THÈSE

présentée à

l'Université de Nice – Sophia Antipolis

pour obtenir le titre de

DOCTEUR

mention : Informatique

COMMUNICATIONS PAR COMMUTATION DE CIRCUITS DANS LES RESEAUX D'INTERCONNEXION

par Olivier DELMAS

Soutenue le 16 Janvier 1997 devant la commission composée de :

M.	Ioan BOND	Président
M.	Jean-Claude BERMOND	Directeur
M.	Igor LITOVSKY	Examinateur
M.	Philippe MUSSI	Examinateur
M.	Joseph G. PETERS	Rapporteur
M.	Denis TRYSTRAM	Rapporteur

Je remercie :

Johny Bond de m'avoir fait l'honneur de présider ma thèse, l'ensemble de mes rapporteurs, J.G. Peters, A. Raspaud et D. Trystram pour avoir eu l'obligeance de relire mon travail dans des délais toujours critiques, et aussi mes examinateurs I. Litovsky et P. Mussi.

Un grand merci également à mes amis de travail de tous les jours (Stéphane, Eric, Bruno, Nausica, Takako, Michel, Bruno, Françoise, Claudine, Olivier, Sandrine, et j'en oublie ...), ainsi qu'à tous les membres du thème PACOM et récemment du projet SLOOP, pour leur sympathie et leur disponibilité lorsque souvent j'ai eu besoin d'eux.

Enfin, toute ma reconnaissance à Jean-Claude, sans qui bien sûr cette thèse n'existerait pas. Mais de toute façon "le BOSS" (c'est écrit sur sa tasse de thé) sait à quel point je l'estime.

Table des matières

Table des matières **1**

Introduction **4**

1 Machines parallèles et mécanismes de communication **11**
- 1.1 Modélisation du réseau d'interconnexion d'une machine distribuée 12
  - 1.1.1 Modélisation d'un nœud du réseau 12
- 1.2 Différents modes de commutation 13
  - 1.2.1 Commutation de messages (*store-and-forward*) 14
  - 1.2.2 Commutation de paquets (*packet-switching*) 14
  - 1.2.3 Routage par déflexion (*deflection or hot-patato routing*) 15
  - 1.2.4 Commutation de circuits (*circuit-switching*) 15
  - 1.2.5 Routage *wormhole* 15
  - 1.2.6 *Virtual-cut-through* 16
  - 1.2.7 Autres modes 17
- 1.3 Commutation de messages *versus* commutation de circuits 17
- 1.4 Graphes et réseaux d'interconnexion 19
  - 1.4.1 Constructions classiques 20
  - 1.4.2 Réseaux classiques 20

2 Introduction aux communications globales en mode commutation de circuits **27**
- 2.1 Communications globales 29
- 2.2 Le modèle de type commutation de circuits 29
- 2.3 Modélisation des problèmes 31
  - 2.3.1 Temps de communication d'un processeur à un autre 31
  - 2.3.2 Coût des protocoles de communications globales 33
  - 2.3.3 Choix d'une fonction de routage 33
  - 2.3.4 Contraintes dues au réseau 35
- 2.4 Notation complète 37
- 2.5 Taxinomie des problèmes de communications globales 37

3 La diffusion en mode commutation de circuits **41**
- 3.1 Généralités 41
  - 3.1.1 Equivalence des modes téléphoniques et télégraphiques pour le nombre d'étapes 41
  - 3.1.2 Compromis entre le nombre d'étapes et le flot d'information 42
- 3.2 La diffusion sans fonction de routage imposée dans le modèle $1-$port 54
  - 3.2.1 Bornes inférieures générales dans le modèle $1-$port 54
  - 3.2.2 La chaîne dans le modèle F_1 56
  - 3.2.3 Le cycle dans le modèle F_1 58
  - 3.2.4 Le tore de dimension k dans le modèle F_1 58

