - 1), (i_2, 1), (i_2, 2), ..., (i_2, r_{i_2} - 1), ..., (i_\alpha, 1), (i_\alpha, 2), ..., (i_\alpha, r_{i_\alpha} - 1)\} \cup \{j_1, j_2, ..., j_\beta\}$.

For every $i \in \{1, 2, ..., p\}$, we split n_i into ℓ elements $n_i^{1\prime}, n_i^{2\prime}, ..., n_i^{\ell\prime}$ as follows.

- If $i \in \{\ell_1, \ell_2, ..., \ell_\gamma\}$, then $n_i^{1\prime} = n_i^1, n_i^{2\prime} = n_i^2, ..., n_i^{\ell\prime} = n_i^\ell$. In particular, one of the ℓ resulting elements is equal to n_i, while all of the other elements are null.
- If $i \in \{i_1, i_2, ..., i_\alpha\}$, then $n_i^{1\prime} = n_i^1 - r_i + 1, n_i^{2\prime} = n_i^2 - r_i + 1, ..., n_i^{\ell\prime} = n_i^\ell - r_i + 1$.

- If $i \in \{(i_1,1)\ ,\ (i_1,2)\ ,\ ...\ ,\ (i_1,r_{i_1}-1)\ ,\ (i_2,1)\ ,\ (i_2,2)\ ,\ ...\ ,\ (i_2,r_{i_2}-1)\ ,\ ...\ ,\ (i_\alpha,1)\ ,\ (i_\alpha,2)\ ,\ ...\ ,\ (i_\alpha,r_{i_\alpha}-1)\}$, then there is a c such that $V_i \subset V(G_c)$. Then let $n_i^{j\prime} = n_i$ for $j = c$, or $n_i^{j\prime} = 1$ otherwise.
- If $i \in \{j_1, j_2, ..., j_\beta\}$, then $n_i^{1\prime} = n_i^1, n_i^{2\prime} = n_i^2, ..., n_i^{\ell\prime} = n_i^\ell$.

Using the resulting elements, we build ℓ new sequences $\pi'_1, \pi'_2, ..., \pi'_\ell$ where each such sequence π'_i is intended to be realized in G_i. Every such sequence π'_i is formally defined as follows:

$$\pi'_i = (n_{i_1}^i{}'\ ,\ n_{i_2}^i{}'\ ,\ ...\ ,\ n_{i_\alpha}^i{}'\ ,\ n_{i_1,1}^i{}'\ ,\ n_{i_1,2}^i{}'\ ,\ ...\ ,\ n_{i_1,r_{i_1}-1}^i{}'\ ,\ n_{i_2,1}^i{}'\ ,\ n_{i_2,2}^i{}'\ ,\ ...\ ,\ n_{i_2,r_{i_2}-1}^i{}'\ ,\ ...\ ,\ n_{i_\alpha,1}^i{}'\ ,\ n_{i_\alpha,2}^i{}'\ ,\ ...\ ,\ n_{i_\alpha,r_{i_\alpha}-1}^i{}'\ ,\ n_{j_1}^i{}'\ ,\ n_{j_2}^i{}'\ ,\ ...\ ,\ n_{j_\beta}^i{}'\ ,\ n_{\ell_1}^i{}'\ ,\ n_{\ell_2}^i{}'\ ,\ ...\ ,\ n_{\ell_\gamma}^i{}').$$

Note that according to how the original elements of π have been split, each resulting sequence π'_i sums up to $|V(G_i)|$. Since G_i is arbitrarily $(u_1^i, u_2^i, ..., u_k^i)$-partitionable by assumption, there exists a $(u_1^i, u_2^i, ..., u_k^i)$-realization $(V_1^{i\prime}, V_2^{i\prime}, ..., V_p^{i\prime})$ of π'_i in G_i. It then follows that

$$(\bigcup_{i=1}^{\ell} V_1^{i\prime}, \bigcup_{i=1}^{\ell} V_2^{i\prime}, ..., \bigcup_{i=1}^{\ell} V_p^{i\prime})$$

is a realization of π in G. In particular, each of its k first parts induces a connected subgraph since it is the union of parts which induce connected subgraphs themselves and include local copies of a same root vertex. Besides, each resulting part has the correct size regarding π, and each root vertex belongs to exactly one resulting subgraph. The desired assumptions are then met.

Let us now assume that $r_i \leq 1$ holds for every $i \in \{1, 2, ..., p\}$. If we have $r_1 = r_2 = ... = r_k = 1$, then the claim is proved. Otherwise, there is a root vertex u which belongs to a part V_x with $x \notin \{1, 2, ..., k\}$, while there exists a $y \in \{1, 2, ..., k\}$ such that $r_y = 0$ and, hence, we have $V_y \subseteq V(G_c) \setminus \{u_1^c, u_2^c, ..., u_k^c\}$ for some $c \in \{1, 2, ..., \ell\}$. Because every two roots of G belong to distinct parts of the realization, note that, because $n_y \geq n_x$, it is sufficient to modify the realization locally, by "swapping" V_x and V_y in G_c. This is done as follows. For every $i \in \{1, 2, ..., p\}$, let $U_i = V_i \cap V(G_c)$, and let $i_1, i_2, ..., i_k$ be the distinct indices of those parts U_i including one root vertex, and $u_{i_1}^c, u_{i_2}^c, ..., u_{i_k}^c$ the root vertices they respectively include. We additionally denote $U_{\ell_1}, U_{\ell_2}, ..., U_{\ell_\alpha}$ those parts different from U_x and $U_{i_1}, U_{i_2}, ..., U_{i_k}$.

We may assume that $x = i_k$. Since G_c is arbitrarily $(u_1^c, u_2^c, ..., u_k^c)$-partitionable, there exists a $(u_{i_1}^c, u_{i_2}^c, ..., u_{i_{k-1}}^c, u^c)$-realization $(U'_{i_1}\ ,\ U'_{i_2}\ ,\ ...\ ,\ U'_{i_{k-1}}\ ,\ V'_y\ ,\ V'_x\ ,\ U'_{\ell_1}\ ,\ U'_{\ell_2}\ ,\ ...\ ,\ U'_{\ell_\alpha})$ of $(|U_{i_1}|, |U_{i_2}|, ..., |U_{i_{k-1}}|, n_y - (|V_x| - |U_x|), n_x, |U_{\ell_1}|, |U_{\ell_2}|, ..., |U_{\ell_\alpha}|)$ in G_c. It then follows that

$$(V_{i_1} \setminus U_{i_1} \cup U'_{i_1}\ ,\ V_{i_2} \setminus U_{i_2} \cup U'_{i_2}\ ,\ ...\ ,\ V_{i_{k-1}} \setminus U_{i_{k-1}} \cup U'_{i_{k-1}}\ ,\ V_x \setminus U_x \cup V'_y\ ,\ V'_x\ ,\ U'_{\ell_1}\ ,\ U'_{\ell_2}\ ,\ ...\ ,\ U'_{\ell_\alpha})$$

is a realization of π in G in which u has been switched from the part with size n_x to the part with size n_y, and all of the other root vertices remain in the same parts (see Figure 3.16). Repeating the same argument as many times as needed, we eventually get a realization of π in G satisfying the conditions of the claim. ■

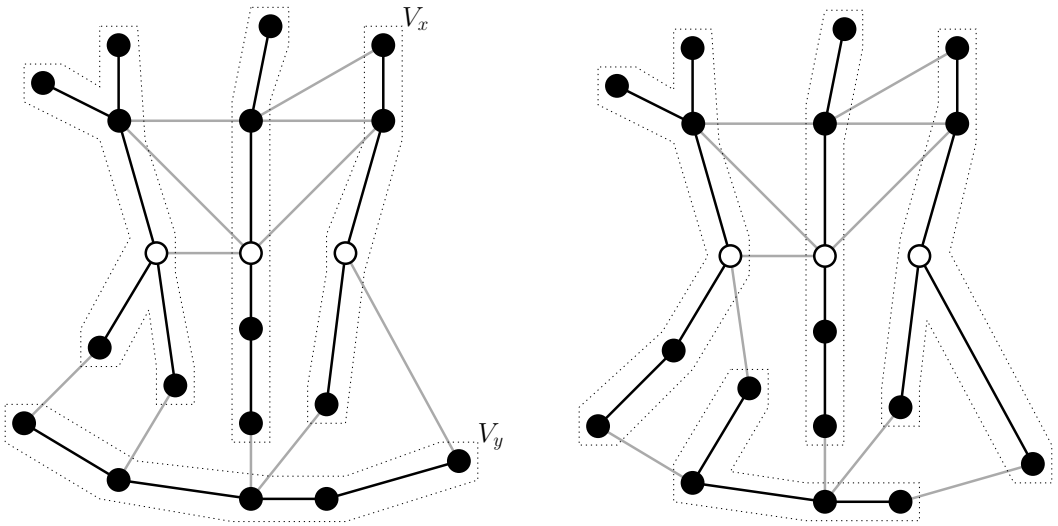
Prior to formulating the main result of this section, we introduce the following notation.

Notation 3.35. Let $k \geq 1$, $\ell \geq 1$, and $n \geq k$ be three integers. We denote by $\mathcal{C}_{k,\ell}(n)$ the set of (k,ℓ)-compound graphs with order n. We denote by $K_{\mathcal{C}_{k,\ell}}(n)$ the following set of n-sequences:

$$K_{\mathcal{C}_{k,\ell}}(n) = \{\pi : \|\pi\| = n \text{ and } |sp(\pi)| \leq 2k + 6\}.$$

We show below that $K_{\mathcal{C}_{k,\ell}}(n)$ is a polynomial kernel for (k,ℓ)-compound graphs with order n verifying $\ell \leq k$.

Theorem 3.36. *For every $k \geq 1$, $\ell \leq k$ and $n \geq k$, the set $K_{\mathcal{C}_{k,\ell}}(n)$ is a kernel for $\mathcal{C}_{k,\ell}(n)$.*



(a) The part V_x includes one root vertex while the part V_y does not include any, but $|V_y| > |V_x|$.

(b) The root vertex from V_x is now included in the part with size $|V_y|$. The realization has only changed locally, i.e. in the bottommost component.

Figure 3.16: Modifying a realization in a compound graph so that each of the biggest parts includes one root vertex. The original graph is in black and grey, its white vertices are its roots, and the two subgraphs induced by the parts are in black only.

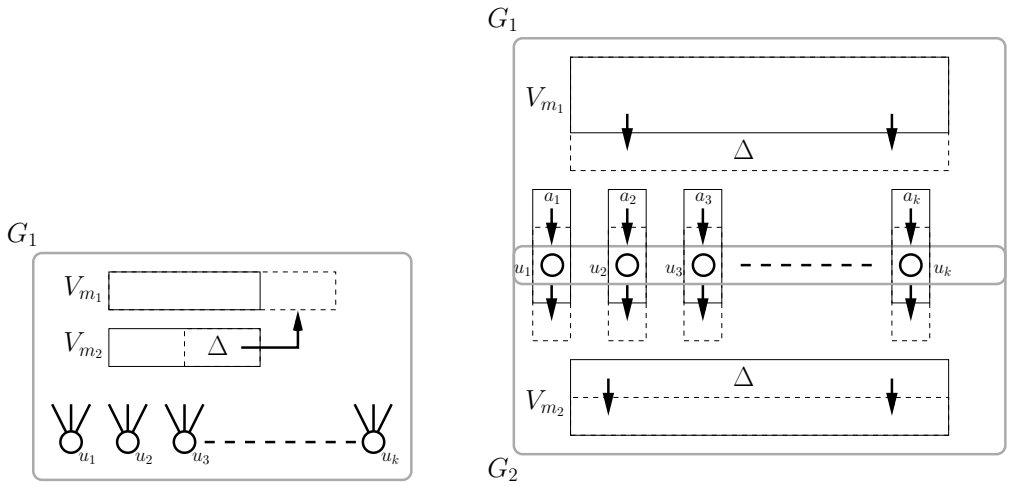
Proof. Let $G = C_{k,\ell}(G_1, G_2, ..., G_\ell)$ be a (k,ℓ)-compound graph with order $n \geq k$ whose parameters k and ℓ respect the conditions of the statement. We prove that G is arbitrarily partitionable if and only if $K_{\mathcal{C}_{k,\ell}}(n)$ is realizable in G.

Since every n-sequence is realizable in G under the assumption that G is arbitrarily partitionable, the necessary condition follows directly from the definitions. We thus narrow down our concern on the sufficient condition. Assume G is not arbitrarily partitionable. We need to prove that $K_{\mathcal{C}_{k,\ell}}(n)$ is not realizable in G. Since G is not arbitrarily partitionable, there is an n-sequence π not realizable in G. If $\pi \in K_{\mathcal{C}_{k,\ell}}(n)$, then we are done. Otherwise, i.e. $\pi \notin K_{\mathcal{C}_{k,\ell}}(n)$, we deduce another n-sequence $\pi' \in K_{\mathcal{C}_{k,\ell}}(n)$ which is not realizable in G, completing the proof. Equivalently, we may show that if π' is realizable in G, then so is π.

Since $\pi \notin K_{\mathcal{C}_{k,\ell}}(n)$, we have $sp(\pi) = \{s_1, s_2, ..., s_t\}$ with $t \geq 2k+7$ and $s_1 > s_2 > ... > s_t$. Among the at least $k+7$ elements in $\{s_{k+1}, s_{k+2}, ..., s_t\}$, there has to be at least four integers $s_{p_1} > s_{p_2} > s_{p_3} > s_{p_4}$ with the same parity and such that $s_{p_1} \geq k$. The sequence π' is obtained by replacing two elements with value s_{p_1} and s_{p_2}, respectively, of π by two elements with value $s_{p_m} = \frac{s_{p_1}+s_{p_2}}{2}$, which is an integer since s_{p_1} and s_{p_2} have the same parity. In doing so, note that we may have $|sp(\pi')| \geq |sp(\pi)|$, but repeating this procedure as many times as necessary, we get successive n-sequences which are all equivalent in terms of realizability in G (as shown below), converging to a sequence of $K_{\mathcal{C}_{k,\ell}}(n)$ since all these successive sequences are obtained by repeatedly dividing original elements of π, making them converge to 1.

Set $\Delta = s_{p_1} - s_{p_m} = s_{p_m} - s_{p_2}$. Let further $\pi' = (n_1, n_2, ..., n_p)$, with $n_1 \geq n_2 \geq ... \geq n_p$, be the n-sequence obtained from π by replacing two elements with value s_{p_1} and s_{p_2}, respectively, with two new occurrences n_{m_1} and n_{m_2} of s_{p_m}. By the choice of s_{p_1} and s_{p_2}, we have $n_1, n_2, ..., n_k > n_{m_1}, n_{m_2}$.

Assume π' is realizable in G. According to Lemma 3.34, there exists a $(u_1, u_2, ..., u_k)$-realization $(V_1, V_2, ..., V_p)$ of π' in G such that each of the root vertices $u_1, u_2, ..., u_k$ of G distinctly belongs to one of $V_1, V_2, ..., V_k$. We may assume that $u_1 \in V_1$, $u_2 \in V_2$, ..., $u_k \in V_k$ for the sake



(a) V_{m_1} and V_{m_2} are both included in G_1. We can directly transfer Δ vertices from V_{m_2} to V_{m_1}.

(b) V_{m_1} and V_{m_2} are included in G_1 and G_2, respectively. The parts containing the root vertices shrink in G_1 so that V_{m_1} can expend, while they expend in G_2, making V_{m_2} shrink in turn.

Figure 3.17: Exchanging Δ vertices between the parts V_{m_1} and V_{m_2} of a realization of π' in G. The original graph G is in black and grey, and its white vertices are its root. The dotted sections represent the transferred vertices.

of simplicity. As in the proof of Lemma 3.34, let us rewrite $(V_1, V_2, ..., V_p)$ as

$$(\bigcup_{i=1}^{\ell} V_1^i, \bigcup_{i=1}^{\ell} V_2^i, ..., \bigcup_{i=1}^{\ell} V_p^i),$$

where each subset V_i^j corresponds to $V_i \cap V(G_j)$. For each such subset, we further write $n_i^j = |V_i^j|$. By assumption, we have $V_{m_1} \subseteq (V(G_c) \setminus \{u_1^c, u_2^c, ..., u_k^c\})$ and $V_{m_2} \subseteq (V(G_{c'}) \setminus \{u_1^{c'}, u_2^{c'}, ..., u_k^{c'}\})$, where $c, c' \in \{1, 2, ..., \ell\}$ are possibly equal. We modify the realization in such a way that Δ vertices are moved from V_{m_1} to V_{m_2} in either direction, and this without altering the connectivity of the subgraphs these parts induce.

Case 1. *We have $c = c'$.*

In this situation the parts V_{m_1} and V_{m_2} are included in a same component G_c of G, say G_1. We obtain the realization of π in G as follows. Since G_1 is arbitrarily $(u_1^1, u_2^1, ..., u_k^1)$-partitionable, there exists a $(u_1^1, u_2^1, ..., u_k^1)$-realization $(V_1^{1\prime}, V_2^{1\prime}, ..., V_p^{1\prime})$ of

$$(n_1^1 \ , \ n_2^1 \ , \ ... \ , \ n_{m_1-1}^1 \ , \ n_{m_1}^1 + \Delta \ , \ n_{m_1+1}^1 \ , \ n_{m_1+2}^1 \ , \ ... \ , \ n_{m_2-1}^1 \ , \ n_{m_2}^1 - \Delta \ , \ n_{m_2+1}^1 \ , \ n_{m_2+2}^1 \ , \ ... \ , \ n_p^1)$$

in G_1. It then follows that

$$((\bigcup_{i=2}^{\ell} V_1^i) \cup V_1^{1\prime}, (\bigcup_{i=2}^{\ell} V_2^i) \cup V_2^{1\prime}, ..., (\bigcup_{i=2}^{\ell} V_p^i) \cup V_p^{1\prime})$$

is a realization of π in G. In particular, each part $(\bigcup_{j=2}^{\ell} V_i^j) \cup V_i^{1\prime}$ with $i \in \{1, 2, ..., k\}$ induces a connected subgraph of G since it is made up of several connected parts each containing one local copy of the root vertex u_i. This process is depicted in Figure 3.17.a.

Case 2. *We have $c \neq c'$.*

From now on, we suppose that the parts V_{m_1} and V_{m_2} are included in two different components G_c and $G_{c'}$, respectively. We may assume that $c = 1$ and $c' = 2$ without loss of generality. We distinguish two cases to deduce a realization of π in G.

Case 2.1. *We have* $\sum_{i=1}^{k}(n_i^1 - 1) \geq \Delta$ *or* $\sum_{i=1}^{k}(n_i^2 - 1) \geq \Delta$.

Assume $\sum_{i=1}^{k}(n_i^1 - 1) \geq \Delta$, and let $a_1, a_2, ..., a_k \geq 0$ be integers such that $a_i \leq n_i^1 - 1$ for every $i \in \{1, 2, ..., k\}$, and $\sum_{i=1}^{k} a_i = \Delta$. Note that we have both

$$(\sum_{i=1}^{p} n_i^1) - (\sum_{i=1}^{k} a_i) + \Delta = |V(G_1)|$$

and

$$(\sum_{i=1}^{p} n_i^2) + (\sum_{i=1}^{k} a_i) - \Delta = |V(G_2)|.$$

A realization of π in G is then obtained as follows. Since G_1 is arbitrarily $(u_1^1, u_2^1, ..., u_k^1)$-partitionable by assumption, there exists a $(u_1^1, u_2^1, ..., u_k^1)$-realization $(V_1^{1\prime}, V_2^{1\prime}, ..., V_p^{1\prime})$ of

$$(n_1^1 - a_1, n_2^1 - a_2, ..., n_k^1 - a_k, n_{k+1}^1, n_{k+2}^1, ..., n_{m_1-1}^1, n_{m_1}^1 + \Delta, n_{m_1+1}^1, n_{m_1+2}^1, ..., n_p^1)$$

in G_1. Similarly G_2 is arbitrarily $(u_1^2, u_2^2, ..., u_k^2)$-partitionable and then admits a $(u_1^2, u_2^2, ..., u_k^2)$-realization $(V_1^{2\prime}, V_2^{2\prime}, ..., V_p^{2\prime})$ of

$$(n_1^2 + a_1, n_2^2 + a_2, ..., n_k^2 + a_k, n_{k+1}^2, n_{k+2}^2, ..., n_{m_2-1}^2, n_{m_2}^2 - \Delta, n_{m_2+1}^2, n_{m_2+2}^2, ..., n_p^2).$$

It then follows that

$$((\bigcup_{i=3}^{\ell} V_1^i) \cup (V_1^{1\prime}, V_1^{2\prime}) \ , \ (\bigcup_{i=3}^{\ell} V_2^i) \cup (V_2^{1\prime}, V_2^{2\prime}) \ , \ ... \ , \ (\bigcup_{i=3}^{\ell} V_p^i) \cup (V_p^{1\prime}, V_p^{2\prime}))$$

is a realization of π in G according to the same arguments as those used so far (see Figure 3.17.b).

Case 2.2. *We have* $\sum_{i=1}^{k}(n_i^1 - 1) < \Delta$ *and* $\sum_{i=1}^{k}(n_i^2 - 1) < \Delta$.

In such a situation, we have $(\sum_{i=1}^{k} n_i^1 + n_i^2) - 2k < 2\Delta - 1$, with $n_1, n_2, ..., n_k > s_{p_1}$ by assumption. Recall also that $n_i = (\sum_{j=1}^{\ell} n_i^j) - \ell + 1$ for every $i \in \{1, 2, ..., k\}$, that $k \geq \ell \geq 2$ and $s_{p_1} \geq k$, and that $\Delta = s_{p_1} - s_{p_m}$ with $s_{p_m} = \frac{s_{p_1} + s_{p_2}}{2}$. Then we get

$$\begin{aligned}
\sum_{i=1}^{k}\sum_{j=3}^{\ell} n_i^j &= (\sum_{i=1}^{k} n_i) - (\sum_{i=1}^{k} n_i^1 + n_i^2) + k\ell - k \\
&> k s_{p_1} - (2\Delta + 2k - 1) + k\ell - k \\
&> k s_{p_1} + k\ell - 3k - 2\Delta + 1 \\
&> (k-2)s_{p_1} - k + 2s_{p_m} \\
&> (k-2)s_{p_1} - s_{p_1} + s_{p_1} \\
&> (k-2)s_{p_1}. \qquad (3.1)
\end{aligned}$$

By assumption we have $\ell \leq k$, and hence $\sum_{i=1}^{k} n_i^\alpha > s_{p_1}$ for an $\alpha \in \{3, 4, ..., \ell\}$ since otherwise we would get $\sum_{i=1}^{k}\sum_{j=3}^{\ell} n_i^j < (k-2)s_{p_1}$, contradicting Inequality (3.1).

Assume $\alpha = 3$ without loss of generality. Since

$$\sum_{i=1}^{k} n_i^3 > s_{p_1} = s_{p_m} + \Delta,$$

we can "move" the part with size m_1 of the realization from G_1 to G_3 as follows (see Figure 3.18). First deduce integers $a_1, a_2, ..., a_k$ such that we have $a_i \leq n_i^3 - 1$ for every $i \in \{1, 2, ..., k\}$, and $\sum_{i=1}^{k} a_i = s_{p_m}$. According to our terminology, since $V_{m_1} \subset V(G_1)$, recall that $n_{m_1}^1 = n_{m_1}$ and

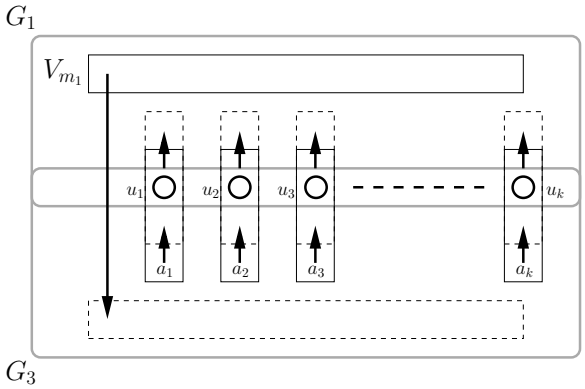


Figure 3.18: Moving the part V_{m_1} of a realization of π' in G from G_1 to G_3. To this end, the parts containing the root vertices shrink in G_3 so that V_{m_1} can be moved, while they expend in G_1 to fill the empty space. The original graph G is in black and grey, and its white vertices are its root. The dotted sections represent the transferred vertices.

$n^3_{m_1} = 0$. Now, since G_1 and G_3 are arbitrarily $(u^1_1, u^1_2, ..., u^1_k)$- and $(u^3_1, u^3_2, ..., u^3_k)$-partitionable, respectively, we can deduce a $(u^1_1, u^1_2, ..., u^1_k)$-realization $(V^{1\prime}_1, V^{1\prime}_2, ..., V^{1\prime}_p)$ of

$$(n^1_1 + a_1, n^1_2 + a_2, ..., n^1_k + a_k, n^1_{k+1}, n^1_{k+2}, ..., n^1_{m_1-1}, 0, n^1_{m_1+1}, n^1_{m_1+2}, ..., n^1_p)$$

in G_1, as well as a $(u^3_1, u^3_2, ..., u^3_k)$-realization $(V^{3\prime}_1, V^{3\prime}_2, ..., V^{3\prime}_p)$ of

$$(n^3_1 - a_1, n^3_2 - a_2, ..., n^3_k - a_k, n^3_{k+1}, n^3_{k+2}, ..., n^3_{m_1-1}, n_{m_1}, n^3_{m_1+1}, n^3_{m_1+2}, ..., n^3_p)$$

in G_3. We then get that

$$((\textstyle\bigcup_{i=1}^{\ell} V^i_1) \setminus (V^1_1, V^3_1) \cup (V^{1\prime}_1, V^{3\prime}_1)\ ,\ (\bigcup_{i=1}^{\ell} V^i_2) \setminus (V^1_2, V^3_2) \cup (V^{1\prime}_2, V^{3\prime}_2)\ ,\ ...\ ,\ (\bigcup_{i=1}^{\ell} V^i_p) \setminus (V^1_p, V^3_p) \cup (V^{1\prime}_p, V^{3\prime}_p))$$

is another realization of π' in G with

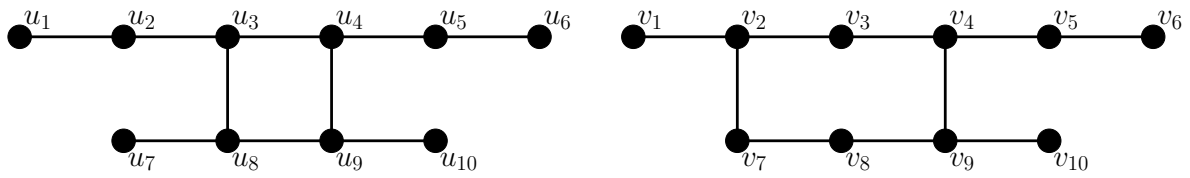
$$\sum_{i=1}^{k}(|V^{3\prime}_i| - 1) = \sum_{i=1}^{k}(n^3_i - 1) - s_{p_m} > \Delta,$$

and the parts with size m_1 and m_2 being now included in $V(G_3)$ and $V(G_2)$, respectively, hence meeting the conditions of Case 2.1. Applying the same strategy as for Case 2.1, we eventually get a realization of π in G. ■

To conclude this section, we point out that the size of every kernel $K_{\mathcal{C}_{k,\ell}}(n)$ verifying $\ell \leq k$ is polynomial regarding n.

Corollary 3.37. *For every $k \geq 1$, $\ell \leq k$, and $n \geq k$, the kernel $K_{\mathcal{C}_{k,\ell}}(n)$ is polynomial.*

Proof. Let k, ℓ, and n be fixed. Every n-sequence π of $K_{\mathcal{C}_{k,\ell}}(n)$ is only defined by the at most $2k + 6$ elements of its spectrum and the number of their occurrences in π. Since these parameters are all upper-bounded by n, we get that the size of $K_{\mathcal{C}_{k,\ell}}(n)$ is $\mathcal{O}(n^{2(2k+6)-1})$, that is $\mathcal{O}(n^{\mathcal{O}(k)})$. ■



(a) G_1. (b) G_2.

Figure 3.19: Two minimal arbitrarily partitionable non-tree graphs with order 10.

3.4 Minimal arbitrarily partitionable graphs

We herein investigate two aspects related to minimal arbitrarily partitionable graphs. Namely, we first investigate the minimum order of a minimal arbitrarily partitionable non-tree graph in Section 3.4.1. We then consider the maximum degree of minimal arbitrarily partitionable graphs in Section 3.4.2. Our most significant result is an improvement of Theorem 2.29, see Theorem 3.41.

3.4.1 Minimum order

Since every path P_n is a minimal arbitrarily partitionable graph, we directly get that, for every $n \geq 1$, there are minimal arbitrarily partitionable graphs with order n. Regarding minimal arbitrarily partitionable non-tree graphs, by Theorem 2.33 we directly get that these graphs can have arbitrarily large order too.

The only remaining question in the same vein is about the minimum order of minimal arbitrarily partitionable graphs which are not trees. The smallest such graphs we found are the two graphs with order 10 depicted in Figure 3.19. We briefly show below that these graphs indeed are minimal arbitrarily partitionable graphs.

Proposition 3.38. *The two graphs from Figure 3.19 are minimal arbitrarily partitionable graphs.*

Proof. Let G_1 and G_2 denote the graphs from Figures 3.19.a and 3.19.b, respectively. We use the terminology introduced in Figure 3.19 to deal with the vertices and edges of G_1 and G_2. The first step is to show that G_1 and G_2 are arbitrarily partitionable. We voluntarily skip this step since it can be done very easily due to the small orders of G_1 and G_2. Convenient tools for proving this are the facts that every traceable graph is arbitrarily partitionable, recall Observation 2.27, plus that plenty of small caterpillars are more than arbitrarily partitionable according to Theorem 2.20, and hence arbitrarily partitionable, recall Theorem 2.19. These tools have to be combined with the following straightforward observation.

Observation 3.39. *If, for some $\lambda \in \{1, 2, ..., |V(G)|\}$, there is a subset of λ vertices $V_\lambda \subseteq V(G)$ of a graph G such that $G[V_\lambda]$ is connected and $G - V_\lambda$ is arbitrarily partitionable, then every $|V(G)|$-sequence including an element with value λ is realizable in G.*

We now focus on the minimality of G_1 and G_2. Namely, we show that removing any edge from G_1 or G_2 results in a graph which is not arbitrarily partitionable. Because an arbitrarily partitionable graph has to be connected, we only have to consider the edges of G_1 and G_2 which belong to their cycle. Then the minimality of G_1 follows e.g. from the following arguments:

- $G_1 - \{u_3u_4\}$ does not admit a realization of $(2, 2, 2, 2, 2)$,
- $G_1 - \{u_4u_9\}$ does not admit a realization of $(5, 5)$,
- $G_1 - \{u_8u_9\}$ does not admit a realization of $(1, 3, 3, 3)$,

- $G_1 - \{u_3u_8\}$ does not admit a realization of $(5, 5)$.

Regarding G_2, first note that the only realization of $(2, 2, 2, 2, 2)$ in G_2 is

$$(\{v_1, v_2\}, \{v_3, v_4\}, \{v_5, v_6\}, \{v_7, v_8\}, \{v_9, v_{10}\}).$$

For this reason, removing either of v_3v_4 or v_7v_8 from G_2 results in a graph which does not admit a realization of $(2, 2, 2, 2, 2)$. Regarding the remaining candidate edges, the minimality of G_2 follows e.g. from the following arguments:

- $G_2 - \{v_2v_3\}$ does not admit a realization of $(5, 5)$,
- $G_2 - \{v_4v_9\}$ does not admit a realization of $(5, 5)$,
- $G_2 - \{v_8v_9\}$ does not admit a realization of $(1, 3, 3, 3)$,
- $G_2 - \{v_2v_7\}$ does not admit a realization of $(5, 5)$. ■

Regarding the main concern of this section, by Proposition 3.38 we get the following upper bound.

Corollary 3.40. *The minimum order of a minimal arbitrarily partitionable non-tree graph is upper-bounded by* 10.

3.4.2 Maximum degree

We herein improve Theorem 2.29 by showing that minimal arbitrarily partitionable graphs with order $n \geq 6$ have maximum degree at most $n - 3$. Note that the value of n for which this claim holds cannot be lowered since $P_3(1, 1, 2)$ has maximum degree $3 = |V(P_3(1, 1, 2))| - 2$ but is arbitrarily partitionable.

Theorem 3.41. *For every minimal arbitrarily partitionable graph G with $|V(G)| \geq 6$, we have $\Delta(G) \leq |V(G)| - 3$.*

Proof. We show that an arbitrarily partitionable graph G with $|V(G)| \geq 6$ cannot be minimal whenever $\Delta(G) \geq |V(G)| - 2$. Since G cannot be minimal arbitrarily partitionable under the assumption $\Delta(G) \geq |V(G)| - 1$ according to Theorem 2.29, the maximum degree of G is $|V(G)| - 2$. So there is a vertex u of G which is adjacent to all vertices $v_0, v_1, ..., v_{n-3}$ of G but one vertex w. Since G has to be connected, its vertex w is adjacent to, say, $v = v_0$.

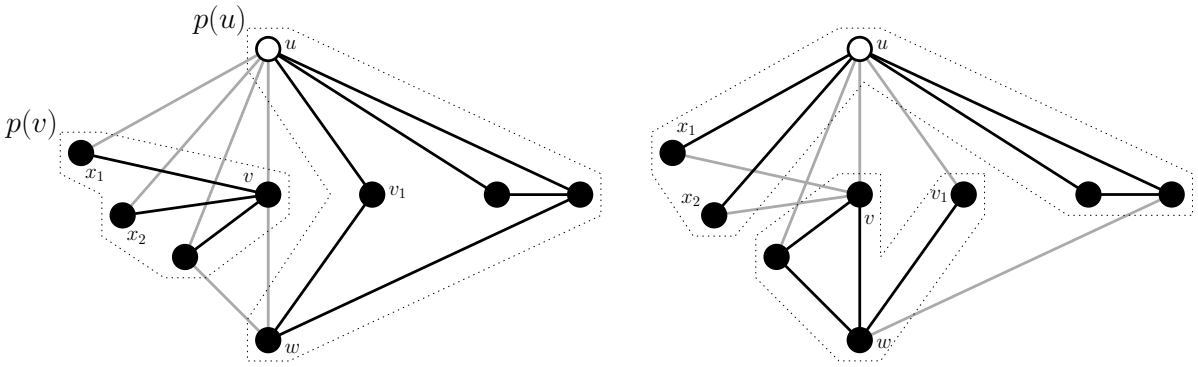
The proof reads as follows. We first show that if any edge from a set F is present in G, then we can modify every realization R of every $|V(G)|$-sequence π in G to get another realization R' of π in G such that some edges of G belong to none of the induced connected subgraphs. In other words, if at least one edge of F is present in G, then G cannot be minimal. The contradiction eventually comes from the fact that F is shown to be so important that G cannot be arbitrarily partitionable.

For the sake of simplicity, we make use of the following notation throughout.

Notation 3.42. For every vertex $z \in V(G)$, we denote $p(z)$ the part of R which contains z.

Claim 1. *G has no edge v_iw with $i \in \{1, 2, ..., n - 3\}$.*

Assume that v_1w is an edge of G without loss of generality. We show that, under this assumption, the edge uv_1 can be removed from G without altering the property of G of being arbitrarily partitionable. If $p(u) \neq p(v_1)$, then $R' = R$ remains a realization of π in $G - \{uv_1\}$. So, from now on, assume $p(u) = p(v_1)$. In case $p(u) = p(v_1) = p(v) = p(w)$, note that the



(a) x_1 and x_2 are two vertices of $p(v)$ such that $G[p(v) \setminus \{x_1, x_2\}]$ remains connected.

(b) x_1 and x_2 are moved to $p(u)$ while w and v_1 are moved to $p(v)$.

Figure 3.20: Exchanging vertices between two parts $p(u)$ and $p(v)$ of a realization in a graph G with maximum degree $|V(G)|-2$ so that the edge uv_1 becomes useless. The original graph G is in black and grey, its white vertex u has degree $|V(G)|-2$, and the subgraphs induced by the parts are in black only.

subgraph $G[p(u) \setminus \{uv_1\}]$ remains connected, so $R' = R$ is again a realization of π in $G - \{uv_1\}$. We then distinguish several subcases.

Case 1.1. *We have* $p(u) = p(v_1) = p(w) \neq p(v)$.

If $|p(v)| = 1$, then the realization R' of π in $G - \{uv_1\}$ is obtained by replacing the parts $p(u)$ and $p(v)$ with $p(u) \setminus \{v_1\} \cup \{v\}$ and $\{v_1\}$, respectively, in R. Similarly, if $|p(v)| = 2$, then R' can be obtained by replacing the parts $p(u)$ and $p(v)$ with $p(u) \setminus \{w, v_1\} \cup p(v)$ and $\{w, v_1\}$, respectively. In particular, note that $p(u) \setminus \{w, v_1\} \cup p(v)$ induces a connected subgraph since u neighbours all vertices of G but w.

Now assume $|p(v)| \geq 3$. Let x_1 and x_2 be two distinct vertices of $p(v) \setminus \{v\}$ such that $G[p(v) \setminus \{x_1, x_2\}]$ remains connected (x_1 and x_2 can be deduced by considering e.g. two successive leaves of a tree spanning $G[p(v)]$). In particular, these two vertices x_1 and x_2 neighbour u. Then R' is obtained by replacing $p(u)$ and $p(v)$ with $p(u) \setminus \{w, v_1\} \cup \{x_1, x_2\}$ and $p(v) \setminus \{x_1, x_2\} \cup \{w, v_1\}$, respectively. This process is illustrated in Figure 3.20.

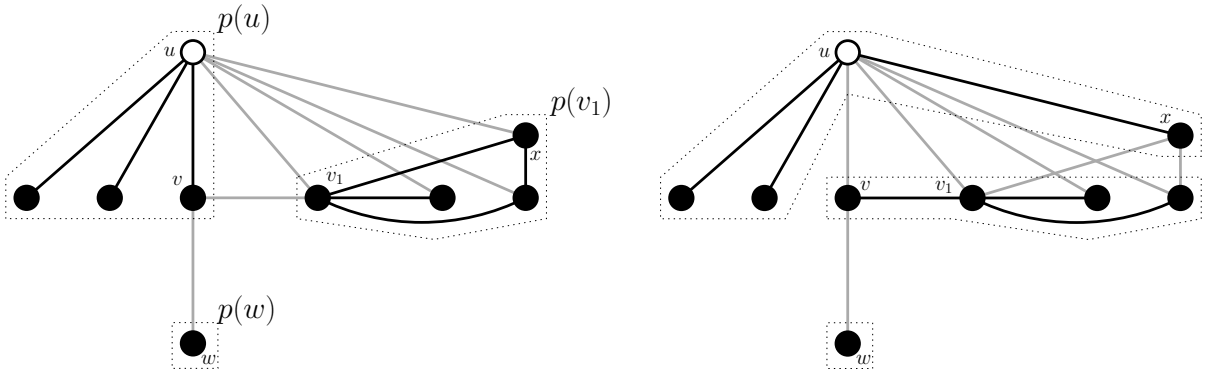
Case 1.2. *We have* $p(u) = p(v) = p(v_1) \neq p(w)$.

As previously, in case $|p(w)| = 1$ the realization R' is directly obtained from R by replacing the parts $p(u)$ and $p(w)$ with $p(u) \setminus \{v_1\} \cup \{w\}$ and $\{v_1\}$, respectively. Now for the general case, i.e. $|p(w)| \geq 2$, consider a vertex $x \in p(w) \setminus \{w\}$ such that $G[p(w) \setminus \{x\}]$ remains connected. Since x is necessarily a neighbour of u, we obtain R' by just replacing $p(u)$ and $p(w)$ with $p(u) \setminus \{v_1\} \cup \{x\}$ and $p(w) \setminus \{x\} \cup \{v_1\}$, respectively.

Case 1.3. *We have* $p(u) = p(v_1) \neq p(v)$ *and* $p(u) = p(v_1) \neq p(w)$.

In case $p(v) = p(w)$, consider a vertex $x \in p(w) \setminus \{w\}$ such that $G[p(w) \setminus \{x\}]$ remains connected (in particular $x = v$ when $|p(w)| = 2$). By our choice of x, note that u and x are adjacent vertices. Then the realization R' is obtained from R by replacing the parts $p(u)$ and $p(w)$ with $p(u) \setminus \{v_1\} \cup \{x\}$ and $p(w) \setminus \{x\} \cup \{v_1\}$, respectively.

If $p(v) \neq p(w)$ and $|p(w)| \geq 2$, then just proceed as above. Now if $|p(v)| = 1$, then we just get R' by replacing the parts $p(u)$ and $p(v)$ with $p(u) \setminus \{v_1\} \cup \{v\}$ and $\{v_1\}$, respectively, in R. The last case we have to handle is the one where $|p(w)| = 1$ and $|p(v)| \geq 2$. In such a situation, let x be a vertex of $p(v) \setminus \{v\}$ such that $G[p(v) \setminus \{x\}]$ remains connected. Clearly we have $ux \in E(G)$. Then R' is obtained by replacing $p(u)$, $p(v)$, and $p(w)$ with $p(u) \setminus \{v_1\} \cup \{x\}$, $p(v) \setminus \{x\} \cup \{w\}$, and $\{v_1\}$, respectively, in R.



(a) x is a vertex of $p(v_1)$ such that $G[p(v_1) \setminus \{x\}]$ remains connected.

(b) x is moved to $p(u)$ while v is moved to $p(v_1)$.

Figure 3.21: Exchanging vertices between two parts $p(u)$ and $p(v_1)$ of a realization in a graph G with maximum degree $|V(G)|-2$ so that the edge uv becomes useless. The original graph G is in black and grey, its white vertex u has degree $|V(G)|-2$, and the subgraphs induced by the parts are in black only.

Claim 2. *G has no edge vv_i with $i \in \{1, 2, ..., n-3\}$.*

According to Claim 1, it is understood that $N(w) = \{v\}$ from now on. Assume vv_1 belongs to G without loss of generality. We show below that we can modify R to get another realization R' of π in G which does not use the edge uv. As very first cases, note that we may directly choose $R = R'$ whenever $p(u) \neq p(v)$ (the edge uv is already useless for R) or $p(u) = p(v) = p(v_1)$ (since $G[p(u) \setminus \{uv\}]$ remains connected in such a situation).

We thus now assume that $p(u) = p(v) \neq p(v_1)$. First suppose $|p(v_1)| = 1$. By our assumptions, recall that either $|p(w)| = 1$ or $p(w) = p(v)$ holds throughout. In the first case, we obtain R' by just replacing $p(u)$ and $p(v_1)$ with $p(u) \setminus \{v\} \cup \{v_1\}$ and $\{v\}$, respectively, in R. In the second case, by replacing $p(u)$ and $p(v_1)$ with $p(u) \setminus \{w\} \cup \{v_1\}$ and $\{w\}$, respectively, in R, all conditions are now met to apply one strategy we used to deal with one of the very first cases above.

Second assume $|p(v_1)| = 2$, say $p(v_1) = \{v_1, v_2\}$ without loss of generality. In case $|p(w)| = 1$, we obtain R' from R by just replacing $p(u)$, $p(v_1)$, and $p(w)$ with $p(u) \setminus \{v\} \cup \{v_2\}$, $\{v, w\}$, and $\{v_1\}$, respectively. Now if $p(w) = p(v)$, then R' can be obtained by replacing $p(u)$ and $p(v_1)$ with $p(u) \setminus \{v, w\} \cup \{v_1, v_2\}$ and $\{v, w\}$, respectively, in R.

Third assume $|p(v_1)| \geq 3$. If on the one hand we have $|p(w)| = 1$, then let x be a vertex from $p(v_1) \setminus \{v_1\}$ such that $G[p(v_1) \setminus \{x\}]$ remains connected. Then R' is obtained by replacing the parts $p(u)$ and $p(v_1)$ from R with $p(u) \setminus \{v\} \cup \{x\}$ and $p(v_1) \setminus \{x\} \cup \{v\}$, respectively. In particular, $p(u) \setminus \{v\} \cup \{x\}$ induces a connected subgraph since u and x are adjacent vertices (see Figure 3.21). On the other hand, assume $p(w) = p(v)$. Then deduce two vertices x_1 and x_2 of $p(v_1) \setminus \{v_1\}$ such that $G[p(v_1) \setminus \{x_1, x_2\}]$ remains connected. R' is then obtained by replacing the parts $p(u)$ and $p(v_1)$ with $p(u) \setminus \{v, w\} \cup \{x_1, x_2\}$ and $p(v_1) \setminus \{x_1, x_2\} \cup \{v, w\}$.

Claim 3. *G has no edge v_iv_j with $i \neq j$ and $i, j \in \{1, 2, ..., n-3\}$.*

Under the assumptions we made so far, we have $N(w) = \{v\}$ and $N(v) = \{u, w\}$. Assume v_1v_2 is an edge of G without loss of generality. We show that we can freely remove the edge uv_1 of G without altering its property of being arbitrarily partitionable. For this purpose, we again prove that we can modify every realization R of π in G to get another realization R' which does not use uv_1.

At very first, note that if $p(u) \neq p(v_1)$, then R remains a realization of π in $G - \{uv_1\}$. Similarly, if $p(u) = p(v_1) = p(v_2)$, then the subgraph $G[p(u) \setminus \{uv_1\}]$ remains connected, so $R' = R$ is a correct realization.

Finally consider $p(u) = p(v_1) \neq p(v_2)$. We choose a vertex $x \in p(v_2)$ as follows. In case $|p(v_2)| = 1$, set $x = v_2$. Otherwise, i.e. $|p(v_2)| \geq 2$, choose, as x, a vertex in $p(v_2) \setminus \{v_2\}$ such that $G[p(v_2) \setminus \{x\}]$ remains connected. According to our assumption, we have $ux \in E(G)$. Since $N(v_2) \cup \{v_2\} \subset N(u)$, note that the partition R' obtained from R by replacing the parts $p(u)$ and $p(v_2)$ with $p(u) \setminus \{v_1\} \cup \{x\}$ and $p(v_2) \setminus \{x\} \cup \{v_1\}$ is a realization of π in G.

To sum up, if G is a minimal arbitrarily partitionable graph, then the only edges of G are vw and uv_i for every $i \in \{0, 1, 2, ..., n-3\}$. So G is isomorphic to a multipode $P_k(1, 1, ..., 1, 2)$ with $k \geq 4$ since $|V(G)| \geq 6$. But such a tree cannot be arbitrarily partitionable since it admits no (possibly quasi-) perfect matching, a contradiction. ■

3.5 Cartesian products

As explained in introductory Section 2.3, the properties of being arbitrarily partitionable and Hamiltonian are related. So one way of investigation is to consider any important result or open question regarding Hamiltonian graphs, and see what can be said in a very same vein regarding partitionable graphs.

We herein consider the following open question about the presence of an Hamiltonian cycle in every Cartesian product of two Hamiltonian graphs.

Conjecture 3.43 ([74])**.** *If two graphs G and H are Hamiltonian, then so is $G \,\square\, H$.*

In our context, it seems legitimate to address the following.

Conjecture 3.44. *If two graphs G and H are arbitrarily partitionable, then so is $G \,\square\, H$.*

We support Conjecture 3.44 by showing it to hold whenever one operand has order at most 4.

Theorem 3.45. *Conjecture 3.44 holds whenever $|V(H)| \leq 4$.*

Theorem 3.45 follows from the study of the following weaker conjecture, as all arbitrarily partitionable graphs on at most four vertices are traceable.

Conjecture 3.46. *If a graph G is arbitrarily partitionable and another graph H is traceable, then $G \,\square\, H$ is arbitrarily partitionable.*

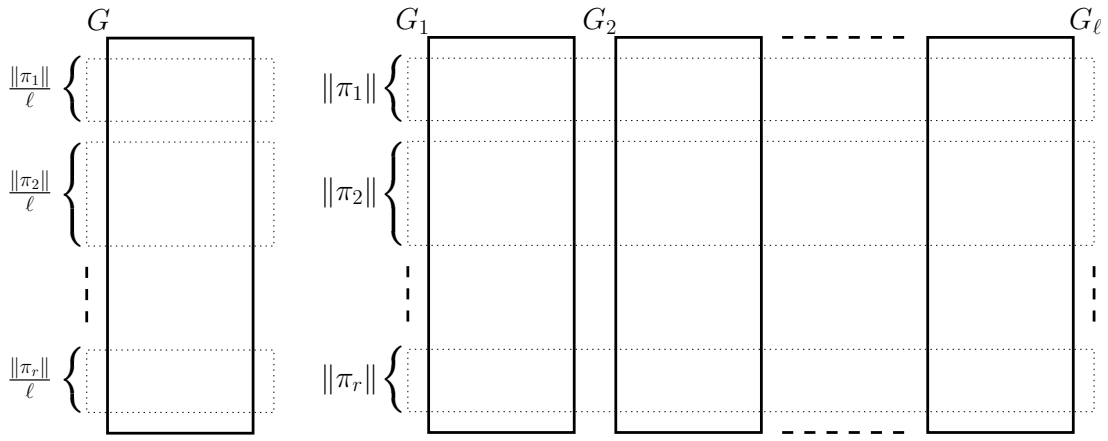
According to Observation 2.27, recall that it is sufficient to consider situations where H is a path to prove Conjecture 3.46. We hence herein focus on Cartesian products of the form $G \,\square\, P_\ell$. Our proof of Theorem 3.45 mainly relies on the following crucial lemma and proposition.

Lemma 3.47. *Let $\ell \geq 2$ be a positive integer, and $\pi = (n_1, n_2, ..., n_p)$ be a sequence of positive integers such that $\|\pi\| \equiv 0 \pmod{\ell}$. If $p > \ell$, then π can be partitioned into two non-empty sequences π_1 and π_2 such that $\|\pi_1\| \equiv 0 \pmod{\ell}$ and $\|\pi_2\| \equiv 0 \pmod{\ell}$.*

Proof. If π contains an element n_i such that $n_i \equiv 0 \pmod{\ell}$, then if suffices to consider $\pi_1 = (n_i)$ and $\pi_2 = \pi \setminus (n_i)$. Suppose then that $n_i \not\equiv 0 \pmod{\ell}$ for every $i \in \{1, 2, ..., p\}$.

For every $j \in \{1, 2, ..., p\}$, let $s_j = \sum_{i=1}^{j} n_i$ be the sum of the first j elements of π. If there exist two indices i_1 and i_2, with $i_1 < i_2$, such that $s_{i_1} \equiv s_{i_2} \pmod{\ell}$, then $\pi_1 = (n_{i_1+1}, n_{i_1+2}, ..., n_{i_2})$ and $\pi_2 = \pi \setminus \pi_1$ satisfy our conditions. Since the size of the sequence $s = (s_1, s_2, ..., s_p)$ is strictly greater than ℓ, the number of distinct residues modulo ℓ, there must exist two elements of s which are congruent modulo ℓ. This ends the proof. ■

Proposition 3.48. *Let $\ell \geq 2$ be a positive integer such that $H \,\square\, P_\ell$ is arbitrarily ℓ'-partitionable for every connected graph H and integer $\ell' \leq \ell$. If a graph G is arbitrarily partitionable, then so is $G \,\square\, P_\ell$.*



(a) π is partitioned into sequences $\pi_1, \pi_2, ..., \pi_r$ such that $\|\pi_i\| \equiv 0 \pmod{\ell}$ and $|\pi_i| \leq \ell$ for every $i \in \{1, 2, ..., r\}$. The sequence $\alpha = (\frac{\|\pi_1\|}{\ell}, \frac{\|\pi_2\|}{\ell}, ..., \frac{\|\pi_r\|}{\ell})$ is then realized in G.

(b) The realization of α in G is extended to $G \,\square\, P_\ell$. A realization of π is then obtained by independently realizing every sequence π_i in one resulting subgraph.

Figure 3.22: Realizing a sequence π with at least $\ell + 1$ elements in a Cartesian product $G \,\square\, P_\ell$ involving an arbitrarily partitionable graph G.

Proof. Let G be an arbitrarily partitionable graph with order n, and π be an $n\ell$-sequence. If π has size at most ℓ, then π is realizable in $G \,\square\, P_\ell$ according to our assumption since G is connected. Let us now suppose that π has size at least $\ell + 1$. By repeatedly applying Lemma 3.47, the sequence π can be partitioned into non-empty sequences $\pi_1, \pi_2, ..., \pi_r$ such that $\|\pi_i\| \equiv 0 \pmod{\ell}$ and $|\pi_i| \leq \ell$ for every $i \in \{1, 2, ..., r\}$.

Now, for every $i \in \{1, 2, ..., r\}$, let $\alpha_i = \frac{\|\pi_i\|}{\ell}$. Since $\|\pi_i\| \equiv 0 \pmod{\ell}$, the number α_i is an integer. Clearly $\alpha = (\alpha_1, \alpha_2, ..., \alpha_r)$ is an n-sequence and, because G is arbitrarily partitionable, admits a realization $(V_1, V_2, ..., V_r)$ in G. Set

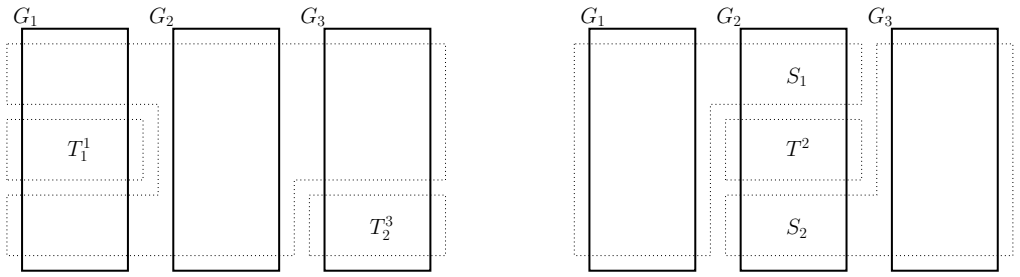
$$U_i = \bigcup_{j=1}^{\ell} (V_i)^j$$

for every $i \in \{1, 2, ..., r\}$. Then $U_1 \cup U_2 \cup ... \cup U_r$ performs a partition of $V(G \,\square\, P_\ell)$ such that each part U_i has size $\alpha_i \ell = \|\pi_i\|$, see Figure 3.22. Since each subgraph $G[V_i]$ is connected, we infer by our assumptions that $(G \,\square\, P_\ell)[U_i]$ admits a realization of π_i. To obtain a realization of π in $G \,\square\, P_\ell$, we then just have to consider independent realizations of $\pi_1, \pi_2, ..., \pi_r$ in the subgraphs of $G \,\square\, P_\ell$ induced by $U_1, U_2, ..., U_r$, respectively. ■

Theorem 3.45 is proved by showing that, for every $\ell \in \{1, 2, 3, 4\}$, the Cartesian product $G \,\square\, P_\ell$ is arbitrarily ℓ'-partitionable for every connected graph G and $\ell' \leq \ell$ so that Proposition 3.48 is applicable. Since $G \,\square\, P_\ell$ is obviously arbitrarily 1-partitionable when G is connected, we focus on the values $\ell \in \{2, 3, 4\}$ below.

Proposition 3.49. *For every connected graph G and integer $\ell \geq 2$, the Cartesian product $G \,\square\, P_\ell$ is arbitrarily 2-partitionable.*

Proof. The result follows from Theorem 2.1 since $G \,\square\, P_\ell$ is 2-connected under our assumptions on G and ℓ. ■



(a) Case 1. (b) Case 2.

Figure 3.23: Realizations in $G \,\square\, P_3$ deduced in the proof of Proposition 3.50.

Consequently, we may now restrict our concern on sequences with size at least 3 when showing that $G \,\square\, P_\ell$ is arbitrarily ℓ'-partitionable for every connected graph G and $\ell' \leq \ell$.

Proposition 3.50. *For every connected graph G, the Cartesian product $G \,\square\, P_3$ is arbitrarily* 3*-partitionable.*

Proof. Let $\pi = (n_1, n_2, n_3)$ be a $3n$-sequence, with $n_1 \geq n_2 \geq n_3$, where $n = |V(G)|$. We may assume that $n_1 > n$ and $n_3 < n$ since otherwise $n_1 = n_2 = n_3$ and $(V(G^1), V(G^2), V(G^3))$ is a realization of π in $G \,\square\, P_3$. We distinguish two cases depending on the value of n_2.

Case 1. $n > n_2$.

Let $T_1, T_2 \subset V(G)$ be subsets (with possibly non-empty intersection) of vertices such that $G[T_1]$ and $G[T_2]$ are connected subgraphs of G with order n_2 and n_3, respectively. Observe then that

$$(V(G \,\square\, P_3) \setminus (T_1^1 \cup T_2^3), T_1^1, T_2^3)$$

is a realization of π in $G \,\square\, P_3$. In particular $(G \,\square\, P_3) - (T_1^1 \cup T_2^3)$ is connected since every vertex from $(V(G^1) \cup V(G^3)) \setminus (T_1^1 \cup T_2^3)$ has a neighbour in $V(G^2)$, see Figure 3.23.a.

Case 2. $2n > n_2 \geq n$.

In such a situation

$$(V(G^1) \cup S_1, V(G^3) \cup S_2, T^2)$$

is a realization of π, where $T \subset V(G)$ is chosen is such a way that $G[T]$ is a connected subgraph with order n_3, and S_1 and S_2 are disjoint subsets of vertices from $V(G^2) \setminus T^2$ chosen arbitrarily but satisfy $|S_1| = n_1 - n$ and $|S_2| = n_2 - n$. The important thing to note is that each vertex of S_1 or S_2 has a neighbour in both $V(G^1)$ and $V(G^3)$, so the subgraphs $(G \,\square\, P_3)[V(G^1) \cup S_1]$ and $(G \,\square\, P_3)[V(G^3) \cup S_2]$ are connected, see Figure 3.23.b. ■

Proposition 3.51. *For every connected graph G, the Cartesian product $G \,\square\, P_4$ is arbitrarily* 3*- and* 4*-partitionable.*

Proof. We refer to $|V(G)|$ as n throughout. First consider a $4n$-sequence $\pi = (n_1, n_2, n_3)$ with size 3, where $n_1 \geq n_2 \geq n_3$. We distinguish three cases for deducing a realization of π in $G \,\square\, P_4$.

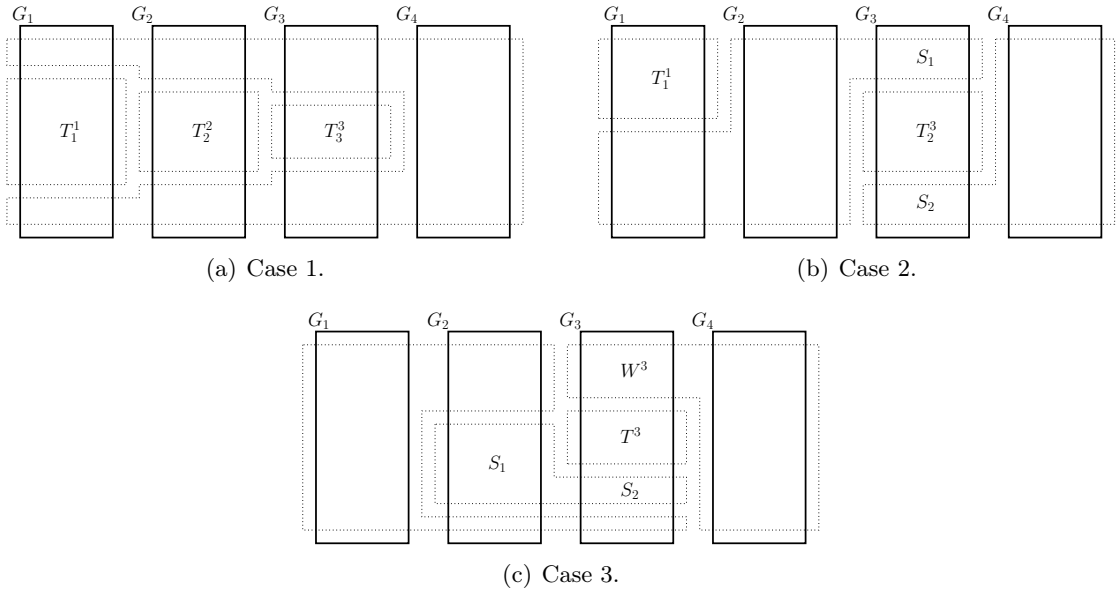
Case 1. $n > n_2$.

For the same reasons as in the proof of Proposition 3.50,

$$(V(G \,\square\, P_4) \setminus (T_1^1 \cup T_2^4), T_1^1, T_2^4)$$

is a realization of π, where $T_1, T_2 \subset V(G)$ are chosen in such a way that they induce connected graphs with order n_2 and n_3, respectively, of G.

Case 2. $2n > n_2 \geq n > n_3$.



(a) Case 1.

(b) Case 2.

(c) Case 3.

Figure 3.24: Realizations in $G \,\square\, P_4$ deduced in the proof of Proposition 3.51.

First deduce a realization (V_1, V_2, V_3) of $(n_1, n_2 - n, n_3)$ in $(G \,\square\, P_4)[V(G^2) \cup V(G^3) \cup V(G^4)]$ as in the proof of Proposition 3.50. This realization can then be extended to a realization of π in $G \,\square\, P_4$ by simply considering $(V_1, V_2 \cup V(G^1), V_3)$.

Case 3. $2n > n_2 \geq n_3 \geq n$.

In such a situation, let $T_1 \subset V(G)$ be a subset of vertices such that $G[T_1]$ is a connected subgraph of G with order $n - (n_3 - n)$. Now choose a vertex $u \in T_1$, and let $T_2 \subset V(G)$ be a subset of vertices such that $G[T_2]$ is a connected subgraph of G on $n_2 - |T_1|$ vertices including u. Consider now the partition

$$(V(G^1) \cup (V(G^2) \setminus T_2^2), T_2^2 \cup T_1^3, V(G^4) \cup (V(G^3) \setminus T_1^3))$$

of $V(G \,\square\, P_4)$. The three subsets of this partition induce connected subgraphs since every vertex of $V(G^2) \setminus T_2^2$ has a neighbour in $V(G^1)$, every one from $V(G^3) \setminus T_1^3$ is adjacent to a vertex in $V(G^4)$, and T_1^3 and T_2^2 induce connected subgraphs connected via the edge u^2u^3. This vertex-partition is thus a realization of π in $G \,\square\, P_4$.

Now suppose that $\pi = (n_1, n_2, n_3, n_4)$ is a $4n$-sequence with size 4 satisfying $n_1 \geq n_2 \geq n_3 \geq n_4$. Again we may assume that $n_1 > n$ and $n_4 < n$ since otherwise we have $n_1 = n_2 = n_3 = n_4 = n$ and $(V(G^1), V(G^2), V(G^3), V(G^4))$ is a straight realization of π in $G \,\square\, P_4$. We deduce a realization of π in $G \,\square\, P_4$ by distinguishing the following three cases.

Case 1. $n > n_2$.

Let $T_3 \subseteq T_2 \subseteq T_1 \subset V(G)$ be subsets inducing connected subgraphs of G with order n_4, n_3, and n_2, respectively. Then

$$(V(G \,\square\, P_4) \setminus (T_1^1 \cup T_2^2 \cup T_3^3), T_1^1, T_2^2, T_3^3)$$

is a realization of π in $G \,\square\, P_4$ since every vertex from $V(G^1) \setminus T_1^1$ has a neighbour in $V(G^2) \setminus T_2^2$, every vertex from $V(G^2) \setminus T_2^2$ is adjacent to a vertex of $V(G^3) \setminus T_3^3$, and every vertex of $V(G^3) \setminus T_3^3$ has a neighbour in $V(G^4)$, see Figure 3.24.a.

Case 2. $n_2 \geq n > n_3$.

In such a situation, we have $3n > n_1 + n_3 \geq 2n$ and $n < n_2 + n_4 \leq 2n$. Let $T_1, T_2 \subset V(G)$ be subsets inducing connected subgraphs with order n_3 and n_4, respectively, in G. Consider the partition

$$(V(G^2) \cup (V(G^1) \setminus T_1^1) \cup S_1, V(G^4) \cup S_2, T_1^1, T_2^3)$$

of $V(G \,\square\, P_4)$, where the subsets S_1 and S_2 are chosen arbitrarily from $V(G^3) \setminus T_2^3$ in such a way that $S_1 \cap S_2 = \emptyset$, and $|S_1| = n_1 + n_3 - 2n$ and $|S_2| = n - |S_1| - n_4 = n_2 - n$ hold. Observe then that this partition is a realization of π in $G \,\square\, P_4$ since every vertex of $(V(G^1) \setminus T_1^1) \cup S_1$ has a neighbour in $V(G^2)$ and all vertices of S_2 are adjacent to some vertices in $V(G^4)$, see Figure 3.24.b.

Case 3. $n_2 \geq n_3 \geq n$.

Let $T \subset V(G)$ be a subset of vertices inducing a connected subgraph with order n_4 in G, and u be a vertex of G which is not a cut vertex. Now construct two subsets $S_1 \subset V(G^1) \cup V(G^2)$ and $S_2 \subset V(G^3)$ of vertices as follows. Start by putting u^1 and u^2 into S_1, and repeatedly add an arbitrary new vertex s^2 of $V(G^2)$ neighbouring a vertex in S_1 to S_1. Also add s^3 to S_2 in case $s^3 \notin T^3$. Continue the procedure until $|S_1| + |S_2| \geq n_2$. Observe that once the procedure is over, we have $|S_1| + |S_2| \in \{n_2, n_2 + 1\}$. In case $|S_1| + |S_2| = n_2 + 1$, remove the vertex u^1 from S_1 so that $S_1 \cup S_2$ is a subset with size n_2. Finally, let W^3 be a subset of $n_3 - n$ arbitrary vertices of $V(G^3) \setminus (S^2 \cup T^3)$.

Then

$$((V(G^1) \cup V(G^2) \cup V(G^3)) \setminus (S_1 \cup S_2 \cup T^3 \cup W^3), S_1 \cup S_2, V(G^4) \cup W^3, T^3)$$

is a realization of π in $G \,\square\, P_4$, see Figure 3.24.c. In particular, the four subgraphs induced by this partition of $V(G \,\square\, P_4)$ are connected since $(G \,\square\, P_4)[S_1]$ is connected by construction, every vertex from S_2 has a neighbour in S_1, every vertex from $V(G^3) \setminus (S_2 \cup T^3 \cup W^3)$ is adjacent to one vertex in $V(G^2) \setminus S_1$ which is itself adjacent to one vertex in $V(G^1)$, and every vertex in W^3 neighbours a vertex in $V(G^4)$. Note further that the choice of u is crucial for the connectedness of the subgraph with order n_1. ■

3.6 Conclusion and open questions

Throughout Section 3.1 we have exhibited new conditions regarding both the sequence π or the graph G under which Realizable Sequence remains NP-complete. Our results in this vein are not surprising as Realizable Sequence had already shown up to be difficult in general, recall Theorem 2.10. However these results remain of interest for a better understanding of the problem. Notably, Theorems 3.10 and 3.11 highlight the fact that Realizable Sequence remains NP-complete when restricted to classes of graphs known to be convenient regarding other notoriously hard problems. Our investigations have led to a proof of the tightness of Theorem 2.1 in Section 3.1.3: although every k-connected graph can be partitioned into at most k connected subgraphs, the same problem becomes NP-complete in general whenever at least $k+1$ connected parts are each required to contain a specific preassigned vertex.

Regarding our results concerning Realizable Sequence, it would be interesting trying to push Theorem 3.18 forwards, namely to consider the following.

Question 3.52. *What is the greatest $c \geq \frac{n}{3}$ such that* Realizable Sequence *remains NP-hard when restricted to graphs with order n and up to c universal vertices?*

Due to Theorem 2.36, the correct answer to Question 3.52 should be closer to $\frac{n}{3}$ than to $\frac{n}{2}$. Moving back to our results on Realizable Sequence, the most incomplete result is Theorem 3.14. So we address the following.

Question 3.53. *Is* Realizable Sequence *NP-complete when restricted to connected 3-regular graphs or connected k-regular graphs with $k \geq 4$ even?*

We believe the cases $k \geq 6$ even of Question 3.53 could be proved by slightly modifying the reduction scheme used in the proof of Theorem 3.14. For the remaining cases $k = 3$ and $k = 4$, things seem harder though, for the following reasons. Our reduction relies on the fact that we are able to construct a regular graph which has a lot of cut vertices (so that the graph can be disconnected easily) whose removal results in a "large" number of components, but this seems harder to ensure notably for $k = 4$. Another annoying fact is that our proof strongly relies on the parity of both B and $s_i - 1$ of the instance $<A, B, s>$ of 3-Partition. Although we can ensure one of these two parameters has given parity, recall Observation 1.54, we cannot be sure of the parity of the second parameter, which forces us to distinguish several cases and hence to define different kinds of gadgets.

Imagining Question 3.53 is true, we would have that Realizable Sequence remains NP-complete when restricted to k-connected graphs (recall Theorem 3.12) or k-regular graphs for every $k \geq 3$. So the next interesting question would be to investigate the consequences on Realizable Sequence of fixing both the connectivity q and the regularity ℓ of G. We formulate this question as follows.

Question 3.54. *For every $\ell \geq 3$, what is the maximum integer $q \leq \ell$ (if such exists) such that* Realizable Sequence *remains NP-complete when restricted to q-connected ℓ-regular graphs ?*

Note that Theorem 3.14 implies that $q \geq 1$ for every $\ell \geq 5$ odd. Question 3.53, if true, would imply the same for the remaining values of ℓ. Conjecture 2.37 would seem to indicate that $k \leq \ell - 1$ for every $\ell \geq 3$.

Although we have pointed out, in Section 3.2, some evidences regarding the complexity status of Arbitrarily Partitionable Graph, we still do not know whether this problem is complete for some complexity class. As this question remains one of the most important open algorithmic questions related to arbitrarily partitionable graphs, we raise the following.

Question 3.55. *Is there a complexity class C such that* Arbitrarily Partitionable Graph *is C-complete?*

One direction we considered regarding Question 3.55 is based on the fact that every traceable graph is arbitrarily partitionable, recall Observation 2.34. Basically, for showing the NP-hardness of Arbitrarily Partitionable Graph, one could modify one NP-hardness reduction from an NP-hard problem Π to Hamiltonian Path so that:

- for every positive instance of Π, the reduction provides a traceable reduced graph,
- for every negative instance of Π, the reduction provides a reduced graph which is not only not traceable, but also not arbitrarily partitionable.

So that the second condition above is fulfilled, by definition we would like a particular sequence π to be not realizable in the reduced graph whenever the original instance of Π is negative. As an evidence, this such sequence π should rather have a small spectrum so that the chances that a realization exists are weaker. Note that this sequence π cannot be $\pi = (2, 2, ..., 2)$ since otherwise we would get that the problem of deciding whether a graph has a perfect matching is NP-hard, which would contradict Theorem 2.11.

Although we did not manage to design such a reduction, we believe that such an approach could be a promising one. We would suggest to combine it with a reduction given by Karger, Motwani and Ramkumar in [80] establishing the NP-hardness of the problem of deciding whether $\varsigma(G) \geq |V(G)| - c$ for a graph G and every fixed positive constant $c \geq 0$.

In Section 3.3, we have exhibited three polynomial kernels of sequences for complete multipartite graphs, graphs with about a half universal vertices, and compound graphs made up of

partitionable components. Our kernel $K_{\mathcal{M}_k}(n)$ yields a polynomial-time algorithm for recognizing arbitrarily partitionable complete multipartite graphs (the existence of such an algorithm is not surprising though due to the density of most of these graphs), while our other two kernels establish the membership of Arbitrarily Partitionable Graph to NP when restricted to graphs of $\mathcal{U}_k(n)$ and $\mathcal{C}_{k,\ell}(n)$ with $\ell \leq k$ (this condition is understood below). Another interesting point regarding $K_{\mathcal{U}_k}(n)$ is that it is based on a new sequence invariant, namely the number of occurrences of the biggest element value.

It would be of interest to focus on whether Realizable Sequence is in P when restricted to either $\mathcal{U}_k(n)$ and sequences of $K_{\mathcal{U}_k}(n)$, or $\mathcal{C}_{k,\ell}(n)$ and sequences in $K_{\mathcal{C}_{k,\ell}}(n)$. If this were the case, then we would get that Arbitrarily Partitionable Graph is in P when restricted to $\mathcal{U}_k(n)$ or $\mathcal{C}_{k,\ell}(n)$, respectively. Though these restrictions are very specific, they would increase our knowledge of Arbitrarily Partitionable Graph. So we ask the following.

Question 3.56. *Is* Realizable Sequence *in* P *when restricted to either* $\mathcal{U}_k(n)$ *and sequences of* $K_{\mathcal{U}_k}(n)$*, or* $\mathcal{C}_{k,\ell}(n)$ *and sequences in* $K_{\mathcal{C}_{k,\ell}}(n)$ *with* $\ell \leq k$*?*

It is worth noting that the NP-hardness of Realizable Sequence when restricted to, say, $\mathcal{U}_k(n)$ and sequences of $K_{\mathcal{U}_k}(n)$ would not imply the NP-hardness of Arbitrarily Partitionable Graph when restricted to graphs of $\mathcal{U}_k(n)$. This fact would indeed not refute the existence of another polynomial kernel $K'_{\mathcal{U}_k}(n)$ for $\mathcal{U}_k(n)$ different from $K_{\mathcal{U}_k}(n)$ such that Realizable Sequence is in P when restricted to $\mathcal{U}_k(n)$ and sequences of $K'_{\mathcal{U}_k}(n)$.

Although our kernel $K_{\mathcal{C}_{k,\ell}}(n)$ is a generalization of the kernel $K'_{\mathcal{T}}(n)$ described in Theorem 2.13, it does not hold for tripodes (which are $(1,3)$-compound graphs) since, for $k = 1$, Theorem 3.36 ensures that $K_{\mathcal{C}_{k,\ell(n)}}$ is a kernel for $\ell = 1$ only (and is hence not interesting in this case as $(k,1)$-compound graphs are arbitrarily partitionable). Note however that the condition "$\ell \leq k$" in the statement of Theorem 3.36 cannot be strengthened as otherwise the existence of the parameter α mentioned in the proof would not be guaranteed. The proof given by Ravaux in [104] for establishing that $K'_{\mathcal{T}}(n)$ is a kernel for tripodes is quite similar to our proof of Theorem 3.36, except that the very special case of a compound graph with exactly three components permits to consider parts with size s_{p_3} or s_{p_4} and use them to deduce the realization.

Actually the proof from [104] could be easily generalized to show that $K'_{\mathcal{T}}(n)$ is actually a polynomial kernel for $(1,3)$-compound graphs. It is however not clear what a polynomial kernel for the remaining compound graphs could look like.

Question 3.57. *Is there a polynomial kernel for* $\mathcal{C}_{k,\ell}(n)$*, where* $(k,\ell) \neq (1,3)$ *and* $\ell > k$*?*

In a more general context, we are still far from a general polynomial kernel for all graphs, if such exists, which would confirm Conjecture 2.12. Because every polynomial kernel for a family $\mathcal{F}$ of graphs is highly dependent of the structure of the members of $\mathcal{F}$, it might actually be the case that every polynomial kernel only holds for a very restricted family of graphs. For this reason, searching polynomial kernels for all graphs could be a quite difficult and tedious task, unless we can imagine a graph invariant according to which we can exhibit polynomial kernels for such or such value of this invariant. In this scope, maybe one interesting starting point could be to exhibit polynomial kernels for graphs with given density. Indeed, as an evidence, and our results just confirm it, the denser a graph G is, the more probable we can modify a realization of a $|V(G)|$-sequence in G to deduce realizations of other $|V(G)|$-sequences in G. So we address the following.

Question 3.58. *For fixed* d*, is there a polynomial kernel for graphs with order* n *and density* d*?*

Our opinion regarding Question 3.58 is that such polynomial kernels should at least exist for large values of d, i.e. graphs with density at least t with t being close to 1. This value t would typically be a density threshold under which a graph does not have to have a predictable structure (making improbable the existence of a systematic polynomial kernel), and above which some local dense structures should necessarily appear (which would be useful to modify realizations).

We have considered two aspects related to minimal arbitrarily partitionable graphs in Section 3.4. On the one hand, we have exhibited two small minimal arbitrarily partitionable non-tree graphs in Section 3.4.1. We believe these two graphs, which have order 10, are among the smallest minimal arbitrarily partitionable graphs in terms of order. Said differently, we strongly believe the following to be true.

Conjecture 3.59. *Minimal arbitrarily partitionable non-tree graphs have order at least* 10.

Conjecture 3.59 should be easy to tackle by hand since the number of graphs with order at most 9 is quite small. In this scope, the spanning argument of Observation 2.27 should be of great use to reject a lot of candidates. Typically, as soon as a graph with order at most 9 is too dense, it is likely to be spanned by a path or another arbitrarily partitionable graph (like e.g. a caterpillar). Although tedious, such an exhaustive method should be successful.

On the other hand, we have slightly improved Theorem 2.29, see Theorem 3.41. Our proof of Theorem 3.41 relies on the fact that the structure of every graph G with maximum degree $|V(G)|-2$ can be initially described as a hierarchy over its vertex set $V(G)$ involving three levels L_0, L_1 and L_2, making up a partition of $V(G)$, with the following properties:

- $|L_0| = 1$,
- $|L_1| = |V(G)| - 2$ and every vertex of L_1 is joined to the vertex in L_0,
- $|L_2| = 1$ and the vertex in L_2 is joined only to some vertices of L_1.

It does not appear obvious to say whether our proof of Theorem 3.41 can be improved to show e.g. the following general conjecture.

Conjecture 3.60. *For every $c \geq 1$, there is a positive integer $f(c)$ such that every arbitrarily partitionable graph G with maximum degree $\Delta(G) - c$ and order at least $f(c)$ is not minimal.*

Theorems 2.29 and 3.41 state that $f(1) = 4$ and $f(2) = 6$, respectively. Generalizing our proof of Theorem 3.41 for showing e.g. that $f(3)$ exists does not seem obvious. Indeed, note that, under the assumption $c = 3$, the initial description of the structure of G as a hierarchy of vertex levels does not have to be unique. Two main hierarchies may indeed arise, namely

- $|L_0| = 1$,
- $|L_1| = |V(G)| - 3$ and every vertex of L_1 is joined to the vertex in L_0,
- $|L_2| = 1$ and the vertex in L_2 is joined to some vertices of L_1,
- $|L_3| = 1$ and the vertex in L_3 is joined to the vertex in L_2;

or

- $|L_0| = 1$,
- $|L_1| = |V(G)| - 3$ and every vertex of L_1 is joined to the vertex in L_0,
- $|L_2| = 2$ and the vertices of L_2 are joined to some vertices of L_1.

So basically, any improvement of our proof of Theorem 3.41 for particular cases of Conjecture 3.59 would have to make the distinction between the different structural hierarchies. Since the number of such descriptions increases as c increases, things quickly seem hard to handle.

Further directions seem of interest regarding minimal arbitrarily partitionable graphs. Towards Conjecture 2.31, it would be interesting investigating how much *locally* dense can a minimal arbitrarily partitionable graph be. The notion of local density can notably be regarded as a clique measure. Since all known minimal arbitrarily partitionable non-tree graphs have girth at least 4, we have no candidate attesting the following is true.

Question 3.61. *For every $k \geq 3$, is there a minimal arbitrarily partitionable graph G with $\omega(G) = k$?*

It is also worth mentioning that we did not find a minimal arbitrarily partitionable 2-connected graph, which would confirm Question 2.32, namely because 2-connected graphs are already dense enough to allow the modification of every realization so that some edges are not used. So in order to prove that Question 2.32 is false, which we believe is the good direction, we propose to first investigate the existence of several realizations of a same sequence in a graph which is connected enough. As a first step, we address the following conjecture regarding sequences with spectrum of size 1.

Conjecture 3.62. *If a $|V(G)|$-sequence $\pi = (k, k, ..., k)$, where $|V(G)| \equiv 0 \pmod{k}$, is realizable in a 2-connected graph G, then there is another realization of π in G.*

The case $k = 2$ of Conjecture 3.62 is a direct corollary of a result by Beineke and Plummer given in [27], wherein it is shown that every 2-connected graph having a perfect matching, i.e. a realization of the sequence $(2, 2, ..., 2)$, admits at least two such. A more general result is actually proved in [27], namely that every k-connected graph admitting a perfect matching admits at least k perfect matchings. This result confirms that the more connected a graph G is, the more freedom we have to modify and "move" the parts of a realization of a sequence in G, and hence the more chances we have to find an edge which is useless for partitioning G.

In Section 3.5, we have considered one still holding conjecture about Hamiltonicity in the context of arbitrarily partitionable graphs. More precisely, we have raised Conjecture 3.46 as a special case of Conjecture 3.44, and have answered it in the affirmative partially, recall Theorem 3.45. Our proof could be extended to more cases by proving that $H \,\square\, P_\ell$ is arbitrarily 2-, 3-, ..., ℓ-partitionable whenever H is connected for values of $\ell \geq 5$ (what we have actually proved for smaller values of ℓ). But proving this seems to get more and more difficult as ℓ increases due to the fact that the connectivity of $H \,\square\, P_\ell$ does not increase (in case H has connectivity 1, the Cartesian product $G \,\square\, P_\ell$ has connectivity only 2 for every $\ell \geq 2$).

Chapter 4

Preassignable arbitrarily partitionable graphs

We herein investigate the notion of k-preassignable arbitrarily partitionable graphs. After giving some elementary properties of these graphs in Section 4.1, we start, in Section 4.2, by exhibiting k-preassignable arbitrarily partitionable graphs for every $k \geq 1$, namely powers of paths or cycles.

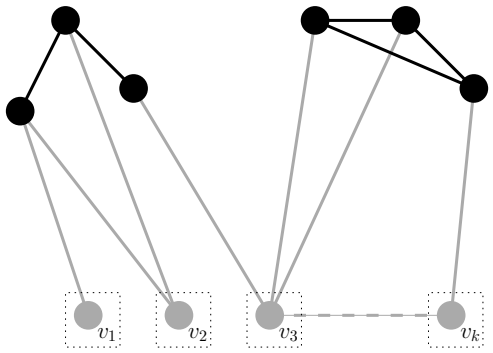
We then consider some structural properties of k-preassignable arbitrarily partitionable graphs. We first investigate the minimum size of such a graph in Section 4.3. In particular, for every $k \geq 1$ and $n \geq k+1$, we show that k-preassignable arbitrarily partitionable graphs on n vertices have size at least $\lceil \frac{n(k+1)}{2} \rceil$, this lower bound being tight. We then consider the relationship between k-preassignable arbitrarily partitionable graphs and Hamiltonian graphs in Section 4.4. As a main result, we prove that the longest paths of a k-preassignable arbitrarily partitionable graph can be arbitrarily small (compared to the order of these graphs).

As for arbitrarily partitionable graphs, see previous Section 3.5, we also consider Cartesian products of graphs involving preassignable arbitrarily partitionable graphs (see Section 4.5). In this scope, we suspect $G \,\square\, P_\ell$ to be k-preassignable arbitrarily partitionable whenever G is k-preassignable arbitrarily partitionable. We support this conjecture by showing it to hold for the very first case $k = 1$.

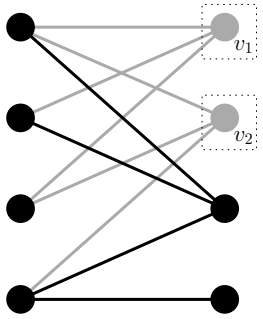
The results from Section 4.2 were obtained jointly with Baudon, Przybyło and Woźniak and were published in [18]. All results from Section 4.3 were obtained with Baudon and Sopena and have been submitted for publication [19]. Our results from Section 4.4 have also been submitted for publication [32]. Finally, our results from Section 4.5 were obtained in collaboration with Baudon, Kalinowski, Marczyk, Przybyło and Woźniak and were published in [16].

4.1 Preliminary remarks and properties

Since a k-preassignable arbitrarily partitionable graph is a graph we can partition into connected subgraphs even when k of its vertices are preassigned, every such graph has order at least k. Besides, it is easily seen that every graph with order exactly k is k-preassignable arbitrarily



(a) A graph with no $(v_1, v_2, ..., v_k)$-realization of $(1, 1, ..., 1, 6)$.



(b) Removing vertices from a bipartite graph so that it admits no perfect matching.

Figure 4.1: Removing vertices (in grey only) from graphs (in black and grey) so that they cannot be partitioned.

partitionable since every graph with order n is partitionable into n parts inducing connected subgraphs (only the n-sequence $(1, 1, ..., 1)$ is concerned). For this reason, it is understood that all k-preassignable arbitrarily partitionable graphs considered throughout this chapter have order at least $k + 1$.

We start by raising the following counterpart of Observation 2.27 for k-preassignable arbitrarily partitionable graphs, which follows from the same arguments.

Observation 4.1. *Let $k \geq 1$. Every graph spanned by a k-preassignable arbitrarily partitionable graph is k-preassignable arbitrarily partitionable.*

Adding more and more edges to a k-preassignable arbitrarily partitionable graph, we then get more and more k-preassignable arbitrarily partitionable graphs. Besides, since every subgraph of a complete graph is traceable, every complete graph on at least k vertices is k-preassignable arbitrarily partitionable. This should convince the reader that k-preassignable arbitrarily partitionable graphs with order n exist for all values of k, where $n \geq k$.

We are here interested in finding k-preassignable arbitrarily partitionable graphs with small size. Our guesses for candidates in this chapter are mainly influenced by the next observation following from the fact that requesting a vertex to belong to a subgraph with order 1 is like removing it from the graph.

Observation 4.2. *Let $k \geq 1$. Every k-preassignable arbitrarily partitionable graph is $(k + 1)$-connected.*

Proof. Assume a graph G is not $(k + 1)$-connected, and let $v_1, v_2, ..., v_k$ be distinct vertices of G such that $G - \{v_1, v_2, ..., v_k\}$ is not connected. Since $G - \{v_1, v_2, ..., v_k\}$ is not connected, the trivial $(|V(G)| - k)$-sequence $(|V(G)| - k)$ cannot be realized in $G - \{v_1, v_2, ..., v_k\}$, see Figure 4.1.a. It then follows that the $|V(G)|$-sequence $(1, 1, ..., 1, |V(G)| - k)$, where the value 1 appears k times at the beginning of the sequence, is not $(v_1, v_2, ..., v_k)$-realizable in G. ■

Using Observation 4.2, we can notably prove the following.

Observation 4.3. *Let $k \geq 1$. Every k-preassignable arbitrarily partitionable graph is also k'-preassignable arbitrarily partitionable for every $k' \in \{1, 2, ..., k - 1\}$.*

Proof. Assume G is a k-preassignable arbitrarily partitionable graph, and let π be a $|V(G)|$-sequence and P' be a k'-preassignation of G. We deduce a P'-realization of π in G. In case $|\pi| \geq k$, let P be a k-preassignation of G including P', i.e. $P' \subset P$. Since G is k-preassignable

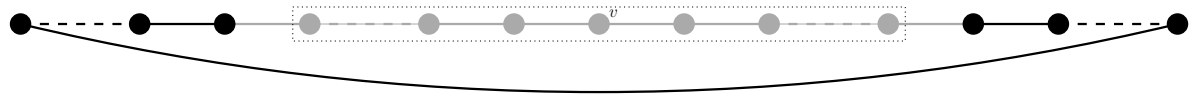


Figure 4.2: A connected part (in grey only) with arbitrary size including an arbitrary vertex v in an Hamiltonian graph (in black and grey) such that the remaining graph (in black only) is traceable.

arbitrarily partitionable, we can deduce a P-realization of π in G. This realization also forms a P'-realization of π in G. Now assume that $|\pi| < k$. Since G is k-preassignable arbitrarily partitionable, by Observation 4.2 we know that G is $(k+1)$-connected. The P'-realization of π in G can hence be deduced using Theorem 2.1. ■

Clearly the necessary condition from Observation 4.2 is not a sufficient one, i.e. $(k+1)$-connected graphs are not necessarily k-preassignable arbitrarily partitionable. We illustrate this statement via the following result, which will be of some use in further sections.

Lemma 4.4. *Let $k \geq 1$ be an integer, and $G = (A \cup B, E)$ be a bipartite graph with order at least $k+2$. If k and $|V(G)|$ have the same parity, then G cannot be k-preassignable arbitrarily partitionable.*

Proof. Assume k and $|V(G)|$ have the same parity, and choose two subsets $X \subset A$ and $Y \subset B$ such that $|X| + |Y| = k$, and $G - (X \cup Y)$ is as unbalanced as possible. These subsets can be obtained e.g. by deriving one of the following two strategies. On the one hand, if k is greater than, say, $|A|$, then let $X = A$ and Y be a subset of B with size $k - |A|$. On the other hand, if k is smaller than both $|A|$ and $|B|$, then, assuming $|A| \leq |B|$, choose X and Y so that $|X| = k$ and $Y = \emptyset$.

Let $A' = A \setminus X$ and $B' = B \setminus Y$. Since $|A' + B'|$ is even and $|A'| \neq |B'|$, the graph $G[A' \cup B']$ cannot admit a perfect matching, see Figure 4.1.b. It then follows that the $|V(G)|$-sequence $(1, 1, ..., 1, 2, 2, ..., 2)$, where the value 1 appears k times at the beginning of the sequence, is not $(v_1, v_2, ..., v_k)$-realizable in G, where $\{v_1, v_2, ..., v_k\} = X \cup Y$. ■

Regarding the algorithmic point of view, the problem of deciding whether a sequence is P-realizable in a graph for some k-preassignation P is NP-complete no matter what is the value of k, recall Theorem 3.5. Since Realizable Sequence and Realizable Size-k Sequence with k'-Preassignation have the same complexity status, i.e. preassigning vertices does not alter the membership of Realizable Sequence to NP, we directly get that the problem

k-Preassignable Arbitrarily Partitionable Graph
Instance: A graph G.
Question: Is G k-preassignable arbitrarily partitionable?

is in Π_2^p, for the same reasons as Arbitrarily Partitionable Graph belongs to Π_2^p. But again the Π_2^p-completeness of k-Preassignable Arbitrarily Partitionable Graph does not seem easy to us to establish, recall our explanations from Section 3.2.

4.2 Powers of graphs with Hamiltonian properties

Every traceable or Hamiltonian graph is arbitrarily partitionable according to Observation 2.34. Actually it is easily seen that every traceable graph is arbitrarily (u)-partitionable, and even arbitrarily (u, v)-partitionable, where u and v denote the endvertices of one of its Hamiltonian paths. Besides, every Hamiltonian graph G is 1-preassignable arbitrarily partitionable since

one can pick a connected part with arbitrary size and including one specific vertex along the Hamiltonian cycle of G in such a way that what remains is traceable (and hence arbitrarily partitionable), see Figure 4.2.

Observation 4.5. *Every Hamiltonian graph is* 1*-preassignable arbitrarily partitionable.*

We generalize Observations 2.34 and 4.5 to powers of traceable or Hamiltonian graphs. More precisely, we prove that kth powers of traceable or Hamiltonian graphs are $(k-1)$- or $(2k-1)$-preassignable arbitrarily partitionable, respectively. These results are tight in the sense that we cannot preassign more vertices while partitioning these graphs in general because of their connectivity, recall Observation 4.2.

We throughout make use of the following notation.

Notation 4.6. If G is a graph with a natural ordering of its vertices (typically graphs spanned by a path or a cycle), then, for every vertex v of G, we denote by v^+ (resp. v^-) the neighbour of v succeeding (resp. preceding) v in this ordering. Besides, assuming u and v are two vertices of G such that u precedes v in the ordering, by uGv we refer to the graph $G[\{u, u^+, (u^+)^+, ..., v^-, v\}]$.

4.2.1 Powers of traceable graphs

We prove below that kth powers of traceable graphs are $(k-1)$-preassignable arbitrarily partitionable. We actually prove a stronger statement, namely that these graphs can even be partitioned following k-preassignations involving one of their endvertices. The proof makes use of the following two observations.

Observation 4.7. *Removing the first or last vertex of a* k*th power of path results in a* k*th power of path.*

Observation 4.8. *Let* $G = P_n^k$ *with* $k \geq 1$ *and* $n \geq k$*, and set* $G' = G - S$ *where* $S \subset V(G)$ *is a subset of vertices of* G *such that no two vertices of* S *are at distance strictly less than* k *in the path* P_n *underlying* G*. Then* G' *is spanned by a* $(k-1)$*th power of* $P_{n-|S|}$ *whose last vertex is also the last vertex of* G *(unless this vertex belongs to* S*).*

Proof. Denote $v_1, v_2, ..., v_n$ the consecutive vertices of G. The result follows from the fact that for every $v_i \notin S$, all but at most one neighbour v_j of v_i with $j < i$ (or $i > j$) belong to G'. ■

Lemma 4.9. *Let* $G = P_n^k$ *with* $k \geq 1$ *and* $n \geq k+1$*, and consider an* n*-sequence* $\pi = (n_1, n_2, ..., n_p)$ *and a* k*-preassignation* P *of* G*. If the first or last vertex of* G *belongs to* P*, then* π *is* P*-realizable in* G*.*

Proof. Let $v_1, v_2, ..., v_n$ denote the consecutive vertices of G, and set $P = (v_{i_1}, v_{i_2}, ..., v_{i_k})$ with $i_1 < i_2 < ... < i_k$. We may assume that $i_k = n$ since otherwise we can just relabel the vertices of G so that this assumption holds. The claim is proved by induction on k. For $k = 1$, the claim is true as pointed out in the forewords of Section 4.2. Assume then that $k \geq 2$ and the claim holds for every $k' < k$.