3.2.5 La grille de dimension k dans le mode F_1 . . . 59
3.3 La diffusion sans fonction de routage imposée dans le modèle k et $\Delta-$ports . . 63
3.3.1 Bornes inférieure générales dans le modèle $k-$ports . . . 63
3.3.2 Diffusion dans le modèle $\Delta-$ports - Méthodologie générale . . . 64
3.3.3 Le cycle dans le modèle H_* . . . 65
3.3.4 Les graphes hamiltoniens dans le modèle H_* . . . 65
3.3.5 Le tore de dimension 2 dans le modèle H_* et F_* . . . 65
3.3.6 Le tore de dimension k dans le modèle H_* . . . 71
3.3.7 La grille de dimension k dans le mode H_* . . . 74
3.3.8 L'hypercube dans le modèle F_* . . . 75
3.3.9 Le Butterfly dans le modèle F_* . . . 79
3.4 La diffusion avec fonction de routage imposée dans le modèle $1-$port . . . 88
3.4.1 Le tore de dimension 2 dans le modèle F_1 . . . 88
3.4.2 La grille de dimension 2 dans le modèle F_1 . . . 88
3.4.3 L'hypercube dans le modèle F_1 . . . 89
3.5 La diffusion avec fonction de routage imposée dans le modèle $\Delta-$ports . . . 90
3.5.1 Le tore de dimension 2 et 3 dans le modèle F_* . . . 90
3.5.2 La grille de dimension 2 et 3 dans le modèle F_* . . . 91
3.5.3 L'hypercube dans le modèle F_* . . . 92

4 L'échange total et la multidistribution en mode commutation de circuits 99
4.1 L'échange total $-$ Hypothèses supplémentaires . . . 99
4.2 L'échange total sans fonction de routage imposée . . . 100
4.2.1 Bornes inférieures générales . . . 100
4.2.2 Le cycle dans le modèle F_* . . . 101
4.2.3 Le tore de dimension k dans le modèle F_* . . . 102
4.2.4 L'hypercube dans le modèle F_* . . . 107
4.3 L'échange total avec fonction de routage imposée . . . 108
4.3.1 L'hypercube dans le modèle F_* . . . 108
4.3.2 L'hypercube dans le modèle F_1 . . . 108
4.4 La multidistribution $-$ Notations et hypothèses supplémentaires . . . 109
4.5 La multidistribution sans fonction de routage imposée . . . 109
4.5.1 Bornes inférieures générales . . . 109
4.5.2 Compromis entre le nombre d'étapes et le flot d'information . . . 110
4.5.3 La chaîne dans le modèle F_1 . . . 111
4.5.4 La grille de dimension 2 dans le modèle F_1 . . . 112
4.5.5 Le tore de dimension 2 dans le modèle F_1 . . . 116
4.5.6 Le tore de dimension 3 dans le modèle F_1 . . . 118
4.6 La multidistribution avec fonction de routage imposée . . . 119
4.6.1 La grille de dimension 2 dans le modèle F_1 . . . 119
4.6.2 L'hypercube dans le modèle F_1 . . . 119
4.6.3 L'hypercube dans le modèle F_* . . . 121
4.7 Conclusion des chapitres 2, 3 et 4 . . . 122

5 Propriétés et construction de graphes 127
5.1 Décomposition hamiltonienne du réseau Butterfly généralisé . . . 128
5.1.1 Motivation . . . 128
5.1.2 Etat de l'art . . . 129
5.1.3 Décomposition hamiltonienne du graphe Butterfly . . . 131
5.2 Problème (Δ, D) . . . 137
5.2.1 Notre approche du problème . . . 138

Conclusion et perspectives 146

A Diffusion en mode commutation de circuits dans les tores de dimension k 149

B Circuit-switched gossiping in the 3−dimensional torus networks 169

C Hamilton circuits in the directed Butterfly network 193

D Hamilton cycle decomposition of the Butterfly network 221

Introduction

Le développement de l'informatique et de ses applications a entraîné une demande sans cesse croissante en puissance de calcul. Pour répondre à ce besoin les machines parallèles ont reçu un intérêt tout particulier. De ce fait, de nombreux domaines de recherche nouveaux ont émergé ; par exemple, les langages parallèles, la parallélisation automatique, les architectures distribuées, les environnements de programmation, l'analyse de complexité parallèle et l'algorithmique parallèle.