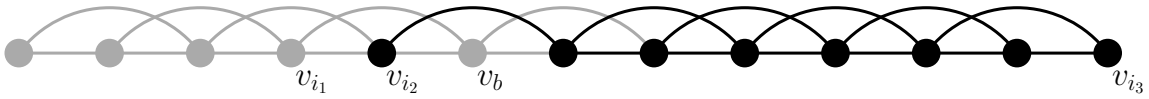
Denote by $r_1, r_2, ..., r_{k-1}$ the residues modulo k of $i_1, i_2, ..., i_{k-1}$, respectively, and let r be an element of the non-empty set $\{0, 1, ..., k-1\} \setminus \{r_1, r_2, ..., r_{k-1}\}$. We construct a sequence $\sigma = (v_{j_1}, v_{j_2}, ..., v_{j_q})$ of $q < n$ distinct vertices of G meeting the following properties:

Rule 1. $v_{j_1} = v_{i_1}$, and $v_{i_2}, v_{i_3}, ..., v_{i_k}$ do not belong to σ,

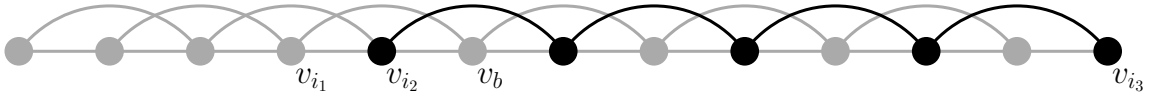
Rule 2. for every $\ell \in \{1, 2, ..., q\}$, the subgraph $G[\{v_{j_1}, v_{j_2}, ..., v_{j_\ell}\}]$ is connected,

Rule 3. for every $\ell \in \{1, 2, ..., q\}$, the subgraph $G - \{v_{j_1}, v_{j_2}, ..., v_{j_\ell}\}$ is spanned by the $(k-1)$th power of a path with last vertex v_n,

Rule 4. every vertex of G either belongs to σ or neighbours a vertex in σ.



(a) Case $n_1 \leq q$. The graph $G^2 - V_1$ is spanned by a path with last vertex v_{i_3} (in black only).



(b) Case $n_1 > q$. The graph $G^2 - V_1$ is spanned by a path (in black only), and every vertex of $V(G) \setminus V_1$ has a neighbour in V_1.

Figure 4.3: Situations described in the proof of Lemma 4.9. A square of a path G (in black and grey) is partitioned following a 3-preassignation $(v_{i_1}, v_{i_2}, v_{i_3})$, where v_{i_3} is the last vertex of G. The part V_1 (in grey only) with size n_1 including v_{i_1} is deduced so that $G^2 - V_1$ has certain properties.

This sequence σ is obtained as follows. First we choose every kth vertex of the sequence $(v_1, v_2, ..., v_{i_1})$ starting from v_{i_1}, i.e. we set $v_{j_1} = v_{i_1}$, $v_{j_2} = v_{i_1-k}, v_{j_3} = v_{i_1-2k}, ..., v_{j_a} = v_{i_1-(a-1)k}$ where $i_1-(a-1)k \in \{1, 2, ..., k\}$. Note that so far Rule 2 and, according to Observation 4.8, Rule 3 are fulfilled. We then add to σ all the yet not chosen consecutive vertices from the sequence $(v_1, v_2, ..., v_{i_1})$ starting from the one with the lowest index, which is either v_1 or v_2. Note that Rule 3 (and obviously Rule 2) is still met according to Observation 4.7, and that the vertices not in σ induce a kth power of a path in G. We end up the construction of σ by adding to σ the vertex v_b not in σ such that b is the smallest index with residue r modulo k which is greater than i_1 and smallest than n (if such a b exists), and then choosing every kth element of the sequence $(v_b, v_{b+1}, ..., v_{n-1})$ starting from v_b, i.e. we set $v_{j_{i_1}+1} = v_b$, $v_{j_{i_1}+2} = v_{b+k}$, $v_{j_{i_1}+3} = v_{b+2k}$, ..., $v_q = v_{b+ck}$ where $b+ck \in \{n-k, n-k+1, ..., n-1\}$. By Observation 4.8, Rule 3 (and obviously Rule 2) is still met. Moreover, Rule 1 is also fulfilled by our choice of r. Finally, because σ contains every kth vertex of the sequence $(v_{i_1}, v_{i_1+1}, ..., v_{n-1})$ including v_{i_1}, Rule 4 is also met.

We now obtain a P-realization of π in G as follows. On the one hand, if $n_1 \leq q$, then set $V_1 = \{v_{j_1}, v_{j_2}, ..., v_{j_{n_1}}\}$. By the properties of σ, the subgraph $G[V_1]$ contains $v_{j_1} = v_{i_1}$ (Rule 1) and is connected (Rule 2). Now let $G' = G - V_1$, as well as $\pi' = \pi \setminus (n_1)$ and $P' = P \setminus (v_{i_1})$. By Rule 1, the vertices of P' belong to G', and G' is spanned by the $(k-1)$th power of a path whose last vertex is v_n according to Rule 3. By induction, we may thus find a P'-realization $(V_2, V_3, ..., V_p)$ of π' in G'. We eventually get that $(V_1, V_2, ..., V_p)$ is a P-realization of π in G, see Figure 4.3.a.

On the other hand, i.e. $n_1 > q$, set $V_1' = \{v_{j_1}, v_{j_2}, ..., v_{j_q}\}$, and let $G' = G - V_1'$, as well as $\pi' = (n_2, n_3, ..., n_p, n_1 - q)$ and $P' = P \setminus (v_{i_1})$. Again by the induction hypothesis, we can find a P'-realization $(V_2, V_3, ..., V_p, V_1'')$ of π' in G'. By Rules 2 and 4, the set $V_1 = V_1' \cup V_1''$ induces a connected subgraph of G with order n_1 and including v_{i_1}. It then follows that $(V_1, V_2, ..., V_p)$ is a P-realization of π in G, see Figure 4.3.b. ■

Using Lemma 4.9, we deduce the following result.

Theorem 4.10. *For every $k \geq 1$ and $n \geq k$, the graph P_n^k is $(k-1)$-preassignable arbitrarily partitionable.*

Proof. Denote $v_1, v_2, ..., v_n$ the consecutive vertices of P_n^k, and consider an n-sequence π and a $(k-1)$-preassignation P of P_n^k. If π has size at most k, then a P-realization of π in P_n^k exists according to Theorem 2.1. Otherwise, i.e. π has size at least $k+1$, then let P' be a k-preassignation of P_n^k including v_1 and all vertices in P (basically, either v_1 does not belong to P and we just add v_1 to P to get P', or P' is obtained by just adding an arbitrary non-preassigned vertex to P if v_1 already belongs to P). Using Lemma 4.9, we can deduce a P'-realization of π in P_n^k. This also performs a P-realization of π in P_n^k. ■

We finally obtain the following result as a corollary of Observation 4.1.

Corollary 4.11. *For every $k \geq 1$, the kth power of every traceable graph is $(k-1)$-preassignable arbitrarily partitionable.*

4.2.2 Powers of Hamiltonian graphs

We now prove the counterpart of Corollary 4.11 for kth powers of Hamiltonian graphs.

Theorem 4.12. *For every $k \geq 1$ and $n \geq 2k$, the graph C_n^k is $(2k-1)$-preassignable arbitrarily partitionable.*

Proof. Let $G = C_n^k$ with $k \geq 1$ and $n \geq 2k$, and denote $v_0, v_1, ..., v_{n-1}$ the consecutive vertices of G. If $k = 1$, then G is 1-preassignable arbitrarily partitionable according to Observation 4.5. Let us thus assume that $k \geq 2$, and consider an n-sequence $\pi = (n_0, n_1, ..., n_{p-1})$ and a $(2k-1)$-preassignation $P = (v_{i_0}, v_{i_1}, ..., v_{i_{2k-2}})$ of G, where $i_0 < i_1 < ... < i_{2k-2}$. We show that there necessarily exists a P-realization of π in G. If π has size at most $2k$, then we know that a P-realization of π in G exists according to Theorem 2.1 since G is $2k$-connected.

Assume then that $p > 2k$. For each vertex v_{i_j} of G, where $j \in \{0, 1, ..., 2k-2\}$, let $D_j = V(v_{i_{j-1}}^+ G v_{i_j})$ denote the set of vertices lying between $v_{i_{j-1}}$ and v_{i_j} including v_{i_j}, and $d_j = |D_j|$ be the number of these vertices, where the indices are counted modulo $2k-1$ (here and further). For every $j \in \{0, 1, ..., 2k-2\}$, let further

$$s_j = d_{j+1} + d_{j+2} + ... + d_{j+k-1} \quad \text{and} \quad q_j = n_{j+1} + n_{j+2} + ... + n_{j+k-1}.$$

In particular, we have $d_0 + d_1 + ... + d_{2k-2} = n$, and $n_0 + n_1 + ... + n_{2k-2} < n$ since $p > 2k$. Therefore, there must exist a $j \in \{0, 1, ..., 2k-2\}$ for which $q_j < s_j$, since otherwise we would obtain the following contradiction:

$$(k-1)n > (k-1)\sum_{j=0}^{2k-2} n_j = \sum_{j=0}^{2k-2} q_j \geq \sum_{j=0}^{2k-2} s_j = (k-1)\sum_{j=0}^{2k-2} d_j = (k-1)n.$$

Case 1. *$q_{j'} \geq s_{j'}$ for some $j' \neq j \in \{0, 1, ..., 2k-2\}$.*

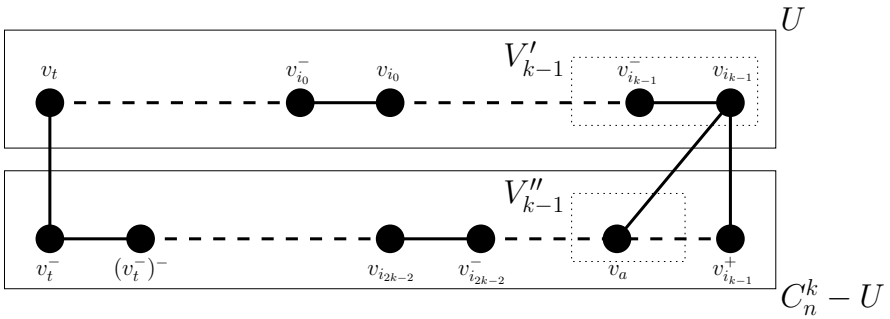
Without loss of generality we may assume that $j' = 0$ and $j = 2k-2$ (we hence have $q_{2k-2} < s_{2k-2}$ and $q_0 \geq s_0$), and $v_{i_{2k-2}}^+ = v_0$, implying both

$$\begin{aligned} n_0 + n_1 + \ldots + n_{k-2} \leq \;& d_0 + d_1 + \ldots + d_{k-2} - 1 \;\; = |\{v_0, v_1, \ldots, v_{i_{k-2}}^-\}| \text{ and} \\ n_0 + n_1 + \ldots + n_{k-1} \geq \;& 1 + d_1 + d_2 + \ldots + d_{k-1} \;\; = |\{v_{i_0}, v_{i_0}^+, \ldots, v_{i_{k-1}}\}|. \end{aligned}$$

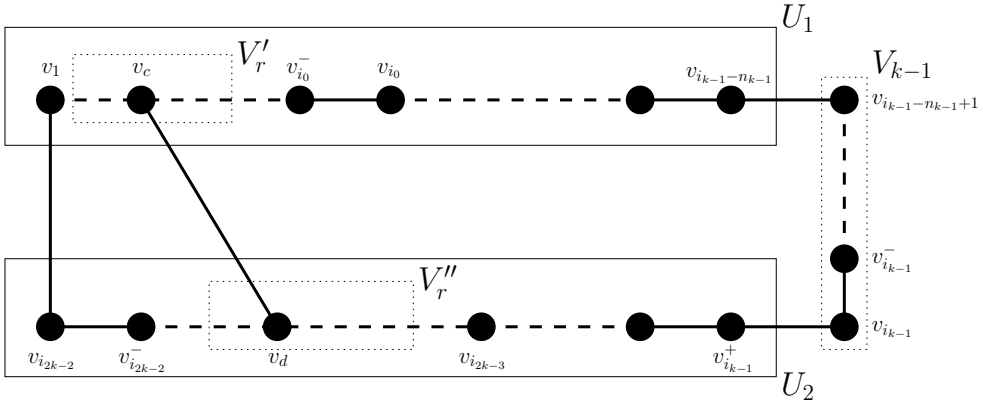
Then there exists a $t \in \{0, 1, ..., i_0\}$ such that for $U = \{v_t, v_{t+1}, \ldots, v_{i_{k-1}}\}$ we have

$$\begin{aligned} n_0 + n_1 + \ldots + n_{k-2} \;&\leq\; |U| - 1 \text{ and} \\ n_0 + n_1 + \ldots + n_{k-1} \;&\geq\; |U|. \end{aligned} \tag{4.1}$$

Note that $U \cap P = \{v_{i_0}, v_{i_1}, \ldots, v_{i_{k-1}}\}$. Thus if on the one hand $|U| = n_0 + n_1 + \ldots + n_{k-1}$, then, using Theorems 2.1 and 4.10, we can find a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization $(V_0, V_1, \ldots, V_{k-1})$ of $(n_0, n_1, \ldots, n_{k-1})$ in $G[U]$, which is k-connected, and a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-2}})$-realization $(V_k, V_{k+1}, \ldots, V_{p-1})$ of $(n_k, n_{k+1}, \ldots, n_{p-1})$ in $G - U$, which is the kth power of a path. If on the other hand we have $n_0 + n_1 + \ldots + n_{k-1} > |U|$, then, according to Inequality 4.1, we can deduce two positive integers n'_{k-1} and n''_{k-1} such that $n_0 + n_1 + \ldots + n_{k-2} + n'_{k-1} = |U|$ and $n'_{k-1} + n''_{k-1} = n_{k-1}$. Let G' and G'' be the kth powers of paths induced by U and $V(G) \setminus U$, respectively, in G, and let $v_a \notin P$ be the first non-preassigned vertex after $v_{i_{k-1}}$ following the orientation of G. In particular, observe that, since $|P \setminus U| = k - 1$, this vertex v_a is a neighbour of $v_{i_{k-1}}$ in G. By



(a) Case where $q_{2k-2} < s_{2k-2}$ and $q_0 \geq s_0$. The graph C_n^k is partitioned into two parts U and $C_n^k - U$ including k and $k-1$ preassigned vertices, respectively. The realization is obtained by considering independent realizations in these two subgraphs, and then unifying some of the resulting parts.



(b) Case where $q_j < s_j$ for every $j \in \{0, 1, ..., 2k-2\}$. We pick the connected part V_{k-1} and partition the remaining graph into two parts U_1 and U_2 each including $k-1$ preassigned vertices. The realization is obtained by filling U_1 and U_2 with as many parts as possible, where an "exceeding" part V_r is partially picked in U_1 and U_2.

Figure 4.4: Situations described in the proof of Theorem 4.12. The graph C_n^k is partitioned following a $(2k-1)$-preassignation $(v_{i_0}, v_{i_1}, ..., v_{i_{2k-2}})$.

Theorem 2.1 and Lemma 4.9, there exist a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization $(V_0, V_1, \ldots, V_{k-2}, V'_{k-1})$ and a $(v_a, v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-2}})$-realization $(V''_{k-1}, V_k, V_{k+1}, \ldots, V_{p-1})$ of $(n_0, n_1, \ldots, n_{k-2}, n'_{k-1})$ and $(n''_{k-1}, n_k, n_{k+1}, \ldots, n_{p-1})$, respectively, in G' and G'', respectively. It then follows that

$$(V_0, V_1, \ldots, V_{k-2}, V'_{k-1} \cup V''_{k-1}, V_k, V_{k+1}, \ldots, V_{p-1})$$

is a P-realization of π in G, see Figure 4.4.a.

Case 2. *$q_j < s_j$ for every $j \in \{0, 1, ..., 2k-2\}$.*

In this situation, note that there is no k consecutive preassigned vertices of G in P. Note also that since

$$n_1 + n_2 + \ldots + n_{k-1} = q_0 < s_0 = d_1 + d_2 + \ldots + d_{k-1},$$

there exists an $i' \in \{1, 2, \ldots, k-1\}$ for which $n_{i'} < d_{i'}$. Without loss of generality we may assume that $i' = k-1$ and $v_{i_{2k-2}} = v_0$. Set $V_{k-1} = \{v_{i_{k-1}-n_{k-1}+1}, v_{i_{k-1}-n_{k-1}+2}, \ldots, v_{i_{k-1}}\}$. Clearly, we have both $|V_{k-1}| = n_{k-1}$ and $V_{k-1} \cap P = \{v_{i_{k-1}}\}$. Moreover, the sets $U_1 = \{v_1, v_2, \ldots, v_{i_{k-1}-n_{k-1}}\}$ and $U_2 = V(G) \setminus (U_1 \cup V_{k-1})$ induce kth powers of paths G' and G'', respectively, in G, such that $P \cap U_1 = \{v_{i_0}, v_{i_1} \ldots, v_{i_{k-2}}\}$ and $P \cap U_2 = \{v_{i_k}, v_{i_{k+1}} \ldots, v_{i_{2k-2}}\}$. Furthermore, we have

$$n_0 + n_1 + \ldots + n_{k-2} = q_{2k-2} < s_{2k-2} = d_0 + d_1 + ... + d_{k-2} < |U_1|,$$

and

$$n_k + n_{k+1} + ... + n_{2k-2} = q_{k-1} < s_{k-1} = d_k + d_{k+1} + ... + d_{2k-2} = |U_2|.$$

If we are then able to partition the remaining elements $n_{2k-1}, n_{2k}, \ldots, n_{p-1}$ of π into two parts π_1 and π_2 such that $\sum_{i=0}^{k-2} n_i + \|\pi_1\| = |U_1|$ and $\sum_{i=k}^{2k-2} n_i + \|\pi_2\| = |U_2|$, then the result follows again by using Theorem 4.10 twice. Otherwise, there is a value $n_r \in \{n_{2k-1}, n_{2k}, ..., n_{p-1}\}$ which can be split into two positive elements n'_r and n''_r in such a way that $n_r = n'_r + n''_r$, as well as a partition of $\{n_{2k-1}, n_{2k}, ..., n_{p-1}\} \setminus \{n_r\}$ into two parts $\pi_1 = (n_{2k-1}, n_{2k}, ..., n_{r-1})$ and $\pi_2 = (n_{r+1}, n_{r+2}, ..., n_{p-1})$ such that $\sum_{i=0}^{k-2} n_i + \|\pi_1\| + n'_r = |U_1|$. Let v_c be the first vertex of U_1 which does not belong to P, and let v_d be the last vertex of U_2 which does not belong to P. Since P does not contain k consecutive vertices of G, the vertices v_c and v_d are adjacent. Using Lemma 4.9, we can then deduce a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-2}}, v_c)$-realization $(V_0, V_1, ..., V_{k-2}, V'_r, V_{2k-1}, V_{2k}, ..., V_{r-1})$ of $(n_0, n_1, ..., n_{k-2}, n'_r, n_{2k-1}, n_{2k}, ..., n_{r-1})$ in $G[U_1]$, and a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-2}}, v_d)$-realization $(V_k, V_{k+1}, ..., V_{2k-2}, V''_r, V_{r+1}, V_{r+2}, ..., V_{p-1})$ of $(n_k, n_{k+1}, ..., n_{2k-2}, n''_r, n_{r+1}, n_{r+2}, ..., n_{p-1})$ in $G[U_2]$ since $G[U_1]$ and $G[U_2]$ are kth powers of paths. It eventually follows that

$$(V_0, V_1, \ldots, V_{r-1}, V'_r \cup V''_r, V_{r+1}, V_{r+2}, \ldots, V_{p-1})$$

is a P-realization of π in G, see Figure 4.4.b. ■

Applying the spanning argument from Observation 4.1, we directly get the following.

Corollary 4.13. *For every $k \geq 1$, the kth power of every Hamiltonian graph is $(2k-1)$-preassignable arbitrarily partitionable.*

4.3 Minimum size

As mentioned earlier, starting from powers of traceable or Hamiltonian graphs, we can construct infinitely many k-preassignable arbitrarily partitionable graphs by repeatedly adding edges in these graphs, recall Observation 4.1, getting consecutive graphs converging towards complete graphs. We herein consider the opposite extremal direction, namely what is the minimum size of a k-preassignable arbitrarily partitionable graph on n vertices? This question is considered regarding both k and n.

Since every k-preassignable arbitrarily partitionable graph is $(k+1)$-connected, recall Observation 4.2, and every $(k+1)$-connected graph has minimum degree $k+1$, we directly get the following lower bound.

Observation 4.14. *Every k-preassignable arbitrarily partitionable graph with order $n \geq k+1$ has size at least $\lceil \frac{n(k+1)}{2} \rceil$.*

Note that this lower bound is not theoretical only as it is reached e.g. for odd values of k as kth powers of cycles, which are minimum in terms of size with regards to their connectivity, are $(2k-1)$-preassignable arbitrarily partitionable, recall Theorem 4.12.

Proposition 4.15. *For every $k \geq 1$ odd and $n \geq k+2$, there are $(k+1)$-preassignable arbitrarily partitionable graphs with order n and size $\frac{n(k+1)}{2}$.*

The aim of this section is to show that the lower bound given by Observation 4.14 is also reached for even values of $k \geq 2$ (and every $n \geq k+1$). For this purpose, we first show in Section 4.3.1 that k-connected Harary graphs with $k \geq 5$ odd are $(k-1)$-preassignable arbitrarily partitionable. Since Harary graphs are known to be graphs with minimum size (regarding their connectivity), we directly get that the lower bound from Observation 4.14 is also reached for every $k \geq 4$ even. Unfortunately, 3-connected Harary graphs are not all 2-preassignable arbitrarily partitionable. We then introduce another family of graphs in Section 4.3.2, and show its members to be 2-preassignable arbitrarily partitionable graphs with minimum size.

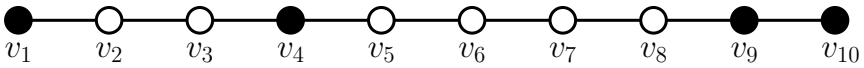


Figure 4.5: A graph with a natural ordering $(v_1, v_2, ..., v_{10})$ of its vertices, some of which are preassigned (in white).

4.3.1 Harary graphs with odd connectivity at least 5

Every Harary graph $H_{2k+1,n}$ is spanned by C_n^k and is thus a $(2k-1)$-preassignable arbitrarily partitionable graph according to Theorem 4.12. We show below, in Section 4.3.1.2, that an additional vertex preassignation can always be requested for partitioning these graphs. We beforehand describe, in Section 4.3.1.1, some particular situations in which the kth power of a path or cycle can be partitioned into connected subgraphs when strictly more than $k-1$ or $2k-1$, respectively, preassigned vertices are specified. These results are of great interest since Harary graphs admit a lot of powers of paths as induced subgraphs, and are spanned by a power of a cycle, as mentioned above.

The vertices constituting a k-preassignation P of a graph G form a crucial parameter when P-realizing a $|V(G)|$-sequence in G. Our proofs below notably depend on whether P form a *preassigned block*.

Definition 4.16. Assuming P is a preassignation of a graph G which admits a natural ordering of its vertices (like e.g. paths, cycles, or every graph spanned by one such graph), a *preassigned block* B of P is a set $\{v_{i_{j_1}}, v_{i_{j_2}}, ..., v_{i_{j_\ell}}\}$ of consecutive preassigned vertices (following the ordering of G). We say that B is *maximal* if neither the vertex preceding $v_{i_{j_1}}$ nor the vertex succeeding $v_{i_{j_\ell}}$ are preassigned vertices.

Example 4.17. In the graph depicted in Figure 4.5, the preassigned vertices form two maximal preassigned blocks, namely $\{v_2, v_3\}$ and $\{v_5, v_6, v_7, v_8\}$. The preassigned block $\{v_6, v_7\}$ is not maximal since e.g. v_8 is preassigned and succeeds v_7.

4.3.1.1 Partitioning powers of paths or cycles following large preassignations

Recall that kth powers of paths are partitionable under k-preassignations involving one endvertex, see Lemma 4.9. In the next two results, we exhibit additional situations under which kth powers of paths can be partitioned following k-preassignations, or even $(k+1)$-preassignations.

Lemma 4.18. *Let $G = P_n^k$ with $k \geq 2$ and $n \geq k+1$, and consider an n-sequence $\pi = (n_1, n_2, ..., n_p)$ and a k-preassignation P of G. If the vertices of P do not form a maximal preassigned block with size k, then π is P-realizable in G.*

Proof. We denote $v_1, v_2, ..., v_n$ the consecutive vertices of the path P_n spanning G, and set $P = (v_{i_1}, v_{i_2}, ..., v_{i_k})$, where $i_1 < i_2 < ... < i_k$. If $s = \sum_{j=k+1}^{p} n_j \leq i_1 - 1$, then a P-realization of π in G is $(V_1, V_2, ..., V_p)$, where $(V_{k+1}, V_{k+2}, ..., V_p)$ is a realization of $(n_{k+1}, n_{k+2}, ..., n_p)$ in the traceable graph $G[\{v_1, v_2, ..., v_s\}]$, and $(V_1, V_2, ..., V_k)$ is a P-realization of $(n_1, n_2, ..., n_k)$ in $G - \{v_1, v_2, ..., v_s\}$ obtained using Theorem 2.1.

Suppose now that $s \geq i_1$. On the one hand, if $n_1 \geq i_1$, then a correct P-realization of π in G is $(V_1' \cup V_1'', V_2, V_3, ..., V_p)$, where $V_1' = \{v_1, v_2, ..., v_{i_1-1}\}$ and $(V_1'', V_2, V_3, ..., V_p)$ is a P-realization of $(n_1 - i_1 + 1, n_2, n_3, ..., n_p)$ in $G - V_1'$ obtained via Lemma 4.9. On the other hand, if $n_1 < i_1$, then let V_1 be a subset of $\{v_1, v_2, ..., v_{i_1}\}$ obtained as follows. First, set $V_1 = \{v_{i_1}\}$ and then repeatedly add to V_1 the vertex located at distance 2 on the left of the last vertex added to V_1 as long as $|V_1| < n_1$ and v_1 is not reached. If there is no vertex at distance 2 on the left of the last vertex added to V_1 (but V_1 needs additional vertices), then add to V_1 every remaining vertex from $\{v_1, v_2, ..., v_{i_1-1}\} \setminus V_1$ from left to right until V_1 has size n_1. Let $X = \{v_1, v_2, ..., v_{i_1-1}\} \setminus V_1$. Note

that, at the end of the procedure, $G[V_1]$ is connected, $G[X]$ is traceable, and $v_{i_1-1} \in X$. Now, if there exists an $r \in \{k+1, k+2, ..., p\}$ such that $\sum_{j=k+1}^{r} n_j = |X|$, then a P-realization of π in G is $(V_1, V_2, ..., V_p)$ where $(V_{k+1}, V_{k+2}, ..., V_r)$ is a realization of $(n_{k+1}, n_{k+2}, ..., n_r)$ in $G[X]$ and $(V_2, V_3, ..., V_k, V_{r+1}, V_{r+2}, ..., V_p)$ is a $(v_{i_2}, v_{i_3}..., v_{i_k})$-realization of $(n_2, n_3, ..., n_k, n_{r+1}, n_{r+2}, ..., n_p)$ in $G - \{v_1, v_2, ..., v_{i_1}\}$ obtained using Theorem 4.10.

If such a value of r does not exist, then let r be such that $\sum_{j=k+1}^{r-1} n_j < |X|$ and $\sum_{j=k+1}^{r} n_j > |X|$. Let further $n'_r = |X| - \sum_{j=k+1}^{r-1} n_j$, $n''_r = n_r - n'_r$, and $v_a \notin P$ be the nearest neighbour of v_{i_1-1} located on the right of v_{i_1}. Such a vertex necessarily exists since otherwise the preassigned vertices would form a maximal prescribed block with size k in G. Moreover, either v_a or v_{i_2} is the first vertex of $G - \{v_1, v_2, ..., v_{i_1}\}$. We then obtain a P-realization $(V_1, V_2, ..., V_{r-1}, V'_r \cup V''_r, V_{r+1}, V_{r+2}, ..., V_p)$ of π in G by considering a (v_{i_1-1})-realization $(V'_r, V_{k+1}, V_{k+2}, ..., V_{r-1})$ of $(n'_r, n_{k+1}, n_{k+2}, ..., n_{r-1})$ in $G[X]$ and a $(v_{i_2}, v_{i_3}, ..., v_{i_k}, v_a)$-realization $(V_2, V_3, ..., V_k, V''_r, V_{r+1}, V_{r+2}, ..., V_p)$ of $(n_2, n_3, ..., n_k, n''_r, n_{r+1}, n_{r+2}, ..., n_p)$ in $G[\{v_{i_1+1}, v_{i_1+2}, ..., v_n\}]$. These two realizations exist according to Lemma 4.9. ■

We now strengthen Lemma 4.9 by showing that kth powers of paths are partitionable following $(k+1)$-preassignations involving their two endvertices.

Lemma 4.19. *Let $G = P_n^k$ with $k \geq 1$ and $n \geq k+2$, and consider an n-sequence $\pi = (n_1, n_2, ..., n_p)$ and a $(k+1)$-preassignation P of G. If the two endvertices of G belong to P, then π is P-realizable in G.*

Proof. We prove this claim by induction on k. For $k = 1$, the result is obvious. We thus now suppose that $k \geq 2$ and that the claim holds for every $k' < k$. The vertices of G are denoted similarly as in the proof of Lemma 4.18, and we put $P = (v_{i_1}, v_{i_2}, ..., v_{i_{k+1}})$, where $1 = i_1 < i_2 < ... < i_{k+1} = n$. If $n_1 \leq i_2 - 1$, then a correct P-realization of π in G is $(V_1, V_2, ..., V_p)$ where $V_1 = \{v_1, v_2, ..., v_{n_1}\}$ and $(V_2, V_3, ..., V_p)$ is a $(v_{i_2}, v_{i_3}, ..., v_{i_{k+1}})$-realization of $(n_2, n_3, ..., n_p)$ in $G - V_1$. This realization necessarily exists according to Lemma 4.9 since $v_{i_{k+1}}$ is the last vertex of $G - V_1$.

Suppose now that $n_1 \geq i_2$. Observe that $\{0, 1, ..., k-1\} \setminus \bigcup_{j=2}^{k} \{i_j \pmod k\}$ is not empty, so let us denote by r one value of this set. The subset V_1 of the realization is constructed as follows. It first contains all the vertices between v_1 and v_{i_2-1}, i.e. $\{v_1, v_2, ..., v_{i_2-1}\} \subseteq V_1$. We then add the vertex v_a to V_1, where $a \in \{i_2+1, i_2+2, ..., i_2+k-1\}$ is such that $a \equiv r \pmod k$. Finally, as long as $|V_1| < n_1$ and we do not reach v_n, we repeatedly add to V_1 the vertex at distance k on the right from the last one added to V_1, i.e. v_{a+k}, then v_{a+2k}, and so on. According to our choice of r, these vertices are not preassigned ones and, at every moment of the procedure, the subgraph $G - V_1$ is spanned by the $(k-1)$th power of a path, recall Observation 4.8, and the subgraph $G[V_1]$ is connected.

On the one hand, if $V_1 = n_1$ holds once the procedure is achieved, then $(V_1, V_2, ..., V_p)$ is a P-realization of π, where $(V_2, V_3, ..., V_p)$ is a $(v_{i_2}, v_{i_3}, ..., v_{i_{k+1}})$-realization of $(n_2, n_3, ..., n_p)$ in $G - V_1$ which necessarily exists by the induction hypothesis since v_{i_2} and $v_{i_{k+1}}$ are the endvertices of $G - V_1$.

On the other hand, if $|V_1| < n_1$ holds after the end of the procedure, then each vertex from $V(G) \setminus V_1$ has a neighbour in V_1. Hence, we can obtain a P-realization $(V_1 \cup V'_1, V_2, V_3, ..., V_p)$ of π in G, where $(V_2, V_3, ..., V_p, V'_1)$ is a $(v_{i_2}, v_{i_3}, ..., v_{i_{k+1}})$-realization of $(n_2, n_3, ..., n_p, n_1 - |V_1|)$ in $G - V_1$. Once again, such a realization necessarily exists according to the induction hypothesis. ■

We now prove an analogous result concerning cycles to the power of at least 2, which is crucial for our upcoming results related to Harary graphs.

Lemma 4.20. *Let $G = C_n^k$ with $k \geq 2$ and $n \geq 2k+1$, and consider an n-sequence $\pi = (n_0, n_1, ..., n_{p-1})$ and a $2k$-preassignation P of G. If the vertices of P do not form two maximal preassigned blocks with size k, then π is P-realizable in G.*

Proof. We denote $v_0, v_1, ..., v_{n-1}$ the consecutive vertices of the cycle C_n underlying G, and set $P = (v_{i_0}, v_{i_1}, ..., v_{i_{2k-1}})$ with $i_0 < i_1 < ... < i_{2k-1}$. Quite similarly as in the proof of Theorem 4.12, for every $j \in \{0, 1, ..., 2k-1\}$, let

$$s_j = \sum_{\ell=j}^{j+k-1} d_\ell \quad \text{and} \quad q_j = \sum_{\ell=j}^{j+k-1} n_\ell,$$

where the values $d_0, d_1, ..., d_{2k-1}$ (and the associated sets $D_0, D_1, ..., D_{2k-1}$) are computed similarly as in the proof of Theorem 4.12, and the indices are taken modulo $2k$. In other words, every value s_j is the order of the graph $v^+_{i_{j-1}} G v_{i_{j+k-1}} = G[\{i^+_{j-1}, (i^+_{j-1})^+, ..., i_{j+k-1}\}]$ including the k preassigned vertices v_{i_j}, $v_{i_{j+1}}$, ..., $v_{i_{j+k-1}}$, and q_j is the amount of vertices required in the connected subgraphs containing these preassigned vertices in a P-realization of π in G.

Note that there necessarily exists a $j \in \{0, 1, ..., 2k-1\}$ such that $q_j \leq s_j$ since having $\sum_{j=0}^{2k-1} q_j > \sum_{j=0}^{2k-1} s_j$ implies $k \sum_{\ell=0}^{2k-1} n_\ell > k \sum_{\ell=0}^{2k-1} d_\ell$, which is impossible as $n = \sum_{\ell=0}^{2k-1} d_\ell$ and $n > \sum_{\ell=0}^{2k-1} n_\ell$ (if π contains at most $2k$ elements, then the P-realization can be obtained using Theorem 2.1). To prove the claim, we distinguish several cases depending on the relationship between the q_j's and s_j's.

Case 1. *$q_j = s_j$ for some $j \in \{0, 1, ..., 2k-1\}$.*

In this situation, a P-realization of π in G is deduced as follows. Assume $j = 0$ without loss of generality. On the one hand, since $G[\bigcup_{\ell=0}^{k-1} D_\ell]$ is the kth power of a path, it is k-connected and thus admits a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization $(V_0, V_1, ..., V_{k-1})$ of $(n_0, n_1, ..., n_{k-1})$ according to Theorem 2.1. On the other hand, the graph $G - \bigcup_{\ell=0}^{k-1} D_\ell$ is the kth power of a path whose last vertex is $v_{i_{2k-1}}$. Therefore, there exists a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization $(V_k, V_{k+1}, ..., V_{p-1})$ of $(n_k, n_{k+1}, ..., n_{p-1})$ in this graph by Lemma 4.9. The partition $(V_0, V_1, ..., V_{p-1})$ is then a P-realization of π in G.

Case 2. *We are not in Case 1 and $q_j > s_j$ for some $j \in \{0, 1, ..., 2k-1\}$.*

In particular, there exists a value of j for which $q_j > s_j$ and $q_{j+1} < s_{j+1}$. Suppose $j = 0$ without loss of generality.

Case 2.1. *There exists a set $X = \{v^+_{i_{2k-1}}, (v^+_{i_{2k-1}})^+, ..., v_a\}$ with $a \in \{i_{k-1}+1, i_{k-1}+2, ..., i_k-1\}$ such that $|X| = q_0$.*

A P-realization of π in G can be obtained as follows. Firstly, let $(V_0, V_1, ..., V_{k-1})$ be a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization of $(n_0, n_1, ..., n_{k-1})$ in $G[X]$. Such a realization exists by Theorem 2.1 since $G[X]$ is the kth power of a path. Secondly, let $(V_k, V_{k+1}, ..., V_{p-1})$ be a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization of $(n_k, n_{k+1}, ..., n_{p-1})$ in $G - X$ which necessarily exists according to Lemma 4.9 since $v_{i_{2k-1}}$ is the last vertex of $G - X$. The partition $(V_0, V_1, ..., V_{p-1})$ is then a P-realization of π in G.

Case 2.2. *Such a set X does not exist.*

In such a situation, we have $q_0 > s_0 + d_k - 1$, i.e. $\sum_{\ell=0}^{k-1} n_\ell > (\sum_{\ell=0}^{k} d_\ell) - 1$. Besides, since the n_ℓ's and the d_ℓ's are strictly greater than 0, we get $\sum_{\ell=0}^{k} n_\ell \geq 1 + \sum_{\ell=1}^{k} d_\ell$. Since $q_1 < s_1$, i.e. $\sum_{\ell=1}^{k} n_\ell < \sum_{\ell=1}^{k} d_\ell$, it follows that there exists an n'_0 such that $1 \leq n'_0 \leq n_0$ and $n'_0 + \sum_{\ell=1}^{k} n_\ell = 1 + \sum_{\ell=1}^{k} d_\ell = |\{v_{i_0}, v^+_{i_0}, ..., v_{i_k}\}|$. A P-realization of π in G is then obtained as follows. On the one hand, let $(V'_0, V_1, V_2, ..., V_k)$ be a $(v_{i_0}, v_{i_1}, ..., v_{i_k})$-realization of $(n'_0, n_1, n_2, ..., n_k)$ in $G[\{v_{i_0}, v^+_{i_0}, ..., v_{i_k}\}]$, which exists according to Lemma 4.19 since v_{i_0} and v_{i_k} are the endvertices of $G[\{v_{i_0}, v^+_{i_0}, ..., v_{i_k}\}]$. On the other hand, let $n''_0 = (n_0 - n'_0) + 1$ (clearly $n''_0 \geq 1$), and let $(V''_0, V_{k+1}, V_{k+2}, ..., V_{p-1})$ be a $(v_{i_0}, v_{i_{k+1}}, v_{i_{k+2}}, ..., v_{i_{2k-1}})$-realization of $(n''_0, n_{k+1}, n_{k+2}, ..., n_{p-1})$ in $G[\{v^+_{i_k}, (v^+_{i_k})^+, ..., v_{i_0}\}]$, which exists according to Lemma 4.9 since $G[\{v^+_{i_k}, (v^+_{i_k})^+, ..., v_{i_0}\}]$ is the kth power of a path with last vertex v_{i_0}, and k preassigned vertices are specified. The partition

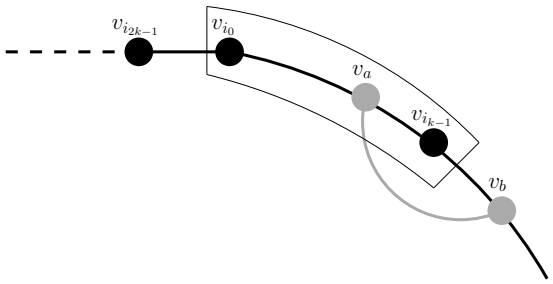


Figure 4.6: Situation described in the proof of Lemma 4.20, Case 3.1.2.1.

$(V_0' \cup V_0'', V_1, V_2, ..., V_{p-1})$ is then a P-realization of π in G since $G[V_0']$ and $G[V_0'']$ are connected and both contain the vertex v_{i_0}.

Case 3. *$q_j < s_j$ for every $j \in \{0, 1, ..., 2k-1\}$.*

We distinguish two subcases.

Case 3.1. *There are two consecutive preassigned vertices.*

Assume $v_{i_0} = v^+_{i_{2k-1}}$ without loss of generality.

Case 3.1.1. *There is an $r \in \{2k, 2k+1, ..., p-1\}$ such that $q_0 + (\sum_{\ell=2k}^{r} n_\ell) = s_0$.*

In this situation, we can deduce a P-realization of π in G as follows. Firstly, let (V_0, V_1, ..., V_{k-1}, V_{2k}, V_{2k+1}, ..., V_r) be a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization of $(n_0, n_1, ..., n_{k-1}, n_{2k}, n_{2k-1}, ..., n_r)$ in $G[\bigcup_{\ell=0}^{k-1} D_\ell]$ which exists according to Lemma 4.9 since v_{i_0} is the first vertex of $G[\bigcup_{\ell=0}^{k-1} D_\ell]$, this graph being the kth power of a path. Secondly, let (V_k, V_{k+1}, ..., V_{2k-1}, V_{r+1}, V_{r+2}, ..., V_{p-1}) be a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization of $(n_k, n_{k+1}, ..., n_{2k-1}, n_{r+1}, n_{r+2}, ..., n_{p-1})$ in $G - \bigcup_{\ell=0}^{k-1} D_\ell$ which exists for the same reason as previously since $v_{i_{2k-1}}$ is the last vertex of $G - \bigcup_{\ell=0}^{k-1} D_\ell$. The partition $(V_0, V_1, ..., V_{p-1})$ is then a P-realization of π in G.

Case 3.1.2. *Such an r does not exist.*

Let $r \in \{2k, 2k+1, ..., p-1\}$ be the value for which we have $q_0 + (\sum_{\ell=2k}^{r-1} n_\ell) < s_0$ and $q_0 + (\sum_{\ell=2k}^{r} n_\ell) > s_0$. Such a value exists since $q_0 < s_0$ and $q_k < s_k$. So let further $n_r' = s_0 - (q_0 + (\sum_{\ell=2k}^{r-1} n_\ell))$ and $n_r'' = n_r - n_r'$. Denote by v_a the last non-preassigned vertex of $G[\bigcup_{\ell=0}^{k-1} D_\ell]$, and by v_b the first non-preassigned vertex of $G - \bigcup_{\ell=0}^{k-1} D_\ell$.

Case 3.1.2.1. *The vertices v_a and v_b are adjacent.*

We obtain a P-realization of π in G as follows (see Figure 4.6). Firstly, let (V_0, V_1, ..., V_{k-1}, V_r', V_{2k}, V_{2k+1}, ..., V_{r-1}) be a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}}, v_a)$-realization of ($n_0$, n_1, ..., n_{k-1}, n_r', n_{2k}, n_{2k+1}, ..., n_{r-1}) in $G[\bigcup_{\ell=0}^{k-1} D_\ell]$, which exists by Lemma 4.19 since $G[\bigcup_{\ell=0}^{k-1} D_\ell]$ is the kth power of a path whose endvertices are v_{i_0} and $v_{i_{k-1}}$. Secondly, let $(V_r'', V_k, V_{k+1}, ..., V_{2k-1}, V_{r+1}, V_{r+2}, ..., V_{p-1})$ be a $(v_b, v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization of $(n_r'', n_k, n_{k+1}, ..., n_{2k-1}, n_{r+1}, n_{r+2}, ..., n_{p-1})$ in $G - \bigcup_{\ell=0}^{k-1} D_\ell$. This realization exists according to Lemma 4.19 since $G - \bigcup_{\ell=0}^{k-1} D_\ell$ is the kth power of a path, either v_b or v_{i_k} is the first vertex of $G - \bigcup_{\ell=0}^{k-1} D_\ell$, and $v_{i_{2k-1}}$ is the last vertex of $G - \bigcup_{\ell=0}^{k-1} D_\ell$. It follows that $(V_0, V_1, ..., V_{r-1}, V_r' \cup V_r'', V_{r+1}, V_{r+2}, ..., V_{p-1})$ is a P-realization of π in G since $G[V_r' \cup V_r'']$ is connected because of the edge $v_a v_b$.

Case 3.1.2.2. *The vertices v_a and v_b are not adjacent.*

In this situation, $v_{i_{k-1}}$ or v_{i_k} (or both of them) belongs to a preassigned block with size at least k. Then one can relabel the preassigned vertices so that v_{i_0} and $v_{i_{2k-1}}$ correspond to two consecutive preassigned vertices from this preassigned block, and use the procedures from

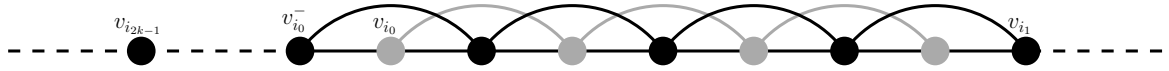


Figure 4.7: Situation described in the proof of Lemma 4.20, Case 3.1.2.2. The part V_0 (in grey only) is picked in such a way that $G[\{v_{i_{2k-1}}, v^+_{i_{2k-1}}, ..., v_{i_1}\} \setminus V_0]$ is spanned by a path with endvertices $v_{i_{2k-1}}$ and v_{i_1} (in black only).

Case 3.1. Since $q_j < s_j$ for every $j \in \{0, 1, ..., 2k-1\}$, note that this time the two vertices v_a and v_b (if these vertices are needed) have to be adjacent since otherwise it would mean that the preassigned vertices form another preassigned block with size at least k (different from the first one), implying that there are two preassigned blocks with size k, contradicting the initial assumption of the lemma.

Case 3.2. *There are no two consecutive preassigned vertices.*

Case 3.2.1. *There exists a set X of consecutive vertices of G such that $X \cap P = \{v_{i_j}, v_{i_{j+1}}, ..., v_{i_{j+k-1}}\}$ and $|X| = q_j$ for some $j \in \{0, 1, ..., 2k-1\}$.*

In this situation, we obtain a P-realization of π in G as follows. Assume $j = 0$ without loss of generality. Firstly, let $(V_0, V_1, ..., V_{k-1})$ be a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization of $(n_0, n_1, ..., n_{k-1})$ in $G[X]$, which exists by Theorem 2.1 since $G[X]$ is the kth power of a path. Secondly, let $(V_k, V_{k+1}, ..., V_{p-1})$ be a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization of $(n_k, n_{k+1}, ..., n_{p-1})$ in $G - X$ obtained using Lemma 4.18 since $G - X$ is the kth power of a path (with $k \geq 2$) and there are no consecutive preassigned vertices. Then $(V_0, V_1, ..., V_{p-1})$ is a P-realization of π in G.

Case 3.2.2. $q_j < s_j - d_j + 1$ *for every* $j \in \{0, 1, ..., 2k-1\}$.

Case 3.2.2.1. *There are two preassigned vertices v_{i_ℓ} and $v_{i_{\ell+1}}$ such that $n_\ell + n_{\ell+1} \geq d_{\ell+1} + 1$.*

Assume $\ell = 0$ without loss of generality. Then there exist two sets of consecutive vertices $X = \{v_{i_0}, v^+_{i_0}, ..., v_a\}$ and $Y = \{v^+_a, (v^+_a)^+, ..., v_{i_1}\}$, with $a \in \{i_0 + 1, i_0 + 2, ..., i_1 - 1\}$, $|X| \leq n_0$ and $|Y| \leq n_1$. A P-realization of π in G can be then obtained as in Case 3.1 by doing as if v_{i_0} and v_{i_1} were consecutive preassigned vertices, but requesting v_{i_0} and v_{i_1} to belong to subgraphs with order $n_0 - |X| + 1$ and $n_1 - |Y| + 1$, respectively. It is worth recalling that we are under the assumption that there are no two consecutive preassigned vertices (so, in particular, the vertices v_a and v_b, if needed, will be adjacent). For the resulting parts V'_0 and V'_1, the graphs $G[V'_0 \cup X]$ and $G[V'_1 \cup Y]$ are connected, and have order n_0 and n_1, respectively.

Case 3.2.2.2. $n_j + n_{j+1} < d_{j+1} + 1$ *for every* $j \in \{0, 1, ..., 2k-1\}$.

In particular, $n_0 + n_1 < d_1 + 1 = |\{v_{i_0}, v^+_{i_0}, ..., v_{i_1}\}|$. We cannot have both $n_0 \geq \lceil \frac{d_1+1}{2} \rceil$ and $n_1 \geq \lceil \frac{d_1+1}{2} \rceil$, since otherwise we would get $n_0 + n_1 \geq d_1 + 1$, a contradiction. Let us thus suppose that $n_0 < \lceil \frac{d_1+1}{2} \rceil$ without loss of generality. Then note that the graph induced by $V_0 = \{v_{i_0}, v_{i_0+2}, v_{i_0+4}..., v_{i_0+2(n_0-1)}\}$ has order n_0 and contains v_{i_0}, and the graph $G[\{v_{i_{2k-1}}, v^+_{i_{2k-1}}, ..., v_{i_1}\} \setminus V_0]$ is traceable with endvertices $v_{i_{2k-1}}$ and v_{i_1}, see Figure 4.7.

Let $t_1 = |\{v_{i_1}, v^+_{i_1}, ..., v_{i_k}\}| - \sum_{\ell=1}^{k} n_\ell$ and $t_2 = |\{v^+_{i_{2k-1}}, (v^+_{i_{2k-1}})^+, ..., v^-_{i_1}\}| - n_0$. From π, we define three sequences π_1, π_2 and π_3. First, let $\pi_1 = (n_1, n_2, ..., n_k, n_{2k}, n_{2k+1}, ..., n_{r_1-1})$, where r_1 is the unique index in $\{2k, 2k+1, ..., p-1\}$ such that $\sum_{\ell=2k}^{r_1-1} n_\ell \leq t_1$ and $\sum_{\ell=2k}^{r_1} n_\ell > t_1$. Now, if $t_1 - \sum_{\ell=2k}^{r_1-1} n_\ell > 0$, then add $n'_{r_1} = t_1 - \sum_{\ell=2k}^{r_1-1} n_\ell$ as the $(k+1)$th element of π_1. Note that the elements of π_1 sum up to $|\{v_{i_1}, v^+_{i_1}, ..., v_{i_k}\}|$.

Let $n''_{r_1} = n_{r_1} - n'_{r_1}$. If $n''_{r_1} \geq t_2$, then let $\pi_2 = (t_2)$, and set $r_2 = r_1$ and $n''_{r_2} = n''_{r_1} - t_2$. Otherwise, let r_2 be the index in $\{r_1 + 1, r_1 + 2, ..., p-1\}$ for which $n''_{r_1} + \sum_{\ell=r_1+1}^{r_2-1} n_\ell \leq t_2$ and $n''_{r_1} + \sum_{\ell=r_1+1}^{r_2} n_\ell > t_2$. Now let $\pi_2 = (n''_{r_1}, n_{r_1+1}, n_{r_1+2}, ..., n_{r_2-1})$. Set $n_{r'_2} = t_2 - (n''_{r_1} +$

$\sum_{\ell=r_1+1}^{r_2-1} n_\ell$) and $n''_{r_2} = n_{r_2} - n'_{r_2}$, and add n'_{r_2} as the second element of π_2 if $n'_{r_2} > 0$. Observe that π_2 is a $(|\{v^+_{i_{2k-1}}, (v^+_{i_{2k-1}})^+, ..., v^-_{i_1}\}| - n_0)$-sequence.

Finally, let $\pi_3 = (n_{k+1}, n_{k+2}, ..., n_{2k-1}, n''_{r_2}, n_{r_2+1}, n_{r_2+2}, ..., n_{p-1})$. Note that π_3 sums up to $|\{v^+_{i_k}, (v^+_{i_k})^+, ..., v_{i_{2k-1}}\}|$.

Remark that every element of π has been associated with one of π_1, π_2 and π_3, and at most two non-preassigned elements have been split so that the π_i's sum up exactly to the orders of some subgraphs of G. In the case where π contains one "big" non-preassigned element, it is even possible that this element was split into three integers among π_1, π_2 and π_3. To obtain the P-realization of π in G, we realize π_1, π_2 and π_3 in vertex-disjoint subgraphs of G, and this in such a way that if an original element of π was dispatched into several of the π_i's, then the associated connected subgraphs perform a whole connected subgraph when unified.

The three realizations R_1, R_2 and R_3 are obtained as follows.

- Let R_1 be a $(v_{i_1}, v_{i_2}, ..., v_{i_k}, v^+_{i_1})$-realization of π_1 in $G[\{v_{i_1}, v^+_{i_1}, ..., v_{i_k}\}]$, which exists according to Lemma 4.19 since v_{i_1} and v_{i_k} are the endvertices of $G[\{v_{i_1}, v^+_{i_1}, ..., v_{i_k}\}]$ and there are $k+1$ preassigned vertices.
- Let R_2 be a realization of π_2 in $G[\{v^+_{i_{2k-1}}, (v^+_{i_{2k-1}})^+, ..., v^-_{i_1}\} \setminus V_0]$, which is traceable by our choice of V_0. Additionally request the realization to satisfy the 2-preassignation $(v^-_{i_1}, v^+_{i_{2k+1}})$ when π_2 has at least two elements. Such a requirement is allowed according to Lemma 4.19.
- Let R_3 be a $(v_{i_{k+1}}, v_{i_{k+2}}, ..., v_{i_{2k-1}}, v^-_{i_{2k-1}})$-realization of π_3 in $G[\{v^+_{i_k}, (v^+_{i_k})^+, ..., v_{i_{2k-1}}\}]$. The existence of such a realization follows from Lemma 4.9 since $G[\{v^+_{i_k}, (v^+_{i_k})^+, ..., v_{i_{2k-1}}\}]$ is the kth power of a path whose last vertex is $v_{i_{2k-1}}$.

The P-realization of π in G is obtained by considering V_0 and the parts from R_1, R_2 and R_3, and unifying those parts whose sizes result from the split of a single element of π (if there are such). By our choice of the preassigned vertices, these parts have neighbouring vertices (this follows from the facts that $k \geq 2$, and that the preassigned vertices of P are not consecutive), and thus induce connected subgraphs. This completes the proof. ■

It is worth noting that Lemma 4.20 directly provides an alternate proof of Theorem 4.12. Indeed, suppose we want to P-realize a sequence π in C^k_n for some $(2k-1)$-preassignation P. If π has size at most $2k$, then a P-realization can be deduced using Theorem 2.1 since C^k_n is $2k$-connected. Otherwise π has size at least $2k+1$ and we can deduce a P-realization by requesting one "convenient" additional preassigned vertex, that is a vertex such that the vertices of P and this new vertex do not form two maximal preassigned blocks with size k, so that Lemma 4.20 is applicable directly.

4.3.1.2 Partitioning Harary graphs following preassignations

Before proving that every Harary graph $H_{2k+1,n}$ is indeed $2k$-preassignable arbitrarily partitionable, we first introduce the following lemma which deals with the traceability of a graph made up of two linked squares of paths.

Lemma 4.21. *If G is a graph such that $V(G) = V_1 \cup V_2$, the subgraphs $G[V_1]$ and $G[V_2]$ are both spanned by the square of a path, and there exists an edge joining one vertex of V_1 and one of V_2, then G is traceable.*

Proof. Let $v_1, v_2, ..., v_\ell$ and $u_1, u_2, ..., u_{\ell'}$ denote the consecutive vertices of $G[V_1]$ and $G[V_2]$, and $v_a \in V_1$ and $u_b \in V_2$ be two vertices of G such that $v_a u_b \in E(G)$. Consider the following subpaths of G:

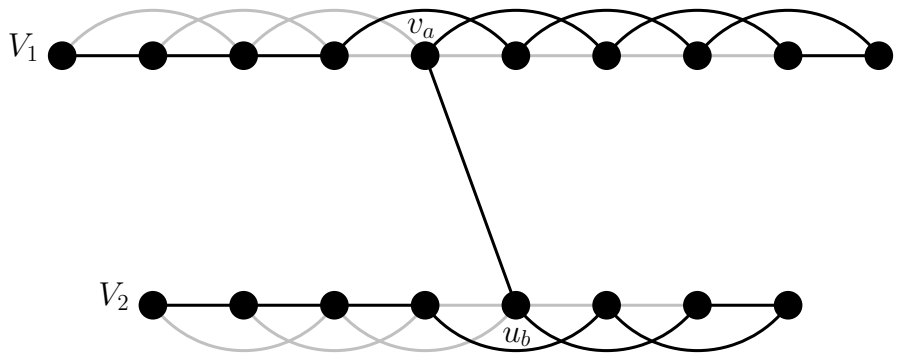


Figure 4.8: A spanning path (in black only) of a graph consisting of two linked squares of paths (in black and grey).

- $P = v_1 v_2 ... v_{a-1}$;
- $Q = \begin{cases} v_{a+1} v_{a+3} ... v_{\ell-1} v_\ell v_{\ell-2} v_{\ell-4} ... v_{a+2} \text{ if } \ell - a \text{ is even,} \\ v_{a+1} v_{a+3} ... v_\ell v_{\ell-1} v_{\ell-3} ... v_{a+2} \text{ otherwise;} \end{cases}$
- $R = \begin{cases} u_{b+2} u_{b+4} ... u_{\ell'} u_{\ell'-1} u_{\ell'-3} ... u_{b+1} \text{ if } \ell' - b \text{ is even,} \\ u_{b+2} u_{b+4} ... u_{\ell'-1} u_{\ell'} u_{\ell'-2} u_{\ell'-4} ... u_{b+1} \text{ otherwise;} \end{cases}$
- $S = u_{b-1} u_{b-2} ... u_1$.

It is then easy to check that $PQv_a u_b RS$ is an Hamiltonian path of G (see Figure 4.8). ■

We are now ready to prove our main result.

Lemma 4.22. *For every $k \geq 2$ and even $n \geq 2k+1$, the Harary graph $H_{2k+1,n}$ is $2k$-preassignable arbitrarily partitionable.*

Proof. Let $k \geq 2$ and even $n \geq 2k+1$ be fixed, and $G = H_{2k+1,n}$ be the $(2k+1)$-connected Harary graph on n vertices $v_0, v_1, ..., v_{n-1}$ (where the ordering follows the ordering of the cycle C_n spanning G). We prove that every n-sequence $\pi = (n_0, n_1, ..., n_{p-1})$ with size $p \geq 2k+1$ is P-realizable in G, where P is a $2k$-preassignation $P = (v_{i_0}, v_{i_1}, ..., v_{i_{2k-1}})$ with $0 \leq i_0 < i_1 < ... < i_{2k-1} \leq n-1$. Since G is $(2k+1)$-connected, we can actually suppose that $p > 2k+1$ since otherwise a realization can be directly obtained using Theorem 2.1.

We distinguish two main cases.

Case 1. *The preassigned vertices do not form two maximal preassigned blocks with size k.*

Because $k \geq 2$, we can directly deduce a P-realization of π in G in its spanning C_n^k via Lemma 4.20. Such a realization is a P-realization of π in G.

Case 2. *The preassigned vertices form two maximal preassigned blocks with size k.*

Denote B_1 and B_2 the two preassigned blocks. In this situation, note that $G - P$ only remains connected by means of some diagonal edges. Indeed, assume $B_1 = \{v_{i_0}, v_{i_1}, ..., v_{i_{k-1}}\}$ and $B_2 = \{v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}}\}$ without loss of generality. Then the antipodal neighbours of $v_{i_0}^-$ and $v_{i_{k-1}}^+$ cannot both belong to B_2 since otherwise there would exist a preassigned block with size at least $k+2$. Let us thus denote by v_a and v_b two antipodal neighbours of G such that $v_a, v_b \notin B_1 \cup B_2$. In particular, we may suppose that $a \in \{i_{k-1}+1, i_{k-1}+2, ..., i_k - 1\}$ and $b \in \{i_{2k-1}+1, i_{2k-1}+2, ..., i_0 - 1\}$ (the indices are here taken modulo n). Let further $a_1 = a - i_{k-1} - 1$, $a_2 = i_k - a - 1$, $a_3 = i_0 - b - 1$ and $a_4 = b - i_{2k-1} - 1$ denote the number of consecutive vertices "between" B_1, B_2 and the two vertices v_a and v_b according to the ordering of G (see Figure 4.9).

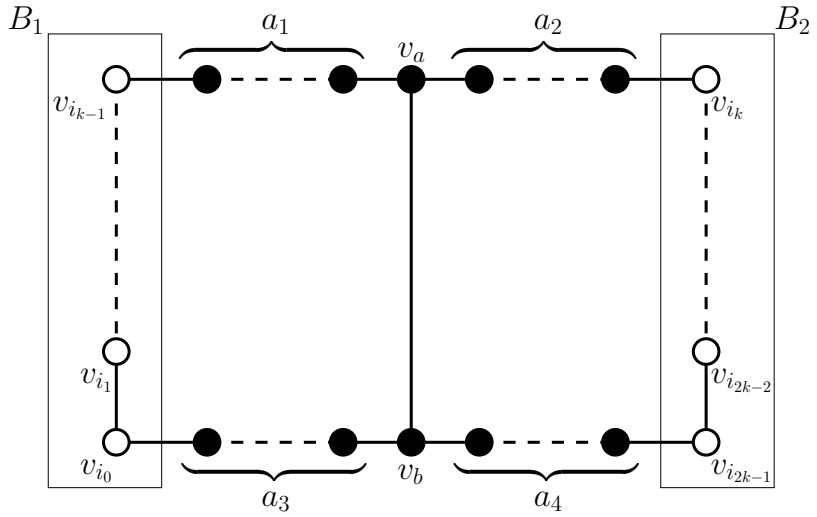


Figure 4.9: Configuration described in Case 2 of the proof of Lemma 4.22. The preassigned vertices are in white.

Case 2.1. $\sum_{j=0}^{k-1} n_j \leq a_1 + a_3 + k$ *and* $\sum_{j=k}^{2k-1} n_j \leq a_2 + a_4 + k$.

In this situation, we can find two subsets X and Y of consecutive vertices of G such that $|X| = \sum_{j=0}^{k-1} n_j$, $|Y| = \sum_{j=k}^{2k-1} n_j$, $\{v_{i_0}, v_{i_1}, ..., v_{i_{k-1}}\} \subseteq X$, $\{v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}}\} \subseteq Y$, and $v_a, v_b \notin X \cup Y$. Since $G[X]$ and $G[Y]$ are kth powers of paths, using Theorem 2.1 we can deduce a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization $(V_0, V_1, ..., V_{k-1})$ and a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization $(V_k, V_{k+1}, ..., V_{2k-1})$ of $(n_0, n_1, ..., n_{k-1})$ and $(n_k, n_{k+1}, ..., n_{2k-1})$, respectively, in $G[X]$ and $G[Y]$, respectively. Now, since $k \geq 2$, the graph $G - (X \cup Y)$ is traceable according to Lemma 4.21 and thus admits a realization $(V_{2k}, V_{2k+1}, ..., V_{p-1})$ of $(n_{2k}, n_{2k+1}, ..., n_{p-1})$. Finally, the partition $(V_0, V_1, ..., V_{p-1})$ is a P-realization of π in G.

Case 2.2. $\sum_{j=0}^{k-1} n_j > a_1 + a_3 + k$ *without loss of generality.*

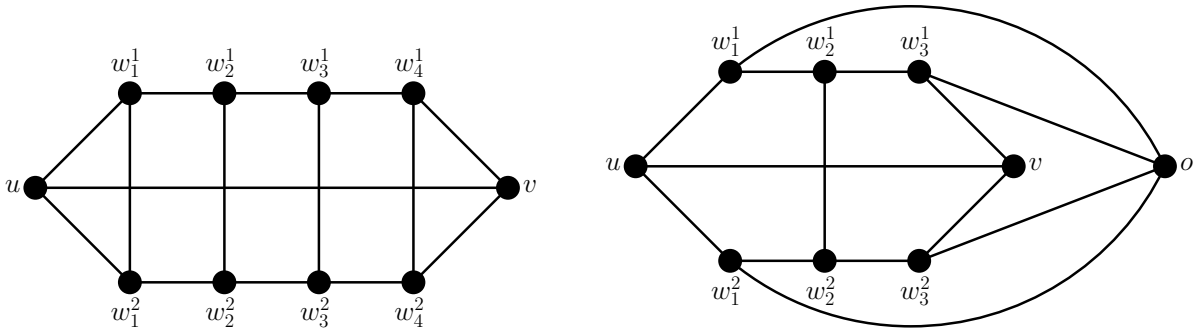
Case 2.2.1. $\sum_{j=0}^{2k-1} n_j \geq a_1 + a_2 + 2k + 1$.

Under this assumption, we can find two subsets of consecutive vertices $X, Y \subseteq V(G)$ such that $\{v_{i_0}, v_{i_1}, ..., v_{i_{k-1}}\} \subseteq X$, $\{v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}}\} \subseteq Y$, $|X| = \sum_{j=0}^{k-1} n_j$, $|Y| = \sum_{j=k}^{2k-1} n_j$, and the last vertex of $G[X]$ precedes the first vertex of $G[Y]$. By Theorem 2.1, we can deduce a $(v_{i_0}, v_{i_1}, ..., v_{i_{k-1}})$-realization $(V_0, V_1, ..., V_{k-1})$ and a $(v_{i_k}, v_{i_{k+1}}, ..., v_{i_{2k-1}})$-realization $(V_k, V_{k+1}, ..., V_{2k-1})$ of $(n_0, n_1, ..., n_{k-1})$ and $(n_k, n_{k+1}, ..., n_{2k-1})$, respectively, in $G[X]$ and $G[Y]$, respectively. Finally, since the graph $G - (X \cup Y)$ is isomorphic to the kth power of a path, there exists a realization $(V_{2k}, V_{2k+1}, ..., V_{p-1})$ of the remaining sequence $(n_{2k}, n_{2k+1}, ..., n_{p-1})$ in it. We get that $(V_0, V_1, ..., V_{p-1})$ is a P-realization of π in G.

Case 2.2.2. $\sum_{j=0}^{2k-1} n_j < a_1 + a_2 + 2k + 1$.

If it is not possible to choose the subsets X and Y in such a way that they have two neighbouring consecutive vertices along the arc $v^+_{i_{k-1}} G v^-_{i_k}$, then two such subsets can be obtained so that they have neighbouring vertices along the arc $v^+_{i_{2k-1}} G v^-_{i_0}$. Indeed, in such a situation we have $\sum_{j=0}^{k-1} n_j > a_1 + a_3 + k$ by hypothesis, but $\sum_{j=0}^{k-1} n_j < a_1 + a_2 + k + 1$. This implies that $a_2 \geq a_3$, and, since $a_1 + a_3 = a_2 + a_4$, that $a_1 \geq a_4$. Hence, we get $\sum_{j=0}^{k-1} n_j \geq a_4 + a_3 + k + 1$, which implies that the satisfying two subsets X and Y can be chosen along the arc $v^+_{i_{2k-1}} G v^-_{i_0}$. With these two subsets, a P-realization of π in G can be then obtained as in Case 2.2.1. ■

Since two Harary graphs $H_{2k+1,n}$ and $H_{2k+1,n'}$, with $k \geq 2$, and $n \geq 2k+1$ and $n' \geq 2k+1$ being even and odd, respectively, are both spanned by C^k_n, Case 1 from the proof of Lemma 4.22



(a) Pr_{10}. (b) Pr_9.

Figure 4.10: Two examples of Pr graphs.

also holds directly for Harary graphs with odd connectivity and odd order. Despite $H_{2k+1,n}$ and $H_{2k+1,n'}$ slightly differ by their diagonal edges, it is easily checked that Case 2 from the proof of Lemma 4.22 also holds when considering Harary graphs with odd connectivity and odd order. So our proof of Lemma 4.22 directly holds for these graphs.

Lemma 4.23. *For every $k \geq 2$ and odd $n \geq 2k+1$, the Harary graph $H_{2k+1,n}$ is $2k$-preassignable arbitrarily partitionable.*

Our results regarding Harary graphs sum up as follows.

Proposition 4.24. *For every $k \geq 4$ even and $n \geq k+1$, there are k-preassignable arbitrarily partitionable graphs with order n and size $\lceil \frac{n(k+1)}{2} \rceil$.*

4.3.2 On 2-preassignable arbitrarily partitionable graphs with minimum size

Recall that Lemmas 4.22 and 4.23 exclude 3-connected Harary graphs, mainly because Lemma 4.21 does not apply to these graphs. Therefore, our proof of Lemma 4.22 cannot be directly used to prove that 3-connected Harary graphs are 2-preassignable arbitrarily partitionable.

It actually turns out that 3-connected Harary graphs are not all 2-preassignable arbitrarily partitionable.

Corollary 4.25. *For every $n \equiv 2 \pmod 4$, the Harary graph $H_{3,n}$ is not 2-preassignable arbitrarily partitionable.*

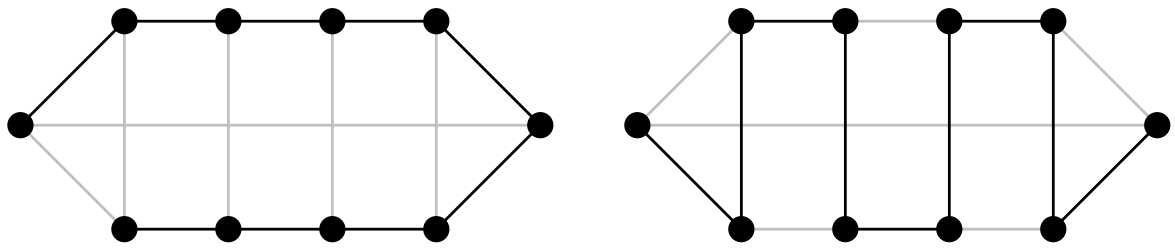
Proof. This follows from Lemma 4.4 since every such Harary graph is a balanced bipartite graph. ■

In order to prove that there exist 2-preassignable arbitrarily partitionable graphs with the minimum size indicated by Observation 4.14, we introduce the following class of 3-connected graphs.

Construction 4.26. Let $n \geq 4$. The graph Pr_n is constructed as follows.

- If n is even, then Pr_n is obtained from the cycle C_n, whose vertices are successively denoted by $u, w_1^1, w_2^1, ..., w_{\frac{n-2}{2}}^1, v, w_{\frac{n-2}{2}}^2, w_{\frac{n-2}{2}-1}^2 ..., w_1^2$, by adding the edge uv, and the edge $w_i^1 w_i^2$ for every $i \in \{1, 2, ..., \frac{n-2}{2}\}$.
- If n is odd, then Pr_n is obtained by first removing the edges $w_1^1 w_1^2$ and $w_{\frac{n-3}{2}}^1 w_{\frac{n-3}{2}}^2$ from Pr_{n-1}, and then adding a new vertex o and the edges ow_1^1, ow_1^2, $ow_{\frac{n-3}{2}}^1$, and $ow_{\frac{n-3}{2}}^2$.

Example 4.27. The graphs Pr_{10} and Pr_9 are depicted in Figure 4.10.



(a) $uP_{1,4}^{1,\rightarrow}vP_{4,1}^{2,\leftarrow}$. (b) $uP_{1,4}^{\nearrow}v$.

Figure 4.11: Two Hamiltonian $\{u, v\}$-paths (in black only) of Pr_{10} (in black and grey).

For every $n \geq 4$, the graph Pr_n is a 3-connected graph which is minimum in terms of size since it has size $\lceil \frac{3n}{2} \rceil$. To prove that Pr graphs are 2-preassignable arbitrarily partitionable, we introduce the following sufficient condition which is easier to check in our context.

Observation 4.28. *Every Hamiltonian-connected graph is* 2*-preassignable arbitrarily partitionable.*

Proof. By definition, an Hamiltonian-connected graph G has an Hamiltonian $\{u, v\}$-path for every two distinct vertices $u, v \in V(G)$. Since every path can be partitioned following every 2-preassignation involving its two endvertices, recall Lemma 4.19, we directly get that G is spanned by an arbitrarily (u, v)-partitionable subgraph. These arguments imply the claim. ■

In order to show that every graph Pr_n is Hamiltonian-connected, we first introduce some notation.

Notation 4.29. Let $G = Pr_n$ for some $n \geq 4$, and set $q = \frac{n-2}{2}$ (resp. $q = \frac{n-3}{2}$) if n is even (resp. odd). Given two distinct integers x and y taking values in $\{1, 2, ..., q\}$ (resp. $\{2, 3, ..., q-1\}$) such that $x < y$, we denote by $P_{x,y}^{\nearrow}(G)$ and $P_{x,y}^{\searrow}(G)$ the following paths of G:

$$P_{x,y}^{\nearrow}(G) = \begin{cases} w_x^2 w_x^1 \text{ if } x = y, \\ w_x^2 w_x^1 P_{x+1,y}^{\searrow}(G) \text{ otherwise;} \end{cases} \quad \text{and} \quad P_{x,y}^{\searrow}(G) = \begin{cases} w_x^1 w_x^2 \text{ if } x = y, \\ w_x^1 w_x^2 P_{x+1,y}^{\nearrow}(G) \text{ otherwise.} \end{cases}$$

The paths $P_{x,y}^{\nwarrow}(G)$ and $P_{x,y}^{\swarrow}(G)$ of G are defined analogously from right to left when $x > y$. For every $\alpha \in \{1, 2\}$, we additionally define $P_{x,y}^{\alpha,\rightarrow}(G)$ (resp. $P_{x,y}^{\alpha,\leftarrow}(G)$) for $x < y$ (resp. $x > y$) to be the path $w_x^\alpha w_{x+1}^\alpha ... w_y^\alpha$ (resp. $w_x^\alpha w_{x-1}^\alpha ... w_y^\alpha$) of G .

It is understood that we have $P_{x,y}^{\nearrow}(G) = P_{x,y}^{\searrow}(G) = P_{x,y}^{\alpha,\rightarrow}(G) = \emptyset$ (resp. $P_{x,y}^{\nwarrow}(G) = P_{x,y}^{\swarrow}(G) = P_{x,y}^{\alpha,\leftarrow}(G) = \emptyset$) whenever x and y have incorrect values (i.e. their values do not belong to the authorized set or when $x > y$ (resp. $x < y$)). To lighten the notation, we will voluntarily omit to mention the parameter G of these paths, when no ambiguity is possible.

Example 4.30. According to our terminology, note that $uP_{1,4}^{1,\rightarrow}vP_{4,1}^{2,\leftarrow}$ and $uP_{1,4}^{\nearrow}v$ are Hamiltonian paths of Pr_{10}, see Figure 4.11.

We are now ready to prove that every Pr_n graph is Hamiltonian-connected, and thus 2-preassignable arbitrarily partitionable according to Observation 4.28.

Lemma 4.31. *For every $n \geq 4$, the graph Pr_n is Hamiltonian-connected.*

Proof. Let $G = Pr_n$, and set $q = \frac{n-2}{2}$ if n is even, or $q = \frac{n-3}{2}$ otherwise. Table 4.12 (resp. Table 4.13) exhibits, given two distinct vertices s and t of G, an Hamiltonian $\{s, t\}$-path P of G when n is even (resp. odd). In Table 4.12 (resp. Table 4.13), it is assumed that $1 \leq i \leq q$ (resp. $1 < i < q$) when j is not defined, and $0 \leq i < j \leq q$ (resp. $1 < i < j < q$) otherwise.

s	t	P
u	v	$uP^{\nearrow}_{1,q}v$
u	w_i^1	$uP^{\nearrow}_{1,i-1}w_i^2P^{2,\rightarrow}_{i+1,q}vP^{1,\leftarrow}_{q,i}$ if $i-1$ is even $uP^{\searrow}_{1,i-1}w_i^2P^{2,\rightarrow}_{i+1,q}vP^{1,\leftarrow}_{q,i}$ otherwise
w_i^1	w_j^1	$P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,1}uvP^{\nwarrow}_{q,j}$ if $q-j$ is even $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,1}uvP^{\swarrow}_{q,j}$ otherwise
w_i^1	w_j^2	$P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,1}uvP^{\swarrow}_{q,j}$ if $q-j$ is even $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,1}uvP^{\nwarrow}_{q,j}$ otherwise

Table 4.12: Proof that Pr_n is Hamiltonian-connected for every even $n \geq 4$.

s	t	P
o	u	$oP^{1,\rightarrow}_{1,q}vP^{2,\leftarrow}_{q,1}u$
o	w_1^1	$ow_q^1vw_q^2P^{\nwarrow}_{q-1,2}w_1^2uw_1^1$ if q is even $ow_q^2vw_q^1P^{\swarrow}_{q-1,2}w_1^2uw_1^1$ otherwise
o	w_i^1	$ow_1^1uw_1^2P^{\nearrow}_{2,i-1}w_i^2P^{2,\rightarrow}_{i+1,q}vP^{1,\leftarrow}_{q,i}$ if i is even $ow_1^2uw_1^1P^{\searrow}_{2,i-1}w_i^2P^{2,\rightarrow}_{i+1,q}vP^{1,\leftarrow}_{q,i}$ otherwise
u	v	$uP^{2,\rightarrow}_{1,q}oP^{1,\rightarrow}_{1,q}v$
u	w_1^1	$uvP^{2,\leftarrow}_{q,1}oP^{1,\leftarrow}_{q,1}$
u	w_q^1	$uvP^{2,\leftarrow}_{q,1}oP^{1,\rightarrow}_{1,q}$
u	w_i^1	$uP^{1,\rightarrow}_{1,i-1}P^{2,\leftarrow}_{i-1,1}ow_q^2vw_q^1P^{\swarrow}_{q-1,i}$ if $q-i$ is even $uP^{1,\rightarrow}_{1,i-1}P^{2,\leftarrow}_{i-1,1}ow_q^1vw_q^2P^{\nwarrow}_{q-1,i}$ otherwise
w_1^1	w_1^2	$w_1^1uvP^{1,\leftarrow}_{q,2}P^{2,\rightarrow}_{2,q}ow_1^2$
w_1^1	w_q^1	$P^{1,\rightarrow}_{1,q-1}P^{1,\leftarrow}_{q-1,1}uvw_q^2ow_q^1$
w_1^1	w_q^2	$w_1^1ow_1^2uvP^{1,\leftarrow}_{q,2}P^{2,\rightarrow}_{2,q}$
w_i^1	w_j^1	$P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^2uw_1^1ow_q^2vw_q^1P^{\swarrow}_{q-1,j}$ if i and $q-j$ are even $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^1uw_1^2ow_q^2vw_q^1P^{\swarrow}_{q-1,j}$ if i is odd and $q-j$ is even $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^2uw_1^1ow_q^1vw_q^2P^{\nwarrow}_{q-1,j}$ if i is even and $q-j$ is odd $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^1uw_1^2ow_q^1vw_q^2P^{\nwarrow}_{q-1,j}$ if i and $q-j$ are odd
w_i^1	w_j^2	$P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^2uw_1^1ow_q^1vw_q^2P^{\nwarrow}_{q-1,j}$ if i and $q-j$ are even $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^1uw_1^2ow_q^1vw_q^2P^{\nwarrow}_{q-1,j}$ if i is odd and $q-j$ is even $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^2uw_1^1ow_q^2vw_q^1P^{\swarrow}_{q-1,j}$ if i is even and $q-j$ is odd $P^{1,\rightarrow}_{i,j-1}P^{2,\leftarrow}_{j-1,i}P^{\nwarrow}_{i-1,2}w_1^1uw_1^2ow_q^2vw_q^1P^{\swarrow}_{q-1,j}$ if i and $n-j$ are odd

Table 4.13: Proof that Pr_n is Hamiltonian-connected for every odd $n \geq 5$.

Every Hamiltonian path which does not appear in Table 4.12 or Table 4.13 can be deduced from another Hamiltonian path using the symmetries of G. ■

Proposition 4.32. *For every $n \geq 4$, there are 2-preassignable arbitrarily partitionable graphs with order n and size $\lceil \frac{3n}{2} \rceil$.*

We gather Propositions 4.15, 4.24 and 4.32 within the next concluding result.

Theorem 4.33. *For every $k \geq 1$ and $n \geq k+1$, there are k-preassignable arbitrarily partitionable graphs with order n and size $\lceil \frac{n(k+1)}{2} \rceil$.*

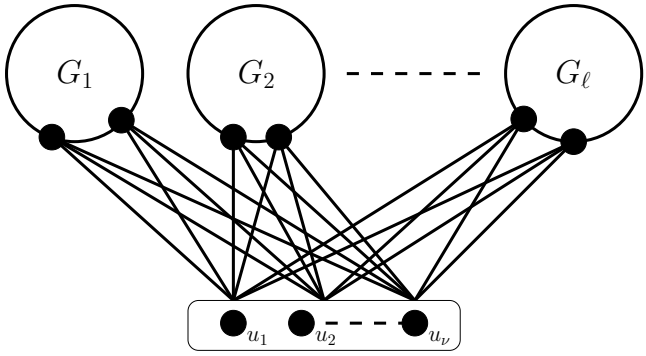


Figure 4.14: A sample graph $K_\nu(G_1, G_2, ..., G_\ell)$.

4.4 On the order of the longest paths

Along the previous sections, we have pointed out that every traceable graph is arbitrarily partitionable (Observation 2.34), every Hamiltonian graph is 1-preassignable arbitrarily partitionable (Observation 4.5), and every Hamiltonian-connected graph is 2-preassignable arbitrarily partitionable (Observation 4.28). We herein show that no Hamiltonian property necessarily appears in preassignable arbitrarily partitionable graphs. More precisely, we show that a k-preassignable arbitrarily partitionable graph can have its longest paths being arbitrarily smaller than its order, and can hence be arbitrarily far from being even traceable. For this purpose, we show that a k-preassignable arbitrarily partitionable graph can have an arbitrarily large path cover number.

The results mentioned above are obtained by studying the following family of graphs.

Construction 4.34. Consider two positive integers $\nu \geq 1$ and $\ell \geq 1$, and ℓ graphs $G_1, G_2, ..., G_\ell$. As $K_\nu(G_1, G_2, ..., G_\ell)$, we refer to the graph obtained as follows (see Figure 4.14):

1. take the disjoint union of $G_1, G_2, ..., G_\ell$,
2. add ν new vertices $u_1, u_2, ..., u_\nu$ to the graph,
3. turn $u_1, u_2, ..., u_\nu$ into universal vertices.

We now exhibit a series of easy remarks related to the path cover number of graphs. As a first result, observe that the path cover number of a graph $K_\nu(G_1, G_2, ..., G_\ell)$ is related to the path cover numbers of $G_1, G_2, ..., G_\ell$.

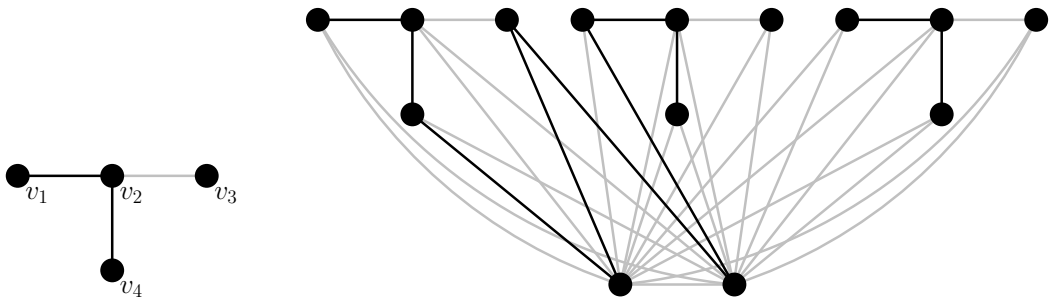
Observation 4.35. *For every $\nu \geq 1$ and $\ell \geq 1$ graphs $G_1, G_2, ..., G_\ell$, we have $\mu(K_\nu(G_1, G_2, ..., G_\ell)) = (\sum_{i=1}^{\ell} \mu(G_i)) - \nu$.*

Proof. Typically a minimum path cover of $K_\nu(G_1, G_2, ..., G_\ell)$ is obtained by considering minimum path covers of $G_1, G_2, ..., G_\ell$ and then "glueing" the endvertices of some of the resulting paths by using the universal vertices to get longer paths (see Figure 4.15). In particular, since $K_\nu(G_1, G_2, ..., G_\ell)$ has ν universal vertices, we can "replace" $\nu + 1$ vertex-disjoint paths $P_1, P_2, ..., P_{\nu+1}$ from minimum path covers of $G_1, G_2, ..., G_\ell$ with the long path

$$P_1 u_1 P_2 u_2 P_3 ... P_\nu u_\nu P_{\nu+1},$$

where $u_1, u_2, ..., u_\nu$ are the universal vertices of $K_\nu(G_1, G_2, ..., G_\ell)$. ■

Previous Observation 4.35 implies the following.



(a) A graph G with $\mu(G) = 2$.

(b) We have $\mu(K_2(G, G, G)) = 3\mu(G) - 2 = 4$.

Figure 4.15: Deducing a minimum path cover (in black only) of a graph $K_\nu(G_1, G_2, ..., G_\ell)$ (in black and grey).

Observation 4.36. *Let $\nu \geq 1$ and $G_1, G_2, ..., G_\ell$ be $\ell \geq 1$ graphs with $\mu(G_1) \geq \mu(G_2) \geq ... \geq \mu(G_\ell)$. If $\nu < \sum_{i=2}^{\ell} \mu(G_i)$, then $\mu(K_\nu(G_1, G_2, ..., G_\ell)) > \mu(G_1)$.*

Proof. The result follows by just replacing $\mu(K_\nu(G_1, G_2, ..., G_\ell))$ in the inequality $\mu(K_\nu(G_1, G_2, ..., G_\ell)) > \mu(G_1)$ with its explicit value given by Observation 4.35. ■

We now point out the following relationship between $\mu(G)$ and $\varsigma(G)$ for every graph G.

Observation 4.37. *For every graph G, we have $\varsigma(G) \leq |V(G)| - \mu(G) + 1$.*

Proof. Assume the claim is not true. Then $\mu(G) > |V(G)| - \varsigma(G) + 1$. Now consider a (not necessarily minimum) path cover C of G including one path with order $\varsigma(G)$. Clearly we have $|C| \leq |V(G)| - \varsigma(G) + 1$ since, in the worst case, all paths of C which do not have order $\varsigma(G)$ are trivial. We hence have

$$|V(G)| - \varsigma(G) + 1 < \mu(G) \leq |C| \leq |V(G)| - \varsigma(G) + 1,$$

a contradiction. ■

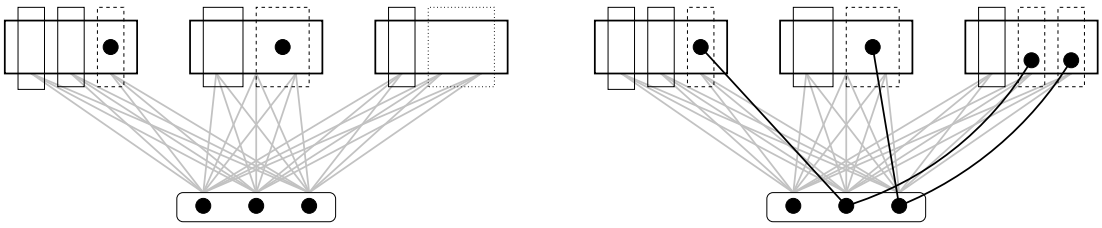
On the path cover number of k-preassignable arbitrarily partitionable graphs

We herein show that, given a non-traceable k-preassignable arbitrarily partitionable graph, we can construct k-preassignable arbitrarily partitionable graphs with arbitrarily large path cover numbers (and hence arbitrarily small longest paths according to Observation 4.37). This relies on the following lemma.

Lemma 4.38. *Let $k \geq 1$, $\nu \geq k + 1$, and $G_1, G_2, ..., G_\ell$ be $\ell \geq 1$ k-preassignable arbitrarily partitionable graphs. Then $K_\nu(G_1, G_2, ..., G_\ell)$ is k-preassignable arbitrarily partitionable if and only if $\nu \geq k + \ell - 1$.*

Proof. Let $G = K_\nu(G_1, G_2, ..., G_\ell)$ be a graph with order n for given k-preassignable arbitrarily partitionable graphs $G_1, G_2, ..., G_\ell$, and denote $u_1, u_2, ..., u_\nu$ the universal vertices of G. Consider further an n-sequence $\pi = (n_1, n_2, ..., n_p)$ and a k-preassignation $P = (v_1, v_2, ..., v_k)$ of G.

Our strategy for deducing a P-realization of π in G relies on the fact that if a partial part of the realization is missing several vertices but contains one of the universal vertices, say u_1, of G, then we can easily complete this part with arbitrary unused vertices since u_1 neighbours every vertex of G. This property allows us to proceed as follows. Consider one of the G_i's, say G_1, and pick as many parts of the realization as possible in G_1 using the fact that G_1 is arbitrarily partitionable. In case some parts fit exactly in G_1, i.e. there is a subsequence of π which sums up to $|V(G_1)|$, everything is fine. Otherwise, we can "fill" G_1 with some parts, but one part



(a) Filling Step: we pick as many connected parts in the components as possible. Full parts are complete, dashed parts are missing vertices, and dotted parts are extra parts used during the Completing Step to complete some incomplete parts.

(b) Completing Step: we include a universal vertex to every incomplete part so that it can be completed with vertices from extra parts (via the black edges).

Figure 4.16: Illustration of the Filling and Completing Steps.

exceeds from G_1. In this situation, we have to add a universal vertex to this part so that it can be completed later (if necessary). The only things we have to make sure of are that we respect the preassignation P when constructing the connected parts, and that not too many parts have to be completed (i.e. at most ν, the number of universal vertices) once we have filled all of the G_i's with as many parts as possible.

We now explicit the two steps necessary to make use of the strategy described above, which we refer to as the Filling and Completing Steps. An illustration of these two steps is depicted in Figure 4.16.

Filling Step.

The Filling Step consists in two steps: we first deal with the preassigned universal vertices before considering the G_i's and filling them as described above.

Step 1. For every vertex $v_i \in P \cap U$, where $U = \{u_1, u_2, ..., u_\nu\}$ is the set of universal vertices of G, start with the partial part $\{v_i\}$ which will be completed during the Completing Step (this will be possible since this part already contains a universal vertex).

Step 2. At any moment of the procedure, we denote by $\pi_r = (r_1, r_2, ..., r_q)$ the sequence made up of the *remaining* non-preassigned part sizes of π, i.e. which have not been considered yet. In particular, we start with $\pi_r = (n_{k+1}, n_{k+2}, ..., n_p)$ (but we always refer to the elements of π_r as $r_1, r_2, ..., r_q$ for the sake of simplicity).

Now consider the G_i's in order. We assume throughout that we deal with G_1 for the sake of clarity. We distinguish several cases.

Case 1: $V(G_1) \cap P = \emptyset$.

We are in the situation where G_1 contains no preassigned vertices. If there is an $i \in \{1, 2, ..., q\}$ such that $r_1 + r_2 + ... + r_i = |V(G_1)|$, then the parts with size $r_1, r_2, ..., r_i$ of the realization can be deduced by considering a realization of $(r_1, r_2, ..., r_i)$ in G_1, which exists since G_1 is arbitrarily partitionable. In this situation, no part has to be completed.

If $r_1 + r_2 + ... + r_q < |V(G_1)|$ (this in particular occurs when $|\pi_r| = 0$), then we can deduce a realization of $(r_1, r_2, ..., r_q, |V(G_1)| - (r_1 + r_2 + ... + r_q))$ in G_1, which again exists. Again, no part has to be completed (actually, the vertices from the part with size $|V(G_1)| - (r_1 + r_2 + ... + r_q)$ will be available during the Completing Step to fill some partial parts).

If we are not in one of the two previous cases, then there is an $i \in \{1, 2, ..., q\}$ such that $r_1 + r_2 + ... + r_{i-1} < |V(G_1)|$ and $r_1 + r_2 + ... + r_i > |V(G_1)|$. Let r'_i and r''_i be two positive

integers such that $r_i = r'_i + r''_i$, and $r_1 + r_2 + ... + r_{i-1} + r'_i = |V(G_1)|$. Since G_1 is arbitrarily partitionable, we can deduce a realization of $(r_1, r_2, ..., r_{i-1}, r'_i)$ in G_1. In this situation, the part V_i with size r'_i is incomplete and has to be completed with $r''_i \geq 1$ additional vertices. Then pick an unused vertex from U, and add it to V_i so that V_i can be completed during the Completing Step (if necessary, i.e. if $r''_i \geq 2$).

In every of these three situations, remove from π_r the elements which have already been treated.

Case 2: *G_1 includes t of the preassigned vertices, where $1 \leq t \leq k$.*

Denote $v_1, v_2, ..., v_t$ the vertices of $V(G_1) \cap P$. We consider three main cases. At first, if $n_1 + n_2 + ... + n_t = |V(G_1)|$, then just consider a $(v_1, v_2, ..., v_t)$-realization of $(n_1, n_2, ..., n_t)$ in G_1, which exists since G_1 is k-preassignable arbitrarily partitionable with $k \geq t$, and hence t-preassignable arbitrarily partitionable, recall Observation 4.3. Note that in doing so, no partial part will have to be completed during the Completing Step.

Now, if on the one hand we have $n_1 + n_2 + ... + n_t > |V(G_1)|$, then let $i \in \{1, 2, ..., t\}$ be the value for which $n_1+n_2+...+n_{i-1}+(t-(i-1)) < |V(G_1)|$ and $n_1+n_2+...+n_i+(t-i) > |V(G_1)|$. Set $n'_i = |V(G_1)| - (n_1+n_2+...+n_{i-1}) - (t-i)$. Since G_1 is k-preassignable arbitrarily partitionable with $k \geq t$, we can deduce a $(v_1, v_2, ..., v_t)$-realization of $(n_1, n_2, ..., n_{i-1}, n'_i, 1, 1, ..., 1)$, where the value 1 is repeated $t - i$ times, in G_1, using e.g. Theorem 2.1 since G_1 is $(k+1)$-connected according to Observation 4.2. In doing so, note that all of the preassigned vertices $v_1, v_2, ..., v_t$ belong to distinct parts. Besides, the parts with size $n_1, n_2, ..., n_{i-1}$ are complete, while the ith part is missing $n_i - n'_i$ vertices, and the other parts are missing all but one vertex (except if these parts are intended to have size 1). Then add an unused universal vertex in each of the parts which have to be completed (there are at most $t - i + 1$ of them).

If, on the other hand, we have $n_1+n_2+...+n_t < |V(G_1)|$, then proceed as follows. First, if π_r is empty, then just consider a $(v_1, v_2, ..., v_t)$-realization of $(n_1, n_2, ..., n_t, |V(G_1)| - (n_1+n_2+...+n_t))$ in G_1, which exists since G_1 is k-preassignable arbitrarily partitionable with $k \geq t$, and hence t-preassignable arbitrarily partitionable according to Observation 4.3. The vertices from the resulting part with size $|V(G_1)| - (n_1 + n_2 + ... + n_t)$ will actually be used during the Completing Step to complete some partial parts. Second, if π_r has elements, then let $i \in \{1, 2, ..., q+1\}$ be the maximum index for which $n_1 + n_2 + ... + n_t + r_1 + r_2 + ... + r_{i-1} < |V(G_1)|$. In case $i = q+1$, i.e. all of the parts with size $n_1, n_2, ..., n_t$ as well as those whose sizes are not preassigned can be picked in G_1, just consider a $(v_1, v_2, ..., v_t)$-realization of

$$(n_1, n_2, ..., n_t, r_1, r_2, ..., r_q, |V(G_1)| - (n_1 + n_2 + ... + n_t + r_1 + r_2 + ... + r_q))$$

in G_1, which again exists since G_1 is t-preassignable arbitrarily partitionable. Again, the vertices from the extra part will be available for the Completing Step. If we are not in this case, i.e. $i < q + 1$, then again split r_i into two integers r'_i and r''_i (with $r_i = r'_i + r''_i$) such that $n_1 + n_2 + ... + n_t + r_1 + r_2 + ... + r_{i-1} + r'_i = |V(G_1)|$, and consider a $(v_1, v_2, ..., v_t)$-realization of $(n_1, n_2, ..., n_t, r_1, r_2, ..., r_{i-1}, r'_i)$ in G_1. Again, add an unused universal vertex to the part with size r'_i so that it can eventually be completed with $r''_i - 1$ additional vertices during the Completing Step.