Actuellement, la plupart des machines existantes reposent sur des architectures à mémoire distribuée. Une des différences essentielles entre les machines traditionnelles, dites de « Von Neumann », et les réseaux de processeurs que sont les machines à mémoire distribuée, est la gestion des communications. Si on cherche à distribuer le plus possible les traitements élémentaires, on obtient le parallélisme potentiellement le plus efficace, mais les communications induites peuvent alors être pénalisantes. Optimiser les communications est donc une des clefs de l'efficacité des algorithmes parallèles.

Si on peut souhaiter une modélisation complète des calculateurs parallèles qui prendrait en compte à la fois le réseau d'interconnexion, les calculs parallèles et les communications associées, la difficulté d'une telle étude apparaît bien vite et seuls quelques résultats particuliers concernant le recouvrement des communications et des calculs existent (dans ce cadre les seuls résultats existants sont en général associés à des instances particulières de machine). Notons au passage que le nombre de modèles « globaux » est immédiatement déraisonnablement élevé puisqu'un modèle complet doit prendre en compte la vitesse relative des instructions de calcul et de communication, leur parallélisme ou non, etc. Ainsi, à l'heure actuelle, il semble raisonnable d'étudier séparément les calculs et les communications, pour ensuite, dans le cas d'une application réelle, tenter de prendre en compte les spécificités de l'architecture utilisée en espérant qu'un bon algorithme muni de bonnes routines de communication fournisse un résultat efficace.

Les résultats obtenus dans cette thèse, qui s'est déroulée au sein du projet SLOOP[1] [1], portent donc principalement sur **l'étude des communications**

1. SLOOP (Simulation, langages orientés objets et parallélisme) est un projet commun entre le CNRS, l'INRIA et L'UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS.

dans les architectures parallèles, distribuées ou réseaux d'interconnexion.

Dans le **chapitre 1**, nous présentons brièvement une rapide classification des machines parallèles. Puis nous décrivons en détail les principaux mécanismes de routage des messages existant à l'heure actuelle dans les réseaux des machines à mémoire distribuée. Nous montrons que de plus en plus dans ce type de machine, les constructeurs tendent à remplacer le routage par commutation de messages utilisé depuis de nombreuses années par un routage du type « *wormhole* ». Ce dernier apparaît posséder de nouvelles qualités importantes, notamment celui d'être beaucoup moins sensible à la distance entre les nœuds communiquant entre eux.

Pour étudier les réseaux d'interconnexion et surtout pouvoir comparer leurs performances mutuelles ou l'efficacité des divers protocoles de communications, il est nécessaire de travailler sur un modèle commun. Il existe de nombreux outils permettant de modéliser tel ou tel aspect d'un réseau d'interconnexion. Cependant, l'un des outils privilégiés pour l'étude des communications et des propriétés inhérentes des réseaux est la théorie des graphes. Le **section 1.4** contient un bref rappel des principales notions de théorie des graphes nécessaires à une telle modélisation ainsi que les définitions des réseaux utilisés dans la suite de cette thèse.

Les problèmes de routage des communications intervenant au cours d'applications parallèles, au sens large du terme (calculs scientifiques, systèmes d'exploitation, etc) peuvent être regroupés en deux grandes classes : les routages *"temps-réels"* et les routages *"précalculés"*.

Le terme *"temps-réel"* est utilisé lorsque, et c'est le cas dans de nombreuses applications, le problème de routage n'est pas connu à l'avance (i.e. non connu avant l'exécution du programme). Cependant, quelquefois nous pouvons connaître par avance des schémas de communications qui interviendront à coup sûr dans l'application. Dans ce cas, il est possible de calculer, avant l'exécution, une solution au routage de tels schémas. Nous utiliserons pour cela le terme *"précalculés"*. Résoudre, avant l'exécution, de tels problèmes peut être particulièrement intéressant dans le cas où ils sont susceptibles d'intervenir un grand nombre de fois au cours de l'exécution d'une même application.

Ainsi, au cours d'une application, les processeurs communiquent en échangeant des messages, mais la répartition des communications sur le réseau dépend fortement de l'algorithme utilisé. Heureusement, l'étude de paradigmes classiques montre qu'il existe dans la classe des routages *"précalculés"* un certain nombre de "communications structurées (globales, généralisées, ou collectives)" qui apparaissent très souvent dans de nombreux problèmes de calculs parallèles ou distribués, comme par exemple en algèbre linéaire ou non-linéaire, en traitement d'images, ou bien encore dans les systèmes de bases de données.