Again, after every of these cases, remove the first elements from π_r whose associated connected subgraphs have already been (possibly partially) picked.

Completing Step.

Once the Filling Step is achieved, every part V_i of the (possibly partial) realization is either complete, i.e. it already has size n_i, or is missing some vertices but V_i contains a universal vertex by construction. Then just add $n_i - |V_i|$ unused vertices to V_i, i.e. vertices which belong to some extra parts. This part still induces a connected subgraph since it contains a universal

vertex. Repeating this operation for every partial part missing some vertices, we eventually get a partition of $V(G)$, which is a P-realization of π in G.

Regarding the correctness of the whole process, note that all of the parts induce connected subgraphs, have the correct sizes, and include, if required, a preassigned vertex. Furthermore note that the number of partial parts we have to deal with during the Completing Step is at most $k+\ell-1$. Indeed, by Step 1 at most k such partial parts may result (this upper bound is typically reached in the extremal case where all preassigned vertices are also universal), while at most $\ell-1$ such partial parts may arise during Step 2 (this is reached when the picked parts exceed from the $\ell-1$ first G_i's considered). The P-realization of π in G is then always eventually obtained when $\nu \geq k+\ell-1$, while one can imagine situations in which the Filling and Completing Steps cannot provide a solution when $\nu < k+\ell-1$. ■

Using Lemma 4.38, we now prove the main result of this section.

Theorem 4.39. *Let $k \geq 1$ and $G_1, G_2, ..., G_{k+2}$ be $k+2$ copies of a k-preassignable arbitrarily partitionable graph G. If G is not traceable, then $\mu(K_{2k+1}(G_1, G_2, ..., G_{k+2})) > \mu(G)$ and $K_{2k+1}(G_1, G_2, ..., G_{k+2})$ is k-preassignable arbitrarily partitionable.*

Proof. Since $2k+1 \geq k+(k+2)-1$, the graph $K_{2k+1}(G_1, G_2, ..., G_{k+2})$ is k-preassignable arbitrarily partitionable according to Lemma 4.38. Besides, since we have

$$\mu(G_1) = \mu(G_2) = ... = \mu(G_{k+2}) = \mu(G) \geq 2$$

by the non-traceability of G, we get

$$\sum_{i=2}^{k+2} \mu(G_i) \geq 2(k+1) > 2k+1.$$

We then also have $\mu(K_{2k+1}(G_1, G_2, ..., G_{k+2})) > \mu(G)$ according to Observation 4.36, as claimed. ■

Corollary 4.40. *Provided a non-traceable k-preassignable arbitrarily partitionable graph, we can construct k-preassignable arbitrarily partitionable graphs with arbitrarily large path cover number and arbitrarily small longest paths (compared to their order).*

Proof. Let G be such a non-traceable k-preassignable arbitrarily partitionable graph, and let $(G_0, G_1, G_2, ..., G_q)$ be the sequence of $q+1$ graphs defined inductively as follows.

- $G_0 = G$.
- For every $i \in \{1, 2, ..., q\}$, set $G_i = K_{2k+1}(G_{i-1}, G_{i-1}, ..., G_{i-1})$ where G_i is made up of $k+2$ components isomorphic to G_{i-1}.

According to Theorem 4.39, all of the graphs from $(G_0, G_1, G_2, ..., G_q)$ are k-preassignable arbitrarily partitionable since $G_0 = G$ is k-preassignable arbitrarily partitionable by assumption. Besides, we have

$$\mu(G_q) > \mu(G_{q-1}) > ... > \mu(G_0).$$

For this reason, if we write $\varsigma(G_i) = |V(G_i)| - c_i$ for every $i \in \{0, 1, ..., q\}$, where $c_i \geq 1$ is an integer, then we get

$$c_q > c_{q-1} > ... > c_0$$

according to Observation 4.37. This completes the proof. ■

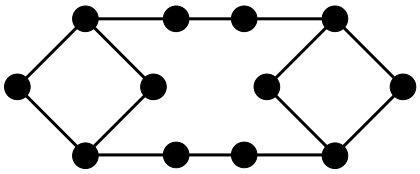


Figure 4.17: A non-traceable 1-preassignable arbitrarily partitionable graph.

On the existence of non-traceable k-preassignable arbitrarily partitionable graphs

Corollary 4.40 relies on the assumption that there exist, for some $k \geq 1$, non-traceable k-preassignable arbitrarily partitionable graphs. We describe below an inductive construction providing a non-traceable k-preassignable arbitrarily partitionable graph from $(k-1)$-preassignable arbitrarily partitionable graphs with large path cover numbers. As a basis, this construction then requires a non-traceable 1-preassignable arbitrarily partitionable graph. Consider the graph depicted in Figure 4.17. This graph is obviously non-traceable, and we previously showed in [29] that this graph is also 1-preassignable arbitrarily partitionable (checking this property is easy due to its symmetric structure and small order). Using Corollary 4.40, we hence directly get the following.

Corollary 4.41. *1-preassignable arbitrarily partitionable graphs can have arbitrarily large path cover number and arbitrarily small longest paths (compared to their order).*

Assuming that $(k-1)$-preassignable arbitrarily partitionable graphs can have arbitrarily large path cover number, we show below that we can deduce a non-traceable k-preassignable arbitrarily partitionable graph.

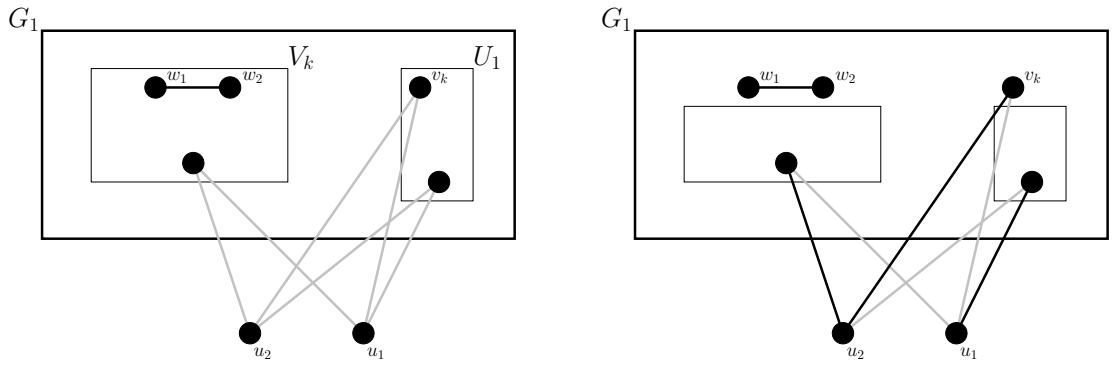
Lemma 4.42. *Let $k \geq 1$, $\nu \geq k+1$, and $G_1, G_2, ..., G_\ell$ be $\ell \geq 1$ $(k-1)$-preassignable arbitrarily partitionable graphs. Then $K_\nu(G_1, G_2, ..., G_\ell)$ is k-preassignable arbitrarily partitionable if and only if $\nu \geq \max\{k+\ell-1, \ell+2\}$.*

Proof. Let $G = K_\nu(G_1, G_2, ..., G_\ell)$ be a graph with order n for given graphs $G_1, G_2, ..., G_\ell$. Let further $\pi = (n_1, n_2, ..., n_p)$ be an n-sequence, and $P = (v_1, v_2, ..., v_k)$ be a k-preassignation of G. We exhibit a P-realization of π in G in a very same manner as in the proof of Lemma 4.38, i.e. by using the Filling and Completing Steps. We hence use the same terminology throughout this proof.

Every situation described in the proof of Lemma 4.38 can actually be handled similarly since $\nu \geq \max\{k+\ell-1, \ell+2\} \geq k+\ell-1$. The only new situation we have to consider is when the k preassigned vertices are all located in G_1 while G_1 is "only" $(k-1)$-preassignable arbitrarily partitionable. In such a situation the Filling Step has to be handled as follows. If $n_1 + n_2 + ... + n_k = |V(G_1)|$, then we can deduce a $(v_1, v_2, ..., v_k)$-realization of $(n_1, n_2, ..., n_k)$ in G_1 using Theorem 2.1 since G_1 is $(k-1)$-preassignable arbitrarily partitionable and is hence k-connected (see Observation 4.2). No part has to be completed via the Completing Step in this situation.

In case $n_1+n_2+...+n_k > |V(G_1)|$, we can proceed as in Step 2 from the proof of Lemma 4.38. Let $i \in \{1, 2, ..., k\}$ be the index for which $n_1 + n_2 + ... + n_{i-1} + (k-(i-1)) < |V(G_1)|$ and $n_1 + n_2 + ... + n_i + (k-i) > |V(G_1)|$. Again, set $n'_i = |V(G_1)| - (n_1 + n_2 + ... + n_{i-1}) - (k-i)$. By the k-connectivity of G_1, using Theorem 2.1 we can deduce a $(v_1, v_2, ..., v_k)$-realization of $(n_1, n_2, ..., n_{i-1}, n'_i, 1, 1, ..., 1)$, where the value 1 is repeated $k-i$ times at the end of the sequence, in G_1. Again, each i'th part, with $i' \geq i$, is (possibly) incomplete, so just add a universal vertex to the corresponding partial part so that it can be completed during the Completing Step (if needed).

Finally consider the case where $n_1 + n_2 + ... + n_k < |V(G_1)|$. Then let $i \in \{1, 2, ..., q\}$ be the index for which $n_1 + n_2 + ... + n_k + r_1 + r_2 + ... + r_{i-1} < |V(G_1)|$ and $n_1 + n_2 + ... + n_k +$



(a) Initial situation: the preassigned vertex v_k has to be moved from U_1 to V_k. The vertices u_1 and u_2 are universal.

(b) The parts $V_k \setminus \{w_1, w_2\} \cup \{v_k, u_2\}$ and $U_1 \setminus \{v_k\} \cup \{u_1\}$ induce connected parts (in black only).

Figure 4.18: Moving a preassigned vertex from a part to another one using two universal vertices.

$r_1 + r_2 + ... + r_i > |V(G_1)|$. We can suppose that such an index exists since π_r cannot be empty at the beginning of the Filling Step (otherwise by the k-connectivity of G we could directly deduce a P-realization of π using Theorem 2.1). In particular, if a component includes the k preassigned vertices, then consider this component as G_1 first so that π_r is not empty. Again, split r_i into two non-null elements r'_i and r''_i such that $r_i = r'_i + r''_i$ and $n_1 + n_2 + ... + n_k + r_1 + r_2 + ... + r_{i-1} + r'_i = |V(G_1)|$. Consider a $(v_1, v_2, ..., v_{k-1})$-realization $(V_1, V_2, ..., V_k, U_1, U_2, ..., U_i)$ of $(n_1, n_2, ..., n_k, r_1, r_2, ..., r_{i-1}, r'_i)$ in G_1, which exists since G_1 is $(k-1)$-preassignable arbitrarily partitionable. The only requirement which may not be fulfilled is the membership of v_k to V_k. If this is already met, then we are done. Otherwise, we modify the parts as follows.

Assume $v_k \in U_1$ for the sake of simplicity. Let further u_1 and u_2 be two unused universal vertices of G. Now let

$$U'_1 = U_1 \setminus \{v_k\} \cup \{u_1\}$$

and

$$V'_k = V_k \setminus \{w_1, w_2\} \cup \{u_2, v_k\},$$

where w_1 and w_2 are any two adjacent vertices of V_k. Clearly $G[U'_1]$ is connected due to the presence of u_1 in this subgraph (which still has order r_1 (or r'_1)). Similarly, the part V'_k still induces a connected subgraph (due to the presence of u_2) on n_k vertices, and now contains v_k. Now w_1 and w_2 belong to no part, but then we can try to complete the part U_i with w_1 and w_2 (recall that U_i is missing r''_i vertices). We may suppose that $i \neq 1$ since otherwise all the arguments below hold directly since U'_1 already contains a universal vertex.

If $r''_i = 1$, then the part is only missing one vertex, so add a universal vertex to it, and create (possibly) partial parts of the realization U_{i+1} and possibly U_{i+2} (intended to have size r_{i+1} and r_{i+2}, respectively) containing w_1, w_2, and at most one universal vertex, and inducing connected subgraphs (if $r_{i+1} \geq 2$, then U_{i+1} is sufficient). If $r''_i = 2$, then add a universal vertex to U_i, as well as, say, w_1, so that U_i has the required size r_i. Now create the (possibly partial) part U_{i+1} by adding w_2 to U_{i+1}, as well as one universal vertex if $r_{i+1} \geq 2$. In any case, w_2 belongs to a part, which can be completed during the Completing Step if necessary. Now if $r''_i \geq 3$, then add w_1, w_2, and a universal vertex to the part U_i.

Again, after every of these strategies, remove the part sizes of π_r whose associated parts have already been (possibly partially) picked.

The important thing to be aware of, is that the number of needed universal vertices is $\max\{k + \ell - 1, \ell + 2\}$. The value $k + \ell - 1$ follows from the same arguments as in the proof of

Lemma 4.38. For the new situations specific to the current proof, note that, in the worst case, we may need two universal vertices to "move" the preassigned vertex v_k from a part to another one, plus another universal vertex to complete a part which is missing only one vertex (i.e. when $r_i'' = 1$), and a last universal vertex to complete the part which is intended to contain w_1 and w_2. Then what remain are $\ell - 1$ components which do not include preassigned vertices, and $\nu - 4$ universal vertices. As seen in the proof of Lemma 4.38, we then need at most $\ell - 2$ universal vertices to pick the remaining parts in the remaining components. For this additional case, we thus need at most $4 + \ell - 2 = \ell + 2$ universal vertices. Again, for bad values of ν, we can deduce situations where a P-realization of π in G cannot be obtained. ■

We are now ready to express the main result of this section.

Theorem 4.43. *For every $k \geq 1$, k-preassignable arbitrarily partitionable graphs can have arbitrarily large path cover number and arbitrarily small longest paths (compared to their order).*

Proof. The claim is already true for $k = 1$, recall Corollary 4.41. Now assume it is true for every value of k up to an $i - 1 \geq 1$, and put $k = i$. The only thing we need to show is that there exists a non-traceable k-preassignable arbitrarily partitionable graph G so that Corollary 4.40 directly implies the claim. Said differently, we want G to have $\mu(G) \geq 2$.

Choose $\nu \geq \max\{k + 1, 4\}$, and let G_1 and G_2 be two $(k-1)$-preassignable arbitrarily partitionable graphs satisfying $\mu(G_1) + \mu(G_2) \geq \nu + 2$. Such graphs exist according to the induction hypothesis. Now set $G = K_\nu(G_1, G_2)$. Then G is a k-preassignable arbitrarily partitionable graph according to Lemma 4.42, and is not traceable since we have

$$\mu(G) = \mu(G_1) + \mu(G_2) - \nu$$

according to Observation 4.35, implying $\mu(G) \geq 2$. This completes the proof. ■

4.5 Cartesian products

Similarly as in Section 3.5, we herein focus on Cartesian products $G \,\square\, H$ involving preassignable arbitrarily partitionable graphs, and more particularly on the situations where H is traceable. The very first observation we raise is that if G is k-preassignable arbitrarily partitionable, then $G \,\square\, P_\ell$ is not necessarily $(k + 1)$-preassignable arbitrarily partitionable. Said differently, the edges of $G \,\square\, P_\ell$ coming from P_ℓ are not sufficient to gain one additional preassignable vertex. The interesting fact is that this does not follow directly from Observation 4.2 since $G \,\square\, P_\ell$ has connectivity $\kappa(G) + 1$ for every $\ell \geq 2$.

Observation 4.44. *If $\ell \geq 2$ and G is a k-preassignable arbitrarily partitionable graph, then $G \,\square\, P_\ell$ is not necessarily $(k + 1)$-preassignable arbitrarily partitionable.*

Proof. Let $G = C_4$ be the cycle with order 4. Then G is 1-preassignable arbitrarily partitionable according to Observation 4.5. Note then that $G \,\square\, P_2$ is isomorphic to Q_3, the hypercube of dimension 3, which is a balanced bipartite graph. Such a graph is not 2-preassignable arbitrarily partitionable according to Lemma 4.4. ■

The argument given in the proof of Observation 4.44 extends to Cartesian products of the form $C_n \,\square\, P_\ell$ with even $n \geq 4$ and every $\ell \geq 1$. One direction of interest is hence to consider whether the number of vertices we can preassign while partitioning a Cartesian product involving a preassignable arbitrarily partitionable graph is preserved. We formulate this in the following weaker form.

Conjecture 4.45. *Let $k \geq 1$ and $\ell \geq 1$ be two positive integers. If a graph G is k-preassignable arbitrarily partitionable, then so is $G \,\square\, P_\ell$.*

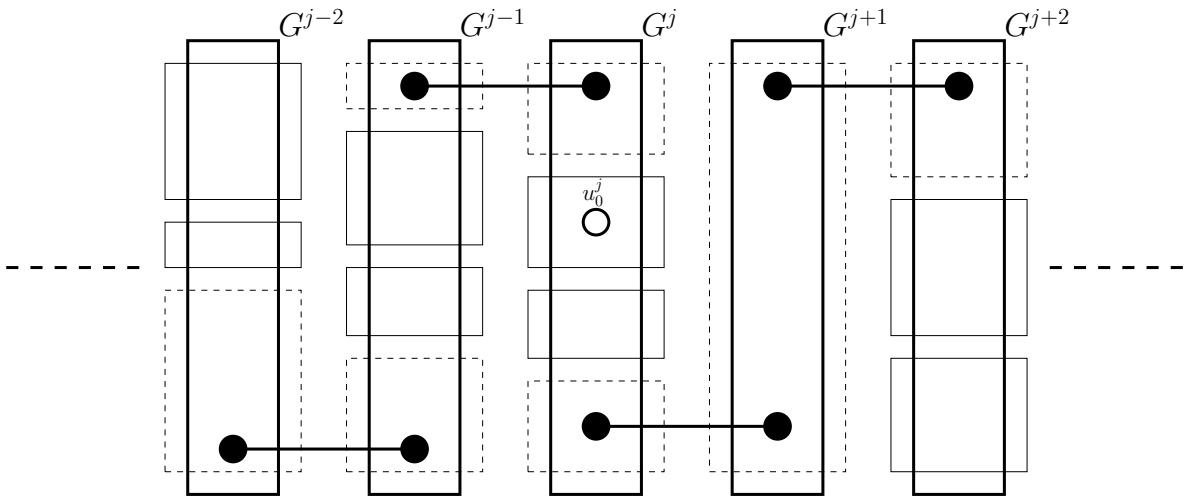


Figure 4.19: Unifying connected parts from successive layers of $G \,\square\, P_\ell$. Dotted parts joined by an edge are parts whose sizes result from the splitting of a same element from the original sequence (Case 1).

We here support Conjecture 4.45 by showing it to hold for $k = 1$.

Theorem 4.46. *For every integer $\ell \geq 1$ and 1-preassignable arbitrarily partitionable graph G, the Cartesian product $G \,\square\, P_\ell$ is 1-preassignable arbitrarily partitionable.*

Proof. Set $n = |V(G)|$, and let $\pi = (n_1, n_2, ..., n_p)$ be an $n\ell$-sequence and u_0^j be a vertex of $G \,\square\, P_\ell$. We explain how to find a (u_0^j)-realization of π in $G \,\square\, P_\ell$. If $\ell = 1$, then $G \,\square\, P_\ell$ is isomorphic to G, so a realization can be deduced directly since G is 1-preassignable arbitrarily partitionable by assumption. Suppose thus that $\ell \geq 2$.

We distinguish two main cases depending on the value of n_1. For each of these cases, the proof is led in two steps. Roughly, we first possibly modify π by dividing some of its terms into two or more addends, so that π can be partitioned into ℓ sequences $\pi_1, \pi_2, ..., \pi_\ell$ satisfying $\|\pi_i\| = n$ for every $i \in \{1, 2, ..., \ell\}$. Then, for every $i \in \{1, 2, ..., \ell\}$, we construct a realization of π_i in the ith layer G^i of $G \square P_\ell$ in such a way that parts whose sizes result from the splitting of a same element of π yield a connected subgraph when glued together.

Case 1. $n_1 < n$.

Consider the sequence $\pi' = (n_2, n_3, ..., n_p)$. For every $i \in \{1, 2, ..., \ell\}$, we define a *capacity* $\varphi(i)$ for the ith layer G^i as follows:

$$\varphi(i) = \begin{cases} n - n_1 & \text{if } i = j, \\ n & \text{otherwise.} \end{cases}$$

We now recursively define ℓ sequences $\pi_1, \pi_2, ..., \pi_\ell$ in such a way that each π_i sums up to $\varphi(i)$ as follows. If there is an index $q \in \{2, 3, ..., p\}$ such that $\sum_{i=2}^{q} n_i = \varphi(1)$, then we set $\pi_1 = (n_2, n_2, ..., n_q)$, $\pi'' = (n_{q+1}, n_{q+2}, ..., n_p)$, and $\alpha(1) = 0$ (for each $i \in \{1, 2, ..., \ell - 1\}$, the number $\alpha(i)$ indicates whether the last term of π_i originates from the division of a term of π). Otherwise, let $q \in \{2, 3, ..., p\}$ be such that $\sum_{i=2}^{q-1} n_i < \varphi(1)$ and $\sum_{i=2}^{q} n_i > \varphi(1)$, and $\eta_1 = \varphi(1) - \sum_{i=2}^{q-1} n_i$ and $\eta_2' = n_q - \eta_1$. We then set $\pi_1 = (n_2, n_3, ..., n_{q-1}, \eta_1)$, $\pi'' = (\eta_2', n_{q+1}, n_{q+2}, ..., n_p)$, and $\alpha(1) = 1$.

Repeat this procedure with the sequence π'' and $\varphi(2)$ (instead of π' and $\varphi(1)$) in order to obtain a sequence π_2 and the number $\alpha(2)$, and so on. After ℓ repetitions of this procedure, we obtain the desired sequences $\pi_1, \pi_2, ..., \pi_\ell$. We write $\pi_i = (n_1^i, n_2^i, ..., n_{p_i}^i)$ for every $i \in \{1, 2, ..., \ell\}$, where $p_i \geq 1$. The important detail to keep in mind is that if an element n_i of π has been

dispatched into two sequences π_{j_1} and π_{j_2} (with $j_2 = j_1 + 1$), then the two resulting addends are the last element of π_{j_1} and the first element of π_{j_2}, respectively.

We begin the construction of a (u_0^j)-realization of π in $G \,\square\, P_\ell$ by choosing a (u_0^j)-realization $(V_1,\ V_1^j,\ V_2^j,\ ...,\ V_{p_j}^j)$ of $(n_1, n_1^j, n_2^j, ..., n_{p_j}^j)$ in G^j. Such exists since $(n_1, n_1^j, n_2^j, ..., n_{p_j}^j)$ is an n-sequence and G^j is 1-preassignable arbitrarily partitionable. Next, for every successive i in $(j-1, j-2, ..., 1)$, we proceed as follows (in case $j = 1$, just skip this step). If $\alpha(i) = 0$, then we take an arbitrary realization $(V_1^i, V_2^i, ..., V_{p_i}^i)$ of π_i in G^i. Otherwise, i.e. $\alpha(i) = 1$, we choose a vertex $u^{i+1} \in V_1^{i+1}$ and consider a realization $(V_1^i, V_2^i, ..., V_{p_i}^i)$ of π_i in G^i satisfying $u^i \in V_{p_i}^i$. This additional requirement can be imposed since G^i is 1-preassignable arbitrarily partitionable. Note that the subgraph induced by $V_{p_i}^i \cup V_1^{i+1}$ in $G \,\square\, P_\ell$ is connected via the edge $u^i u^{i+1}$.

Now for successive i in $(j+1, j+2, ..., \ell)$ (if $j = \ell$, then skip this step), if $\alpha(i-1) = 0$, then consider any realization $(V_1^i, V_2^i, ..., V_{p_i}^i)$ of π_i in G^i. Otherwise, i.e. $\alpha(i-1) = 1$, we choose a vertex $u^{i-1} \in V_{p_{i-1}}^{i-1}$, and choose the realization of π_i in G^i so that $u^i \in V_1^i$. This is again possible since G^i is 1-preassignable arbitrarily partitionable. For the same argument as above, the subgraph $(G \,\square\, P_\ell)[V_{p_{i-1}}^{i-1} \cup V_1^i]$ is connected.

We then obtain a realization

$$R = (V_1, V_1^1, V_2^1, ..., V_{p_1}^1, V_1^2, V_2^2, ..., V_{p_2}^2, ..., V_1^\ell, V_2^\ell, ..., V_{p_\ell}^\ell)$$

of the sequence

$$(n_1, n_1^1, n_2^1, ..., n_{p_1}^1, n_1^2, n_2^2, ..., n_{p_2}^2, ..., n_1^\ell, n_2^\ell, ..., n_{p_\ell}^\ell)$$

in $G \,\square\, P_\ell$. Suppose that in the procedure of defining the sequences $\pi_1, \pi_2, ..., \pi_\ell$, a certain term n_ν of π has been divided into $s \geq 2$ addends $n_{p_i}^i, n_1^{i+1}, ..., n_1^{i+s-1}$. In particular, we have $p_{i+1} = p_{i+2} = ... = p_{i+s-2} = 1$ if $s \geq 3$. Set $V_\nu = V_{p_i}^i \cup V_1^{i+1} \cup ... \cup V_1^{i+s-1}$. By our construction, the subgraph $(G \,\square\, P_\ell)[V_\nu]$ is connected, see Figure 4.19. Then replace the parts $V_{p_i}^i, V_1^{i+1}, ..., V_1^{i+s-1}$ of R with V_ν. A (u_0^j)-realization of π in $G \,\square\, P_\ell$ is eventually obtained by repeating the same operation for all divided terms of π.

Case 2. $n_1 \geq n$.

Let $d = \lfloor \frac{n_1}{n-1} \rfloor$ and $r = n_1 \pmod{(n-1)}$ so that $n_1 = d(n-1) + r$. In order to make the proof more legible, we distinguish two cases.

Case 2.1. $d < j$.

In such a situation, we have to pick the connected part V_1 including u_0^j in the "middle" of $G \,\square\, P_\ell$, that is we cannot just pick all vertices from all layers located on the left of the jth one. This has to be done in such a way that $(G \,\square\, P_\ell) - V_1$ remains connected, since otherwise there is a high risk that the realization of π cannot be deduced.

To take care of this mater, we modify the capacity function φ as follows:

$$\varphi(i) = \begin{cases} 1 & \text{if } i \in \{j-d+1, j-d+2, ..., j\}, \\ n-r & \text{if } i = j-d, \\ n & \text{otherwise.} \end{cases}$$

Then set $\pi' = (n_2, n_3, ..., n_p)$, and define the sequences $\pi_1, \pi_2, ..., \pi_\ell$ and the integers $\alpha(1), \alpha(2), ..., \alpha(\ell)$ similarly as in Case 1 (but using the new capacity function φ).

Choose a vertex $v \in V(G) \setminus \{u_0\}$. We start by deducing a realization in G^{j-d}. In case $r \neq 0$, deduce a realization $(W,\ V_1^{j-d},\ V_2^{j-d},\ ...,\ V_{p_{j-d}}^{j-d})$ of $(r, n_1^{j-d}, n_2^{j-d}, ..., n_{p_{j-d}}^{j-d})$ in G^{j-d} satisfying $v^{j-d} \in V_{p_{j-d}}^{j-d}$. In case $r = 0$, consider any realization $(V_1^{j-d}, V_2^{j-d}, ..., V_{p_{j-d}}^{j-d})$ of π_{j-d} in G^{j-d} verifying $v^{j-d} \in V_{p_{j-d}}^{j-d}$. Such realizations exist since G^{j-d} is 1-preassignable arbitrarily partitionable.

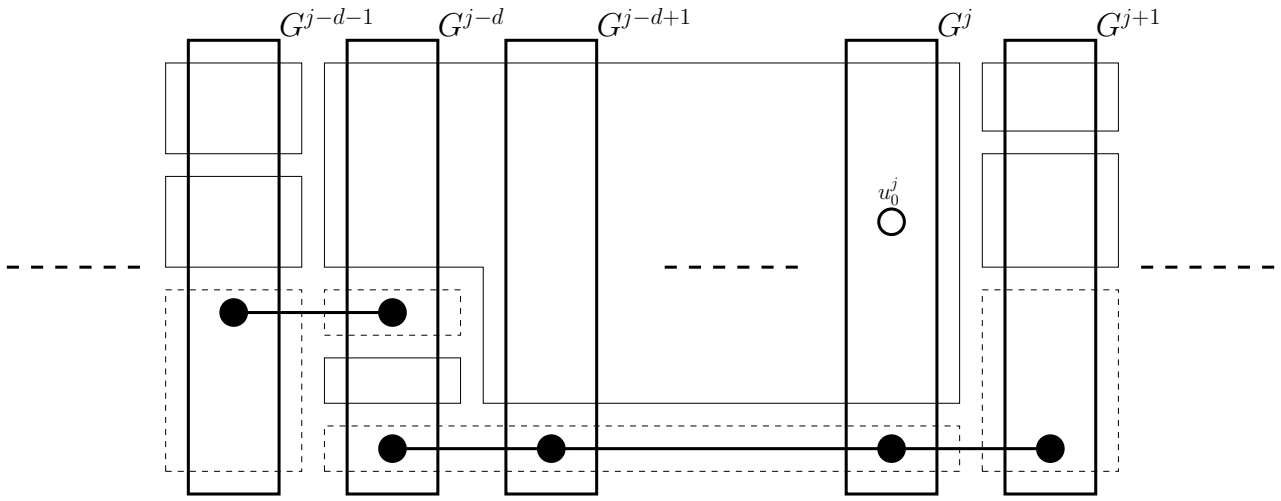


Figure 4.20: Unifying connected parts from successive layers of $G \,\square\, P_\ell$. Dotted parts joined by an edge are parts whose sizes result from the splitting of a same element from the original sequence (Case 2.1).

Then, we consecutively produce realizations of $\pi_{j-d-1}, \pi_{j-d-2}, ..., \pi_1$ in $G^{j-d-1}, G^{j-d-2}, ..., G^1$, respectively, following the same rules as in Case 1. That is, for every successive i in $(j-d-1, j-d-2, ..., 1)$, when $\alpha(i) = 1$ we choose a vertex $u^{i+1} \in V_1^{i+1}$, and, using the fact that G is 1-preassignable arbitrarily partitionable, we deduce a realization $(V_1^i, V_2^i, ..., V_{p_i}^i)$ of π^i in G^i verifying $u^i \in V_{p_i}^i$. If $\alpha(i) = 0$, then we deduce an arbitrary realization instead. For every $i \in \{j-d+1, j-d+2, ..., j\}$, we consider $(\{v^i\})$ as a realization of $\pi_i = (1)$. Next, for successive i in $(j+1, j+2, ..., \ell)$ we proceed exactly in the same way as in Case 1. Finally set

$$V_1 = W \cup \bigcup_{i=j-d+1}^{j} (V(G^i) \setminus \{v^i\}),$$

which induces a connected subgraph of $(G \,\square\, P_\ell)[V_1]$. The same arguments as in Case 1 then imply that we can unify some subsets of the realization

$$(V_1, V_1^1, V_2^1, ..., V_{p_1}^1, V_1^2, V_2^2, ...V_{p_2}^2, ..., V_1^\ell, V_2^\ell, ..., V_{p_\ell}^\ell)$$

of the sequence

$$(n_1, n_1^1, n_2^1, ..., n_{p_1}^1, n_1^2, n_2^2, ...n_{p_2}^2, ..., n_1^\ell, n_2^\ell, ..., n_{p_\ell}^\ell)$$

in $G \,\square\, P_\ell$ in order to get a (u_0^j)-realization of π in $G \,\square\, P_\ell$, see Figure 4.20.

Case 2.2. $d \geq j$.

If $p = 1$, then $\pi = (n\ell)$ and π is trivially realizable in $G \,\square\, P_\ell$. For $p \geq 2$, we choose a vertex $v \in V(G) \setminus \{u_0\}$ and define

$$V_1 = (\bigcup_{i=1}^{r} V(G^i)) \cup (\bigcup_{i=r+1}^{d} (V(G^i) \setminus \{v^i\})).$$

Clearly V_1 induces a connected subgraph including u_0^j. Then consider the capacity function φ defined as

$$\varphi(i) = \begin{cases} 1 & \text{if } i \in \{r+1, r+2, ..., d\}, \\ n & \text{if } i \in \{d+1, d+2, ..., \ell\}, \end{cases}$$

and deduce sequences $\pi_{r+1}, \pi_{r+2}, ..., \pi_\ell$ and their respective realizations in $G^{r+1}, G^{r+2}, ..., G^\ell$ using the same rules as in Case 1. We can then eventually obtain a (u_0^j)-realization $(V_1, V_2, ..., V_p)$ of π in $G \,\square\, P_\ell$ similarly as previously, i.e. where each of $V_2, V_3, ..., V_p$ possibly results from the union of several parts picked independently in consecutive layers of $G \,\square\, P_\ell$. ■

4.6 Conclusion and open questions

The goal of this chapter was to study preassignable arbitrarily partitionable graphs, and especially to give very first properties of these graphs. We have first exhibited several families of preassignable arbitrarily partitionable graphs with less edges than complete graphs. In Section 4.2, we have showed that powers of traceable or Hamiltonian graphs can be partitioned even when the maximum number of vertices (indicated by their connectivity) is preassigned, these results generalizing easy observations regarding the partition of traceable or Hamiltonian graphs. We have then considered Harary graphs and another class of 3-connected graphs in Sections 4.3.1 and 4.3.2, respectively. As a main result from our investigations of these two classes of graphs, we have showed that there are k-preassignable arbitrarily partitionable graphs with order n and the least possible size (i.e. the one specified by Observation 4.14) for every $k \geq 1$ and $n \geq k+1$, recall Theorem 4.33. Due to the strong relationship between powers of paths or cycles and Harary graphs, as another consequence of our investigations we have also exhibited situations in which powers of paths or cycles can be partitioned under more preassigned vertices than indicated by their connectivity, typically when the preassigned vertices do not form a cutset (recall Lemmas 4.9, 4.18, 4.19 and 4.20).

It would be interesting considering the same concerns regarding arbitrarily P-partitionable graphs. We in particular address the following question:

Question 4.47. *For given $k \geq 1$ and $n \geq k+1$, what is the minimum size of a graph with order n which is arbitrarily P-partitionable for some k-tuple P of vertices?*

Since a k-preassignable arbitrarily partitionable graph is arbitrarily P-partitionable for every k-tuple P of its vertices by definition, by Theorem 4.33 we directly get, for every $k \geq 1$ and $n \geq k+1$, that $\lceil \frac{n(k+1)}{2} \rceil$ is an upper bound on the quantity asked in Question 4.47. But this upper bound should not be tight since an arbitrarily P-partitionable graph does not need to be as dense as a k-preassignable arbitrarily partitionable graph (in particular, an arbitrarily P-partitionable graph should not need to be $(|P|+1)$-connected). To be convinced of this statement, just recall that a path with order n and endvertices u and v is arbitrarily (u,v)-partitionable (Observation 2.34) and has size $n-1$, while a 2-preassignable arbitrarily partitionable graph with order n has size at least $\lceil \frac{3n}{2} \rceil$ (Observation 4.14). So it would be interesting studying how these two quantities of edges can differ. Answering Question 4.47 would also be interesting regarding our investigations from Section 3.3.3 of Chapter 3, wherein we have considered a family of graphs made up of several arbitrarily P-partitionable graphs. This would in particular give us an indication on how dense these particular graphs can be.

Inspired by those investigations regarding the relationship between arbitrarily partitionable graphs and Hamiltonian graphs, we have then considered the order of the longest paths in a preassignable arbitrarily partitionable graph in Section 4.4. In particular, we have provided several graph constructions showing that k-preassignable arbitrarily partitionable graphs can be arbitrarily not traceable for every $k \geq 1$, i.e. that $\varsigma(G)$ can be arbitrarily smaller than $|V(G)|$ in general (recall Theorem 4.43).

Similarly as for arbitrarily partitionable graphs, recall Section 3.5, we have then considered the construction of k-preassignable arbitrarily partitionable graphs using the Cartesian product of graphs in Section 4.5. In this vein, we have mainly considered whether $G \,\square\, P_\ell$ can always be partitioned following a k-preassignation assuming G is k-preassignable arbitrarily partitionable itself, recall Conjecture 4.45. Towards this conjecture, we have verified the case $k = 1$, so the question for all remaining cases is still open and is thus one potential direction for future works. A mere feeling we have is that a proof for the general case could rely on an induction on both k and ℓ, and a distinction of all main ways for the preassigned vertices to be located in $G \,\square\, P_\ell$. In case e.g. some of the k preassigned vertices are located on the "left side" of $G \,\square\, P_\ell$ while the

remaining preassigned vertices are located on the "right side" of $G \square P_\ell$, we could then use the induction on k regarding one "left subgraph" and one "right subgraph" of $G \square P_\ell$ to obtain two realizations satisfying parts of the preassignation, and then try to unify these two realizations to get one whole realization.

In case Conjecture 4.45 is tackled, one could then turn his concern to the following counterpart of Conjecture 3.44 for preassignable arbitrarily partitionable graphs, which seems way harder to tackle.

Conjecture 4.48. *If two graphs G and H are k-preassignable arbitrarily partitionable, then so is $G \square H$.*

Further directions for studying k-preassignable arbitrarily partitionable graphs are roughly the same as for arbitrarily partitionable graphs. Studying general structural properties of k-preassignable arbitrarily partitionable graphs appears to be a convenient direction. It could be especially interesting investigating what subgraphs cannot appear in a k-preassignable arbitrarily partitionable graph. As an illustration, removing k vertices from a k-preassignable arbitrarily partitionable graph G results in a graph which admits a (possibly quasi-) perfect matching, implying that some local substructures cannot occur in G.

A point behind such considerations is that such studies could have consequences on the algorithmic aspects related to k-preassigned arbitrarily partitionable graphs, in particular on the status of k-Preassignable Arbitrarily Partitionable Graph, which we still do not know much about (as briefly mentioned in Section 4.1). Such advancement could in turn have algorithmic consequences on Arbitrarily Partitionable Graph due to the strong relationship between Arbitrarily Partitionable Graph and k-Preassignable Arbitrarily Partitionable Graph. We hence raise the following counterpart of Question 3.55.

Question 4.49. *Is there a complexity class C such that k-*Preassignable Arbitrarily Partitionable Graph *is C-complete?*

Due to the similarities between the property of being k-preassignable arbitrarily partitionable and other classic notions of graph theory where one aims at removing vertices from a graph to make some properties appear (like e.g. the notion of feedback vertex set, which attracted a lot of attention, see the survey [58] by Festa, Pardalos and Resende), other natural directions for future works can surely be suggested by the literature.

Chapter 5

On-line and recursively arbitrarily partitionable graphs

We herein consider the two recursive variants of the notion of arbitrarily partitionable graphs, namely on-line and recursively arbitrarily partitionable graphs. After giving some easy remarks and observations related to on-line or recursively arbitrarily partitionable graphs in Section 5.1, and pointing out some hardness evidences regarding On-Line Arbitrarily Partitionable Graph and Recursively Arbitrarily Partitionable Graph in Section 5.2, we then mainly focus on the structure of on-line and recursively arbitrarily partitionable graphs.

Our first concern is to step towards a variant of Theorem 2.23 for recursively arbitrarily partitionable graphs. Towards such, we show, in Section 5.3, that deleting a k-cutset from a recursively arbitrarily partitionable cannot result in arbitrarily many components, that is at most $4k - 1$. We then focus more specifically on the case $k = 2$ in Section 5.4 by studying the structure of on-line arbitrarily partitionable balloons, pursuing works initiated by Baudon, Gilbert and Woźniak in [24]. Our results in this scope yield structural properties of on-line or recursively arbitrarily partitionable graphs with a 2-cutset, see summarising Section 5.4.4.

We finally investigate, in Section 5.5, how far from traceable a recursively arbitrarily partitionable graph can be. In particular, we exhibit two families of recursively arbitrarily partitionable graphs illustrating the fact that the longest paths of a recursively arbitrarily partitionable graph G can be arbitrarily smaller than $|V(G)|$ (up to an additive factor).

The results presented in Section 5.5 were published in [31]. All results presented in Section 5.4 were obtained jointly with Baudon, Foucaud and Pilśniak and have been submitted for publication [15].

5.1 Preliminary remarks and observations

Throughout this chapter, we often make use of the following equivalent condition, first given by Gilbert [64], for a graph to be recursively arbitrarily partitionable, which is usually easier to check than the condition from Definition 2.7.

Observation 5.1 ([64]). *A graph G is recursively arbitrarily partitionable if either*

- $G \simeq K_1$, *or*
- *for every $\lambda \in \{1, 2, ..., \lfloor \frac{|V(G)|}{2} \rfloor\}$, there is a bipartition $V_\lambda \cup V_{|V(G)|-\lambda}$ of $V(G)$ such that $G[V_\lambda]$ and $G[V_{|V(G)|-\lambda}]$ are recursively arbitrarily partitionable, and have order λ and $|V(G)| - \lambda$, respectively.*

We now introduce some known structural properties of arbitrarily partitionable balloons. The following first observation, which is again due to Gilbert [64], follows from the non-existence of a (possibly quasi-) perfect matching in a balloon under certain circumstances (this is actually nothing but an application of Theorem 2.24).

Observation 5.2 ([64]). *A balloon B is not arbitrarily partitionable if*

- *$|V(G)|$ is even and B has at least three odd branches,*
- *$|V(G)|$ is odd and B has at least four odd branches.*

The following lemma is nothing but a particular case of Theorem 2.23, though it was first proved in this specific form by Baudon, Gilbert and Woźniak in [24].

Lemma 5.3 ([24]). *If $B(b_1, b_2, ..., b_k)$ is an arbitrarily partitionable balloon satisfying $b_1 \leq b_2 \leq ... \leq b_k$, then we have*

$$2b_i \geq \sum_{j=1}^{i-1} b_j$$

for every $i \in \{2, 3, ..., k\}$.

5.2 Algorithmic remarks

Due to the fact that the number of recursive calls needed to check whether a graph G is on-line or recursively arbitrarily partitionable is not constant but dependent of $|V(G)|$, the membership of either of On-Line Arbitrarily Partitionable Graph and Recursively Arbitrarily Partitionable Graph to the polynomial hierarchy seems improbable. As suggested by this observation, PSPACE is a more legitimate candidate for the membership of these two problems. We confirm it in the following result.

Theorem 5.4. On-Line Arbitrarily Partitionable Graph *and* Recursively Arbitrarily Partitionable Graph *are in PSPACE.*

Proof. By definition, for each of On-Line Arbitrarily Partitionable Graph and Recursively Arbitrarily Partitionable Graph we just have to propose a polynomial-space algorithm solving it. We herein only propose such an algorithm for Recursively Arbitrarily Partitionable Graph, but it should be clear that every algorithm for Recursively Arbitrarily Partitionable Graph can easily be modified for On-Line Arbitrarily Partitionable Graph due to the likeness of the two underlying checking processes (compare Definition 2.6 and Observation 5.1).

```
1  if G ≃ K_1 then
2      return true;
3  else if G is not connected then
4      return false;
5  else
6      foreach λ ∈ {1, 2, ..., ⌊|V(G)|/2⌋} do
7          foreach subset V_λ ⊂ V(G) with size λ do
8              if G[V_λ] and G − V_λ are recursively arbitrarily partitionable then
9                  Consider the next value of λ;
10         return false;
11     return true;
```

Algorithm 1: Determining whether a graph G is recursively arbitrarily partitionable.

The correctness of Algorithm 1 should be clear as it is nothing but a straight implementation of Observation 5.1. At Line 8, it is understood that Algorithm 1 is recursively invoked twice. Recall that we do not care about the time complexity used by the whole process. Line 3 can be achieved using $\mathcal{O}(|V(G)|)$ space, i.e. by just performing e.g. a breadth-first search algorithm using an array of booleans in order to memorize the already encountered vertices. All subsets V_λ mentioned at Line 7 can be generated using an array of $|V(G)|$ booleans, where the ith one is intended to depict the membership (or not) of the ith vertex of G (following an arbitrary ordering of $V(G)$) to V_λ. So at each step of the recursion, the algorithm uses $\mathcal{O}(|V(G)|)$ space. The recursion depth is $|V(G)|$ since the induced subgraphs are obtained by removing at least one vertex (typically for $\lambda = 1$) from G. Dedicating one array of $|V(G)|$ booleans for each specific level of the recursion, i.e. one array for calls at depth 1, one array for calls at depth 2, etc., we get that the whole process uses $\mathcal{O}(|V(G)|^2)$ space. ■

5.3 Removing k-cutsets from recursively arbitrarily partitionable graphs

As suggested by Theorem 2.25, an equivalent to Theorem 2.23 for recursively arbitrarily partitionable graphs should be quite different. We herein only focus on the maximum number of components which may arise after the removal of a k-cutset from a recursively arbitrarily partitionable graph.

While deleting a k-cutset of an arbitrarily partitionable graph may result in arbitrarily many components, recall Theorem 2.23, a quick investigation on small recursively arbitrarily partitionable graphs suggests that this property should not hold for these graphs. From our investigations, we believe the following should be a reasonable upper bound on the maximum number of such resulting components.

Conjecture 5.5. *Removing a cutset with size $k \geq 1$ from a recursively arbitrarily partitionable graph results in at most $2k + 1$ components.*

The value $2k + 1$ from Conjecture 5.5 is only a pure guess based on investigations on small values of k, so it is not based on strong formal arguments. It is however worth mentioning that Conjecture 5.5 is verified for $k = 2$, recall Theorem 2.25, while the case $k = 1$ follows as a special case of upcoming Theorem 5.6 (though verifying it by hand should be an easy exercise).

We herein prove a weaker version of Conjecture 5.5 where $2k + 1$ is replaced with $4k - 1$. Although weaker than Conjecture 5.5, this result nevertheless implies that removing a cutset from a recursively arbitrarily partitionable graph cannot yield arbitrarily many components.

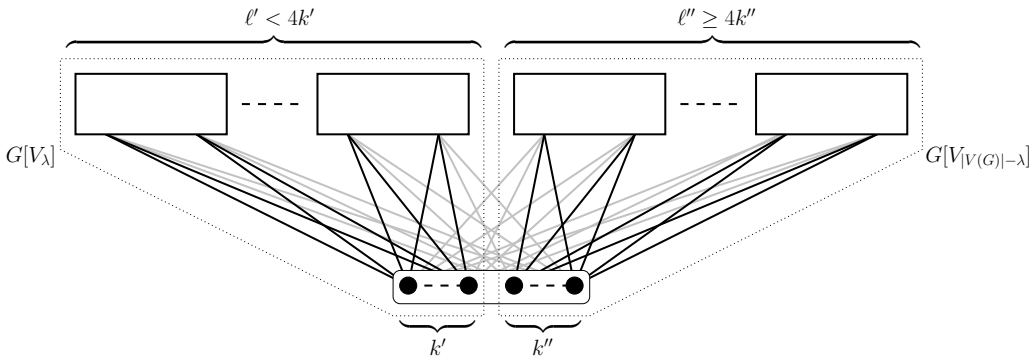


Figure 5.1: Situation described in the proof of Theorem 5.6. A compound graph G (in black and grey) with k roots and at least $4k$ components is bipartitioned into two connected subgraphs $G[V_\lambda]$ and $G[V_{|V(G)|-\lambda}]$ (in black only) each of which contains root vertices. One of $G[V_\lambda]$ and $G[V_{|V(G)|-\lambda}]$ is then a compound graph with too many components (compared to its number of roots) and hence cannot be recursively arbitrarily partitionable.

Theorem 5.6. *Removing a cutset with size $k \geq 1$ from a recursively arbitrarily partitionable graph results in at most $4k-1$ components.*

Proof. We show the claim by induction on k. At each step of the induction, i.e. for a given value of k, we consider a counterexample G to the claim, i.e. G is recursively arbitrarily partitionable and has a k-cutset whose removal result in $\ell \geq 4k$ components, and show that G cannot exist. So that some of our arguments are correct, we further suppose that G is minimal in terms of order. According to Observation 2.27, we may actually suppose that G is a (k,ℓ)-compound graph $C_{k,\ell}(K_{n_1+k}, K_{n_2+k}, ..., K_{n_\ell+k})$ made up of complete graphs since if such a graph contradicts the claim, then so does G according to the spanning argument. The integers $n_1, n_2, ..., n_\ell$ refer to the orders of the ℓ components resulting from the deletion of the k-cutset of G made up of all its root vertices.

Consider a value of k whose previous values have already been successively treated (this can be $k = 1$). We make use of Observation 5.1 (and its associated terminology) throughout to deal with the property of being recursively arbitrarily partitionable. Let $u_1, u_2, ..., u_k$ denote the root vertices of G. For a given value of $\lambda \in \{1, 2, ..., \lfloor \frac{|V(G)|}{2} \rfloor\}$, note that if both V_λ and $V_{|V(G)|-\lambda}$ contain some of the u_i's (this case can only occur when $k \geq 2$), then one of $G[V_\lambda]$ and $G[V_{|V(G)|-\lambda}]$ cannot be recursively arbitrarily partitionable according to the induction hypothesis. Indeed, suppose V_λ contains k' of the u_i's, and includes non-root vertices from ℓ' components constituting G. Denote similarly k'' and ℓ'' these integers regarding $V_{|V(G)|-\lambda}$. Clearly, we have $k = k' + k''$ and $\ell \leq \ell' + \ell''$. If $G[V_\lambda]$ is recursively arbitrarily partitionable, then $\ell' < 4k'$ according to the induction hypothesis. We then get that $\ell'' \geq 4k''$, and thus that $G[V_{|V(G)|-\lambda}]$ cannot be recursively arbitrarily partitionable according to the induction hypothesis, as claimed. This situation is illustrated in Figure 5.1.

Thus, one of the two parts of the partition, say V_λ, only contains non-root vertices from one component of G, say K_{n_1+k}, while $V_{|V(G)|-\lambda}$ includes all the other vertices of G (in particular, the k roots of G belong to $V_{|V(G)|-\lambda}$). Note that V_λ actually has to include all n_1 non-root vertices of K_{n_1+k} since otherwise the remaining graph $G[V_{|V(G)|-\lambda}]$ would be isomorphic to

$$C_{k,\ell}(K_{n_1-\lambda+k}, K_{n_2+k}, K_{n_3+k}, ..., K_{n_\ell+k}),$$

which cannot be recursively arbitrarily partitionable by the minimality of G.

Since $|V(G)| \geq 5k$, we must consider λ up to at least $2k+1$. In particular, consider every $\lambda \in \{1, 2, ..., 2k+1\}$. Because of the arguments above and $\ell \geq 4k$, we get that for every such value of λ, at least one of the n_i's is equal to λ. Thus at least one of the n_i's is equal to 1, at

least one of them is equal to 2, and so on up to $2k+1$. Our goal is to show, using Theorem 2.24, that G has no (possibly quasi-) perfect matching. Note that if sufficiently many of the n_i's have been revealed to be odd, then Theorem 2.24 is directly applicable. Otherwise, note that, because the revealed n_i's take value among $\{1, 2, ..., 2k+1\}$, actually $|V(G)|$ is larger than $5k$. So we can actually consider larger values of λ and, by applying the above arguments again, reveal one or two additional odd n_i's, and then invoke Theorem 2.24 again to show that G is not arbitrarily partitionable. In any case, because G is not arbitrarily partitionable, it cannot be recursively arbitrarily partitionable, recall Theorem 2.19, a contradiction. ■

5.4 Structural properties of on-line arbitrarily partitionable balloons

This section is dedicated to the study of the second part of Theorem 2.23 in the context of on-line arbitrarily partitionable graphs having 2-cutsets. Our results are more precisely obtained by studying the family of on-line arbitrarily partitionable balloons, and are in the line of some investigations initiated by Baudon, Gilbert and Woźniak in [24].

5.4.1 Number of branches

So that we prove an on-line version of Theorem 2.25, we herein show that an on-line arbitrarily partitionable balloon cannot have more than five branches. Since a recursively arbitrarily partitionable graph is also on-line arbitrarily partitionable, see Theorem 2.19, our result directly provides a slightly different proof of Theorem 2.25.

Theorem 5.7. *An on-line arbitrarily partitionable balloon has at most five branches.*

Proof. The proof is by contradiction. Let $\mathcal{B}$ be the set of on-line arbitrarily partitionable balloons with at least six branches, and B denote a k-balloon of $\mathcal{B}$ whose order n is minimal (with regards to all members of $\mathcal{B}$).

Recall that B is on-line arbitrarily partitionable if and only if, for every $\lambda \in \{1, 2, ..., n-1\}$, there is an on-line λ-partition of B. Actually, because of the minimality of B, the subgraph $B[V_{n-\lambda}]$ cannot be a partial k'-balloon with $k' \geq 6$ since otherwise there would exist a balloon of $\mathcal{B}$ with less vertices than B according to Observation 2.27. So $B[V_{n-\lambda}]$ can be either an on-line arbitrarily partitionable 5-balloon (since $B[V_\lambda]$ has to be connected) or an on-line arbitrarily partitionable tree (i.e. a tree from Table 2.2.a).

We claim that B has branches with order 1, 2, 3, 4, 5, and 6. Consider successive values of λ in $(1, 2, ..., 6)$. We show that if B does not have a branch with order λ, then B cannot be on-line λ-partitioned.

- $\lambda \in \{1, 2, 3\}$: for every choice of V_λ inducing a connected subgraph, the subgraph $B[V_{n-\lambda}]$ is either a partial k-balloon having less vertices than B, or a tree with maximum degree at least 4. In both cases, the subgraph $B[V_{n-\lambda}]$ is not on-line arbitrarily partitionable.

- $\lambda = 4$: so far, we know that B necessarily has branches with order 1, 2 and 3. Similarly as in the previous case, observe that for every choice of V_λ inducing a connected subgraph, the subgraph $B[V_{n-\lambda}]$ is either a partial k- or $(k+1)$-balloon having less vertices than B, or a tree with maximum degree at least 3. Hence, the only possibility here is to choose V_λ in such a way that $B[V_{n-\lambda}]$ is a tree with maximum degree 3, but this is only possible when $B = B(1, 1, 1, 2, 3, ...)$. According to Observation 5.2, such a balloon is not arbitrarily partitionable and thus cannot be on-line arbitrarily partitionable, recall Theorem 2.19.

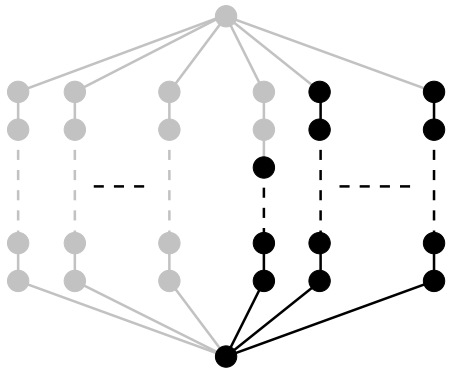


Figure 5.2: Situation described in the proof of Theorem 5.7. So that a balloon B (in black and grey) with a large number of branches can be on-line λ-partitioned, the connected part (in grey only) has to cover one root of B, as well as all vertices of all but at most three branches so that the remaining part (in black only) is potentially an on-line arbitrarily partitionable tree.

- $\lambda = 5$: by the previous cases, we know that B has branches including 1, 2, 3 and 4 vertices. For the same reasons as above, the connected part V_λ must be chosen in such a way that $B[V_{n-\lambda}]$ is either a path or an on-line arbitrarily partitionable 3-pode. Hence, because B has at least six branches, V_λ must contain one root of B and all the vertices of at least three of its branches, see Figure 5.2. Observe that V_λ can only be chosen in this way when $k = 6$ and $B = B(1, 1, 1, 2, 3, 4)$. It follows that B has four branches of odd order, and thus that it is not arbitrarily partitionable according to Observation 5.2. So it cannot be on-line arbitrarily partitionable.

- $\lambda = 6$: we know that B has branches with 1, 2, 3, 4 and 5 vertices. Moreover, since $k \geq 6$, the balloon B has one additional branch of order b_i. If $b_i \leq 7$, then B is not arbitrarily partitionable by Lemma 5.3, and thus is not on-line arbitrarily partitionable. Hence, $b_i \geq 8$ but, again, we cannot exhibit a subset V_λ for which $B[V_{n-\lambda}]$ is on-line arbitrarily partitionable, for the same reasons as above. Hence B is not on-line arbitrarily partitionable.

We eventually get that B is isomorphic to $B(1, 2, 3, 4, 5, 6, ...)$ which is not arbitrarily partitionable following Lemma 5.3. It thus cannot be on-line arbitrarily partitionable, a contradiction. ■

5.4.2 Some families of 4- or 5-balloons

It is worth mentioning that the upper bound on the number of branches in an on-line arbitrarily partitionable balloon exhibited in Theorem 5.7 is not only theoretical since there exist on-line arbitrarily partitionable 4- or 5-balloons. We exhibit such in the following results.

Lemma 5.8. *For every $k \geq 1$, the partial 4-balloon $PB(1, 1, 2, \overline{k})$ is recursively arbitrarily partitionable.*

Proof. As a base case, note that the claim is true when $k = 1$, $k = 2$ and $k = 3$ since the corresponding partial balloons are spanned by $Cat(2, 5)$, $Cat(3, 5)$ and $Cat(4, 5)$, respectively. Suppose now that the claim holds for all values of k up to an i, and now consider the partial balloon $B = PB(1, 1, 2, \overline{k})$ with $k = i + 1$ and order n. Recall that B is recursively arbitrarily partitionable if and only if, for every $\lambda \in \{1, 2, ..., \lfloor \frac{n}{2} \rfloor\}$, there is a bipartition of $V(B)$ inducing two recursively arbitrarily partitionable graphs on λ and $n - \lambda$ vertices, respectively, see Observation 5.1. For a given value of λ, we do not explicitly exhibit a bipartition of $V(B)$ but point out what subgraphs can be induced by a bipartition instead. Given these, the reader can then

easily deduce a correct bipartition. Between parenthesis, we exhibit an argument following which an obtained subgraph is recursively arbitrarily partitionable (if needed).

- $\lambda = 1$: P_1 and $PB(1, 2, \overline{k})$ (traceable).
- $\lambda = 2$: P_2 and $PB(1, 1, \overline{k})$ (traceable).
- $\lambda \in \{3, 4\}$: P_λ and $P_{n-\lambda}$.
- $\lambda = 5$: $Cat(2, 3)$ and P_{k+1}.
- $\lambda = 6$: $B(1, 1, 2)$ (traceable) and P_k.
- $\lambda \geq 7$: $PB(1, 1, 2, \overline{\lambda - 6})$ (induction hypothesis) and $P_{k-\lambda+6}$. ■

Lemma 5.9. *For every $k \geq 1$, the partial 4-balloon $PB(1, 2, 3, \overline{k})$ is recursively arbitrarily partitionable.*

Proof. The proof is by induction on k. Note first that $PB(1, 2, 3, \overline{1})$, $PB(1, 2, 3, \overline{2})$ and $PB(1, 2, 3, \overline{3})$ are recursively arbitrarily partitionable since they are spanned by the recursively arbitrarily partitionable caterpillars $Cat(2, 7)$, $Cat(3, 7)$ and $Cat(4, 7)$, respectively.

Let us now suppose that the lemma holds for all k up to a value of i, and consider the partial balloon $B = PB(1, 2, 3, \overline{k})$ with $k = i + 1$ and order n. For every possible value of $\lambda \in \{1, 2, ..., \lfloor \frac{n}{2} \rfloor\}$, as in the proof of Lemma 5.8 we exhibit possible recursive bipartitions of B.

- $\lambda = 1$: P_1 and $PB(2, 3, \overline{k})$ (traceable).
- $\lambda = 2$: P_2 and $PB(1, 3, \overline{k})$ (traceable).
- $\lambda = 3$: P_3 and $PB(1, 2, \overline{k})$ (traceable).
- $\lambda \in \{4, 5, 6\}$: P_λ and $P_{n-\lambda}$.
- $\lambda = 7$: $Cat(3, 4)$ and P_{k+1}.
- $\lambda = 8$: $B(1, 2, 3)$ (traceable) and P_k.
- $\lambda \geq 9$: $PB(1, 2, 3, \overline{\lambda - 8})$ (induction hypothesis) and $P_{n-\lambda+8}$. ■

According to Observation 2.27, Lemmas 5.8 and 5.9 directly imply that there exist infinitely many recursively arbitrarily partitionable 4-balloons (and, thus, infinitely many on-line arbitrarily partitionable 4-balloons, recall Theorem 2.19).

Corollary 5.10. *For every $k \geq 1$, the 4-balloons $B(1, 1, 2, k)$ and $B(1, 2, 3, k)$ are on-line and recursively arbitrarily partitionable.*

We now prove that there exist infinitely many recursively arbitrarily partitionable 5-balloons.

Theorem 5.11. *For every $k \geq 1$, the partial 5-balloon $PB(1, 1, 2, 3, \overline{2k})$ is recursively arbitrarily partitionable.*

Proof. Let $B = PB(1, 1, 2, 3, \overline{2k})$ with $k \geq 1$, and set $n = |V(B)|$. Recall that, according to Observation 5.1, the partial balloon B is recursively arbitrarily partitionable if and only if we can partition B into two recursively arbitrarily partitionable subgraphs with order λ and $n - \lambda$, respectively, for every $\lambda \in \{1, 2, ..., \lfloor \frac{n}{2} \rfloor\}$. One can obtain e.g. the following bipartitions for the first values of λ.

- $\lambda = 1$: P_1 and $B(1, 2, 3, \overline{2k})$ (Lemma 5.9).

- $\lambda = 2$: P_2 and $B(1,1,3,\overline{2k})$ (spanned by $Cat(2,5+2k)$).
- $\lambda = 3$: P_3 and $B(1,1,2,\overline{2k})$ (Lemma 5.8).
- $\lambda = 4$: P_4 and $Cat(4,2k+1)$.
- $\lambda = 5$: $Cat(2,3)$ and P_{2k+4}.
- $\lambda = 6$: P_6 and $Cat(2,2k+1)$.
- $\lambda = 7$: $Cat(3,4)$ and P_{2k+2}.

By now, it should be clear that the claim holds for every partial balloon $PB(1,1,2,3,\overline{2k})$ satisfying $n \leq 15$ (that is for every $k \in \{1,2,3\}$). Let us suppose, as an induction hypothesis, that the claim is true for every $k \leq i$, and consider the partition of $B = PB(1,1,2,3,\overline{2k})$, with $k = i+1$, into two recursively arbitrarily partitionable subgraphs for the remaining values of λ, that is for every $\lambda \in \{8,9,...,\lfloor \frac{n}{2} \rfloor\}$. Again, one can consider the following bipartitions.

- $\lambda \geq 8$ even: observe that $\lambda \leq 2k$ since $\lambda \leq \lfloor \frac{n}{2} \rfloor$ and we handled the cases where $k \leq 3$. We can thus partition B into P_λ and either $B(1,1,2,3)$ (when $k = 4$) or $PB(1,1,2,3,\overline{2k-\lambda})$ (when $k > 4$). These graphs are recursively arbitrarily partitionable according to Lemma 5.9 and by the induction hypothesis (since $2k - \lambda$ is even), respectively.
- $\lambda = 9$: $B(1,1,2,3)$ (spanned by $Cat(2,7)$) and P_{2k}.
- $\lambda > 9$ odd: $PB(1,1,2,3,\overline{\lambda-9})$ (induction hypothesis since $\lambda - 9$ is even) and $P_{n-\lambda+9}$. ■

Again, combining Observation 2.27 and Theorem 5.11 we get that the 5-balloon $B(1,1,2,3,2k)$ is recursively arbitrarily partitionable for every $k \geq 1$. Since every recursively arbitrarily partitionable graph is also on-line arbitrarily partitionable (recall Observation 5.1), we deduce the following.

Corollary 5.12. *For every $k \geq 1$, the 5-balloon $B(1,1,2,3,2k)$ is on-line and recursively arbitrarily partitionable.*

5.4.3 Order of the smallest branches

As mentioned above, every balloon with at most three branches is traceable and hence on-line arbitrarily partitionable (no matter how many vertices these branches contain). So that we step towards an on-line analogue of Theorem 2.23, we now focus on the order of the branches in an on-line arbitrarily partitionable 4- or 5-balloon. Our main result reads as follows.

Theorem 5.13. *Let $b_1, b_2, ..., b_k \geq 1$ be $k \in \{4,5\}$ positive integers satisfying $b_1 \leq b_2 \leq ... \leq b_k$. If $B(b_1, b_2, ..., b_k)$ is on-line arbitrarily partitionable, then $b_1 \leq 11$.*

The proof of Theorem 5.13 below is as follows. We consider any 4- or 5-balloon B whose all branches have at least 12 vertices, and show that B cannot be on-line λ-partitioned for some $\lambda \in \{1,2,...,|V(B)|-1\}$. For this purpose, we first exhibit a family $\mathcal{I}$ of partial balloons which are not on-line arbitrarily partitionable. We then show that when on-line λ-partitioning B, the on-line part necessarily induces a graph of $\mathcal{I}$ or another graph which is not on-line arbitrarily partitionable (e.g. a disconnected graph or a caterpillar which does not appear in the classification from Theorem 2.20). The partial balloons of $\mathcal{I}$ are depicted in Figure 5.4.

We throughout make use of the following terminology.

Notation 5.14. For a given integer $x \geq 1$, by x^- (resp. x^+) we refer to any positive integer $y \leq x$ (resp. $y \geq x$).

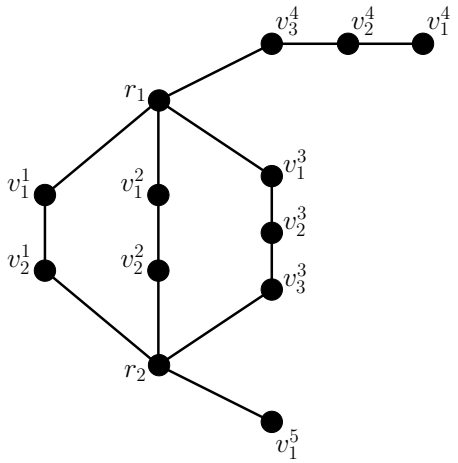


Figure 5.3: Illustration of the terminology introduced in Notation 5.15 on the partial 5-balloon $PB(2, 2, 3, \overline{3}, \underline{1})$.

Notation 5.15. We refer to the roots of every (possibly partial) k-balloon B as r_1 and r_2. Assuming the branches of B are ordered (e.g. following their length), we refer to $b_i(B)$ as the order of the ith branch of B for every $i \in \{1, 2, ..., k\}$. In case the ith branch of B is not hanging, then its vertices are denoted by $v_1^i, v_2^i, ..., v_{b_i(B)}^i$ in such a way that $v_1^i r_1, v_{b_i(B)}^i r_2 \in E(B)$ and $v_j^i v_{j+1}^i \in E(B)$ for every $j \in \{1, 2, ..., b_i(B) - 1\}$. Otherwise, i.e. the ith branch is hanging, its vertices are successively denoted by $v_1^i, v_2^i, ..., v_{b_i(B)}^i$ in such a way that v_1^i is the degree-1 vertex of the branch.

Refer to Figure 5.3 for an illustration of how the vertices of a partial balloon are denoted accordingly to Notation 5.15. We start by exhibiting, in Lemmas 5.16 to 5.23 below, the graphs constituting the above mentioned family $\mathcal{I}$.

Lemma 5.16. *For every $x, y \geq 1$, the partial 4-balloon $PB(12^+, 12^+, \overline{x}, \underline{y})$ is not on-line arbitrarily partitionable.*

Proof. For the sake of simplicity, we here and further denote n the order of every considered balloon (unless an ambiguity is possible). Besides, the parts of every on-line λ-partition are denoted V_λ (connected part) and $V_{n-\lambda}$ (on-line part). We prove the claim by induction on $x+y$. As a base case, consider $x = y = 1$ and the partial balloon $B = PB(12^+, 12^+, \overline{1}, \underline{1})$. Then B cannot be on-line 2-partitioned since every possible choice of V_2 inducing a connected subgraph makes $B[V_{n-2}]$ being either disconnected, a caterpillar $Cat(13^+, 13^+)$ or $Cat(11^+, 15^+)$, or a tree with two degree-3 nodes. Since none of these graphs is on-line arbitrarily partitionable, by definition B is not on-line arbitrarily partitionable.

To complete the base case, let us now suppose that $x + y = 3$ and denote by B the partial balloon $PB(12^+, 12^+, \overline{1}, \underline{2})$. As in the previous base case B is not on-line arbitrarily partitionable since it cannot be on-line 3-partitioned. In particular, for every correct choice of V_3, the graph $B[V_{n-3}]$ is not on-line arbitrarily partitionable because it is either disconnected, a non-caterpillar 3-pode different from $P_3(2, 4, 6)$, a caterpillar $Cat(10^+, 16^+)$ or $Cat(13^+, 13^+)$, or a tree with two degree-3 nodes.

Consider now that the claim holds for every partial balloon $PB(12^+, 12^+, \overline{x}, \underline{y})$ satisfying $x + y \leq k$ for some $k \geq 3$. We now prove that it is also true for every partial balloon $B = PB(12^+, 12^+, \overline{x}, \underline{y})$ satisfying $x + y = k + 1$. We distinguish the following two cases.

- $x > 1$ and $y > 1$: B is not on-line arbitrarily partitionable since it cannot be on-line 1-partitioned. Indeed, we have to consider either $V_1 = \{v_1^3\}$ or $V_1 = \{v_1^4\}$ since otherwise $B[V_{n-1}]$ would be either disconnected, or isomorphic to a large 3-pode or a tree with two

degree-3 nodes. But for each of these two choices of V_1, the remaining graph $B[V_{n-1}]$ is isomorphic to a partial balloon $PB(12^+, 12^+, \overline{x'}, \underline{y'})$ with $x' + y' = x + y - 1 = k$, which is not on-line arbitrarily partitionable by the induction hypothesis.

- $x = 1$ and $y > 2$: consider we want to on-line 2-partition B. For the same reasons as above, we have to consider $V_2 = \{v_1^4, v_2^4\}$. But then $B[V_{n-2}]$ is isomorphic to $PB(12^+, 12^+, \overline{x}, \underline{y-2})$ which is not on-line arbitrarily partitionable according to the induction hypothesis. Hence B is not on-line arbitrarily partitionable. These arguments hold analogously when $x > 2$ and $y = 1$. ■

Lemma 5.17. *For every $x \geq 1$ and $y \geq 10$, the partial 4-balloon $PB(12^+, 12^+, \overline{x}, \overline{y})$ is not on-line arbitrarily partitionable.*

Proof. Let us prove this claim by induction on $x + y$ as we did to prove Lemma 5.16. As a base case, let us consider the graph $B = PB(12^+, 12^+, \overline{1}, \overline{10})$. Then B is not on-line arbitrarily partitionable as it does not admit an on-line 11-partition. Indeed, every possible choice of V_{11} inducing a connected subgraph of B makes $B[V_{n-11}]$ being either disconnected, a caterpillar $Cat(11, 15^+)$, or a tree with maximum degree 4. For similar reasons, observe that neither $PB(12^+, 12^+, \overline{1}, \overline{11})$ nor $PB(12^+, 12^+, \overline{2}, \overline{10})$ are on-line arbitrarily partitionable since they do not admit an on-line 12- or 11-partition, respectively.

Let us now suppose that this lemma holds whenever $x + y \leq k$ for some $k \geq 12$, and consider a graph $B = PB(12^+, 12^+, \overline{x}, \overline{y})$ verifying $x + y = k + 1$. We claim that there exists a $\lambda \in \{1, 2, ..., n-1\}$ such that B does not admit an on-line λ-partition, and thus that B is not on-line arbitrarily partitionable:

- $x > 1$ and $y > 10$: under these conditions, there does not exist an on-line 1-partition of B. Indeed, every possible choice for V_1 which does not make $B[V_{n-1}]$ being disconnected makes this subgraph being isomorphic to either a non-caterpillar 3-pode different from $P_3(2, 4, 6)$, a tree with maximum degree 4, or a graph which is not on-line arbitrarily partitionable according to the induction hypothesis.

- $x = 1$ and $y > 11$: observe that there does not exist an on-line 2-partition of B, since every coherent choice for V_2 makes $B[V_{n-2}]$ being disconnected or isomorphic to either a caterpillar $Cat(13^+, 13^+)$, a tree with maximum degree 4, or a partial balloon which is not on-line arbitrarily partitionable by the induction hypothesis.

- $x > 2$ and $y = 10$: once again B does not admit an on-line 11-partition since every choice as V_{11} of eleven vertices inducing a connected subgraph of B makes $B[V_{n-11}]$ being either disconnected, a tree with maximum degree 4, a 3-pode which is not on-line arbitrarily partitionable, or a partial balloon which is not on-line arbitrarily partitionable according to the induction hypothesis. ■

Lemma 5.18. *For every $x, y, z \geq 1$, the partial 5-balloon $PB(12^+, 12^+, \overline{x}, \overline{y}, \underline{z})$ is not on-line arbitrarily partitionable.*

Proof. We prove the claim by induction on $x+y+z$. Let us first suppose that $x = y = z = 1$ and consider the associated graph $B = PB(12^+, 12^+, \overline{1}, \overline{1}, \underline{1})$. Once again B is not on-line arbitrarily partitionable since there does not exist an on-line 2-partition of B. Indeed, every possible choice for V_2 makes $B[V_{n-2}]$ being either disconnected, or isomorphic to either a tree with maximum degree 4 or a tree having two degree-3 nodes.

To complete the base case, observe that $PB(12^+, 12^+, \overline{1}, \overline{2}, \underline{1})$ and $PB(12^+, 12^+, \overline{1}, \overline{1}, \underline{2})$ are not on-line arbitrarily partitionable since they do not admit an on-line 3-partition: for every coherent choice of V_3, the subgraph $B[V_{n-3}]$ is disconnected, or isomorphic to either a tree with

maximum degree 4, a tree having two degree-3 nodes, or a non-caterpillar 3-pode different from $P_3(2,4,6)$.

Suppose now that the lemma is true whenever $x+y+z \leq k$ for some $k \geq 4$, and consider a balloon $B = PB(12^+, 12^+, \overline{x}, \overline{y}, \underline{z})$ where $x+y+z = k+1$. Once again, we consider two main cases:

- $z > 1$: in this case B is not on-line arbitrarily partitionable since it cannot be on-line 1-partitioned. Indeed, observe that removing one vertex from B makes the remaining subgraph being disconnected, isomorphic to a tree with maximum degree 4 or two degree-3 nodes, or isomorphic to a partial balloon which is not on-line arbitrarily partitionable according to the induction hypothesis or Lemma 5.16 (typically if $x = 1$ and we set $V_1 = \{v_1^3\}$).

- $z = 1$: once again B is not on-line arbitrarily partitionable under this condition since it cannot be on-line 2-partitioned: for every coherent choice as V_2, the remaining graph $B[V_{n-2}]$ is indeed either not connected, a tree with maximum degree 4 or two degree-3 nodes, or a partial balloon which is not on-line arbitrarily partitionable according to the induction hypothesis or previous Lemma 5.16. ■

Lemma 5.19. *For every $x, y \geq 1$, the partial 5-balloon $PB(12^+, 12^+, 12^+, \overline{x}, \underline{y})$ is not on-line arbitrarily partitionable.*

Proof. Once more, let us prove this claim by induction on $x+y$. Consider first that $x = y = 1$ and let $B = PB(12^+, 12^+, 12^+, \overline{1}, \underline{1})$. Then B is not on-line arbitrarily partitionable because it cannot be on-line 2-partitioned. Indeed, every possible choice for V_2 makes $B[V_{n-2}]$ being disconnected, isomorphic to a tree with maximum degre 4, to a partial balloon which is not on-line arbitrarily partitionable by Lemma 5.18 (typically if e.g. $V_2 = \{v_1^1, v_2^1\}$), or to a partial 6-balloon (e.g. if $V_2 = \{v_2^1, v_3^1\}$). In the latter case, such a graph cannot be on-line arbitrarily partitionable since otherwise we could raise a contradiction to Theorem 5.7 using Observation 2.27.

Additionally, observe that $B = PB(12^+, 12^+, 12^+, \overline{1}, \underline{2})$ cannot be on-line 3-partitioned: for every coherent choice for V_3, the subgraph $B[V_{n-3}]$ is not on-line arbitrarily partitionable for the same reasons as in the previous case. Hence B is not on-line arbitrarily partitionable.

We now suppose that the claim holds for every $x+y \leq k$ for some $k \geq 3$, and consider a partial balloon $B = PB(12^+, 12^+, 12^+, \overline{x}, \underline{y})$ verifying $x+y = k+1$. Let us take the following two cases in consideration to show that B is not on-line arbitrarily partitionable.

- $x > 1$ and $y > 1$: note that, in this situation, B cannot be on-line 1-partitioned. Indeed, for similar reasons as the ones we used for the base cases, we have to consider $V_1 = \{v_1^4\}$ or $V_1 = \{v_1^5\}$. But in both cases $B[V_{n-1}]$ cannot be on-line arbitrarily partitionable according to the induction hypothesis.

- $x = 1$ and $y > 2$: once again, observe that B cannot be on-line 2-partitioned. Indeed, observe that we must consider $V_2 = \{v_1^5, v_2^5\}$ since otherwise there would exist an on-line arbitrarily partitionable 6-balloon, an on-line arbitrarily partitionable tree having maximum degre 4, or a graph contradicting Lemma 5.18. But for this choice of V_2, we have $B[V_{n-2}] = PB(12^+, 12^+, 12^+, \overline{1}, \underline{y-2})$ which is not on-line arbitrarily partitionable according to the induction hypothesis. ■

Lemma 5.20. *For every $x \geq 1$, the partial 4-balloon $PB(12^+, 12^+, 12^+, \overline{x})$ is not on-line arbitrarily partitionable.*

Proof. Once again, this claim is proved by induction on x. Let us first suppose that $x = 1$ and let B be the partial balloon $PB(12^+, 12^+, 12^+, \overline{1})$. This time B is not on-line arbitrarily partitionable since it cannot be on-line 2-partitioned: for every possible choice of V_2, the remaining graph

$B[V_{n-2}]$ is not on-line arbitrarily partitionable since it is either disconnected, isomorphic to a tree with maximum degre 4, to a non-caterpillar 3-pode different from $P_3(2,4,6)$ or to a partial balloon which is not on-line arbitrarily partitionable by Lemma 5.16 (e.g. for $V_2 = \{v_1^1, v_2^1\}$), 5.17 (e.g. for $V_2 = \{v_{b_1(B)-1}^1, v_{b_1(B)}^1\}$) or 5.18 (e.g. for $V_2 = \{v_2^1, v_3^1\}$).

Let us now suppose that the claim holds for every $x \leq k$ for some $k \geq 1$. To complete the proof, observe that the partial balloon $B = PB(12^+, 12^+, 12^+, \overline{k+1})$ is not on-line arbitrarily partitionable since it cannot be on-line 1-partitioned. Indeed, for every choice of V_1, the subgraph $B[V_{n-1}]$ is not on-line arbitrarily partitionable according to the induction hypothesis, or because of one reason used to deal with the base case. ■

Lemma 5.21. *For every $x, y, z \geq 1$, the partial 5-balloon $PB(12^+, 12^+, \overline{x}, \overline{y}, \overline{z})$ is not on-line arbitrarily partitionable.*

Proof. We prove this claim by induction on $x + y + z$. First, let us suppose that $x = y = z = 1$ and consider the partial balloon $B = PB(12^+, 12^+, \overline{1}, \overline{1}, \overline{1})$. Note that B is not on-line arbitrarily partitionable since it cannot be on-line 2-partitioned. Indeed, every choice of V_2 for which $B[V_2]$ is connected implies that $B[V_{n-2}]$ is either disconnected or isomorphic to a tree with maximum degree at least 4. Analogously, observe that neither $PB(12^+, 12^+, \overline{1}, \overline{1}, \overline{2})$ nor $PB(12^+, 12^+, \overline{1}, \overline{2}, \overline{2})$ are on-line arbitrarily partitionable since they cannot be on-line 3-partitioned.

Suppose now that the claim holds by induction whenever $x + y + z \leq k$ for some $k \geq 5$, and consider a partial balloon $B = PB(12^+, 12^+, \overline{x}, \overline{y}, \overline{z})$ where $x + y + z = k + 1$. We distinguish the following two main cases depending on x, y and z:

- $x > 1$, $y > 1$ and $z > 1$: suppose we want to on-line 1-partition B. Then, we must consider $V_1 = \{v_1^3\}$, $V_1 = \{v_1^4\}$ or $V_1 = \{v_1^5\}$ since, for every other choice of V_1, the remaining graph $B[V_{n-1}]$ is either disconnected or isomorphic to a tree having maximum degree at least 4. But for every of these three choices for V_1, the subgraph $B[V_{n-1}]$ is not on-line arbitrarily partitionable by the induction hypothesis. Thus B is not on-line arbitrarily partitionable.

- $x = 1$: let $\alpha = \min(\{2,3,4\} \setminus \{y, z\})$. In this situation B does not admit an on-line α-partition. Indeed, for every coherent choice of V_α, the remaining graph $B[V_{n-\alpha}]$ is not on-line arbitrarily partitionable either according to the induction hypothesis, or because it is isomorphic to a non-connected graph or a tree with maximum degree at least 4. ■

Lemma 5.22. *For every $x, y \geq 1$, the partial 5-balloon $PB(12^+, 12^+, 12^+, \overline{x}, \overline{y})$ is not on-line arbitrarily partitionable.*

Proof. Once again, we prove this claim by induction on $x+y$. We first suppose that $x = y = 1$ and let $B = PB(12^+, 12^+, 12^+, \overline{1}, \overline{1})$. Similarly as in the proofs of the previous lemmas, B is not on-line arbitrarily partitionable because it cannot be on-line 2-partitioned. Indeed, for every possible choice of V_2 inducing a connected subgraph, the remaining graph $B[V_{n-2}]$ is either not connected, a tree with maximum degree 5, a partial balloon which is not on-line arbitrarily partitionable according to Lemma 5.18 (if e.g. $V_2 = \{v_1^1, v_2^1\}$) or 5.21 (if e.g. $V_2 = \{v_{b_1(B)-1}^1, v_{b_1(B)}^1\}$), or a partial 6-balloon. For the latter case, recall that a partial 6-balloon cannot be on-line arbitrarily partitionable since otherwise there would exist a 6-balloon contradicting Theorem 5.7. Similarly, observe that $PB(12^+, 12^+, 12^+, \overline{1}, \overline{2})$ is not on-line arbitrarily partitionable since it cannot be on-line 3-partitioned.

We finally suppose that the induction hypothesis is true whenever $x + y \leq k$ for some $k \geq 3$, and consider a partial balloon $B = PB(12^+, 12^+, 12^+, \overline{x}, \overline{y})$ satisfying $x + y = k + 1$. We distinguish two main cases, depending on the values of x and y, to prove that B is not on-line arbitrarily partitionable.

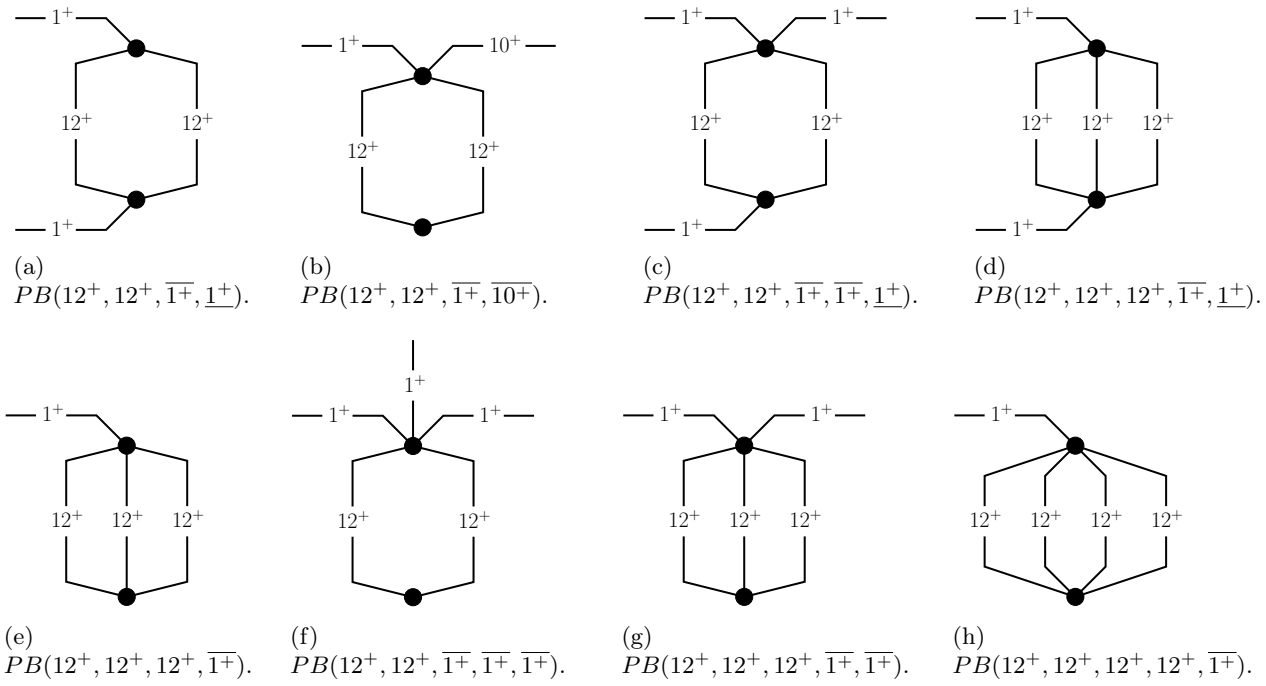


Figure 5.4: All not on-line arbitrarily partitionable partial balloons exhibited in Lemmas 5.16 to 5.23.

- $x > 1$ and $y > 1$: in this situation, B is not on-line arbitrarily partitionable since it cannot be on-line 1-partitioned. Indeed, for every choice of V_1, the remaining graph $B[V_{n-1}]$ is not on-line arbitrarily partitionable either for one of the reasons used for the base cases or according to the induction hypothesis.

- $x = 1$ and $y > 2$: the above arguments hold to prove that B cannot be on-line 2-partitioned. Thus, B is not on-line arbitrarily partitionable. ■

Lemma 5.23. *For every $x \geq 1$, the partial 5-balloon $PB(12^+, 12^+, 12^+, 12^+, \overline{x})$ is not on-line arbitrarily partitionable.*

Proof. Let us prove this claim by induction on x. We first suppose that $x = 1$ and consider the on-line 2-partitioning of $B = PB(12^+, 12^+, 12^+, 12^+, \overline{1})$. An on-line 2-partition does not exist, since, for every choice of V_2, the remaining graph $B[V_{n-2}]$ cannot be on-line arbitrarily partitionable: indeed, this subgraph is either not connected, a tree with maximum degree at least 4, a partial balloon which cannot be on-line arbitrarily partitionable according to Lemma 5.19 (if e.g. $V_2 = \{v_1^1, v_2^1\}$) or 5.22 (if e.g. $V_2 = \{v_{b_1(B)-1}^1, v_{b_1(B)}^1\}$), or a partial 6-balloon.

Suppose now that $PB(12^+, 12^+, 12^+, 12^+, \overline{x})$ is not on-line arbitrarily partitionable for every $x \leq k$ for some $k \geq 1$, and set $B = PB(12^+, 12^+, 12^+, 12^+, \overline{k+1})$. Once again B cannot be on-line arbitrarily partitionable since it cannot be on-line 1-partitioned. Indeed, for every possible choice of V_1, the graph $B[V_{n-1}]$ cannot be on-line arbitrarily partitionable either according to the induction hypothesis or because of one of the reasons used to handle the base case. ■

Using Lemmas 5.16 to 5.23, we can now prove Theorem 5.13.

Proof of Theorem 5.13. Let $B = B(12^+, 12^+, 12^+, 12^+)$ be a 4-balloon. Then B is not on-line arbitrarily partitionable since it cannot be on-line 1-partitioned. Indeed, for every choice of V_1, the graph $B[V_{n-1}]$ is not on-line arbitrarily partitionable since it is either a tree with maximum degre 4 or a partial balloon which is not on-line arbitrarily partitionable by Lemma 5.19 (e.g.

for $V_1 = \{v_2^1\}$) or 5.20 (e.g. for $V_1 = \{v_1^1\}$). It follows that an on-line arbitrarily partitionable 4-balloon must have a branch of order at most 11.

Now let $B = B(12^+, 12^+, 12^+, 12^+, 12^+)$ be a 5-balloon. Similarly as above, B is not on-line arbitrarily partitionable since it does not admit an on-line 1-partition. Indeed, every possible choice of V_1 makes $B[V_{n-1}]$ being either a tree with maximum degree 5, a partial balloon which is not on-line arbitrarily partitionable according to Lemma 5.23 (typically for $V_1 = \{v_1^1\}$), or a partial 6-balloon (e.g. for $V_1 = \{v_2^1\}$). In the latter case, observe that $B[V_{n-1}]$ cannot be on-line arbitrarily partitionable since otherwise there would exist, by Observation 2.27, an on-line arbitrarily partitionable 6-balloon contradicting Theorem 5.7. Hence, a 5-balloon cannot be on-line arbitrarily partitionable when its smallest branch has order at least 12. ■

Since every recursively arbitrarily partitionable graph is also on-line arbitrarily partitionable, recall Theorem 2.19, Theorem 5.13 directly implies that recursively arbitrarily partitionable 4- or 5-balloons have their smallest branch of order at most 11 too. However, using the fact that recursively arbitrarily partitionable caterpillars are generally smaller than on-line arbitrarily partitionable caterpillars, compare Tables 2.2.a and 2.2.b, one can easily improve Lemmas 5.16 to 5.23 above for recursively arbitrarily partitionable partial balloons to get a better upper bound on the order of the smallest branch in a recursively arbitrarily partitionable 4- or 5-balloon. The proof of this statement is omitted here[1] since it is very similar to the proof of Theorem 5.13.

Theorem 5.24. *Let $b_1, b_2, ..., b_k \geq 1$ be $k \in \{4, 5\}$ positive integers satisfying $b_1 \leq b_2 \leq ... \leq b_k$. If $B(b_1, b_2, ..., b_k)$ is recursively arbitrarily partitionable, then $b_1 \leq 7$.*

One can also get a similar constant upper bound on the order of the second smallest branch in a recursively arbitrarily partitionable 4- or 5-balloon, that is that $b_2 \leq 39$. The main argument in a proof of Theorem 5.24 is that partial 4-balloons of the form $PB(8^+, 8^+, \overline{x}, \underline{y})$ are generally not recursively arbitrarily partitionable. Because of this fact, plenty of other partial balloons cannot be recursively arbitrarily partitionable too, and the result follows. Such a statement is also true when considering that $b_1 \leq 7$ and $b_2 \geq 40$. Indeed, most of partial balloons $PB(7^-, 40^+, \overline{x}, \underline{y})$ cannot be recursively arbitrarily partitionable because they cannot be partitioned into two balanced recursively arbitrarily partitionable subgraphs. The main argument behind this statement is that, in general, one of the two subgraphs has to be isomorphic to a caterpillar $Cat(a, b)$ with a and b being greater than the values given by Theorem 2.20. Once again, the proof of the following claim is omitted here, but it can be obtained by deriving Lemmas 5.16 to 5.23 adequately.

Theorem 5.25. *Let $b_1, b_2, ..., b_k \geq 1$ be $k \in \{4, 5\}$ positive integers satisfying $b_1 \leq b_2 \leq ... \leq b_k$. If $B(b_1, b_2, ..., b_k)$ is recursively arbitrarily partitionable, then $b_2 \leq 39$.*

Something more can be deduced from the previous upper bounds. Observe that if $B = B(b_1, b_2, ..., b_k)$ is a k-balloon such that $k \geq 4$ and $b_1 \leq b_2 \leq ... \leq b_k$, then

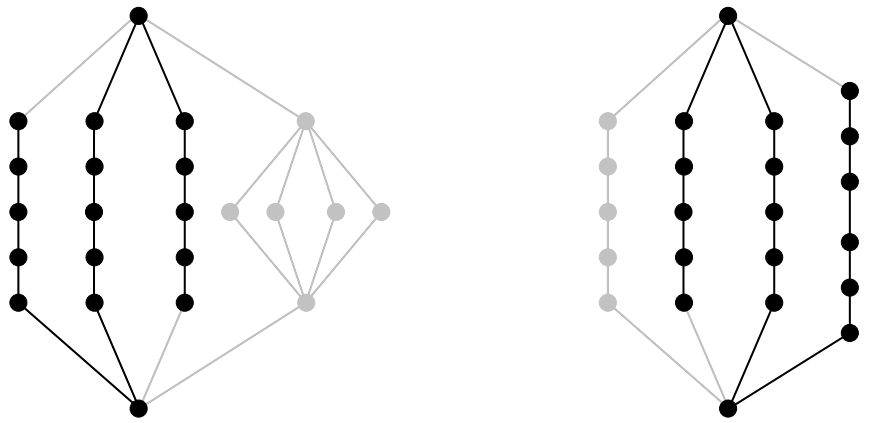
$$\varsigma(B) = b_k + b_{k-1} + b_{k-2} + 2.$$

From previous Theorems 5.13, 5.24 and 5.25, we then deduce the following lower bounds on the order of the longest paths in an on-line or recursively arbitrarily partitionable balloon.

Corollary 5.26. *Let B be a k-balloon.*

- *If B is on-line arbitrarily partitionable and $k = 4$, then $\varsigma(B) \geq |V(B)| - 11$.*
- *If B is recursively arbitrarily partitionable and $k = 4$, then $\varsigma(B) \geq |V(B)| - 7$.*
- *If B is recursively arbitrarily partitionable and $k = 5$, then $\varsigma(B) \geq |V(B)| - 46$.*

[1] A complete proof of Theorem 5.24 can be found in an early version of [15], which is available online at http://hal.archives-ouvertes.fr/docs/00/67/25/05/PDF/rap-balloons.pdf.