Dans le **chapitre 2** nous commençons par redéfinir brièvement la notion de communications globales. Dans la **section 2.2** nous construisons un modèle général appelé «modèle de type commutation de circuits» synchrone, c'est-à-dire dans lequel les communications globales s'effectuent comme une succession d'étapes. Nous verrons dans la suite que c'est ce modèle qui est en fait le plus abondamment utilisé dans la littérature traitant de ce sujet. Nous montrons que pour le problème particulier des communications globales, nous pouvons regrouper sous notre modèle général la plupart des modes de commutation les plus utilisés actuellement que sont le *wormhole*, le *virtual-cut-through*, le *direct-connect* ou le *mode commutation de circuits*. Cette approche nous permettra dans la suite d'effectuer une synthèse des divers travaux effectués sur le sujet, et aussi de pouvoir les comparer entre eux. Cette synthèse devrait également servir de base à un article [4] futur.

Dans la **section 2.3** nous définissons, dans notre modèle général, le temps de communication d'un processeur à un autre à partir duquel nous définissons le coût total d'un protocole de communications globales qui dépend de trois paramètres : le nombre d'étapes, les distances de communication et le flot d'information. De plus, comme les machines parallèles existantes implémentent des technologies parfois très différentes (composants, logiciels, etc) il apparaît que leur modélisation unique est impossible. Ainsi, il est nécessaire de prendre en compte dans nos modèles les principales contraintes qui portent sur la nature des liens de communications (télégraphique ou téléphonique), les communications (chemins disjoints) et les émetteurs/récepteurs (1, k ou $\Delta-$ports).

Le **chapitre 3** dresse une synthèse des travaux qui nous paraissent les plus significatifs sur le problème de la diffusion par commutation de circuits, tout du moins lorsque l'on cherche essentiellement à minimiser le nombre d'étapes des protocoles. Une partie de nos travaux s'insère dans ce chapitre. Dans la **section 3.1.2** nous prouvons qu'il est contradictoire de vouloir optimiser plusieurs paramètres simultanément, notamment le nombre d'étapes et le flot d'information. Pour cela nous donnons quelques premières bornes inférieures exprimant la quantité minimum du flot d'information en fonction du nombre d'étapes. Le but est de pouvoir comparer entre eux les divers algorithmes du type pipeline.

Nous étudions ensuite successivement le modèle $1-$port (**section 3.2 et 3.4**), puis le modèle $\Delta-$ports (**section 3.3 et 3.5**). L'expérience acquise dans ce dernier nous permet de dégager une méthodologie générale que nous synthétisons en **section 3.3.2**.

Dans la **section 3.3.6-ii** nous résumons nos travaux sur le tore de dimension k. Nous y construisons notamment un algorithme optimal en nombre d'étapes et donnons une estimation fine de la longueur des chemins. Ce travail qui est repris en détail dans l'**annexe A** page 149, a fait l'objet des articles [7, 8]. Nous montrons que notre technique s'appuie essentiellement sur deux méthodes combinant des outils d'algèbre linéaire et de théorie des codes.

En **section 3.3.9** nous détaillons notre approche pour effectuer une diffusion sur le graphe orienté *Butterfly* $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2, n)$. Nous proposons un algorithme récursif efficace bien que non optimal pour le nombre d'étapes. Nous évoquons également la méthode à utiliser pour effectuer la même opération sur la version non orientée de ce graphe.

Le **chapitre 4** traite des problèmes d'échange total et de multidistribution. Dans la **section 4.2.1** nous donnons une borne inférieure générale non triviale sur le nombre d'étapes de l'échange total dans un graphe G. La preuve entière qui se trouve en **annexe B** page 169 (Cf. [5]) est basée sur une énumération précise de la charge des chemins pouvant être utilisés au cours d'un algorithme. Cette notion est à rapprocher de l'indice arc de transmission.