(a) $\varsigma(G) = 17$. (b) $\varsigma(B(5,5,5,6)) = 18$.

Figure 5.5: A 2-connected graph G and its underlying balloon $B(5,5,5,6)$ (in black and grey). The longest path of G (in black only) does not go through its biggest component, while the longest path of $B(5,5,5,6)$ (in black only) goes through its biggest branch.

5.4.4 Structural consequences on graphs with 2-cutsets

As mentioned in Section 2.3, properties of on-line or recursively arbitrarily partitionable balloons can be derived to properties of on-line or recursively arbitrarily partitionable graphs with 2-cutsets, respectively. In particular, the results we pointed out in Sections 5.4.1, 5.4.2 and 5.4.3 can be derived in the following way.

Corollary 5.27. *Let G be a graph with a 2-cutset $\{u,v\}$, and $b_1 \leq b_2 \leq ... \leq b_k$ be the orders of the $k \geq 2$ components of $G - \{u,v\}$. If G is on-line or recursively arbitrarily partitionable, then the following properties hold:*

- *$k \leq 5$,*
- *b_k can be arbitrarily large,*
- *if G is on-line arbitrarily partitionable and $k \in \{4,5\}$, then $b_1 \leq 11$,*
- *if G is recursively arbitrarily partitionable and $k \in \{4,5\}$, then $b_1 \leq 7$ and $b_2 \leq 39$.*

The first item of Corollary 5.27 follows directly from Theorem 5.7, the second item is derived from Corollaries 5.10 and 5.12 and the fact that 2- and 3-balloons are traceable, while the third and fourth items result from Theorems 5.13, 5.24 and 5.25. Since 2- or 3-balloons are always on-line and recursively arbitrarily partitionable because they are traceable, we can also deduce that if $G - \{u,v\}$ has only two or three components, then these components can all be arbitrarily large.

Note that Corollary 5.26 cannot be extended in a same vein since there is no direct relationship between the longest paths of a balloon $B = B(b_1, b_2, ..., b_k)$, with $b_1 \leq b_2 \leq ... \leq b_k$, and the longest paths of a graph G with a 2-cutset $\{u,v\}$ whose removal yields k components with orders $b_1, b_2, ..., b_k$, respectively. To be convinced of this statement, just imagine that the component with order b_k in $G - \{u,v\}$ is a $(b_k - 2)$-balloon $B(1,1,...,1)$ whose roots are connected to u and v in G. Depending on the structure of G, the longest path of G may not pass through its component with order b_k. In comparison, there is always one longest path of B going through its kth branch (see Figure 5.5).

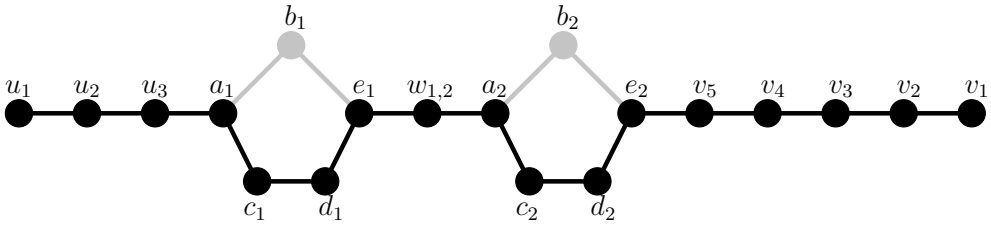


Figure 5.6: The connected-cycles graph $CC_2(3,5)$ (in black and grey), and its longest path (in black only).

5.5 On the order of the longest paths in a recursively arbitrarily partitionable graph

We now focus on the order of the longest paths in recursively arbitrarily partitionable graphs. Previous works on the topic could suggest that these graphs are "almost" traceable. This intuition is notably supported by the following results and facts:

- Corollary 5.27,
- we have $\varsigma(T) \geq |V(T)| - 2$ for every recursively arbitrarily partitionable tree T, recall Theorem 2.20,
- we have $\varsigma(S) \geq |V(S)| - 3$ for every recursively arbitrarily partitionable sun S, according to the characterization of Baudon, Gilbert and Woźniak [23],
- it was empirically observed[2] that we have $\varsigma(B) \geq |V(B)| - 4$ for every recursively arbitrarily partitionable 4- or 5-balloon B with relatively small order.

The main purpose of this section is to show that recursively arbitrarily partitionable graphs can be arbitrarily not traceable, i.e. $\varsigma(G)$ can be arbitrarily smaller than $|V(G)|$. For this purpose, we first prove, in Section 5.5.1, that there is no positive absolute constant $c \geq 1$ such that every recursively arbitrarily partitionable graph G verifies $\varsigma(G) \geq |V(G)| - c$. We then propose, in Section 5.5.2, an upper bound on the maximum value c', with $0 < c' < 1$, for which we have $\varsigma(G) \geq c' \cdot |V(G)|$ for every recursively arbitrarily partitionable graph G. These two results are obtained by introducing two new classes of recursively arbitrarily partitionable graphs.

5.5.1 Additive factor

The main result of this section reads as follows.

Theorem 5.28. *There does not exist a positive constant $c \geq 1$ such that we have $\varsigma(G) \geq |V(G)| - c$ for every recursively arbitrarily partitionable graph G.*

Theorem 5.28 is proved by exhibiting a counterexample for every possible value of c. For this purpose, we introduce the family of *connected-cycles graphs*.

Construction 5.29. Let $k \geq 1$ and $x, y \geq 0$ be three positive integers. The *connected-cycles graph* $CC_k(x,y)$ is the graph with the following vertices:

- let $u_1u_2...u_x$ and $v_1v_2...v_y$ be paths with order x and y, respectively;

[2]Using the proof scheme described in Section 5.4.3, Petit implemented a program for computing recursively arbitrarily partitionable 4- or 5-balloons. His program is available online at http://sourceforge.net/projects/rapbalchecker/, while his conclusions are gathered online at http://www.labri.fr/perso/jbensmai/students/enguerrand-petit.pdf (in French).

- for every $i \in \{1, 2, ..., k\}$, let $a_i b_i e_i d_i c_i a_i$ be a cycle with length 5;
- for every $i \in \{1, 2, ..., k-1\}$, let $w_{i,i+1}$ be a vertex.

These components are linked in $CC_k(x, y)$ in the following way: $u_x a_1, v_y e_k \in E(CC_k(x, y))$ and we have $w_{i,i+1}e_i, w_{i,i+1}a_{i+1} \in E(CC_k(x, y))$ for every $i \in \{1, 2, ..., k-1\}$.

Example 5.30. The connected-cycles graph $CC_2(3, 5)$ is depicted in Figure 5.6.

We throughout use the terminology introduced in Construction 5.29 to deal with the vertices of every connected-cycles graph we consider. Note that

$$\varsigma(CC_k(1, 1)) = |V(CC_k(1, 1))| - k$$

for every $k \geq 1$, see Figure 5.6. So we just have to prove that every graph $CC_k(1, 1)$ is recursively arbitrarily partitionable to prove Theorem 5.28. Before showing this, we first introduce another graph structure we encounter while partitioning a connected-cycles graph.

Construction 5.31. Let $k \geq 1$ and $x \geq 0$ be two positive integers. The *partial connected-cycles graph* $PCC_k(x)$ is the graph obtained by removing the vertex e_k from $CC_k(x, 0)$.

We now characterize recursively arbitrarily partitionable (partial) connected-cycles graphs.

Lemma 5.32. *Let $k, x, y \geq 1$ be three positive integers. The graph $PCC_k(x)$ is recursively arbitrarily partitionable if and only if $x \not\equiv 2 \pmod 3$. The graph $CC_k(x, y)$ is recursively arbitrarily partitionable if and only if $x \not\equiv 2 \pmod 3$ or $y \not\equiv 2 \pmod 3$.*

Proof. The necessity follows from the fact that every graph $PCC_k(x)$ with $x \equiv 2 \pmod 3$ or $CC_k(x, y)$ with $x, y \equiv 2 \pmod 3$ has order congruent to 0 modulo 3 but does not admit a (recursive) realization of the sequence $(3, 3, ..., 3)$.

The sufficiency is proved by induction on k. For each value of $k \geq 1$, we prove that the result is true for all possible values of x and (possibly) y which satisfy the claim. For the sake of clarity, we throughout denote by n the order of every graph considered during the proof.

Case 1. $k = 1$.

As a base case, note that every graph $PCC_1(x)$ is recursively arbitrarily partitionable since it is spanned by $Cat(3, x + 1)$, which is recursively arbitrarily partitionable according to the assumption on x (recall Theorem 2.20).

We now prove that every graph $C = CC_1(x, y)$ is recursively arbitrarily partitionable whenever the conditions of the claim are met. This is proved by induction on $x + y$ by showing that there is a partition of $V(C)$ into two parts V_λ and $V_{n-\lambda}$ satisfying the conditions of Observation 5.1 for every $\lambda \in \{1, 2, ..., \lfloor \frac{n}{2} \rfloor\}$. For each such value of λ, we exhibit a satisfying subset V_λ, and it is understood that $V_{n-\lambda} = V(C) \setminus V_\lambda$. We further assume $x \not\equiv 2 \pmod 3$ since the graphs $CC_1(x, y)$ and $CC_1(y, x)$ are isomorphic.

First, when dealing with $\lambda \geq x + 5$, we can pick up, as V_λ, the λ first vertices of the ordering

$$(u_1, u_2, ..., u_x, a_1, b_1, c_1, d_1, e_1, v_y, v_{y-1}, ..., v_1)$$

of $V(C)$ to get a partition of C into a traceable graph or $CC_1(x, \lambda - (x + 5))$ which is recursively arbitrarily partitionable by the induction hypothesis, and a path. For $\lambda = x$, one can consider $V_\lambda = \{u_1, u_2, ..., u_x\}$ so that the two induced graphs are traceable. Now, if $\lambda = x + 2$ or $\lambda = x + 3$, then we can choose $\{u_1, u_2, ..., u_x, a_1, b_1\}$ or $\{u_1, u_2, ..., u_x, a_1, c_1, d_1\}$, respectively, as V_λ, so that the two induced subgraphs are paths. Next, consider $\lambda = x + 4$. Then $V_\lambda = \{u_1, u_2, ..., u_x, a_1, b_1, c_1, d_1\}$ yields a correct partition of C. Indeed, on the one hand, $C[V_\lambda]$ is a caterpillar $Cat(3, x + 1)$ which is recursively arbitrarily partitionable since otherwise it would mean that $x \equiv 2 \pmod 3$, a contradiction. On the other hand, the graph $C[V_{n-\lambda}]$ is a path.

Now consider $\lambda = x+1$. If $V_\lambda = \{u_1, u_2, ..., u_x, a_1\}$ does not provide a satisfying partition of C, then $y \equiv 2 \pmod 3$ since $C[V_{n-\lambda}]$ is $Cat(3, y+1)$ and is not recursively arbitrarily partitionable. Consider then, as V_λ, the λ first vertices of the ordering

$$(v_1, v_2, ..., v_y, e_1, b_1, d_1, c_1, a_1, u_x, u_{x-1}, ..., u_1)$$

of $V(C)$. If this choice of V_λ does not yield a correct partition of C once again, then it means that either $C[V_\lambda]$ is the caterpillar $Cat(3, y+1)$, or a connected-cycles graph $CC_1(x', y)$ with $x' \equiv 2 \pmod 3$. But then we get that either $x + 1 = y + 4$ or $x + 1 = y + 5 + x'$, respectively, which both imply that $x \equiv 2 \pmod 3$, a contradiction.

Finally consider every value $\lambda \in \{1, 2, ..., x-1\}$. On the one hand, if $x - \lambda \not\equiv 2 \pmod 3$, then choose $V_\lambda = \{u_1, u_2, ..., u_\lambda\}$ so that $C[V_\lambda]$ and $C[V_{n-\lambda}]$ are a path and $CC_1(x - \lambda, y)$, which is recursively arbitrarily partitionable by the induction hypothesis. On the other hand, i.e. $x - \lambda \equiv 2 \pmod 3$, we have $\lambda \not\equiv 0 \pmod 3$ since otherwise we would have $x \equiv 2 \pmod 3$. We can assume that $\lambda \notin \{y, y+2, y+3\}$, since otherwise we could deduce a correct partition of C as in the cases $\lambda \in \{x, x+2, x+3\}$, respectively. Then consider, as V_λ, the λ first vertices of

$$(v_1, v_2, ..., v_y, e_1, b_1, d_1, c_1, a_1, u_x, u_{x-1}, ..., u_1).$$

If this choice of V_λ does not yield a correct partition of C, then $C[V_\lambda]$ is either a caterpillar $Cat(3, y+1)$ which is not recursively arbitrarily partitionable, or a graph $CC_1(x', y)$ with $x' \equiv 2 \pmod 3$. But note then that the first situation cannot occur since $\lambda \not\equiv 0 \pmod 3$. For the second situation, because $\lambda \not\equiv 0 \pmod 3$, we have $y \not\equiv 2 \pmod 3$. Since we have $x', y < x$, the graph $CC_1(y, x')$ is actually recursively arbitrarily partitionable by the induction hypothesis.

Case 2. *Arbitrary k.*

Let us now suppose that the result is true for every k up to an i, and now assume $k = i + 1$. Consider first $C = PCC_k(x)$ for consecutive values of $x \not\equiv 2 \pmod 3$. As we did before, to prove that C is recursively arbitrarily partitionable we show that there exists a partition of $V(C)$ satisfying the conditions of Observation 5.1 for every possible value of λ. One may choose V_λ as follows.

- If $\lambda \equiv 1 \pmod 3$, then one may consider, as V_λ, the first λ vertices of the ordering

  $$(b_k, d_k, c_k, a_k, w_{k-1,k}, e_{k-1}, b_{k-1}, d_{k-1}, c_{k-1}, a_{k-1}, ..., w_{1,2}, e_1, b_1, d_1, c_1, a_1, u_x, u_{x-1}, ..., u_1)$$

  of $V(C)$. On the one hand, note that $C[V_\lambda]$ is either a path, or spanned by a recursively arbitrarily partitionable caterpillar or a partial connected-cycles graph $PCC_{k'}(x')$ with $k' \leq k-1$ and $x' \equiv 0 \pmod 3$, which is recursively arbitrarily partitionable by the induction hypothesis. On the other hand, observe that $C[V_{n-\lambda}]$ is either traceable, or spanned by a connected-cycles graph $CC_{k''}(x, y)$ for some $k'' \leq k-1$ and y, which is recursively arbitrarily partitionable according to the induction hypothesis since $x \not\equiv 2 \pmod 3$.

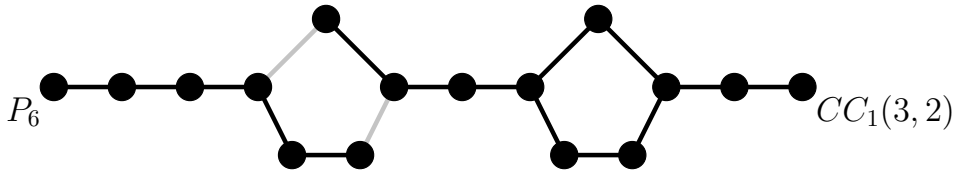
- If $\lambda \equiv 2 \pmod 3$, then one can obtain similar partitions of C from the ordering

  $$(d_k, c_k, b_k, a_k, w_{k-1,k}, e_{k-1}, d_{k-1}, c_{k-1}, b_{k-1}, a_{k-1}, ..., w_{1,2}, e_1, d_1, c_1, b_1, a_1, u_x, u_{x-1}, ..., u_1)$$

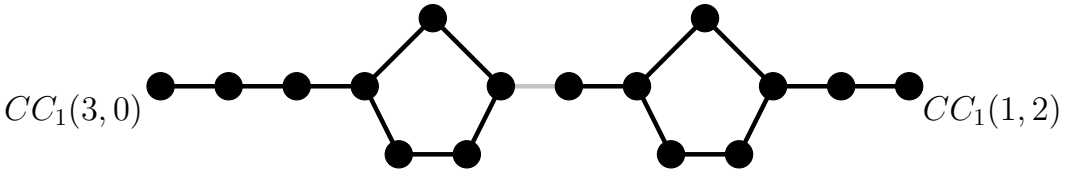
  of $V(C)$.

- Otherwise, if $\lambda \equiv 0 \pmod 3$, then one has to consider as V_λ the first λ vertices of the ordering

  $$(u_1, u_2, ..., u_x, a_1, b_1, c_1, d_1, e_1, w_{1,2}, ..., a_{k-1}, b_{k-1}, c_{k-1}, d_{k-1}, e_{k-1}, w_{k-1,k}, a_k, b_k, c_k, d_k)$$



(a) Case $\lambda \equiv 0 \pmod 3$.



(b) Case $\lambda \not\equiv 0 \pmod 3$ and $\lambda - (y+1) \equiv 2 \pmod 3$.

Figure 5.7: Two recursive bipartitions (in black only) of $CC_2(3,2)$ (in black and grey) obtained in the proof of Lemma 5.32.

of $V(C)$ when $x \equiv 1 \pmod 3$, or of the ordering

$$(u_1, u_2, ..., u_x, a_1, c_1, d_1, b_1, e_1, w_{1,2}, ..., a_{k-1}, c_{k-1}, d_{k-1}, b_{k-1}, e_{k-1}, w_{k-1,k}, a_k, c_k, d_k, b_k)$$

otherwise, i.e. when $x \equiv 0 \pmod 3$. The two induced subgraphs $C[V_\lambda]$ and $C[V_{n-\lambda}]$ are then recursively arbitrarily partitionable. Indeed, on the one hand, $C[V_\lambda]$ is either isomorphic to a path or spanned by a connected-cycles graph $CC_{k'}(x,y)$ for $k' \leq k-1$ and a y. On the other hand, the subgraph $C[V_{n-\lambda}]$ is spanned by a $PCC_{k''}(x')$ graph with $k'' \leq k$ and $x' \not\equiv 2 \pmod 3$.

To end up proving the claim, we have to show that $CC_k(x,y)$ is recursively arbitrarily partitionable whenever $x \not\equiv 2 \pmod 3$ or $y \not\equiv 2 \pmod 3$. As for the base case, we show this by induction on $x+y$. Once again, we assume that $x \not\equiv 2 \pmod 3$ for a given graph $C = CC_k(x,y)$.

For a given $\lambda \in \{1, 2, ..., y\}$, one can consider $V_\lambda = \{v_1, v_2, ..., v_\lambda\}$ so that C is partitioned into a path and $CC_k(x, y-\lambda)$ which is recursively arbitrarily partitionable according to the induction hypothesis on $x+y$. When $\lambda = y+1$, one can choose $V_\lambda = \{v_1, v_2, ..., v_y, e_k\}$ so that C is partitioned into a path and $PCC_1(x)$, which is recursively arbitrarily partitionable by the induction hypothesis since $x \not\equiv 2 \pmod 3$. For other values of λ, one may choose V_λ as follows.

- If $\lambda \equiv 0 \pmod 3$, one can consider, as V_λ, the λ first vertices from the ordering

  $$(u_1, u_2, ..., u_x, a_1, b_1, c_1, d_1, e_1, w_{1,2}, ..., w_{k-1,k}, a_k, b_k, c_k, d_k, e_k, v_y, v_{y-1}, ..., v_1)$$

  of $V(C)$ when $x \equiv 1 \pmod 3$, from

  $$(u_1, u_2, ..., u_x, a_1, c_1, d_1, b_1, e_1, w_{1,2}, ..., w_{k-1,k}, a_k, c_k, d_k, b_k, e_k, v_y, v_{y-1}, ..., v_1)$$

  otherwise, i.e. when $x \equiv 0 \pmod 3$. The two induced subgraphs are then recursively arbitrarily partitionable since they are traceable or isomorphic to connected-cycles graphs which are recursively arbitrarily partitionable according to the induction hypotheses (each of the obtained connected-cycles graphs has its "left" hanging path being of order not congruent to 2 modulo 3), see Figure 5.7.a.

- If $\lambda \not\equiv 0 \pmod 3$ and $\lambda - (y+1) \equiv 0 \pmod 3$, then one can consider the λ first vertices of the ordering

  $$(v_1, v_2, ..., v_y, e_k, b_k, d_k, c_k, a_k, w_{k-1,k}, ..., e_1, b_1, d_1, c_1, a_1, u_x, u_{x-1}, ..., u_1)$$

of $V(C)$. For each such partition, we get, on the one hand, that $C[V_\lambda]$ is either a path, a recursively arbitrarily partitionable caterpillar, or a recursively arbitrarily partitionable (partial) connected-cycles graph. In particular, note that when $C[V_\lambda]$ is a caterpillar or a partial connected-cycles graph, then this graph is recursively arbitrarily partitionable since $y \not\equiv 2 \pmod 3$ because of the assumptions on λ. On the other hand, the graph $C[V_{n-\lambda}]$ is either a path, or a (partial) connected-cycles graph which is recursively arbitrarily partitionable by the induction hypothesis since $x \not\equiv 2 \pmod 3$.

- If $\lambda \not\equiv 0 \pmod 3$ and $\lambda - (y+1) \equiv 1 \pmod 3$, then one may pick up, as V_λ, the λ first vertices from the ordering given to deal with the previous case. This choice of V_λ makes, on the one hand, $C[V_\lambda]$ being spanned by either a path, or $CC_{k'}(x', y)$ where $k' \leq k-1$ and $x' \not\equiv 2 \pmod 3$ which is recursively arbitrarily partitionable by the induction hypothesis. On the other hand, $C[V_{n-\lambda}]$ is a path, or is spanned by a graph $CC_{k''}(x, y')$ for $k'' \leq k-1$ and a y' which is recursively arbitrarily partitionable, again by the induction hypothesis.

- Otherwise, if $\lambda \not\equiv 0 \pmod 3$ and $\lambda - (y+1) \equiv 2 \pmod 3$, then some similar partitions of C may be obtained from the ordering

$$(v_1, v_2, ..., v_y, e_k, d_k, c_k, b_k, a_k, w_{k-1,k}, ..., w_{1,2}, e_1, d_1, c_1, b_1, a_1, u_x, u_{x-1}, ..., u_1)$$

of $V(C)$, see Figure 5.7.b. ■

We finally deduce Theorem 5.28 as a corollary of Lemma 5.32.

Proof of Theorem 5.28. For every value $c \geq 1$ of the constant mentioned in Theorem 5.28, the graph $CC_{c+1}(1,1)$ is recursively arbitrarily partitionable according to Lemma 5.32 and satisfies

$$\varsigma(CC_{c+1}(1,1)) = |V(CC_{c+1}(1,1))| - (c+1).$$

So for every value of c, we have a recursively arbitrarily partitionable graph showing that c does not contradict the claim. ■

Note that every graph $CC_k(1,1) + \{u_1v_1\}$ is 2-connected, is still recursively arbitrarily partitionable according to Observation 2.27, and satisfies

$$\varsigma(CC_k(1,1) + \{u_1v_1\}) = \varsigma(CC_k(1,1)) + 1.$$

We thus get the following refinement of Theorem 5.28.

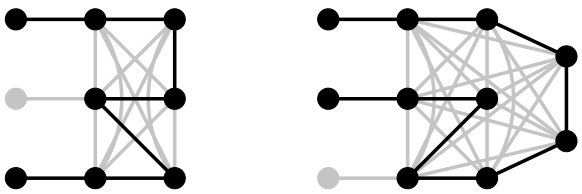
Observation 5.33. *Theorem 5.28 remains true when restricted to recursively arbitrarily partitionable 2-connected graphs.*

5.5.2 Multiplicative factor

Every connected-cycles graph $CC_k(1,1)$ has order $n = 6k+1$ and satisfies $\varsigma(CC_k(1,1)) = n - k$ for every $k \geq 1$. Thus, even if connected-cycles graphs confirm that the order of the longest paths in a recursively arbitrarily partitionable graph with order n is not constantly lower than n up to an additive factor, recall the proof of Theorem 5.28, they do not reject the strong relationship between the properties of being recursively arbitrarily partitionable and traceable. We now suggest to catch this relationship by involving a multiplicative factor.

Question 5.34. *What is the biggest $c' < 1$ such that we have $\varsigma(G) \geq c' \cdot |V(G)|$ for every recursively arbitrarily partitionable graph G?*

Regarding connected-cycles graphs, we get that $c' \leq \frac{5}{6}$. In this section, we deduce a better upper bound on c' by studying the following graph construction.



(a) $W(3,3)$. (b) $W(3,5)$.

Figure 5.8: Two urchins (in black and grey), and one of their longest paths (in black only).

Construction 5.35. Let $k, k' \geq 1$ be two positive integers. The *urchin* $W(k,k')$ is the graph obtained as follows.

- Let A, B, C be three sets of k, k and k' distinct vertices, respectively.
- Add a perfect matching between the vertices of A and B.
- Add all possible edges between distinct vertices in $B \cup C$.

Example 5.36. Figure 5.8 depicts two examples of urchins, namely $W(3,3)$ and $W(3,5)$.

Note that every urchin $W(k,k)$ has order $3k$ and satisfies $\varsigma(W(k,k)) = 2k+2$, see Figure 5.8. We then get that

$$\frac{\varsigma(W(k,k))}{|V(W(k,k))|}$$

tends to $\frac{2}{3}$ as k grows to infinity. In what follows, we show that every urchin $W(k,k)$ is recursively arbitrarily partitionable, and thus that the following holds regarding Question 5.34.

Theorem 5.37. *Regarding Question 5.34, we have* $c' \leq \frac{2}{3}$.

The following result actually provides a full characterization of recursively arbitrarily partitionable urchins.

Lemma 5.38. *Let* $k \geq 2$ *and* $k' \geq 0$ *be two positive integers. The urchin* $W(k,k')$ *is recursively arbitrarily partitionable if and only if* $k' \geq k-2$.

Proof. We introduce some terminology to deal with the vertices of every urchin $W(k,k')$. The partition $A \cup B \cup C$ of $V(W(k,k'))$ corresponds to the one given in Construction 5.35. The vertices of A are denoted $u_1, u_2, ..., u_k$, and those of B are denoted $v_1, v_2, ..., v_k$ in such a way that $u_i v_i$ is an edge for every $i \in \{1, 2, ..., k\}$. The vertices of C are denoted $w_1, w_2, ..., w_{k'}$ arbitrarily. Again, we herein denote by n the order of every urchin we consider.

The sufficient condition is proved by induction on both k and k'. As a base case, note that every urchin $W(2,k')$ is traceable, and thus recursively arbitrarily partitionable. Suppose now that $W(k,k')$ is recursively arbitrarily partitionable for every k up to an i and every $k' \geq k-2$. We now prove that every urchin graph $W = W(k,k')$ is recursively arbitrarily partitionable whenever $k = i+1$ and $k' \geq k-2$. For this purpose, we show, for every value of $\lambda \in \{1, 2, ..., \lfloor \frac{n}{2} \rfloor\}$, that $V(W)$ can be partitioned into two parts V_λ and $V_{n-\lambda}$ inducing recursively arbitrarily partitionable graphs on λ and $n-\lambda$ vertices, respectively.

We first deal with the easy cases, i.e. $\lambda \in \{1, 2, 3\}$. For $\lambda = 1$, consider $V_\lambda = \{u_1\}$ so that the two induced subgraphs are K_1 and $W(k-1, k'+1)$. Since $k' \geq k-2$, this second subgraph is recursively arbitrarily partitionable by the induction hypothesis. For $\lambda = 2$, let $V_\lambda = \{u_1, v_1\}$. The two induced subgraphs then are K_2 and $W(k-1,k')$, which is recursively arbitrarily partitionable for the same reason as in the previous case. Now, for $\lambda = 3$, choose $V_\lambda =$

$\{u_1, v_1, w_1\}$. The two induced subgraphs then are the path P_3, and the urchin $W(k-1, k'-1)$ which is recursively arbitrarily partitionable, again by the induction hypothesis.

We now deal with the remaining values of λ, i.e. $\lambda \geq 4$. The part V_λ is obtained by choosing two disjoint sets V'_λ and V''_λ, and then considering their union. On the one hand, in the case where $\lambda \equiv 1 \pmod 3$, let $x = \lfloor \frac{\lambda-4}{3} \rfloor$. Clearly x is an integer. First, let $V'_\lambda = \emptyset$ if $x = 0$, or $V'_\lambda = \bigcup_{i=1}^{x} \{u_i, v_i, w_i\}$ otherwise. Then set $V''_\lambda = \{v_{x+1}, u_{x+1}, v_{x+2}, u_{x+2}\}$. The two induced subgraphs then are a path or $W(x+2, x)$, and $W(k-(x+2), k'-x)$, which are recursively arbitrarily partitionable by the induction hypothesis since $k' \geq k-2$.

On the other hand, i.e. $\lambda \not\equiv 1 \pmod 3$, let $x = \lfloor \frac{\lambda}{3} \rfloor$ and $y = \lambda \pmod 3$ (then $y \in \{0, 2\}$). Then, let $V'_\lambda = \bigcup_{i=1}^{x} \{u_i, v_i, w_i\}$. The strategy for choosing V''_λ depends on whether $y = 0$ or $y = 2$.

- $y = 0$. Choose $V''_\lambda = \emptyset$. In this situation, the two induced subgraphs are $W(x, x)$ and $W(k-x, k'-x)$ which are recursively arbitrarily partitionable by the induction hypothesis since $k' \geq k-2$.
- $y = 2$. Let $V''_\lambda = \{v_{x+1}, u_{x+1}\}$. The two induced subgraphs then are $W(x+1, x)$ and $W(k-(x+1), k'-x)$, which are recursively arbitrarily partitionable according to the induction hypothesis. This concludes the proof of the sufficient condition.

The necessary condition follows from the fact that every urchin W not respecting the conditions of the claim cannot be partitioned into too many connected subgraphs with order 3. Indeed, as a set V_λ with size 3 inducing a recursively arbitrarily partitionable subgraph of W, one has to consider a part of the form $\{u_i, v_i, w_j\}$ or $\{w_i, w_j, w_\ell\}$. After having successively picked several subsets of three vertices from W, one necessarily gets an urchin $W(k', 0)$ with $k' \geq 3$. Such a graph is clearly not partitionable for $\lambda = 3$ once again. ■

Theorem 5.37 follows as a corollary of Lemma 5.38. It can further be strengthened as follows. Let $W = W(k, k')$ be a recursively arbitrarily partitionable urchin. Observe that by adding the edges $u_1u_2, u_1u_3, ..., u_1u_k$ to W (following the terminology of Lemma 5.38), we get a 2-connected graph W_2 which is recursively arbitrarily partitionable by Observation 2.27. By then adding the edges $u_2u_3, u_2u_4, ..., u_2u_k$ to W_2, we get another recursively arbitrarily partitionable graph W_3 which is 3-connected. By repeating this procedure as many times as wanted, we get a q-connected recursively arbitrarily partitionable graph W_q for every value of $q \geq 2$ assuming k and k' are big enough. Note further that we have

$$\varsigma(W_q) = \varsigma(W) + 2q,$$

and thus that $\frac{\varsigma(W_q)}{\varsigma(W)}$ tends to 1 as k grows to infinity. Therefore, the statement of Theorem 5.37 is also true when restricted to q-connected recursively arbitrarily partitionable graphs, no matter what is the value q.

Theorem 5.39. *Theorem 5.37 remains true when restricted to recursively arbitrarily partitionable graphs with arbitrarily large connectivity.*

5.6 Conclusion and open questions

Throughout this chapter, we have focused on structural properties of on-line or recursively arbitrarily partitionable graphs. Our main inspiration was Theorem 2.23, which we have considered in the context of on-line or recursively arbitrarily partitionable graphs. In Section 5.3 we have first showed that removing a cutset from a recursively arbitrarily partitionable graph cannot result in arbitrarily many components, recall Theorem 5.6. There is still a gap between what we

estimate to be the right maximum number of such components and the upper bound we proved, compare Conjecture 5.5 and Theorem 5.6, but Theorem 5.6 is already significant anyway since it again highlights the structural differences between recursively arbitrarily partitionable graphs and arbitrarily partitionable graphs. But in order for our result to be complete, it would be interesting to prove whether Conjecture 5.5 is true.

Showing that a structural property cannot occur in a recursively arbitrarily partitionable graph is obviously easier to show than in an on-line arbitrarily partitionable graph due to the fact that the recursive property is required for the two subgraphs induced by the vertex bipartition rather than for just one of these. So getting to a contradiction is easier when dealing with recursively arbitrarily partitionable graphs than it is for on-line arbitrarily partitionable graphs. But one direction for future works would be to wonder how many components can result from the removal of a k-cutset from an on-line arbitrarily partitionable graph. This question makes even more sense in view of the relationship between arbitrarily partitionable graphs, on-line arbitrarily partitionable graphs and recursively arbitrarily partitionable graphs, recall Theorem 2.19. Our intuition is that the number of resulting components should still be upper-bounded by a function of k independent from every graph parameter, but may be bigger than $2k + 1$.

Question 5.40. *What is the maximum number of components which may result when removing a k-cutset from an on-line arbitrarily partitionable graph?*

In Section 5.4, we have mainly focused on another aspect of Theorem 2.23 regarding on-line arbitrarily partitionable graphs, namely on the orders of the components resulting from the removal of a k-cutset. Pursuing works initiated by Baudon, Gilbert and Woźniak in [24], we have more precisely focused on the case $k = 2$ by studying the class of balloon graphs. As a main result, we have proved that removing a 2-cutset from an on-line or recursively arbitrarily partitionable graph results in small components whenever at least four such components are obtained. For the same reasons as above, we have obtained more results regarding recursively arbitrarily partitionable graphs. So again it would be judicious to study whether such results apply directly to on-line arbitrarily partitionable graphs (i.e. those which are not also recursively arbitrarily partitionable) as well. But for this purpose, it would be judicious to come up with a proof scheme different from the one we have been using throughout Section 5.4, which is based on nothing but a tedious case distinction.

We have then investigated, in Section 5.5, the order of the longest paths in recursively arbitrarily partitionable graphs, especially Question 5.34. Towards Question 5.34, our best result so far is Theorem 5.37. An important lack for knowing how much further we can push Theorem 5.37 is a general lower bound on the order of the longest paths of every recursively arbitrarily partitionable graph. So we address the following counterpart of Question 5.34.

Question 5.41. *What is the biggest $c'' < 1$ such that the order of the longest path of every recursively arbitrarily partitionable graph G cannot be smaller than $|V(G)| \cdot c''$?*

As mentioned earlier, it would be also interesting studying whether out results from Section 5.5 can be improved regarding on-line arbitrarily partitionable graphs. So considering our investigations from Section 5.5 in the context of on-line arbitrarily partitionable graphs could be a worthy direction for additional works on this topic.

It is worth mentioning that Conjecture 5.5, if true, could have another implication. As for the previously considered variants of the property of being arbitrarily partitionable, one can wonder whether an on-line or recursive version of Conjecture 3.46 (and more generally Conjecture 3.44) holds. We notably considered this question in [29], wherein we addressed the following.

Conjecture 5.42 ([29])**.** *Let $\ell \geq 1$ be a positive integer. If a graph G is recursively arbitrarily partitionable, then so is $G \,\square\, P_\ell$.*

In [29], we pointed out the hardness to infirm Conjecture 5.42 even for the case $\ell = 2$ due to the traceability of all Cartesian products $G \,\square\, P_2$ involving a known recursively arbitrarily partitionable graph G (this remains true for all new classes of recursively arbitrarily partitionable graphs we have exhibited throughout this chapter). Hence all such Cartesian products are directly recursively arbitrarily partitionable. So we implicitly raised the following.

Conjecture 5.43 ([29])**.** *For every recursively arbitrarily partitionable graph G, the Cartesian product $G \,\square\, P_2$ is traceable.*

An intriguing fact is that $C_{k,\ell}(K_{n_1+k}, K_{n_2+k}, ..., K_{n_\ell+k}) \,\square\, P_2$ is traceable whenever $\ell \leq 2k+1$, but not traceable otherwise. This, of course, does not imply the correctness of Conjecture 5.43 as a compound graph made up of complete components has very convenient local Hamiltonian properties. But it implies that if a graph G has a k-cutset whose removal results in at least $2k + 2$ components, then $G \,\square\, P_2$ cannot be traceable. So there is a strong relationship between the traceability of a graph and the fact that removing a cutset from it cannot result in too may components. Conjecture 5.5, if true, would then be a strong support to Conjecture 5.43.

In Section 5.2, we have also pointed out the membership of On-Line Arbitrarily Partitionable Graph and Recursively Arbitrarily Partitionable Graph to PSPACE. Though this result is evident, it would be interesting investigating whether these two problems are complete in PSPACE. Proving the PSPACE-completeness of these problems would be quite interesting since we still do not know the hardness status of Arbitrarily Partitionable Graph, recall our investigations from Chapter 3.

Question 5.44. *Are* On-Line Arbitrarily Partitionable Graph *and* Recursively Arbitrarily Partitionable Graph *PSPACE-complete?*

Chapter 6

Conclusion to Part I

Throughout Part I, we have considered several problems related to the problem of partitioning a graph into connected subgraphs. Not only we have considered the original problem, asking whether a graph can be partitioned into a certain number of connected subgraphs whose orders are specified, but also more elaborated problems asking whether a graph can be partitioned into arbitrarily many connected subgraphs with arbitrary orders (and possibly additional properties). Our studies have been made regarding mainly the algorithmic and structural aspects.

We have first confirmed, in Chapter 3 (Section 3.1), the algorithmic hardness of the original problem. Although some restrictions of Realizable Sequence were already known to be NP-complete, recall Theorem 2.10, we have exhibited new conditions under which Realizable Sequence remains NP-complete. These concern both the input sequence π and the input graph G. Our most meaningful results in this line are perhaps Theorems 3.10 and 3.11, which establish the hardness of Realizable Sequence when restricted to graphs which sometimes appeared to be convenient regarding other hard graph problems. We have also established the NP-completeness of Realizable Sequence when restricted to multipodes (Theorem 3.8) and balloons (Theorem 3.11), which, although simple, are among the most studied classes of graphs in the context of arbitrarily partitionable graphs.

Then, we have considered the notion of arbitrarily partitionable graphs in Chapter 3, especially regarding the algorithmic aspect. Although we did not manage to tell much about the complexity of Arbitrarily Partitionable Graph, we have exhibited new polynomial kernels of sequences for several families of graphs in Section 3.3. These would rather advocate the membership of Arbitrarily Partitionable Graph to NP, though nothing is certain as these kernels concern very restricted families of graphs.

We have finally considered two main variants of the notion of arbitrarily partitionable graphs. On the one hand, we have considered the notion of preassignable arbitrarily partitionable graphs in Chapter 4, which are basically arbitrarily partitionable graphs which can be partitioned even when a fixed number of vertex-membership constraints must be met. Our results regarding this notion are mainly structural. Namely, we have mainly proved a tight lower bound on the size of a k-preassignable arbitrarily partitionable graph with order n (regarding both k and n) in Section 4.3, and have showed in Section 4.4 that, though these graphs need to be dense enough, preassignable arbitrarily partitionable graphs can be arbitrarily far from being traceable.

On the other hand, we have also considered two recursive notions of arbitrarily partitionable graphs in Chapter 5, namely on-line and recursively arbitrarily partitionable graphs, which are, roughly set, arbitrarily partitionable graphs which can be partitioned into partitionable subgraphs. Our motivation was mainly to step towards a recursive analogue of structural Theorem 2.23 for these graphs. In this scope, we have confirmed that the structures of arbitrarily partitionable graphs and recursively arbitrarily partitionable graphs can be quite different, compare Theorems 5.6 and 5.13 to Theorem 2.23. We have notably expressed these differences in terms of some structural properties of the components resulting from the removal of a cutset in these graphs.

In the light of all remaining gaps to fill and open questions related to the notion of arbitrarily partitionable graphs and its variants, perspectives for future works are wide. Our investigations

in particular gave birth to several open questions, refer to concluding Sections 3.6, 4.6, and 5.6. But many other directions for future works have not been even mentioned in this thesis. Among the most appealing directions, let us mention the following ones.

Hypotraceable arbitrarily partitionable graphs.

Every traceable graph is obviously arbitrarily partitionable, recall Observation 2.34. Then one natural question is to wonder about sufficient conditions for a graph which is "nearly traceable" to be arbitrarily partitionable. In this scope, the case of the following class of graphs seems quite intriguing.

Definition 6.1. A graph G is *hypotraceable* if G is not traceable but $G - \{u\}$ is traceable for every $u \in V(G)$.

Only few works have been dedicated to hypotraceable graphs, regarding mainly extremal aspects related to these graphs, e.g. given a property P, what is the minimum order of an hypotraceable graph having property P (if such exist)? We refer the interested reader to works of Thomassen [116] and Araya and Wiener [10] for examples of such results.

For every hypotraceable graph G, we have $\varsigma(G) = |V(G)| - 1$ and G has a lot of different paths with order $\varsigma(G)$. Due to a theorem proved by Ravaux in [105], every $|V(G)|$-sequence π satisfying $|sp(\pi)| \geq 2$ is realizable in G. So in order to check that an hypotraceable graph is arbitrarily partitionable, we can narrow down our concern on sequences with spectrum of size 1.

This observation implies that every hypotraceable graph with prime order is arbitrarily partitionable. Now, for the general case, it does not seem obvious whether every non-trivial sequence $(k, k, ..., k)$ is always realizable in an hypotraceable graph whose order is a multiple of k. So we ask the following.

Question 6.2. *Is every non-trivial sequence $(k, k, ..., k)$ realizable in an hypotraceable graph whose order is a multiple of k?*

Assume we want to realize a non-trivial sequence $\pi = (k, k, ..., k)$ in an hypotraceable graph G. It is easily seen that if $P = v_1v_2...v_{|V(G)|-1}$ denotes one Hamiltonian path of $G-\{u\}$ for some $u \in V(G)$ and there is an edge $uv_i \in E(G)$ with $i \not\equiv 0 \pmod k$, then a realization of π in G can be deduced (just pick parts along P from left to right, and add u to the part containing v_i before continuing). So the hard case to consider is the case where, for every $u \in V(G)$, the connection between u and P is only made via some kth vertices of P, i.e. vertices among $\{v_k, v_{2k,}, v_{3k}, ...\}$. Actually u has to be joined to at least three such vertices since otherwise either G would not be hypotraceable, or we could deduce a favourable situation by removing a neighbour of u from G.

Although this question does not seem so complicated to handle (G has strong structural properties, and π is well identified), things are actually hard, due mainly to the following argument. The fact is that two Hamiltonian paths P and P' of $G - \{u\}$ and $G - \{u'\}$, where $u, u' \in V(G)$ and $u \neq u'$, can be quite different. Although the majority of their vertices are the same, the two orderings over these can differ a lot. So even if we cannot deduce a realization from P, the consequences on P' do not seem obvious in general.

Conversely, it has to be kept in mind that practically checking whether a non-trivial sequence $(k, k, ..., k)$ is realizable in an hypotraceable graph is tedious due to the fact that these graphs have relatively large order (for the context of arbitrarily partitionable graphs)[1]. So refuting Question 6.2 does not appear easier than proving it.

1-tough arbitrarily partitionable graphs.

According to Theorem 2.1, every 2-connected graph is arbitrarily 2-partitionable. So another natural question is to wonder whether every 2-connected graph can always be partitioned into three connected subgraphs, or, equivalently, whether Theorem 2.1 could be improved. This

[1]So far the smallest known hypotraceable graph has order 34 and was exhibited by Thomassen in [116].

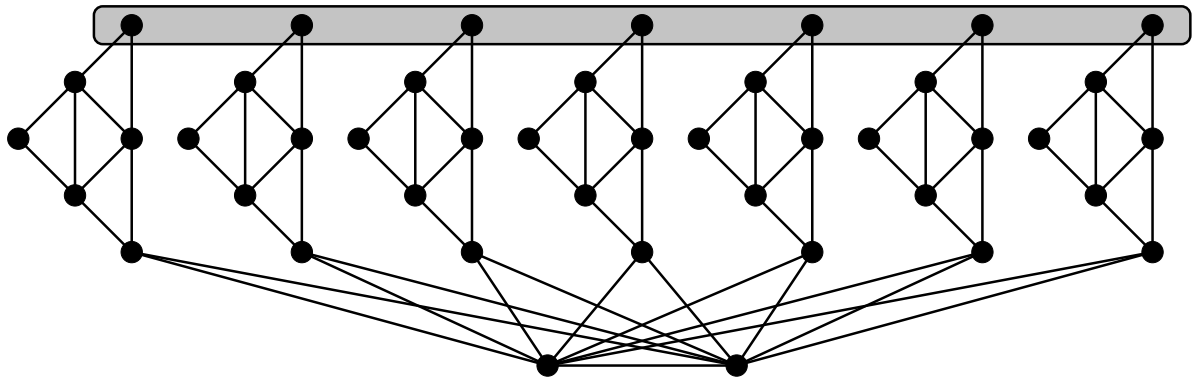


Figure 6.1: A 1-tough graph which does not admit any realization of $(11, 11, 11, 11)$. The vertices in the grey area form a clique.

question could of course be generalized to k-connected graphs for any fixed value of $k \geq 2$, but let us just focus on the following question for now.

Question 6.3. *Is every* 2*-connected graph arbitrarily* 3*-partitionable?*

Question 6.3 can be easily refuted as $M_2(2, 4)$ is 2-connected but does not admit any perfect matching (and hence no realization of $(2, 2, 2)$). This counterexample can be easily generalized, but all such counterexamples share a common property based on the following definition.

Definition 6.4. A graph G is 1*-tough* if, for every integer $k \geq 1$, removing a k-cutset from G results in at most k components.

Note that every 2-connected graph is 1-tough. However the contrary does not have to hold. In particular, all mentioned above graphs refuting Question 6.3 are not 1-tough. So we ask the following refined question.

Question 6.5. *Is every* 1*-tough graph arbitrarily* 3*-partitionable?*

Despite many efforts, we did not manage to confirm or reject Question 6.5. However, Question 6.5, if true, would be tight in the sense that 1-tough graphs do not have to be arbitrarily 4-partitionable. As an illustration of this statement, consider the graph depicted in Figure 6.1. The construction of this graph is inspired from a construction given by Bauer, Broersma and Veldman in [26] for constructing 1-tough graphs. It can be then checked by hand that this graph, which has order 44, does not admit any realization of $(11, 11, 11, 11)$. So we have the following.

Theorem 6.6. 1*-tough graphs are not all arbitrarily* 4*-partitionable.*

Arbitrarily partitionable graphs in terms of forbidden subgraphs.

Sufficient conditions for a graph to have Hamiltonian properties have been sometimes expressed in terms of forbidden subgraphs. The goal is then to exhibit graphs $H_1, H_2, ..., H_k$ such that every $\{H_1, H_2, ..., H_k\}$-free graph has a given Hamiltonian property. Perhaps the most famous such result is the one of Duffus, Gould and Jacobson stating that every connected claw- and net-free graph is traceable, and every 2-connected claw- and net-free graph is Hamiltonian [53]. So, as explained in introductory Section 2.3, it would be interesting exhibiting similar such sufficient conditions for a graph to be arbitrarily partitionable.

Since every traceable graph is arbitrarily partitionable, recall Observation 2.34, the result of Duffus, Gould and Jacobson directly ensures that every claw- and net-free graph is arbitrarily partitionable. So the next question is about whether we can refine these two forbidden patterns so that we get a result specific to arbitrarily partitionable graphs. This would formulate e.g. as follows.

Question 6.7. *Are there graphs $H_1, H_2, ..., H_k$ such that every $\{H_1, H_2, ..., H_k\}$-free graph is necessarily arbitrarily partitionable but not necessarily traceable (or Hamiltonian)?*

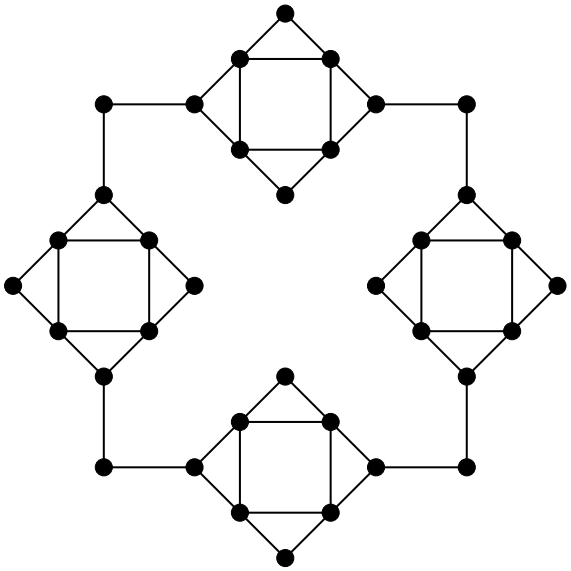


Figure 6.2: A 2-connected claw-free graph which is not arbitrarily partitionable.

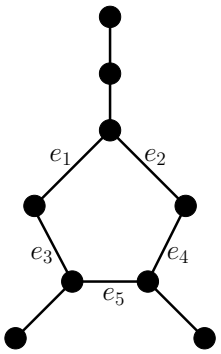


Figure 6.3: A minimal recursively arbitrarily partitionable graph which is not minimal arbitrarily partitionable.

Question 6.7 does not seem easy to handle. However, it is worth mentioning that the claw and the net both seem to be important patterns towards such a sufficient condition for a graph to be arbitrarily partitionable. Note indeed that e.g. every balloon $B(5, 5, ..., 5)$ with at least four odd branches is 2-connected and net-free but not arbitrarily partitionable, recall Observation 5.2. Regarding the claw, note that the graph depicted in Figure 6.2 is 2-connected and claw-free, but does not admit any realization of $(4, 4, ..., 4)$.

Minimal recursively arbitrarily partitionable graphs.

Similarly as for arbitrarily partitionable graphs, recall our investigations from Section 3.4, it would be interesting studying what does a minimal recursively arbitrarily partitionable graph look like, and whether minimal recursively arbitrarily partitionable graphs are quite different from minimal arbitrarily partitionable graphs. One reason to think that these two families of graphs could be different is notably the fact that recursively arbitrarily partitionable trees have a much more restricted structure than arbitrarily partitionable trees, compare Theorems 2.17 and 2.20.

To illustrate this statement, consider the graph G drawn in Figure 6.3. It can be checked that G is recursively arbitrarily partitionable (and is hence arbitrarily partitionable, recall Theorem 2.19). Besides, this graph G is minimal recursively arbitrarily partitionable because the graphs $G - \{e_1\}$ and $G - \{e_3\}$ are trees with two degree-3 nodes, and $G - \{e_5\}$ is a tripode different from $P_3(2, 4, 6)$. However G is not minimal arbitrarily partitionable as $G - \{e_1\}$ is

arbitrarily partitionable. So there are minimal recursively arbitrarily partitionable graphs with order 9, while the smallest minimal arbitrarily partitionable graphs we know have order 10, recall our investigations from Section 3.4.1. Although this difference is not meaningful, we believe more interesting differences could be pointed out between these graphs, about which still little is known.

Part II

Distinguishing the neighbours of a graph via an edge-weighting

Chapter 7
Introduction to Part II

This chapter is dedicated to the introduction of all materials necessary to understand our works related to vertex-distinguishing edge-colourings of graphs in Chapters 8, 9, 10 and 11. Notions motivating our investigations are presented in Section 7.1. Definitions, notation and terminology we use throughout are then given in Section 7.2. We then survey additional notions related to our investigations in Section 7.3. We finally describe concretely our contributions in Section 7.4.

7.1 Motivations

Vertex-distinguishing edge-weightings of graphs emerged with several works aiming at defining what an *irregular graph* could be. If we want such a notion of irregularity to be an antonym to regularity rather than just the property of being not regular, then perhaps the most natural definition for it is the one of *total irregularity* defined as follows.

Definition 7.1. A graph G is *totally irregular* if every two vertices of G have distinct degrees.

However, the notion of total irregularity is not suitable for simple graphs as an easy argument shows that every simple graph with at least two vertices necessarily has two vertices with the same degree. Assume indeed that G is a simple graph with order n. If G is totally irregular, then the vertices of G have degree $0, 1, ..., n-1$, respectively. But then the existence of a vertex with degree $n-1$ in G contradicts the existence of a vertex with degree 0.

This argument does not hold in the context of multigraphs as one can easily design totally irregular multigraphs. Based on that fact, Chartrand, Jacobson, Lehel, Oellermann, Ruiz and Saba studied how to turn a simple graph G, which cannot be totally irregular, into a totally irregular multigraph G' by multiplying every edge e of G a given number $n_e \geq 1$ of times [42], i.e. replacing e with n_e parallel edges. The main reason for just multiplying the edges of G (instead of adding new vertices to G or edges joining two non-adjacent vertices of G) to get G' is that this transformation does not alter the adjacencies, in the sense that every two adjacent vertices in G' are also adjacent in G. So the structure of G' is representative of the structure of G. The main concern of Chartrand, Jacobson, Lehel, Oellermann, Ruiz and Saba was in particular to perform this transformation in an optimal way, namely in such a way that the quantity $\max\{n_e : e \in E(G)\}$ is minimized. This optimization problem can be formalized with the notions of *weighted degree* of a vertex and *vertex-sum-distinguishing edge-weighting* of a graph.

Definition 7.2. Let w be an improper k-edge-weighting of a graph G. The *weighted degree* of a vertex v of G is defined as

$$s_w(v) = \sum_{u \in N(v)} w(vu).$$

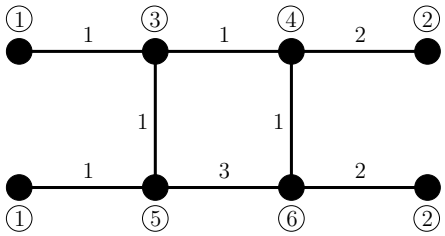


Figure 7.1: A graph and a 3-edge-weighting of it. Weighted degrees are circled.

We say that w is *vertex-sum-distinguishing* if s_w is injective.

Example 7.3. The 3-edge-weighting w of the graph depicted in Figure 7.1 is not vertex-sum-distinguishing since two vertices have weighted degree 1, and two vertices have weighted degree 2 by w.

If w is vertex-sum-distinguishing then, by multiplying every edge e of G exactly $w(e)$ times, we obtain a multigraph G' which is totally irregular since the weighted degree $s_w(v)$ by w of every vertex v of G directly converts to the degree of v in G'. Regarding the motivation above, we are hence interested in finding the least number of weights used by a vertex-sum-distinguishing k-edge-weighting of a graph. We call this parameter the *vertex-sum-distinguishing chromatic index* throughout this thesis for the sake of consistency, though this parameter is rather known under the name of *irregularity strength* in the literature.

Definition 7.4. The *vertex-sum-distinguishing chromatic index* of a graph G is the minimum number of weights of a vertex-sum-distinguishing k-edge-weighting of G (if any).

Irregularity strength of graphs has been receiving great interest in the last decades, and gave birth to dozens variants obtained by changing the parameters constituting the underlying problem. Since our goal here is not to give a complete survey of the investigations on the irregularity strength of graphs, but rather to explain what motivated our own investigations, we refer the interested reader to surveys of e.g. Gallian [61], Lehel [86], West [119] and Seamone [109] for more details on this topic.

In particular, one variant of the notion of irregularity strength strongly related to our investigations is motivated as follows. Since the notion of totally irregularity is not suitable for simple graphs, one can consider the following weaker *local* notion of irregularity, which was first introduced under the name of *high irregularity* by Alavi, Chartrand, Chung, Erdős, Graham and Oellermann [5].

Definition 7.5. A graph G is *locally irregular* if every two adjacent vertices of G have distinct degrees.

If a graph G is not locally irregular, then, as above, one can turn G into a locally irregular multigraph by multiplying every edge of G a certain number of times. This can again be regarded as an edge-weighting problem.

Definition 7.6. Let w be an improper k-edge-weighting of a graph G. We say that w is *neighbour-sum-distinguishing* if s_w is proper.

Example 7.7. The 3-edge-weighting drawn in Figure 7.1 is neighbour-sum-distinguishing.

Similarly as above, if w is a neighbour-sum-distinguishing edge-weighting of G then, by multiplying every edge e of G exactly $w(e)$ times, we get a locally irregular multigraph whose structure is representative of the structure of G. So, again, we are interested in finding the least number of weights in a neighbour-sum-distinguishing k-edge-weighting of a graph. This parameter is the *neighbour-sum-distinguishing chromatic index*.

Definition 7.8. The *neighbour-sum-distinguishing chromatic index* $\chi'_{nsd}(G)$ of a graph G is the minimum number of weights of a neighbour-sum-distinguishing k-edge-weighting of G (if any).

Neighbour-sum-distinguishing edge-weighting of graphs in turn motivated numerous variants obtained by modifying (parts of) the definitions. Our investigations throughout Part II are dedicated to known variants of the notions of vertex-sum-distinguishing and neighbour-sum-distinguishing edge-weighting of graphs. Inspired by these variants, we also introduce new ways for distinguishing the vertices of a graph by means of an edge-weighting.

7.2 Definitions, terminology and notation

Every problem asking whether some vertices of a graph G can be distinguished by means of an improper weighting w of the edges of G can be described by formally defining two parameters:

Parameter 1: which vertices of G must be distinguished,

Parameter 2: according to which parameter related to w two vertices of G to distinguish are considered distinguished.

Example 7.9. If we want to distinguish all vertices of G (Parameter 1) via the sums of their incident weights by w (Parameter 2), then we want w to be vertex-sum-distinguishing.

Example 7.10. If we want to distinguish the adjacent vertices of G (Parameter 1) via the sums of their incident weights by w (Parameter 2), then we want w to be neighbour-sum-distinguishing.

To clarify Parameters 1 and 2 when dealing with a vertex-distinguishing edge-weighting w of G, and to keep consistency with the terminology and notation introduced in Section 7.1, we say that w is *d-f-distinguishing*, where $d \in \mathbb{N}^*$ and f is a function associating an object deduced from w with every vertex of G, if, for every two vertices u and v of G such that $\text{dist}(u, v) \leq d$, we have $f(u) \neq f(v)$. We rather refer to an ∞-f-distinguishing edge-weighting as a *vertex-f-distinguishing edge-weighting*. Similarly, we refer to a 1-f-distinguishing edge-weighting as a *neighbour-f-distinguishing edge-weighting*. For the sake of simplicity, we rather use a term describing f rather than f itself when the elements associated by f with the vertices of G are clear from the context. Note that the notions of vertex-sum-distinguishing and neighbour-sum-distinguishing edge-weighting are consistent with these definitions.

As usual, we need a parameter referring to the minimum number of weights of a d-f-distinguishing k-edge-weighting of a graph G. We call it the *d-f-distinguishing chromatic index* of G, and denote it $\chi'_{d,term}(G)$, where the subscript *term* is generally a term describing f. We voluntarily make use of the symbol χ', which is the standard notation for the chromatic index of graphs, to make clear the fact that we are weighting the edges only. As most of the vertex-distinguishing edge-weighting notions we consider throughout concern the distinguishing of the adjacent vertices of graphs (i.e. notions with $d = 1$), we write $\chi'_{term}(G)$ instead of $\chi'_{1,term}(G)$ for the sake of simplicity.

There may be graphs which do not admit a d-f-distinguishing edge-weighting for given d and f, consider for instance a vertex-sum-distinguishing edge-weighting of K_2. For such graphs, the d-f-distinguishing chromatic index is not finite by definition. We refer to those graphs as *exceptions* (for k-f-distinguishing edge-weighting of graphs). Conversely, a *weightable* (or *colourable* in case we are dealing with an edge-colouring notion) graph (for k-f-distinguishing edge-weighting of graphs) is a graph which is not an exception.

Every notion of d-f-distinguishing edge-weighting of G we consider naturally extends to a notion of *d-f^t-distinguishing total-weighting* of G, where a d-f^t-distinguishing total-weighting of G is a total-weighting distinguishing the vertices at distance at most d of G and $f^t(v)$ is

computed as $f(v)$ but with additionally taking the value assigned to v by the weighting into account (i.e. as if the weight on v were the weight assigned to an additional edge incident with v). Speaking of *d-f^t-distinguishing total chromatic number* then directly makes sense. Assuming the d-f-distinguishing chromatic index of graphs is denoted $\chi'_{d,term}$, we denote $\chi''_{d,term}$ the d-f^t-distinguishing total chromatic number in view of the relationship between the two underlying weighting notions.

Regarding list versions of some d-f-distinguishing edge-weighting or d-f^t-distinguishing total-weighting notions, assuming χ'_{term} and χ''_{term} denote the d-f-distinguishing chromatic index and d-f^t-distinguishing total chromatic number, respectively, we directly denote ch'_{term} and ch''_{term} the edge-choosability and total-choosability of the list versions of these problems.

7.3 Related work

We herein survey some of the background associated with the notion of neighbour-sum-distinguishing edge-weighting of graphs and some of its variants related to our investigations.

1-2-3 Conjecture

The study of neighbour-sum-distinguishing edge-weighting of graphs was initiated by Karoński, Łuczak and Thomason, who conjectured in [81] that every graph with no component isomorphic to K_2 should admit a neighbour-sum-distinguishing 3-edge-weighting.

1-2-3 Conjecture ([81]). *If G is a weightable graph, then $\chi'_{nsd}(G) \leq 3$.*

Weaker versions of the 1-2-3 Conjecture have been successively proved since its introduction. As a first result, Karoński, Łuczak and Thomason proved that the neighbour-sum-distinguishing chromatic index of weightable graphs is upper-bounded by 183 [81]. This upper bound was then decreased to 30 by Addario-Berry, Dalal, McDiarmid, Reed and Thomason [2], to 16 by Addario-Berry, Dalal and Reed [3], to 13 by Wang and Yu [118], and to 6 by Kalkowski, Karoński and Pfender in [78]. This last result has been then improved by the same authors in [79] to the following best result towards the 1-2-3 Conjecture.

Theorem 7.11 ([79]). *If G is a weightable graph, then $\chi'_{nsd}(G) \leq 5$.*

Exhibiting constant upper bounds on the neighbour-sum-distinguishing chromatic index of weightable graphs aside, most of works aimed at verifying the 1-2-3 Conjecture for various classes of graphs. One strategy for finding a neighbour-sum-distinguishing edge-weighting of a weightable graph G is to start from a proper vertex-colouring of its vertices, and then weight every edge of G accordingly to which parts of the vertex-partition its ends belong to. A strong relationship between the chromatic number and the neighbour-sum-distinguishing chromatic index of weightable graphs has hence been pointed out in several references of the literature. In this vein, let us mention the following result by Karoński, Łuczak and Thomason which implies that weightable graphs which admit a 3-vertex-colouring do not refute the 1-2-3 Conjecture [81].

Theorem 7.12 ([81]). *If a weightable graph G is k-colourable with $k \geq 1$ odd, then $\chi'_{nsd}(G) \leq k$.*

It is worth mentioning that the 1-2-3 Conjecture, if true, would be tight (consider e.g. C_6), but that if G is a random graph from $G(n,p)$, then asymptotically almost surely $\chi'_{nsd}(G) \leq 2$ according to a result by Addario-Berry, Dalal and Reed [3]. No easy classification of graphs with neighbour-sum-distinguishing chromatic index at most 2 is known so far. Actually the existence of such a classification would imply that P=NP holds, due to the fact that the decision problem below was shown to be NP-complete in general.

Neighbour-Sum-Distinguishing $\{a,b\}$-Edge-Weighting
Instance: a graph G.
Question: does G admit a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting?

The complexity of Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting was first considered by Dudek and Wajc in [52], wherein its NP-completeness is established for two values of $\{a, b\}$.

Theorem 7.13 ([52]). Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting *is NP-complete for* $\{a, b\} = \{1, 2\}$ *and* $\{a, b\} = \{0, 1\}$.

Later on, Ahadi, Dehghan and Sadeghi proved that Neighbour-Sum-Distinguishing $\{1, 2\}$-Edge-Weighting remains NP-complete when restricted to cubic graphs [4]. Sufficient conditions for particular classes of graphs to have neighbour-sum-distinguishing chromatic index at most 2 have consequently been gathered in several research works. Again, we refer the reader to the survey of Seamone [109] wherein most of these results at the moment are gathered. Perhaps one of the most intriguing open question in this scope is the case of bipartite graphs.

Problem 7.14. *Characterize bipartite graphs with neighbour-sum-distinguishing chromatic index at most 2.*

Two directions then arise regarding Problem 7.14, namely showing that Neighbour-Sum-Distinguishing $\{1, 2\}$-Edge-Weighting is NP-complete when restricted to bipartite graphs, which would reject the existence of an easy characterization (unless $\mathsf{P} = \mathsf{NP}$), or exhibiting such an easy classification. The prevalent feeling is that an easy characterization should exist, as only a few bipartite graphs needing the three weights among $\{1, 2, 3\}$ have been exhibited so far, refer e.g. to the paper of Davoodi and Omoomi [46] which summarizes these graphs. Towards this direction, many authors gathered sufficient conditions for specific classes of bipartite graphs to have neighbour-sum-distinguishing chromatic index at most 2, see e.g. the works of Chang, Lu, Wu and Yu [41], Davoodi and Omoomi [46], and Khatirinejad, Naserasr, Newman, Seamone and Stevens [84].

Alternate versions of the 1-2-3 Conjecture

Many notions related to neighbour-sum-distinguishing edge-weighting of graphs have been considered since the introduction of the 1-2-3 Conjecture. We only survey below those notions which are related to our investigations, but again we refer the reader to the up-to-date survey by Seamone [109] for more details on this topic.

Multiset version

One approach considered by Karoński, Łuczak and Thomason to deal with the 1-2-3 Conjecture is to consider the distinguishing of adjacent vertices by the *multisets* of their incident colours by an edge-colouring. This yields to the notion of *neighbour-multiset-distinguishing edge-colouring* of graphs.

Definition 7.15. Let c be an improper k-edge-colouring of a graph G. For every vertex v of G, let $m_c(v)$ be the multiset of colours incident with v by c. We say that w is *neighbour-multiset-distinguishing* if m_c is proper.

As usual, given a graph G the goal is to determine the *neighbour-multiset-distinguishing chromatic index* of G, i.e. the least number of colours used by a neighbour-multiset-distinguishing k-edge-colouring of G (if any).

Definition 7.16. The *neighbour-multiset-distinguishing chromatic index* $\chi'_{nmd}(G)$ of a graph G is the minimum number of colours of a neighbour-multiset-distinguishing k-edge-colouring of G (if any).

Note that there is a strong relationship between neighbour-sum-distinguishing edge-weighting and neighbour-multiset-distinguishing edge-colouring of graphs.

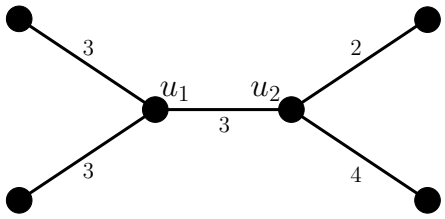


Figure 7.2: A graph and a neighbour-multiset-distinguishing 3-edge-colouring of it which is not neighbour-sum-distinguishing.

Observation 7.17. *Every neighbour-sum-distinguishing edge-weighting of a graph is neighbour-multiset-distinguishing.*

Therefore, we have $\chi'_{nmd}(G) \leq \chi'_{nsd}(G)$ for every graph G with no component isomorphic to K_2. It is worth noting that the counterpart of Observation 7.17 does not hold, i.e. a neighbour-multiset-distinguishing edge-colouring is not necessarily neighbour-sum-distinguishing.

Example 7.18. In Figure 7.2 is drawn a graph and a 3-edge-weighting w of it. It is easily checked that w is neighbour-multiset-distinguishing (in particular, we have $m_w(u_1) = \{3,3,3\} \neq \{2,3,4\} = m_w(u_2)$). But w is not neighbour-sum-distinguishing as $s_w(u_1) = 9 = s_w(u_2)$.

Not surprisingly in view of this relationship, as for the neighbour-sum-distinguishing chromatic index, it was conjectured by Addario-Berry, Aldred, Dalal and Reed that the neighbour-multiset-distinguishing chromatic index of every colourable graph should be upper-bounded by 3 [1].

Conjecture 7.19 ([1])**.** *If G is a colourable graph, then $\chi'_{nmd}(G) \leq 3$.*

Concerns regarding Conjecture 7.19 are roughly the same as those for the 1-2-3 Conjecture, though the 1-2-3 Conjecture has been receiving much more interest since, if true, it would imply Conjecture 7.19, recall Observation 7.17. Regarding Conjecture 7.19, the best known constant upper bound on $\chi'_{nmd}(G)$ is 4 and is due to Addario-Berry, Aldred, Dalal and Reed [1].

Theorem 7.20 ([1])**.** *If G is a colourable graph, then $\chi'_{nmd}(G) \leq 4$.*

It is worth noting that Theorem 7.11 directly implies that colourable graphs have neighbour-multiset-distinguishing chromatic index at most 5 according to Observation 7.17, but this upper bound is not better than the one of Theorem 7.20. Unfortunately, the proof of Theorem 7.20 cannot be adapted to prove that $\chi'_{nsd}(G) \leq 4$ for every weightable graph G.

Conjecture 7.19 was confirmed for several classes of graphs, namely for colouring graphs with large enough maximum degree by Addario-Berry, Aldred, Dalal and Reed [1], and some cubic and bipartite graphs by Havet, Paramaguru and Sampathkumar [68]. Let us in particular mention the following two results.

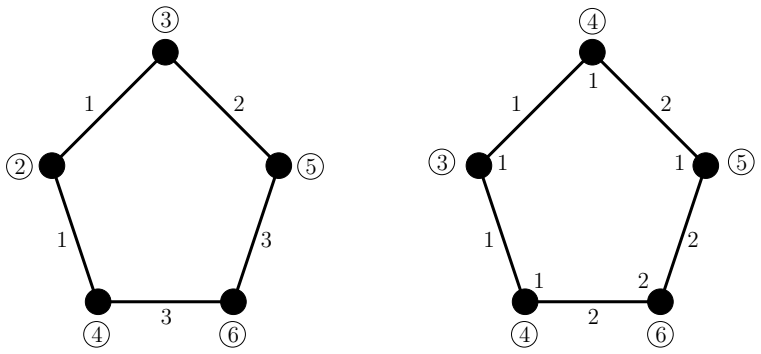
Theorem 7.21 ([1])**.** *If G is a colourable graph with $\Delta(G) \geq 1000$, then $\chi'_{nmd}(G) \leq 3$.*

Theorem 7.22 ([68])**.** *If G is a bipartite graph with $\delta(G) \geq 3$, then $\chi'_{nmd}(G) \leq 2$.*

We now turn our attention to the algorithmic point of view. Consider the following problem.

Neighbour-Multiset-Distinguishing k-Edge-Colouring
Instance: a graph G.
Question: do we have $\chi'_{nmd}(G) \leq k$?

Havet, Paramaguru and Sampathkumar established the hardness of Neighbour-Multiset-Distinguishing 2-Edge-Colouring in [68], relating to P=NP the existence of an easy characterisation of graphs with neighbour-multiset-distinguishing chromatic index at most 2.



(a) A 3-edge-weighting of C_5. (b) A 2-total-weighting of C_5.

Figure 7.3: Optimal neighbour-sum-distinguishing weightings of C_5. Weighted degrees are circled.

Theorem 7.23 ([68]). Neighbour-Multiset-Distinguishing 2-Edge-Colouring *is* NP*-complete, even when restricted to cubic graphs.*

As for the neighbour-sum-distinguishing chromatic index (Problem 7.14), the existence of an easy classification of bipartite graphs with neighbour-multiset-distinguishing chromatic index at most 2 is still an open question.

Problem 7.24. *Characterize bipartite graphs with neighbour-multiset-distinguishing chromatic index at most 2.*

The same remarks apply to Problem 7.24, namely that the NP-completeness of Neighbour-Multiset-Distinguishing 2-Edge-Colouring when restricted to bipartite graphs would reject the existence of an easy classification (unless P=NP), but that the small number of known bipartite graphs for which three colours are necessary tend to confirm the existence of such a classification (this was notably pointed out by Havet, Paramaguru and Sampathkumar in [68]).

1-2 Conjecture

Note that if a graph G has a neighbour-sum-distinguishing k-edge-weighting w for some $k \geq 1$, then a neighbour-sum-distinguishing k-total-colouring of G can be obtained by considering w and assigning every vertex of G the same value, say 1. The 1-2-3 Conjecture, if true, would then imply that $\chi''_{nsd}(G) \leq 3$ holds for every graph G admitting a neighbour-sum-distinguishing 3-edge-weighting. Noting that even K_2 admits a neighbour-sum-distinguishing 2-total-weighting, Przybyło and Woźniak raised the following conjecture [102].

1-2 Conjecture ([102]). *For every graph G, we have $\chi''_{nsd}(G) \leq 2$.*

The 1-2-3 Conjecture, if true, would imply that we can obtain a neighbour-sum-distinguishing edge-weighting of every weightable graph G using the weights among $\{1, 2, 3\}$ only. The 1-2 Conjecture states that if we drop the weight 3 for the edges but compensate by the use of a local weight at each vertex, then we should be able to obtain a neighbour-sum-distinguishing total-weighting of G using the weights among $\{1, 2\}$ only.

Example 7.25. C_5 is an example of graph whose neighbour-sum-distinguishing chromatic index is exactly 3 (Figure 7.3.a depicts a neighbour-sum-distinguishing 3-edge-weighting of C_5). But C_5 admits a neighbour-sum-distinguishing 2-total-weighting, see Figure 7.3.b.

Przybyło and Woźniak verified the 1-2 Conjecture for complete graphs, 3-colourable graphs, and 4-regular graphs [102]. As a first result towards the 1-2 Conjecture, they also proved that $\chi''_{nsd}(G) \leq 11$ for every graph G. This upper bound was then lowered to 7 in the case of regular graphs by Przybyło [100], before a breakthrough to 3 in the general case by Kalkowski [77].

Theorem 7.26 ([77]). *For every graph G, we have $\chi''_{nsd}(G) \leq 3$.*

Since a neighbour-sum-distinguishing k-total-weighting does not imply the existence of a neighbour-sum-distinguishing k-edge-weighting, Theorem 7.26 does not imply the 1-2-3 Conjecture. It is however worth mentioning that the proof of Theorem 7.11 is actually derived from the proof of Theorem 7.26, which consists in starting from an initial total-weighting and processing the vertices of the input graph linearly, and fixing the possible conflicts which may arise between a vertex and its predecessors by modifying the weighting locally. The proof of Theorem 7.11 was actually obtained by compensating the weighting of the vertices of the graph by the use of more edge weights, so that the process introduced by Kalkowski is applicable.

Product versions

One straight variant of most of the notions considered so far consists in considering, at each vertex, the product of incident weights rather than their sum. This yields the following definitions.

Definition 7.27. Let w be an improper k-edge-weighting of a graph G. For every vertex v of G, let

$$p_w(v) = \prod_{u \in N(v)} w(vu)$$

be the product of weights incident to v by w. We say that w is *neighbour-product-distinguishing* if p_w is proper. The minimum number of weights of a neighbour-product-distinguishing k-edge-weighting of G (if any) is the *neighbour-product-distinguishing chromatic index* of G, denoted $\chi'_{npd}(G)$.

This variant was mainly investigated by Skowronek-Kaziów in [112], wherein neighbour-product-distinguishing total-colouring of graphs is considered. Regarding this problem, Skowronek-Kaziów raised the following conjecture derived from the 1-2 Conjecture.

Conjecture 7.28 ([112]). *For every graph G, we have $\chi''_{npd}(G) \leq 2$.*

Conjecture 7.28 is still an open question, but Skowronek-Kaziów proved a weaker version of it in [112] using a technique similar to the one used to prove Theorem 7.20.

Theorem 7.29 ([112]). *For every graph G, we have $\chi''_{npd}(G) \leq 3$.*

List versions

One usual way of investigation regarding a weighting problem is to wonder how much harder a list version of the same problem is. We sum up below some such results regarding list versions of the 1-2-3 and 1-2 Conjectures.

The list version of the 1-2-3 Conjecture was addressed by Bartnicki, Grytczuk and Niwczyk in [14].

List 1-2-3 Conjecture ([14]). *For every weightable graph G, we have $ch'_{nsd}(G) \leq 3$.*

Most of the results concerning the List 1-2-3 Conjecture have been proved using Alon's Combinatorial Nullstellensatz method (see [6]). Bartnicki, Grytczuk and Niwczyk first verified the List 1-2-3 Conjecture for complete graphs, trees, and complete bipartite graphs [14]. More results regarding some particular classes of graphs may also be found in works by Seamone [110, 111], as well as the following upper bound on ch'_{nsd}, which is the best known bound related to the List 1-2-3 Conjecture, though not constant.

Theorem 7.30 ([110], [111]). *For every weightable graph G, we have $ch'_{nsd}(G) \leq 2\Delta(G) + 1$.*

The list version of the 1-2 Conjecture was raised by Przybyło and Woźniak in [103].

List 1-2 Conjecture ([103]). *For every graph G, we have $ch''_{nsd}(G) \leq 2$.*

As first results, Przybyło and Woźniak showed in [103] that the List 1-2 Conjecture is verified for trees, wheels, unicyclic graphs, and complete graphs. No weaker version of the List 1-2 Conjecture has however been proved so far. An upper bound on ch''_{nsd}, dependent of the maximum degree, was again given by Seamone, who showed that $ch''_{nsd}(G) \leq \lceil \frac{2}{3}\Delta(G) \rceil + 1$ holds for every graph G [110, 111].

Directed versions

Vertex-distinguishing problems can also be considered for directed graphs, either by imagining a distinguishing parameter specific to directed graphs (e.g. function of the indegrees and/or outdegrees) or by deriving a vertex-distinguishing problem related to undirected graphs for directed graphs. It is worth specifying that, depending on the adopted definitions, there may be no systematic relationship between a vertex-distinguishing undirected problem and one directed variant of it.

To our knowledge, the only directed notion related to the 1-2-3 Conjecture which has been investigated in the literature is the following one.

Definition 7.31. Let w be an improper arc-weighting of a directed graph D. For every vertex v of D, let

$$q_w(v) = \Big(\sum_{u \in N^-(v)} w(\overrightarrow{uv})\Big) - \Big(\sum_{u \in N^+(v)} w(\overrightarrow{vu})\Big).$$

We say that w is *neighbour-potential-distinguishing* if q_w is proper. The minimum number of weights of a neighbour-potential-distinguishing k-arc-weighting of D is the *neighbour-potential-distinguishing chromatic index* of D, denoted $\chi'_{nqd}(D)$.

These definitions were considered by Borowiecki, Grytczuk and Pilśniak in [37] and Khatirinejad, Naserasr, Newman, Seamone and Stevens in [83], who proved that $ch'_{nqd}(D) \leq 2$ holds for every directed graph D. The proof by Borowiecki, Grytczuk and Pilśniak is a constructive one, while the one by Khatirinejad, Naserasr, Newman, Seamone and Stevens makes use of the Combinatorial Nullstellensatz method mentioned earlier.

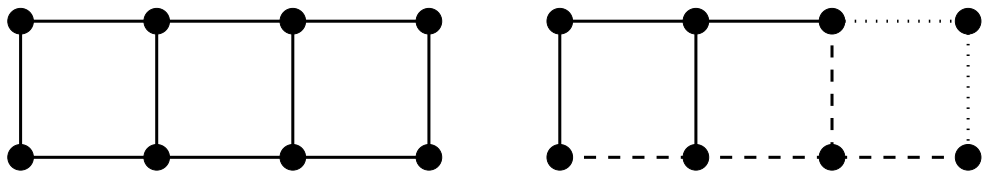
Theorem 7.32 ([37] and [83], independently). *For every directed graph D, we have $ch'_{nqd}(D) \leq 2$.*

7.4 Contributions of Part II

Chapter 8: Complexity of Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting

In [52], Dudek and Wajc suggested that their proof of Theorem 7.13 could be generalized to a proof that Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting is NP-complete no matter what is $\{a, b\}$, with a and b being real weights. Chapter 8 is dedicated to the proof of this statement. To that end, we introduce a reduction framework, and provide three implementations of it involving different kinds of gadgets (dedicated to three main values of $\{a, b\}$) which eventually stand as a proof that Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting is indeed NP-complete for every pair $\{a, b\}$ of real weights.

It is worth mentioning that, quite recently, Ahadi, Dehghan and Sadeghi improved Theorem 7.13 by showing Neighbour-Sum-Distinguishing $\{1, 2\}$-Edge-Weighting to remain NP-complete when restricted to cubic graphs [4]. It is easily seen that, in a cubic graph, we can easily deduce a neighbour-sum-distinguishing $\{a', b'\}$-edge-weighting from a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting w (just weight a' all edges weighted a by w, and weight b' all



(a) A graph G.

(b) A decomposition of G into three locally irregular subgraphs (in solid, dashed, and dotted edges, respectively).

Figure 7.4: Decomposing a graph into locally irregular subgraphs.

edges weighted b by w), and vice versa. So their proof for cubic graphs is a more straight proof of the main result we prove in Chapter 8. We however think that our gadgets remain of interest, in particular for the study of the List 1-2-3 Conjecture or for exhibiting new graphs for which three edge weights are necessary to obtain a neighbour-sum-distinguishing edge-weighting.

Chapter 9: Locally irregular edge-colouring of graphs

In Chapter 9, we introduce another way for introducing local irregularity in a graph which is not locally irregular itself. More precisely, if a graph G is not locally irregular, then we would like to edge-partition G into locally irregular subgraphs.

Definition 7.33. A graph G is *decomposable into k locally irregular subgraphs* if there is a partition $E_1 \cup E_2 \cup ... \cup E_k$ of $E(G)$ such that $G[E_i]$ is locally irregular for every $i \in \{1, 2, ..., k\}$.

Example 7.34. The graph G depicted in Figure 7.4.a is not locally irregular as it has neighbouring 2-vertices and neighbouring 3-vertices. But G admits an edge-partition into three locally irregular subgraphs, see Figure 7.4.b.

Actually every decomposition $E_1 \cup E_2 \cup ... \cup E_k$ of G into k locally irregular subgraphs can be regarded as an improper k-edge-colouring where each colour class induces a locally irregular subgraph of G. If we now consider the minimum number of locally irregular subgraphs into which a graph can be decomposed (assuming it can be decomposed), we get the following additional definitions.

Definition 7.35. An improper k-edge-colouring c of a graph G is *locally irregular* if each of the k colour classes of c induces a locally irregular subgraph of G. The least number of colours used by a locally irregular edge-colouring of G (if any) is called the *irregular chromatic index*, and is denoted $\chi'_{irr}(G)$.

One important source of motivation for studying locally irregular edge-colouring of graphs is that this kind of edge-colouring is related to the notions of neighbour-sum-distinguishing edge-weighting and neighbour-multiset-distinguishing edge-colouring of graphs. So finding upper bounds on the irregular chromatic index can have consequences on the neighbour-sum-distinguishing and neighbour-multiset-distinguishing chromatic indices. In particular, the following observation provides another way for dealing with Conjecture 7.19.

Observation 7.36. *Every locally irregular edge-colouring of a graph is neighbour-multiset-distinguishing.*

Indeed, note that if c is a locally irregular edge-colouring of a graph G, then, for every two adjacent vertices u and v of G, the i-degree of u is different from the i-degree of v assuming $c(uv) = i$. Then the element i does not appear the same number of times in $m_c(u)$ and $m_c(v)$, making $m_c(u)$ and $m_c(v)$ being different. Observation 7.36 does not hold the other way around though, since, if c' is a neighbour-multiset-distinguishing edge-colouring of G, it might be the case that $c'(uv) = i$ and $m_{c'}(u)$ and $m_{c'}(v)$ include the same number of i's (as long as there is a

value $j \neq i$ which does not appear the same number of times in $m_{c'}(u)$ and $m_{c'}(v)$). In such a situation, the vertices u and v have the same i-degree and are adjacent in the i-subgraph, so c' is not locally irregular.

There is no systematic relationship between a locally irregular edge-colouring and a neighbour-sum-distinguishing edge-weighting. However, in specific situations, a locally irregular edge-colouring is also a neighbour-sum-distinguishing edge-weighting. This is especially the case regarding regular graphs and when only two colours are used. Since the status of the 1-2-3 Conjecture regarding regular graphs is still not clear, the following observation is of interest.

Observation 7.37. *Every locally irregular* 2*-edge-colouring of a regular graph is neighbour-sum-distinguishing, and vice-versa.*

As mentioned in Definition 7.35, some graphs cannot be decomposed into locally irregular subgraphs at all. So our first result, in Section 9.1, is a concrete classification of those graphs whose irregular chromatic index is not finite. Though the number of such non-colourable graphs is more substantial than, for instance, the number of graphs which do not admit a neighbour-sum-distinguishing edge-weighting, these graphs can nevertheless be recognized in polynomial time.

We then investigate, in Section 9.2, the irregular chromatic index of colourable graphs. Our experiments on common classes of graphs suggest that 3 should be a reasonable upper bound on finite irregular chromatic indices, see Conjecture 9.9. Among the classes of graphs we consider, it is worth mentioning that regular graphs (whose irregular chromatic index is investigated in Section 9.2.2) are of high interest because of Observation 7.37. Since the status of the 1-2-3 Conjecture is still not clear regarding regular graphs, studying locally irregular decompositions of regular graphs gains yet more interest.

In Section 9.3, we consider the algorithmic hardness of the problem of determining the irregular chromatic index of a graph, which can be formally expressed as follows.

Locally Irregular k-Edge-Colouring
Instance: a graph G.
Question: do we have $\chi'_{irr}(G) \leq k$?

We give both positive and negative results regarding the complexity of Locally Irregular k-Edge-Colouring. We first propose, in Section 9.3.2, a linear-time algorithm for determining the irregular chromatic index of every tree, establishing, for every $k \geq 1$, the membership of Locally Irregular k-Edge-Colouring to P when restricted to trees. We however show that Locally Irregular 2-Edge-Colouring is NP-complete in general, see Section 9.3.3.

Chapter 10: Neighbour-outsum-distinguishing arc-weighting of oriented graphs

As mentioned in Section 7.3, there were only a few attempts to generalize vertex-distinguishing notions to the context of oriented graphs. One such oriented variant is considered in Chapter 10, wherein we introduce an oriented version of the 1-2-3 Conjecture based on the following notions of *weighted outdegree* of a vertex and *neighbour-outsum-distinguishing arc-weighting* of an oriented graph.

Definition 7.38. Let w be an improper arc-weighting of an oriented graph $\overrightarrow{G}$. The *weighted outdegree* of a vertex v of $\overrightarrow{G}$ is defined as

$$s^+_w(v) = \sum_{u \in N^+(v)} w(\overrightarrow{vu}).$$

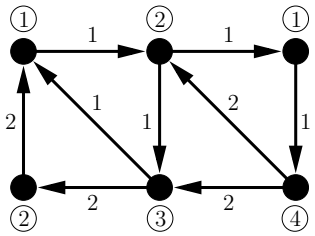


Figure 7.5: An oriented graph and a neighbour-outsum-distinguishing 2-arc-weighting of it. Weighted outdegrees are circled.

We say that w is *neighbour-outsum-distinguishing* if s_w^+ is proper. The minimum number of weights of a neighbour-outsum-distinguishing k-arc-weighting of $\overrightarrow{G}$ is the *neighbour-outsum-distinguishing chromatic index* of $\overrightarrow{G}$, denoted $\chi'_{nsd}(\overrightarrow{G})$.

Example 7.39. Figure 7.5 depicts an oriented graph and a neighbour-outsum-distinguishing 2-arc-weighting of it.

Note that we could have, in Definition 7.38 and in further ones, considered the sum of ingoing weights instead on the sum of outgoing weights, but the two related problems are equivalent. Regarding these definitions, we prove an oriented version of the 1-2-3 Conjecture in Section 10.1. Namely, we prove that every oriented graph admits a neighbour-outsum-distinguishing 3-arc-weighting. Our proof of this statement is very simple, and also holds notably for a list version of the same problem. We then focus on those oriented graphs which even admit a neighbour-outsum-distinguishing 2-arc-weighting. Although we show, in Section 10.2, that several families of oriented graphs admit such a weighting, we prove in Section 10.3 that an easy classification of oriented graphs with neighbour-outsum-distinguishing chromatic index at most 2 exists if and only if $\mathsf{P} = \mathsf{NP}$. This is done by showing that the following problem is NP-complete.

Neighbour-Outsum-Distinguishing 2-Arc-Weighting
Instance: an oriented graph $\overrightarrow{G}$.
Question: do we have $\chi'_{nsd}(\overrightarrow{G}) \leq 2$?

In Section 10.4 is discussed an analogous oriented version of the 1-2 Conjecture, which is defined very similarly as above. Again, we give a concrete answer to this conjecture. Namely, we prove it to be false.

Chapter 11: Locally irregular arc-colouring of oriented graphs

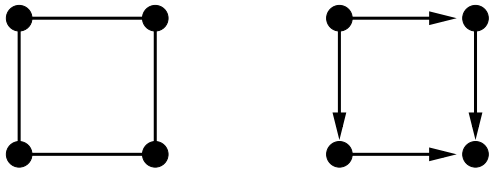
The second oriented problem we turn our attention to, in Chapter 11, is about the decomposition of oriented graphs into locally irregular subraphs, where the notions of *totally* and *locally irregular oriented graph* are again defined with respect to the outdegree parameter only.

Definition 7.40. An oriented graph $\overrightarrow{G}$ is *totally* (resp. *locally*) *irregular* if every two vertices (resp. neighbours) of $\overrightarrow{G}$ have distinct outdegrees.

Observation 7.41. *An oriented graph which is not totally or locally irregular may admit orientations which are totally or locally irregular (see Figure 7.6).*

The notion of *decomposition* of an oriented graph *into locally irregular subgraphs* and of *locally irregular arc-colouring* are then defined analogously as in the undirected case.

Definition 7.42. An oriented graph $\overrightarrow{G}$ is *decomposable into k locally irregular subgraphs* if there is a partition $A_1 \cup A_2 \cup ... \cup A_k$ of $A(\overrightarrow{G})$ such that $\overrightarrow{G}[A_i]$ is locally irregular for every $i \in \{1, 2, ..., k\}$. Equivalently, an improper k-arc-colouring c of an oriented graph $\overrightarrow{G}$ is *locally*



(a) An undirected graph which is not locally irregular.

(b) An oriented graph which is locally irregular.

Figure 7.6: An undirected graph which is not locally irregular, and a locally irregular orientation of it.

irregular if each of the k colour classes of c induces a locally irregular subgraph of $\overrightarrow{G}$. The least number of colours used by a locally irregular edge-colouring of $\overrightarrow{G}$ is called the *irregular chromatic index*, and is denoted $\chi'_{irr}(\overrightarrow{G})$.

As for the undirected case, our investigations are somehow justified since an oriented graph $\overrightarrow{G}$ is totally irregular if and only if $\overrightarrow{G}$ is a transitive tournament. Hence totally irregular oriented graphs have a very restricted structure. But note that contrary to the undirected case, the irregular chromatic index of every oriented graph is finite since a single arc is locally irregular.

Similarly as for the undirected case, we suspect every oriented graph to be decomposable into at most three locally irregular subgraphs. This conjecture is the main concern of Chapter 11. As a support, we show it to hold for several common classes of oriented graphs in Section 11.1. Towards the conjecture, we then prove, in Section 11.2, that every oriented graph is decomposable into at most six locally irregular subgraphs. We then focus on the algorithmic point of view in Section 11.3, and show that even if our guess were true, it would remain difficult to determine the exact irregular chromatic index of an oriented graph in general (unless $\mathsf{P} = \mathsf{NP}$). This is done by showing that the following problem is NP-complete when $k = 2$.

Locally Irregular k-Arc-Colouring
Instance: an oriented graph $\overrightarrow{G}$.
Question: do we have $\chi'_{irr}(\overrightarrow{G}) \leq k$?

Chapter 8

Complexity of Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting

In this chapter, we complete Theorem 7.13 by proving the following result.

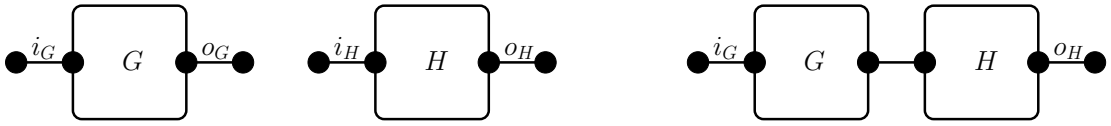
Theorem 8.1. Neighbour-Sum-Distinguishing {a,b}-Edge-Weighting *is NP-complete for every pair* $\{a, b\}$ *of distinct real weights.*

It is worth recalling that Theorem 8.1 also directly follows from a result proved recently by Ahadi, Dehghan and Sadeghi in [4], wherein it is shown that Neighbour-Sum-Distinguishing {1,2}-Edge-Weighting is NP-complete when restricted to cubic graphs. This result implies Theorem 8.1, as explained in Section 7.4.

Our proof of Theorem 8.1 is much more complicated since it consists in three slightly different implementations of a same reduction scheme. These three implementations, which are basically dedicated to distinct particular values of $\{a, b\}$, involve different gadgets and are here necessary as two problems Neighbour-Sum-Distinguishing {a,b}-Edge-Weighting and Neighbour-Sum-Distinguishing {a′,b′}-Edge-Weighting may actually be quite different. In particular, gadgets admitting a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting may not admit a neighbour-sum-distinguishing $\{a', b'\}$-edge-weighting. Although our proof of Theorem 8.1 is less significant due to the simpler proof by Ahadi, Dehghan and Sadeghi, we believe some of the techniques and gadgets (in particular the notion of *replacement gadget*) we use remain of interest and could be of some use for further studies on neighbour-sum-distinguishing edge-weighting of graphs (especially regarding Problem 7.14 or the List 1-2-3 Conjecture).

This chapter is organized as follows. We first give some notation and preliminary results in Section 8.1. These are used in Section 8.2 to introduce the reduction framework for showing that a Neighbour-Sum-Distinguishing {a,b}-Edge-Weighting problem is NP-complete by reduction from 3-Satisfiability. Three implementations of this framework are then provided in Sections 8.3, 8.4, and 8.5 for distinct values of $\{a, b\}$. These three implementations eventually perform a proof of Theorem 8.1.

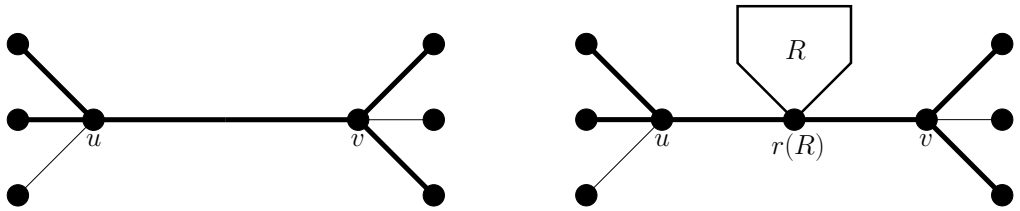
The whole content of this chapter is part of an article which was submitted for publication [34].



(a) Two graphs G and H with input i_G and i_H, respectively, and output o_G and o_H, respectively.

(b) The connection of G and H along o_G and i_H.

Figure 8.1: Illustration of the connection operation.



(a) A graph G with an edge uv.

(b) An R-subdivision of uv.

Figure 8.2: Illustration of the R-subdivision operation.

8.1 Notation, terminology and preliminary remarks

We start with some terminology and notation related to a graph construction used in next sections to describe our reduction framework. Let G be a graph. An *input* (resp. *output*) of G is an edge $i = uv$ (resp. $o = vu$) such that $d(u) = 1$. Assuming G has x (resp. y) inputs (resp. outputs) ordered arbitrarily, we sometimes refer to these inputs (resp. outputs) as $i_1(G), i_2(G), ..., i_x(G)$ (resp. $o_1(G), o_2(G), ..., o_y(G)$). Consider now two graphs G and H such that o and i are an output of G and an input of H, respectively. The *connection* of G and H *along* o *and* i is the graph obtained by taking the disjoint union of G and H, and then *identifying* the edges o and i, i.e. identifying the vertices u_1 and v_1 and identifying the vertices u_2 and v_2 assuming u_2 and v_1 are the 1-vertices of $o = u_1u_2$ and $i = v_1v_2$. Assuming we are given an x-tuple $(o_1, o_2, ..., o_x)$ of x outputs of G and an x-tuple $(i_1, i_2, ..., i_x)$ of x inputs of H, one can similarly define the connection of G and H along x outputs of G and x inputs of H where the resulting graph is obtained by identifying o_j and i_j for every $j \in \{1, 2, ..., x\}$. In this situation we say that G and H are *connected along* $(o_1, o_2, ..., o_x)$ *and* $(i_1, i_2, ..., i_x)$. The inputs and outputs of any graph resulting from the connection of G and H are those of G and H which have not been used for the connection.

Example 8.2. Figure 8.1 illustrates the connection of two graphs G and H along one output o_G of G and one input i_H of H. The resulting graph has one input i_G, which is the original input i_G of G, and one output o_H, which is the original output o_H of H.

Denote by G' the graph obtained by connecting G and H along $(o_1, o_2, ..., o_x)$ and $(i_1, i_2, ..., i_x)$. Given a W-edge-weighting w of G, an *extension* of w from G to G' is a W-edge-weighting w' of G' such that we have $w'(e) = w(e)$ for every edge e which originally belonged to G. Note in particular that if e is the edge resulting from the identification of o_j and i_j for some $j \in \{1, 2, ..., x\}$, then we have $w'(e) = w(o_j)$. When no ambiguity is possible, an extension of w from a graph to another graph is denoted w as well.

Assume uv is an edge of G. Now consider a graph R such that R has two inputs $u''z$ and zv'' where u'' and v'' are the degree-1 vertices of these inputs. By *R-subdividing* uv, we mean that we "replace" the edge uv with R. More precisely, we first remove the edge uv from G, then attach a new vertex u' to u and one new vertex v' to v (so that $d(u') = d(v') = 1$), and finally connect G and R along (uu', vv') and $(u''z, v''z)$.

Figure 8.3: Two possible neighbour-sum-distinguishing $\{a, b\}$-edge-weightings of P_4. Thick (resp. thin) edges represent a- (resp. b-) weighted edges.

Example 8.3. An R-subdivision of the edge uv of the graph depicted in Figure 8.2.a is depicted in Figure 8.2.b.

We now give properties of neighbour-sum-distinguishing $\{a, b\}$-edge weighting of graphs.

Observation 8.4. *Let P_4 denote the path $u_1u_2u_3u_4$ of length 3. If w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of P_4, then $w(u_1u_2) \neq w(u_3u_4)$.*

Proof. Suppose $w(u_1u_2) = a$ without loss of generality. Then u_2u_3 can be weighted either a or b. In the first case, we have $w(u_3u_4) = b$ since otherwise we would have $s_w(u_2) = s_w(u_3) = 2a$. In the second case, we have $w(u_3u_4) = b$ since otherwise we would have $s_w(u_2) = s_w(u_3) = a+b$ ■

Example 8.5. Two neighbour-sum-distinguishing $\{a, b\}$-edge-weightings of $P_4 = u_1u_2u_3u_4$ are depicted in Figure 8.3. As claimed in Observation 8.4, the two endedges have distinct weights.

Suppose we have $w(o) = a$ for an output o of a graph G and a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting w of G. Then, in the graph resulting from the connection of G and $P_4 = u_1u_2u_3u_4$ along o and u_1u_2 (this operation is similar to performing a P_4-subdivision of o), necessarily for every neighbour-sum-distinguishing extension of w we have $w(u_3u_4) = b$ by Observation 8.4. Therefore, P_4-subdividing an output is an operation that can be used to "invert" the weight at an output of a graph by a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting.

The next result gives an equivalent condition for two adjacent vertices with the same degree to have distinct weighted degrees.

Lemma 8.6. *Let u and v be two adjacent vertices of a graph G such that $d(u) = d(v)$, and w be an $\{a, b\}$-edge-weighting of G. Then we have $s_w(u) = s_w(v)$ if and only if $m_w(u) = m_w(v)$.*

Proof. We clearly have $s_w(u) = s_w(v)$ when $m_w(u) = m_w(v)$. Now suppose that $m_w(u) \neq m_w(v)$. Then, u and v are incident to x and x' edges weighted a by w, respectively, and y and y' edges weighted b, respectively. Besides, we have $x \neq x'$ and $y \neq y'$ since $m_w(u) \neq m_w(v)$ and $d(u) = d(v)$. Because $d(u) = d(v)$, we have $x + y = x' + y'$. Now, if $s_w(u) = s_w(v)$, then we have $xa + yb = x'a + y'b$, and, because $x - x' \neq 0$, we get that $a = b$ which is impossible by definition of an $\{a, b\}$-edge-weighting. It then follows that $s_w(u) \neq s_w(v)$. ■

By Lemma 8.6, it follows that if u and v are two adjacent vertices of G such that $d(u) = d(v)$, then we only have to check whether $m_w(u) \neq m_w(v)$ while checking, regarding u and v, whether an edge-weighting w of G is neighbour-sum-distinguishing.

We now introduce the notion of *replacement gadget*.

Definition 8.7. A *replacement gadget* R for a pair $\{a, b\}$ of real weights is a graph with the following structural and weighting properties:

1. R has two inputs $i_1(R) = uz$ and $i_2(R) = zv$ (hence $d(u) = d(v) = 1$), and no output,
2. for every neighbour-sum-distinguishing $\{a, b\}$-edge-weighting w of R, we have $w(i_1(R)) = w(i_2(R))$,
3. there are neighbour-sum-distinguishing $\{a, b\}$-edge-weightings of R such that $i_1(R)$ is weighted a,
4. there are neighbour-sum-distinguishing $\{a, b\}$-edge-weightings of R such that $i_1(R)$ is weighted b,

5. there is a real number x such that, for every neighbour-sum-distinguishing $\{a,b\}$-edge-weighting w of R, if $w(i_1(R)) = a$, then $s_w(z) = x$,

6. there is a real number y such that, for every neighbour-sum-distinguishing $\{a,b\}$-edge-weighting w of R, if $w(i_1(R)) = b$, then $s_w(z) = y$,

We refer to the vertex z as $r(R)$ for convenience, where r stands for "root". To make apparent the possible weighted degrees of $r(R)$ by all neighbour-sum-distinguishing $\{a,b\}$-edge-weightings of R, we call R an *(x,y)-replacement gadget*.

Replacement gadgets can be used to reduce the number of conflicts by a non-neighbour-sum-distinguishing $\{a,b\}$-edge-weighting w of a graph G. Indeed, suppose that for an edge uv of G we have $w(uv) = a$ and $s_w(u) = s_w(v)$. To possibly solve this local conflict, one way to proceed is to "replace" uv by an (x_1,y_1)-replacement gadget R for $\{a,b\}$, i.e. to R-subdivide uv, and then try to extend w to the resulting graph, i.e. to the edges of R, in a neighbour-sum-distinguishing way.

Clearly, the weighted degrees of u and v are not altered by the extension of w to the augmented graph. Since R admits neighbour-sum-distinguishing $\{a,b\}$-edge-weightings where its inputs are weighted a, the only new possible conflicts which may arise are $s_w(u) = x_1$ or $s_w(v) = x_1$. If such a situation occurs, then we reveal what is the weighted degree of one of u and v by w. By then repeating the procedure above but with an (x_2,y_2)-replacement gadget for $\{a,b\}$ such that $x_1 \neq x_2$, the possibilities for getting another conflict when extending w in a neighbour-sum-distinguishing way to the resulting graph are reduced.

Example 8.8. Consider the graph G and the (partial) $\{a,b\}$-edge-weighting w of G depicted in Figure 8.2.a (where thick (resp. thin) edges represent a- (resp. b-) weighted edges). Clearly w is not neighbour-sum-distinguishing since $s_w(u) = s_w(v)$. Then uv is replaced with a replacement gadget R, and w is extended to the resulting graph, see Figure 8.2.b. As u and v are no longer adjacent, the conflict between u and v is "solved" (unless $s_w(u) = s_w(r(R))$ or $s_w(v) = s_w(r(R))$). Besides, since the weighted degrees of u and v are not altered by the modifications, new conflicts involving two vertices of G cannot arise.

We catch this observation within the next lemma.

Lemma 8.9. *Assume w is an $\{a,b\}$-edge-weighting of a graph G, and suppose we are given one (x_1,y_1)-, one (x_2,y_2)- and one (x_3,y_3)-replacement gadget R_1, R_2 and R_3 for $\{a,b\}$, respectively, where the x_i's are distinct and the y_i's are distinct. If $s_w(u) = s_w(v)$ for an edge uv of G, then there is an $i \in \{1,2,3\}$ for which there is an extension of w to the graph resulting from the R_i-subdivision of uv such that $s_w(u) \neq s_w(r(R_i))$ and $s_w(v) \neq s_w(r(R_i))$.*

Proof. We may suppose that $w(uv) = a$. Start by R_1-subdividing uv, and then extend w in a neighbour-sum-distinguishing way to the resulting graph in such a way that $w(i_1(R_1)) = w(i_2(R_1)) = a$. Such an extension exists by definition. If the claim is not verified, then $s_w(u) = s_w(r(R_1)) = x_1$ without loss of generality. Start over from the original graph and weighting. Now, R_2-subdivide uv before extending w in a neighbour-sum-distinguishing way to the resulting graph in such a way that $w(i_1(R_2)) = w(i_2(R_2)) = a$. Clearly we cannot have $s_w(u) = s_w(r(R_2))$ since $x_1 \neq x_2$. Thus, if the claim is still not verified, then $s_w(v) = x_2$. In this situation, repeat the same procedure a third time but with an R_3-subdivision of uv. Now the claim has to be true since other we would have either $s_w(u) = s_w(r(R_3))$ or $s_w(v) = s_w(r(R_3))$, which is impossible since $s_w(r(R_3)) = x_3$, $s_w(u) = x_1$, $s_w(v) = x_2$, and x_1, x_2 and x_3 are distinct. ■

Hence, assuming we are provided three sufficiently different replacement gadgets for $\{a,b\}$, by repeating the procedure used in the proof of Lemma 8.9 for every conflicting edge of G, i.e. every edge uv such that $s_w(u) = s_w(v)$, we can deduce a graph which looks like G and which admits a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting which looks like w.

Corollary 8.10. *Assume w is an $\{a,b\}$-edge-weighting of a graph G, and suppose we are given one (x_1,y_1)-, one (x_2,y_2)-, and one (x_3,y_3)-replacement gadget R_1, R_2 and R_3 for $\{a,b\}$, respectively, where the x_i's are distinct and the y_i's are distinct. Then there is a combination of subdivisions of the edges of G involving the R_i's such that the resulting graph admits a neighbour-sum-distinguishing extension of w.*

Proof. Suppose there are $x < |V(G)|^2$ edges $e_1, e_2, ..., e_x$ of G whose ends have the same weighted degree by w. Consider every such edge $e_i = uv$. Suppose $w(e_i) = a$ without loss of generality. By Lemma 8.9, there is one replacement gadget R among $\{R_1, R_2, R_3\}$ for which we can R-subdivide e_i and then extend w in a neighbour-sum-distinguishing way to the resulting graph in such a way that $w(i_1(R)) = w(i_2(R)) = a$, $s_w(u) \neq s_w(r(R))$ and $s_w(v) \neq s_w(r(R))$. Since this operation does not alter the weighted degrees of u and v, no new pair of adjacent vertices with the same weighted degree appeared after the modification. Hence, once the procedure has been applied to each of $e_1, e_2, ..., e_x$, there cannot be two adjacent vertices of G with the same weighted degree (by the last extension of w). ■

8.2 The hardness reduction framework

8.2.1 Overview of the framework

It should be clear that every Neighbour-Sum-Distinguishing {a,b}-Edge-Weighting problem is in NP: given an $\{a,b\}$-edge-weighting w of a graph G, one can compute the vertex-colouring s_w of G and check whether it is proper. This checking process can clearly be performed in polynomial time no matter what is the value of $\{a,b\}$.

The reduction framework we describe here aims at establishing the NP-hardness of every problem Neighbour-Sum-Distinguishing {a,b}-Edge-Weighting. The reduction is from 3-Satisfiability. Recall that we may assume that all possible literals over the variables of an instance formula F appear in F (such a restriction remains NP-hard, recall Observation 1.44). The reduction framework presented here has been frequently used in the literature to prove the hardness of edge-weighting problems (it was e.g. used by Holyer in [69] to prove the hardness of determining the chromatic index of a graph). From F, we produce a graph G_F such that, given a pair $\{a,b\}$ of real weights,

$$F \text{ is satisfiable}$$
$$\Leftrightarrow$$
$$G_F \text{ admits a neighbour-sum-distinguishing } \{a,b\}\text{-edge-weighting } w_F.$$

We use a simple analogy to describe the reduction scheme. The reduced graph G_F has to be thought of as an *electrical circuit* made up of *gadgets*, i.e. recurrent subgraphs, connected in a specific way. These gadgets are interconnected along several *inputs* and *outputs* that permit two *signals*, the *positive* and the *negative* ones, to be propagated along G_F. This propagation fulfils properties which are inspired by the propagation of a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting in a graph, where the positive and negative signals intend to represent the weights a and b, respectively. The structure of G_F is representative of the structure of F in the sense that the propagation of the positive signal through the gadgets is representative of the consequences on F of setting such or such variable of F to true. In this way, we get a straight analogy between spreading the positive signal through G_F and satisfying F.

The gadgets of G_F are the following. The graph G_F is made up of one *generator gadget* $G_F(S)$, of m *clause gadgets* $G_F(C_1)$, $G_F(C_2)$, ... , $G_F(C_m)$, and of $2n$ *literal gadgets* $G_F(\ell_1), G_F(\ell_2), ..., G_F(\ell_{2n})$. Every clause C_i in F is associated with the clause gadget $G_F(C_i)$, and similarly for every literal ℓ_i of F and the literal gadget $G_F(\ell_i)$. The graph G_F is obtained by originally considering $G_F(S)$, and then successively connecting gadgets to it. The generator

gadget is first connected to all of the m clause gadgets, which are each connected to some of the literal gadgets. In particular, if we denote by $\ell_{i_1}, \ell_{i_2}, ..., \ell_{i_{m(C_i)}}$ the distinct literals in C_i, then $G_F(C_i)$ is connected to $G_F(\ell_{i_1}), G_F(\ell_{i_2}), ..., G_F(\ell_{i_{m(C_i)}})$. Every literal gadget $G_F(\ell_i)$ of G_F thus has $n(\ell_i)$ inputs. Finally, the outputs of every two literal gadgets $G_F(\ell_i)$ and $G_F(\overline{\ell_i})$ are connected in a specific way.

Assuming every clause gadget $G_F(C_i)$ is supplied with the positive signal by the generator gadget, the main property of $G_F(C_i)$ is that it propagates the positive signal through at least one of its outputs, i.e. to at least one literal gadget $G_F(\ell_j)$ such that $\ell_j \in C_i$. The main property of a literal gadget $G_F(\ell_i)$ is that it outputs a signal if and only if the same signal comes in from all of its $n(\ell_i)$ inputs. Moreover, if a given signal comes in from the $n(\ell_i)$ inputs of $G_F(\ell_i)$, then $G_F(\ell_i)$ outputs the same signal, which must be different from the signal outputted by $G_F(\overline{\ell_i})$.

Hence, we have an equivalence between satisfying F and propagating the positive signal through G_F from the generator gadget:

- every clause C_i in F must have at least one true literal and $G_F(C_i)$ must spread the positive signal to at least one literal gadget it is connected to,
- every literal ℓ_i must have the same truth value in all clauses it appears in and all the inputs of $G_F(\ell_i)$ must spread the same signal in,
- every variable x_i and its negation $\overline{x_i}$ must have distinct truth values and the outputs of $G_F(x_i)$ and $G_F(\overline{x_i})$ must spread different signals out.

Example 8.11. The electrical circuit G_F obtained from the formula $F = C_1 \wedge C_2$ with $C_1 = (x_1 \vee \overline{x_1} \vee x_2)$ and $C_2 = (\overline{x_1} \vee x_2 \vee \overline{x_2})$ is depicted in Figure 8.4. The generator gadget $G_F(S)$ spreads the positive signal to both $G_F(C_1)$ and $G_F(C_2)$, which in turn propagate it to at least one of their attached literal gadgets, here to $G_F(\overline{x_1})$ and $G_F(x_2)$. The propagation is here correct as $G_F(\overline{x_1})$ and $G_F(x_2)$ are only supplied with the positive signal, while $G_F(x_1)$ and $G_F(\overline{x_2})$ are only supplied with the negative signal.

Now consider the graph theory point of view. The graph G_F, which is associated with the electrical circuit, is obtained by successively connecting graphs to the generator gadget $G_F(S)$. These graphs, i.e. the clause and literal gadgets, must fulfil the structural and weighting properties summed up above. In particular, assuming weight a (resp. b) of w_F is associated with the positive (resp. negative) signal, the propagation of the positive and negative signals may be seen as successive neighbour-sum-distinguishing extensions of w_F to the graphs successively obtained after the connections. We then get an analogy between satisfying F and extending w_F along G_F in a neighbour-sum-distinguishing way.

8.2.2 The reduction framework into details

In this section, we go into the details of the reduction framework by listing the properties the gadgets mentioned in Section 8.2.1 must fulfil, and how these gadgets are connected exactly to form G_F. Every implementation of the reduction framework in further sections will thus only consist in exhibiting gadgets and showing that these have the properties exhibited throughout this section.

8.2.2.1 Spreading gadget $G^{\curlywedge}$ and generator gadget $G_F(S)$

A generator gadget $G_F(S)$ for a given pair $\{a, b\}$ is obtained by connecting several *spreading gadgets* $G^{\curlywedge}$. A *spreading gadget* $G^{\curlywedge}$ for $\{a, b\}$ has one input, two outputs, and the following weighting property.

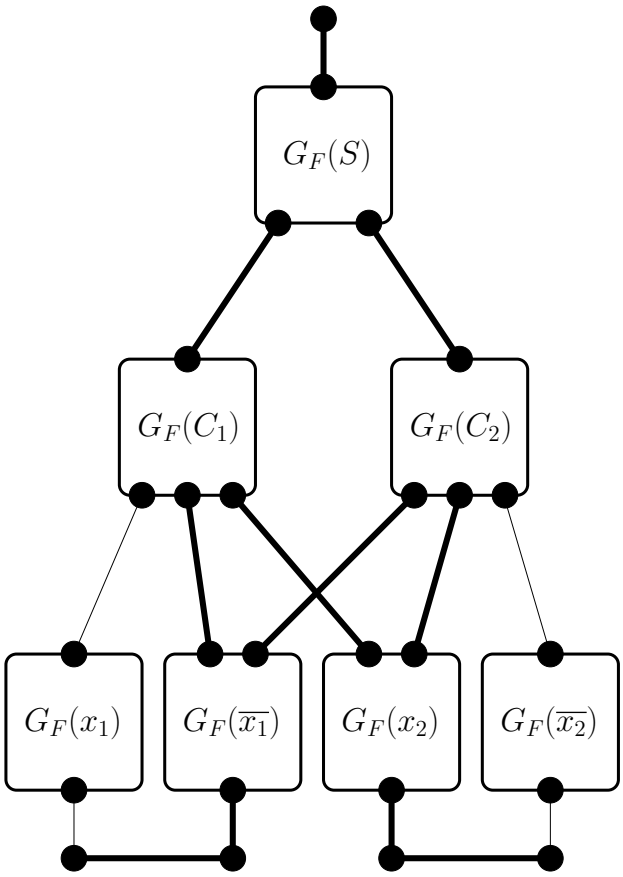


Figure 8.4: The electrical circuit G_F associated with the formula F with two clauses $C_1 = (x_1 \vee \overline{x_1} \vee x_2)$ and $C_2 = (\overline{x_1} \vee x_2 \vee \overline{x_2})$, and a propagation of the positive and negative signals in G_F. Thick (resp. thin) edges represent the positive (resp. negative) signal.

Property 8.12. *If w is a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting of $G^{\curlywedge}$, then we have $w(i_1(G^{\curlywedge})) = w(o_1(G^{\curlywedge})) = w(o_2(G^{\curlywedge}))$.*

Remark that by connecting two copies G_1 and G_2 of $G^{\curlywedge}$ along $o_1(G_1)$ and $i_1(G_2)$, we obtain a graph G' whose input and three outputs all receive the same weight by every neighbour-sum-distinguishing $\{a,b\}$-edge-weighting of G'. Indeed, suppose w is a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting of G' initiated with G_1, and that we have $w(i_1(G_1)) = a$ without loss of generality. Then, we have $w(o_1(G_1)) = w(o_2(G_1)) = a$ by Property 8.12. Note then that in every neighbour-sum-distinguishing extension of w from G_1 to G', we have $w(i_1(G')) = w(o_1(G')) = w(o_2(G')) = w(o_3(G'))$ since G_2 satisfies Property 8.12, and $o_1(G_1)$ and $i_1(G_2)$ refer to the same edge of G'.

Note that by repeating this construction several times, one obtains a graph with one input and arbitrarily many outputs such that all of these input and outputs necessarily receive the same weight by a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting. Now consider $P_4 = u_1u_2u_3u_4$, the path with length 3, let $i_1(P_4) = u_1u_2$ and $o_1(P_4) = u_3u_4$, and suppose w is a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting of $G^{\curlywedge}$. Note then that if G' is the graph obtained by connecting $G^{\curlywedge}$ and P_4 along $o_1(G^{\curlywedge})$ and $i_1(P_4)$ (in other words, the graph G' results from a P_4-subdivision of $o_1(G^{\curlywedge})$), then, assuming $w(i_1(G^{\curlywedge})) = w(o_1(G^{\curlywedge})) = a$, we have $w(o_1(G')) = b$ in every neighbour-sum-distinguishing extension of w from $G^{\curlywedge}$ to G' by Observation 8.4, where $o_1(G') = u_3u_4$. Consequently, by connecting arbitrarily many copies of $G^{\curlywedge}$ and then inverting some outputs of the resulting graph, we are able to propagate both the weights a and b towards an arbitrary number of directions assuming the input weight is known.

The generator gadget $G_F(S)$ is obtained in this way, i.e. by connecting several copies of

$G^{\curlywedge}$ and then inverting some outputs (so that we get the necessary number of "positive" and "negative" weights). The number of necessary connections (and hence of spreading gadgets) is not clarified here, but one can easily check that this number in polynomial in the size of F.

From now on, we suppose that w_F is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of G_F initiated with $G_F(S)$, and extended in a neighbour-sum-distinguishing way progressively as new gadgets are connected in order to form G_F. Suppose $w(i_1(G_F(S))) = a$ without loss of generality. Then, according to the remarks above, we know which outputs of $G_F(S)$ receive weight a (resp. b) by w_F. We say that these outputs are *positive* (resp. *negative*).

8.2.2.2 Clause gadget $G_F(C_i)$

The structure of a clause gadget $G_F(C_i)$ for $\{a, b\}$ depends on the value of $m(C_i)$. If $m(C_i) = 1$, then a clause gadget is not necessary. In every other situation, i.e. $m(C_i) = 2$ or $m(C_i) = 3$, the gadget $G_F(C_i)$ has a constant number of inputs and $m(C_i)$ outputs, and the following weighting property.

Property 8.13. *Assume w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of $G_F(C_i)$ such that particular inputs of $G_F(C_i)$ are weighted a by w, while its other inputs are weighted b. Then, up to $m(C_i)$ outputs, but at least one, are weighted a by w.*

Now connect $G_F(S)$ and every $G_F(C_i)$ in such a way that every input of $G_F(C_i)$ which is supposed to be weighted a (resp. b) is identified with one distinct positive (resp. negative) output of $G_F(S)$. Then, by Property 8.13, we know that arbitrarily many, but at least one, outputs of $G_F(C_i)$ can be assigned weight a in a neighbour-sum-distinguishing extension of w_F from $G_F(S)$ to G'_F, where G'_F is the graph resulting from the connections.

8.2.2.3 Collecting gadget $G^{\curlyvee}$ and literal gadget $G_F(\ell_i)$

The structure of every literal gadget $G_F(\ell_i)$ for $\{a, b\}$ depends on the value of $n(\ell_i)$. Once again, if $n(\ell_i) = 1$, then there is no need for a literal gadget. In every other case, i.e. whenever $n(\ell_i) \geq 2$, a literal gadget is obtained by connecting exactly $n(\ell_i) - 1$ *collecting gadgets* $G^{\curlyvee}$ in a specific way. A *collecting gadget* for $\{a, b\}$ has two "regular" inputs $i_1(G^{\curlyvee})$ and $i_2(G^{\curlyvee})$, and one output. It also has some "forcing" inputs which are supposed to be connected with positive or negative outputs of $G_F(S)$ so that the following property is fulfilled.

Property 8.14. *Assume w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of $G^{\curlyvee}$ such that particular forcing inputs of $G^{\curlyvee}$ are weighted a by w, while its other forcing inputs are weighted b. Then we have $w(i_1(G^{\curlyvee})) = w(i_2(G^{\curlyvee})) = w(o_1(G^{\curlyvee}))$.*

Now consider every literal ℓ_i of F. For every distinct clause C_j which contains ℓ_i, associate a distinct output of G'_F with $G_F(\ell_i)$ as follows:

- if ℓ_i is forced to true by C_j, then consider a positive output of $G_F(S)$,
- if ℓ_i is forced to false by C_j, then consider a negative output of $G_F(S)$,
- otherwise, consider one output of $G_F(C_j)$.

This association has to be done in such a way that every chosen output of G'_F is associated with exactly one literal gadget. The two first items above depict the fact that if a clause includes three occurrences of ℓ_i (resp. $\overline{\ell_i}$), then ℓ_i is forced to true (resp. false) by this clause. The third item reflects the fact that the truth value of a clause by a truth assignment of F depends on the truth values of the literals it contains.

The literal gadget $G_F(\ell_i)$ is obtained as follows. First define an arbitrary ordering (o_1, o_2, ..., $o_{n(\ell_i)}$) over the outputs of G'_F chosen above for $G_F(\ell_i)$. Now consider $n(\ell_i) - 1$ copies

$G_1, G_2, ..., G_{n(\ell_i)-1}$ of G^Υ, and connect these with G'_F as follows. Start by connecting G'_F and G_1 along (o_1, o_2) and $(i_1(G_1), i_2(G_1))$. Then connect the resulting graph and G_2 along $(o_1(G_1), o_3)$ and $(i_1(G_2), i_2(G_2))$. Next, connect the obtained graph and G_3 along $(o_1(G_2), o_4)$ and $(i_1(G_3), i_2(G_3))$. And so on. Denote by G''_F the resulting graph.

Note that if any two of the $n(\ell_i)$ outputs chosen above for $G_F(\ell_i)$ do not receive the same weight by w_F, then there is no neighbour-sum-distinguishing extension of w_F from G'_F to G''_F by Property 8.14. Thus, all the outputs of G'_F connected to $G_F(\ell_i)$ must receive the same weight by w_F.

8.2.2.4 Connecting the literal gadgets

The reduced graph G_F is finally obtained by adding some edges to G''_F. Consider every pair $\{\ell_i, \ell_j\}$ of literals of F such that $\ell_j = \overline{\ell_i}$. Now consider the respective outputs o_i and o_j of G''_F of the literal gadgets $G_F(\ell_i)$ and $G_F(\ell_j)$. More precisely, if $n(\ell_i) = 1$ and ℓ_i is forced to true (resp. false) by the only clause of F which contains ℓ_i, then o_i is a positive (resp. negative) output of $G_F(S)$. If $n(\ell_i) = 1$ and ℓ_i is not forced to some truth value by the only clause C_j which contains it, then consider one distinct output of $G_F(C_j)$. Otherwise, i.e. $n(\ell_i) \geq 2$, the output o_i is $o_1(G_F(\ell_i))$. Now, if u and v are the vertices with degree 1 of o_i and o_j, respectively, then let uv be an edge in G_F.

Now suppose any two such outputs o_i and o_j receive the same weight in an extension of w_F from G'_F to G''_F. Then note that w_F cannot be extended from G''_F to G_F in a neighbour-sum-distinguishing way since the edges o_i, uv and o_j induce a path on 4 vertices whose first and last edges have the same weight (Observation 8.4). On the contrary, if we have $w_F(o_i) \neq w_F(o_j)$, then assigning any weight to uv is correct. This simulates the fact that in a truth assignment of the variables of F, a variable and its negation are assigned distinct truth values.

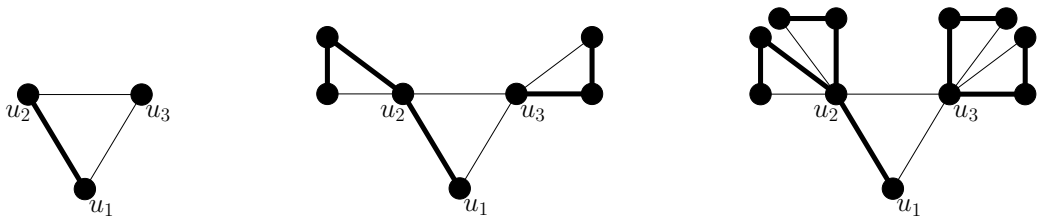
8.2.3 Final details

In every implementation of our framework, the extension of w_F in a neighbour-sum-distinguishing way along G_F is guaranteed by the different gadgets we use. In particular, if w_F is neighbour-sum-distinguishing at the end of the process, then no conflict involving two adjacent vertices from a same gadget can occur. But, for the sake of correctness, one would also have to check that, for specific values of $\{a, b\}$, no unexpected conflicts involving adjacent vertices from different gadgets may arise when two gadgets are connected.

As an illustration, suppose e.g. that uz is an output of a graph G, and $z'v$ is an input of another graph H. Suppose we are also given neighbour-sum-distinguishing $\{a, b\}$-edge-weightings w_G and w_H of G and H, respectively, such that $w_G(uz) = w_H(z'v) = a$. Now let G' be the graph resulting from the connection of G and H along uz and $z'v$, and consider the neighbour-sum-distinguishing $\{a, b\}$-edge-weighting $w_{G'}$ of G' where every edge e which originally belonged to G is weighted following w_G, and following w_H otherwise. Clearly the only possible conflict is $s_{w_{G'}}(u) = s_{w_{G'}}(v)$, which may occur for some particular values of a and b.

According to Lemma 8.9, we actually do not need to deeply study every possible connection of two gadgets of a given implementation, i.e. for a given value of $\{a, b\}$, in order to find out whether unexpected conflicts may arise when extending an $\{a, b\}$-edge-weighting in a neighbour-sum-distinguishing way. Assuming we are given one (x_1, y_1)-, one (x_2, y_2)-, and one (x_3, y_3)-replacement gadget for $\{a, b\}$ such that the x_i's are distinct and the y_i's are distinct, we can "replace" a conflicting edge by one of these gadgets so that we solve the conflict locally, and this without altering the weighting properties of G_F, i.e. without providing new ways for weighting G_F. In this way, the equivalence between satisfying F and weighting G_F is preserved.

Hence, even if we do not know exactly which edges are possibly conflicting in an $\{a, b\}$-edge-weighting of G_F for a specific value of $\{a, b\}$, we know that another good implementation of



(a) T_0. (b) T_1. (c) T_2.

Figure 8.5: The graphs T_0, T_1 and T_2, and neighbour-sum-distinguishing $\{a, b\}$-edge-weightings of T_0, T_1 and T_2. Thick (resp. thin) edges represent a- (resp. b-) weighted edges.

our reduction framework can be obtained for $\{a, b\}$ by simply replacing some edges of G_F by a convenient replacement gadget among a triplet of three replacement gadgets for $\{a, b\}$. Each of our framework implementations is thus provided with a *replacement triplet*, i.e. a triplet (R_1, R_2, R_3) where every R_i is an (x_i, y_i)-replacement gadget for $\{a, b\}$, and such that the x_i's are distinct and the y_i's are distinct.

8.3 First implementation: $0 \notin \{a, b\}$ and $b \neq -a$

In this section, we give a first implementation of our reduction framework in order to show that Neighbour-Sum-Distinguishing $\{a,b\}$-Edge-Weighting is NP-complete whenever $0 \notin \{a, b\}$ and $b \neq -a$. For this purpose, we introduce several graphs and notably show that some of them are spreading, clause, collecting or replacement gadgets for $\{a, b\}$.

Auxiliary gadget T_k and replacement triplet for $\{a, b\}$

We define the graphs T_k, where $k \geq 0$, which are used in other gadgets to "force" the propagation of a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting.

Construction 8.15. For a given value of $k \geq 0$, the graph T_k is obtained as follows. First consider a triangle $u_1u_2u_3u_1$. If $k = 0$, then we are done. Now, if $k \geq 1$, then identify u_2 with one arbitrary vertex of each of k new triangles. Finally, repeat the last step but with u_3 instead of u_2. We refer to the vertex u_1 as the *root* of T_k.

Example 8.16. The graphs T_0, T_1, and T_2 are depicted in Figure 8.5.

In the figures of the next sections, every triangle having "T_k" marked in it indicates that a vertex is identified with the root of a graph T_k. Every graph T_k with $k \geq 0$ has the following weighting properties.

Lemma 8.17. *Let $k \geq 0$ be fixed. If w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of T_k, then one of u_2 and u_3 has weighted degree $(k+1)(a+b)$, while the other vertex has weighted degree $(k+2)a+kb$ or $ka+(k+2)b$. Besides, we have $\{w(u_1u_2), w(u_1u_3)\} = \{a, b\}$.*

Proof. Note that for every triangle $v_1v_2u_iv_1$ different from $u_1u_2u_3u_1$, i.e. $i \in \{2, 3\}$, we have $w(u_iv_1) \neq w(u_iv_2)$ since otherwise we would have $s_w(v_1) = s_w(v_2)$. Therefore, the weighting of the triangles attached to u_2 and u_3 provide $k(a+b)$ in the weighted degrees of u_2 and u_3. Since $s_w(u_2) \neq s_w(u_3)$, we necessarily have $w(u_2u_1) \neq w(u_3u_1)$. Depending on whether $w(u_2u_3) = a$ or $w(u_2u_3) = b$, one of u_2 and u_3 has weighted degree $(k+2)a+kb$ or $ka+(k+2)b$. The other vertex has weighted degree $(k+1)(a+b)$. ■

Lemma 8.18. *Let $k \geq 0$ be fixed. If $0 \notin \{a, b\}$ and $b \neq -a$, then there is an $\{a, b\}$-edge-weighting w of T_k which is neighbour-sum-distinguishing unless $s_w(u_1) \in \{s_w(u_2), s_w(u_3)\}$.*

Proof. Recall that for every two adjacent vertices of T_k with the same degree, we only have to make sure that their multisets of incident weights by w are different according to Lemma 8.6. Note next that, for every triangle $v_1v_2u_iv_1$ different from $u_1u_2u_3u_1$, i.e. $i \in \{2,3\}$, one of v_1 and v_2 has weighted degree $a+b$, while the other vertex has weighted degree either $2a$ or $2b$ depending on how $v_1v_2u_iv_1$ is weighted (but we can "choose" this weighted degree by reweighting locally). Besides, one of u_2 and u_3 has weighted degree $(k+1)(a+b)$ by w, while the other vertex has weighted degree either $(k+2)a+kb$ or $ka+(k+2)b$ according to Lemma 8.17. Once again, this last weighted degree can be "chosen" freely by reweighting the edge u_2u_3.

Suppose $s_w(u_2) = (k+1)(a+b)$, and $s_w(u_3)$ is either $(k+2)a+kb$ or $ka+(k+2)b$ without loss of generality. On the one hand, consider u_2 and any triangle attached to it. Note first that if we have $s_w(u_2) = a+b$, i.e. $(k+1)(a+b) = a+b$, then either $k=0$ or $b=-a$. Now, observe that we cannot have both $s_w(u_2) = 2a$ and $s_w(u_2) = 2b$, unless $a=b$ which is impossible. On the other hand, consider u_3. Firstly, if we have both $(k+2)a+kb = a+b$ and $ka+(k+2)b = a+b$, then $a=b$. Secondly, if both $s_w(u_3) = 2a$ and $s_w(u_3) = 2b$ hold, then $a=b$ once again. Hence, the only possible conflict by w under our assumptions on a and b is $s_w(u_1) = s_w(u_2)$ or $s_w(u_1) = s_w(u_3)$. ■

Now consider the following graphs.

Construction 8.19. Let $k \geq 1$ be fixed, and assume $P_3 = v_1v_2v_3$ is the path of length 2. As T'_k, we refer to the graph obtained by identifying v_2 and the root of each of k graphs T_k. The two inputs of T'_k are $i_1(T'_k) = v_1v_2$ and $i_2(T'_k) = v_2v_3$.

We show that T'_k is a replacement gadget for $\{a,b\}$ under our assumptions on a and b.

Lemma 8.20. *Let $k \geq 1$ be fixed. If $0 \notin \{a,b\}$ and $b \neq -a$, then T'_k is a $((k+2)a+kb, ka+(k+2)b)$-replacement gadget for $\{a,b\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting of T'_k. Without loss of generality, we may suppose that $w(i_1(T'_k)) = a$ holds. For each of the graphs T_k attached to v_2, one of the two edges incident with v_2 is weighted a by w, while the other one is weighted b according to Lemma 8.17. Hence, the graphs T_k attached to v_2 provide $k(a+b)$ in the weighted degree of v_2. Besides, in every graph T_k there is a vertex neighbouring v_2 with weighted degree $(k+1)(a+b)$, while we may suppose that the other vertex neighbouring v_2 has weighted degree $ka+(k+2)b$.

Note then that if $w(v_2v_3) = b$, then we get $s_w(v_2) = (k+1)(a+b)$ and v_2 has the same weighted degree as some of its neighbours. Therefore, we have $w(v_2v_3) = a$. In this situation we have $s_w(v_2) = (k+2)a+kb$ while the vertices neighbouring v_2 from the graphs T_k have weighted degree $(k+1)(a+b)$ and $ka+(k+2)b$, respectively. Since v_2 and these vertices have the same degree, these weighted degrees are distinct by Observation 8.4. ■

Corollary 8.21. *Let $i,j,k \geq 1$ be three distinct positive integers. If $0 \notin \{a,b\}$ and $b \neq -a$, then (T'_i, T'_j, T'_k) is a replacement triplet for $\{a,b\}$.*

Spreading gadget $G^{\curlywedge}$ for $\{a,b\}$

Now consider the graph $G^{\curlywedge}$ depicted in Figure 8.6, whose input is u_1u_2, and whose two outputs are u_9u_{10} and $u_{12}u_{13}$. We show that $G^{\curlywedge}$ is a spreading gadget for $\{a,b\}$, i.e. that $G^{\curlywedge}$ satisfies Property 8.12, under our assumptions on a and b.

Proposition 8.22. *If $0 \notin \{a,b\}$ and $b \neq -a$, then $G^{\curlywedge}$ satisfies Property 8.12 for $\{a,b\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a,b\}$-edge-weighting w of $G^{\curlywedge}$. Note that we cannot have $w(u_3u_5) \neq w(u_4u_6)$. Indeed, suppose e.g. that $w(u_3u_5) = a$ and $w(u_4u_6) = b$. Because u_5 and u_6 are both attached to two graphs T_2, which form a graph T'_2, then we have

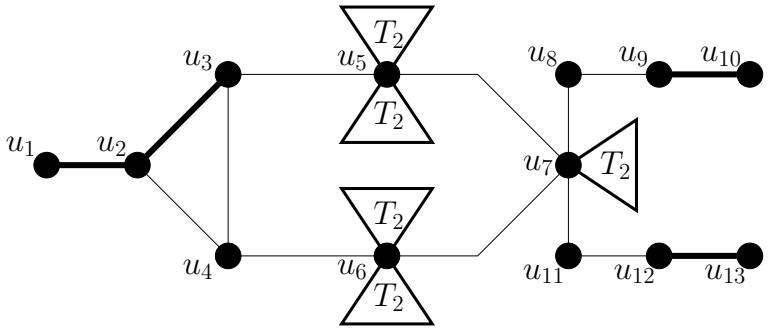


Figure 8.6: The spreading gadget $G^{\curlywedge}$ for the main implementation of the reduction framework, and a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of $G^{\curlywedge}$. Thick (resp. thin) edges represent a- (resp. b-) weighted edges.

$w(u_5u_7) = a$ and $w(u_6u_7) = b$ by Lemma 8.20. Besides, we have $s_w(u_5) = 4a + 2b$ and $s_w(u_6) = 2a + 4b$. We also know that a neighbour of u_7 from the graph T_2 attached to it has weighted degree $3a + 3b$, and that this graph T_2 provides $a + b$ in the weighted degree of u_7 according to Lemma 8.17. Then the vertex u_7 has weighted degree at least $2a + 2b$, and the two edges u_7u_8 and u_7u_{11} are weighted in such a way that the weighted degree of u_7 does not meet a value among $\{2a + 4b, 3a + 3b, 4a + 2b\}$, which is impossible.

On the contrary, if $w(u_3u_5) = w(u_4u_6) = a$ without loss of generality, then w can be neighbour-sum-distinguishing. Because of the arguments above, we have $w(u_5u_7) = w(u_6u_7) = a$ and $s_w(u_5) = s_w(u_6) = 4a + 2b$. Recall that we may assume that the weighting of the graph T_2 attached to u_7 is such that the two vertices which are adjacent with u_7 have weighted degree $3a + 3b$ and $4a + 2b$. Besides, the weighting of this graph T_2 provides $a + b$ in the weighted degree of u_7. Thus, the weighted degree of u_7 is at least $3a + b$, and the edges u_7u_8 and u_7u_{11} are weighted in such a way that the weighted degree of u_7 is not $3a + 3b$ or $4a + 2b$. The only possibility is to have $w(u_7u_8) = w(u_7u_{11}) = a$ since, in this situation, we get $s_w(u_7) = 5a + b$.

Now suppose $w(u_1u_2) = a$, and consider the edges u_2u_3 and u_2u_4. First, if $w(u_2u_3) = w(u_2u_4)$, then note that w cannot be neighbour-sum-distinguishing according to the arguments above since we would necessarily have $w(u_3u_5) \neq w(u_4u_6)$ so that $s_w(u_3) \neq s_w(u_4)$. Thus, $w(u_2u_3) = a$ and $w(u_2u_4) = b$ without loss of generality, and $s_w(u_2) = 2a + b$. Now note that if $w(u_3u_4) = a$, then we necessarily get that $s_w(u_3)$ or $s_w(u_4)$ is equal to $s_w(u_2)$ since we need $w(u_3u_5) = w(u_4u_6)$. Thus $w(u_3u_4) = b$. We then have $w(u_3u_5) = b$ so that $s_w(u_3) \neq s_w(u_2)$, and also $w(u_4u_6) = b$ so that $s_w(u_4) \neq s_w(u_3)$.

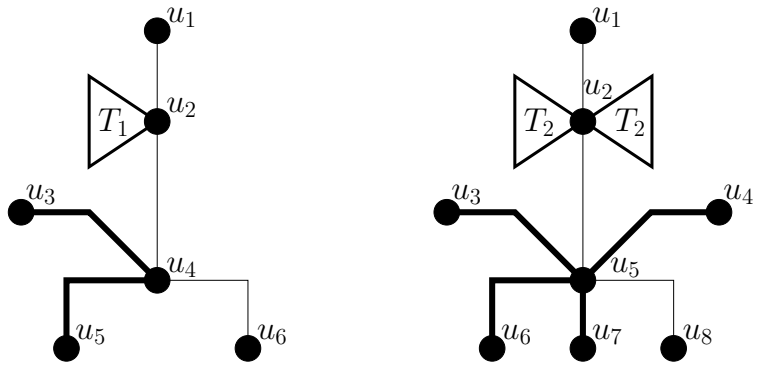
According to the arguments above, we have $w(u_3u_5) = w(u_4u_6) = b$ and $w(u_7u_8) = w(u_7u_{11}) = b$ under the assumption $w(u_1u_2) = a$. By Observation 8.4, we have $w(u_9u_{10}) = w(u_{12}u_{13}) = a = w(u_1u_2)$, as required. ■

Clause gadgets $G_F(C_i)$ for $\{a, b\}$

We distinguish two forms for $G_F(C_i)$, depending on whether $m(C_i) = 2$ or $m(C_i) = 3$. These two forms are depicted in Figure 8.7. In the first case, i.e. $m(C_i) = 2$, the inputs of $G_F(C_i)$ are u_3u_4, which is supposed to be weighted a, and u_1u_2, which is supposed to be weighted b, while the two outputs of $G_F(C_i)$ are u_4u_5 and u_4u_6. In the second case, i.e. $m(C_i) = 3$, the three inputs of $G_F(C_i)$ are u_1u_2, which is supposed to be weighted b, and u_3u_5 and u_4u_5 which are supposed to be weighted a. The three outputs of $G_F(C_i)$ are u_5u_6, u_5u_7 and u_5u_8 in this case.

We prove that these two types of gadgets both satisfy Property 8.13 under our assumptions on a and b.

Proposition 8.23. *Assume $m(C_i) = 2$. If $0 \notin \{a, b\}$ and $b \neq -a$, then $G_F(C_i)$ satisfies Property 8.13 for $\{a, b\}$.*



(a) Case $m(C_i) = 2$. (b) Case $m(C_i) = 3$.

Figure 8.7: The two forms of the clause gadget $G_F(C_i)$ for the main implementation of the reduction framework, and neighbour-sum-distinguishing $\{a, b\}$-edge-weightings of $G_F(C_i)$. Thick (resp. thin) edges represent a- (resp. b-) weighted edges.

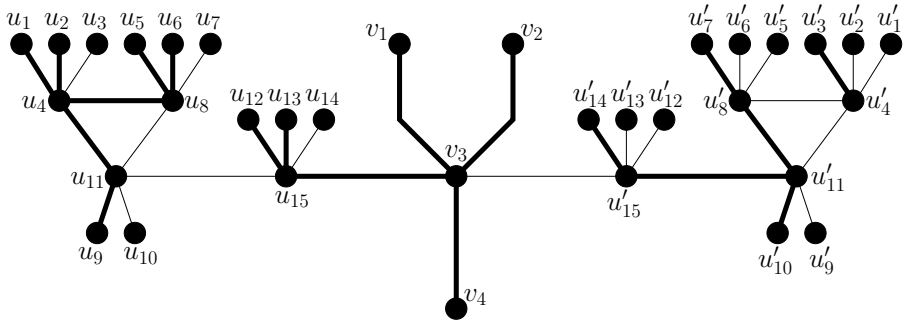


Figure 8.8: The collecting gadget $G^\curlyvee$ for the main implementation of the reduction framework, and a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of $G^\curlyvee$. Thick (resp. thin) edges represent a- (resp. b-) weighted edges.

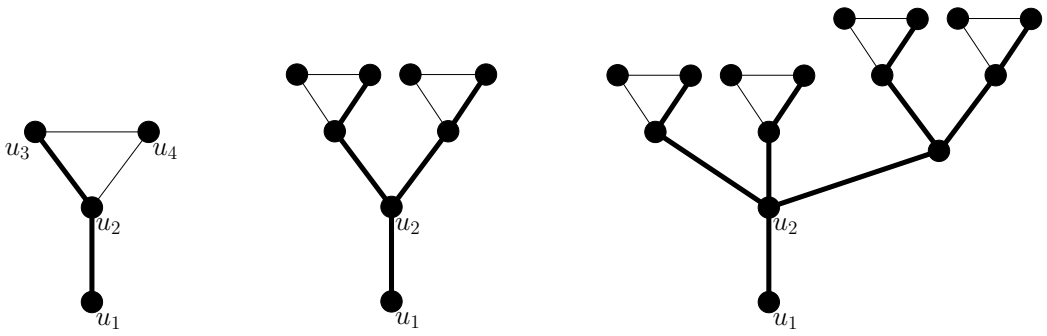
Proof. Assume w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of $G_F(C_i)$ such that $w(u_1u_2) = b$ and $w(u_3u_4) = a$. By Lemma 8.20, we have $w(u_2u_4) = b$, and $s_w(u_2) = a+3b$. Note then that we cannot have $w(u_4u_5) = w(u_4u_6) = b$ since otherwise we would get $s_w(u_4) = a+3b = s_w(u_2)$. Hence, we have either $\{w(u_4u_5), w(u_4u_6)\} = \{a, a\}$ or $\{w(u_4u_5), w(u_4u_6)\} = \{a, b\}$. ■

Proposition 8.24. *Assume $m(C_i) = 3$. If $0 \notin \{a, b\}$ and $b \neq -a$, then $G_F(C_i)$ satisfies Property 8.13 for $\{a, b\}$.*

Proof. Assume similarly that w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of $G_F(C_i)$ such that $w(u_1u_2) = b$ and $w(u_3u_5) = w(u_4u_5) = a$. Then we have $w(u_2u_5) = b$ and $s_w(u_2) = 2a + 4b$ according to Lemma 8.20 since the two graphs T_2 attached to u_2 form a graph T'_2. Now note that if $w(u_5u_6) = w(u_5u_7) = w(u_5u_8) = b$, then $s_w(u_5) = 2a + 4b = s_w(u_2)$. Thus, at least one of u_5u_6, u_5u_7 and u_5u_8 has weight a by w. ■

Collecting gadget $G^\curlyvee$ for $\{a, b\}$

The collecting gadget $G^\curlyvee$ for this main implementation is depicted in Figure 8.8. The two regular inputs of $G^\curlyvee$ are v_1v_3 and v_2v_3, and its output is v_3v_4. The edges u_1u_4, u_2u_4, u_5u_8, u_6u_8, u_9u_{11}, $u_{12}u_{15}$, $u_{13}u_{15}$, $u'_3u'_4$, $u'_7u'_8$, $u'_{10}u'_{11}$ and $u'_{14}u'_{15}$ of $G^\curlyvee$ are forcing inputs which are supposed to be weighted a. The edges u_3u_4, u_7u_8, $u_{10}u_{11}$, $u_{14}u_{15}$, $u'_1u'_4$, $u'_2u'_4$, $u'_5u'_8$, $u'_6u'_8$, $u'_9u'_{11}$, $u'_{12}u'_{15}$ and $u'_{13}u'_{15}$ are forcing inputs supposed to be weighted b.



(a) Y_1. (b) Y_2. (c) Y_3.

Figure 8.9: The graphs Y_1, Y_2 and Y_3, and neighbour-sum-distinguishing $\{a, 0\}$-edge-weightings of Y_1, Y_2 and Y_3. Thick (resp. thin) edges represent a- (resp. 0-) weighted edges.

Under our assumptions on a and b, we prove that G^Υ is a collecting gadget, i.e. it satisfies Property 8.14.

Proposition 8.25. *If $0 \notin \{a, b\}$ and $b \neq -a$, then G^Υ satisfies Property 8.14 for $\{a, b\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of G^Υ such that the forcing inputs of G^Υ are weighted as mentioned. Consider first the left side of G^Υ, i.e. the subgraph of G^Υ induced by the u_i's. Note first that we have $w(u_4u_{11}) \neq w(u_8u_{11})$ so that $s_w(u_4) \neq s_w(u_8)$. Now note that we cannot have $w(u_4u_8) = b$, since otherwise we would have $\{s_w(u_4), s_w(u_8)\} = \{2a + 3b, 3a + 2b\}$, and we would necessarily get $s_w(u_{11}) \in \{s_w(u_4), s_w(u_8)\}$ whatever is $w(u_{11}u_{15})$. Thus $w(u_4u_8) = a$, and $\{s_w(u_4), s_w(u_8)\} = \{4a + b, 3a + 2b\}$. Note now that we have $w(u_{11}u_{15}) = b$ since otherwise we would have $s_w(u_{11}) = 3a + 2b$. It follows that $s_w(u_{11}) = 2a+3b$, and we have $w(u_{15}v_3) = a$ since otherwise we would have $s_w(u_{15}) = 2a+3b = s_w(u_{11})$. So $s_w(u_{15}) = 3a + 2b$.

Due to the symmetric structure of G^Υ, we can deduce similar facts regarding the right side of G^Υ, i.e. the subgraph of G^Υ induced by the u'_i's. In particular, we have $w(v_3u'_{15}) = b$ and $s_w(u'_{15}) = 2a+3b$. Now observe that we cannot have $w(v_1v_3) \neq w(v_2v_3)$ since otherwise by having $w(v_3v_4) = a$ or $w(v_3v_4) = b$ we would get $s_w(v_3) = 3a + 2b = s_w(u_{15})$ or $s_w(v_3) = 2a + 3b = s_w(u'_{15})$, respectively. Therefore, we have $w(v_1v_3) = w(v_2v_3)$, and also $w(v_3v_4) = w(v_1v_3)$ since otherwise we would get $s_w(v_3) \in \{s_w(u_{15}), s_w(u'_{15})\}$. ■

8.4 Second implementation: $b = 0$

The second implementation of our reduction framework is dedicated to the case where one of the two weights from $\{a, b\}$ is 0. We assume throughout this section that $b = 0$.

Auxiliary gadget Y_k and replacement triplet for $\{a, 0\}$

Similarly as for the first implementation, we first give auxiliary graphs which are used in our gadgets to "force" the propagation of a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting.

Construction 8.26. The graphs Y_k with $k \geq 1$ are defined inductively. By construction, every graph Y_k has only one vertex u_1 with degree 1, called the *root* of Y_k. Start with an edge u_1u_2. To build Y_1, just identify u_2 and one vertex from a triangle. Now for the general case, i.e. $k \geq 2$, start over from the edge u_1u_2, and identify u_2 and the root of each of $k - 1$ copies of Y_1 and one copy of Y_{k-1}.

Example 8.27. The graphs Y_1, Y_2, and Y_3 are depicted in Figure 8.9.

In the figures of the following sections, every pendant triangle marked "Y_k" indicates that a vertex is identified with the root of a copy of Y_k. Every graph Y_k has the following weighting properties.

Lemma 8.28. *Let $k \geq 1$ be fixed. If w is a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of Y_k, then $w(u_1u_2) = a$ and $s_w(u_2) = (k + 1)a$.*

Proof. We prove this lemma by induction on k. Consider Y_1 first, and denote $u_2u_3u_4u_2$ the triangle attached to u_2. Note that if $w(u_2u_3) = w(u_3u_4)$, then $s_w(u_3) = s_w(u_4)$. Then $w(u_2u_3) = a$ and $w(u_2u_4) = 0$ without loss of generality, and $w(u_3u_4) = 0$ since otherwise we would have either $s_w(u_2) = s_w(u_3)$ or $s_w(u_2) = s_w(u_4)$ by setting $w(u_1u_2) = a$ or $w(u_1u_2) = 0$, respectively. Then $\{s_w(u_3), s_w(u_4)\} = \{0, a\}$, and we have $w(u_1u_2) = a$ since otherwise we would have $s_w(u_2) = a$. In particular, we have $s_w(u_2) = 2a$, as claimed.

Now suppose that the claim is true for every k up to an i, and consider $k = i + 1$. The graph Y_k is made of $k - 1$ copies of Y_1 and one copy of Y_{k-1} whose roots are identified with u_2. By the induction hypothesis, these copies are weighted by w in such a way that their respective edge incident with u_2 is weighted a, and the vertex from Y_{k-1} neighbouring u_2 has weighted degree ka. Thus, the weighting of the copies of Y_1 and Y_{k-1} provides ka in the weighted degree of u_2. We hence have $w(u_1u_2) = a$ so that $s_w(u_2) \neq ka$, and we get $s_w(u_2) = (k + 1)a$. This concludes the proof. ■

Now consider the following graph construction.

Construction 8.29. Let $k \geq 1$ be fixed, and let $P_3 = v_1v_2v_3$ be the path with length 2. As L_k, we refer to the graph obtained by identifying v_2 and the roots of k copies of the graph Y_k. The two inputs of L_k are the edges v_1v_2 and v_2v_3.

We show that L_k is a replacement gadget for $\{a, 0\}$.

Lemma 8.30. *Let $k \geq 1$ be fixed. Then L_k is a $((k + 2)a, ka)$-replacement gadget for $\{a, 0\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of L_k. By Lemma 8.28, the k copies of Y_k attached to v_2 provide ka to the weighted degree of v_2, and v_2 is adjacent to vertices with weighted degree $(k + 1)a$. Note then that if $\{w(v_1v_2), w(v_2v_3)\} = \{a, 0\}$, then the weighted degree of v_2 is $(k + 1)a$, and w is hence not neighbour-sum-distinguishing. On the contrary, if $w(v_1v_2) = w(v_2v_3) = a$ or $w(v_1v_2) = w(v_2v_3) = 0$, then the weighted degree of v_2 is $(k + 2)a$ or ka, respectively. ■

Corollary 8.31. *Let $i, j, k \geq 1$ be three distinct positive integers. Then (L_i, L_j, L_k) is a replacement triplet for $\{a, 0\}$.*

Spreading gadget $G^{\curlywedge}$ for $\{a, 0\}$

Consider, as $G^{\curlywedge}$, the graph depicted in Figure 8.10, whose input is u_1u_2, and whose two outputs are u_6u_7 and u_6u_8. We show that $G^{\curlywedge}$ is a spreading gadget for $\{a, 0\}$.

Proposition 8.32. *The graph $G^{\curlywedge}$ satisfies Property 8.12 for $\{a, 0\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of $G^{\curlywedge}$. Recall that the graphs Y_2, Y_3 and Y_4 attached to u_6 provide $3a$ in the weighted degree of u_6, and that u_6 is adjacent to vertices with weighted degree $3a$, $4a$, and $5a$ according to Lemma 8.28. Then we cannot have $\{w(u_4u_6), w(u_5u_6)\} = \{0, 0\}$ since otherwise we would have $s_w(u_6) \in \{3a, 4a, 5a\}$ whatever are $w(u_6u_7)$ and $w(u_6u_8)$. Observe also that if $\{w(u_4u_6), w(u_5u_6)\} = \{0, a\}$, then we have $w(u_6u_7) = w(u_6u_8) = a$. In this situation, we have $s_w(u_6) = 6a$.

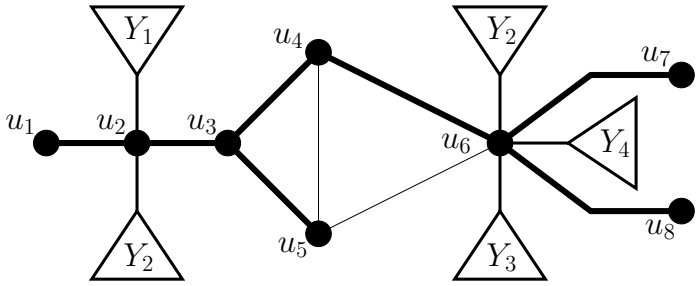
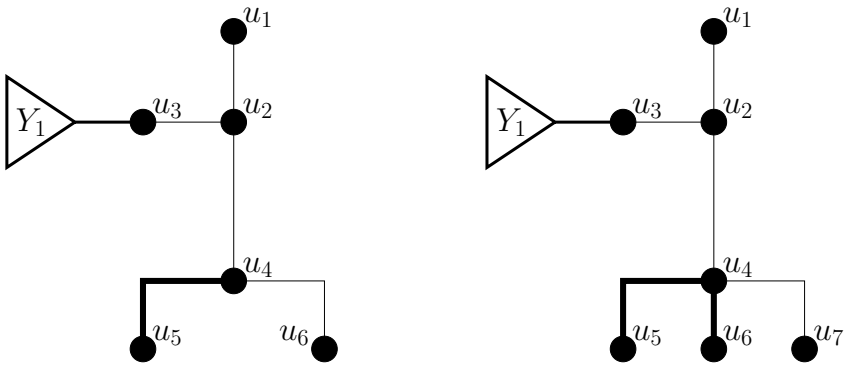


Figure 8.10: The spreading gadget $G^{\curlywedge}$ for the second implementation of the reduction framework, and a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of $G^{\curlywedge}$. Thick (resp. thin) edges represent a- (resp. 0-) weighted edges.



(a) Case $m(C_i) = 2$. (b) Case $m(C_i) = 3$.

Figure 8.11: The two forms of the clause gadget $G_F(C_i)$ for the second implementation of the reduction framework, and neighbour-sum-distinguishing $\{a, 0\}$-edge-weightings of $G_F(C_i)$. Thick (resp. thin) edges represent a- (resp. 0-) weighted edges.

Consider the edge u_1u_2. By Lemma 8.28, the weighted degree of u_2 is at least $2a$, and u_2 is adjacent to vertices whose weighted degrees are $2a$ and $3a$. Then we have $w(u_1u_2) = w(u_2u_3) = a$ since otherwise we would have $s_w(u_2) \in \{2a, 3a\}$. In particular, we have $s_w(u_2) = 4a$. Now note that we cannot have $w(u_3u_4) = w(u_3u_5) = 0$ since one of u_4 or u_5 would have weighted degree $s_w(u_3) = a$. Indeed, no matter what is the weight of u_4u_5, we have $\{w(u_4u_6), w(u_5u_6)\} = \{0, a\}$ so that $s_w(u_4) \neq s_w(u_5)$. But then, one of u_4 or u_5 necessarily gets weighted degree a, which is $s_w(u_3)$.

Suppose now $w(u_3u_4) = w(u_3u_5) = a$. In this situation, we have $s_w(u_3) = 3a$. Note that if $w(u_4u_5) = a$, then we have $\{w(u_4u_6), w(u_5u_6)\} = \{0, a\}$ so that u_4 and u_5 have distinct weighted degrees. But then, one of these two vertices has weighted degree $3a$. So $w(u_4u_5) = 0$. Once again, we have $\{w(u_4u_6), w(u_5u_6)\} = \{0, a\}$ so that u_4 and u_5 are distinguished. According to the remarks above, we then have $w(u_6u_7) = w(u_6u_8) = a$, as requested.

Suppose finally that $w(u_3u_4) = a$ and $w(u_3u_5) = 0$ without loss of generality. Then $s_w(u_3) = 2a$. Note that we cannot have $w(u_4u_5) = 0$ since otherwise we would have $w(u_4u_6) = 0$ so that $s_w(u_4) \neq s_w(u_3)$, and $w(u_5u_6) = 0$ so that $s_w(u_4) \neq s_w(u_5)$. But then $\{w(u_4u_6), w(u_5u_6)\} = \{0, 0\}$, and w is not neighbour-sum-distinguishing according to the early arguments above. Thus, $w(u_4u_5) = a$. Because $s_w(u_4) \neq s_w(u_3)$ and $s_w(u_5) \neq s_w(u_3)$, we have both $w(u_4u_6) = a$ and $w(u_5u_6) = 0$. According to the arguments above, we have $w(u_6u_7) = w(u_6u_8) = a$ once again. ■

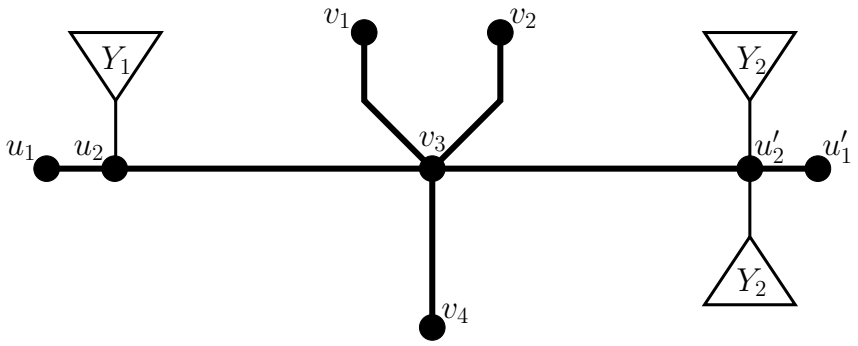


Figure 8.12: The collecting gadget $G^\curlyvee$ for the second implementation of the reduction framework, and a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of $G^\curlyvee$. Thick (resp. thin) edges represent a- (resp. 0-) weighted edges.

Clause gadgets $G_F(C_i)$ for $\{a, 0\}$

The two forms of $G_F(C_i)$ for $\{a, 0\}$, i.e. for the cases $m(C_i) = 2$ and $m(C_i) = 3$, are depicted in Figure 8.11. In both cases, the input of $G_F(C_i)$ is u_1u_2 and is supposed to be weighted 0. The outputs of $G_F(C_i)$ are u_4u_5, u_4u_6, and also u_4u_7 when $m(C_i) = 3$. We show that $G_F(C_i)$ satisfies Property 8.13 in both cases.

Proposition 8.33. *The graph $G_F(C_i)$ satisfies Property 8.13 for $\{a, 0\}$ whatever is the value of $m(C_i)$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of $G_F(C_i)$ such that $w(u_1u_2) = 0$. Recall that the edge of the graph Y_1 incident with u_3 has weight a, and that the vertex of the graph Y_1 adjacent with u_3 has weighted degree $2a$ (Lemma 8.28). Therefore, we have $w(u_3u_2) = 0$ so that $s_w(u_3) = a \neq 2a$, and $w(u_2u_4) = 0$ so that $s_w(u_2) \neq s_w(u_3)$. In particular, we get $s_w(u_2) = 0$. Then note that at least one of the outputs of $G_F(C_i)$ receives weight a by w since otherwise we would have $s_w(u_4) = 0 = s_w(u_2)$. ■

Collecting gadget $G^\curlyvee$ for $\{a, 0\}$

Now consider the graph depicted in Figure 8.12 as $G^\curlyvee$. The two regular inputs of $G^\curlyvee$ are v_1v_3 and v_2v_3, and its output is v_3v_4. The forcing inputs of $G^\curlyvee$ are u_1u_2 and $u'_1u'_2$, which are supposed to be weighted a. We prove that $G^\curlyvee$ satisfies Property 8.14 for $\{a, 0\}$.

Proposition 8.34. *The graph $G^\curlyvee$ satisfies Property 8.14 for $\{a, 0\}$.*

Proof. Suppose w is a neighbour-sum-distinguishing $\{a, 0\}$-edge-weighting of $G^\curlyvee$ such that we have $w(u_1u_2) = w(u'_1u'_2) = a$. Then we have $w(u_2v_3) = w(u'_2v_3) = a$ according to Lemma 8.30 since the copy of Y_1 attached to u_2 forms a graph L_1, while the two copies of Y_2 attached to u'_2 form a graph L_2. Plus, we have $s_w(u_2) = 3a$ and $s_w(u'_2) = 4a$. Under these assumptions, we cannot have $w(v_1v_3) \neq w(v_2v_3)$. Indeed, in such a situation, by having $w(v_3v_4) = a$ or $w(v_3v_4) = 0$, we would get $s_w(v_3) = 4a$ or $s_w(v_3) = 3a$, respectively.

Now suppose $w(v_1v_3) = w(v_2v_3)$. On the one hand, if $w(v_1v_3) = w(v_2v_3) = a$, then we have $w(v_3v_4) = a$ since otherwise we would get $s_w(v_3) = 4a = s_w(u'_2)$. In this situation, we get $s_w(v_3) = 5a$. On the other hand, suppose $w(v_1v_3) = w(v_2v_3) = 0$. Note that if $w(v_3v_4) = a$, then $s_w(v_3) = 3a = s_w(u_2)$. On the contrary, we have $s_w(v_3) = 2a$ when $w(v_3v_4) = 0$. ■

8.5 Third implementation: $b = -a$

In this section, we give gadgets for implementing our reduction framework in the case where $\{a, b\} = \{a, -a\}$.

Auxiliary gadget T and replacement triplet for $\{a, -a\}$

Once again, we use several gadgets to force the propagation of a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting in a graph. The first such gadget, denoted T, is just a triangle $u_1u_2u_3u_1$ whose vertex u_1 is the *root* of T. Hence, every triangle marked "T" in our figures of further sections refers to the graph T. This graph T has some interesting properties when dealing with neighbour-sum-distinguishing $\{a, -a\}$-edge-weightings.

Lemma 8.35. *If w is a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of T, then one of u_2 and u_3 has weighted degree 0, while the other vertex has weighted degree $2a$ or $-2a$. Besides, we have $\{w(u_1u_2), w(u_1u_3)\} = \{a, -a\}$.*

Proof. The proof is similar to the one of Lemma 8.17 since T is nothing but T_0. Because $s_w(u_2) \neq s_w(u_3)$, we have $w(u_1u_2) \neq w(u_1u_3)$. Suppose e.g. that $w(u_1u_2) = a$ and $w(u_1u_3) = -a$ without loss of generality. Now, by setting either $w(u_2u_3) = a$ or $w(u_2u_3) = -a$, we get $s_w(u_3) = 0$ or $s_w(u_2) = 0$, respectively. Besides, we have $s_w(u_2) = 2a$ or $s_w(u_3) = -2a$, respectively. ■

We now introduce the replacement gadgets for $\{a, -a\}$.

Construction 8.36. The first replacement gadget R_1 is obtained by identifying the root of T and v_2, where v_2 denotes the inner vertex of $P_3 = v_1v_2v_3$, the path with length 2. The two inputs of R_1 then are v_1v_2 and v_2v_3.

Lemma 8.37. *The graph R_1 is a $(2a, -2a)$-replacement gadget for $\{a, -a\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of R_1. According to Lemma 8.35, the weighted degrees of the vertices of T adjacent to v_2 are 0, and either $2a$ or $-2a$, where this last weighted degree can be "chosen" by reweighting of T locally. Besides, the weighting of T provides $a + (-a) = 0$ to the weighted degree of v_2. Note then that if $w(v_1v_2) \neq w(v_2v_3)$, then we have $s_w(v_2) = 0$. Hence, we have $w(v_1v_2) = w(v_2v_3)$, and $s_w(v_2) = 2a$ or $s_w(v_2) = -2a$ depending on whether $w(v_1v_2) = a$ or $w(v_1v_2) = -a$, respectively. ■

The second replacement gadget R_2 for $\{a, -a\}$ is obtained as follows.

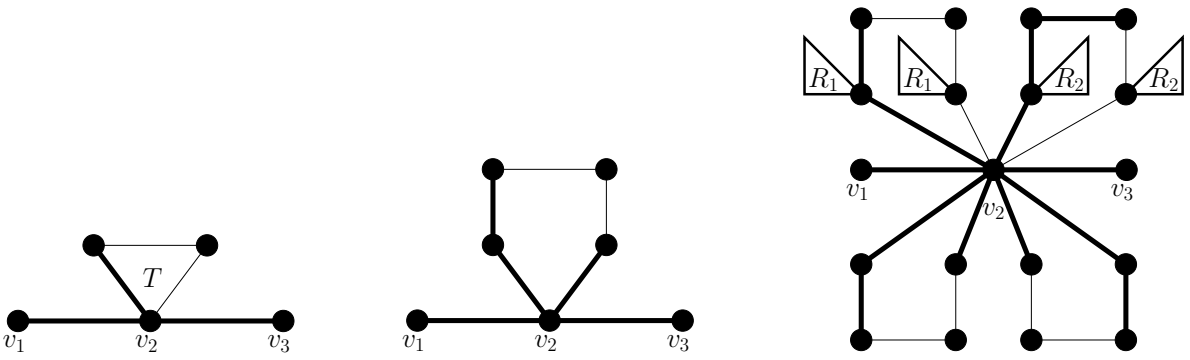
Construction 8.38. As for R_1, start from the path $P_3 = v_1v_2v_3$ with length 2, and identify v_2 and u_1, where $u_1u_2u_3u_4u_5u_1$ is a cycle with length 5. The inputs of R_2 are v_1v_2 and v_2v_3.

Lemma 8.39. *The graph R_2 is a $(4a, -4a)$-replacement gadget for $\{a, -a\}$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of R_2. Note first that $w(u_1u_2) = w(u_5u_1)$ according to Observation 8.4. Besides, we have $\{s_w(u_2), s_w(u_5)\} = \{0, 2w(u_1u_2)\}$ and the weighting of the cycle attached to v_2 provides $2w(u_1u_2)$ to the weighted degree of v_2. Suppose now that $w(u_1u_2) = w(u_5u_1) = a$. Then note that if $w(v_1v_2) = w(v_2v_3) = -a$ or $w(v_1v_2) \neq w(v_2v_3)$, then we have $s_w(v_2) = 0$ or $s_w(v_2) = 2a$, and w cannot be neighbour-sum-distinguishing. Hence $w(v_1v_2) = w(v_2v_3) = a$ and $s_w(v_2) = 4a$ in this situation. The proof follows similarly from the assumption $w(u_1u_2) = w(u_5u_1) = -a$. ■

The other replacement gadgets for $\{a, -a\}$ are defined inductively.

Construction 8.40. To obtain the graph R_k for some $k \geq 3$, start from the path $P_3 = v_1v_2v_3$ with length 2. Next identify v_2 and the root of each of $k - 1$ copies of the graph T. For every such ith resulting copy $v_2u_2u_3v_2$ of T, with $i \in \{1, 2, ..., k - 1\}$, now R_i-subdivide each of the edges v_2u_2 and v_2u_3. This results in a cycle $s_1s_2s_3s_4s_5s_1$ with length 5 such that $s_1 = v_2$, and the edges s_1s_2 and s_2s_3, and s_1s_5 and s_5s_4 are the inputs of two replacement gadgets R_i. To finish the construction of R_k, identify v_2 and one vertex of each of $k - 1$ cycles with length 5. The inputs of R_k are v_1v_2 and v_2v_3.



(a) R_1. (b) R_2. (c) R_3.

Figure 8.13: The graphs R_1, R_2 and R_3, and neighbour-sum-distinguishing $\{a, -a\}$-edge-weightings of R_1, R_2 and R_3. Thick (resp. thin) edges represent a- (resp. $(-a)$-) weighted edges.

Example 8.41. The graphs R_1, R_2, and R_3 are depicted in Figure 8.13.

Lemma 8.42. *Let $k \geq 3$ be fixed. Then R_k is a $(2ka, -2ka)$-replacement gadget for $\{a, -a\}$.*

Proof. Assume the claim is true for every k up to a value of i, and consider $k = i + 1$. Let w be a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of R_k. Consider first every cycle $v_2s_2s_3s_4s_5v_2$ with length 5 such that the edges v_2s_2 and s_2s_3, and s_4s_5 and s_5v_2 are the inputs of two graphs $R_{k'}$, with $k' < k$. Note that we cannot have $w(s_2s_3) = w(s_4s_5)$ since otherwise we would have $s_w(s_3) = s_w(s_4)$. Thus we have $w(s_2s_3) = a$ and $w(s_4s_5) = -a$ without loss of generality, and $w(v_2s_2) = a$ and $w(v_2s_5) = -a$ according to the induction hypothesis. Besides, $s_w(s_2) = 2k'a$ and $s_w(s_5) = -2k'a$. Hence, the $k-1$ cycles of this form attached to v_2 provide $(k-1)(a+(-a)) = 0$ in the weighted degree of v_2, and v_2 is adjacent to vertices whose weighted degrees lie among

$$\{-2(k-1)a, -2(k-2)a, ..., -2a, 2a, ..., 2(k-2)a, 2(k-1)a\}.$$

Now consider every "regular" cycle $v_2s_2s_3s_4s_5v_2$ with length 5 attached to v_2. For the same reasons as those given in the proof of Lemma 8.39, we have $w(v_2s_2) = w(v_2s_5)$, and $0 \in \{s_w(s_2), s_w(s_5)\}$. Hence, every regular cycle provides either $2a$ or $-2a$ in the weighted degree of v_2. Is is then easy to check that the only way for w to be neighbour-sum-distinguishing is to have each of the $k-1$ regular cycles providing $2a$ (resp. $-2a$) to the weighted degree of v_2, and $w(v_1v_2) = w(v_2v_3) = a$ (resp. $w(v_1v_2) = w(v_2v_3) = -a$). In this situation, we get $s_w(v_2) = 2ka$ (resp. $s_w(v_2) = -2ka$). For every other possible weighting, we necessarily get that $s_w(v_2)$ is equal to a value among $\{-2(k-1)a, -2(k-2)a, ..., -2a, 0, 2a, ..., 2(k-2)a, 2(k-1)a\}$. ■

Corollary 8.43. *Let $i, j, k \geq 1$ be three distinct positive integers. Then (R_i, R_j, R_k) is a replacement triplet for $\{a, -a\}$.*

Spreading gadget $G^{\curlywedge}$ for $\{a, -a\}$

The spreading gadget $G^{\curlywedge}$ for $\{a, -a\}$ is depicted in Figure 8.14. The input of $G^{\curlywedge}$ is u_1u_2, while its outputs are u_9u_{10} and $u_{12}u_{13}$. We prove that $G^{\curlywedge}$ satisfies the spreading gadget property.

Proposition 8.44. *The graph $G^{\curlywedge}$ satisfies Property 8.12 for $\{a, -a\}$.*

Proof. Suppose w is a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of $G^{\curlywedge}$. Note first that we cannot have $w(u_3u_5) \neq w(u_4u_6)$. Indeed, suppose e.g. that $w(u_3u_5) = a$ and $w(u_4u_6) = -a$. Then $w(u_5u_7) = a$ and $w(u_6u_7) = -a$ according to Lemma 8.37. Besides, $s_w(u_5) = 2a$

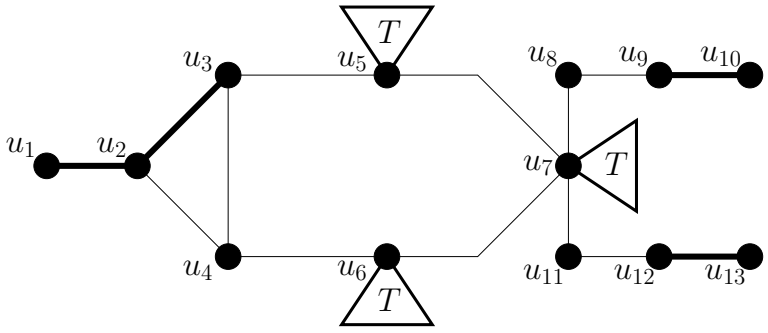


Figure 8.14: The spreading gadget $G^\curlywedge$ for the third implementation of the reduction framework, and a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of $G^\curlywedge$. Thick (resp. thin) edges represent a- (resp. $(-a)$-) weighted edges.

and $s_w(u_6) = -2a$. Note further that the weighting of the copy of T attached to u_7 provides $a + (-a) = 0$ in the weighted degree of u_7, and that u_7 has a neighbour with weighted degree 0 (Lemma 8.35). Then note that for every value of $\{w(u_7u_8), w(u_7u_{11})\}$, i.e. $\{a, a\}$, $\{a, -a\}$ or $\{-a, -a\}$, we get that $s_w(u_7)$ is either $2a$, 0 or $-2a$, respectively. Hence w is not neighbour-sum-distinguishing under the assumption $w(u_3u_5) \neq w(u_4u_6)$.

On the contrary, note that if $w(u_3u_5) = w(u_4u_6) = -a$, then w can be neighbour-sum-distinguishing. Note first that we have $w(u_5u_7) = w(u_6u_7) = -a$ according to Lemma 8.37. Besides, $s_w(u_5) = s_w(u_6) = -2a$. Recall that u_7 has a neighbour with weighted degree 0, and that the graph T attached to u_7 provides 0 to the weighted degree of u_7. Now note that if $\{w(u_7u_8), w(u_7u_{11})\}$ is $\{a, a\}$ or $\{a, -a\}$, then we have $s_w(u_7) = 0$ or $s_w(u_7) = -2a$, respectively. On the contrary, if $w(u_7u_8) = w(u_7u_{11}) = -a$, then we get $s_w(u_7) = -4a$. Besides, we have $w(u_9u_{10}) = w(u_{12}u_{13}) = a$ according to Observation 8.4.

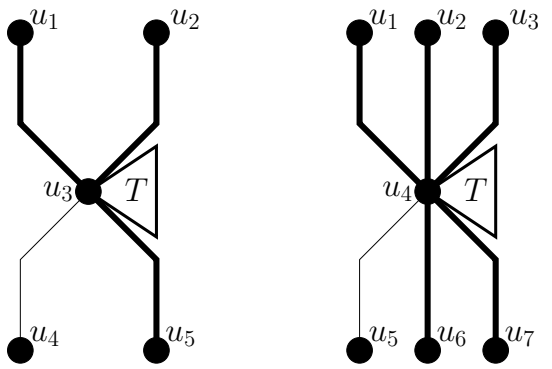
Now assume $w(u_1u_2) = a$. First, note that we cannot have $w(u_2u_3) = w(u_2u_4)$. Indeed, in this situation, we would have $w(u_3u_5) \neq w(u_4u_6)$ so that u_3 and u_4 have distinct weighted degrees, and this whatever is $w(u_3u_4)$. But according to the arguments above w cannot be neighbour-sum-distinguishing under this assumption. Then, $w(u_2u_3) = a$ and $w(u_2u_4) = -a$ without loss of generality. In this situation, $s_w(u_2) = a$. On the one hand, if $w(u_3u_4) = a$, then w cannot be neighbour-sum-distinguishing. Indeed, we would have $w(u_3u_5) = a$ so that $s_w(u_3) \neq s_w(u_2)$, and $w(u_4u_6) = -a$ so that $s_w(u_4) \neq s_w(u_2)$. But then $w(u_3u_5) \neq w(u_4u_6)$, and w is not neighbour-sum-distinguishing, again according to the arguments above. On the other hand, i.e. $w(u_3u_4) = -a$, then we have $w(u_3u_5) = -a$ so that $s_w(u_2) \neq s_w(u_3)$, and $w(u_4u_6) = -a$ so that $s_w(u_3) \neq s_w(u_4)$. As pointed out above, the weighting propagates along $G^\curlywedge$ in such a way that we have $w(u_9u_{10}) = w(u_{12}u_{13}) = a$, as requested. ■

Clause gadgets $G_F(C_i)$ for $\{a, -a\}$

Consider, as $G_F(C_i)$, the graphs depicted in Figure 8.15. In the first (resp. second) form, i.e. for $m(C_i) = 2$ (resp. $m(C_i) = 3$), the inputs of $G_F(C_i)$ are u_1u_3 and u_2u_3 (resp u_1u_4, u_2u_4 and u_3u_4) and are supposed to be weighted a. The outputs of $G_F(C_i)$ are u_3u_4 and u_3u_5 (resp. u_4u_5, u_4u_6 and u_4u_7). We show that the two forms of $G_F(C_i)$ are clause gadgets for $\{a, -a\}$.

Proposition 8.45. *The graph $G_F(C_i)$ satisfies Property 8.13 for $\{a, -a\}$ whatever is the value of $m(C_i)$.*

Proof. Assume w is a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of $G_F(C_i)$ such that all the inputs of $G_F(C_i)$ are weighted a. We show the claim to be true when $m(C_i) = 2$, but the proof is similar for the case $m(C_i) = 3$. Recall that the graph T attached to u_3 provides $a + (-a) = 0$ to the weighted degree of u_3, and that one of its vertices has weighted degree 0



(a) Case $m(C_i) = 2$.

(b) Case $m(C_i) = 3$.

Figure 8.15: The two forms of the clause gadget $G_F(C_i)$ for the third implementation of the reduction framework, and neighbour-sum-distinguishing $\{a, -a\}$-edge-weightings of $G_F(C_i)$. Thick (resp. thin) edges represent a- (resp. $(-a)$-) weighted edges.

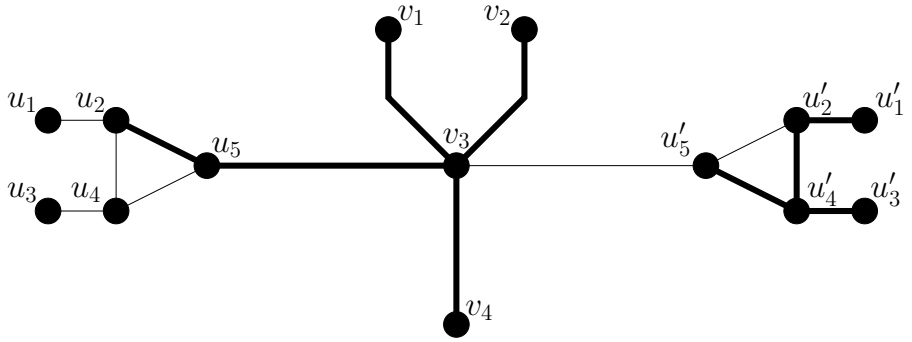


Figure 8.16: The collecting gadget $G^\curlyvee$ for the third implementation of the reduction framework, and a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of $G^\curlyvee$. Thick (resp. thin) edges represent a- (resp. $(-a)$-) weighted edges.

(Lemma 8.35). Note then that if $w(u_3u_4) = w(u_3u_5) = -a$, then $s_w(u_3) = 0$. Therefore, at least one output of $G_F(C_i)$ receives weight a by w. ■

Collecting gadget $G^\curlyvee$ for $\{a, -a\}$

Let $G^\curlyvee$ be the graph depicted in Figure 8.16. The two regular inputs of $G^\curlyvee$ are v_1v_3 and v_2v_3, while its output is v_3v_4. The forcing inputs of $G^\curlyvee$ are u_1u_2 and u_3u_4, which are supposed to be weighted $-a$, and $u_1'u_2'$ and $u_3'u_4'$ which are supposed to be weighted a. We show that $G^\curlyvee$ is a collecting gadget for $\{a, -a\}$.

Proposition 8.46. *The graph $G^\curlyvee$ satisfies Property 8.14 for $\{a, -a\}$.*

Proof. Suppose w is a neighbour-sum-distinguishing $\{a, -a\}$-edge-weighting of $G^\curlyvee$ such that $w(u_1u_2) = w(u_3u_4) = -a$ and $w(u_1'u_2') = w(u_3'u_4') = a$. Note that we cannot have $w(u_2u_4) = a$. Indeed, in this situation, we would have $w(u_2u_5) \neq w(u_4u_5)$ so that $s_w(u_2) \neq s_w(u_4)$. But then we would get $\{s_w(u_2), s_w(u_4)\} = \{a, -a\}$ and we would have $s_w(u_5) \in \{a, -a\}$ no matter what is $w(u_5v_3)$. Therefore $w(u_2u_4) = -a$. For the same reasons, we have $w(u_2u_5) = a$ and $w(u_4u_5) = -a$ without loss of generality. Then, $s_w(u_2) = -a$ and $s_w(u_4) = -3a$, and we have $w(u_5v_3) = a$ since otherwise we would have $s_w(u_5) = -a$. Besides, we have $s_w(u_5) = a$.

Repeating the same arguments regarding the subgraph induced by $\{u_1', u_2', u_3', u_4', u_5', v_3\}$, we get that $w(u_5'v_3) = -a$ and $s_w(u_5') = -a$. Therefore, the edges u_5v_3 and $u_5'v_3$ provide $a+(-a) = 0$

to the weighted degree of v_3, and v_3 is adjacent to vertices with respective weighted degree a and $-a$. Now observe that we cannot have $w(v_1v_3) \neq w(v_2v_3)$. Indeed, by then having $w(v_3v_4) = a$ or $w(v_3v_4) = -a$, we would get $s_w(v_3) = a$ or $s_w(v_3) = -a$, respectively. On the contrary, if $w(v_1v_3) = w(v_2v_3) = a$ (resp. $w(v_1v_3) = w(v_2v_3) = -a$), then we have $w(v_3v_4) = a$ (resp. $w(v_3v_4) = -a$) since otherwise we would have $s_w(v_3) = a$ (resp. $s_w(v_3) = -a$). In particular, we get $s_w(v_3) = 3a$ (resp. $s_w(v_3) = -3a$). ■

8.6 Conclusion and open questions

In this chapter, we have provided another proof that Neighbour-Sum-Distinguishing $\{a,b\}$-Edge-Weighting remains NP-complete for all values of $\{a, b\}$ involving real weights. Surely the most interesting notion we have introduced in this chapter is the notion of replacement gadget, which we believe could be of some use regarding some of the open questions related to neighbour-sum-distinguishing $\{a, b\}$-edge-weighting of graphs mentioned in Section 7.3. In particular, we think it could help to exhibit new bipartite graphs with no neighbour-sum-distinguishing $\{1, 2\}$-edge-weighting. This approach could be pertinent in order to deal with Problem 7.14.

Speaking of Problem 7.14, it is not clear whether the existence of a fair characterization of bipartite graphs admitting a neighbour-sum-distinguishing $\{1, 2\}$-edge-weighting would imply the existence of a fair characterization of bipartite graphs admitting a neighbour-sum-distinguishing $\{a, b\}$-edge-weighting for every $\{a, b\} \neq \{1, 2\}$. So Problem 7.14 can be refined to the following.

Question 8.47. *Are there two distinct real numbers a and b such that* Neighbour-Sum-Distinguishing $\{a,b\}$-Edge-Weighting *remains NP-complete when restricted to bipartite graphs?*

The different gadgets we have had to exhibit throughout this chapter for distinct values of $\{a, b\}$ are a good illustration of how much different two problems Neighbour-Sum-Distinguishing $\{a,b\}$-Edge-Weighting and Neighbour-Sum-Distinguishing $\{a',b'\}$-Edge-Weighting can be, and hence justifies Question 8.47. Regarding this question, we believe $\{a, b\} = \{a, 0\}$ and $\{a, b\} = \{a, -a\}$ are two appealing cases. It is however important mentioning that each of our three implementations involves some gadgets which are not bipartite, and hence which cannot be used directly for dealing with Question 8.47. So one first important task would be to investigate whether some bipartite gadgets with similar properties as ours can be designed for specific values of $\{a, b\}$. The existence or the non-existence of such gadgets would be a good hint on the actual status of Question 8.47.

Chapter 9
Locally irregular edge-colouring of graphs

This chapter is devoted to the study of locally irregular edge-colouring of graphs. As many graphs do not admit any locally irregular edge-colouring, e.g. paths or cycles of odd length, we start our investigations, in Section 9.1, by clarifying the range of colourable graphs for this type of edge-colouring. For this purpose, we give a concrete characterization of exceptions in Section 9.1.1 before eventually showing in Section 9.1.2 that every non-exception graph is indeed colourable.

In Section 9.2, we raise a conjecture on the maximum number of colours needed to obtain a locally irregular edge-colouring of every colourable graph. Namely, we believe that all colourable graphs have irregular chromatic index at most 3. We verify this conjecture for various classes of graphs in Section 9.2.1, including trees, complete graphs, and Cartesian products of colourable graphs verifying the conjecture themselves. Using of a probabilistic approach, we also support our conjecture by showing, in Section 9.2.2, that regular graphs with sufficiently large degree agree it. This result also has consequences on the 1-2-3 Conjecture, recall Observation 7.37.

Section 9.3 is dedicated to the study of the algorithmic hardness of determining the irregular chromatic index of a graph. Since the notion of exception is important in this context, we first show in Section 9.3.1 that recognizing whether a graph is an exception can be done in polynomial time. We then focus on the complexity of Locally Irregular 2-Edge-Colouring. We show this problem to be handleable in linear time when restricted to trees (Section 9.3.2), before showing it to be NP-complete in general (Section 9.3.3).

The proof of Theorem 9.44 was presented as a poster at the EuroComb 2013 Conference [28], and is, as well as the results from Section 9.3.2, part of a joint work with Baudon and Sopena submitted for publication [20]. All results from Sections 9.1 and 9.2 were obtained jointly with Baudon, Przybyło and Woźniak, and have been submitted for publication [17].

9.1 Decomposing graphs into locally irregular subgraphs

We start by characterizing, in Section 9.1.1, those exception graphs which do not admit any locally irregular edge-colouring. Every other graph is then shown to be colourable in Section 9.1.2.

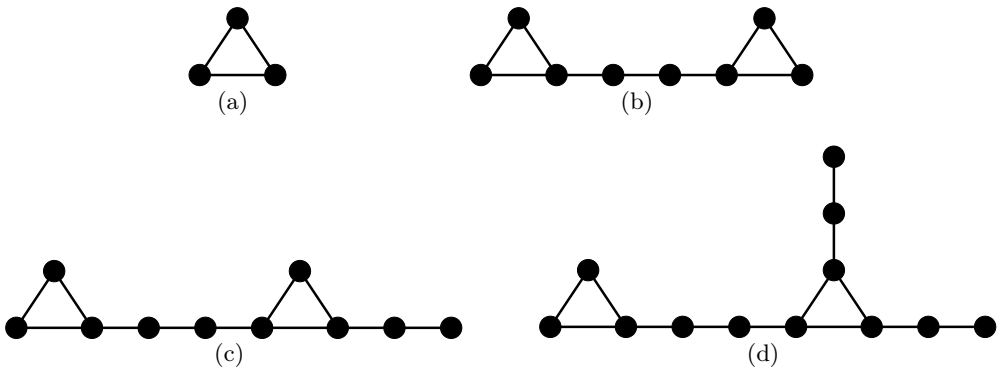


Figure 9.1: Some members of $\mathcal{T}$.

9.1.1 Characterization of exceptions

Although locally irregular edge-colouring of graphs was initially inspired by neighbour-sum-distinguishing edge-weighting of graphs, the range of exceptions is much wider for the first notion than for the second one. Namely, an exception for the notion of locally irregular edge-colouring is either a path or cycle with odd length, or a member from the family $\mathcal{T}$ described below (while, for other related problems, only K_2 is an exception).

We start by characterizing paths and cycles which are exceptions. An easy proof relies on the following straightforward observation.