Nous résumons en **section 4.2.3** notre protocole optimal en nombre d'étapes pour l'échange total sur des tores de dimension 3 (Cf. **annexe B** page 169 et articles [6, 5]). Nous montrons en outre que les longueurs des chemins utilisés sont proches de l'optimal. Ici encore, la méthode utilisée repose sur des notions de théorie des codes.

Les propriétés nécessaires au bon fonctionnement d'un réseau dépendent bien entendu de l'utilisation de ce dernier, ce qui implique de trouver des algorithmes de communications efficaces, mais aussi d'étudier ses propriétés topologiques. Plusieurs paramètres et propriétés, considérés comme importants et en tout cas révélateurs des qualités d'un réseau sont étudiés dans la littérature. Dans le **chapitre 5** nous résumons nos contributions sur deux de ces propriétés.

Tout d'abord en **section 5.1** nous considérons les décompositions hamiltoniennes de réseaux. En particulier nous résumons en **section 5.1.3** les résultats obtenus dans le *Butterfly* généralisé. Ceci fait l'objet des articles en **annexes C** page 193 et **annexe D** page 221. Plus précisément, nous prouvons [2] que le réseau Butterfly généralisé orienté de degré d admet $(d-1)$ circuits Hamiltoniens arc-disjoints (ceci répond à une conjecture de D. Barth). Nous conjecturons que pour $n \geq 2$ et hormis trois exceptions, le réseau se décompose en circuits Hamiltoniens. Nous prouvons qu'il suffit de vérifier la conjecture pour $n = 2$ et un degré premier. Par une recherche exhaustive très contrainte, effectuée sur ordinateur, nous avons vérifié la propriété pour les nombres premiers inférieurs à 12000. Les démonstrations utilisent la structure récursive du réseau *Butterfly*. Nous montrons aussi [3] que le réseau *Butterfly* généralisé dans sa version non orienté, se décompose en cycles Hamiltoniens (ceci répond à une conjecture de D. Barth et A. Raspaud).

Enfin, en **section 5.2**, nous introduisons le désormais classique problème des larges $(\Delta, D)-$graphes. Ce problème a été posé en 1964 par Elspas sous la forme suivante : « quel est le nombre maximum de sommets d'un (Δ, D)-graphe, un (Δ, D)-graphe étant un graphe de degré maximum Δ et de diamètre D ? ». On note $N(\Delta, D)$ le nombre maximum de sommets d'un (Δ, D)-graphe. Depuis fort longtemps maintenant, de nombreux auteurs n'ont cessé d'essayer de construire les plus grands graphes possible. Une table résumant les résultats est maintenue à jour par l'équipe de graphes d'Espagne de l'*Universitat Politècnica de Catalunya* et est accessible via le réseau Internet [2].

Généralement ce type de problème demande de posséder de fortes puissances de calcul, car la recherche de tels graphes est souvent astreinte à des vérifications ou des recherches sur ordinateur. Pour notre part nous avons abordé le problème, et malgré des moyens de calcul relativement limité (3 ou 4 Sun Sparc Station 10, ou 16 processeurs $i860$), nous donnons en **section 5.2.1** le résultat principal que nous avons obtenu, à savoir améliorer la valeur $N(4,10)$ qui est maintenant de 17525 sommets, au lieu de 16555 qui était la précédente valeur. Du fait du peu de puissance de calcul à notre disposition, nous n'avons pas pu étendre la recherche au delà de $\Delta = 5$ et $D = 8$. Cependant il est à signaler que nous avons retrouvé des graphes aussi bons que certains existant et non isomorphes.

Enfin, nous concluons en présentant les perspectives de nouveaux axes de recherches sur les communications et routages qu'il semble d'ores et déjà intéressant d'approfondir.

2. Par l'URL : `http://www_mat.upc.es/grup_de_grafs/`

Bibliographie

[1] Françoise Baude, Fabrice Belloncle, Jean-Claude Bermond, Denis Caromel, Olivier Dalle, Eric Darrot, Olivier Delmas, Nathalie Furmento, Bruno Gaujal, Philippe Mussi, Stéphane Perennes, Yves Roudier, and Günther Siegel. The *SLOOP* project: Simulations, Parallel Object-Oriented Languages, Interconnection Networks. In *2nd European School of Computer Science, Parallel Programming Environments for High Performance Computing ESP-PE'96*, pages 85–88, Alpe d'Huez, April 1996.