Observation 9.1. *The only locally irregular non-trivial path is P_3.*

Proof. Just note that P_2 has its two vertices having degree 1, while P_n has neighbouring vertices with degree 2 for every $n \geq 4$. So these graphs are not locally irregular. On the contrary, P_3 is locally irregular since its consecutive vertices have degree 1, 2 and 1, respectively, from one endvertex to the other one. ■

Corollary 9.2. *A path or cycle is an exception if and only if it has odd length.*

Proof. Recall that a non-trivial path P_n is locally irregular if and only if $n = 3$ (see Observation 9.1), and note that no cycle is locally irregular. Therefore, we have to use at least two colours to produce a locally irregular edge-colouring c of every cycle or path with order at least 4. Since the only non-trivial subgraphs of every path or cycle are paths, each colour of c induces a forest of P_3. Colourable paths and cycles thus necessarily have even length. In particular, a locally irregular edge-colouring of every colourable path or cycle can be obtained by colouring every two consecutive adjacent edges with a new colour, starting from one endedge for a path. ■

We now introduce the last family $\mathcal{T}$ of exceptions.

Construction 9.3. First, the triangle K_3 belongs to $\mathcal{T}$. Another member G' of $\mathcal{T}$ can then be obtained as follows. On the one hand, let G be a member of $\mathcal{T}$ in which at least one vertex, say u, belongs to a triangle of G and has degree 2. On the other hand, let v be either an endvertex of a path with even length, or the endvertex of a path with odd length whose other endvertex is identified with a vertex of a new triangle. We obtain G' by identifying u and v.

Example 9.4. The construction of successive members of $\mathcal{T}$ is depicted in Figure 9.1.

It is worth noting that members of $\mathcal{T}$ have odd size, maximum degree at most 3, and circumference 3. Besides, these graphs have a tree-like structure consisting of triangles and pendant vertices, and paths with specific lengths connecting them. The recognition of these exceptions can then be done easily, as pointed out in Section 9.3.1.

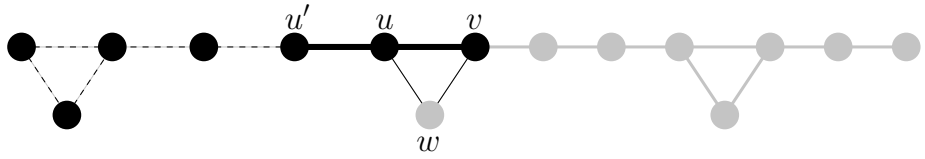


Figure 9.2: Argument used in the proof of Proposition 9.5. A locally irregular edge-colouring of a member of $\mathcal{T}$ (induced by the black and grey vertices) induces a locally irregular edge-colouring of a smaller member of $\mathcal{T}$ (induced by the black vertices only). Thick edges represent represent 1-coloured edges. No supposition is made on the colour of the dashed edges.

Proposition 9.5. *Members of $\mathcal{T}$ are exceptions.*

Proof. Suppose G is a member of $\mathcal{T}$ which is a minimal counterexample to the claim in terms of size, i.e. G is not an exception, but every member of $\mathcal{T}$ with fewer edges than G is an exception. Let c be a locally irregular edge-colouring of G, and consider a triangle $uvwu$ of G. Clearly $c(uv)$, $c(vw)$ and $c(wu)$ cannot be all equal, say to 1, since otherwise at least two of u, v and w would be neighbouring vertices with the same degree in the 1-subgraph, a contradiction. One colour of c is then uniquely used to colour the edges uv, vw and wu. We may assume that $c(uv) = 1$, $c(vw) \neq 1$ and $c(wu) \neq 1$ without loss of generality. Because K_2 is not locally irregular, one of the vertices incident with uv, say u, is necessarily incident with another edge uu' coloured 1 by c, where $u' \notin \{v, w\}$.

Denote by $G_{u'}$ the component of $G-\{u\}$ which contains u', and note that the subgraph G' of G induced by $V(G_{u'}) \cup \{u, v\}$ is either a path with odd length or a member of $\mathcal{T}$. Note then that even if v is incident with an edge vv' with $v' \notin \{u, w\}$, we cannot have $c(vv') = 1$ since otherwise the 1-subgraph would include two adjacent vertices having the same degree. Then c induces a locally irregular edge-colouring of G', contradicting either the minimality of G, or Corollary 9.2, see Figure 9.2. ■

9.1.2 Non-exception graphs are colourable

We now show that every graph which is not one of the exceptions exhibited in Section 9.1.1, i.e. a path or cycle with odd length, or a member of $\mathcal{T}$, admits a locally irregular edge-colouring. Our proof relies on the following lemma.

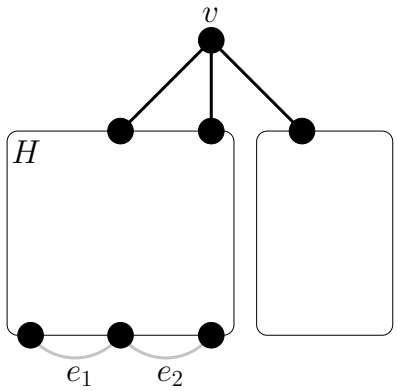
Lemma 9.6. *Let G be a connected graph whose edge set can be partitioned into two parts $O \cup I$ inducing connected subgraphs of G. If $|O| \geq 2$, then there are two adjacent edges e_1 and e_2 of O such that $G - \{e_1, e_2\}$ is connected.*

Proof. We only need to show that we can modify, if necessary, the sets O and I so that they retain the properties of the claim, but O consists of exactly two edges. For this purpose, we repeat the following procedure until $|O| = 2$ holds.

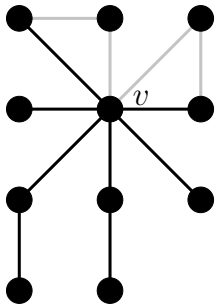
Denote by G_O and G_I the subgraphs of G induced by O and I, respectively. Since G is connected, there exists a vertex v of G_O which is incident to some edges of I. Let $G_1, G_2, ..., G_k$ be the components of $G_O - \{v\}$, and denote by H_i the subgraph $G_i + \{v\}$ for every $i \in \{1, 2, ..., k\}$. If $k \geq 2$ and one H_i contains at least two edges, then we move all the edges of $O \setminus E(H_i)$ to I. Otherwise, we choose any edge of O incident with v and move it to I.

Since in each step we decrease the size of O, we eventually get O consisting of only two adjacent edges. ■

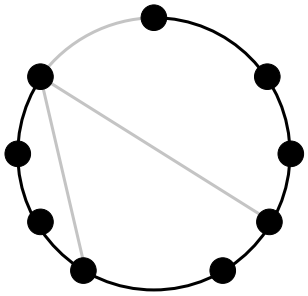
Theorem 9.7. *Every non-exception connected graph admits a locally irregular edge-colouring.*



(a) Case where $d(v) = 3$ and the neighbourhood of v is independent, and $G-\{v\}$ has a component H with size at least 2. Two adjacent edges e_1 and e_2 (in grey only) of H are removed so that $G - \{e_1, e_2\}$ is not an exception.



(b) Case where $d(v) \geq 4$ and all components of $G - \{v\}$ have size at most 1. Pairs of adjacent edges (in grey only) are removed so that the remaining graph is a non-exception tree.



(c) Case where G is spanned by a cycle with odd length. Edges (in grey only) inducing a tree with maximum degree at least 3 are removed from G so that the remaining graph is a non-exception path.

Figure 9.3: Situations described in the proof of Theorem 9.7.

Proof. Let G be a graph which is not an exception. We prove the claim by induction on the size of G. As a base case, one can easily check that G admits a locally irregular edge-colouring whenever it has small size, e.g. up to 4 edges. Suppose then the claim is true whenever G has up to to $m-1$ edges, and assume G has size m.

Note first that if G has a vertex v such that either $d(v) = 3$ and $N(v)$ is an independent set, or $d(v) \geq 4$, then we can deduce a locally irregular edge-colouring of G as follows. On the one hand, if a component H of $G - \{v\}$ has size at least 2, then we may remove two adjacent edges e_1 and e_2 from $E(H)$ so that $G' = G - \{e_1, e_2\}$ remains connected. The existence of these two edges is guaranteed by Lemma 9.6, basically by initially setting $O = E(H)$ and $I = E(G) \setminus O$, see Figure 9.3.a. Since the neighbourhood of v in G' retains the properties it had in G, the graph G' cannot be an exception and thus admits a locally irregular edge-colouring according to the induction hypothesis. By then colouring e_1 and e_2 with a new colour we obtain a locally irregular edge-colouring of G.

On the other hand, assume every component of $G - \{v\}$ has size at most 1. A locally irregular edge-colouring of G is then obtained as follows. For every edge e_1 whose two ends are adjacent with v, let e_2 be an edge adjacent to e_1 and incident with v, and colour e_1 and e_2 with a new colour. Once no more such edges exist, let G' denote the subgraph of G induced by all uncoloured edges. Then G' is a tree different from a path with odd length (this follows from our assumptions on $d(v)$ and $N(v)$), and admits a locally irregular edge-colouring according to the induction hypothesis (see Figure 9.3.b). The coloured pairs of adjacent edges removed from G and the edge-colouring of G' then perform a locally irregular edge-colouring of G.

We thus now suppose that $\Delta(G) \leq 3$ and every degree-3 vertex of G belongs to some triangles. We may additionally suppose that G has circumference 3. Indeed, assume C is a cycle with length at least 4 in G. If a component H of $G - E(C)$ is of size at least 2, then, analogously as above, we may find two adjacent edges e_1 and e_2 from $E(G) \setminus E(C)$, again by using Lemma 9.6 (with O being $E(H)$), so that $G' = G - \{e_1, e_2\}$ is connected. Again, by starting from a locally irregular edge-colouring of G', which exists by the induction hypothesis unless $G' = C$ and C is a cycle with odd length, and then colouring e_1 and e_2 with a new colour, we obtain a locally irregular edge-colouring of G. In the particular case where $G' = C$ and C is a cycle with odd length, then

G consists in one cycle with odd length to which are attached e_1 and e_2, see Figure 9.3.c. In such a situation, one can easily colour e_1, e_2 and some additional edges of C with a same colour α so that the α-subgraph is a tree with maximum degree 3, and the remaining graph is a path with even length. According to Corollary 9.2, such a path admits a locally irregular edge-colouring. Together with the previously coloured edges, this performs a locally irregular edge-colouring of G. Using the same technique, we can deduce similar decompositions when all components of $G - E(C)$ are of size at most 1.

Then G has every of its vertices with degree 3 belonging to exactly one triangle since otherwise we would either have $\delta(G) \geq 4$ or a cycle with length at least 4 in G (typically when two triangles share an edge). Note that the structure of G is actually quite similar to the one of the members of $\mathcal{T}$, except for the lengths of the maximal paths of G induced by its vertices with degree 2. Now if $\delta(G) = 1$, then let u be a vertex incident with a pendant vertex v. Let e_1 be the edge uv. If $d(u) = 2$, then let e_2 be the other edge incident with u. Otherwise, let e_2 be an edge incident with u, different from e_1, and whose other end has smallest degree among those vertices in $N(u) \setminus \{v\}$. Otherwise, i.e. $\delta(G) \geq 2$, there must be, in G, a triangle with at least two vertices v and w of degree 2 (because of the assumption on the circumference of G), so let e_1 and e_2 be two adjacent edges incident with v and w. In both cases, note that $G - \{e_1, e_2\}$ cannot be an exception since otherwise G would be an exception too. Then we can first consider a locally irregular edge-colouring of $G - \{e_1, e_2\}$, which exists according to the induction hypothesis, and colour e_1 and e_2 with a new colour. This results in a locally irregular edge-colouring of G, completing the proof. ■

Using the term "colourable" to designate a graph which is not an exception (regarding locally irregular edge-colouring of graphs) hence now makes sense. Since Theorem 9.7 only deals with the existence of a locally irregular edge-colouring of every colourable graph, the number of colours used in the proof is clearly not optimal. In particular, every colour of a resulting locally irregular edge-colouring induces a connected graph, while a locally irregular graph can consist in several locally irregular vertex-disjoint components. Since the smallest locally irregular connected non-trivial graph, in terms of size, is P_3, the number of colours used by a locally irregular edge-colouring of a graph G obtained in the proof of Theorem 9.7 is roughly upper-bounded by $\lfloor \frac{|E(G)|}{2} \rfloor$.

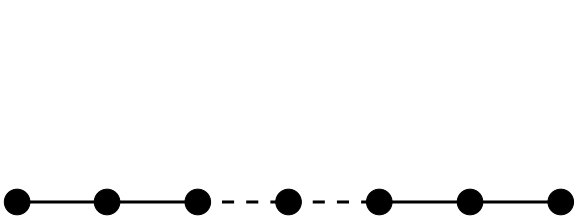
Corollary 9.8. *For every colourable graph G, we have $\chi'_{irr}(G) \leq \lfloor \frac{|E(G)|}{2} \rfloor$.*

9.2 Families with irregular chromatic index at most 3

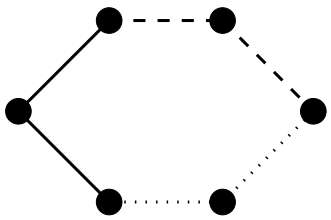
The upper bound on the irregular chromatic index of every colourable graph exhibited in Corollary 9.8 is only informative and far from being optimal. Studies on common families of graphs suggest that an optimal upper bound would rather be a constant integer depending of no graph parameter. Namely, only three colours seem sufficient to obtain a locally irregular edge-colouring of every colourable graph at first glance. We thus conjecture the following.

Conjecture 9.9. *For every colourable graph G, we have $\chi'_{irr}(G) \leq 3$.*

Throughout this section, we support Conjecture 9.9 by showing it to hold for numerous classes of colourable graphs. Namely, we verify it, in Section 9.2.1, for paths, cycles, some bipartite graphs including trees, complete graphs, and Cartesian products of graphs with irregular chromatic index at most 3. A more significant result in Section 9.2.2 states that Conjecture 9.9 is also true for regular graphs with sufficiently large degree.



(a) A locally irregular 2-edge-colouring of P_7. Solid and dashed edges represent 1- and 2-coloured edges, respectively.



(b) A locally irregular 3-edge-colouring of C_6. Solid, dashed, and dotted edges represent 1-, 2-, and 3-coloured edges, respectively.

Figure 9.4: Locally irregular edge-colouring paths and cycles.

9.2.1 Some common families of graphs

Paths and cycles

Recall that colourable paths and cycles are those with even length according to Corollary 9.2. We start by showing that colourable paths have irregular chromatic index at most 2, while cycles have irregular chromatic index at most 3, this showing that Conjecture 9.9, if true, would be tight.

Proposition 9.10. *We have $\chi'_{irr}(P_3) = 1$, and $\chi'_{irr}(P_n) = 2$ for every $n \geq 5$ odd.*

Proof. The statement for P_3 follows from Observation 9.1. Recall that each colour of a locally irregular edge-colouring of P_n, with $n \geq 5$ odd, induces a forest of P_3, see Observation 9.1. A locally irregular 2-edge-colouring of P_n can then be obtained by colouring pairs of subsequent edges with a same colour starting from one endedge of P_n, using colours 1 and 2 alternatively (see Figure 9.4.a) ■

Proposition 9.11. *Let $n \geq 4$ even. We have $\chi'_{irr}(C_n) = 2$ if $n \equiv 0 \pmod 4$, or $\chi'_{irr}(C_n) = 3$ otherwise.*

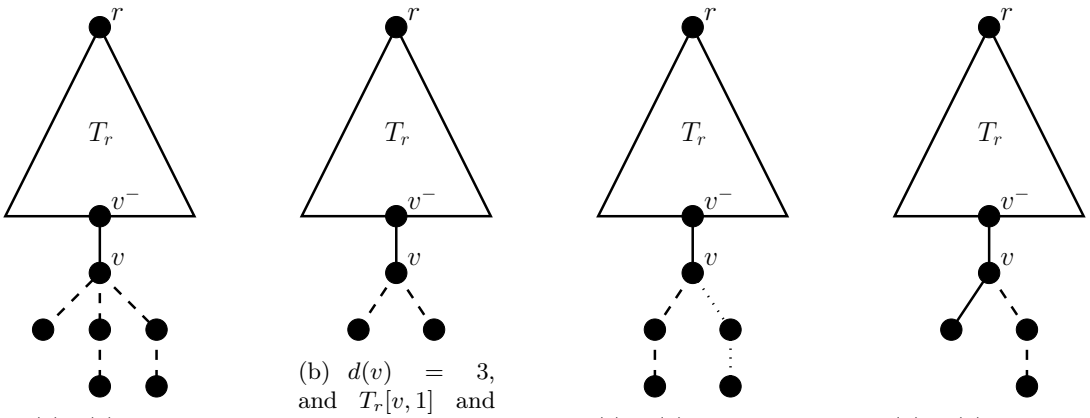
Proof. Repeating the colouring procedure from the proof of Proposition 9.10 (except that the first pair of subsequent edges is chosen arbitrarily) with two colours, we obtain a locally irregular 2-edge-colouring of C_n whenever $n \equiv 0 \pmod 4$. When $n \equiv 2 \pmod 4$, stop the procedure once only two adjacent edges e_1 and e_2 remain to colour. At this very moment, these two edges are adjacent to a pair of consecutive edges coloured 1, and a pair of consecutive edges coloured 2. In this situation, we then have to colour e_1 and e_2 with colour 3, see Figure 9.4.b. ■

Bipartite graphs

Although dealing with bipartite graphs for other vertex-distinguishing edge-colouring notions showed up to be easy, things seem to be harder when dealing with locally irregular edge-colouring. According to Corollary 9.2, we even know that infinitely many bipartite graphs are not colourable, namely paths with odd length. We here only show Conjecture 9.9 to hold for specific classes of bipartite graphs, namely complete bipartite graphs, regular bipartite graphs with minimum degree at least 3, and trees (next Section 9.2.1 is devoted to this specific case).

Proposition 9.12. *Let $p \geq 2$ and $q \geq 1$. We have $\chi'_{irr}(M_2(p,q)) = 1$ if $p \neq q$, or $\chi'_{irr}(M_2(p,p)) = 2$ otherwise.*

Proof. If $p \neq q$, then $M_2(p,q)$ is locally irregular and thus has irregular chromatic index 1. Now assume $p = q \geq 2$ (if $p = q = 1$, then $M_2(p,p)$ is isomorphic to K_2, which is not colourable),



(a) $d(v) \geq 4$.

(b) $d(v) = 3$, and $T_r[v,1]$ and $T_r[v,2]$ have length 1.

(c) $d(v) = 3$, and $T_r[v,1]$ and $T_r[v,2]$ have length 2.

(d) $d(v) = 3$, and $T_r[v,1]$ and $T_r[v,2]$ have length 1 and 2, respectively.

Figure 9.5: Situations described in the proof of Theorem 9.14. A subtree of T_r rooted at v is removed so that we can deduce a locally irregular 3-edge-colouring of the remaining tree. This locally irregular edge-colouring is then extended to the removed subtree. Solid, dashed, and dotted edges represent 1-, 2-, and 3-coloured edges.

and let v be an arbitrary vertex of $M_2(p,p)$. We produce a locally irregular 2-edge-colouring of $M_2(p,p)$ as follows. Start by colouring 1 all edges of $M_2(p,p) - \{v\}$. Then colour 2 all remaining edges, i.e. those incident with v. Then note that the 1- and 2-subgraphs are complete unbalanced bipartite graphs, and are then locally irregular. ■

Recall that a neighbour-multiset-distinguishing 2-edge-colouring of a regular graph is also locally irregular according to Observations 7.17 and 7.37. Since every bipartite graph with minimum degree at least 3 admits a neighbour-multiset-distinguishing 2-edge-colouring, recall Theorem 7.22, we directly get the following.

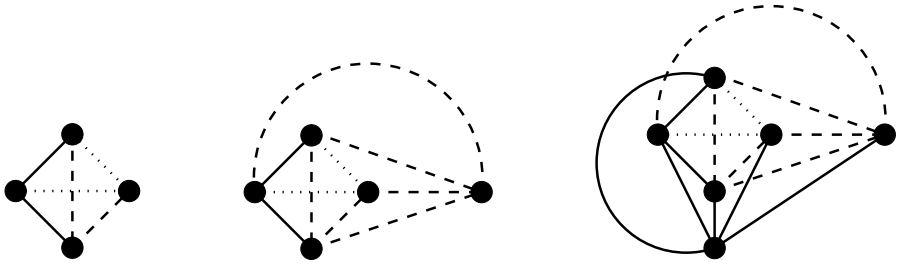
Corollary 9.13. *For every regular bipartite graph G with $\delta(G) \geq 3$, we have $\chi'_{irr}(G) \leq 2$.*

Trees

We now give a positive answer to Conjecture 9.9 in the context of trees. It is worth mentioning that a deeper study of the irregular chromatic index of trees can be found in dedicated Section 9.3.2, wherein it is shown that the irregular chromatic index of a tree can be determined in linear time.

Theorem 9.14. *For every colourable tree T, we have $\chi'_{irr}(T) \leq 3$.*

Proof. We prove the claim by induction on the number of nodes with degree at least 3 in T. Let us consider, as base cases, the following two situations. On the one hand, if $\delta(T) \leq 2$, i.e. T has no node with degree at least 3, then T is a path with even length and thus has irregular chromatic index at most 2 according to Proposition 9.10. On the other hand, if T has only one node r with degree at least 3, then let T_r be the rooted tree obtained by rooting T at r. Since r is the only node with degree at least 3 in T, note that each of $T_r[r,1], T_r[r,2], ..., T_r[r,d(r)]$ is a path. Then a locally irregular 3-edge-colouring of T_r is obtained as follows. Use colour 1 to colour every edge incident with r, plus possibly the edge rr^+ of each of $T_r[r,1], T_r[r,2], ..., T_r[r,d(r)]$ so that the uncoloured edges form a forest of paths with even length (possibly of length 0). The 1-subgraph is then locally irregular since r has degree at least 3 in this subgraph by assumption,



(a) K_4. (b) K_5. (c) K_6.

Figure 9.6: Extending a locally irregular 3-edge-colouring of K_4 to complete graphs with larger order. Solid, dashed, and dotted edges represent 1-, 2-, and 3-coloured edges.

and is adjacent to nodes with degree 2 or 1. Besides, every degree-2 node of the 1-subgraph is adjacent to r and a degree-1 node. Then use colours 2 and 3 (if necessary) to colour the edges of the remaining forest so that the two induced subgraphs are locally irregular. This is possible by Proposition 9.10 since this forest only consists in paths with even length. Colours 1, 2 and 3 then form a locally irregular 3-edge-colouring of T.

Suppose now that the claim is true whenever T has up to $i-1$ nodes with degree at least 3, and assume T has i such nodes. Let r be a node of T, and let T_r be the rooted tree obtained by rooting T at r. Let $v \neq r$ be a node of T_r such that $d(v) \geq 3$ and v does not have any node with degree at least 3 in its descendants. Note that $T - V(T_r[v])$ is a colourable tree since it has nodes with degree at least 3 (this remains true if we put back a subgraph of $T_r[v]$ in T). By our choice of v, note further that each of $T_r[v,1], T_r[v,2], ..., T_r[v,d(v)-1]$ is a path. According to how a locally irregular edge-colouring is propagated along a path, we may also assume that each of the $T_r[v,i]$'s is a path with length 1 or 2.

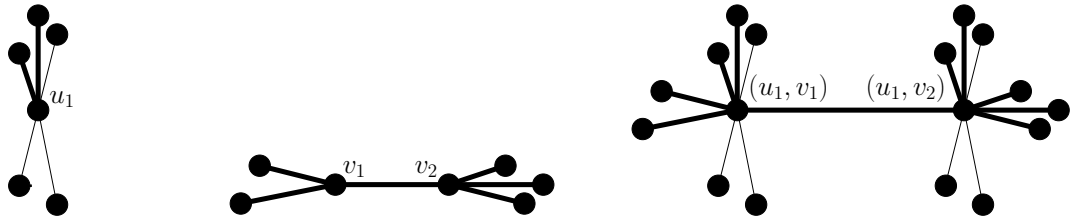
Note that if $d(v) \geq 4$, then $T_r[v]$ is a locally irregular tree. A locally irregular 3-edge-colouring of T can then be obtained by starting from a locally irregular 3-edge-colouring of $T - V(T_r[v]) + \{v\}$, which exists according to the induction hypothesis, and then colouring all the remaining edges of T, i.e. those of $T_r[v]$, with a colour different from the one used for v^-v (see Figure 9.5.a).

A locally irregular 3-edge-colouring of T can be obtained in an analogous way when $d(v) = 3$. First, when both $T_r[v,1]$ and $T_r[v,2]$ have length 1, the strategy used in the case $d(v) \geq 4$ provides a locally irregular 3-edge-colouring of T since two adjacent edges form a locally irregular graph (see Figure 9.5.b). Now, if $T_r[v,1]$ and $T_r[v,2]$ have length 2, then the same strategy can again be used, except that two colours are necessary to colour the edges of $T_r[v]$, which is a path with length 4. Assuming v^-v is coloured, say, 1, using colour 2 for the edges of $T_r[v,1]$ and colour 3 for the edges of $T_r[v,2]$ provides a locally irregular 3-edge-colouring of T (see Figure 9.5.c). Finally, if $T_r[v,1]$ and $T_r[v,2]$ have length 1 and 2, respectively, without loss of generality, then a locally irregular 3-edge-colouring of T can be obtained as follows. Consider first a locally irregular 3-edge-colouring of $T - V(T_r[v,2]) + \{v\}$, which exists according to the induction hypothesis, and colour the two remaining adjacent edges of T, i.e. those of $T_r[v,2]$, with a colour different from the one used to colour the edges v^-v and vv^+ in $T - V(T_r[v,2]) + \{v\}$ (these two edges necessarily have the same colour). The resulting 3-edge-colouring remains locally irregular, as shown in Figure 9.5.d. This completes the proof. ■

Complete graphs

We now prove that every complete graph with order at least 4 admits a locally irregular 3-edge-colouring.

Theorem 9.15. *For every $n \geq 4$, we have $\chi'_{irr}(K_n) \leq 3$.*



(a) u_1 has some 1- and 2-degree in the 1- and 2-subgraphs of G induced by c_G.

(b) v_1 and v_2 are adjacent vertices with distinct 1-degrees of the 1-subgraph of H induced by c_H.

(c) (u_1, v_1) and (u_1, v_2) are adjacent vertices with distinct 1-degrees of the 1-subgraph of $G \square H$ induced by c.

Figure 9.7: Deduction of a locally irregular 2-edge-colouring c of $G \square H$ from locally irregular 2-edge-colourings c_G and c_H of G and H, respectively. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

Proof. We prove this statement by induction on n. We actually prove a slightly stronger claim. Namely, for every $n \geq 4$, we prove that there exists a locally irregular 3-edge-colouring of K_n in which either there is no vertex whose all $n-1$ incident edges are coloured 1, or there is no vertex whose all $n-1$ incident edges are coloured 2.

As a base case, note that a locally irregular 3-edge-colouring of K_4 satisfying these requirements can be obtained by colouring two adjacent edges 1, two adjacent edges 2, and the remaining two adjacent edges 3, see Figure 9.6.a. Suppose now the claim is true for every n up to an i, and consider $n = i + 1$. By the induction hypothesis, there exists a locally irregular 3-edge-colouring of any induced K_{n-1} subgraph of K_n such that no vertex has its $n-2$ incident edges coloured, say, 1. Then, by just colouring 1 all of the $n-1$ uncoloured edges, it should be clear that the edge-colouring of K_n remains locally irregular. Besides, there is no vertex whose all $n-1$ incident edges are coloured 2, so the additional requirement is met. The stronger statement is thus also true for n. See Figure 9.6 for an illustration of how this locally irregular 3-edge-colouring is extended from one complete graph to the next one. ■

An interesting fact about the proof of Theorem 9.15 is that colour 3 is only used twice in every deduced locally irregular 3-edge-colouring of a complete graph K_n, basically to colour two edges of an induced K_4. So using the same colouring procedure, two colours actually suffice to obtain a locally irregular edge-colouring if we remove one or two edges from K_n.

Theorem 9.16. *For every $n \geq 4$ and arbitrary edges e and e' of K_n, we have $\chi'_{irr}(K_n - \{e\}) = \chi'_{irr}(K_n - \{e, e'\}) = 2$.*

Proof. The proof is analogous to the one of Theorem 9.15. First colour a subgraph of $K_n - \{e\}$ or $K_n - \{e, e'\}$ of order 4 which includes the vertices incident with the removed edges. This time, only two colours suffice to obtain a locally irregular edge-colouring of this subgraph. Then colour the remaining edges as in the proof of Theorem 9.15. ■

Cartesian products of graphs with irregular chromatic index at most 3

Using the Cartesian product of graphs, we can provide numerous more examples of graphs supporting Conjecture 9.9. Namely, we show that if two graphs G and H have irregular chromatic index at most 3, then so has their Cartesian product $G \square H$.

Theorem 9.17. *For every two graphs G and H, we have $\chi'_{irr}(G \square H) \leq \max\{\chi'_{irr}(G), \chi'_{irr}(H)\}$.*

Proof. Let c_G and c_H be locally irregular $\chi'_{irr}(G)$- and $\chi'_{irr}(H)$-edge-colourings of G and H, respectively. Then denote by c the $\max\{\chi'_{irr}(G), \chi'_{irr}(H)\}$-edge-colouring of $G \square H$ defined as follows (see Figure 9.7).

$$c((u_1,v_1)(u_2,v_2)) = \begin{cases} c_H(v_1v_2) & \text{if } u_1 = u_2, \\ c_G(u_1u_2) & \text{otherwise.} \end{cases}$$

Given a colour of c, say 1, and a vertex (u_1,v_1) of $G \,\square\, H$, note that $d_{c,1}(u_1,v_1) = d_{c_G,1}(u_1) + d_{c_H,1}(v_1)$. Now consider an edge $(u_1,v_1)(u_2,v_2)$ of $G \,\square\, H$ coloured 1 by c. We just need to show that $d_{c,1}(u_1,v_1) \neq d_{c,1}(u_2,v_2)$. Since (u_1,v_1) and (u_2,v_2) are adjacent, we may assume that $u_1 = u_2$ without loss of generality. Then $d_{c,1}(u_1,v_1) = d_{c_G,1}(u_1) + d_{c_H,1}(v_1)$ and $d_{c,1}(u_2,v_2) = d_{c_G,1}(u_1) + d_{c_H,1}(v_2)$. Since c_H is locally irregular, we have $d_{c_H,1}(v_1) \neq d_{c_H,1}(v_2)$, implying that (u_1,v_1) and (u_2,v_2) are adjacent vertices in the 1-subgraph induced by c, but have distinct 1-degrees. ■

Corollary 9.18. *For every two graphs G and H with $\chi'_{irr}(G), \chi'_{irr}(H) \leq 3$, we have $\chi'_{irr}(G \,\square\, H) \leq 3$*

Corollary 9.18 in particular implies that some other bipartite graphs, like e.g. hypercubes and grids, do not refute Conjecture 9.9. Since these graphs result from the Cartesian product of two graphs with irregular chromatic index at most 2 (with the exception that, since $Q_1 \simeq K_2$, one has to check by hand that Q_2 and Q_3 have irregular chromatic index at most 2), Corollary 9.18 ensures that these graphs even admit a locally irregular 2-edge-colouring.

9.2.2 Regular graphs with large degree

We now prove that regular graphs with sufficiently large degree, i.e. greater than 10^7, agree with Conjecture 9.9. This result is significant as regular graphs are, in some sense, the least locally irregular graphs. We make use of a probabilistic approach to prove it. For this purpose, we need to introduce the following result by Addario-Berry, Dalal, McDiarmid, Reed and Thomason [2] on the existence of spanning subgraphs of a graph in which every vertex has a prescribed degree. Refer e.g. to works of Addario-Berry, Aldred, Kalal and Reed [1] and of Addario-Berry, Kalal and Reed [3] for other applications of this result.

Theorem 9.19 ([2])**.** *Suppose that, for every vertex v of a graph G, we have chosen two integers a_v^- and a_v^+ with*

$$a_v^- \in \left\{ \frac{d(v)}{3} - 1, \frac{d(v)}{3}, ..., \frac{d(v)}{2} \right\}, \quad a_v^+ \in \left\{ \frac{d(v)}{2} - 1, \frac{d(v)}{2}, ..., \frac{2d(v)}{3} \right\}.$$

Then there exists a spanning subgraph H of G such that

$$d_H(v) \in \{a_v^-, a_v^- + 1, a_v^+, a_v^+ + 1\}$$

for every vertex v of G.

Corollary 9.20. *Given a graph G, a positive integer $\lambda \leq \frac{\delta(G)}{6}$, and an assignment*

$$t : V \to \{0, 1, ..., \lambda - 1\},$$

there exists a spanning subgraph H of G such that $d_H(v) \in \{\frac{d(v)}{3}, \frac{d(v)}{3} + 1, ..., \frac{2d(v)}{3}\}$, and either $d_H(v) \equiv t(v) \pmod{\lambda}$ or $d_H(v) \equiv t(v) + 1 \pmod{\lambda}$ for every vertex v of G.

Proof. Note that for every vertex v of G, we have

$$\left\lfloor \frac{d(v)}{2} \right\rfloor - \left\lfloor \frac{d(v)}{3} \right\rfloor + 1 \geq \frac{d(v) - 1}{2} - \frac{d(v)}{3} + 1 > \frac{d(v)}{6} \geq \lambda.$$

Hence, since both sides of the inequality are integers,

$$\left\lfloor \frac{d(v)}{2} \right\rfloor - \left\lfloor \frac{d(v)}{3} \right\rfloor \geq \lambda.$$

Analogously, we have

$$\left\lfloor \frac{2d(v)}{3} \right\rfloor - \left\lfloor \frac{d(v)}{2} \right\rfloor + 1 \geq \frac{2d(v)-2}{3} - \frac{d(v)}{2} + 1 > \frac{d(v)}{6} \geq \lambda,$$

implying

$$\left\lfloor \frac{2d(v)}{3} \right\rfloor - \left\lfloor \frac{d(v)}{2} \right\rfloor \geq \lambda.$$

Therefore, the sets of integers

$$\left\{ \left\lfloor \frac{d(v)}{3} \right\rfloor + 1, \left\lfloor \frac{d(v)}{3} \right\rfloor + 2, ..., \left\lfloor \frac{d(v)}{2} \right\rfloor \right\} \quad \text{and} \quad \left\{ \left\lfloor \frac{d(v)}{2} \right\rfloor, \left\lfloor \frac{d(v)}{2} \right\rfloor + 1, ..., \left\lfloor \frac{2d(v)}{3} \right\rfloor - 1 \right\}$$

both contain all remainders modulo λ. The claim then follows by applying Theorem 9.19 with the a_v^-'s and a_v^+'s belonging to these sets, i.e. with having $a_v^-, a_v^+ \equiv t(v) \pmod{\lambda}$ for every vertex v. ■

We are now ready to prove the main result of this section.

Theorem 9.21. *For every d-regular graph G with $d \geq 10^7$, we have $\chi'_{irr}(G) \leq 3$.*

Proof. We first randomly and independently choose one value $c_1(v)$ in $\{0, 1, ..., \lceil d^{0.35} \rceil - 1\}$ for every vertex v of G, each with equal probability. We then repeat this drawing for every vertex v of G, and denote the second obtained value by $c_2(v)$. For each vertex v, let us denote

$$A(v) = \{u \in N_G(v) : c_1(u) = c_1(v)\},$$

$$B(v) = \{u \in N_G(v) : c_2(u) = c_2(v)\},$$

$$C(v) = \{u \in N_G(v) : c_1(u) + c_2(u) \equiv c_1(v) + c_2(v) \pmod{\lceil d^{0.35} \rceil}\},$$

and note that

$$D(v) = B(v) \cap C(v) = \{u \in N_G(v) : c_1(u) = c_1(v) \wedge c_2(u) = c_2(v)\}.$$

We first use Theorems 1.28 and 1.29 to prove the following probabilistic claim.

Claim 9.22. *With positive probability, we have*

$$|A(v)|, |B(v)|, |C(v)| \leq 2d^{0.65} \quad \textit{and} \tag{9.1}$$

$$|D(v)| \leq 2d^{0.3} - 1 \tag{9.2}$$

for every vertex v of G.

Proof. For every vertex v, let X_v, Y_v, Z_v, and T_v be the random variables of the cardinalities of the sets $A(v)$, $B(v)$, $C(v)$, and $D(v)$, respectively, and let A_v, B_v, C_v, and D_v denote the events $X_v > 2d^{0.65}$, $Y_v > 2d^{0.65}$, $Z_v > 2d^{0.65}$, and $T_v > 2d^{0.3} - 1$, respectively. Consider any vertex u neighbouring v. Then, we have

$$\Pr(u \in A(v)) = \frac{1}{\lceil d^{0.35} \rceil} \leq \frac{1}{d^{0.35}} \quad \text{and}$$

$$\Pr(u \in B(v)) = \frac{1}{\lceil d^{0.35} \rceil} \leq \frac{1}{d^{0.35}}.$$

Since, for every fixed value of $c_1(v)$ and $c_2(v)$ (and e.g. $c_1(u)$), the probability that $c_1(u) + c_2(u) \equiv c_1(v) + c_2(v) \pmod{\lceil d^{0.35} \rceil}$ equals exactly $\frac{1}{\lceil d^{0.35} \rceil}$, by the total probability we also get

$$\Pr(u \in C(v)) = \frac{1}{\lceil d^{0.35} \rceil} \leq \frac{1}{d^{0.35}}.$$

Finally, since the drawings of the $c_1(v)$'s and $c_2(v)$'s are independent, we have

$$\Pr(u \in D(v)) = \left(\frac{1}{\lceil d^{0.35} \rceil}\right)^2 \leq \frac{1}{d^{0.7}}.$$

Consequently, since again all our drawings are independent, by Theorem 1.29 we obtain the following. To be strict, note that we should have first written the conditional probability with respect to some fixed values of $c_1(v)$, but since all our choices are independent we would have ended up with the same upper bound no matter what is $c_1(v)$. So

$$\begin{aligned}
\Pr(A_v) &= \Pr\left(X_v > 2d^{0.65}\right) \leq \Pr\left(\mathrm{BIN}\left(d, \frac{1}{d^{0.35}}\right) > 2d^{0.65}\right) \\
&\leq \Pr\left(\left|\mathrm{BIN}\left(d, \frac{1}{d^{0.35}}\right) - d^{0.65}\right| > d^{0.65}\right) \\
&< 2e^{-\frac{d^{0.65}}{3}} \leq 2e^{-\frac{2}{7}d^{0.3}}. \qquad (9.3)
\end{aligned}$$

Analogously, we have

$$\Pr(B_v) < 2e^{-\frac{2}{7}d^{0.3}} \quad \text{and} \quad \Pr(C_v) < 2e^{-\frac{2}{7}d^{0.3}}. \qquad (9.4)$$

Finally, again by Theorem 1.29, we obtain

$$\begin{aligned}
\Pr(D_v) &= \Pr\left(T_v > 2d^{0.3} - 1\right) \leq \Pr\left(\mathrm{BIN}\left(d, \frac{1}{d^{0.7}}\right) > 2d^{0.3} - 1\right) \\
&\leq \Pr\left(\left|\mathrm{BIN}\left(d, \frac{1}{d^{0.7}}\right) - d^{0.3}\right| > d^{0.3} - 1\right) \\
&< 2e^{-\frac{(d^{0.3}-1)^2}{3d^{0.3}}} \leq 2e^{-\frac{\left(\sqrt{\frac{6}{7}}d^{0.3}\right)^2}{3d^{0.3}}} = 2e^{-\frac{2}{7}d^{0.3}} \qquad (9.5)
\end{aligned}$$

whenever $d \geq \left(\frac{1}{1-\sqrt{\frac{6}{7}}}\right)^{\frac{10}{3}} \approx 5,831$.

Since each of the events A_v, B_c, C_v, and D_v only depends on the random drawings for v and its neighbours, each of these events is mutually independent of all other events $A_{v'}$, $B_{v'}$, $C_{v'}$ and $D_{v'}$ where v' is a vertex at distance at least 3 from v in G, and hence dependent of at most $3 + 4d^2$ other events. Moreover, by Inequalities 9.3, 9.4 and 9.5, the probability of each of these events equals at most $2e^{-\frac{2}{7}d^{0.3}}$. So that Theorem 1.28 is applicable, we thus need to prove that the following holds:

$$e2e^{-\frac{2}{7}d^{0.3}}(4 + 4d^2) \leq 1 \qquad (9.6)$$

For this purpose, we first show that

$$f(d) > 0, \qquad (9.7)$$

where $f(d) = e^{\frac{1}{7}d^{0.3}} - 5d$ is continuous in $\mathbb{R}^+$ for $d \geq 10^7$. Note that

$$\begin{aligned}
f'(d) &= \frac{0.3}{7}d^{-0.7}e^{\frac{1}{7}d^{0.3}} - 5, \\
f''(d) &= \frac{0.09}{49}d^{-1.7}e^{\frac{1}{7}d^{0.3}}\left(d^{0.3} - \frac{49}{3}\right),
\end{aligned}$$

then for $d \geq (\frac{49}{3})^{\frac{10}{3}} \approx 11,056$, we get $f''(d) \geq 0$ and thus $f'(d)$ is increasing. Since, at the same time, we have $f'(9,425,780) \approx 22 > 0$, then $f'(d) > 0$ for every $d \geq 9,425,780$. Besides, since

$f(9,425,780) \approx 3 > 0$, we have $f(d) > 0$ for $d \geq 10^7$, implying Inequality 9.7 ($9,425,780$ is actually the smallest integer for which f has a positive value).

Using Inequality 9.7, we then get

$$e^{\frac{2}{7}d^{0.3}} > 25d^2 \geq 6(4+4d^2) \geq e2(4+4d^2)$$

for every $d^2 \geq 24$, implying Inequality 9.6. By Theorem 1.28, we thus obtain

$$\Pr\left(\bigcap_{v\in V(G)} \overline{A_v} \cap \overline{B_v} \cap \overline{C_v} \cap \overline{D_v}\right) > 0,$$

concluding the proof. ■

Suppose then that the assignments c_1 and c_2 have been chosen in such a way that Inequalities 9.1 and 9.2 are fulfilled for every vertex v of G. Since $|D(v)|$ is an integer, we have $|D(v)| \leq \lfloor 2d^{0.3} \rfloor - 1$ according to Inequality 9.2. Let G' be the graph obtained from G by removing it all edges uv such that $c_1(u) = c_1(v)$. According to Inequality 9.1, we have

$$\delta(G') \geq d - 2d^{0.65} = d^{0.3}(d^{0.7} - 2d^{0.35}) \geq d^{0.3}(36d^{0.35} + 36), \tag{9.8}$$

where Inequality 9.8, which is equivalent to $d^{0.7} - 38d^{0.35} - 36 \geq 0$, holds for $d^{0.35} \geq \frac{38+\sqrt{38^2+4\cdot 36}}{2}$, i.e. for $d \geq 34,955$. By Inequality 9.8, we thus obtain

$$\frac{\delta(G')}{6} \geq 6d^{0.3}(d^{0.35}+1) \geq 3\lfloor 2d^{0.3}\rfloor \lceil d^{0.35}\rceil. \tag{9.9}$$

Using Corollary 9.20, we may thus find a subgraph H_1 of G' such that $d_{H_1}(v)$ has either remainder $3\lfloor 2d^{0.3}\rfloor c_1(v)$ or $3\lfloor 2d^{0.3}\rfloor c_1(v)+1$ modulo $\lambda = 3\lfloor 2d^{0.3}\rfloor \lceil d^{0.35}\rceil$, that is

$$d_{H_1}(v) \equiv 3\lfloor 2d^{0.3}\rfloor c_1(v), 3\lfloor 2d^{0.3}\rfloor c_1(v)+1 \pmod{3\lfloor 2d^{0.3}\rfloor \lceil d^{0.35}\rceil}, \tag{9.10}$$

for every vertex v of G, and

$$\Delta(H_1) \geq \frac{2\Delta(G')}{3} \geq \frac{2d}{3}. \tag{9.11}$$

Colour the edges of H_1 with colour 1. According to Inequality 9.10, we have $d_{H_1}(u) \neq d_{H_1}(v)$ if $c_1(u) \neq c_1(v)$, what is fulfilled for every edge uv of G', hence also for every edge of H_1 since H_1 is a subgraph of G'. Therefore, the graph H_1, and thus the subgraph of G induced by edges coloured 1, is locally irregular.

Let G_1 be the graph obtained by removing from G all edges coloured 1, i.e. the edges of H_1. By Inequality 9.11, we have

$$\delta(G_1) \geq \frac{d}{3}. \tag{9.12}$$

Let G'' be the graph obtained from G_1 by removing from it all edges uv such that $c_2(u) = c_2(v)$ or $c_1(u)+c_2(u) \equiv c_1(v)+c_2(v) \pmod{\lceil d^{0.35}\rceil}$. Using Inequalities 9.3 and 9.12, we get

$$\delta(G'') \geq \frac{d}{3} - 4d^{0.65} = \frac{d^{0.3}}{3}(d^{0.7} - 12d^{0.35}) \geq \frac{d^{0.3}}{3}(108d^{0.35} + 108), \tag{9.13}$$

where Inequality 9.13, which is equivalent to $d^{0.7} - 120d^{0.35} - 108 \geq 0$, holds for $d^{0.35} \geq \frac{120+\sqrt{120^2+4\cdot 108}}{2}$, that is for $d \geq 890,679$. By Inequality 9.13, we thus get

$$\frac{\delta(G'')}{6} \geq 6d^{0.3}(d^{0.35}+1) \geq 3\lfloor 2d^{0.3}\rfloor \lceil d^{0.35}\rceil. \tag{9.14}$$

Let C be the subgraph induced by these edges uv of G_1 for which $c_1(u)+c_2(u) \equiv c_1(v)+c_2(v) \pmod{\lceil d^{0.35} \rceil}$. Note that C and G'' are edge-disjoint. Now, for every vertex v of G, let c_v be

$$c_v = d_C(v) = |C(v) \cap N_{G_1}(v)|, \tag{9.15}$$

that is the number of edges uv incident with v in G_1 such that $c_1(u) + c_2(u) \equiv c_1(v) + c_2(v) \pmod{\lceil d^{0.35} \rceil}$. Consider the subgraph D induced by these edges uv of G_1 for which $c_1(u) = c_1(v)$ and $c_2(u) = c_2(v)$. Note that D is a subgraph of C. According to Inequality 9.4, we have

$$\Delta(D) \leq \lfloor 2d^{0.3} \rfloor - 1,$$

and, hence, we may greedily find a proper vertex-colouring

$$h : V(D) \to \{0, 1, ..., \lfloor 2d^{0.3} \rfloor - 1\}$$

of D, where we set e.g. $h(v) = 0$ if v is not incident with an edge of D. By Corollary 9.20 and Inequality 9.14, we may find a subgraph H_2 of G'' such that

$$\begin{aligned} d_{H_2}(v) \quad \equiv \quad & 3\lfloor 2d^{0.3} \rfloor c_2(v) + 3h(v) - c_v, \\ & 3\lfloor 2d^{0.3} \rfloor c_2(v) + 3h(v) - c_v + 1 \pmod{3\lfloor 2d^{0.3} \rfloor \lceil d^{0.35} \rceil} \end{aligned} \tag{9.16}$$

for every vertex v of G. Then we colour the edges of H_2 and C with colour 2, while the remaining edges of G_1 are coloured 3. Let H_2' and H_3' denote the subgraphs induced by colours 2 and 3, respectively. Then, since H_2 and C are edge-disjoint, by Inequalities 9.15 and 9.16 we have

$$\begin{aligned} d_{H_2'}(v) \quad \equiv \quad & 3\lfloor 2d^{0.3} \rfloor c_2(v) + 3h(v), \\ & 3\lfloor 2d^{0.3} \rfloor c_2(v) + 3h(v) + 1 \pmod{3\lfloor 2d^{0.3} \rfloor \lceil d^{0.35} \rceil} \end{aligned} \tag{9.17}$$

for every vertex v of G. Therefore, we have $d_{H_2'}(u) \neq d_{H_2'}(v)$ if $c_2(u) \neq c_2(v)$ or $h(u) \neq h(v)$. The latter of these two conditions is obviously fulfilled for every edge uv of D since h is proper. Besides, we have $c_2(u) \neq c_2(v)$ for the remaining edges of H_2' according to the definitions of C and G''. The subgraph of G induced by colour 2 is thus locally irregular.

By Inequalities 9.10 and 9.17, we have

$$\begin{aligned} d_{H_1}(v) + d_{H_2'}(v) \quad \equiv \quad & 3\lfloor 2d^{0.3} \rfloor (c_1(v) + c_2(v)) + 3h(v), \\ & 3\lfloor 2d^{0.3} \rfloor (c_1(v) + c_2(v)) + 3h(v) + 1, \\ & 3\lfloor 2d^{0.3} \rfloor (c_1(v) + c_2(v)) + 3h(v) + 2 \pmod{3\lfloor 2d^{0.3} \rfloor \lceil d^{0.35} \rceil} \end{aligned} \tag{9.18}$$

for every vertex v of G. Since G is d-regular, and hence we have $d_{H_3'}(v) = d - d_{H_1}(v) + d_{H_2'}(v)$ for every vertex v of G, then, according to Inequality 9.18, we have $d_{H_3'}(u) \neq d_{H_3'}(v)$ if $c_1(u) + c_2(u) \not\equiv c_1(v) + c_2(v) \pmod{\lceil d^{0.35} \rceil}$ or $h(u) \neq h(v)$. However, all edges $uv \in E(G) \setminus E(H_1)$ with $c_1(u) + c_2(u) \equiv c_1(v) + c_2(v) \pmod{\lceil d^{0.35} \rceil}$, i.e. the edges of C, have been coloured 2. The graph H_3' is then locally irregular, implying that colour 3 induces a locally irregular subgraph of G. Colours 1, 2 and 3 then form a locally irregular edge-colouring of G, implying that $\chi'_{irr}(G) \leq 3$. ■

Regarding the decomposition of regular graphs into only two locally irregular subgraphs, using Corollary 9.20 we can even deduce the following.

Corollary 9.23. *For every d-regular graph G with $d \geq 12\chi(G)$, we have $\chi'_{irr}(G) \leq 2$*

Proof. Let $t : V \to \{0, 2, 4, \ldots, 2\chi - 2\}$ be a proper vertex-colouring of G, and set $\lambda = 2\chi$. Since $\lambda \leq \frac{d}{6}$, by Corollary 9.20 there exists a spanning subgraph H of G satisfying $d_H(v) \equiv t(v) \pmod{\lambda}$ or $d_H(v) \equiv t(v)+1 \pmod{\lambda}$ for every vertex $v \in V(G)$. Then, for every edge $uv \in E(G)$, we have $d_H(u) \neq d_H(v)$, and hence also $d - d_H(u) \neq d - d_H(v)$. The graphs H and $G - E(H)$ thus make up a decomposition of G into two locally irregular subgraphs. ■

Similarly as done by Addario-Berry, Dalal and Reed in [3], we may in turn derive Corollary 9.23 so that we get a counterpart of Theorem 9.21 for regular graphs with irregular chromatic index at most 2. This stands as another support for Conjecture 9.9.

Corollary 9.24. *There exists an absolute integer constant d_0 such that if G_d is a random d-regular graph (sampled uniformly from the family of all d-regular graphs with order $|V(G_d)|$) for a constant $d > d_0$, then asymptotically almost surely we have $\chi'_{irr}(G_d) \leq 2$.*

Proof. By a result of Frieze and Łuczak [60], there exists a constant d'_0 such that if $d > d'_0$ is a constant (and $d = o(n^\theta)$ for a constant $\theta < \frac{1}{3}$), then

$$\chi(G_d) \leq \frac{d}{2\ln d}\left(1 + \frac{32\ln(\ln(d))}{\ln(d)}\right) \tag{9.19}$$

with probability going to 1 as $|V(G)|$ grows to infinity. This means that if $d > d'_0$, then asymptotically almost surely Inequality 9.19 holds. Since for d sufficiently large, i.e. for $d > d''_0$ where d''_0 is a (other) constant, Inequality 9.19 implies that $\chi(G_d) \leq \frac{d}{12}$, the thesis follows by Corollary 9.23 with $d_0 = \max\{d'_0, d''_0\}$. ■

Using Observation 7.37, recall that we can directly derive Corollary 9.24 for neighbour-sum-distinguishing edge-weighting of regular graphs.

9.3 Determining the irregular chromatic index of a graph

Throughout this section, we study the algorithmic complexity of determining the irregular chromatic index of a graph. Since the notions of locally irregular edge-colouring and exception graphs are directly related, we first prove, in Section 9.3.1, that deciding whether a graph is an exception can be done in polynomial time.

We then focus on the complexity of the Locally Irregular k-Edge-Colouring problem for fixed values of k. Recall that a graph has irregular chromatic index 1 if and only if it is locally irregular. Since this property can be checked in quadratic time, the problem Locally Irregular 1-Edge-Colouring is in P. We cannot tell much about the complexity of every problem Locally Irregular k-Edge-Colouring with $k \geq 3$ since this is highly dependent on whether Conjecture 9.9 is true or not. In case Conjecture 9.9 turned out to be true, every problem Locally Irregular k-Edge-Colouring with $k \geq 3$ would be equivalent to the problem of deciding whether a graph is an exception, which we show to be in P (see Section 9.3.1). All problems Locally Irregular k-Edge-Colouring with $k \geq 3$ would thus be in P. On the contrary, if any problem Locally Irregular k-Edge-Colouring with $k \geq 3$ were shown to be NP-complete, then this result would refute Conjecture 9.9.

We thus mainly consider the complexity of Locally Irregular 2-Edge-Colouring herein. In particular, we show this problem to be in P when restricted to trees, and NP-complete in general (see Sections 9.3.2 and 9.3.3, respectively). This result implies that Locally Irregular k-Edge-Colouring should not be fixed-parameter tractable when parameterized by k, the number of colours.

9.3.1 Recognizing exceptions

Recall that a graph G is an exception (for the notion of locally irregular edge-colouring) if and only if G is either a path or cycle with odd length, or a member of the family $\mathcal{T}$ introduced in Construction 9.3. Since the structure of these three kinds of graphs is quite constrained, we can easily, i.e. in polynomial time, recognize whether G is an exception. We propose Algorithm 2 as such a naive polynomial-time recognition algorithm, though a refined linear one should be easy to design.

```
1  if Δ(G) ≥ 4 then
2      return false;
3  else
4      let T_1, T_2, ..., T_x denote the triangles of G;
5      if two distinct triangles T_i and T_j have common vertices then
6          return false;
7      if G has a degree-3 vertex belonging to none of the T_i's then
8          return false;
9      if G has a cycle of length at least 4 then
10         return false;
11     let P_1, P_2, ..., P_y denote the maximal paths of G involving vertices of degree at most 2;
12     foreach path P_j with j ∈ {1, 2, ..., y} do
13         if the two endvertices of P_j are pendant and P_j has even length then
14             return false;
15         else if one endvertex of P is pendant and P_j has odd length then
16             return false;
17         else if the endvertices of P_j belong to two of the T_i's and P_j has even length then
18             return false;
19     return true;
```

Algorithm 2: Determining whether a graph G is an exception.

Theorem 9.25. *Algorithm 2 determines whether a graph G is an exception in time $\mathcal{O}(|V(G)|^6)$.*

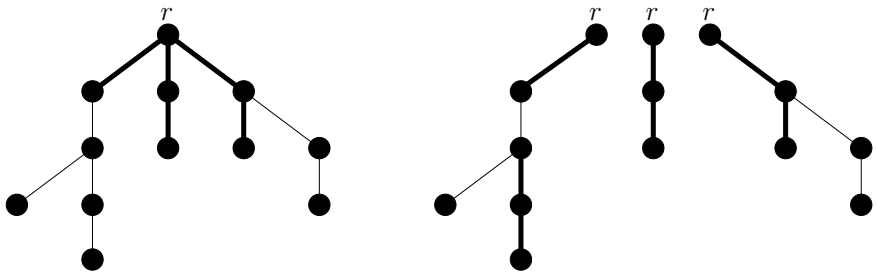
Proof. The correctness of Algorithm 2 follows from the description of the set of exceptions which we gave in Section 9.1.1. We now clarify the running time of Algorithm 2. The list of triangles mentioned at Line 4 can be computed in time $\mathcal{O}(|V(G)|^3)$ by considering every triple of vertices and checking whether they form a triangle. Every three vertices of G can form a triangle, so x is roughly $\mathcal{O}(|V(G)|^3)$. Comparing every two triangles at Line 5 can thus be done within $\mathcal{O}(|V(G)|^6)$ steps. Line 7 requires time $\mathcal{O}(|V(G)|^4)$, while Line 9 can be executed within time $\mathcal{O}(|V(G)|^5)$. This can be typically done during a depth-first search algorithm by checking whether every return edge forms a triangle.

The paths $P_1, P_2, ..., P_y$ can be computed again using a depth-first search algorithm. There are at most $|V(G)|$ of them, so $y \leq |V(G)|$. Once these paths are computed, Lines 13, 15 and 17 only consists in comparing the degrees of (some of) their vertices (there are at most $|V(G)|$ of them), or checking whether they belong to some of the $\mathcal{O}(|V(G)|^3)$ triangles $T_1, T_2, ..., T_x$. The most costly of these three instructions is Line 17, which can be achieved within time $\mathcal{O}(|V(G)|^4)$.

The most costly instruction of Algorithm 2 is thus Line 5, which implies that Algorithm 2 has time complexity $\mathcal{O}(|V(G)|^6)$. ■

9.3.2 Trees

Recall that every colourable tree has irregular chromatic index at most 3 according to Theorem 9.14. In this section, we propose a linear-time algorithm for determining the irregular chromatic index of a colourable tree. The existence of such an algorithm implies the membership of Locally Irregular 2-Edge-Colouring to P when restricted to trees. The keystone of our algorithm is the decomposition of a given tree T into specific trees, which we call *shrubs*, and which admit *almost locally irregular 2-edge-colourings*. These edge-colourings can then be combined to form a locally irregular edge-colouring of T.



(a) A locally irregular 2-edge-colouring of T_r.

(b) Almost locally irregular 2-edge-colourings $c^1_{1,2}$, $c^2_{1,2}$, and $c^3_{1,2}$ of $T_r[r,1]$, $T_r[r,2]$, and $T_r[r,3]$, respectively.

Figure 9.8: Decomposition of a rooted tree T_r into shrubs, and 2-edge-colourings of these. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

This section is organized as follows. We first introduce the notion of shrubs and almost locally irregular 2-edge-colouring in Section 9.3.2.1. We then show in Section 9.3.2.2 that every shrub admits an almost locally irregular 2-edge-colouring. In Section 9.3.2.3, we explain how independent almost locally irregular 2-edge-colourings of several shrubs can be unified to form a locally irregular 2-edge-colouring of an overlying tree. The success of this unification process is not guaranteed, in particular when the overlying tree has irregular chromatic index 3. By characterizing the situations in which this unification process may fail, we get, in Section 9.3.2.4, a concrete characterization of trees with irregular chromatic index 3. This emerges in a linear-time algorithm for determining the irregular chromatic of a tree in Section 9.3.2.5.

9.3.2.1 Preliminary definitions and notation

Let T_r be a rooted tree. In the case where r^+ is defined, i.e. the node r has only one child in T_r, we call T_r a *shrub*. If u is a node of T_r with children $v_1, v_2, ..., v_p$, then note that $T_r[u,i]$ is a shrub with $v_i = u^+$ and $u = v_i^-$ for every $i \in \{1,2,...,p\}$. Besides, by identifying the roots of the shrubs $T_r[r,1], T_r[r,2], ..., T_r[r,d(r)]$ we obtain T_r.

The upcoming notions and notation concern shrubs only. Assume T_r is a shrub, and let c be a 2-edge colouring of T_r. To make the colour of the edge rr^+ by c explicit, we also denote c by $c_{1,2}$ when $c(rr^+) = 1$, or $c_{2,1}$ when $c(rr^+) = 2$. If $c_{1,2}$ is a 2-edge colouring of T_r using colours 1 and 2, and 3 and 4 are two distinct colours, then we can obtain a 2-edge colouring $c_{3,4}$ of T_r using colours 3 and 4 by *swapping* $\{1,2\}$ and $\{3,4\}$: $c_{3,4}(uv) = 3$ if $c_{1,2}(uv) = 1$, or $c_{3,4}(uv) = 4$ otherwise. A swapping of $c_{1,2}$ to $c_{2,1}$ is called an *inversion*. Clearly, a node with 1-degree x in T_r by $c_{1,2}$ has 2-degree x by $c_{2,1}$. We further say that $c_{1,2}$ is an *almost locally irregular 2-edge-colouring* of T_r if either $c_{1,2}$ is a locally irregular 2-edge-colouring of T_r, or rr^+ is isolated in the 1-subgraph and $c_{1,2}$ is a locally irregular 2-edge-colouring of $T_r[r,1]$.

Example 9.26. In Figure 9.8 is shown how a rooted tree T_r (Figure 9.8.a) can be decomposed into several shrubs (Figure 9.8.b). Almost locally irregular 2-edge-colourings of these shrubs may directly form a locally irregular 2-edge-colouring of T_r.

9.3.2.2 Constructing almost locally irregular 2-edge-colourings of shrubs

An almost locally irregular 2-edge-colouring $c_{1,2}$ of every shrub can be obtained inductively using Algorithm 3. In this algorithm, we denote by $p \geq 0$ the number of children of r^+. Roughly speaking, the algorithm first inductively constructs almost locally irregular 2-edge-colourings $c^1_{1,2}, c^2_{1,2}, ..., c^p_{1,2}$ of $T_r[r^+,1], T_r[r^+,2], ..., T_r[r^+,p]$, respectively. It then inverts some of the $c^i_{1,2}$'s so that their union is an almost locally irregular 2-edge-colouring of T_r with rr^+ being coloured 1.

```
1 if p = 0 then
2     c_{1,2}(rr^+) = 1;
3 else
4     foreach i ∈ {1, 2, ..., p} do
5         compute an almost locally irregular 2-edge-colouring c^i_{1,2} of T_r[r^+, i] inductively;
6     c^0_{1,2}(rr^+) = 1;
7     for every i ∈ {1, 2, ..., p}, choose c^i_{α_i,β_i} = c^i_{1,2} or c^i_{α_i,β_i} = c^i_{2,1} so that c_{1,2} = c^0_{1,2} + c^1_{α_1,β_1}
      + c^2_{α_2,β_2} + ... + c^p_{α_p,β_p} is an almost locally irregular 2-edge-colouring of T_r;
```

Algorithm 3: Construction of an almost locally irregular 2-edge-colouring $c_{1,2}$ of a shrub T_r, where $p \geq 0$ denotes the number of children of r^+.

The keystone of Algorithm 3 is Line 7. Let us prove that the claimed almost locally irregular 2-edge-colouring $c_{1,2}$ of T_r, obtained by inverting some of the $c^i_{1,2}$'s, necessarily exists.

Lemma 9.27. *The almost locally irregular 2-edge-colouring $c_{1,2}$ of T_r claimed at Line 7 necessarily exists.*

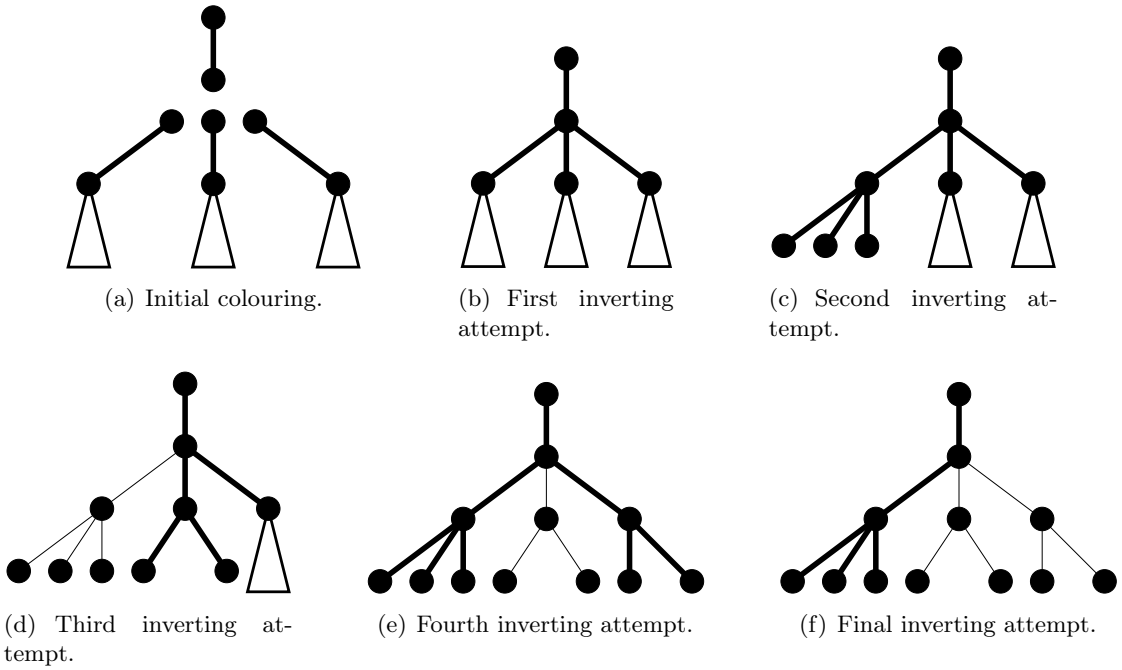
Proof. If $p = 0$, then there is nothing to prove. Thus, the node r^+ has $p \geq 1$ children $v_1, v_2, ..., v_p$ in T_r. We first consider small values of p, i.e. $p \in \{1, 2, 3\}$, before generalizing our arguments. All assumption below are made without loss of generality.

- Suppose $p = 1$. If $c_{1,2} = c^0_{1,2} + c^1_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then v_1 has 1-degree 2 in $T_r[r^+, 1]$ by $c^1_{1,2}$. Besides, the 2-edge-colouring $c^1_{1,2}$ of $T_r[r^+, 1]$ is locally irregular. Then $c_{1,2} = c^0_{1,2} + c^1_{2,1}$, obtained by inverting $c^1_{1,2}$, is an almost locally irregular 2-edge-colouring of T_r.

- Suppose $p = 2$. If $c_{1,2} = c^0_{1,2} + c^1_{1,2} + c^2_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then v_1 has 1-degree 3 in $T_r[r^+, 1]$ by $c^1_{1,2}$, and $c^1_{1,2}$ is a locally irregular 2-edge-colouring of $T_r[r^+, 1]$. Now consider $c_{1,2} = c^0_{1,2} + c^1_{2,1} + c^2_{1,2}$. If $c_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then the other child v_2 of r^+ has 1-degree 2 in $T_r[r^+, 2]$ by $c^2_{1,2}$. Moreover, the 2-edge-colouring $c^2_{1,2}$ of $T_r[r^+, 2]$ is locally irregular. It follows that $c_{1,2} = c^0_{1,2} + c^1_{1,2} + c^2_{2,1}$ is a (almost) locally irregular 2-edge-colouring of T_r.

- Suppose $p = 3$. If $c_{1,2} = c^0_{1,2} + c^1_{1,2} + c^2_{1,2} + c^3_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then v_1 has 1-degree 4 in $T_r[r^+, 1]$ by $c^1_{1,2}$, and $c^1_{1,2}$ is a locally irregular 2-edge-colouring of $T_r[r^+, 1]$. Now, if $c^0_{1,2} + c^1_{2,1} + c^2_{1,2} + c^3_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then v_2 has 1-degree 3 in $T_r[r^+, 2]$ by $c^2_{1,2}$, and $c^2_{1,2}$ is a locally irregular 2-edge-colouring of $T_r[r^+, 2]$. Again, the 1-degree of the last child v_3 of r^+ in $T_r[r^+, 3]$ by $c^3_{1,2}$ is 3 if $c^0_{1,2} + c^1_{1,2} + c^2_{2,1} + c^3_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r. Under all these assumptions, we get that $c_{1,2} = c^0_{1,2} + c^1_{1,2} + c^2_{2,1} + c^3_{2,1}$ is a (almost) locally irregular 2-edge-colouring of T_r.

By following the same scheme whenever $p \geq 4$, i.e. inverting none of the $c^i_{1,2}$'s, then one, two, three, etc., of them, we either find an almost locally irregular 2-edge-colouring $c_{1,2}$ of T_r or find out what are all of the 1-degrees of $v_1, v_2, ..., v_p$ in $T_r[r^+, 1], T_r[r^+, 2], ..., T_r[r^+, p]$, respectively, by $c^1_{1,2}, c^2_{1,2}, ..., c^p_{1,2}$, respectively. More precisely, in this last situation, we get that one of these 1-degrees is equal to $p + 1$, two of them are equal to p, three of them are equal to $p - 1$ (unless p is not big enough), and so on. Besides, note that, under our assumption on p, the biggest $\lceil \frac{p}{2} \rceil$ values of the resulting 1-degree sequence are strictly greater than $\lceil \frac{p}{2} \rceil + 1$, while its other



(a) Initial colouring. (b) First inverting attempt. (c) Second inverting attempt.

(d) Third inverting attempt. (e) Fourth inverting attempt. (f) Final inverting attempt.

Figure 9.9: Colouring scheme used in the proof of Lemma 9.27. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

values are strictly greater than $\lfloor \frac{p}{2} \rfloor$. Considering that the 1-degrees of $v_1, v_2, ..., v_p$ are ordered decreasingly, i.e. v_1 has 1-degree $p+1$, v_2 has 1-degree p, and so on, the 2-edge-colouring

$$c_{1,2} = c^0_{1,2} + c^1_{1,2} + ... + c^{\lceil \frac{p}{2} \rceil}_{1,2} + c^{\lceil \frac{p}{2} \rceil + 1}_{2,1} + c^{\lceil \frac{p}{2} \rceil + 2}_{2,1} + ... + c^p_{2,1},$$

obtained by inverting the last $\lfloor \frac{p}{2} \rfloor$ almost locally irregular 2-edge-colourings, is an almost locally irregular 2-edge-colouring of T_r since r^+ then has 1- and 2-degree $\lceil \frac{p}{2} \rceil + 1$ and $\lfloor \frac{p}{2} \rfloor$, respectively. ■

Example 9.28. Figure 9.9 depicts an example of the inversion procedure led in Lemma 9.27 on a shrub T_r whose vertex r^+ has three children v_1, v_2, v_3. Almost locally irregular 2-edge-colourings $c^1_{1,2}$, $c^2_{1,2}$ and $c^3_{1,2}$ of $T_r[r^+, 1]$, $T_r[r^+, 2]$ and $T_r[r^+, 3]$, respectively, are first computed (Figure 9.9.a). We then combine all these almost locally irregular 2-edge-colourings, and set $c^0_{1,2}(rr^+) = 1$ (Figure 9.9.b). If this 2-edge-colouring $c^0_{1,2} + c^1_{1,2} + c^2_{1,2} + c^3_{1,2}$ of T_r is not almost locally irregular, then we reveal that v_1 has 1-degree 4 in $T_r[r^+, 1]$ by $c^1_{1,2}$ (Figure 9.9.c). Now, if $c^0_{1,2} + c^1_{2,1} + c^2_{1,2} + c^3_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then we reveal that v_2 has 1-degree 3 in $T_r[r^+, 2]$ by $c^2_{1,2}$ (Figure 9.9.d). In turn, if $c^0_{1,2} + c^1_{1,2} + c^2_{2,1} + c^3_{1,2}$ is not an almost locally irregular 2-edge-colouring of T_r, then we get that v_3 has 1-degree 3 in $T_r[r^+, 3]$ by $c^3_{1,2}$ (Figure 9.9.e). Now that all the 1-degrees of the v_i's are revealed, it is clear that $c^0_{1,2} + c^1_{1,2} + c^2_{2,1} + c^3_{2,1}$ is an almost locally irregular 2-edge-colouring of T_r (Figure 9.9.f).

Using Algorithm 3, and by Lemma 9.27, we get:

Theorem 9.29. *Every shrub admits an almost locally irregular 2-edge-colouring.*

9.3.2.3 From shrubs to trees

Now consider the following procedure based on Algorithm 3 for possibly computing a locally irregular 2-edge-colouring of a colourable tree T. In this procedure, the node r of T has degree $p \geq 1$.

Inverting Procedure. *Start by decomposing T_r into the p shrubs $T_r[r, 1]$, $T_r[r, 2]$, ..., $T_r[r, p]$ and, then, compute almost locally irregular 2-edge-colourings $c^1_{1,2}, c^2_{1,2}, ..., c^p_{1,2}$ of $T_r[r, 1]$, $T_r[r, 2]$,*

..., $T_r[r,p]$, respectively. Finally, invert some of the $c_{1,2}^i$'s so that their union is a locally irregular 2-edge-colouring of T_r.

Recall that the almost locally irregular 2-edge-colourings of the shrubs attached to r exist according to Theorem 9.29. The success of the Inverting Procedure is not guaranteed since, in special cases, inverting the $c_{1,2}^i$'s in every possible way does not lead to a locally irregular 2-edge-colouring of T_r. However, the more children r has, the more possible ways for inverting the $c_{1,2}^i$'s there are. Hence, the choice of r for rooting T before applying the Inverting Procedure is crucial. Because the number of possibilities for inverting the $c_{1,2}^i$'s grows exponentially in front of $d(r)$, the Inverting Procedure actually yields a locally irregular 2-edge-colouring of T_r whenever $d(r) \geq 5$.

Theorem 9.30. *Let T be a colourable tree. If $\Delta(T) \geq 5$, then $\chi'_{irr}(T) \leq 2$.*

Proof. Let r be a node of T with $p \geq 5$ neighbours $v_1, v_2, ..., v_p$. Let further $c_{1,2}^1, c_{1,2}^2, ..., c_{1,2}^p$ be almost locally irregular 2-edge-colourings of $T_r[r,1], T_r[r,2], ..., T_r[r,p]$, respectively, which necessarily exist according to Theorem 9.29. Consider the successive 2-edge-colourings $c_{1,2}$ of T_r obtained by inverting none, one, two, and so on, of the $c_{1,2}^i$'s, i.e. we exhaustively apply the Inverting Procedure. If, at a step, the 2-edge-colouring $c_{1,2}$ is locally irregular, then the claim is true for T. Otherwise, at each step, a conflict arises because, for at least one child v_i of r, the 1- (or 2-) degree of r by $c_{1,2}$ and the 1-degree of v_i by $c_{1,2}^i$ are the same.

A careful application of the Inverting Procedure can actually allow us to deduce more refined information when a conflict occurs. In particular, if the 2-edge-colouring obtained by inverting none of the $c_{1,2}^i$'s is not a locally irregular 2-edge-colouring of T_r, then we reveal that at least one of the v_i's has 1-degree p. Similarly, if all of the 2-edge-colourings of T_r obtained by inverting one of the $c_{1,2}^i$'s are not locally irregular, then we reveal that at least two of the v_i's have 1-degree $p-1$. If all of the 2-edge-colourings of T_r obtained by inverting two of the $c_{1,2}^i$'s are not locally irregular, then we reveal that at least three of the v_i's have 1-degree $p-2$. And so on. New 1-degrees keep on being revealed because no conflict involving the 2-degrees can arise. We stop the procedure once all of the 1-degrees have been revealed (unless a locally irregular 2-edge-colouring of T_r is found at some step).

If no locally irregular 2-edge-colouring has been obtained once the Inverting Procedure has stopped, we get that the 1-degree sequence is

$$(p, p-1, p-1, p-2, p-2, p-2, ...),$$

where the value $p-k$ appears exactly $k+1$ times, except maybe in the case where $p-k$ is the last value of the sequence. When $p \geq 5$, each of the 1-degrees is strictly greater than $\lfloor \frac{p}{2} \rfloor$. Besides the first $\lceil \frac{p}{2} \rceil$ greatest 1-degrees are strictly greater than $\lceil \frac{p}{2} \rceil$. Hence, if the 1-degrees of $v_1, v_2, ..., v_p$ are ordered decreasingly, then the 2-edge-colouring

$$c_{1,2} = c_{1,2}^1 + c_{1,2}^2 + ... + c_{1,2}^{\lceil \frac{p}{2} \rceil} + c_{2,1}^{\lceil \frac{p}{2} \rceil + 1} + c_{2,1}^{\lceil \frac{p}{2} \rceil + 2} + ... + c_{2,1}^p$$

of T_r, obtained by inverting the last $\lfloor \frac{p}{2} \rfloor$ $c_{1,2}^i$'s, is locally irregular since the 1- and 2-degrees of r are then $\lceil \frac{p}{2} \rceil$ and $\lfloor \frac{p}{2} \rfloor$, respectively, which are strictly less than the 1- and 2-degrees of its neighbours in the 1- and 2-subgraphs, respectively. ■

The proof of Theorem 9.30 can actually be modified to produce an alternate proof of Theorem 9.14.

Alternate proof of Theorem 9.14. If we have $\Delta(T) \leq 2$ or $\Delta(T) \geq 5$, then $\chi'_{irr}(T) \leq 2$ according to Proposition 9.10 or Theorem 9.30, respectively. Let us thus suppose that $\Delta(T) \in \{3, 4\}$, and let r be a node of T with degree $p = \Delta(T)$ whose neighbours are denoted by $v_1, v_2, ..., v_p$. As in the proof of Theorem 9.30, let $c_{1,2}^1, c_{1,2}^2, ..., c_{1,2}^p$ be almost locally irregular 2-edge-colourings of

$T_r[r,1], T_r[r,2], ..., T_r[r,p]$ (these exist according to Theorem 9.29), respectively, and try out the Inverting Procedure. If no locally irregular 2-edge-colouring $c_{1,2}$ of T_r can be found, then the revealed 1-degree sequence is necessarily $(3,2,2)$ when $p=3$, or $(4,3,3,2)$ when $p=4$ (up to permutation). Assuming that the 1-degrees of $v_1, v_2, ..., v_p$ are ordered decreasingly, the 3-edge-colouring $c_{1,2} = c^1_{1,2} + c^2_{2,1} + c^3_{3,1}$ of T_r is locally irregular when $p=3$ since r thus has 1-, 2- and 3-degree 1 while its neighbours have degree strictly greater than 1 in the 1-, 2- and 3-subgraphs. When $p=4$, a locally irregular 3-edge-colouring of T_r is e.g. $c_{1,2} = c^1_{1,2} + c^2_{2,1} + c^3_{2,1} + c^4_{3,1}$ since r then has 1-, 2- and 3-degree 1, 2, and 1, respectively, while its neighbours have 1-degree 4, 2-degree 3, and 3-degree 2. ■

9.3.2.4 Characterization of trees with irregular chromatic index 3

We now focus on trees with maximum degree 3 or 4. In our alternate proof of Theorem 9.14, we have pointed out that the Inverting Procedure does not always provide a locally irregular 2-edge-colouring of a tree T_r. This namely occurs when inverting the $c^i_{1,2}$'s in every possible manner never yields a locally irregular 2-edge-colouring. A simple computation shows that such a situation occurs if and only if the 1-degree sequence of the v_i's in the $T_r[r,i]$'s by the $c^i_{1,2}$'s is (1), $(2,1)$, $(3,2,2)$, or $(4,3,3,2)$, when $p = d(r)$ is 1, 2, 3 or 4, respectively. In what follows, we call these four 1-degree sequences *bad*.

Definition 9.31. Assume $c^1_{1,2}, c^2_{1,2}, ..., c^p_{1,2}$ are almost locally irregular 2-edge-colourings of $T_r[r,1]$, $T_r[r,2]$, ..., $T_r[r,p]$, respectively, for a rooted tree T_r whose root's children are denoted v_1, v_2, ..., v_p (where $v_i = r^+$ in $T_r[r,i]$ for every $i \in \{1,2,...,p\}$). The 1-degree sequence

$$(d_{c^1_{1,2},1}(v_1), d_{c^2_{1,2},1}(v_2), ..., d_{c^p_{1,2},1}(v_p))$$

is *bad* whenever it is either (1), $(2,1)$, $(3,2,2)$, or $(4,3,3,2)$ (up to permutation).

Consequently, if there exist almost locally irregular 2-edge-colourings $c^1_{1,2}, c^2_{1,2}, ..., c^p_{1,2}$ of $T_r[r,1]$, $T_r[r,2]$, ..., $T_r[r,p]$, respectively, leading to a 1-degree sequence which is not bad, then we can necessarily invert some of the $c^i_{1,2}$'s to get a locally irregular 2-edge-colouring of T_r. We thus now focus on the structure of those shrubs T_r with maximum degree at most 4 in which r^+ has the same 1-degree by all of the possible almost locally irregular 2-edge-colourings of T_r. This yields the following definition.

Definition 9.32. Let $k \geq 1$. A shrub T_r is *k-bad* if r^+ has 1-degree k by every almost locally irregular 2-edge-colouring $c_{1,2}$ of T_r.

Deciding whether a shrub is k-bad can be decided via a bottom-up algorithm in the line of the strategy used in Algorithm 3. For this purpose, we need to introduce the notion of *signature*.

Definition 9.33. Let T_r be a shrub whose node r^+ has $p \geq 0$ children $v_1, v_2, ..., v_p$ (where $v_i = (r^+)^+$ in $T_r[r^+,i]$ for every $i \in \{1,2,...,p\}$). For every node v_i, we denote by D_i the set of all possible 1-degrees of v_i in $T_r[r^+,i]$ by an almost locally irregular 2-edge-colouring $c_{1,2}$ of $T_r[r^+,i]$. The *signature* of T_r is the p-tuple $(D_1, D_2, ..., D_p)$. Analogously, we denote by D_0 the set of all possible 1-degrees of r^+ by an almost locally irregular 2-edge-colouring $c_{1,2}$ of T_r.

Refer to Figure 9.10 to see to the weighted degrees of which nodes of a shrub T_r the sets D_0 and $D_1, D_2, ..., D_{d(r^+)-1}$ refer. According to Definition 9.33, note that a shrub T_r is k-bad if and only if, regarding its signature $(D_1, D_2, ..., D_{d(r^+)-1})$, we have $D_0 = \{k\}$. In such a situation, we call $(D_1, D_2, ..., D_{d(r^+)-1})$ *k-bad*.

Definition 9.34. Let $k \geq 1$. A signature $(D_1, D_2, ..., D_{d(r^+)-1})$ of a shrub T_r is *k-bad* whenever it causes $D_0 = \{k\}$.

As mentioned above, the set D_0 of a given shrub T_r can easily be computed by applying an inductive scheme inspired by Algorithm 3. Roughly explained, we first compute inductively the

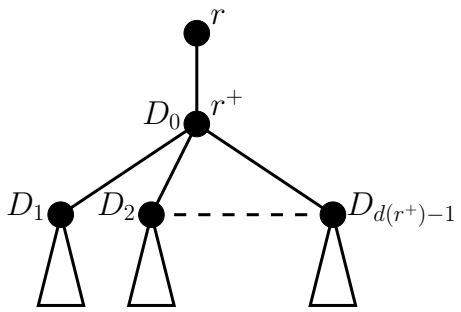


Figure 9.10: Sets $D_0, D_1, D_2, ..., D_p$ of a shrub T_r whose node r^+ has $p = d(r^+)-1$ children.

$\mathbf{D_0 = \{k\}}$	**p**	**k-bad signatures**
$\{1\}$	0	-
	1	$(\{2\})$
$\{2\}$	1	$(\{1\})$
	2	$(\{2\},\{3\})$
	3	$(\{3\},\{3\},\{4\})$
$\{3\}$	2	$(\{2\},\{2\})$
	3	$(\{2\},\{3\},\{4\})$
$\{4\}$	3	$(\{2\},\{3\},\{3\}$

Table 9.11: List of all k-bad signatures for a shrub T_r (where $p = d(r^+) - 1$).

set D_0 of each of the p shrubs $T_r[r^+, 1], T_r[r^+, 2], ..., T_r[r^+, d(r^+) - 1]$. By definition, the set D_0 of $T_r[r^+, i]$ corresponds to the set D_i of T_r. Knowing the signature $(D_1, D_2, ..., D_{d(r^+)-1})$ of T_r, the set D_0 of T_r can then be deduced. Using this procedure, we are able to identify, in the next result, all k-bad signatures with $k \in \{1, 2, 3, 4\}$.

Theorem 9.35. *All k-bad signatures with $k \in \{1, 2, 3, 4\}$ are those given in Table 9.11.*

Proof. We consider every possible signature of T_r with regards to $p \leq 3$, the number of children of r^+. For the sake of simplicity, we here only detail the proof for the easy cases, i.e. $p = 0$ and $p = 1$, so that the reader gets an idea of the technique we use. The remaining canonical cases, i.e. for $p = 2$ and $p = 3$, are given in Tables 9.12 and 9.13. Signatures in bold are those which are k-bad for some k. All cases which do not appear in these tables do not concern bad signatures and can be deduced from the canonical cases by using the following two rules which are easy to check.

Inclusion Rule: if $(D_1, D_2, ..., D_p)$ is not a bad signature of T_r, then $(D'_1, D'_2, ..., D'_p)$ is not a bad signature whenever $D_i \subseteq D'_i$ for every $i \in \{1, 2, ..., p\}$.

Union Rule: if $(D_1, D_2, ..., D_{i-1}, D_i, D_{i+1}, ..., D_p)$ is a k-bad signature and $(D_1, D_2, ..., D_{i-1}, D'_i, D_{i+1}, D_{i+2}, ..., D_p)$ is a k'-bad signature with $k' \neq k$ for some $D'_i \neq D_i$, then $(D_1, ..., D_{i-1}, D_i \cup D'_i, D_{i+1}, D_{i+2}, ..., D_p)$ is not a bad signature.

If $p = 0$, then rr^+ has to be coloured 1 and r^+ thus necessarily has 1-degree 1. Therefore, the empty signature is a 1-bad signature. Now suppose that $p = 1$ and denote v_1 the child of r^+. If $D_1 = \{1\}$, then, in every almost locally irregular 2-edge-colouring of $T_r[r^+, 1]$, the vertex v_1 has 1-degree 1 and we have to colour rr^+ with colour 1. Thus $D_0 = \{2\}$, and $(\{1\})$ is a 2-bad signature. Similarly, if $D_1 = \{2\}$, then every almost locally irregular 2-edge-colouring of $T_r[r^+, 1]$ is actually locally irregular and we have to invert it before colouring rr^+ with colour 1. Therefore, $(\{2\})$ is a 1-bad signature. If there exists an almost locally irregular 2-edge-colouring

Signature (D_1, D_2)	**Resulting** $\mathbf{D_0}$
$(\{1\},\{1\})$	$\{1,3\}$
$(\{1\},\{2\})$	$\{2,3\}$
$(\{1\},\{3\})$	$\{1,2\}$
$(\{1\},\{4\})$	$\{1,2,3\}$
$(\{\mathbf{2}\},\{\mathbf{2}\})$	$\{\mathbf{3}\}$

Signature (D_1, D_2)	**Resulting** $\mathbf{D_0}$
$(\{\mathbf{2}\},\{\mathbf{3}\})$	$\{\mathbf{2}\}$
$(\{2\},\{4\})$	$\{2,3\}$
$(\{3\},\{3\})$	$\{1,2\}$
$(\{3\},\{4\})$	$\{1,2\}$
$(\{4\},\{4\})$	$\{1,2,3\}$

Table 9.12: All possible canonical signatures for a shrub T_r and resulting sets D_0 when $p = 2$ (where $p = d(r^+) - 1$).

Signature (D_1, D_2, D_3)	**Resulting** $\mathbf{D_0}$
$(\{1\},\{1\},\{1\})$	$\{1,2,4\}$
$(\{1\},\{1\},\{2\})$	$\{1,3,4\}$
$(\{1\},\{1\},\{3\})$	$\{2,3,4\}$
$(\{1\},\{1\},\{4\})$	$\{1,2,3\}$
$(\{1\},\{2\},\{2\})$	$\{1,3,4\}$
$(\{1\},\{2\},\{3\})$	$\{3,4\}$
$(\{1\},\{2\},\{4\})$	$\{1,3\}$
$(\{1\},\{3\},\{3\})$	$\{2,4\}$
$(\{1\},\{3\},\{4\})$	$\{2,3\}$
$(\{1\},\{4\},\{4\})$	$\{1,2,3\}$

Signature (D_1, D_2, D_3)	**Resulting** $\mathbf{D_0}$
$(\{2\},\{2\},\{2\})$	$\{1,3,4\}$
$(\{2\},\{2\},\{3\})$	$\{3,4\}$
$(\{2\},\{2\},\{4\})$	$\{1,3\}$
$(\{\mathbf{2}\},\{\mathbf{3}\},\{\mathbf{3}\})$	$\{\mathbf{4}\}$
$(\{\mathbf{2}\},\{\mathbf{3}\},\{\mathbf{4}\})$	$\{\mathbf{3}\}$
$(\{2\},\{4\},\{4\})$	$\{1,3\}$
$(\{3\},\{3\},\{3\})$	$\{2,4\}$
$(\{\mathbf{3}\},\{\mathbf{3}\},\{\mathbf{4}\})$	$\{\mathbf{2}\}$
$(\{3\},\{4\},\{4\})$	$\{2,3\}$
$(\{4\},\{4\},\{4\})$	$\{1,2,3\}$

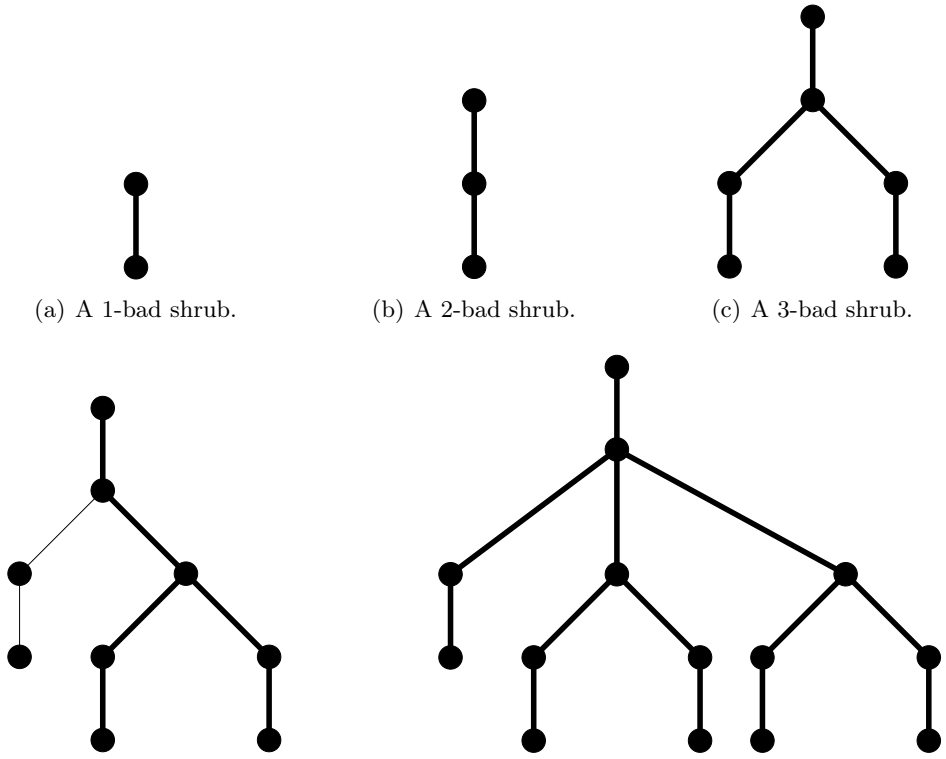
Table 9.13: All possible canonical signatures for a shrub T_r and resulting sets D_0 when $p = 3$ (where $p = d(r^+) - 1$).

$c^1_{1,2}$ of $T_r[r^+, 1]$ such that v_1 has 1-degree 3 or 4, then we may either colour rr^+ with colour 1 directly, or invert $c^1_{1,2}$ before. In the first situation, the vertex r^+ has 1-degree 2, while it has 1-degree 1 in the second situation. Therefore, we have $\{1, 2\} \subseteq D_0$ whenever 3 or 4 belongs to D_1. Thus, there is no 1-bad signature involving a set containing either 3 or 4. Finally, the signature $(\{1, 2\})$ is not bad since we get $\{1, 2\} \subseteq D_0$ by the Union Rule. Every other possibility for D_1 leads to a D_0 which is not a singleton by the Inclusion and Union Rules. Therefore, $(\{1\})$ and $(\{2\})$ are the only bad signatures when $p = 1$. ■

Arbitrarily many k-bad shrubs can be constructed from Theorem 9.35 by connecting "bad pieces" together. First choose a k-bad signature, i.e. let p and $D_1 = \{d_1\}, D_2 = \{d_2\}, ..., D_p = \{d_p\}$ be sets such that $(D_1, D_2, ..., D_p)$ corresponds to one row of Table 9.11. Now consider a single edge rr^+, as well as d_1-,d_2-, ..., d_p-bad shrubs $T_1, T_2, ..., T_p$, respectively. Then identify the roots of $T_1, T_2, ..., T_p$ with r^+. The resulting shrub T_r is clearly k-bad by definition.

Example 9.36. Successive bad shrubs obtained in this way are depicted in Figure 9.14. The base case is the 1-bad shrub from Figure 9.14.a. According to Theorem 9.35, the only bad shrubs made up of 1-shrubs only are 2-bad shrubs obtained from two 1-bad shrubs, see Figure 9.14.b. Now using two such 2-bad shrubs, we construct the 3-bad shrub depicted in Figure 9.14.c. Using the 2-bad and the 3-bad shrubs we have constructed, we obtain the 2-bad shrub of Figure 9.14.d. Finally, using the 2-bad shrub from Figure 9.14.b and two copies of the 3-bad shrub from Figure 9.14.c, we obtain the 4-bad shrub depicted in Figure 9.14.e.

Suppose r has $p \geq 1$ neighbours in a colourable tree T. As explained above, if the shrubs $T_r[r, 1], T_r[r, 2], ..., T_r[r, p]$ are k_1-, k_2-, ..., k_p-bad, respectively, and the sequence $(k_1, k_2, ..., k_p)$ is one of the bad 1-degree sequences (1), $(2, 1)$, $(3, 2, 2)$ or $(4, 3, 3, 2)$ (up to permutation), then we cannot deduce a locally irregular 2-edge-colouring of T_r by applying the Inverting Procedure introduced in Section 9.3.2.3. In this situation, we say that r is *bad*.



(a) A 1-bad shrub. (b) A 2-bad shrub. (c) A 3-bad shrub.

(d) A 2-bad shrub. (e) A 4-bad shrub.

Figure 9.14: Some bad shrubs (whose roots are the top-most nodes), and almost locally irregular 2-edge-colourings of these. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

Definition 9.37. A node r from a tree T is said *bad* whenever the Inverting Procedure fails on T_r, i.e. does not yield a locally irregular 2-edge-colouring of T_r.

One could think that the choice of r is crucial in the sense that if the Inverting Procedure fails on T_r, i.e. r is bad, then maybe it can work for $T_{r'}$ with $r \neq r'$. We show below that it is actually not the case as if r is bad, then so is every other node r' of T. This implies that $\chi'_{irr}(T) = 3$ if and only if any node of T is bad.

First remark, by comparing the bad 1-degree sequences and the bad signatures from Table 9.11, that the following holds.

Observation 9.38. *If $(\{d_1\}, \{d_2\}, ..., \{d_p\})$ is a d_0-bad signature, then $(d_0, d_1, d_2, ..., d_p)$ is a bad 1-degree sequence. Conversely, if σ is any permutation of $\{d_0, d_1, d_2, ..., d_p\}$ and $(d_0, d_1, d_2, ..., d_p)$ is a bad 1-degree sequence, then $(\{\sigma(d_1)\}, \{\sigma(d_2)\}, ..., \{\sigma(d_p)\})$ is a $\sigma(d_0)$-bad signature.*

We are now ready to prove our main result.

Theorem 9.39. *If r is a bad node of a tree T, then so is every other node $r' \neq r$ of T.*

Proof. Note that it suffices to show the claim when r and r' are neighbours in T. Suppose that $p \geq 1$ and $p' \geq 1$ denote the degrees of r and r', respectively, and r' (resp. r) is the first child of r (resp. r') in T_r (resp. $T_{r'}$), i.e. $r' = r^+$ (resp. $r = (r')^+$) in $T_r[r, 1]$ (resp. $T_{r'}[r', 1]$).

Because r is bad, the shrubs $T_r[r, 1], T_r[r, 2], ..., T_r[r, p]$ are k_1-, k_2-, ..., k_p-bad, respectively, and $(k_1, k_2, ..., k_p)$ is a bad 1-degree sequence. According to Theorem 9.35, if $T_r[r, 1]$ is k_1-bad, then $T_r[r', 1], T_r[r', 2], ..., T_r[r', p' - 1]$ are ℓ_1-, ℓ_2-, ..., $\ell_{p'-1}$-bad, respectively, and $(\{\ell_1\}, \{\ell_2\}, ..., \{\ell_{p'-1}\})$ is a k_1-bad signature. Now, because r is bad, it means that $(\{k_2\}$, $\{k_3\}$, ..., $\{k_p\})$ is a k_1-bad signature again by Observation 9.38 and $T_{r'}[r', 1]$ is a k_1-bad shrub.

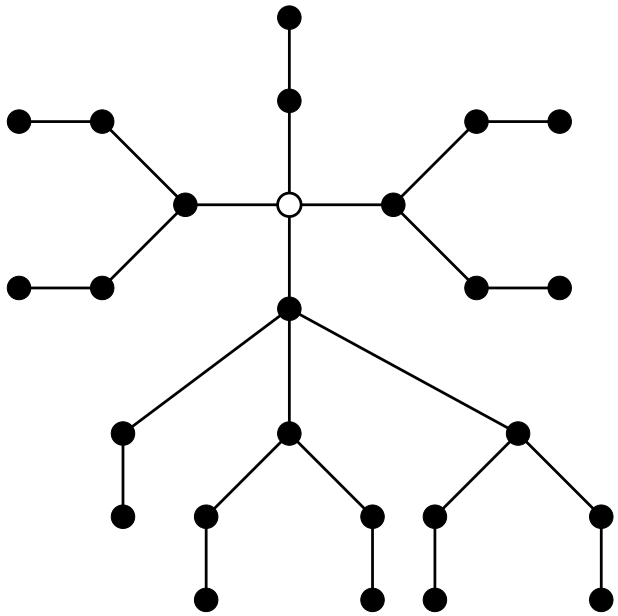


Figure 9.15: A tree with maximum degree 4 and irregular chromatic index 3.

Thus, the shrubs $T_{r'}[r', 1], T_{r'}[r', 2], ..., T_{r'}[r', p']$ are k_1-, ℓ_1-, ℓ_2-, ..., $\ell_{p'-1}$-bad, respectively, and $(k_1, \ell_1, \ell_2, ..., \ell_{p'-1})$ is a bad 1-degree sequence. It follows that r' is bad. ■

Corollary 9.40. *For every tree T, we have $\chi'_{irr}(T) = 3$ if and only if any node of T is bad.*

Every tree with irregular chromatic index 3 can hence be constructed as follows. First choose one of the bad 1-degree sequences $(d_1, d_2, ..., d_p)$, and construct p shrubs $T_1, T_2, ..., T_p$ which are d_1-, d_2-, ..., d_p-bad, respectively. Recall that there are infinitely many such shrubs as pointed out above. Finally identify the roots of $T_1, T_2, ..., T_p$. By construction, the node resulting from the identification is bad, and the obtained tree thus has irregular chromatic index 3 according to Corollary 9.40.

Example 9.41. In Figure 9.15 is depicted a tree T obtained by identifying the roots of one 4-bad shrub, two 3-bad shrubs, and one 2-bad shrub (exhibited in Figure 9.14). The node r resulting from the identification is the white one. Since $(4, 3, 3, 2)$ is a bad 1-degree sequence, the node r is bad, and hence T has irregular chromatic index 3 according to Corollary 9.40.

9.3.2.5 A linear-time algorithm for the irregular chromatic index of trees

We now propose an algorithm that determines, based on our previous results, the irregular chromatic index of an input tree T.

Theorem 9.42. *Algorithm 4 determines the irregular chromatic index of a tree T in time $\mathcal{O}(|V(T)|)$.*

Proof. As a first remark, recall that T is an exception if and only if T is an odd length path, so Line 1 has time complexity $\mathcal{O}(|V(T)|)$. Besides, because T is a tree, Line 2 can be achieved within time $\mathcal{O}(|V(T)|)$ by applying a depth-first search algorithm. The correctness of the next instructions of Algorithm 4, i.e. the determination of whether the irregular chromatic index is 2 or 3, then follows from the previous results and observations. In particular, the correctness of Lines 5-6 follows from Theorem 9.30, while the correctness of Lines 11-12 and Lines 15-16 follows from observations raised in Section 9.3.2.4. The correctness of Lines 17-18 follows from Corollary 9.40. The most costly instruction of Algorithm 4 is Line 10, which is achieved in time $\mathcal{O}(|V(T)|)$ by computing the values of D_0 from the leaves to the root r of T_r (where r is chosen

```
1  if T is an exception then
2      χ'_irr(T) is undefined;
3  else if T is locally irregular then
4      χ'_irr(T) = 1;
5  else if Δ(T) ≤ 2 or Δ(T) ≥ 5 then
6      χ'_irr(T) = 2;
7  else
8      choose an arbitrary node r of T with degree p ≥ 1;
9      foreach i ∈ {1, 2, ..., p} do
10         let D_i denote the set D_0 of T_r[r, i] computed inductively;
11         if D_i is not a singleton then
12             χ'_irr(T) = 2;
13             exit algorithm;
14     let D_i = {d_i} for every i ∈ {1, 2, ..., p};
15     if (d_1, d_2, ..., d_p) is not a bad 1-degree sequence then
16         χ'_irr(T) = 2;
17     else
18         χ'_irr(T) = 3;
```

Algorithm 4: Determining the irregular chromatic index of a tree T.

arbitrarily) for each shrub as in the proof of Theorem 9.35. Every other line of the algorithm runs in time either $\mathcal{O}(1)$ or $\mathcal{O}(|V(T)|)$. Therefore, we get that Algorithm 4 has running time $\mathcal{O}(|V(T)|)$. ■

Regarding the main concern of this section, as a direct corollary of Theorem 9.42, we get the following.

Corollary 9.43. Locally Irregular 2-Edge-Colouring *is in* P *when restricted to trees.*

9.3.3 General graphs

This section is devoted to the following complexity result.

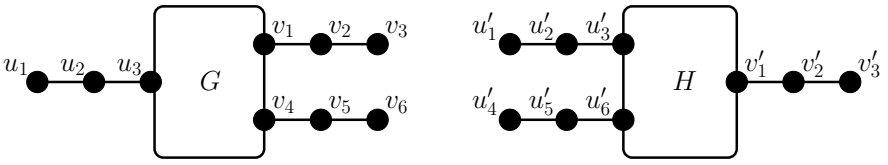
Theorem 9.44. Locally Irregular 2-Edge-Colouring *is* NP*-complete, even when restricted to planar graphs with maximum degree at most 6.*

Proof. Given a 2-edge-colouring c of a graph G, one can check whether the subgraphs of G induced by the two colours of c are locally irregular. Since checking whether a graph is locally irregular can be done in quadratic time and c uses a fixed number of colours, checking whether c is locally irregular can be done in quadratic time. Therefore, the Locally Irregular 2-Edge-Colouring problem is in NP.

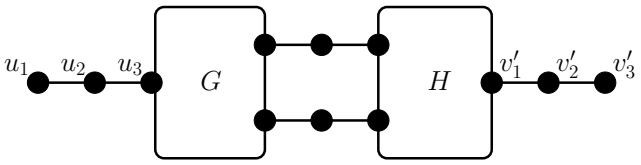
We now prove the NP-hardness of Locally Irregular 2-Edge-Colouring. This is done by reduction from 1-in-3 Satisfiability. From a formula F in conjunctive normal form involving 3-clauses, we produce a graph G_F such that

$$F \text{ is 1-in-3 satisfiable}$$
$$\Leftrightarrow$$
$$G_F \text{ admits a locally irregular 2-edge-colouring } c_F.$$

The reduction used herein is essentially the same as the one used to prove Theorem 8.1. We thus refer the reader to Section 8.2 wherein all the details on how the reduction works are



(a) A graph G with input $u_1u_2u_3$ and two outputs $v_1v_2v_3$ and $v_4v_5v_6$, and a graph H with two inputs $u'_1u'_2u'_3$ and $u'_4u'_5u'_6$ and output $v'_1v'_2v'_3$.



(b) The connection of G and H along $(v_1v_2v_3, v_4v_5v_6)$ and $(u'_1u'_2u'_3, u'_4u'_5u'_6)$.

Figure 9.16: Illustration of the connection operation.

clarified. Since there is no systematic relationship between a neighbour-sum-distinguishing 2-edge-weighting and a locally irregular 2-edge-colouring, and the reduction is not performed from the same problem, some details of the reduction have to be modified though.

To begin with, an *input* (or *output*) of a graph G now designates a path uvw with length 2 such that u (resp. w) has degree 1, and v has degree 2. Assuming G has x (resp. y) inputs (resp. outputs), we denote these by $I_1(G), I_2(G), ..., I_x(G)$ (resp. $O_1(G), O_2(G), ..., O_y(G)$) for convenience (where, again, the ordering is arbitrary). If c is a locally irregular edge-colouring of G, then by writing $c(I_i(G)) = j$ or $c(O_i(G)) = j$ we mean that the two edges constituting the ith input or output, respectively, of G both receive colour j by c. By *connecting two graphs along an output $u_1v_1w_1$ and an input $u_2v_2w_2$*, we mean that we identify the edges u_1v_1 and u_2v_2, and v_1w_1 and v_2w_2. This is similar to identifying u_1 and u_2, then v_1 and v_2, and w_1 and w_2.

Example 9.45. Figure 9.16 illustrates the connection of two graphs G and H along two outputs $v_1v_2v_3$ and $v_4v_5v_6$ of G and two inputs $u'_1u'_2u'_3$ and $u'_4u'_5u'_6$ of H. The inputs and outputs of the resulting graph are the ones of G and H which have not been used for the connection, namely its input is the input $u_1u_2u_3$ of G and its output is the outputs $v'_1v'_2v'_3$ of H.

We now introduce the necessary spreading, clause and collecting gadgets in next sections, with each time pointing out the possible differences with the reduction from Chapter 8.

Spreading gadget $G^{\curlywedge}$ and generator gadget

We start by exhibiting one spreading gadget, where a spreading gadget still designates a graph whose unique input and two outputs necessarily have the same colour by a locally irregular 2-edge-colouring. Consider, as $G^{\curlywedge}$, the graph depicted in Figure 9.17, whose input is $I_1(G^{\curlywedge}) = u_1u_2u_3$, and two outputs are $O_1(G^{\curlywedge}) = u_{10}u_{11}u_{12}$ and $O_2(G^{\curlywedge}) = u_{19}u_{20}u_{21}$. We show that $G^{\curlywedge}$ is a spreading gadget.

Proposition 9.46. *If c is a locally irregular 2-edge-colouring of $G^{\curlywedge}$, then we have $c(I_1(G^{\curlywedge})) = c(O_1(G^{\curlywedge})) = c(O_2(G^{\curlywedge}))$.*

Proof. We initiate c by first colouring the input of $G^{\curlywedge}$. Refer to Figure 9.17 so see how c is propagated along $G^{\curlywedge}$. Let us suppose, without loss of generality, that $c(u_1u_2) = 1$. Then we have $c(u_2u_3) = 1$ since otherwise u_1 and u_2 would be 1-neighbours in the 1-subgraph. We then have $c(u_3u_4) = c(u_3u_{13})$ since otherwise u_2 and u_3 would be 2-neighbours in the 1-subgraph.

Let us first suppose $c(u_3u_4) = c(u_3u_{13}) = 1$. Clearly, we cannot have $c(u_4u_5) = c(u_4u_6)$ since, in this case, when, colouring u_5u_6, the vertices u_5 and u_6 would be neighbouring vertices with

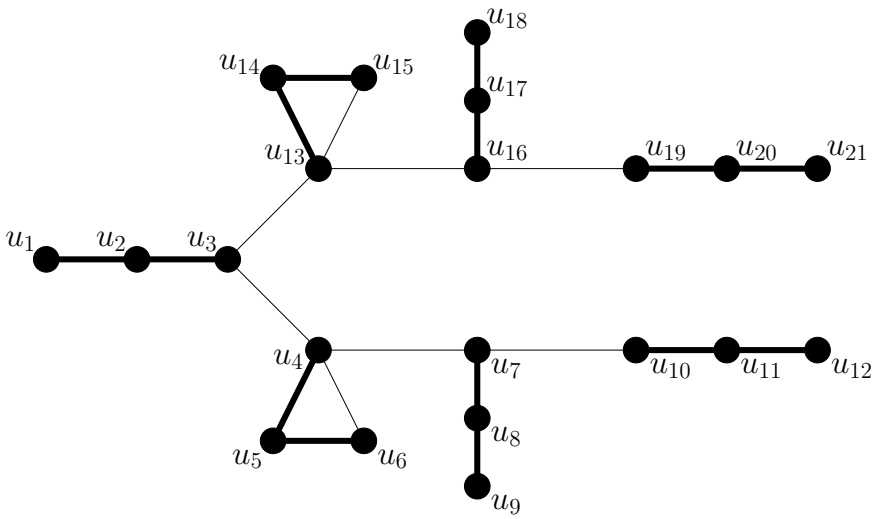


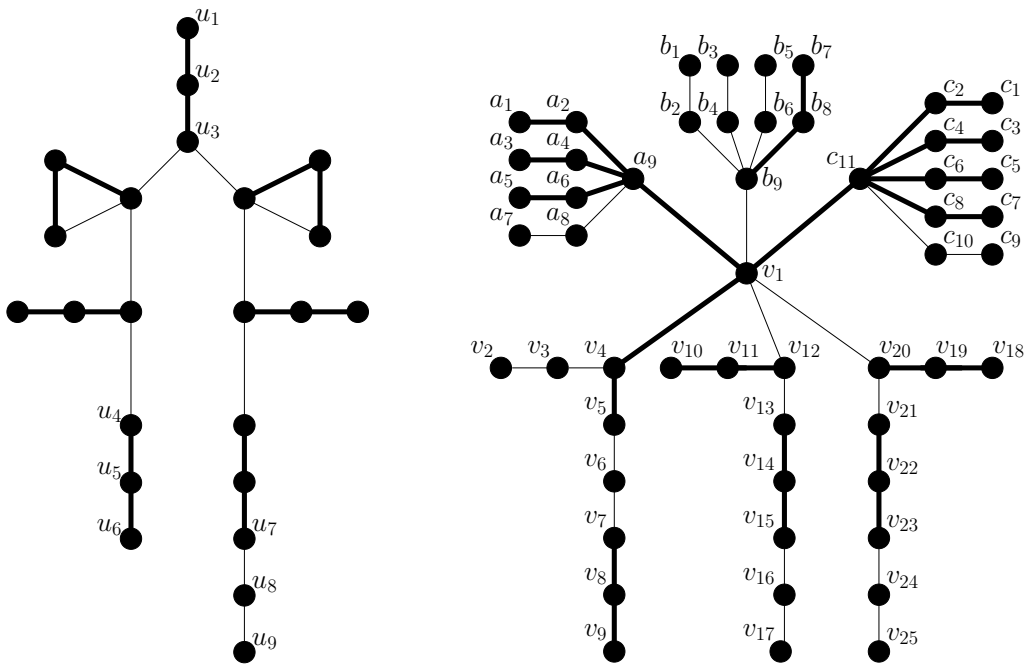
Figure 9.17: The spreading gadget $G^{\curlywedge}$, and a locally irregular 2-edge colouring of $G^{\curlywedge}$. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

the same degree in either the 1- or 2-subgraph. Suppose then $c(u_4u_5) = 1$ and $c(u_4u_6) = 2$. Now observe that if $c(u_5u_6) = 1$, then we must set $c(u_4u_7) = 2$ so that u_4 and u_6 are not neighbouring vertices with the same degree in the 2-subgraph. But note then that u_4 and u_5 are 2-neighbours in the 1-subgraph. Similarly, if $c(u_5u_6) = 2$, then we must have $c(u_4u_7) = 1$ since otherwise u_4 and u_6 would be 2-neighbours in the 2-subgraph. But u_3 and u_4 now are 3-neighbours in the 1-subgraph. Hence, we cannot extend c to $G^{\curlywedge}$ if $c(u_3u_4) = c(u_3u_{13}) = 1$.

Thus, we must have $c(u_3u_4) = c(u_3u_{13}) = 2$. For the same reasons as above, we have $c(u_4u_5) = 1$ and $c(u_4u_6) = 2$ without loss of generality. Now, if $c(u_5u_6) = 2$, then we must have $c(u_4u_7) = 2$ too so that u_4 and u_6 are not neighbouring vertices with the same degree in the 2-subgraph. But then u_4 and u_5 are 1-neighbours in the 1-subgraph. Therefore, we must set $c(u_5u_6) = 1$. To ensure that u_4 and u_3 are neighbouring vertices with distinct degrees in the 2-subgraph, we now need $c(u_4u_7) = 2$. Similarly as before, we must have $c(u_9u_8) = c(u_8u_7)$. Clearly, if $c(u_9u_8) = c(u_8u_7) = 2$, then we must have $c(u_7u_{10}) = 2$ so that u_7 and u_8 are not neighbouring vertices with the same degree in the 2-subgraph; but then u_4 and u_7 are 3-neighbours in the 2-subgraph. So $c(u_9u_8) = c(u_8u_7) = 1$, and $c(u_7u_{10}) = 2$ since otherwise u_7 and u_8 would be 2-neighbours in the 1-subgraph. Because $d_{c,2}(u_7) = 2$, we must have $c(u_{10}u_{11}) = 1$. Besides, so that u_{10} and u_{11} are neighbours with distinct degrees in the 1-subgraph, we have to set $c(u_{11}u_{12}) = 1$.

The locally irregular 2-edge colouring c is propagated to the remaining edges of $G^{\curlywedge}$ in a symmetric way. We finally get that the input and outputs of $G^{\curlywedge}$ have the same colour via c, as claimed. ■

As for the spreading gadgets used for the reduction to Neighbour-Sum-Distinguishing $\{a, b\}$-Edge-Weighting in Chapter 8, note that by connecting several copies of $G^{\curlywedge}$ consecutively, the input colour by a locally irregular edge-colouring c of the first copy of $G^{\curlywedge}$ is propagated towards an arbitrary number of outputs. Assume now we have e.g. $c(O_1(G^{\curlywedge})) = 1$, and let $P_5 = u_1u_2u_3u_4u_5$ be the path with order 5. Now let G' be the graph obtained by connecting $G^{\curlywedge}$ and $u_1u_2u_3u_4u_5$ along $O_1(G^{\curlywedge})$ and $u_1u_2u_3$. Then note that c propagates in a locally irregular way from $G^{\curlywedge}$ to G' in such a way that $c(u_3u_4u_5) = 2$. Hence, as for the latter reduction, we are able to "invert" the colour at some outputs of a spreading gadget. Combining the previous two observations, we get a generator gadget, i.e. a graph with one input and arbitrarily many outputs which are deterministically coloured 1 (such are sometimes called *positive*) or 2 (such are sometimes called *negative*) by a locally irregular 2-edge-colouring (assuming the input's colour is known, say 1), for our reduction.



(a) Case $m(C_i) = 2$. (b) Case $m(C_i) = 3$.

Figure 9.18: The two forms of the clause gadget $G_F(C_i)$, and locally irregular 2-edge colourings of $G_F(C_i)$. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

Clause gadgets $G_F(C_i)$

We now introduce the clause gadgets, where a clause gadget is a graph with a fixed number of already coloured inputs (which actually correspond to outputs of the generator gadget) and up to three outputs with the property that exactly one output is coloured 1 by every locally irregular 2-edge-colouring (respecting the constraints on the input colours). This slight difference with the reduction from Chapter 8 is necessary as we are performing the reduction from 1-IN-3 SATISFIABILITY (otherwise the equivalence with F could be not meet).

We distinguish two forms for $G_F(C_i)$ depending on whether $m(C_i) = 2$ or $m(C_i) = 3$. If $m(C_i) = 2$, then consider as $G_F(C_i)$ the graph depicted in Figure 9.18.a, whose input is $u_1u_2u_3$, which is supposed to be coloured 1, and two outputs are $u_4u_5u_6$ and $u_7u_8u_9$. If $m(C_i) = 3$, then let $G_F(C_i)$ be the graph depicted in Figure 9.18.b, whose inputs are $a_1a_2a_9$, $a_3a_4a_9$, $a_5a_6a_9$, $b_7b_8b_9$, $c_1c_2c_{11}$, $c_3c_4c_{11}$, $c_5c_6c_{11}$ and $c_7c_8c_{11}$, which are supposed to be coloured 1, and $a_7a_8a_9$, $b_1b_2b_9$, $b_3b_4b_9$, $b_5b_6b_9$ and $c_9c_{10}c_{11}$, which are supposed to be coloured 2. Its three outputs are $v_7v_8v_9$, $v_{15}v_{16}v_{17}$, and $v_{23}v_{24}v_{25}$. We show below that $G_F(C_i)$ is a clause gadget whatever is the value of $m(C_i)$.

Proposition 9.47. *Assume c is a locally irregular 2-edge-colouring of $G_F(C_i)$ with $m(C_i) \in \{2, 3\}$ such that the inputs of $G_F(C_i)$ are coloured as described above. Then exactly one output of $G_F(C_i)$ has colour 1 by c.*

Proof. Note that for $m(C_i) = 2$, the clause gadget $G_F(C_i)$ is actually a spreading gadget $G^{\curlywedge}$ whose one output was connected with a path on 5 vertices. For this value of $m(C_i)$, the claim then follows from Proposition 9.46.

Assume now that $m(C_i) = 3$. Since a_9 and a_8 are adjacent in the 2-subgraph and $d_{c,2}(a_8) = 2$, we must set $c(a_9v_1) = 1$. For the same reason, we have to set $c(b_9v_1) = 2$ and $c(c_{11}v_1) = 1$. So we get $d_{c,1}(a_9) = d_{c,2}(b_9) = 4$ and $d_{c,1}(c_{11}) = 5$. Observe that if no edge, two edges or three edges among those in $\{v_1v_4, v_1v_{12}, v_1v_{20}\}$ were coloured 1 via c and the other ones were coloured 2,

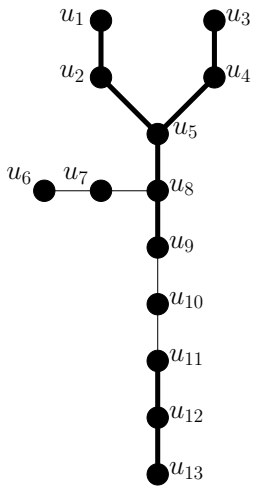


Figure 9.19: The collecting gadget $G^\curlyvee$, and a locally irregular 2-edge colouring of $G^\curlyvee$. Thick (resp. thin) edges represent 1- (resp. 2-) coloured edges.

then we would get that v_1 is neighbouring and has the same degree as one of a_9, b_9 and c_{11} in either the 1- or 2-subgraph. Therefore, exactly one edge in $\{v_1v_4, v_1v_{12}, v_1v_{20}\}$ is coloured 1 via c, while the other two are coloured 2.

Let us suppose $c(v_1v_4) = 1$ and $c(v_1v_{12}) = c(v_1v_{20}) = 2$ without loss of generality. Observe that we have $d_{c,1}(v_1) = d_{c,2}(v_1) = 3$. Once again, the edges v_2v_3 and v_3v_4 have to be coloured with the same colour, but this cannot be 1. Indeed, if $c(v_2v_3) = c(v_3v_4) = 1$, then v_3 and v_4 would be neighbours in the 1-subgraph, and would both have degree 2 in it so far. We would thus have to set $c(v_4v_5) = 1$ but then we would get that v_1 and v_4 are 3-neighbours in the 1-subgraph. So we necessarily have $c(v_2v_3) = c(v_3v_4) = 2$, and $c(v_4v_5) = 1$ since otherwise v_3 and v_4 would be 2-neighbours in the 2-subgraph. Now, because $d_{c,1}(v_4) = 2$, we need $c(v_5v_6) = c(v_6v_7) = 2$. Analogously, we have $c(v_7v_8v_9) = 1$.

Repeating the same arguments towards v_{17} and v_{25} knowing that $c(v_1v_{12}) = c(v_1v_{20}) = 2$, we get $c(v_{15}v_{16}v_{17}) = 2$ and $c(v_{23}v_{24}v_{25}) = 2$. The important thing to keep in mind is that c is not unique in the sense that the obtained colouring of the outputs of $G_F(C_i)$ mainly depends on which edge among $\{v_1v_4, v_1v_{12}, v_1v_{20}\}$ is coloured 1. This completes the claim. ■

Collecting gadget $G^\curlyvee$ and literal gadget $G_F(\ell_i)$

As in Chapter 8, a collecting gadget is a graph with two inputs and one output which are necessarily assigned the same colour by a locally irregular 2-edge-colouring. Consider, as $G^\curlyvee$, the graph depicted in Figure 9.19, whose inputs are $u_1u_2u_5$ and $u_3u_4u_5$, and output is $u_{11}u_{12}u_{13}$. There are no forcing inputs in $G^\curlyvee$, contrary to the collecting gadgets exhibited in Chapter 8. We show that $G^\curlyvee$ is a collecting gadget.

Proposition 9.48. *If c is a locally irregular 2-edge-colouring of $G^\curlyvee$, then we have $c(I_1(G^\curlyvee)) = c(I_2(G^\curlyvee)) = c(O_1(G^\curlyvee))$.*

Proof. Suppose first that $c(u_1u_2u_5) = 1$ and $c(u_3u_4u_5) = 2$ without loss of generality. Observe then that if $c(u_5u_8) = 1$, then u_2 and u_5 are 2-neighbours in the 1-subgraph. Similarly, if $c(u_5u_8) = 2$, then u_4 and u_5 are 2-neighbours in the 2-subgraph. So we cannot have $c(u_1u_2u_5) \neq c(u_3u_4u_5)$.

Let us now suppose that $c(u_1u_2u_5) = c(u_3u_4u_5) = 1$ without loss of generality. Then, we have to set $c(u_5u_8) = 1$ since otherwise u_5 would be adjacent to u_2 and u_4 in the 1-subgraph, with all these vertices having the same 1-degree. For the same reasons as previously, we necessarily have $c(u_6u_7) = c(u_7u_8)$. If this colour is 1, then we need $c(u_8u_9) = 1$ so that u_7 and u_8 are

not neighbours with the same degree in the 1-subgraph, but then u_5 and u_8 are 3-neighbours in the 1-subgraph. So, we set $c(u_6u_7) = c(u_7u_8) = 2$ and we need to set $c(u_8u_9) = 1$ since otherwise the 2-subgraph would have two adjacent vertices with degree 2. Since $d_{c,1}(u_8) = 2$, the colouring c is propagated alternatively along the path $u_9u_{10}u_{11}u_{12}u_{13}$ in such a way that $c(u_9u_{10}) = c(u_{10}u_{11}) = 2$ and $c(u_{11}u_{12}u_{13}) = 1$. ■

Every literal gadget $G_F(\ell_i)$ is obtained as in Chapter 8 by connecting several copies of $G^\curlyvee$.

Connecting the literal gadgets

As in Chapter 8, we have to check whether the outputs of every two literal gadgets $G_F(\ell_i)$ and $G_F(\overline{\ell_i})$ have distinct colours by c_F so that the equivalence is correct. This is done as follows. For each pair $\{\ell_i, \overline{\ell_i}\}$ of negated literals of F, let u and u' denote the vertices with degree 1 of $O_1(G_F(\ell_i))$ and $O_1(G_F(\overline{\ell_i}))$, respectively, and just identify u and u'.

Recall that if $c_F(O_1(G_F(\ell_i))) = j$, then the vertex resulting from the identification of u and u' is adjacent to a vertex with degree 2 in the j-subgraph, and similarly regarding the colour of $O_1(G_F(\overline{\ell_i}))$. Therefore, if $c_F(O_1(G_F(\ell_i))) = c_F(O_1(G_F(\overline{\ell_i}))) = j$, then the vertex resulting from the identification of u and u' has j-degree 2 and is adjacent to two vertices with j-degree 2 in the j-subgraph, contradicting the fact that c_F is locally irregular. For this reason, we must have $c_F(O_1(G_F(\ell_i))) \neq c_F(O_1(G_F(\overline{\ell_i})))$. In such a situation, note that there is no such contradiction.

Concluding remarks

With all the gadgets and modifications exhibited throughout this section, one can implement the reduction framework described in Section 8.2 for our concern. Since the number of different gadgets is quite small, one can easily check by hand that no conflict arises when connecting different gadgets. Because our gadgets have constant order and the number of used gadgets is polynomial with the size of F, it should be clear that the reduction is performed in polynomial time. Note further that all our gadgets are planar, and that we may suppose that F is planar, recall Theorem 1.43, so that we can have no edge crossing involving edges from the clause gadgets to the collecting gadgets. For these reasons, the reduced graph G_F can always be drawn in a planar way. It further has maximum degree at most 6 as its vertices with the largest degree are those from clause gadgets $G_F(C_i)$ whose associated values $m(C_i)$ are equal to 3.

In the light of these explanations, we get that Locally Irregular 2-Edge-Colouring is NP-complete, even when restricted to planar graphs with maximum degree at most 6. ■

9.4 Conclusion and open questions

In this chapter, we have introduced the notion of locally irregular edge-colouring of graphs and exhibited very first results on it. One important first task was to separate those graphs which admit a locally irregular edge-colouring from those which do not. This has been done in Section 9.1. The set of exceptions for locally irregular edge-colouring has appeared to be less trivial than e.g. for neighbour-sum-distinguishing edge-weighting of graphs, but can nevertheless be recognized in polynomial time, recall Theorem 9.25.

We have then raised Conjecture 9.9 and have supported it by showing it to hold for several common classes of graphs. Regarding particular classes of graphs, we have intriguingly not been able to prove Conjecture 9.9 in the context of bipartite graphs, while these graphs can be handled easily when dealing with other related edge-colouring problems.

Problem 9.49. *Prove that we have $\chi'_{irr}(G) \leq 3$ for every colourable bipartite graph G.*

Maybe the most natural idea to tackle Problem 9.49 is to proceed by induction on the size of bipartite graphs, as it has been often done to prove many edge-colouring results. But two issues make such a strategy unlikely to work in our context. The first (minor) problem is that a graph resulting from the removal of an edge uv from a bipartite graph G may not be colourable anymore. The second (major) problem is that, when extending a locally irregular edge-colouring c of G to $G + \{uv\}$, not only one has to find a colour for uv such that u and v have distinct $c(uv)$-degrees by c, but also to make sure that none of u or v has the same $c(uv)$-degree by c as one of its other neighbours (there may be a large number of them). Because of these two issues, applying this extension strategy appears quite tricky in general.

Another way for dealing with Problem 9.49 could be to investigate whether there are a lot of different bipartite graphs with irregular chromatic index 3. Namely, one could consider the following refinement of Theorem 9.44.

Question 9.50. *Is* Locally Irregular 2-Edge-Colouring *NP-complete when restricted to bipartite graphs?*

The reduction we gave in Section 9.3.3 does not answer to Question 9.50 as the reduced graphs are clearly not bipartite (the spreading gadget has triangles, and other cycles with odd length can also appear in G_F depending on the structure of F). To overcome these problems and answer Question 9.50 in the affirmative using our reduction scheme, one would first have to design a bipartite spreading gadget. To then get rid of the remaining odd cycles in G_F, one could use the following idea. If we denote by W and B the bipartition of the vertices of G_F (assuming G_F is bipartite), an output uvw of the generator gadget would be said *even* (resp. *odd*) if we had $u, w \in W$ and $v \in B$ (resp. $u, w \in B$ and $v \in W$). By designing a generator gadget with four distinct kinds of outputs, namely positive even, positive odd, negative even and negative odd, one could force the propagation of a locally irregular 2-edge-colouring along G_F as an usual generator gadget does, but with using the right outputs to avoid creating cycles with odd length.

Among all classes of graphs we have considered regarding Conjecture 9.9, the class of regular graphs is the most interesting since these graphs are the least locally irregular of all. Although we have not been able to prove Conjecture 9.9 to hold for all regular graphs, Theorem 9.21 is nevertheless significant and holds as a strong support for Conjecture 9.9.

One direction for future work would be to check whether the probabilistic method we used can be generalized to general graphs with sufficiently large minimum degree. Stated formally, one could ask the following.

Question 9.51. *Is there an absolute integer constant d_0 such that we have $\chi'_{irr}(G) \leq 3$ for every graph G with $\delta(G) \geq d_0$?*

Due to the dropping of the regularity condition, on which our proof of Theorem 9.21 highly relies, a proof of the existence of the constant d_0 mentioned in Question 9.51 would probably be way more complicated than our proof of Theorem 9.21. Besides, this constant d_0 would surely be larger than 10^7.

Question 9.51 was recently proved in the affirmative by Przybyło in [101], who proved the following.

Theorem 9.52 ([101])**.** *For every graph G with $\delta(G) \geq 10^{10}$, we have $\chi'_{irr}(G) \leq 3$.*

So it would be next interesting trying to improve the constant 10^{10} from Theorem 9.52 to a constant in between 10^8 and 10^9. The same matter is of course also of interest regarding the constant 10^7 from Theorem 9.21.

As Conjecture 9.9 seems out of reach at the moment, it should be more relevant to first try to prove a weaker version of it.

Problem 9.53. *Prove that there exists an absolute integer constant $d \geq 3$ such that we have $\chi'_{irr}(G) \leq d$ for every colourable graph G.*

Another direction towards Conjecture 9.9 and Problem 9.53 could be to relate the irregular chromatic index with other graph parameters, e.g. the maximum degree. One discerning approach could also be to use the following observation.

Observation 9.54. *Let G be a graph whose edge set $E(G)$ can be partitioned into k parts $E_1 \cup E_2 \cup ... \cup E_k$ such that*

$$\chi'_{irr}(G[E_1]) \leq x_1, \chi'_{irr}(G)[E_2]) \leq x_2, ..., \chi'_{irr}(G[E_k]) \leq x_k$$

for given values of $x_1, x_2, ..., x_k$. Then $\chi'_{irr}(G) \leq \sum_{i=1}^{k} x_i$.

Proof. Let $c_1, c_2, ..., c_k$ be locally irregular x_1-, x_2-, ..., x_k-edge-colourings of $G[E_1], G[E_2], ..., G[E_k]$, respectively, and denote by c the $(\sum_{i=1}^{k} x_i)$-edge-colouring of G defined as

$$c(e) = (c_i(e), i) \text{ for every } e \in E(G) \cap E_i.$$

By the partition of $E(G)$, every edge of G receives a colour by c, and c uses $\sum_{i=1}^{k} x_i$ colours. Besides, the subgraph of G induced by colour (j, i) of c is nothing but the subgraph of $G[E_i]$ induced by colour j of c_i, which is locally irregular by the definition of c_i. All subgraphs of G induced by c are then locally irregular as required. ■

Hence, if we could prove that every colourable graph can be edge-partitioned into a constant number of subgraphs with small irregular chromatic index (ideally upper-bounded by a constant), then we would come up with an upper bound on the irregular chromatic index of colourable graphs. In this scope, maybe one good direction could be to relate the irregular chromatic index and the arboricity of graphs. Since a colourable forest has irregular chromatic index at most 3, recall Theorem 9.14, we get that the irregular chromatic index of G should be upper-bounded by $3a(G)$ according to Observation 9.54 (except in some situations, see below). This idea would be especially judicious regarding graphs with small arboricity, e.g. planar graphs which have arboricity at most 3 (due to Schnyder [108]). Applying this principle, we would roughly get that every colourable planar graph should have irregular chromatic index at most 9.

Regarding this colouring strategy, we could even expect less colours to be sufficient as the structure of trees (and hence forests) with irregular chromatic index 3 is quite restricted. As mentioned in Section 9.3.2.4, trees with irregular chromatic index 3 have a predictable structure made up of "bad pieces", i.e. those given in Table 9.11. By carefully studying how these pieces must be connected, we could find easy sufficient conditions for a tree to have irregular chromatic index at most 2. Such conditions would mainly concern the location of nodes with degree 3 or 4 and the way they are organized in a tree T. Observe, for example, that no bad signature includes $\{1\}$ whenever $p \geq 2$. This means that if any node with degree at least 3 of T is connected to an hanging path with odd length, then T has irregular chromatic index at most 2. Additionally, note that if the Inverting Procedure from Section 9.3.2.3 fails on T_r, i.e. r is bad, when r has degree $\Delta(T) = 4$, then r necessarily has a neighbour with degree 4 since one of the $T_r[r, i]$'s is a 4-bad shrub. Therefore, if T has a node r' with degree 4 which has no neighbour with degree 4, then r' is not bad and T has irregular chromatic index at most 2 by Corollary 9.40.

Although edge-partitioning a graph into forests and then independently decomposing these forests into locally irregular subgraphs may seem a good idea in theory, it may practically be unusable as one of the forests may include a path with odd length, and may hence be not decomposable at all according to Corollary 9.2. It is not clear whether an edge-partition into forests can always be arranged so that we get rid of odd length path components. Maybe this problem could be overcome by using more colours to arrange the edge-colouring locally, so we ask the following.

Problem 9.55. *Prove that, for an absolute integer constant d, we have $\chi'_{irr}(G) \leq 3a(G) + d$ for every colourable graph G.*

Towards Conjecture 9.9 and Problem 9.53, one could also first investigate the impact of allowing a locally irregular graph to induce components isomorphic to K_2. Then we would come up with the following definition.

Definition 9.56. An improper k-edge-colouring c of a graph G is *nearly locally irregular* if each of the k subgraphs of G induced by c is made up of components which are either locally irregular or isomorphic to K_2. The least number of colours used by a nearly locally irregular edge-colouring of G is denoted $\chi'_{nirr}(G)$.

Of course a locally irregular edge-colouring is also nearly locally irregular (so $\chi'_{nirr}(G) \leq \chi'_{irr}(G)$ for every graph G), but obviously the contrary does not have to hold. Since a subgraph induced by a nearly locally irregular edge-colouring can include components isomorphic to K_2, which are a (main) source of trouble when dealing with locally irregular edge-colouring, we believe that only two colours should be sufficient to obtain a nearly locally irregular edge-colouring of every graph[1].

Conjecture 9.57. *For every graph G, we have $\chi'_{nirr}(G) \leq 2$.*

All ideas above apply to nearly locally irregular edge-colouring of graphs. In particular, the inequality $\chi'_{nirr}(G) \leq 3a(G)$ now directly holds for every graph G, and even $\chi'_{nirr}(G) \leq 2a(G)$ since it is easily seen that we have $\chi'_{nirr}(T) \leq 2$ for every tree T (a nearly locally irregular 2-edge-colouring of T can be obtained by just edge-partitioning T into two forests of stars by applying a breadth-first search algorithm). But what would be more interesting would be to consider the status of Problem 9.49 in the context of nearly locally irregular edge-colouring. Though this refined problem should clearly be easier than Problem 9.49, it does not seem immediate at first glance. So we raise the following.

Problem 9.58. *Prove that we have $\chi'_{nirr}(G) \leq 3$ for every bipartite graph G.*

In this chapter, we have also considered the algorithmic complexity of Locally Irregular 2-Edge-Colouring in Section 9.3, wherein we have showed that this problem is in P when restricted to trees, recall Corollary 9.43, and NP-complete in general, recall Theorem 9.44. As the status of the remaining problems Locally Irregular k-Edge-Colouring with $k \geq 3$ is intimately dependent of the status of Conjecture 9.9, we cannot tell much about the complexity of these problems at the moment.

Regarding our result on trees, it is important to keep in mind that, although we have proved that determining the irregular chromatic index of a tree T can be done in linear time, we have not mentioned how hard is it to practically obtain a locally irregular $\chi'_{irr}(T)$-edge-colouring of T. So we raise the following, which we believe to be true.

Conjecture 9.59. *For every colourable tree T, we can obtain a locally irregular $\chi'_{irr}(T)$-edge-colouring of T in time $\mathcal{O}(|V(T)|)$.*

Recall that the Inverting Procedure we have introduced uses almost locally irregular 2-edge-colourings of shrubs of T computed using Algorithm 3. Algorithm 3 basically just computes some 2-edge-colourings of smaller shrubs inductively, and then invert some of these to get an almost locally irregular 2-edge-colouring of a shrub. Though such always exists, the number of necessary inversions to perform before obtaining the almost locally irregular 2-edge-colouring may be exponential, so this procedure cannot be used to answer Conjecture 9.59 in the affirmative.

[1]Note that every graph G admits a nearly locally irregular $|E(G)|$-edge-colouring: assigning one different colour to each edge of G is nearly locally irregular. So there is no exception for this kind of colouring.

However, we believe a linear algorithm for constructing a locally irregular $\chi'_{irr}(T)$-edge-colouring of T could be obtained by first determining $\chi'_{irr}(T)$, which can be done in linear time, recall Theorem 9.42, and then applying a bottom-up strategy for edge-colouring T with taking into account the fact that we know whether $\chi'_{irr}(T) = 2$ or $\chi'_{irr}(T) = 3$.

Chapter 10
Neighbour-outsum-distinguishing arc-weighting of oriented graphs

We now study an oriented version of the 1-2-3 Conjecture, where the definitions we adopt are those given in introductory Section 7.4. Our main result in Section 10.1 states that this oriented analogue of the 1-2-3 Conjecture is true, namely that every oriented graph admits a neighbour-outsum-distinguishing 3-arc-weighting, this upper bound being tight since some oriented graphs have neighbour-outsum-distinguishing chromatic index 3. Our proof of this oriented conjecture relies on a simple argument which may be easily derived for product or list versions of the same result.

We then investigate the existence of an easy classification of oriented graphs with neighbour-outsum-distinguishing chromatic index at most 2. Although we exhibit, in Section 10.2, conditions for particular families of oriented graphs to admit a neighbour-outsum-distinguishing 2-arc-weighting, we show in Section 10.3 that the classification mentioned above is unlikely to exist due to the NP-completeness of Neighbour-Outsum-Distinguishing 2-Arc-Weighting.

In Section 10.4 is discussed an oriented version of the 1-2 Conjecture which is defined using definitions similar to those above. We in particular answer negatively to this conjecture.

Most of the results of this chapter, namely those from Sections 10.1, 10.2 and 10.3, were obtained with Baudon and Sopena and are part of an article to be published [21]. The results from Section 10.4 and the last theorem of Section 10.5 were obtained jointly with Baudon, Sopena, Stevens and Woźniak.

10.1 On oriented versions of the 1-2-3 Conjecture

We show in Theorem 10.1 below that oriented graphs have neighbour-outsum-distinguishing chromatic index at most 3, this upper bound being tight since some oriented graphs, such as the circuit on 3 vertices, do not admit a neighbour-outsum-distinguishing 2-arc-weighting. Note that, contrary to the undirected case, we do not have to make sure whether an oriented graph is weightable since all oriented graphs admit neighbour-outsum-distinguishing arc-weightings (assuming we can use a large number of weights). In particular, components isomorphic to an oriented K_2 are no longer annoying since assigning any weight to their arc does the job.

Our proof of Theorem 10.1 relies on the fact that every oriented graph has a "convenient" vertex for our purpose, that is a vertex which admits a large number of potential weighted outdegrees compared to its number of neighbours. The existence of such a vertex allows the use of an inductive proof scheme which directly yields a polynomial-time algorithm for finding a neighbour-outsum-distinguishing 3-arc-weighting of every oriented graph.

Theorem 10.1. *For every oriented graph $\overrightarrow{G}$, we have $\chi'_{nsd}(\overrightarrow{G}) \leq 3$.*

Proof. The claim is proved by induction on the size of $\overrightarrow{G}$. As a base case, the claim is clearly true when $\overrightarrow{G}$ has size 0 or 1. Suppose now that the claim is true for every oriented graph with at most $m-1$ arcs, and assume $\overrightarrow{G}$ has size $m \geq 2$.

Note that $\overrightarrow{G}$ necessarily has a vertex v such that $d^+(v) > 0$ and $d^+(v) \geq d^-(v)$ since otherwise we would have $\sum_{v \in V(\overrightarrow{G})} d^-(v) \neq \sum_{v \in V(\overrightarrow{G})} d^+(v)$. A neighbour-outsum-distinguishing 3-arc-weighting of $\overrightarrow{G}$ is then obtained as follows. Start by removing the arcs outgoing from v. According to the induction hypothesis, the remaining oriented graph admits a neighbour-outsum-distinguishing 3-arc-weighting w. Now put back the arcs outgoing from v to $\overrightarrow{G}$, and extend w to these arcs in such a way that the weighted outdegree of v is different from the weighted outdegrees of the $d^-(v)+d^+(v)$ vertices neighbouring v. This is possible since there are $2d^+(v)+1$ potential weighted outdegrees for v, namely those among $\{d^+(v), d^+(v)+1, ..., 3d^+(v)\}$, while the number of forbidden weighted outdegrees is at most $d^-(v) + d^+(v) \leq 2d^+(v) < 2d^+(v) + 1$. Because weighting the arcs outgoing from v does not affect the weighted outdegrees by w of the vertices neighbouring v, this extension of w to $\overrightarrow{G}$ remains neighbour-outsum-distinguishing. ■

Due to its simplicity, our proof of Theorem 10.1 can also be derived to list or product versions of the same result. We describe how to prove such below.

List version

The proof of Theorem 10.1 mainly relies on the fact that the number of possible weighted outdegrees by an arc-weighting for a vertex with outdegree d is sufficiently large, i.e. at least $2d+1$, when the weights from $\{1,2,3\}$ are allowed for each arc. By showing this property to hold for every triple $\{a,b,c\}$ of weights, we can strengthen Theorem 10.1 to the following.

Lemma 10.2. *Let v be a vertex with outdegree d of an oriented graph $\overrightarrow{G}$, and $\{a,b,c\}$ be a set of three real numbers. Then there are at least $2d+1$ possible weighted outdegrees for v by an arc-weighting of $\overrightarrow{G}$ assigning values among $\{a,b,c\}$ to the arcs outgoing from v.*

Proof. We prove this claim by induction on d. If $d = 1$, then the arc outgoing from v can be weighted either a, b, or c by an arc-weighting of $\overrightarrow{G}$. Since a, b, and c are distinct, there are exactly three weighted outdegrees for v, namely a, b, and c.

Assume the claim is true for every value of d up to $i-1$, and assume $d = i$. Let $\overrightarrow{G'}$ be the oriented graph obtained by removing exactly one arc $\overrightarrow{vu}$ outgoing from v. Then, according to the induction hypothesis, there are at least $2(d-1)+1$ possible weighted outdegrees for v by an arc-weighting of $\overrightarrow{G'}$ assigning values among $\{a,b,c\}$ to the remaining arcs outgoing from v. Let D' be the set of these possible weighted outdegrees, and denote by inf and sup the minimum and maximum elements of D', respectively, and by w'_{inf} and w'_{sup} two arc-weightings of $\overrightarrow{G'}$ such that $s^+_{w'_{inf}}(v) = inf$ and $s^+_{w'_{sup}}(v) = sup$, respectively.

Assume $a < b < c$. Note that if the result holds for $\{a,b,c\}$, then it also holds for $\{-a,-b,-c\}$. Hence, we only have two cases to consider, namely

1. $0 \leq a < b < c$, and
2. $a < 0 \leq b < c$.

In the first case, by extending every arc-weighting of $\overrightarrow{G'}$ to $\overrightarrow{G}$ by weighting the arc $\overrightarrow{vu}$ with weight a, we directly obtain that the set $D = \{x + a : x \in D'\}$ is a set of at least $2(d-1)+1$ possible weighted outdegrees for v. The two remaining weighted outdegrees for v are obtained

by extending w'_{sup} by weighting b or c the arc $\overrightarrow{vu}$. We then obtain that $sup + b$ and $sup + c$ are two other possible weighted outdegrees for v. Since these values do not appear in D (because $a < b < c$), there are thus at least $2d + 1$ possible weighted outdegrees for v.

In the second case, by extending every arc-weighting of $\overrightarrow{G'}$ to $\overrightarrow{G}$ by weighting b the arc $\overrightarrow{vu}$, we get that $D = \{x + b : x \in D'\}$ is a set of at least $2(d-1)+1$ weighted outdegrees for v. The two remaining weighted outdegrees for v are obtained by extending w'_{inf} and w'_{sup} to $\overrightarrow{G}$ by weighting a and c, respectively, the arc $\overrightarrow{vu}$. From these two extensions, we get that v can also have weighted outdegree $inf + a$ and $sup + c$, which do not appear in D by our assumptions on a, b and c. This completes the proof. ■

As a corollary of Lemma 10.2, we directly get that the proof of Theorem 10.1 is applicable no matter what are the three weights used to weight the arcs outgoing from every vertex. This implies the following list version of Theorem 10.1.

Corollary 10.3. *For every vertex v of an oriented graph $\overrightarrow{G}$, let $L(v)$ be an arbitrary list of three distinct real weights allowed at v. Then $\overrightarrow{G}$ admits a neighbour-outsum-distinguishing arc-weighting such that the arcs outgoing from every vertex v are weighted with values among $L(v)$.*

Product version

As for the undirected case, one can also consider a variant of Theorem 10.1 where the weighted outdegree of a vertex is the *product* of its outgoing weights rather than their sum. We then come up with the following usual definitions.

Definition 10.4. Let w be an improper arc-weighting of an oriented graph $\overrightarrow{G}$. For every vertex v of $\overrightarrow{G}$, let

$$p^+_w(v) = \prod_{u \in N^+(v)} w(\overrightarrow{vu}).$$

We say that w is *neighbour-outproduct-distinguishing* if p^+_w is proper. The minimum number of weights of a neighbour-outproduct-distinguishing k-arc-weighting of $\overrightarrow{G}$ is the *neighbour-outproduct-distinguishing chromatic index* of $\overrightarrow{G}$, denoted $\chi'_{npd}(\overrightarrow{G})$.

Regarding an arc-weighting w of an oriented graph $\overrightarrow{G}$, note that the range of possible $p^+_w(v)$'s for a vertex v is as wide as the range of possible $s^+_w(v)$'s when the weights among $\{1, 2, 3\}$ are used (this can be proved in a similar manner as Lemma 10.2). Hence, our proof of Theorem 10.1 is also a proof that every oriented graph admits a neighbour-outproduct-distinguishing 3-arc-weighting.

Theorem 10.5. *For every oriented graph $\overrightarrow{G}$, we have $\chi'_{npd}(\overrightarrow{G}) \leq 3$.*

10.2 Families with neighbour-outsum-distinguishing chromatic index at most 2

By Theorem 10.1, we know that every oriented graph admits a neighbour-outsum-distinguishing 3-arc-weighting. Throughout this section, we focus on some common families of oriented graphs and exhibit conditions for their members to admit a neighbour-outsum-distinguishing 2-arc-weighting. Please keep in mind that easy conditions for an oriented graph to have neighbour-outsum-distinguishing chromatic index at most 2 should not exist according to upcoming Theorem 10.20 (unless $\mathsf{P} = \mathsf{NP}$).

All our results are based on the fact that, for every vertex v with outdegree k of an oriented graph $\overrightarrow{G}$, there are $k + 1$ possible values as $s^+_w(v)$ by a 2-arc-weighting w of $\overrightarrow{G}$ (namely those among $\{k, k + 1, ..., 2k\}$). But this is also the case regarding $p^+_w(v)$ since the range of possible

values is $\{1, 2, 4, ..., 2^k\}$. Since there are as many possible values for $p_w^+(v)$ and $s_w^+(v)$, our results from Section 10.2 also hold directly regarding neighbour-outproduct-distinguishing 2-arc-weighting of oriented graphs.

Acyclic oriented graphs

We start by showing that every acyclic oriented graph admits a neighbour-outsum-distinguishing 2-arc-weighting.

Theorem 10.6. *For every acyclic oriented graph $\overrightarrow{G}$, we have $\chi'_{nsd}(\overrightarrow{G}) \leq 2$.*

Proof. We prove the claim by induction on the order n of $\overrightarrow{G}$. As a starting point, note that the claim is true when $n = 1$. Suppose now that the claim is true for every n up to $i - 1$ for some $i \geq 2$, and assume $\overrightarrow{G}$ is an acyclic oriented graph on $n = i$ vertices.

Since $\overrightarrow{G}$ is acyclic, there are vertices of $\overrightarrow{G}$ with indegree 0. Let v be such a vertex, and consider the graph $\overrightarrow{G'}$ obtained by removing v from $\overrightarrow{G}$. Clearly $\overrightarrow{G'}$ is acyclic and admits a neighbour-outsum-distinguishing 2-arc-weighting w according to the induction hypothesis. We now extend w to $\overrightarrow{G}$, i.e. we weight the arcs outgoing from v, in such a way that w remains neighbour-outsum-distinguishing. There are $d^+(v)+1$ possible weighted outdegrees for v, namely those among $\{d^+(v), d^+(v) + 1, ..., 2d^+(v)\}$, while there are at most $d^+(v)$ forbidden weighted outdegrees for v, namely the weighted outdegrees by w of the vertices in $N^+(v)$. Since weighting the arcs outgoing from v does not alter the weighted outdegrees of the vertices neighbouring v, we can freely choose an available weighted outdegree for v and weight its outgoing arcs consequently. This completes the proof. ■

Oriented graphs whose underlying graph is k-colourable

As already mentioned in Section 7.3, first partitioning a graph into several independent sets before weighting its edges can be a good method for finding a specific edge-weighting. This is also (and especially) the case regarding neighbour-outsum-distinguishing arc-weighting, as shown in the following result.

Theorem 10.7. *For every oriented graph $\overrightarrow{G}$, we have $\chi'_{nsd}(\overrightarrow{G}) \leq \chi(und(\overrightarrow{G}))$.*

Proof. Let $k = \chi(und(\overrightarrow{G}))$, and $V_0 \cup V_1 \cup ... \cup V_{k-1}$ be a proper k-vertex-colouring of $und(\overrightarrow{G})$. We obtain a neighbour-outsum-distinguishing k-arc-weighting of $\overrightarrow{G}$ as follows. Process the vertices of $\overrightarrow{G}$ in arbitrary order. If the considered vertex v belongs to the part V_i, then weight the arcs outgoing from v with weights from $\{1, 2, ..., k\}$ in such a way that the weighted outdegree of v is congruent to i modulo k, e.g. by assigning i to one arc outgoing to v (or k if $i = 0$), and k to all of its other outgoing arcs. This is possible unless $d^+(v) = 0$ since, in such a situation, the only possible weighted outdegree for v is 0. Once the process is achieved, two adjacent vertices u and v cannot have the same weighted outdegrees since otherwise either u and v would belong to a same part V_i, which is impossible since V_i is an independent set, or we would have $d^+(u) = d^+(v) = 0$, which is impossible since u and v are adjacent. ■

As a corollary of Theorem 10.7, we get in particular the following result.

Corollary 10.8. *For every oriented graph $\overrightarrow{G}$ whose underlying graph is bipartite, we have $\chi'_{nsd}(\overrightarrow{G}) \leq 2$*

Another way for expressing Theorem 10.7 is e.g. to consider the following variant.

Theorem 10.9. *Let $\overrightarrow{G}$ be an oriented graph whose underlying graph is neither a complete graph nor an odd length cycle. If $\delta^+(\overrightarrow{G}) \geq \lceil \frac{\Delta(und(\overrightarrow{G}))}{2} \rceil$, then we have $\chi'_{nsd}(\overrightarrow{G}) \leq 2$*

Proof. Set $k = \Delta(und(\overrightarrow{G}))$. Under the assumptions on $und(\overrightarrow{G})$, by Theorem 1.17 we have $\chi(und(\overrightarrow{G})) \leq k$, so let c be a proper k-vertex-colouring of $und(\overrightarrow{G})$. Now let w be the 2-arc-weighting of $\overrightarrow{G}$ obtained as follows: for every vertex $v \in V(\overrightarrow{G})$, weight the arcs outgoing from v so that $s_w^+(v) \equiv c(v) - 1 \pmod{k}$. This is possible since $\delta^+(\overrightarrow{G}) \geq \lceil \frac{\Delta(und(\overrightarrow{G}))}{2} \rceil$. Since c is a proper k-vertex-colouring of $und(\overrightarrow{G})$, it should be clear that s_w^+ is proper too and is hence neighbour-outsum-distinguishing. ■

Tournaments

Our strategy for weighting the arcs of a tournament $\overrightarrow{T}$ is based on the following lemma, which could be also deduced from a result by Landau regarding so-called *score sequences* (see e.g. Theorem 29 from the survey [95] by Moon).

Lemma 10.10. *For every $k \in \{1, 2, ..., |V(\overrightarrow{T})|\}$, let $n_k \geq 0$ denote the number of vertices with outdegree at most k of a tournament $\overrightarrow{T}$. Then $n_k \leq 2k + 1$.*

Proof. The claim is true for $k = 0$ since every two vertices of a tournament are joined by an arc. Let k be fixed, with $1 \leq k \leq |V(\overrightarrow{T})|$. Denote by $X \subseteq V(\overrightarrow{T})$ the set of the n_k vertices of $\overrightarrow{T}$ with outdegree at most k, and by s the sum of the outdegrees of the vertices in X. Clearly we have $s \leq n_k k$. We also have $s \geq \frac{n_k(n_k-1)}{2}$ since X induces a tournament, and there may be arcs of $\overrightarrow{T}$ whose tails lie in X, and whose heads do not lie in X. We hence get $\frac{n_k(n_k-1)}{2} \leq n_k k$, which implies that $n_k \leq 2k + 1$. ■

We now give an easy sufficient condition for a tournament to admit a neighbour-outsum-distinguishing 2-arc-weighting.

Theorem 10.11. *For every $k \in \{1, 2, ..., |V(\overrightarrow{T})|\}$, let $n_k \geq 0$ denote the number of vertices with outdegree at most k of a tournament $\overrightarrow{T}$. If we have $n_k \leq k + 1$ for every $k \in \{1, 2, ..., |V(\overrightarrow{T})|\}$, then $\chi'_{nsd}(\overrightarrow{T}) \leq 2$.*

Proof. The proof is based on the following simple weighting scheme for $\overrightarrow{T}$. Process the vertices of $\overrightarrow{T}$ in increasing order of their outdegrees. For each vertex v, weight the arcs outgoing from v in such a way that the weighted outdegree of v gets the smallest possible value which does not appear among the weighted outdegrees of the vertices considered in earlier steps of the process.

It has to be noted that this weighting scheme necessarily produces a neighbour-outsum-distinguishing arc-weighting of $\overrightarrow{T}$ when the weights among $\{1, 2, 3\}$ are used. Suppose indeed that, at some point of the process, we are dealing with a vertex v but we cannot weight v satisfyingly. Set $k = d^+(v)$. This situation means that we have attributed all the weighted outdegrees among $\{k, k + 1, ..., 3k\}$ to the vertices considered before v, i.e. that at least $2k + 1$ vertices have been treated before v. Due to how the process is led, these vertices have outdegree at most k. But then it means that $n_k > 2k + 1$, which is impossible according to Lemma 10.10.

Now assume we are using the weights among $\{1, 2\}$ only. Since the weighted outdegree of v can take every value among $\{k, k + 1, ..., 2k\}$ and at most $n_k - 1 < k + 1$ vertices have been considered in earlier steps of the process, there is necessarily one non-conflicting value which can be chosen as the weighted outdegree of v. We then just have to weight the arcs outgoing from v consequently. ■

It is worth mentioning that a tournament $\overrightarrow{T}$ admits a neighbour-outsum-distinguishing 1-arc-weighting if and only if the vertices of $\overrightarrow{T}$ have distinct outdegrees, i.e. $\overrightarrow{T}$ is transitive. This observation improves Theorem 10.11 for transitive tournaments.

Cartesian products of oriented graphs with neighbour-outsum-distinguishing chromatic index at most 2

We now show that if two oriented graphs $\overrightarrow{G}$ and $\overrightarrow{H}$ both admit a neighbour-outsum-distinguishing 2-arc-weighting, then so does their Cartesian product.

Theorem 10.12. *For every two oriented graphs $\overrightarrow{G}$ and $\overrightarrow{H}$, we have $\chi'_{nsd}(\overrightarrow{G} \,\square\, \overrightarrow{H}) \leq \max\{\chi'_{nsd}(\overrightarrow{G}), \chi'_{nsd}(\overrightarrow{H})\}$.*

Proof. Let $w_{\overrightarrow{G}}$ and $w_{\overrightarrow{H}}$ be neighbour-outsum-distinguishing $\chi'_{nsd}(\overrightarrow{G})$- and $\chi'_{nsd}(\overrightarrow{H})$-arc-weightings of $\overrightarrow{G}$ and $\overrightarrow{H}$, respectively. Let w be the $\max\{\chi'_{nsd}(\overrightarrow{G}), \chi'_{nsd}(\overrightarrow{H})\}$-arc-weighting of $\overrightarrow{G} \,\square\, \overrightarrow{H}$ defined as follows:

$$w(\overrightarrow{(u_1,v_1)(u_2,v_2)}) = \begin{cases} w_{\overrightarrow{H}}(\overrightarrow{v_1v_2}) & \text{if } u_1 = u_2, \\ w_{\overrightarrow{G}}(\overrightarrow{u_1u_2}) & \text{otherwise.} \end{cases}$$

Assume $\overrightarrow{(u_1,v_1)(u_2,v_2)}$ is an arc of $\overrightarrow{G} \,\square\, \overrightarrow{H}$. Then we have $s^+_w((u_1,v_1)) = s^+_{w_{\overrightarrow{G}}}(u_1) + s^+_{w_{\overrightarrow{H}}}(v_1)$ and $s^+_w((u_2,v_2)) = s^+_{w_{\overrightarrow{G}}}(u_2) + s^+_{w_{\overrightarrow{H}}}(v_2)$. Since (u_1,v_1) and (u_2,v_2) are adjacent, we have either $u_1 = u_2$ or $v_1 = v_2$ by construction. Assume $u_1 = u_2$ without loss of generality. Then $s^+_{w_{\overrightarrow{G}}}(u_1) = s^+_{w_{\overrightarrow{G}}}(u_2)$. Now, because $w_{\overrightarrow{H}}$ is neighbour-outsum-distinguishing, we have $s^+_{w_{\overrightarrow{H}}}(v_1) \neq s^+_{w_{\overrightarrow{H}}}(v_2)$. It then follows that $s^+_w((u_1,v_1)) \neq s^+_w((u_2,v_2))$. ■

An immediate corollary of Theorem 10.12 is the following result.

Corollary 10.13. *For every two oriented graphs $\overrightarrow{G}$ and $\overrightarrow{H}$ with $\chi'_{nsd}(\overrightarrow{G}), \chi'_{nsd}(\overrightarrow{H}) \leq 2$, we have $\chi'_{nsd}(\overrightarrow{G} \,\square\, \overrightarrow{H}) \leq 2$.*

10.3 Neighbour-Outsum-Distinguishing 2-Arc-Weighting is NP-complete

In this section, we focus on the complexity of the Neighbour-Outsum-Distinguishing k-Arc-Weighting problems. Since an oriented graph $\overrightarrow{G}$ has neighbour-outsum-distinguishing chromatic index 1 if and only if every two adjacent vertices of $\overrightarrow{G}$ have distinct outdegrees, the problem Neighbour-Outsum-Distinguishing 1-Arc-Weighting is in P. According to Theorem 10.1, every problem Neighbour-Outsum-Distinguishing k-Arc-Weighting with $k \geq 3$ is also in P since every of its instances is positive.

We herein focus on Neighbour-Outsum-Distinguishing 2-Arc-Weighting. We show this problem to be NP-complete in Theorem 10.20 below (and hence Neighbour-Outsum-Distinguishing k-Arc-Weighting should not be fixed-parameter tractable when parameterized by k), by reduction from 3-Satisfiability. For this purpose, we first introduce several gadgets to "force" the propagation of a neighbour-outsum-distinguishing 2-arc-weighting along an oriented graph.

We first introduce two kinds of *forbidding gadgets*. A forbidding gadget $\overrightarrow{F}$ is made up of one *root vertex* with outdegree 0 adjacent to *forcing vertices*. The weighting property of $\overrightarrow{F}$ is that its forcing vertices always have the same weighted outdegrees by a neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{F}$. Assume $x_1, x_2, ..., x_k$ denote the respective outdegrees of the forcing vertices. Then, after having identified the root of $\overrightarrow{F}$ with a vertex v of an oriented graph $\overrightarrow{G}$, the vertex v cannot have weighted outdegree among $\{x_1, x_2, ..., x_k\}$ by a neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{G}$ because of the forcing vertices of $\overrightarrow{F}$ now neighbouring v.

First, we define a $(2k-1, 2k)$-*forbidding gadget*, denoted $\overrightarrow{F_{2k-1,2k}}$, for every integer $k \geq 2$. These gadgets are defined inductively.

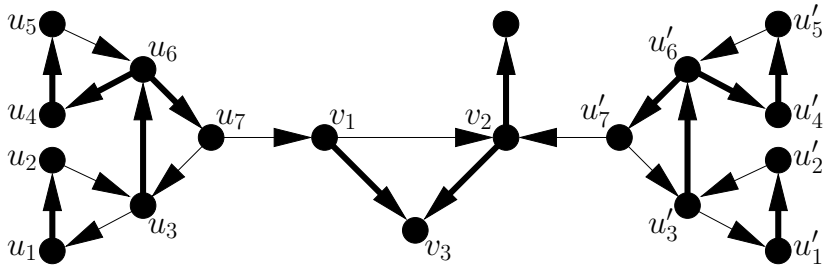


Figure 10.1: The forbidding gadget $\overrightarrow{F_{3,4}}$, and a neighbour-outsum-distinguishing 2-arc-weighting of it. Thick (resp. thin) arcs represent 1- (resp. 2-) weighted arcs.

Construction 10.14. The $(3,4)$-*forbidding gadget* $\overrightarrow{F_{3,4}}$ is the oriented graph depicted in Figure 10.1. The root of $\overrightarrow{F_{3,4}}$ is v_3, while its forcing vertices are v_1 and v_2. Now, for every value of $k \geq 3$ such that the oriented graphs $\overrightarrow{F_{2k'-1,2k'}}$ have been defined for every $k' < k$, the $(2k-1,2k)$-*forbidding gadget* $\overrightarrow{F_{2k-1,2k}}$ is constructed as follows. Let v_1^k, v_2^k and v_3^k be three distinct vertices joined by $\overrightarrow{v_1^k v_2^k}$, $\overrightarrow{v_1^k v_3^k}$ and $\overrightarrow{v_2^k v_3^k}$. Now, for every $k' \in \{2,3,...,k-1\}$, identify v_1^k and the root of a copy of $\overrightarrow{F_{2k'-1,2k'}}$. Repeat the same procedure but with v_2^k instead of v_1^k and new copies of the forbidding gadgets. Finally add an arc from v_1^k towards $k-2$ new vertices with outdegree 0, and similarly from v_2^k towards $k-1$ new vertices with outdegree 0. The root of $\overrightarrow{F_{2k-1,2k}}$ is v_3^k, while its forcing vertices are v_1^k and v_2^k.

Lemma 10.15. *Let $k \geq 2$ be fixed. In every neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{F_{2k-1,2k}}$, one of the forcing vertices has weighted outdegree $2k-1$, while the other forcing vertex has weighted outdegree $2k$.*

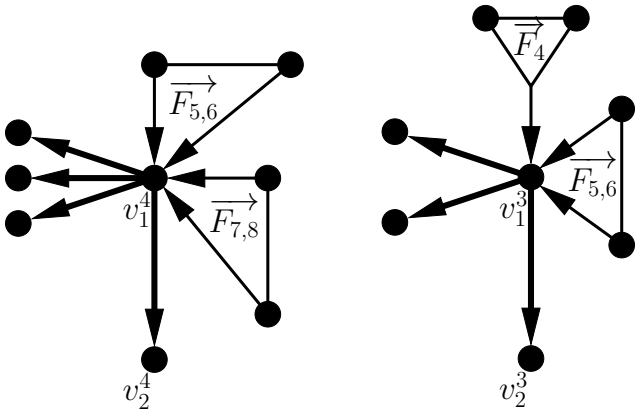
Proof. We prove the claim by induction on k. At each step, let w be a neighbour-outsum-distinguishing 2-arc-weighting of the considered forbidding gadget. Start with $\overrightarrow{F_{3,4}}$. Since u_1 and u_2 are adjacent and have outdegree 1, we have $\{s_w^+(u_1), s_w^+(u_2)\} = \{1,2\}$. By the same argument, we have $\{s_w^+(u_4), s_w^+(u_5)\} = \{1,2\}$. Since u_3 and u_6 are adjacent, both adjacent to vertices with weighted outdegree 2, and have outdegree 2, we necessarily have $\{s_w^+(u_3), s_w^+(u_6)\} = \{3,4\}$. Because u_7 is adjacent to u_3 and u_6 and has outdegree 2, we necessarily get $s_w^+(u_7) = 2$. Repeating the same arguments for the oriented subgraph of $\overrightarrow{F_{3,4}}$ induced by the u'_i's, we also obtain $s_w^+(u'_7) = 2$. Finally, since v_1 and v_2 are adjacent, both adjacent to a vertex with weighted outdegree 2, and have outdegree 2, we have $\{s_w^+(v_1), s_w^+(v_2)\} = \{3,4\}$ as claimed.

Assume the claim is true for every k up to $i-1$, and consider the gadget $\overrightarrow{F_{2k-1,2k}}$ with $k = i$. Because v_1^k and v_2^k have outdegree k by construction, their weighted outdegrees by w can only take value among $\{k, k+1, ..., 2k\}$. However, since these two vertices are both identified with the roots of forbidding gadgets $\overrightarrow{F_{3,4}}, \overrightarrow{F_{5,6}}, ..., \overrightarrow{F_{2k-3,2k-2}}$, their weighted outdegrees cannot actually take value among $\{3,4,...,2k-3,2k-2\}$ according to the induction hypothesis. Therefore, we have $\{s_w^+(v_1^k), s_w^+(v_2^k)\} = \{2k-1, 2k\}$ since v_1^k and v_2^k are adjacent. ■

We now define a *k-forbidding gadget*, denoted $\overrightarrow{F_k}$, for every integer $k \geq 3$.

Construction 10.16. The *k-forbidding gadget* $\overrightarrow{F_k}$ originally consists in an arc $\overrightarrow{v_1^k v_2^k}$. We call v_2^k and v_1^k the *root* and the *forcing vertex* of $\overrightarrow{F_k}$, respectively. Next add an arc from v_1^k towards $k-1$ new vertices with outdegree 0. The end of the construction depends on the parity of k. If k is even, then identify v_1^k and the root of each of the forbidding gadgets $\overrightarrow{F_{k+1,k+2}}, \overrightarrow{F_{k+3,k+4}}, ..., \overrightarrow{F_{2k-1,2k}}$. Otherwise, i.e. if k is odd, then identify v_1^k and the roots of $\overrightarrow{F_{k+1}}$, and $\overrightarrow{F_{k+2,k+3}}, \overrightarrow{F_{k+4,k+5}}, ..., \overrightarrow{F_{2k-1,2k}}$.

Example 10.17. The 3- and 4-forbidding gadgets $\overrightarrow{F_3}$ and $\overrightarrow{F_4}$ are depicted in Figure 10.2.



(a) The gadget $\overrightarrow{F_4}$. (b) The gadget $\overrightarrow{F_3}$.

Figure 10.2: Two examples of forbidding gadgets, and neighbour-outsum-distinguishing 2-arc-weightings of these. A triangle represents a forbidding gadget. Thick (resp. thin) arcs represent 1- (resp. 2-) weighted arcs.

Every k-forbidding gadget has the following weighting property.

Lemma 10.18. *Let $k \geq 3$ be fixed. In every neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{F_k}$, the forcing vertex has weighted outdegree k.*

Proof. Let w be a neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{F_k}$. Assume k is even. Since v_1^k has outdegree k, its weighted outdegree by w can only take value among $\{k, k+1, ..., 2k\}$. But, because v_1^k is the root of forbidding gadgets $\overrightarrow{F_{k+1,k+2}}$, $\overrightarrow{F_{k+3,k+4}}$, ..., $\overrightarrow{F_{2k-1,2k}}$, it is adjacent to vertices with weighted outdegrees $k+1, k+2, ..., 2k$ according to Lemma 10.15. Hence, the only remaining weighted outdegree for v_1^k by w is k. The claim follows similarly when k is odd, the value $k+1$ being forbidden as the weighted outdegree of v_1^k because this vertex was identified with the root of a forbidding gadget $\overrightarrow{F_{k+1}}$ with $k+1$ being even. ■

Using the two kinds of forbidding gadgets introduced above, we can now "force" a vertex of an oriented graph to have a specific weighted outdegree by a neighbour-outsum-distinguishing 2-arc-weighting.

Construction 10.19. Let v be a vertex of an oriented graph $\overrightarrow{G}$, and $k \geq d^+(v)$ be an integer. Assume we are given a set $D \subseteq \{k, k+1, ..., 2k\}$ of "allowed" weighted outdegrees for v by a neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{G}$. Then, by "*turning v into a D-vertex*", we refer to the following operations:

- first add arcs from v towards $k - d^+(v)$ new vertices with outdegree 0 so that v has outdegree k,
- then identify v and the root of every forbidding gadget $\overrightarrow{F_i}$ with $i \in \{k, k+1, ..., 2k\} \setminus D$.

Clearly, because of the forcing vertices neighbouring a D-vertex v of an oriented graph $\overrightarrow{G}$, the weighted outdegree of v by every neighbour-outsum-distinguishing 2-arc-colouring of $\overrightarrow{G}$ necessarily takes value among D.

We are now ready to introduce our hardness result.

Theorem 10.20. Neighbour-Outsum-Distinguishing 2-Arc-Weighting *is NP-complete.*

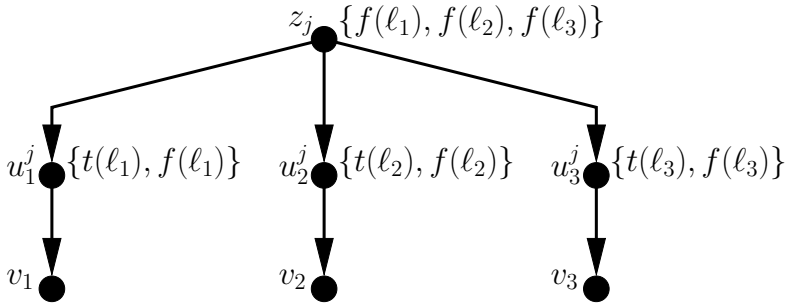


Figure 10.3: Partial resulting clause gadget for a clause $C_j = (\ell_1 \vee \ell_2 \vee \ell_3)$ with $m(C_j) = 3$. The integer sets represent the allowed weighted outdegrees at each vertex by a neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{G_F}$.

Proof. Given a 2-arc-weighting w of $\overrightarrow{G}$, one can first compute the vertex-colouring s_w^+ of $\overrightarrow{G}$ from w, and then check whether it is proper. Since this procedure can be achieved in polynomial time, Neighbour-Outsum-Distinguishing 2-Arc-Weighting is in NP.

We now prove that Neighbour-Outsum-Distinguishing 2-Arc-Weighting is NP-hard by reduction from 3-Satisfiability. Recall that we may suppose that every possible literal appears in F (Observation 1.44), and also recall what is a forced literal when dealing with 3-Satisfiability (Observation 1.46). From a formula F in conjunctive normal form involving 3-clauses, we construct an oriented graph $\overrightarrow{G_F}$ such that

$$F \text{ is satisfiable}$$
$$\Leftrightarrow$$
$$\overrightarrow{G_F} \text{ admits a neighbour-outsum-distinguishing 2-arc-weighting } w_F.$$

Let t and f be two injective mappings from $\{x_1, x_2, ..., x_n\}$ to $\{2n, 2n+1, ..., 3n-1\}$ and $\{3n, 3n+1, ..., 4n-1\}$, respectively. Assuming $\ell_j = x_i$ and $\ell_{j'} = \overline{x_i}$, i.e. ℓ_j and $\ell_{j'}$ are the literals associated with the variable x_i, we set $t(\ell_j) = f(\ell_{j'}) = t(x_i)$ and $f(\ell_j) = t(\ell_{j'}) = f(x_i)$.

First, for every literal ℓ_i of F, add a vertex v_i in $\overrightarrow{G_F}$. Now consider every clause C_j of F. We associate a *clause gadget* with C_j in $\overrightarrow{G_F}$, its structure depending on the value of $m(C_j)$. Denote by $\ell_{j_1}, \ell_{j_2}, ..., \ell_{j_{m(C_j)}}$ the distinct literals of C_j. Let $\overrightarrow{u_{j_1}^j v_{j_1}}, \overrightarrow{u_{j_2}^j v_{j_2}}, ..., \overrightarrow{u_{j_{m(C_j)}}^j v_{j_{m(C_j)}}}$ be $m(C_j)$ arcs of $\overrightarrow{G_F}$, where $u_{j_1}^j, u_{j_2}^j, ..., u_{j_{m(C_j)}}^j$ are new vertices. If $m(C_j) = 1$, i.e. ℓ_{j_1} is forced to true by C_j, then turn $u_{j_1}^j$ into a $\{t(\ell_{j_1})\}$-vertex. Otherwise, i.e. $m(C_j) \in \{2, 3\}$, turn every vertex $u_{j_i}^j$ into a $\{t(\ell_{j_i}), f(\ell_{j_i})\}$-vertex, add a vertex z_j to $\overrightarrow{G_F}$, add arcs from z_j towards $u_{j_1}^j, u_{j_2}^j, ..., u_{j_{m(C_j)}}^j$, and turn z_j into a $\{f(\ell_{j_1}), f(\ell_{j_2}), ..., f(\ell_{j_{m(C_j)}})\}$-vertex. This construction is depicted in Figure 10.3.

Claim 10.21. *If C_j is a clause of F with distinct literals $\ell_{j_1}, \ell_{j_2}, ..., \ell_{j_{m(C_j)}}$, then at least one of $t(\ell_{j_1}), t(\ell_{j_2}), ..., t(\ell_{j_{m(C_j)}})$ belongs to $\{s_{w_F}^+(u_{j_1}^j), s_{w_F}^+(u_{j_2}^j), ..., s_{w_F}^+(u_{j_{m(C_j)}}^j)\}$.*

Proof. The claim is true when $m(C_j) = 1$ since, in this situation, $u_{j_1}^j$ is a $\{t(\ell_{j_1})\}$-vertex. When $m(C_j) \in \{2, 3\}$, note that we cannot have $s_{w_F}^+(u_{j_1}^j) = f(\ell_{j_1})$, $s_{w_F}^+(u_{j_2}^j) = f(\ell_{j_2})$, ..., $s_{w_F}^+(u_{j_{m(C_j)}}^j) = f(\ell_{j_{m(C_j)}})$ since z_j is a $\{f(\ell_{j_1}), f(\ell_{j_2}), ..., f(\ell_{j_{m(C_j)}})\}$-vertex adjacent to $u_{j_1}^j, u_{j_2}^j, ...,$ $u_{j_{m(C_j)}}^j$. On the contrary, note that if there is an $i \in \{1, 2, ..., m(C_j)\}$ such that $s_{w_F}^+(u_{j_i}^j) = t(\ell_{j_i})$, then we can weight the arcs outgoing from z_j in such a way that the weighted outdegree of z_j by w_F is $f(\ell_{j_i})$. ■

Let $i \in \{1, 2, ..., 2n\}$. Note that, so far, the vertex v_i has indegree $n(\ell_i)$. Consider $i' \in \{1, 2, ..., 2n\}$ such that $\ell_{i'} = \overline{\ell_i}$. To finish the construction of $\overrightarrow{G_F}$, add the arc $\overrightarrow{v_{i'} v_i}$, and turn v_i

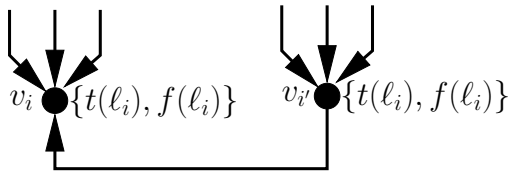


Figure 10.4: Partial subgraph of $\overrightarrow{G_F}$ for two literals ℓ_i and $\ell_{i'}$ such that $\ell_{i'} = \overline{\ell_i}$. The integer sets represent the allowed weighted outdegrees at each vertex by a neighbour-outsum-distinguishing 2-arc-weighting of $\overrightarrow{G_F}$.

and $v_{i'}$ into $\{t(\ell_i), f(\ell_i)\}$-vertices. This step of the construction is illustrated in Figure 10.4.

Claim 10.22. *Let $i \in \{1, 2, ..., 2n\}$, and $i_1, i_2, ..., i_{n(\ell_i)}$ be the indices of the distinct clauses of F which contain ℓ_i. Then $s^+_{w_F}(u_i^{i_1}) = s^+_{w_F}(u_i^{i_2}) = ... = s^+_{w_F}(u_i^{i_{n(\ell_i)}})$.*

Proof. Recall that the $u_i^{i_j}$'s can only have weighted outdegree $t(\ell_i)$ or $f(\ell_i)$ by w_F. Now note that if one of the $u_i^{i_j}$'s has weighted outdegree $t(\ell_i)$ by w_F while another such vertex has weighted outdegree $f(\ell_i)$, then w_F cannot be extended to the arcs outgoing from v_i since v_i is a $\{t(\ell_i), f(\ell_i)\}$-vertex. On the contrary, if all the $u_i^{i_j}$'s neighbouring v_i have the same weighted outdegree, say $t(\ell_i)$, then the arcs outgoing from v_i can be weighted so that $s^+_{w_F}(v_i) = f(\ell_i)$. ■

Claim 10.23. *Let $i, i' \in \{1, 2, ..., 2n\}$ be two integers such that $\ell_{i'} = \overline{\ell_i}$. Then $s^+_{w_F}(v_i) \neq s^+_{w_F}(v_{i'})$.*

Proof. The claim follows from the fact that v_i and $v_{i'}$ are adjacent. ■

We claim that F has a satisfying truth assignment if and only if $\overrightarrow{G_F}$ admits the neighbour-outsum-distinguishing 2-arc-weighting w_F. Assume $C_j = (\ell_{j_1} \vee \ell_{j_2} \vee \ell_{j_3})$ is a clause of F, and that having $s^+_{w_F}(u^j_{j_i}) = t(\ell_{j_i})$ (resp. $f(\ell_{j_i})$) simulates the assignment of ℓ_{j_i} to true (resp. false) in C_j by a truth assignment of F. Then, by Claim 10.21, every clause gadget of $\overrightarrow{G_F}$ must have a vertex $u^j_{j_i}$ whose weighted outdegree by w_F is $t(\ell_{j_i})$. This simulates the fact that every clause of F must have (at least) one true literal by a satisfying truth assignment of F. Claim 10.22 depicts the fact that, by a truth assignment of F, every literal provides the same truth value to every clause it appears in. Finally, Claim 10.23 represents the fact that two opposite literals cannot be assigned the same truth value by a truth assignment of F. With these arguments, we can deduce a satisfying truth assignment of F from w_F, and vice-versa. ■

The reduction scheme from the proof of Theorem 10.20 can be adapted to prove that it is NP-complete to decide whether a given oriented graph admits a neighbour-outproduct-distinguishing 2-arc-weighting. The forbidding gadgets can be obtained e.g. as follows, where w is a neighbour-outproduct-distinguishing 2-edge-weighting of some oriented graphs. Start from the circuit $\overrightarrow{u_1u_2u_3u_1}$ on three vertices, and add an arc $\overrightarrow{u_1u_4}$ where u_4 is a new vertex. This resulting oriented graph $\overrightarrow{F_4}$ is a 4-forbidding gadget since u_1 necessarily gets $p^+_w(u_1) = 4$. The root of $\overrightarrow{F_4}$ is u_4. Now consider another oriented graph $\overrightarrow{F_{1,2}}$ with vertices v_1, v_2, v_3 and v_4 such that $\overrightarrow{v_1v_2}$, $\overrightarrow{v_1v_3}$, $\overrightarrow{v_2v_3}$ and $\overrightarrow{v_2v_4}$ are arcs, and v_1 and v_2 are each identified with the root of one copy of $\overrightarrow{F_4}$. Clearly, since v_1 and v_2 are adjacent vertices with outdegree 2, and they are both identified with the root of a gadget $\overrightarrow{F_4}$, we necessarily have $\{p^+_w(v_1), p^+_w(v_2)\} = \{1, 2\}$, and hence $\overrightarrow{F_{1,2}}$ is a $(1, 2)$-forbidding gadget with root v_3. Now to obtain a 2^k-forbidding gadget $\overrightarrow{F_{2^k}}$ assuming that a $2^{k'}$-forbidding gadget has been defined for every $k' < k$ (with the exception that there is a $(1, 2)$-forbidding gadget rather than a 1-forbidding gadget and a 2-forbidding gadget), start from the arc $\overrightarrow{w_1w_2}$, then add arcs from w_1 towards $k-1$ new vertices so that w_1 has outdegree k, and finally identify w_1 and the roots of all the forbidding gadgets constructed in previous steps. Then we necessarily have $p^+_w(w_1) = 2^k$. Thus, $\overrightarrow{F_{2^k}}$ is a 2^k-forbidding gadget with root w_2. With these

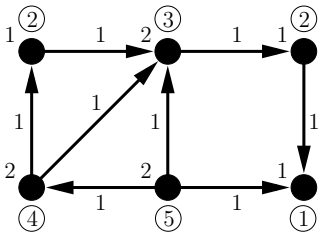


Figure 10.5: A neighbour-sum-distinguishing 2-total-weighting of an oriented graph. Weighted degrees are circled.

forbidding gadgets, our reduction scheme can then be directly adapted for the product version of the problem.

10.4 About an oriented version of the 1-2 Conjecture

Since Theorem 10.1 is an oriented version of the 1-2-3 Conjecture, it seems legitimate to wonder whether a total version of Theorem 10.1 would hold as an oriented analogue of the 1-2 Conjecture. We consider this question throughout. For this purpose, to make consistency with the fact that the weight on a vertex v by a total-weighting cannot be regarded as an "outgoing weight" (hence using the notation $s_w^+(v)$ would not make sense in this context), we first need to refine the original definition for total-weighting of oriented graphs.

Definition 10.24. Let w be an improper total-weighting of an oriented graph $\overrightarrow{G}$. The *weighted degree* of a vertex v of $\overrightarrow{G}$ is defined as

$$s_w(v) = w(v) + \sum_{u \in N^+(v)} w(\overrightarrow{vu}).$$

We say that w is *neighbour-sum-distinguishing* if s_w is proper. The minimum number of weights of a neighbour-sum-distinguishing k-total-weighting of $\overrightarrow{G}$ is the *neighbour-sum-distinguishing total chromatic number* of $\overrightarrow{G}$, denoted $\chi''_{nsd}(\overrightarrow{G})$.

Example 10.25. A neighbour-sum-distinguishing 2-total-weighting of an oriented graph is depicted in Figure 10.5.

We start by raising the following easy remark deduced from Theorem 10.1

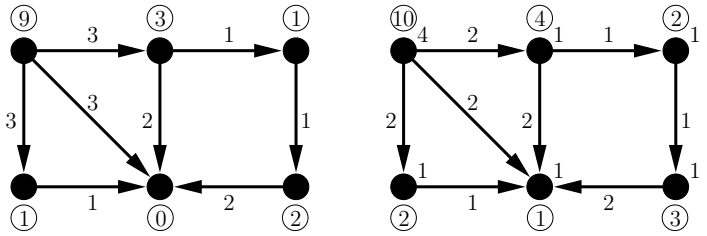
Observation 10.26. *For every $x \geq 1$, every oriented graph $\overrightarrow{G}$ admits a neighbour-sum-distinguishing $(x, 3)$-total-weighting.*

Proof. The claim actually holds for $x = 1$, and hence for every $x \geq 2$. Let w be a neighbour-outsum-distinguishing 3-arc-weighting of $\overrightarrow{G}$, which necessarily exists according to Theorem 10.1. Now define a $(1, 3)$-total-weighting w' of $\overrightarrow{G}$ as

$$\begin{cases} w'(v) &= 1 \text{ for every } v \in V(\overrightarrow{G}), \text{ and} \\ w'(\overrightarrow{uv}) &= w(\overrightarrow{uv}) \text{ for every } \overrightarrow{uv} \in A(\overrightarrow{G}). \end{cases}$$

Then for every vertex $v \in V(\overrightarrow{G})$, we have $s_{w'}(v) = s_w^+(v) + 1$. Since the adjacent vertices of $\overrightarrow{G}$ are distinguished by s_w^+, the distinguishing still holds by $s_{w'}$. ■

We now consider the existence of neighbour-sum-distinguishing $(x, 2)$-total-weightings of all oriented graphs. To begin with, we point out the existence of a neighbour-sum-distinguishing $(\Delta^+(\overrightarrow{G}) + 1, 2)$-total-weighting of every oriented graph $\overrightarrow{G}$ obtained by compensating the use of the weight 3 on the arcs with the use of big enough weights on the vertices.



(a) A neighbour-outsum-distinguishing 3-arc-weighting of $\overrightarrow{G}$.

(b) A neighbour-sum-distinguishing $(\Delta^+(\overrightarrow{G})+1, 2)$-total-weighting of $\overrightarrow{G}$.

Figure 10.6: Strategy for deducing a neighbour-sum-distinguishing $(\Delta^+(\overrightarrow{G})+1, 2)$-total-weighting of an oriented graph $\overrightarrow{G}$ from a neighbour-outsum-distinguishing 3-arc-weighting of it. Weighted outdegrees (a) and weighted degrees (b) are circled.

Observation 10.27. *Every oriented graph $\overrightarrow{G}$ admits a neighbour-sum-distinguishing $(\Delta^+(\overrightarrow{G})+1, 2)$-total-weighting.*

Proof. Let w be a neighbour-outsum-distinguishing 3-arc-weighting of $\overrightarrow{G}$. Such exists according to Theorem 10.1. Now for every vertex $v \in V(\overrightarrow{G})$, define

$$x_3(v) = |\{\overrightarrow{vu} \in A(\overrightarrow{G}) : w(\overrightarrow{vu}) = 3\}|,$$

the number of arcs outgoing from v weighted 3 by w. Clearly, we have $x_3(v) \leq \Delta^+(\overrightarrow{G})$.

Now consider the $(\Delta^+(\overrightarrow{G})+1, 2)$-total-weighting w' of $\overrightarrow{G}$ defined as

$$\begin{cases} w'(v) &= x_3(v)+1 \text{ for every } v \in V(\overrightarrow{G}), \text{ and} \\ w'(\overrightarrow{uv}) &= \min\{2, w(\overrightarrow{uv})\} \text{ for every } \overrightarrow{uv} \in A(\overrightarrow{G}). \end{cases}$$

By the way w' is defined, we have $s_{w'}(v) = s^+_w(v)+1$ for every $v \in V(\overrightarrow{G})$. Since w is neighbour-outsum-distinguishing, then w' is neighbour-sum-distinguishing (see Figure 10.6). ■

The next question is of great interest since it is a direct analogue of the 1-2 Conjecture. Namely, we investigate whether there is a constant positive integer $x \geq 1$ such that every oriented graph admits a neighbour-sum-distinguishing $(x, 2)$-total-weighting. We show that such an x does not exist.

Proposition 10.28. *There is no $x \geq 1$ such that every oriented graph admits a neighbour-sum-distinguishing $(x, 2)$-total-weighting.*

Proof. Consider the following family of tournaments.

Construction 10.29. Choose an odd integer $n \geq 5$, and let $\overrightarrow{T_n}$ be the tournament on n vertices defined as follows. Denote $0, 1, ..., n-1$ the vertices of $\overrightarrow{T_n}$, and, for every vertex i of $\overrightarrow{T_n}$, add the arcs $\overrightarrow{ii+1}, \overrightarrow{ii+1}, ..., \overrightarrow{ii+\lfloor\frac{n}{2}\rfloor}$, where the indices are taken modulo n.

Example 10.30. The oriented graph $\overrightarrow{T_5}$ is depicted in Figure 10.7.