[2] J-C. Bermond, E. Darrot, O. Delmas, and S. Perennes. Hamilton circuits in the directed Butterfly network. Research Report 2925 — Thème 1, INRIA Sophia Antipolis, July 1996. A paraître dans *Discrete Applied Mathematics*.

[3] J-C. Bermond, E. Darrot, O. Delmas, and S. Perennes. Hamilton cycle decomposition of the Butterfly network. Research Report 2920 — Thème 1, INRIA Sophia Antipolis, June 1996. A paraître dans *Parallel Processing Letters*.

[4] Jean de Casanice. An overview of wormhole routing. Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis CNRS URA 1376 - Equipe : *"Communication Algorithms at Sophia Antipolis - Nice"* (J-C. Bermond, E. Darrot, O. Delmas, E. Fleury, S. Perennes, M. Syska), En préparation.

[5] O. Delmas and S. Perennes. Circuit-Switched Gossiping in 3-Dimensional Torus Networks. Rapport de Recherche 2930, Institut National de Recherche en Informatique et en Automatique, Unité de recherche INRIA Sophia Antipolis - France, July 1996. (Version étendue de [6]).

[6] O. Delmas and S. Perennes. Circuit-switched gossiping in 3-dimensional torus networks. In L. Bougé, P. Fraigniaud, A. Mignotte, and Y. Robert, editors, *Proceedings of the Euro-Par'96 Parallel Processing / Second International EURO-PAR Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 370–373, Lyon, France, August 1996. Springer Verlag.

[7] O. Delmas and S. Perennes. Diffusion en mode commutation de circuits. In R. Castanet and J. Roman, editors, *Proceedings of the 8th RenPar Conference*, pages 53–56, Bordeaux, France, May 1996.

[8] O. Delmas and S. Perennes. Diffusion en mode commutation de circuits dans les tores de dimension k. Laboratoire I3S - CNRS URA 1376 - Accepté avec révisions à la revue Technique et Science Informatiques. Hermès, AFCET, Paris., 1997.

Chapitre 1

Machines parallèles et mécanismes de communication

✔ *Après une brève classification des machines parallèles, ce chapitre décrit en détail les principaux mécanismes de routage des messages existant à l'heure actuelle dans les réseaux de machines à mémoire distribuée. Puis, nous rappellons quelques définitions de graphes modélisant les réseaux d'interconnexion qui seront les plus fréquemment utilisés dans cette thèse.*

A l'heure actuelle l'informatique apparaît être un outil de plus en plus indispensables dans presque tous les secteurs de la société. L'évolution technologique et les contraintes économiques ont entraîné une demande et un besoin sans cesse croissant en puissance de calcul susceptible d'être fourni par les ordinateurs. Pour répondre en puissance à ces besoins il a fallu fabriquer des machines de plus en plus performantes.

Historiquement, les premiers ordinateurs étaient mono processeur (i.e. ils ne disposait en tout et pour tout que d'une seule unité de calcul, et toutes les tâches devaient s'effectuer de façon purement séquentielle). Plus tard, est apparu une nouvelle génération de machines dites « machine vectorielles ».

Mais, désormais, la meilleure réponse à ces besoins semble venir des « machines et de l'algorithmique parallèles ». Ces machines parallèles ont elles mêmes constamment évolué dans le temps, et il semble de plus en plus difficile de faire une quelconque comparaison de l'efficacité des solutions proposées. En effet, les différents choix technologiques ont conduit à une grande diversité des machines parallèles [1, 2, 3, 5, 9, 10, 20]. Flynn a introduit une classification [7] qui même si elle semble aujourd'hui un peu obsolète, fournit tout de même une classification cohérente, même s'il est parfois difficile d'y classer de façon précise certaines familles d'architectures (par exemple, les machines systoliques) ou de distinguer les machines à mémoire distribuée des machines à mémoire partagée. Dans cette classification on trouve notamment les machines du type SIMD (*Single Instruction, Multiple Data flow*) ou bien du type MIMD (*Multiple Instructions, Multiple Data flow*).