From now on, assume $x \geq 1$ is fixed. By construction, every vertex v of $\overrightarrow{T_n}$ has outdegree $\lfloor\frac{n}{2}\rfloor$. For this reason, the possible values of $s_w(v)$ by an $(x, 2)$-total-weighting w of $\overrightarrow{T_n}$ are those among the set

$$\left\{\lfloor\frac{n}{2}\rfloor+1, \lfloor\frac{n}{2}\rfloor+2, ..., 2\lfloor\frac{n}{2}\rfloor+x\right\},$$

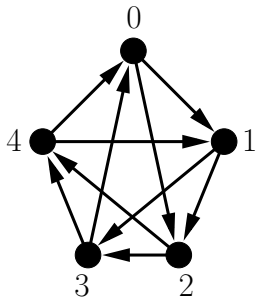


Figure 10.7: The oriented graph $\overrightarrow{T_5}$.

which includes $\lfloor \frac{n}{2} \rfloor + x$ values. Because $\overrightarrow{T_n}$ is a tournament, we need to have $s_w(u) \neq s_w(v)$ for every two vertices u and v of $\overrightarrow{T_n}$ so that w is neighbour-sum-distinguishing. Thus we want x to be big enough so that $\lfloor \frac{n}{2} \rfloor + x \geq n$, and hence $x \geq \lceil \frac{n}{2} \rceil$. Because x is fixed, by making n tends to infinite we get that using the weights among $\{1, 2, ..., x\}$ for weighting the vertices of $\overrightarrow{T_n}$ is not enough. This implies the claim. ■

Consequently, as a side result we directly get that the natural oriented version of the 1-2 Conjecture (defined accordingly to our definitions) is not true.

Corollary 10.31. *For every oriented graph $\overrightarrow{G}$, we do not have $\chi''_{nsd}(\overrightarrow{G}) \leq 2$.*

So Observation 10.26 actually provides the tightest constant upper bound on χ''_{nsd}.

Corollary 10.32. *For every oriented graph $\overrightarrow{G}$, we have $\chi''_{nsd}(\overrightarrow{G}) \leq 3$.*

10.5 Conclusion and open questions

In this chapter, we have first investigated an oriented version of the 1-2-3 Conjecture where the distinguishing parameter is the sum of weights on the arcs outgoing from the vertices. This problem has shown up to be way easier than the undirected problem, mainly due to the fact that weighting an arc only affects the weighted outdegree of one of the two attached vertices. So not only we have managed to prove Theorem 10.1 as an oriented analogue of the 1-2-3 Conjecture, but also the proof of this statement is easy and relies on simple arguments which we have generalized to notably a list version of the same result.

In Sections 10.2 and 10.3, we have considered the existence of an easy classification of oriented graphs with neighbour-outsum-distinguishing chromatic index at most 2. Though Theorem 10.20 ensures that such an easy classification exists if and only if $\mathsf{P} = \mathsf{NP}$, we have proved that, for some common families of oriented graphs, neighbour-outsum-distinguishing 2-arc-weightings can be obtained easily. Regarding tournaments, it is important to keep in mind that Theorem 10.11 only provides a sufficient condition for a tournament $\overrightarrow{T}$ to admit a neighbour-outsum-distinguishing 2-arc-weighting, which should not be necessary, in particular when the sequence of outdegrees of $\overrightarrow{T}$ includes "large gaps". So we address the following problem.

Problem 10.33. *Exhibit a necessary and sufficient condition for a tournament to admit a neighbour-outsum-distinguishing 2-arc-weighting.*

We believe such a characterization should be dependent of several parameters, in particular the number of vertices with given outdegree, or the sequence of outdegrees itself. In a more general context, it would be interesting investigating, in further works, whether we can obtain similar results for other classes of graphs.

One further direction for extending our results is to consider undirected graphs.

Question 10.34. *What is the least $k \in \{1, 2, 3\}$ such that every undirected graph admits an orientation which admits a neighbour-outsum-distinguishing k-arc-weighting?*

Recall that an oriented graph admits a neighbour-outsum-distinguishing 1-arc-weighting if and only if every two of its adjacent vertices have distinct outdegrees. According to a result first proved (to the best of our knowledge) by Borowiecki, Grytczuk and Pilśniak in [37], the answer to Question 10.34 is 1. We give a reformulated proof of this statement using our terminology.

Lemma 10.35 ([37])**.** *Every undirected graph G admits an orientation in which every two adjacent vertices have distinct outdegrees.*

Proof. We prove this result by induction on the order n of G. Since the result is true for $n = 1$, we assume the claim is true for every n up to $i - 1$, and now consider $n = i$. Let v be a vertex whose degree is maximum in G. According to the induction hypothesis, the graph $G' = G - v$ admits an orientation $\overrightarrow{G'}$ in which every two adjacent vertices have distinct outdegrees. Note that in $\overrightarrow{G'}$, the outdegree of every vertex in $N(v)$ is at most $d(v) - 1$ since v has maximum degree in G. Now start from $\overrightarrow{G'}$, and let $\overrightarrow{G}$ be the orientation of G obtained by orienting all edges incident with v from v towards its neighbours. Since the outdegree of v in $\overrightarrow{G}$ is then $d(v)$, and the outdegrees of all vertices neighbouring v are not altered, the obtained orientation still satisfies the claim. ■

Corollary 10.36. *Every undirected graph admits an orientation which admits a neighbour-outsum-distinguishing 1-arc-weighting.*

In a more general context, it would be interesting investigating how many weights are needed to distinguish the vertices at distance at most d of every oriented graph via their sums of outgoing weights by an arc-weighting. This would be formally rephrased as follows.

Question 10.37. *Given a $d \geq 1$, what is the least $k \geq 1$ such that every oriented graph $\overrightarrow{G}$ admits a k-arc-weighting w verifying $s^+_w(u) \neq s^+_w(v)$ for all vertices u and v at distance at most d in $\overrightarrow{G}$?*

From Theorem 10.1, we basically get that $k = 3$ for $d = 1$. But it would be interesting investigating how k increases in front of d depending on whether we consider the weak of the strong notion of oriented distance. It would be in particular interesting first considering whether there is a constant upper bound on k for every d.

In Section 10.4 we have discussed an oriented version of the 1-2 Conjecture defined accordingly to the oriented notions we have been considering throughout. Our main result is a reject of this conjecture, recall Corollary 10.31.

It would be interesting considering new distinguishing parameters for which the associated analogues of the 1-2 Conjecture would be true. As an illustration, we point out below such a distinguishing total-weighting notion for which the 1-2 Conjecture-like conjecture is true.

Definition 10.38. Let w be an improper total-weighting of an oriented graph $\overrightarrow{G}$. For every vertex v of $\overrightarrow{G}$, let

$$c_w(v) = (w(v), \sum_{u \in N^+(v)} w(\overrightarrow{vu})).$$

We say that w is *neighbour-couple-distinguishing* if c_w is proper. The minimum number of weights of a neighbour-couple-distinguishing k-total-weighting of $\overrightarrow{G}$ is the *neighbour-couple-distinguishing total chromatic number* of $\overrightarrow{G}$, denoted $\chi''_{ncd}(\overrightarrow{G})$.

As mentioned above, the analogue of the 1-2 Conjecture for neighbour-couple-distinguishing total-weighting of oriented graphs is true. It reads as follows.

Theorem 10.39. *For every oriented graph $\overrightarrow{G}$, we have $\chi''_{ncd}(\overrightarrow{G}) \leq 2$.*

Proof. Let $\overrightarrow{G}$ be an oriented graph with order n. Consider the following ordering $(v_1, v_2, ..., v_n)$ over the vertices of $\overrightarrow{G}$. Let v_n be a vertex of $\overrightarrow{G}$ satisfying $d^-_{\overrightarrow{G}}(v_n) \leq d^+_{\overrightarrow{G}}(v_n)$. Such a vertex exists since

$$\sum_{v \in V(G)} d^-_{\overrightarrow{G}}(v) = \sum_{v \in V(G)} d^+_{\overrightarrow{G}}(v).$$

Now consider the oriented graph $\overrightarrow{G} - \{v_n\}$, and denote v_{n-1} one vertex of $V(\overrightarrow{G}) \setminus \{v_n\}$ satisfying $d^-_{\overrightarrow{G}-\{v_n\}}(v_{n-1}) \leq d^+_{\overrightarrow{G}-\{v_n\}}(v_{n-1})$. Repeat the same procedure until all vertices of $\overrightarrow{G}$ are labelled. Namely, assuming that the vertices $v_{n-i+1}, v_{n-i+2}, ..., v_n$ have been defined, we choose v_{n-i} as a vertex of $\overrightarrow{G} - \{v_{n-i+1}, v_{n-i+2}, ..., v_n\}$ satisfying

$$d^-_{\overrightarrow{G}-\{v_{n-i+1}, v_{n-i+2}, ..., v_n\}}(v_{n-i}) \leq d^+_{\overrightarrow{G}-\{v_{n-i+1}, v_{n-i+2}, ..., v_n\}}(v_{n-i}),$$

which again exists according to the same argument as above.

We construct a neighbour-couple-distinguishing 2-total-weighting w of $\overrightarrow{G}$ by considering the vertices $v_1, v_2, ..., v_n$ in increasing order of their indices. Assume $v_1, v_2, ..., v_{i-1}$ have already been correctly treated, i.e. $c_w(v_1)$, $c_w(v_2)$, ..., $c_w(v_{i-1})$ are defined (these vertices and their outgoing arcs have been each assigned a weight) and $c_w(v_j) \neq c_w(v_{j'})$ for every $j, j' \in \{1, 2, ..., i-1\}$ with $j \neq j'$ such that v_j and $v_{j'}$ are adjacent. Denote $\overrightarrow{G_i} = \overrightarrow{G} - \{v_{i+1}, v_{i+2}, ..., v_n\}$ for the sake of simplicity. We now assign a weight to v_i and its outgoing arcs by w in such a way that no conflict arises. An important thing to keep in mind is that when weighting an arc $\overrightarrow{v_i v_j}$, the couple $c_w(v_j)$ is not altered. Note further that $c_w(v_i) \neq c_w(v_j)$ whenever $w(v_i) \neq w(v_j)$.

For every $\alpha \in \{1, 2\}$, let

$$x_\alpha = |\{v_j \in V(\overrightarrow{G}) \cap N^+_{\overrightarrow{G_i}}(v_i) : j < i \text{ and } w(v_j) = \alpha\}|$$

be the number of already treated adjacent vertices which received weight α. There has to be a value of $\alpha \in \{1, 2\}$ for which

$$x_\alpha \leq \left\lfloor \frac{d^+_{\overrightarrow{G_i}}(v_i) + d^-_{\overrightarrow{G_i}}(v_i)}{2} \right\rfloor.$$

Let us assume $\alpha = 1$ throughout.

Set $w(v_i) = 1$. Then v_i is already distinguished from all its already treated adjacent vertices which received weight 2 by w. Now what remains to do is to weight the arcs outgoing from v_i so that v_i is distinguished by its outsum from all its already treated adjacent vertices which received weight 1 by w. The possible outsums for v_i in $\overrightarrow{G}$ by w are those among

$$D_i = \{d^+_{\overrightarrow{G}}(v_i), d^+_{\overrightarrow{G}}(v_i) + 1, ..., 2d^+_{\overrightarrow{G}}(v_i)\},$$

forming a set with cardinality $d^+_{\overrightarrow{G}}(v_i) + 1$. But the outsum of v_i has to be different from the outsums of its x_1 previously treated adjacent vertices which also received weight 1 by w. By the ordering of the vertices of $\overrightarrow{G}$, we have

$$d^+_{\overrightarrow{G_i}}(v_i) + d^-_{\overrightarrow{G_i}}(v_i) \leq 2d^+_{\overrightarrow{G_i}}(v_i),$$

yielding

$$x_1 \leq d^+_{\overrightarrow{G_i}}(v_i) < |D_i|.$$

There is thus at least one value d_i among D_i which does not appear as the outsum by w of any vertex v_j with $j < i$ neighbouring v_i which received weight 1. Then just weight the arcs outgoing from v_i so that the outsum of v_i is d_i. Now v_i gets also distinguished from its previously considered neighbours weighted 1. ■

Chapter 11

Locally irregular arc-colouring of oriented graphs

This chapter is dedicated to the study of locally irregular arc-colouring of oriented graphs. As for the undirected case, recall Conjecture 9.9, investigations on small classes of oriented graphs suggest that every oriented graph should be decomposable into at most 3 locally irregular subgraphs. This yields the main conjecture investigated in this chapter.

Conjecture 11.1. *For every oriented graph $\overrightarrow{G}$, we have $\chi'_{irr}(\overrightarrow{G}) \leq 3$.*

Conjecture 11.1, if true, would be tight, since e.g. the circuit on 3 vertices has irregular chromatic index 3. We support Conjecture 11.1 by showing it to hold for several common classes of oriented graphs in Section 11.1. We then prove, in Section 11.2, a weaker version of Conjecture 11.1. Namely, we prove that every oriented graph admits a locally irregular 6-arc-colouring. We finally consider the algorithmic aspect in Section 11.3. In this scope, we prove that deciding whether an oriented graph has irregular chromatic index at most 2 is NP-complete, even when restricted to oriented graphs with maximum indegree and outdegree at most 4. So even if Conjecture 11.1 turned out to be true, a classification of oriented graphs with irregular chromatic index at most 2 should not exist, unless P = NP.

The whole content of this chapter is made up of results obtained with Renault, and is part of an article submitted for publication [35].

11.1 Families with irregular chromatic index at most 3

Throughout this section, we exhibit families of oriented graphs for which Conjecture 11.1 holds. Namely, we prove Conjecture 11.1 to hold for oriented graphs whose underlying graph has chromatic number at most 3, acyclic oriented graphs, and Cartesian products of oriented graphs with irregular chromatic index at most 3. The most important of these results is the one related to acyclic oriented graphs, as it plays a crucial role in next Section 11.2.

Oriented graphs whose underlying graph is k-colourable

We show that every oriented graph whose underlying graph is k-colourable admits a locally irregular k-arc-colouring.

Theorem 11.2. *For every oriented graph $\overrightarrow{G}$, we have $\chi'_{irr}(\overrightarrow{G}) \leq \chi(und(\overrightarrow{G}))$.*

Proof. Without loss of generality, we may assume that $\overrightarrow{G}$ is connected. Let $\chi(und(\overrightarrow{G})) = k$, and $V_1 \cup V_2 \cup ... \cup V_k$ be a proper k-vertex-colouring of $und(\overrightarrow{G})$. Consider the k-arc-colouring c of $\overrightarrow{G}$ obtained by colouring i every arc whose tail lies in V_i for every $i \in \{1, 2, ..., k\}$. Now consider two adjacent vertices u and v. By definition of a proper vertex-colouring, we have $u \in V_i$ and $v \in V_j$ for $i, j \in \{1, 2, ..., k\}$ with $i \neq j$. Besides, according to how c was obtained, we have $d^+_{c,i}(u) \geq 1$ (unless u has outdegree 0) and $d^+_{c,j}(u) = 0$, and $d^+_{c,i}(v) = 0$ and $d^+_{c,j}(v) \geq 1$ (unless v has outdegree 0), while the arc between u and v is coloured either i or j. As u and v cannot be adjacent and both have outdegree 0, they cannot have the same outdegree in the $c(\overrightarrow{uv})$- or $c(\overrightarrow{vu})$-subgraph. Repeating the same argument for every pair of adjacent vertices of $\overrightarrow{G}$, we get that c is locally irregular. ■

As a special case of Theorem 11.2, we get that every oriented graph whose underlying graph is a tree, a bipartite graph, or even a 3-colourable graph, agrees with Conjecture 11.1.

Corollary 11.3. *For every oriented graph $\overrightarrow{G}$ whose underlying graph is 3-colourable, we have $\chi'_{irr}(\overrightarrow{G}) \leq 3$.*

Acyclic oriented graphs

We herein show that every acyclic oriented graph admits a locally irregular 3-arc-colouring. This result is crucial for our proof that every oriented graph has irregular chromatic index at most 6, see Theorem 11.9 in Section 11.2.

Theorem 11.4. *For every acyclic oriented graph $\overrightarrow{G}$, we have $\chi'_{irr}(\overrightarrow{G}) \leq 3$.*

Proof. We actually prove a stronger statement, namely that every acyclic oriented graph $\overrightarrow{G}$ admits a locally irregular 3-arc-colouring in which at most two colours are used at each vertex, i.e. the arcs outgoing from every vertex are coloured with at most two colours only. The proof is by induction on the order n of $\overrightarrow{G}$. The claim can be easily verified for small values of n, e.g. for $n \leq 3$. Let us thus assume the thesis holds for every oriented graph with order at most $n-1$. Since $\overrightarrow{G}$ is acyclic, there has to be a vertex of $\overrightarrow{G}$ with indegree 0. Let v be such a vertex, and denote its neighbours by $u_1, u_2, ..., u_d$, i.e. $\overrightarrow{vu_i}$ is an arc for every $i \in \{1, 2, ..., d\}$, where $d = d^+(v)$.

Let $\overrightarrow{H} = \overrightarrow{G} - \{v\}$. Since removing vertices from an acyclic graph does not create new circuits, the oriented graph $\overrightarrow{H}$ is still acyclic. Besides, it admits a locally irregular 3-arc-colouring $c_{\overrightarrow{H}}$ satisfying the restrictions above according to the induction hypothesis. Now put back v and its adjacent arcs, and try to extend $c_{\overrightarrow{H}}$, i.e. colour the arcs outgoing from v, to a locally irregular 3-arc-colouring $c_{\overrightarrow{G}}$ of $\overrightarrow{G}$ satisfying the conditions above. We show below that such an extension from $c_{\overrightarrow{H}}$ to $c_{\overrightarrow{G}}$ necessarily exists.

For this purpose, we first show that such an extension necessarily exists whenever $d \leq 3$ before generalizing our arguments. If $d = 1$, then, by our assumptions on $c_{\overrightarrow{H}}$, at most two colours, say 1 and 2, are used at u_1. Then by colouring 3 the arc $\overrightarrow{vu_1}$, no conflict arises and $c_{\overrightarrow{G}}$ remains locally irregular. Besides, only one colour is used at v.

Now, if $d = 2$, then start by colouring 1 all arcs outgoing from v. If $c_{\overrightarrow{G}}$ is not locally irregular, then one vertex u_{i_1} has 1-outdegree 2 by $c_{\overrightarrow{H}}$. Now colour 2 all arcs outgoing from v. Again, if $c_{\overrightarrow{G}}$ is not locally irregular, then it means that one vertex u_{i_2} has 2-outdegree 2 by $c_{\overrightarrow{H}}$. Now colour 3 all arcs outgoing from v. If $c_{\overrightarrow{G}}$ is not locally irregular again, then one vertex u_{i_3} has 3-outdegree 2. Since $d = 2$ and there are at most two colours used at each of the u_i's, it means that we have revealed all the colours used at one of the u_i's. Assume $i_1 = i_2 = 1$ without loss of generality. Then u_1 has 1- and 2-outdegree 2, while u_2 has 3-outdegree 2. Note then that by setting $c_{\overrightarrow{G}}(\overrightarrow{vu_1}) = 1$ and $c_{\overrightarrow{G}}(\overrightarrow{vu_2}) = 3$, the arc-colouring $c_{\overrightarrow{G}}$ is locally irregular. Since $d = 2$, note further that at most two colours are used at v, as required.

Finally consider $d = 3$. As previously, start by colouring all arcs outgoing from v with a same colour. Again, if $c_{\overrightarrow{G}}$ is not locally irregular for one of these three extensions of $c_{\overrightarrow{H}}$, then we get that a vertex u_{i_1} has 1-outdegree 3, a vertex u_{i_2} has 2-outdegree 3, and one vertex u_{i_3} has 3-outdegree 3. Now fix $c_{\overrightarrow{G}}(\overrightarrow{vu_{i_1}}) = 1$ (there is no conflict in the 1-subgraph since u_{i_1} has 1-outdegree 3) and colour all of the remaining arcs outgoing from v with a same colour different from 1. Again, if $c_{\overrightarrow{G}}$ is never locally irregular, then we get that one vertex u_{i_4} has 2-outdegree 2, and one vertex u_{i_5} has 3-outdegree 2. Similarly, if $c_{\overrightarrow{G}}$ is not locally irregular when assigning $\overrightarrow{vu_{i_2}}$ colour 2 (again, there is not conflict in the 2-subgraph since u_{i_2} has 2-outdegree 3) and all of the other arcs outgoing from v colour 1, then we get that one vertex u_{i_6} has 1-outdegree 2. At this point, all of the coloured outdegrees of the u_i's are revealed. Since it was revealed that colour 1 is used at u_{i_6}, either colour 2 or 3 is not used at v_{i_6}. Assume this colour is 2 without loss of generality. Now just assign colour 2 to $\overrightarrow{vu_{i_6}}$, and colour 1 to all of the other arcs outgoing from v. Then v and u_{i_6} are adjacent in the 2-subgraph but have distinct 2-outdegrees, namely 1 and 0, respectively, while v and its other two neighbours are adjacent in the 1-subgraph and have distinct 1-outdegrees since v has 1-outdegree 2 and only u_{i_6} was revealed to have 1-outdegree 2. It then follows that $c_{\overrightarrow{G}}$ is locally irregular. Note further that at most two colours are used to colour the arcs outgoing from v at every moment of the procedure.

We now generalize our arguments to every $d \geq 4$. The colouring scheme we use below is quite similar to the one used so far. If, at some step, the resulting arc-colouring $c_{\overrightarrow{G}}$ is locally irregular, then we are done. Suppose then this never occurs. We start by colouring with only one colour all arcs outgoing from v (Step 1). Since $c_{\overrightarrow{G}}$ is never locally irregular, it means that one vertex u_{i_1} has 1-outdegree d, one vertex u_{i_2} has 2-outdegree d, and one vertex u_{i_3} has 3-outdegree d. Next, we try to extend $c_{\overrightarrow{H}}$ to $c_{\overrightarrow{G}}$ by colouring with colour α one arc outgoing from v whose head was shown to have α-outdegree strictly more than 1 during Step 1 (i.e. u_{i_1}, u_{i_2} or u_{i_3}), and then colouring all of the other arcs outgoing from v with a colour different from α (Step 2.a). Again, if $c_{\overrightarrow{G}}$ is not locally irregular for every of these attempts, then we reveal that one vertex u_{i_4} has 1-outdegree $d-1$, one vertex u_{i_5} has 2-outdegree $d-1$, and one vertex u_{i_6} has 3-outdegree $d-1$. Once the vertices u_{i_4}, u_{i_5} and u_{i_6} are revealed, we can reveal additional 1-, 2- and 3-outdegrees as follows. Since u_{i_4} has 1-outdegree $d-1$, it means that a colour different from 1, say 2, is not used at u_{i_4}. Then colour 2 the arc $\overrightarrow{vu_{i_4}}$, and 1 all of the other arcs outgoing from v. Then we reveal that a vertex u_{i_7} different from u_{i_1} and u_{i_4} has 1-outdegree $d-1$. Repeating the same strategy with u_{i_5} and u_{i_6} (Step 2.b), we reveal also that two vertices u_{i_8} (different from u_{i_2} and u_{i_5}) and u_{i_9} (different from u_{i_3} and u_{i_6}) have 2- and 3-outdegree $d-1$, respectively.

Repeat the same strategy as many times as necessary until $c_{\overrightarrow{G}}$ is locally irregular, or all of the 1-, 2- and 3-outdegrees of the u_i's are revealed. More precisely, for every $j = 3, 4, ..., \lceil \frac{d}{2} \rceil$ taken consecutively, colour with colour α exactly $j-1$ of the arcs outgoing from v whose heads were shown to have α-outdegree strictly more than $j-1$ in earlier steps, and colour the remaining $d-j+1$ arcs with a colour β different from α (Step j.a). At Step j.a, we reveal that one vertex $u_{i_{3+6(j-2)+1}}$ has 1-outdegree $d-j+1$, one vertex $u_{i_{3+6(j-2)+2}}$ has 2-outdegree $d-j+1$, and one vertex $u_{i_{3+6(j-2)+3}}$ has 3-outdegree $d-j+1$. Repeating Step j.a but with "forcing" $u_{i_{3+6(j-2)+1}}$ to be one of the $j-1$ arcs coloured with a colour not appearing at it, and then similarly for $u_{i_{3+6(j-2)+2}}$ and $u_{i_{3+6(j-2)+3}}$ (Step j.b), we reveal that three other vertices, $u_{i_{3+6(j-2)+4}}$, $u_{i_{3+6(j-2)+5}}$ and $u_{i_{3+6(j-1)}}$, have 1-, 2- and 3-outdegree $d-j+1$, respectively. We refer to Steps j.a and j.b as Step j.

Hence, at each Step j, we reveal that two of the u_i's have 1-outdegree $d-j+1$, two of the u_i's have 2-outdegree $d-j+1$ and two of the u_i's have 3-outdegree $d-j+1$ (except for Step 1 where only one outdegree of each colour is revealed). Since $d \geq 4$ and there are only two colours used at each vertex u_i according to the assumption on $c_{\overrightarrow{H}}$, and hence at most $2d$ outdegrees to be revealed, it should be clear that all of the 1-, 2- and 3-outdegrees of the u_i's are revealed once j reaches $\lceil \frac{d}{2} \rceil$. Besides, every 1-, 2- or 3-outdegree is revealed to be strictly greater than $\lceil \frac{d}{2} \rceil$ (except when $d = 5$, see below). One can then obtain the locally irregular 3-arc-colouring $c_{\overrightarrow{G}}$

by assigning colour 1 to $\lceil \frac{d}{2} \rceil$ arcs outgoing from v whose head were shown to have 1-outdegree strictly more than $\lceil \frac{d}{2} \rceil$, and colour 2 to the remaining arcs (there are $\lfloor \frac{d}{2} \rfloor$ of them). Under this colouring, the vertex v has 1- and 2-outdegree $\lceil \frac{d}{2} \rceil$ and $\lfloor \frac{d}{2} \rfloor$, respectively, while its neighbours have 1- and 2-outdegree 0 or strictly greater than these in the 1- and 2-subgraphs, respectively (when $d = 5$, one of the u_i's, say u_1, is revealed to have 1-outdegree 3 - in this special case, force $\overrightarrow{vu_1}$ to be coloured 2. For every other value of d, it is true that all revealed outdegrees are strictly greater than $\lceil \frac{d}{2} \rceil$). Besides, only two colours are used at v. This ends up the proof. ■

It is worth noting that the stronger statement proved in the proof of Theorem 11.4 is crucial for our colouring scheme. Indeed, assume e.g. that $d = 1$, that three colours are allowed at each vertex, and that u_1 has 1-, 2- and 3-outdegree exactly 1 by $c_{\overrightarrow{H}}$. In such a situation, we clearly cannot extend $c_{\overrightarrow{H}}$ to $c_{\overrightarrow{G}}$.

Cartesian products of oriented graphs with irregular chromatic index at most 3

We herein investigate a last family of oriented graphs, namely Cartesian products of oriented graphs with irregular chromatic index at most 3. Our result below provides numerous more examples of oriented graphs verifying Conjecture 11.1, assuming that we are provided oriented graphs agreeing with Conjecture 11.1 themselves.

Theorem 11.5. *For every two oriented graphs $\overrightarrow{G}$ and $\overrightarrow{H}$, we have $\chi'_{irr}(\overrightarrow{G} \,\square\, \overrightarrow{H}) \leq \max\{\chi'_{irr}(\overrightarrow{G}), \chi'_{irr}(\overrightarrow{H})\}$.*

Proof. Let $c_{\overrightarrow{G}}$ and $c_{\overrightarrow{H}}$ be locally irregular $\chi'_{irr}(\overrightarrow{G})$- and $\chi'_{irr}(\overrightarrow{H})$-arc-colourings of $\overrightarrow{G}$ and $\overrightarrow{H}$, respectively. Now let c be the $\max\{\chi'_{irr}(\overrightarrow{G}), \chi'_{irr}(\overrightarrow{H})\}$-arc-colouring of $\overrightarrow{G} \,\square\, \overrightarrow{H}$ obtained from $c_{\overrightarrow{G}}$ and $c_{\overrightarrow{H}}$ as follows:

$$c(\overrightarrow{(u_1, v_1)(u_2, v_2)}) = \begin{cases} c_H(\overrightarrow{v_1v_2}) \text{ if } u_1 = u_2, \\ c_G(\overrightarrow{u_1u_2}) \text{ otherwise.} \end{cases}$$

Note that we have $d^+_{c,i}((u_1, v_1)) = d^+_{c_{\overrightarrow{G}},i}(u_1) + d^+_{c_{\overrightarrow{H}},i}(v_1)$ for every colour $i \in \{1, 2, ..., \max\{\chi'_{irr}(\overrightarrow{G}), \chi'_{irr}(\overrightarrow{H})\}\}$. Assume $\overrightarrow{(u_1, v_1)(u_2, v_2)}$ is an arc of $\overrightarrow{G} \,\square\, \overrightarrow{H}$ with $c(\overrightarrow{(u_1, v_1)(u_2, v_2)}) = i$. By definition, we have either $u_1 = u_2$ or $v_1 = v_2$. Suppose $u_1 = u_2$ without loss of generality. Then we have $d^+_{c_{\overrightarrow{G}},i}(u_1) = d^+_{c_{\overrightarrow{G}},i}(u_2)$, and, because $c_{\overrightarrow{H}}$ is locally irregular, also $d^+_{c_{\overrightarrow{H}},i}(v_1) \neq d^+_{c_{\overrightarrow{H}},i}(v_2)$. It then follows that $d^+_{c,i}((u_1, v_1)) \neq d^+_{c,i}((u_2, v_2))$. Repeating the same argument for every arc of $\overrightarrow{G} \,\square\, \overrightarrow{H}$, we get that c is locally irregular. ■

Regarding Conjecture 11.1, we get the following.

Corollary 11.6. *For every two oriented graphs $\overrightarrow{G}$ and $\overrightarrow{H}$ with $\chi'_{irr}(\overrightarrow{G}), \chi'_{irr}(\overrightarrow{H}) \leq 3$, we have $\chi'_{irr}(\overrightarrow{G} \,\square\, \overrightarrow{H}) \leq 3$.*

11.2 Decomposing oriented graphs into six locally irregular subgraphs

In this section, we show, in Theorem 11.9 below, that every oriented graph has irregular chromatic index at most 6. For this purpose, we first introduce the following observation stating that if an oriented graph $\overrightarrow{G}$ can be decomposed into arc-disjoint subgraphs $\overrightarrow{G_1}, \overrightarrow{G_2}, ..., \overrightarrow{G_k}$, then a locally irregular arc-colouring of $\overrightarrow{G}$ can be obtained by considering independent locally irregular arc-colourings of $\overrightarrow{G_1}, \overrightarrow{G_2}, ..., \overrightarrow{G_k}$. An undirected version of the same lemma was raised in Section 9.4, recall Observation 9.54.

Observation 11.7. *Let $\overrightarrow{G}$ be an oriented graph whose arc set $A(\overrightarrow{G})$ can be partitioned into k parts $A_1 \cup A_2 \cup ... \cup A_k$ such that*

$$\chi'_{irr}(\overrightarrow{G}[A_1]) \leq x_1, \chi'_{irr}(\overrightarrow{G}[A_2]) \leq x_2, ..., \chi'_{irr}(\overrightarrow{G}[A_k]) \leq x_k$$

for given values of $x_1, x_2, ..., x_k$. Then $\chi'_{irr}(\overrightarrow{G}) \leq \sum_{i=1}^{k} x_i$.

Proof. Let $c_1, c_2, ..., c_k$ be locally irregular x_1-, x_2-, ..., x_k-arc-colourings of $\overrightarrow{G}[A_1]$, $\overrightarrow{G}[A_2]$, ..., $\overrightarrow{G}[A_k]$, respectively, and denote by c the $(\sum_{i=1}^{k} x_i)$-arc-colouring of $\overrightarrow{G}$ defined as

$$c(\overrightarrow{a}) = (c_i(\overrightarrow{a}), i) \text{ for every } \overrightarrow{a} \in A(\overrightarrow{G}) \cap A_i.$$

By the partition of $A(\overrightarrow{G})$, every arc of $\overrightarrow{G}$ receives a colour by c, and c uses $\sum_{i=1}^{k} x_i$ colours. Besides, the subgraph of $\overrightarrow{G}$ induced by colour (j, i) is nothing but the subgraph of $\overrightarrow{G}[A_i]$ induced by colour j of c_i, which is locally irregular by the definition of c_i. All subgraphs of $\overrightarrow{G}$ induced by c are then locally irregular as required. ■

Observation 11.7 provides an easy upper bound on the irregular chromatic index of every oriented graph which can be partitioned into arc-disjoint subgraphs with upper-bounded irregular chromatic index. In particular, by showing below that every oriented graph can be arc-partitioned into two acyclic oriented graphs (which have irregular chromatic index at most 3, recall Theorem 11.4), we directly get that every oriented graph has irregular chromatic index at most 6.

Lemma 11.8. *The arc set of every oriented graph $\overrightarrow{G}$ can be partitioned into two parts $A_1 \cup A_2$ such that $\overrightarrow{G}[A_1]$ and $\overrightarrow{G}[A_2]$ are acyclic.*

Proof. Let $v_1, v_2, ..., v_n$ denote the vertices of $\overrightarrow{G}$ following an arbitrary ordering. Now consider every arc $\overrightarrow{v_i v_j}$ of $\overrightarrow{G}$, and

$$\begin{cases} \text{add } \overrightarrow{v_i v_j} \text{ to } A_1 \text{ if } i < j, \\ \text{add } \overrightarrow{v_i v_j} \text{ to } A_2 \text{ otherwise.} \end{cases}$$

Then observe that if $\overrightarrow{v_{i_1} v_{i_2} ... v_{i_k} v_{i_1}}$, with $i_1 < i_2 < ... < i_k$, were a circuit of $\overrightarrow{G}[A_1]$, then we would have both $i_1 < i_k$ and $i_k < i_1$, a contradiction. A similar contradiction can be deduced from any circuit of $\overrightarrow{G}[A_2]$. ■

Theorem 11.9. *For every oriented graph $\overrightarrow{G}$, we have $\chi'_{irr}(\overrightarrow{G}) \leq 6$.*

Proof. According to Lemma 11.8, there exists a partition $A_1 \cup A_2$ of $A(\overrightarrow{G})$ such that $\overrightarrow{G}[A_1]$ and $\overrightarrow{G}[A_2]$ are acyclic. Since every acyclic oriented graph has irregular chromatic index at most 3 according to Theorem 11.4, the thesis follows directly from Observation 11.7. ■

11.3 Locally Irregular 2-Arc-Colouring is NP-complete

In this section, we deal with the algorithmic complexity of Locally Irregular k-Arc-Colouring. Since checking whether an oriented graph $\overrightarrow{G}$ is locally irregular can be done in quadratic time, the problem Locally Irregular 1-Arc-Colouring is in P. In case Conjecture 11.1 turned out to be true, note that every problem Locally Irregular k-Arc-Colouring with $k \geq 3$ would be in P. At the moment, by Theorem 11.9 we get that Locally Irregular k-Arc-Colouring is in P for every $k \geq 6$ only. On the contrary, if Locally Irregular k-Arc-Colouring were shown to be NP-complete for some $k \in \{3, 4, 5\}$, then one would disprove Conjecture 11.1.

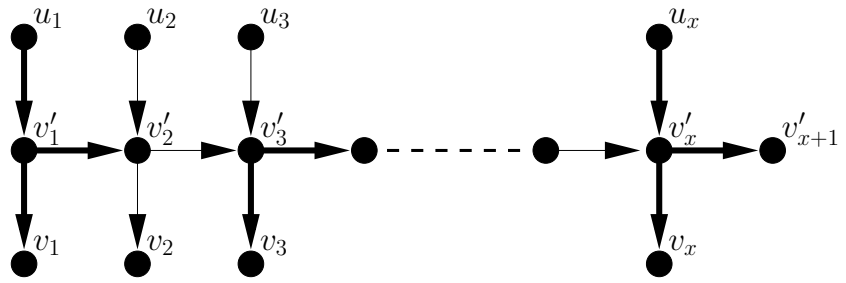


Figure 11.1: The 2-fiber gadget $\overrightarrow{F_2}$, and a locally irregular 2-arc-colouring of $\overrightarrow{F_2}$. Thick (resp. thin) arcs represent 1- (resp. 2-) coloured arcs.

In the light of the previous explanations, only Locally Irregular 2-Arc-Colouring is of interest at the moment. We prove this problem to be NP-complete in Theorem 11.15 below. This result implies that, even if Conjecture 11.1 turned out to be true, no good characterization of oriented graphs with irregular chromatic index at most 2 can exist, unless P=NP. This result also implies that Locally Irregular k-Arc-Colouring should not be fixed-parameter tractable when parameterized by k, the number of colours.

To prove Theorem 11.15, we first need to introduce some forcing gadgets, i.e. some oriented graphs which will be used in our reduction to "force" the propagation of a locally irregular 2-arc-colouring within an oriented graph.

Construction 11.10. The 2-*fiber gadget*, denoted $\overrightarrow{F_2}$, is depicted in Figure 11.1. We refer to the arcs $\overrightarrow{v'_1v_1}$, $\overrightarrow{v'_2v_2}$, .., $\overrightarrow{v'_xv_x}$ as the *outputs* of $\overrightarrow{F_2}$. Every output $\overrightarrow{v'_iv_i}$ with i odd is referred to as an *odd output*, or as an *even* output otherwise. Note that $\overrightarrow{F_2}$ is actually made of a same small pattern repeated consecutively from left to right. The dashed section of $\overrightarrow{F_2}$ means that this pattern can be repeated an arbitrary number of times, i.e. x can be arbitrarily large, so that $\overrightarrow{F_2}$ has arbitrarily many outputs, which are either even or odd, alternatively.

The 2-fiber gadget has the following colouring property.

Lemma 11.11. *In every locally irregular 2-arc-colouring c of $\overrightarrow{F_2}$, all of the even outputs of $\overrightarrow{F_2}$ have the same colour, while all of the odd outputs have the second colour. Besides, for every output $\overrightarrow{v'_iv_i}$ of $\overrightarrow{F_2}$, the vertex v'_i has $c(\overrightarrow{v'_iv_i})$-outdegree 2.*

Proof. Note that for every $i \in \{1, 2, ..., x\}$, the vertex u_i has α-outdegree 1 by c for an $\alpha \in \{1, 2\}$ and is adjacent to v'_i in the α-subgraph. For this reason, the two arcs $\overrightarrow{v'_iv_i}$ and $\overrightarrow{v'_iv'_{i+1}}$ cannot have distinct colours by c since otherwise v'_i would have α-outdegree 1 too.

Hence, all of the arcs outgoing from v'_i have the same colour. Suppose e.g. that all of the arcs outgoing from v'_1 are coloured 1. Then v'_1 has 1-outdegree 2, and v'_1 and v'_2 are adjacent in the 1-subgraph. For these reasons, all of the arcs outgoing from v'_2 cannot be coloured 1 since otherwise v'_2 would have 1-outdegree 2 too. Then all of the arcs outgoing from v'_2 are coloured 2 by c. Repeating the same argument from the left to the right of $\overrightarrow{F_2}$, we get that v'_i has 1-outdegree 2 for every odd i, while v'_i has 2-outdegree 2 otherwise, i.e. when i is even. Then every two consecutive outputs of $\overrightarrow{F_2}$ have distinct colours. ■

We now generalize the notion of k-*fiber gadget* for every $k \geq 3$.

Construction 11.12. Consider a value of $k \geq 3$ such that the i-fiber gadget has been defined for every $i \in \{2, 3, ..., k-1\}$. To obtain the k-*fiber gadget*, proceed as follows. Start from a directed path $\overrightarrow{v'_1v'_2...v'_xv'_{x+1}}$ for an arbitrary value of x. For every v'_i with $i \in \{1, 2, ..., x\}$, add arcs from v'_i towards $k-1$ new vertices with outdegree 0. Call v_i one such resulting vertex. Finally, identify v'_i with the heads of one distinct even output and one distinct odd output of

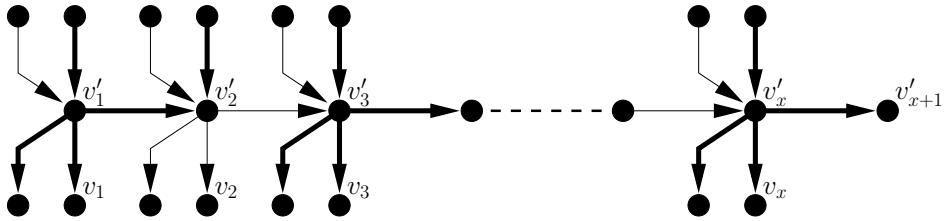


Figure 11.2: The 3-fiber gadget $\overrightarrow{F_3}$, and a locally irregular 2-arc-colouring of $\overrightarrow{F_3}$. Thick (resp. thin) arcs represent 1- (resp. 2-) coloured arcs.

each of $\overrightarrow{F_2}, \overrightarrow{F_3}, ..., \overrightarrow{F_{k-1}}$. Similarly as for $\overrightarrow{F_2}$, we refer to the arcs $\overrightarrow{v'_i v_i}$ of $\overrightarrow{F_k}$ as its *outputs*, making again the distinction between *even* and *odd outputs*

Example 11.13. Refer to Figure 11.2 for an illustration of the 3-fiber gadget $\overrightarrow{F_3}$, where the top-most arcs are outputs of $\overrightarrow{F_2}$.

The generalized fiber gadgets share the following colouring property (which is similar to the one we proved in Lemma 11.11 for $\overrightarrow{F_2}$).

Lemma 11.14. *In every locally irregular 2-arc-colouring c of $\overrightarrow{F_k}$, all of the even outputs of $\overrightarrow{F_k}$ have the same colour, while all of the odd outputs have the second colour. Besides, for every output $\overrightarrow{v'_i v_i}$ of $\overrightarrow{F_k}$, the vertex v'_i has $c(\overrightarrow{v'_i v_i})$-outdegree k.*

Proof. The proof is similar to the one of Lemma 11.11. Consider a vertex v'_i of $\overrightarrow{F_k}$. Since the heads of one even output and one odd output of $\overrightarrow{F_j}$ are identified with v'_i for every $j \in \{2, 3, ..., k-1\}$, there are two vertices w_1 and w_2 neighbouring v'_i such that:

- w_1 and v'_i are adjacent in the 1-subgraph induced by c,
- w_2 and v'_i are adjacent in the 2-subgraph induced by c,
- w_1 has 1-outdegree j and w_2 has 2-outdegree j.

Since this observation holds for every $j \in \{2, 3, ..., k-1\}$, note that all the arcs outgoing from v'_i must have the same colour by c since otherwise v'_i would have the same outdegree as one of its neighbours in either the 1- or 2-subgraph. Assume all of the arcs outgoing from v'_1 have colour 1 by c without loss of generality. Then all arcs outgoing from v'_2 cannot all be coloured 1 since otherwise v'_1 and v'_2 would be adjacent vertices with outdegree k in the 1-subgraph. Then all arcs outgoing from v'_2 have colour 2 by c. Again, by repeating this argument from left to right, similarly as in the proof of Lemma 11.11, we get that the colours of the outputs of $\overrightarrow{F_k}$ alternate between 1 and 2, and that the tail of each output has α-outdegree k, where α is the colour of this output by c. ■

The generalized fiber gadgets described above are actually not all necessary to prove our main result, but using these we can "generate" vertices with arbitrarily large outdegree in either the 1- or 2-subgraph induced by a locally irregular 2-arc-colouring of an oriented graph. Used conveniently (note in particular that if we identify the heads of one even output and one odd output of, say, $\overrightarrow{F_2}$, with a vertex v, then v cannot have outdegree 2 in the 1- and 2-subgraphs by a locally irregular 2-arc-colouring), one can construct arbitrarily many oriented graphs with various structures and which have irregular chromatic index 3. This should convince the reader that even if Conjecture 11.1 turned out to be true, oriented graphs with irregular chromatic index 2 do not have a predictable structure.

We are now ready to prove the main result of this section.

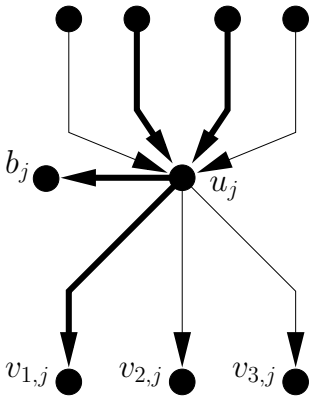


Figure 11.3: The clause gadget $\overrightarrow{G_F}(C_j)$, where the top-most arcs are outputs of the 3- and 4-fiber gadgets $\overrightarrow{F_3}$ and $\overrightarrow{F_4}$, and a locally irregular 2-arc-colouring of $\overrightarrow{G_F}(C_j)$. Thick (resp. thin) arcs represent 1- (resp. 2-) coloured arcs.

Theorem 11.15. Locally Irregular 2-Arc-Colouring *is NP-complete, even when restricted to oriented graphs with maximum indegree and outdegree at most* 4.

Proof. Clearly Locally Irregular 2-Arc-Colouring is in NP since, given a 2-arc-colouring of $\overrightarrow{G}$, one can easily check whether the two subgraphs it induces are locally irregular (this property can be checked in quadratic time).

We now prove that Locally Irregular 2-Arc-Colouring is NP-hard, and thus NP-complete since it is also in NP, by reduction from Monotone Not-All-Equal 3-Satisfiability. Recall that every formula F of Monotone Not-All-Equal 3-Satisfiability can be supposed to have all its clauses having three distinct literals, see Observation 1.53. From F, we construct an oriented graph $\overrightarrow{G_F}$ such that

$$F \text{ is not-all-equal satisfiable}$$
$$\Leftrightarrow$$
$$\overrightarrow{G_F} \text{ admits a locally irregular 2-arc-colouring } c_F.$$

We design $\overrightarrow{G_F}$ in such a way that the propagation of c_F along $\overrightarrow{G_F}$ is representative of the constraints attached to Monotone Not-All-Equal 3-Satisfiability, i.e. of the consequences on F of setting such or such variable of F to true. This is typically done by designing gadgets with specific colouring properties. Throughout this proof, colour 1 of c_F must be thought of as the truth value *true*, while colour 2 represents the truth value *false* of a truth assignment to the variables of F (one could actually switch these two equivalences as we are dealing with Monotone Not-All-Equal 3-Satisfiability).

The first requirement of Monotone Not-All-Equal 3-Satisfiability we have to "translate" is that a clause of F is considered satisfied if and only if it has at least one true and one false variable. This is done by "transforming" every clause $C_j = (x_{i_1} \vee x_{i_2} \vee x_{i_3})$ into a *clause gadget* $\overrightarrow{G_F}(C_j)$ in $\overrightarrow{G_F}$ with three special arcs $\overrightarrow{a_1}$, $\overrightarrow{a_2}$ and $\overrightarrow{a_3}$ representing the variables of C_j, and such that all of these three arcs cannot have the same colour by c_F. Assuming that, say, $c_F(\overrightarrow{a_1}) = 1$ (resp. $c_F(\overrightarrow{a_1}) = 2$) simulates the fact that x_{i_1} supplies C_j with value true (resp. false), the requirement above then follows directly from the colouring property of $\overrightarrow{G_F}(C_j)$.

Consider then every clause $C_j = (x_{i_1} \vee x_{i_2} \vee x_{i_3})$, whose some variables (at most two of them, recall Observation 1.48) may be the same. The clause gadget $\overrightarrow{G_F}(C_j)$, associated with C_j, is obtained as follows (see Figure 11.3). Add five vertices u_j, $v_{1,j}$, $v_{2,j}$, $v_{3,j}$ and b_j to $\overrightarrow{G_F}$, as well as all arcs from u_j towards every vertex in $\{v_{1,j}, v_{2,j}, v_{3,j}, b_j\}$. Now identify u_j with the heads of one even output and one odd output of each of $\overrightarrow{F_3}$ and $\overrightarrow{F_4}$, where $\overrightarrow{F_3}$ and $\overrightarrow{F_4}$ are the 3- and

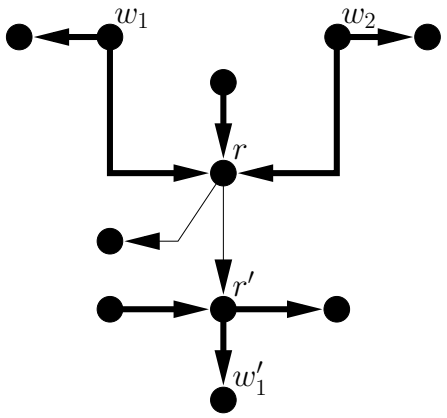


Figure 11.4: The collecting gadget $\overrightarrow{G^Y}$, and a locally irregular 2-arc-colouring of $\overrightarrow{G^Y}$. Thick (resp. thin) arcs represent 1- (resp. 2-) coloured arcs.

4-fiber gadgets. The arcs $\overrightarrow{a_1}$, $\overrightarrow{a_2}$ and $\overrightarrow{a_3}$ mentioned in the explanations above actually refer to $\overrightarrow{u_j v_{1,j}}$, $\overrightarrow{u_j v_{2,j}}$ and $\overrightarrow{u_j v_{3,j}}$. Besides, one has to think of every vertex $v_{i,j}$ as a vertex associated with the ith variable of C_j. We show that $\overrightarrow{G_F}(C_j)$ cannot have all of its arcs $\overrightarrow{u_j v_{1,j}}$, $\overrightarrow{u_j v_{2,j}}$ and $\overrightarrow{u_j v_{3,j}}$ having the same colour by c_F, as required.

Claim 11.16. *If C_j is a clause of F, then one arc of $\overrightarrow{u_j v_{1,j}}$, $\overrightarrow{u_j v_{2,j}}$ and $\overrightarrow{u_j v_{3,j}}$ has one colour by c_F, while the other two arcs have the second colour.*

Proof. The claim follows from the facts that u_j has outdegree 4 and is adjacent to vertices with outdegree 3 or 4 in the 1- and 2-subgraphs induced by c_F, namely the tails of some outputs of $\overrightarrow{F_3}$ and $\overrightarrow{F_4}$ whose heads were identified with u_j. ■

The second requirement of Monotone Not-All-Equal 3-Satisfiability we have to model is that, by a truth assignment to the variables of F, a variable provides the same truth value to every clause it appears in. At the moment, this requirement is not met as c_F may be locally irregular but with, say, $c_F(\overrightarrow{u_j v_{i,j}}) = 1$ and $c_F(\overrightarrow{u_{j'} v_{i',j'}}) = 2$ with the ith variable of C_j being identical to the i'th variable of $C_{j'}$, say x_ℓ. Following our analogy above, this would simulates that x_ℓ belongs to both of the clauses C_j and $C_{j'}$, but x_ℓ provides true to C_j and false to $C_{j'}$ by a truth assignment, which is impossible. Hence, we have to check somehow whether all the arcs $\overrightarrow{u_{j_1} v_{i_1,j_1}}$, $\overrightarrow{u_{j_2} v_{i_2,j_2}}$, ..., $\overrightarrow{u_{j_{n(x_\ell)}} v_{i_{n(x_\ell)},j_{n(x_\ell)}}}$, representing the membership of x_ℓ to the clauses $C_{j_1}, C_{j_2}, ..., C_{j_{n(x_\ell)}}$ of F which contain x_ℓ, have the same colour by c_F.

This is done by using the *collecting gadget* $\overrightarrow{G^Y}$ depicted in Figure 11.4. The arcs $\overrightarrow{w_1 r}$ and $\overrightarrow{w_2 r}$ are called the *inputs* of $\overrightarrow{G^Y}$, while $\overrightarrow{r' w'_1}$ is its *output*. Note that w_1, w_2 and r' have outdegree 2. This gadget $\overrightarrow{G^Y}$ has the following colouring property.

Claim 11.17. *Let c be a locally irregular 2-arc-colouring of $\overrightarrow{G^Y}$ such that the two arcs outgoing from w_1 have the same colour, and the two arcs outgoing from w_2 have the same colour. Then $c(\overrightarrow{w_1 r}) = c(\overrightarrow{w_2 r}) = c(\overrightarrow{r' w'_1})$.*

Proof. Assume $c(\overrightarrow{w_1 r}) = 1$ and $c(\overrightarrow{w_2 r}) = 2$ without loss of generality. In particular, note that w_1 and r are adjacent in the 1-subgraph, and that w_2 and r are adjacent in the 2-subgraph. Besides, by assumption w_1 has 1-outdegree 2 while w_2 has 2-outdegree 2. For these reasons, note that the two arcs outgoing from r cannot have the same colour since otherwise r would have 1- or 2-outdegree 2, a contradiction. Then one arc outgoing from r has colour 1 by c while the other arc has colour 2, implying that r has both 1- and 2-outdegree 1. But then we necessarily get a contradiction while colouring the arc outgoing from the vertex with outdegree 1 attached to r.

On the contrary, note that if $c(\overrightarrow{w_1r}) = c(\overrightarrow{w_2r}) = 1$ without loss of generality, then, so that we avoid every contradiction mentioned above, we have to colour 2 all arcs outgoing from r. Then r and r' are neighbouring vertices in the 2-subgraph, and r has 2-outdegree 2. Since there is a vertex with outdegree 1 attached to r', again we cannot colour the two arcs outgoing from r' with distinct colours. Then we have to colour 1 the two arcs outgoing from r'. ■

Roughly speaking, assuming we are given two arcs $\overrightarrow{a_1}$ and $\overrightarrow{a_2}$ whose tails necessarily have outdegree 2 in the $c_F(\overrightarrow{a_1})$- and $c_F(\overrightarrow{a_2})$-subgraphs, respectively, we can "check" whether $c_F(\overrightarrow{a_1}) = c_F(\overrightarrow{a_2})$. Namely, take a copy of $\overrightarrow{G^Y}$ and "replace" the arcs $\overrightarrow{w_1r}$ and $\overrightarrow{w_2r}$ with $\overrightarrow{a_1}$ and $\overrightarrow{a_2}$, respectively. We refer to this operation as *collecting* $\overrightarrow{a_1}$ and $\overrightarrow{a_2}$ (with a copy of $\overrightarrow{G^Y}$). According to Claim 11.17, the arc-colouring c_F cannot then be extended to the collecting gadget if $c_F(\overrightarrow{a_1}) \neq c_F(\overrightarrow{a_2})$. Recall further that if $c_F(\overrightarrow{a_1}) = c_F(\overrightarrow{a_2})$, then all of the arcs outgoing from the tail of the output of the collecting gadget have colour $c_F(\overrightarrow{a_1})$, and the tail of the output thus has $c_F(\overrightarrow{a_1})$-outdegree 2. In some sense, this property means that the output of a collecting gadget "memorizes" the colour used at its two inputs.

To end up the construction of $\overrightarrow{G_F}$, proceed as follows. Consider every variable x_i of F with $i \in \{1, 2, ..., n\}$, and let $\overrightarrow{o_1^i}, \overrightarrow{o_2^i}, ..., \overrightarrow{o_{n(x_i)}^i}$ denote the $n(x_i)$ arcs of $\overrightarrow{G_F}$ representing the membership of x_i to a clause. More precisely, these arcs are of the form $\overrightarrow{u_jv_{i',j}}$, where $i' \in \{1, 2, 3\}$ and $j \in \{1, 2, ..., m\}$, and x_i is the i'th variable of C_j. Recall further that if one of these arcs $\overrightarrow{o}$ is coloured, say, 1 by c_F, then the tail of $\overrightarrow{o}$ has 1-outdegree 2. Then start by collecting $\overrightarrow{o_1^i}$ and $\overrightarrow{o_2^i}$ with a copy $\overrightarrow{G_1^i}$ of $\overrightarrow{G^Y}$. Next collect the output of $\overrightarrow{G_1^i}$ and $\overrightarrow{o_3^i}$ with a new copy $\overrightarrow{G_2^i}$ of $\overrightarrow{G^Y}$. Then collect the output of $\overrightarrow{G_2^i}$ and $\overrightarrow{o_4^i}$ with a new copy $\overrightarrow{G_3^i}$ of $\overrightarrow{G^Y}$. And so on. This procedure uses $n(x_i) - 1$ copies of $\overrightarrow{G^Y}$.

We claim that we have the desired equivalence between not-all-equal satisfying F and propagating c_F in $\overrightarrow{G_F}$ in a locally irregular way. If c_F can be propagated in this way, then, for every clause $C_j = (x_{i_1} \vee x_{i_2} \vee x_{i_3})$ of F, one arc of $u_jv_{1,j}, u_jv_{2,j}, u_jv_{3,j}$ has a colour by c_F while the other two arcs have the other colour (Claim 11.16). Besides, this arc-colouring, because of the collecting gadgets, has the property that all arcs corresponding to the membership of a same variable to some clauses have the same colour (Claim 11.17). Assuming that having $c_F(\overrightarrow{u_jv_{i',j}}) = 1$ (resp. $c_F(\overrightarrow{u_jv_{i',j}}) = 2$) simulates the fact that the i'th variable of C_j is set to true (resp. false), we can directly deduce a truth assignment not-all-equal satisfying F from c_F, and vice-versa. This completes the proof.

The last part of the statement follows from the fact that $\overrightarrow{G_F}$ necessarily has maximum indegree and outdegree at most 4. In particular, the fiber gadgets $\overrightarrow{F_2}$, $\overrightarrow{F_3}$ and $\overrightarrow{F_4}$, which are the only fiber gadgets used for the reduction, have this property. It is then easily checked by hand that the clause gadgets also have this property, as well as the collecting gadgets. It is actually worth mentioning that we only designed these collecting gadgets in order to reduce the indegrees and outdegrees of the vertices of $\overrightarrow{G_F}$. Our reduction could be performed without making use of these, but, in doing so, (in particular) the maximum indegree of $\overrightarrow{G_F}$ is generally not upper-bounded (it would be roughly equal to $\max\{n(x_i) : x_i \in F\}$). ■

11.4 Conclusion and open questions

In this chapter, we have introduced and investigated locally irregular arc-colouring of oriented graphs, our investigations being guided by Conjecture 11.1 we have raised. Although we have not managed to prove Conjecture 11.1, which we have verified for several classes of graphs in Section 11.1, we have however proved a weaker version of Conjecture 11.1, recall Theorem 11.9, which is not so far from the conjecture. One good point behind Theorem 11.9 is that this result provides a constant upper bound on the irregular chromatic index of oriented graphs, while

such a constant upper bound is still not known in the context of undirected graphs, recall our investigations from Chapter 9.

Maybe our proof of Theorem 11.9, i.e. first edge-partitioning an oriented graph into k subgraphs and then independently decomposing each of these subgraphs into locally irregular subgraphs, could be pushed forwards to get a better constant upper bound of the irregular chromatic index of oriented graphs. It would be e.g. interesting investigating whether every oriented graph can be edge-partitioned into two subgraphs with irregular chromatic index at most 2 (to basically get that $\chi'_{irr}(\overrightarrow{G}) \leq 4$ for every oriented graph $\overrightarrow{G}$). Towards this direction, one judicious direction would be to study which classes of oriented graphs have irregular chromatic index at most 2.

Question 11.18. *Which classes of oriented graphs have irregular chromatic index at most 2?*

In particular, it is worth mentioning that Theorem 11.4 is tight since there are acyclic oriented graphs with irregular chromatic index exactly 3 (so we cannot improve Theorem 11.9 from 6 to 4 by just improving Theorem 11.4). To be convinced of this statement, one just has to note that Locally Irregular 2-Arc-Colouring remains NP-complete when restricted to acyclic oriented graphs as it can be easily checked that the reduction in the proof of Theorem 11.15 actually provides acyclic oriented graphs.

In case this direction shown up to be a wrong track, any improvement of Theorem 11.9 would be a good step towards Conjecture 11.1 anyway.

Among all classes of oriented graphs we have considered regarding Conjecture 11.1, and which do not appear in Section 11.1, the case of tournaments is intriguing. Although it is easy to show that some restricted families of tournaments agree with Conjecture 11.1, e.g. transitive tournaments (which are locally irregular), we have not been able to find any argument for the general case. Until a proof of Conjecture 11.1 is exhibited, which would solve the problem, we address the following weaker problem.

Problem 11.19. *Prove that we have $\chi'_{irr}(\overrightarrow{T}) \leq 3$ for every tournament $\overrightarrow{T}$.*

Chapter 12

Conclusion to Part II

In Part II, we have studied several edge- and arc-weighting notions permitting the distinguishing of the adjacent vertices of an undirected or oriented graph. We have first focused on the case of undirected graphs in Chapters 8 and 9, before moving to oriented graphs, regarding which still few such notions have been studied, in Chapters 10 and 11.

In Chapter 8, we have considered the notion of neighbour-sum-distinguishing edge-weighting of graphs, which is the notion underlying the well-known 1-2-3 Conjecture. Our main result in this context is algorithmic, and states that Neighbour-Sum-Distinguishing $\{a,b\}$-Edge-Weighting is NP-complete no matter what is $\{a, b\}$, recall Theorem 8.1.

Then we have introduced the notion of locally irregular edge-colouring of graphs in Chapter 9, and have given very first results and properties related to this notion. So that the range of our investigations is clear, we have first had to formally identify the range of graphs which are colourable (with regards to locally irregular edge-colouring). This has resulted in Proposition 9.5. Once done, we have been then able to study the decomposition of colourable graphs into locally irregular subgraphs. As a guiding thread, we have raised Conjecture 9.9, which we have verified for various classes of graphs throughout Section 9.2. From all our results, Theorem 9.21 on regular graphs is surely the most interesting one. We have finally closed Chapter 9 by studying algorithmic aspects related to locally irregular edge-colouring of graphs. In particular, we have proved that determining the irregular chromatic index of a colourable graph is NP-complete in general (Theorem 9.44), but can be done in linear time in the context of trees (Corollary 9.43).

In Chapter 10, we have introduced oriented notions according to which we have studied analogues of the 1-2-3 and 1-2 Conjectures. The arising problems have then appeared to be way easier to tackle than the original undirected ones. This has resulted in proofs that every oriented graph has neighbour-outsum-distinguishing chromatic index at most 3 (Theorem 10.1), and that not all oriented graphs admit a neighbour-sum-distinguishing 2-total-weighting (Corollary 10.31). We have then focused on the algorithmic aspects related to neighbour-outsum-distinguishing arc-weighting of oriented graphs. Namely, we have proved that Neighbour-Outsum-Distinguishing 2-Arc-Weighting is NP-complete, recall Theorem 10.20, rejecting the existence of an easy classification of oriented graphs with neighbour-outsum-distinguishing chromatic index at most 2 (unless $\mathsf{P} = \mathsf{NP}$).

Finally, we have studied, in Chapter 11, an oriented version of the problem considered in Chapter 9, about which we have raised a similar conjecture, recall Conjecture 11.1. Again, the oriented version of this conjecture has shown up to be easier to deal with. Namely, we have managed to prove a weaker version of Conjecture 11.1 by showing that every oriented graph has irregular chromatic index at most 6 (Theorem 11.9). This is significant as we did not manage to prove such a constant upper bound on the irregular chromatic index of undirected graphs. Then, similarly as in the undirected case, we have proved that, even if Conjecture 11.1 turned out to be true, it would remain NP-complete to determine the irregular chromatic index of an oriented graph, recall Theorem 11.15.

Our mentioned above investigations have given rise to new directions for future works, see concluding Sections 8.6, 9.4, 10.5, and 11.4. It is also worth mentioning that possibilities for

imagining new vertex-distinguishing edge-weighting notions and problems are wide, so many other perspectives may arise by introducing new such notions by either combining two existing vertex-distinguishing edge-weighting notions, or introducing new vertex-distinguishing parameters. In particular, continuing studying oriented analogues of undirected vertex-distinguishing problems seems an appealing direction to us.

Bibliography

[1] L. Addario-Berry, R.E.L. Aldred, K. Dalal, and B.A. Reed. Vertex colouring edge partitions. *Journal of Combinatorial Theory, Series B*, 94(2):237–244, 2005.

[2] L. Addario-Berry, K. Dalal, C. McDiarmid, B. A. Reed, and A. Thomason. Vertex-colouring edge-weightings. *Combinatorica*, 27(1):1–12, 2007.

[3] L. Addario-Berry, K. Dalal, and B.A. Reed. Degree constrained subgraphs. *Discrete Applied Mathematics*, 156(7):1168–1174, 2008.

[4] A. Ahadi, A. Dehghan, and M-R. Sadeghi. Algorithmic complexity of proper labeling problems. *Theoretical Computer Science*, 495:25–36, 2013.

[5] Y. Alavi, G. Chartrand, F.R.K. Chung, P. Erdős, R.L. Graham, and O.R. Oellermann. How to define an irregular graph. *Journal of Graph Theory*, 11(2):235–249, 1987.

[6] N. Alon. Combinatorial Nullstellensatz. *Combinatorics, Probability and Computing*, 8(1-2):7–29, 1999.

[7] N. Alon and J.H. Spencer. *The Probabilistic Method, 3rd edition*. Wiley-Interscience, 2008.

[8] G.E. Andrews. *The Theory of Partitions*. Cambridge University Press, 1998.

[9] K. Appel and W. Haken. Solution of the Four Color Map Problem. *Scientific American*, 237(4):108–121, 1977.

[10] M. Araya and G. Wiener. On Cubic Planar Hypohamiltonian and Hypotraceable Graphs. *Electronic Journal of Combinatorics*, 18(1):P85, 2011.

[11] D. Barth, O. Baudon, and J. Puech. Decomposable trees: a polynomial algorithm for tripodes. *Discrete Applied Mathematics*, 119(3):205–216, 2002.

[12] D. Barth and H. Fournier. A degree bound on decomposable trees. *Discrete Mathematics*, 306(5):469–477, 2006.

[13] D. Barth, H. Fournier, and R. Ravaux. On the shape of decomposable trees. *Discrete Mathematics*, 309:3882–3887, 2009.

[14] T. Bartnicki, J. Grytczuk, and S. Niwczyk. Weight choosability of graphs. *Journal of Graph Theory*, 60(3):242–256, 2009.

[15] O. Baudon, J. Bensmail, F. Foucaud, and M. Pilśniak. Structural properties of recursively partitionable graphs with connectivity. 2012. Preprint available online at http://hal.archives-ouvertes.fr/hal-00672505.

[16] O. Baudon, J. Bensmail, R. Kalinowski, A. Marczyk, J. Przybyło and M. Woźniak. On the Cartesian product of an arbitrarily partitionable graph and a traceable graph. *Discrete Mathematics and Theoretical Computer Science*, 16(2):225–232, 2014.

[17] O. Baudon, J. Bensmail, J. Przybyło and M. Woźniak. On decomposing regular graphs into locally irregular subgraphs. 2013. Preprint available online at http://www.ii.uj.edu.pl/documents/12980385/26042491/MD_65.pdf.

[18] O. Baudon, J. Bensmail, J. Przybyło and M. Woźniak. Partitioning powers of traceable or Hamiltonian graphs. *Theoretical Computer Science*, 520:133–137, 2014.

[19] O. Baudon, J. Bensmail and É. Sopena. Partitioning Harary graphs into connected subgraphs containing prescribed vertices. 2012. Preprint available online at http://hal.archives-ouvertes.fr/hal-00687607.

[20] O. Baudon, J. Bensmail and É. Sopena. On the complexity of determining the irregular chromatic index of a graph. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00921849.

[21] O. Baudon, J. Bensmail and É. Sopena. An oriented version of the 1-2-3 Conjecture. To appear in *Discussiones Mathematicae Graph Theory*. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00849065.

[22] O. Baudon, F. Foucaud, J. Przybyło, and M. Woźniak. On the structure of arbitrarily partitionable graphs with given connectivity. *Discrete Applied Mathematics*, 162:381–385, 2014.

[23] O. Baudon, F. Gilbert, and M. Woźniak. Recursively arbitrarily vertex-decomposable suns. *Opuscula Mathematica*, 31(4):533–547, 2011.

[24] O. Baudon, F. Gilbert, and M. Woźniak. Recursively arbitrarily vertex-decomposable graphs. *Opuscula Mathematica*, 32(4):689–706, 2012.

[25] O. Baudon, J. Przybyło, and M. Woźniak. On minimal arbitrarily partitionable graphs. *Information Processing Letters*, 112:697–700, 2012.

[26] D. Bauer, H.J. Broersma, and H.J. Veldman. Not every 2-tough graph is Hamiltonian. *Discrete Applied Mathematics*, 99:317–321, 2000.

[27] L.W. Beineke and M.D. Plummer. On the 1-factors of a non-separable graph. *Journal of Combinatorial Theory*, 2(3):285–289, 1967.

[28] J. Bensmail. Determining the irregular chromatic index of a graph. In *Proceedings European Conference on Combinatorics, Graph Theory and Applications, EuroComb 2013*, 2013.

[29] J. Bensmail. *Préaffectation de sommets dans les graphes arbitrairement partitionnables*. Master's thesis (in French), Université Bordeaux 1, 2011.

[30] J. Bensmail. On the complexity of partitioning a graph into a few connected subgraphs. To appear in *Journal of Combinatorial Optimization*. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00762612.

[31] J. Bensmail. On the longest path in a recursively arbitrarily partitionable graph. *Opuscula Mathematica*, 33(4):631–640, 2013.

[32] J. Bensmail. On the path cover number of k-assignable arbitrarily partitionable graphs. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00881861.

[33] J. Bensmail. On three polynomial kernels of sequences for arbitrarily partitionable graphs. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00875371.

[34] J. Bensmail. The vertex-colouring $\{a, b\}$-edge-weighting problem is NP-complete for every pair of weights. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00826346.

[35] J. Bensmail and G. Renault. Decomposing oriented graphs into 6 locally irregular oriented graphs. 2013. Preprint available online at http://hal.archives-ouvertes.fr/hal-00869778.

[36] J.A. Bondy and U.S.R. Murty. *Graph Theory with Applications*. North Holland, 1985.

[37] M. Borowiecki, J. Grytczuk, and M. Pilśniak. Coloring chip configurations on graphs and digraphs. *Information Processing Letters*, 112(1-2):1–4, 2012.

[38] S. Brandt. Finding vertex decompositions in dense graphs. In *Proceedings 15th Workshop on Graph Theory - Colourings, Independence and Dominations (CID)*, 2013.

[39] H. Broesma, D. Kratsch, and G. Woeginger. Fully decomposable split graphs. *European Journal of Combinatorics*, 34(3):567–575, 2013.

[40] R.L. Brooks. On colouring the nodes of a network. In *Proceedings Cambridge Philosophical Society, Mathematical and Physical Sciences*, 37:194–197, 1941.

[41] G.J. Chang, C. Lu, J. Wu, and Q. Yu. Vertex-coloring edge-weightings of graphs. *Taiwanese Journal of Mathematics*, 15(4):1807–1813, 2011.

[42] G. Chartrand, M.S. Jacobson, J. Lehel, O.R. Oellermann, S. Ruiz, and F. Saba. Irregular networks. *Congressus Numerantium*, 64:197–210, 1988.

[43] S. Cichacz, A. Görlich, A. Marczyk, J. Przybyło, and M. Woźniak. Arbitrarily vertex decomposable caterpillars with four or five leaves. *Discussiones Mathematicae Graph Theory*, 26:291–305, 2006.

[44] S. Cook. The complexity of theorem proving procedures. In *Proceedings 3rd Annual ACM Symposium on Theory of Computing*, 151–158, 1971.

[45] A. Czumaj and W-B. Strothmann. Bounded degree spanning trees. In *Algorithms - ESA'97, Lecture Notes in Computer Science*, 1284:104–117, 1997.

[46] A. Davoodi and B. Omoomi. On the 1-2-3-conjecture. 2012. Preprint available online at http://arxiv.org/abs/1205.3266.

[47] M. Dell'Amico and S. Martello. Reduction of the Three-Partition Problem. *Journal of Combinatorial Optimization*, 3:17–30, 1999.

[48] R. Diestel. *Graph Theory, 4th edition*. Springer, 2010.

[49] A.A. Diwan. Partitioning 2-connected graphs into connected subgraphs. 2003. Preprint available online at http://www.cse.iitb.ac.in/∼aad/postscript/4part.ps.

[50] A.A. Diwan and M.P. Kurhekar. Plane triangulations are 6-partitionable. *Discrete Mathematics*, 256:91–103, 2002.

[51] R.G. Downey and M.R. Fellows. *Parameterized Complexity*. Springer, 1999.

[52] A. Dudek and D. Wajc. On the complexity of vertex-coloring edge-weightings. *Discrete Mathematics Theoretical Computer Science*, 13(3):45–50, 2011.

[53] D. Duffus, R.J. Gould, and M.S. Jacobson. Forbidden subgraphs and the Hamiltonian theme. In *Proceedings 4th International Conference on Combinatorics, Graph Theory, and Computing*, 297–315, 1980.

[54] M.E. Dyer and A.M. Frieze. On the complexity of partitioning graphs into connected subgraphs. *Discrete Applied Mathematics*, 10:139–153, 1985.

[55] M.E. Dyer and A.M. Frieze. Planar 3DM is NP-complete. *Journal of Algorithms*, 7:174–184, 1986.

[56] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[57] P. Erdős and A. Rényi. On Random Graphs. I. *Publicationes Mathematicae*, 6:290–297, 1959.

[58] P. Festa, P.M. Pardalos, and M.G.C. Resende. *Feedback set problems*. In *Handbook of combinatorial optimization, Supplement Volume A*, Kluwer Academic, 209–258, 1999.

[59] P. Flajolet and R. Sedgewick. *Analytic combinatorics*. Cambridge University Press, 2009.

[60] A.M. Frieze and T. Łuczak. On the independence and chromatic numbers of random regular graphs. *Journal of Combinatorial Theory, Series B*, 54:123–132, 1992.

[61] J.A. Gallian. A dynamic survey of graph labeling. *Electronic Journal of Combinatorics*, 6, 1998.

[62] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.

[63] M.R. Garey, D.S. Johnson, and R.E. Tarjan. The planar Hamiltonian circuit problem is NP-complete. *SIAM Journal on Computing*, 5(4):704–714, 1976.

[64] F. Gilbert. *Graphes arbitrairement décomposables*. Master's thesis (in French), Université Bordeaux 1, 2007.

[65] R.J. Gould. Advances on the Hamiltonian problem - A survey. *Graphs and Combinatorics*, 19:7–52, 2003.

[66] E. Győri. On division of graphs to connected subgraphs. In *Proceedings 5th Hungarian Combinational Colloquium*, 485–494, 1978.

[67] E. Győri and C. Palmer. A new type of edge-derived vertex coloring. *Discrete Mathematics*, 309(22):6344–6352, 2009.

[68] F. Havet, N. Paramaguru, and R. Sampathkumar. Detection number of bipartite graphs and cubic graphs. 2012. Preprint available online at http://hal.inria.fr/hal-00744365/.

[69] I. Holyer. The NP-Completeness of Edge-Colouring. *SIAM Journal on Computing*, 10(4):718–720, 1981.

[70] L. Hofer and T. Lambert. Study of the article: "An $\mathcal{O}(k^2n^2)$ algorithm to find a k-partition in a k-connected graph". 2014. Technical report available online at http://www.labri.fr/perso/jbensmai/students/hofer-lambert.pdf

[71] M. Horňák, A. Marczyk, I. Schiermeyer, and M. Woźniak. Dense arbitrarily vertex decomposable graphs. *Graphs and Combinatorics*, 28:807–821, 2012.

[72] M. Horňák, Zs. Tuza, and M. Woźniak. On-line arbitrarily vertex decomposable trees. *Discrete Applied Mathematics*, 155:1420–1429, 2007.

[73] M. Horňák and M. Woźniak. Arbitrarily vertex decomposable trees are of degree at most 6. *Opuscula Mathematica*, 23:49–62, 2003.

[74] W. Imrich and S. Klavžar. *Product Graphs: Structure and Recognition*. Wiley-Interscience, 2000.

[75] R. Kalinowski, M. Pilśniak, M. Woźniak, and I.A. Zioło. Arbitrarily vertex decomposable suns with few rays. *Discrete Mathematics*, 309:3726–3731, 2009.

[76] R. Kalinowski, M. Pilśniak, M. Woźniak, and I.A. Zioło. On-line arbitrarily vertex decomposable suns. *Discrete Mathematics*, 309:6328–6336, 2009.

[77] M. Kalkowski. A note on the 1,2-conjecture. 2009. Private communication.

[78] M. Kalkowski, M. Karoński, and F. Pfender. Vertex colouring edge weightings with integer weights at most 6. *Rostocker Mathematisches Kolloquium*, 64:39–43, 2009.

[79] M. Kalkowski, M. Karoński, and F. Pfender. Vertex-coloring edge-weightings: towards the 1-2-3-conjecture. *Journal of Combinatorial Theory, Series B*, 100(3):347–349, 2010.

[80] D. Karger, R. Motwani and G.D.S. Ramkumar. On approximating the longest path in a graph. *Algorithmica*, 18(1):82–98, 1997.

[81] M. Karoński, T. Łuczak, and A. Thomason. Edge weights and vertex colours. *Journal of Combinatorial Theory, Series B*, 91(1):151–157, 2004.

[82] R.M. Karp. *Reducibility Among Combinatorial Problems*. In *Complexity of Computer Computations*, R.E. Miller and J.W. Thatcher, New York: Plenum. Press, 85–103, 1972.

[83] M. Khatirinejad, R. Naserasr, M. Newman, B. Seamone, and B. Stevens. Digraphs are 2-weight choosable. *Electronic Journal of Combinatorics*, 18(1): P21, 2011.

[84] M. Khatirinejad, R. Naserasr, M. Newman, B. Seamone, and B. Stevens. Vertex-colouring edge-weightings with two edge weights. *Discrete Mathematics Theoretical Computer Science*, 14(1), 2012.

[85] P. Laroche. Planar 1-in-3 satisfiability is NP-complete. In *Proceedings ASMICS Workshop on Tilings, Deuxième Journée Polyominos et pavages, École Normale Supérieure de Lyon*, 117–126, 1992.

[86] J. Lehel. Facts and quests on degree irregular assignments. In *Graph Theory, Combinatorics, and Applications*, Wiley-Interscience, 765–781, 1991.

[87] D. Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1981.

[88] L. Lovász. Coverings and coloring of hypergraphs. In *Proceedings 4th South-eastern Conference on Combinatorics, Graph Theory, and Computing*, 3–12, 1973.

[89] L. Lovász. A homology theory for spanning trees of a graph. *Acta Mathematica Academiae Scientiarum Hungaricae*, 30(3-4):241–251, 1977.

[90] J. Ma and S.H. Ma. An $\mathcal{O}(k^2n^2)$ algorithm to find a k-partition in a k-connected graph. *Journal of Computer Science and Technology*, 9(1):86–91, 1994.

[91] A. Marczyk. An Ore-type condition for arbitrarily vertex decomposable graphs. *Discrete Mathematics*, 309:3588-3594, 2009.

[92] A.R. Meyer and L.J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential time. In *Proceedings 13th Annual Symposium on Switching and Automata Theory*, 125–129, 1972.

[93] T.H. Miyano, T. Nishizeki, N. Takahashi, and S. Uneo. An algorithm for tripartitioning 3-connected graphs. *Journal of Information Processing Society of Japan*, 31(5):584–592, 1990.

[94] M. Molloy and B. Reed. *Graph Colouring and the Probabilistic Method*. Springer, 2002.

[95] J. W. Moon. *Topics on tournaments*. Holt, Rinehart and Winston, 1968.

[96] B.M.E. Moret. Planar NAE3SAT is in P. *SIGACT News*, 19(2):51–54, 1988.

[97] S-I. Nakano, S. Rahman, and T. Nishizeki. A linear-time algorithm for four-partitioning four-connected planar graphs. *Information Processing Letters*, 62(6):315–322, 1997.

[98] T. Nierhoff. A tight bound on the irregularity strength of graphs. *SIAM Journal on Discrete Mathematics*, 13(3):313–323, 2000.

[99] O. Ore. A note on Hamiltonian circuits. *American Mathematical Monthly*, 67: 55, 1960.

[100] J. Przybyło. A note on a neighbour-distinguishing regular graphs total-weighting. *Electronic Journal of Combinatorics*, 15(1): N35, 2008.

[101] J. Przybyło. On decomposing graphs of large minimum degree into locally irregular subgraphs. 2013. Private communication.

[102] J. Przybyło and M. Woźniak. On a 1,2 conjecture. *Discrete Mathematics Theoretical Computer Science*, 12(1):101–108, 2010.

[103] J. Przybyło and M. Woźniak. Total weight choosability of graphs. *Electronic Journal of Combinatorics*, 18(1): P112, 2011.

[104] R. Ravaux. *Graphes arbitrairement partitionnables : propriétés structurelles et algorithmiques*. Ph.D. thesis (in French), Université Versailles Saint-Quentin, 2009.

[105] R. Ravaux. Decomposing trees with large diameter. *Theoretical Computer Science*, 411:3068-3072, 2010

[106] T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th Annual ACM Symposium on Theory of Computing*, 216–226, 1978.

[107] M. Schaefer and C. Umans. Completeness in the Polynomial-Time Hierarchy: A compendium. *SIGACT News*, 33(3):32–49, 2002.

[108] W. Schnyder. Embedding planar graphs on the grid. In *1st Annual ACM-SIAM Symposium on Discrete Algorithms*, 138–148, 1990.

[109] B. Seamone. The 1-2-3 Conjecture and related problems: a survey. 2012. Preprint available online at http://arxiv.org/abs/1211.5122.

[110] B. Seamone. *Derived Colourings of Graphs*. Ph.D. thesis, Carleton University, 2012.

[111] B. Seamone. On weight choosability and additive choosability numbers of graphs. 2012. Preprint available online at http://arxiv.org/abs/1210.6944.

[112] J. Skowronek-Kaziów. 1,2 conjecture - the multiplicative version. *Information Processing Letters*, 107(3-4):93–95, 2008.

[113] L.J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1-22, 1976.

[114] L.J. Stockmeyer and A.R. Meyer. Word problems requiring exponential time. In *Proceedings 5th Annual ACM Symposium on Theory of Computing*, 1–9, 1973.

[115] H. Suzuki, N. Takahashi, and T. Nishizeki. A linear algorithm for bipartition of biconnected graphs. *Information Processing Letters*, 33:227–231, 1990.

[116] C. Thomassen. Hypohamiltonian and hypotraceable graphs. *Discrete Mathematics*, 9(1):91–96, 1974.

[117] W.T. Tutte. The Factorization of Linear Graphs. *Journal of the London Mathematical Society*, 22:107–111, 1947.

[118] T. Wang and Q. Yu. On vertex-coloring 13-edge-weighting. *Frontiers of Mathematics in China*, 3(4):581–587, 2008.

[119] D. West. Irregularity strength of graphs and digraphs. 2008. Web page available online at http://math.uiuc.edu/∼west/regs/irreg.html.

[120] C. Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):23-33, 1976.

Index of definitions

List of notation

$<...>$	Problem instance with given inputs, page 20
$\bullet$	Representation of a vertex in a cotree, page 17
$A(D)$	Arc set of D, page 3
$a(G)$	Arboricity of G, page 13
$B(...)$	Balloon with branches of given orders, page 16
$b_i(B)$	Order of the ith branch of B, page 133
$\mathrm{BIN}(n,p)$	Binomial distribution with parameters n and p, page 19
$c_{a,b}(T_r)$	$\{a,b\}$-edge-colouring of T_r verifying $c(rr^+)=a$, page 209
C_n	Cycle with order n, page 15
$Cat(a,b)$	Caterpillar isomorphic to $P_3(1,a-1,b-1)$, page 14
$\mathcal{C}_{k,\ell}(n)$	Set of (k,ℓ)-compound graphs with order n, page 74
$C_{k,\ell}(...)$	Compound graph with k roots and ℓ given components, page 16
$CC_k(x,y)$	Connected-cycles graph with parameters k, x and y, page 140
$ch(G)$	List analogue of $\chi(G)$, page 13
$ch'(G)$	List analogue of $\chi'(G)$, page 13
$ch''(G)$	List analogue of $\chi''(G)$, page 13
$ch'_{term}(G)$	List analogue of $\chi'_{term}(G)$, page 160
$ch''_{term}(G)$	List analogue of $\chi''_{term}(G)$, page 160
$\chi'_{irr}(\overrightarrow{G})$	Irregular chromatic index of $\overrightarrow{G}$ (oriented), page 169
$\chi'_{irr}(G)$	Irregular chromatic index of G (undirected), page 166
$\chi'_{npd}(\overrightarrow{G})$	Neighbour-outproduct-distinguishing chromatic index of $\overrightarrow{G}$ (oriented), page 231
$\chi'_{nsd}(\overrightarrow{G})$	Neighbour-outsum-distinguishing chromatic index of $\overrightarrow{G}$ (oriented), page 168
$\chi'_{nsd}(G)$	Neighbour-sum-distinguishing chromatic index of G (undirected), page 159
$\chi''_{nsd}(\overrightarrow{G})$	Neighbour-sum-distinguishing total chromatic number of $\overrightarrow{G}$ (oriented), page 239
$\chi'_{term}(G)$	Chromatic index related to a distinguishing edge-weighting notion, page 159
$\chi''_{term}(G)$	Total analogue of $\chi'_{term}(G)$, page 160
$\chi(G)$	Chromatic number of G, page 12
$\chi'(G)$	Chromatic index of G, page 12
$\chi''(G)$	Total chromatic number of G, page 12
D_0	Set of possible 1-degrees of a shrub's root's child, page 213
$(D_1,...,D_{d(r^+)-1})$	Signature of a shrub with root r, page 213
$d(v)$	Degree of v, page 4
$d_G(v)$	Degree of v in G, page 4
$d_{c,a}(v)$	Degree of v in the a-subgraph induced by c, page 13
$d^+(v)$	Outdegree of v, page 5
$d^+_D(v)$	Outdegree of v in D, page 5
$d^+_{c,a}(v)$	Outdegree of v in the a-subgraph induced by c, page 13
$d^-(v)$	Indegree of v, page 5
$d^-_D(v)$	Indegree of v in D, page 5
$d^-_{c,a}(v)$	Indegree of v in the a-subgraph induced by c, page 13
$\Delta(G)$	Maximum degree of G, page 4
$\Delta^+(D)$	Maximum outdegree of D, page 5
$\Delta^-(D)$	Maximum indegree of D, page 5
$\delta(G)$	Minimum degree of G, page 4

$\delta^+(D)$	Minimum outdegree of D, page 5
$\delta^-(D)$	Minimum indegree of D, page 5
$\mathrm{dist}(u, v)$	Distance from u to v, page 9
$E(G)$	Edge set of G, page 3
$G \,\square\, H$	Cartesian product of G and H, page 14
$G + H$	Disjoint union of G and H, page 8
$G \times H$	Complete join of G and H, page 13
$G \simeq H$	G and H are isomorphic, page 6
$G + S$	Addition of S to G, page 8
$G - S$	Removal of S from G, page 8
$G[S]$	Subgraph of G induced by S, page 6
$G(n, p)$	Erdős-Rényi random graph model with parameters n and p, page 19
$G_{a,b}$	Grid with a rows and b columns, page 18
G^i	ith layer of G in $G \,\square\, P_\ell$, page 19
G^k	kth power of G, page 14
$H_{k,n}$	k-connected Harary graph with order n, page 15
$i_j(G)$	jth input of G, page 172
$I_j(G)$	jth input of G, page 219
$\kappa(G)$	Connectivity of G, page 9
$K_{\mathcal{C}_{k,\ell}}(n)$	Set of sequences for $\mathcal{C}_{k,\ell}(n)$, page 74
$K_{\mathcal{M}_k}(n)$	Set of sequences for $\mathcal{M}_k(n)$, page 66
$K_\nu(...)$	Graph with ν universal vertices and given components, page 112
$K_{\mathcal{S}}(n)$	Polynomial kernel for $\mathcal{S}(n)$, page 39
$K_{\mathcal{T}}(n)$	Polynomial kernel for $\mathcal{T}(n)$, page 39
$K'_{\mathcal{T}}(n)$	Polynomial kernel for $\mathcal{T}(n)$, page 39
K_n	Complete graph with order n, page 17
$K_{\mathcal{U}_k}(n)$	Set of sequences for $\mathcal{U}_k(n)$, page 70
$m_c(v)$	Multiset of colours incident to v by c, page 161
$m(C_j)$	Number of distinct literals in C_j, page 27
$\mathcal{M}_k(n)$	Set of complete k-partite graphs with order n, page 66
$M_k(...)$	Complete k-partite graph with parts of given orders, page 17
$\mu(G)$	Path cover number of G, page 13
$n(\ell_i)$	Number of distinct clauses which contain ℓ_i, page 27
$N(v)$	Neighbourhood of v, page 4
$N_G(v)$	Neighbourhood of v in G, page 4
$N^+(v)$	Set of outneighbours of v, page 5
$N^+_D(v)$	Set of outneighbours of v in D, page 5
$N^-(v)$	Set of inneighbours of v, page 5
$N^-_D(v)$	Set of inneighbours of v in D, page 5
$\mathcal{O}$	Big-O notation of Landau, page 20
o	Little-o notation of Landau, page 20
$o_j(G)$	jth output of G, page 172
$O_j(G)$	jth output of G, page 219
$\omega(G)$	Clique number of G, page 7
$p_w(v)$	Product of weights incident to v by w, page 164
$p^+_w(v)$	Product of outgoing weights incident to v by w, page 231
$PB(...)$	Partial balloon with (possibly hanging) branches of given orders, page 16
$PCC_k(x)$	Partial connected-cycles graph with parameters k and x, page 141
$\Pi \leq \Pi'$	Π is reducible to Π', page 22
$\Pi \leq_p \Pi'$	Π is reducible to Π' in polynomial time, page 22
$\lvert\pi\rvert$	Size of π, page 35
$\lVert\pi\rVert$	Sum of the elements of π, page 35

$P_k(...)$	k-pode with arms of given orders, page 14
P_n	Path with order n, page 14
$\Pr(A_i)$	Probability of A_i to occur, page 19
Q_n	Hypercube of dimension n, page 18
r_1	Root of a (possibly partial) balloon, page 133
r_2	Root of a (possibly partial) balloon, page 133
$\mathcal{S}(n)$	Set of split graphs with order n, page 38
$s_w(v)$	Sum of weights incident to v by w, page 158
$s_w^+(v)$	Sum of outgoing weights incident to v by w, page 168
S^i	ith layer of S in $G \square P_\ell$, page 19
$\varsigma(G)$	Order of the longest paths of G, page 8
S_n	Star with order n, page 14
$sp(\pi)$	Spectrum of π, page 35
$\mathcal{T}(n)$	Set of tripodes with order n, page 39
$\mathcal{T}$	Family of exceptions for locally irregular edge-colouring, page 194
$T_r[u]$	Subtree of T_r rooted at u, page 11
$T_r[u,i]$	ith subtree of T_r rooted at u, page 11
T_r	Rooted tree with root r and underlying tree T, page 10
u^+	Successor of u, page 96
u^+	Unique child of u, page 10
u^-	Father of u, page 10
u^-	Predecessor of u, page 96
uGv	Subgraph $G[\{u, u^+, (u^+)^+, ..., v^-, v\}]$ of G, page 96
u^i	ith layer of u in $G \square P_\ell$, page 19
u^j	Projection to the jth component of the root u, page 16
$\mathcal{U}_k(n)$	Set of graphs with k universal vertices and order n, page 70
$und(\overrightarrow{G})$	Undirected graph underlying $\overrightarrow{G}$, page 4
uv	Edge between u and v, page 3
$\overrightarrow{uv}$	Arc directed from u to v, page 3
v_i^j	ith vertex of the jth branch of a balloon, page 133
$V(G)$	Vertex set of G, page 3
$v_1v_2...v_k$	Undirected path with consecutive vertices $v_1, v_2, ..., v_k$, page 7
$\overrightarrow{v_1v_2...v_k}$	Directed path with consecutive vertices $v_1, v_2, ..., v_k$, page 8
$\lvert v_1v_2...v_k\rvert$	Order of $v_1v_2...v_k$, page 7
$\lVert v_1v_2...v_k\rVert$	Length of $v_1v_2...v_k$, page 7
$v_1v_2...v_kv_1$	Undirected cycle with consecutive vertices $v_1, v_2, ..., v_k, v_1$, page 7
$\overrightarrow{v_1v_2...v_kv_1}$	Circuit with consecutive vertices $v_1, v_2, ..., v_k, v_1$, page 8
$\lvert v_1v_2...v_kv_1\rvert$	Order of $v_1v_2...v_kv_1$, page 7
$\lVert v_1v_2...v_kv_1\rVert$	Length of $v_1v_2...v_kv_1$, page 7
x^+	Any positive integer greater than x, page 132
x^-	Any positive integer smaller than x, page 132

Partitions et décompositions de graphes

Résumé :

Cette thèse est dédiée à l'étude de deux familles de problèmes de partition de graphe.

Nous considérons tout d'abord le problème de sommet-partitionner un graphe en sous-graphes connexes. Plus précisément, étant donnés p entiers positifs $n_1, n_2, ..., n_p$ dont la somme vaut l'ordre d'un graphe G, peut-on partitionner $V(G)$ en p parts $V_1, V_2, ..., V_p$ de sorte que chaque V_i induise un sous-graphe connexe d'ordre n_i ? Nous nous intéressons ensuite à des questions plus fortes. Que peut-on dire si l'on souhaite que G soit partitionnable de cette manière quels que soient p et $n_1, n_2, ..., n_p$? Si l'on souhaite que des sommets particuliers de G appartiennent à des sous-graphes particuliers de la partition ? Et si l'on souhaite que les sous-graphes induits soient plus que connexes ? Nous considérons toutes ces questions à la fois du point de vue structurel (sous quelles conditions structurelles une partition particulière existe-t-elle nécessairement ?) et algorithmique (est-il difficile de trouver une partition particulière ?).

Nous nous intéressons ensuite à la 1-2-3 Conjecture, qui demande si tout graphe G admet une 3-pondération voisin-somme-distinguante de ses arêtes, i.e. une 3-pondération par laquelle chaque sommet de G peut être distingué de ses voisins en comparant uniquement leur somme de poids incidents. Afin d'étudier la 1-2-3 Conjecture, nous introduisons notamment la notion de coloration localement irrégulière d'arêtes, qui est une coloration d'arêtes dont chaque classe de couleur induit un sous-graphe dans lequel les sommets adjacents sont de degrés différents. L'intérêt principal de cette coloration est que, dans certaines situations, une pondération d'arêtes voisin-somme-distinguante peut être déduite d'une coloration d'arêtes localement irrégulière. Nos préoccupations dans ce contexte sont principalement algorithmiques (est-il facile de trouver une pondération d'arêtes voisin-somme-distinguante ou une coloration d'arêtes localement irrégulière utilisant le plus petit nombre possible de poids ou couleurs ?) et structurelles (quel est le plus petit nombre de couleurs d'une coloration d'arêtes localement irrégulière ?). Nous considérons également ces questions dans le contexte des graphes orientés.

Mots-clefs :

partition en sous-graphes connexes, graphe (récursivement) arbitrairement partitionnable (k-préassignable, à la volée), coloration voisin-distinguante d'arêtes ou d'arcs, coloration localement irrégulière d'arêtes ou d'arcs

Laboratoire Bordelais de Recherche en Informatique (LaBRI)
Université de Bordeaux
351, cours de la Libération
33405 Talence Cedex, France