Désormais dans le cadre d'applications nécessitant de fortes puissances de calcul, les machines SIMD tendent à être remplacées par des machines MIMD. Pour s'en convaincre, le lecteur pourra se référer à la page Web :

`http://www.crpc.rice.edu/CRPC/newsletters/oct93/news.top500.html`

Celle-ci est maintenue par Meuer, Strohmaier et Dongarra et fait état des 500 sites possédant les plus fortes puissances de calcul. Bien entendu, cette page n'a pas l'intention d'être exhaustive, mais représente tout de même une vue statistique des systèmes informatiques les plus puissants. Cette liste montre clairement que la quasi-totalité des machines puissantes sont du type MIMD. De plus, au moins la moitié de cette liste est constitué de machine parallèles à mémoire distribuée utilisant un réseau d'interconnexion point-à-point.

Ces machine parallèles à mémoire distribuée correspondent à celles que nous abordons dans cette thèse. La plupart de ces machines utilisent un routage du type « mode commutation de circuits » (que nous décrivons plus en détails dans la suite). Ce routage est celui que nous utilisons dans les chapitres suivant.

Sur ce type de machine, l'échange d'informations (messages) entre processeurs est essentielle pour mettre au point des applications parallèles ou distribuées efficaces. *Cette thèse porte donc essentiellement sur des problèmes de communication utilisant un routage de type « mode commutation de circuits » sur les machines parallèles à mémoire distribuée utilisant un réseau d'interconnexion point-à-point.*

1.1 Modélisation du réseau d'interconnexion d'une machine distribuée

1.1.1 Modélisation d'un nœud du réseau

Une machine parallèle à mémoire distribuée se présente comme un réseau d'interconnexion point-à-point. Chaque point ou nœud est généralement matériellement une entité complexe composée de différents modules (processeur(s), mémoire(s) externes et/ou composants de gestion et routage ou routeur des communications). La figure 1.1 donne une schématisation ou modèle d'un nœud d'un réseau.

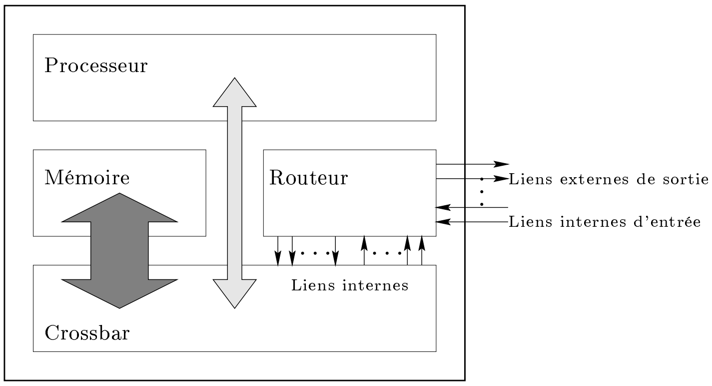

FIG. 1.1 – : *Schématisation d'un nœud.*

Dans une telle schématisation, nous considérons que le processeur (généralement de calculs) échange des messages avec le reste du réseau par l'intermédiaire d'un commutateur ou routeur. Dans ce cas, certains liens alimentent le routeur depuis le processeur, ou utilisent des messages en provenance du réseau. Ces liens sont les canaux internes du nœud. D'autres au contraire permettent le transit de messages par le routeur, ce sont les canaux externes. Comme nous le verrons plus tard, des contraintes sur les capacités d'émission ou de réception de messages seront liées au nombre et à l'usage concurrent ou non de ces divers canaux internes et externes.

Le rôle d'un routeur est d'acheminer les messages entre nœuds non voisins. Les routeurs réalisent de façon distribuée un algorithme de routage qui spécifie le chemin à suivre dans le réseau pour se rendre de tout nœud x vers tout nœud y. Cet algorithme de routage est décrit par une *fonction de routage [8].*

1.2 Différents modes de commutation

Dans un réseaux d'interconnexion point-à-point, la transmission d'un message entre processeurs non directement voisins, doit être routé via des nœuds intermédiaires. Comme schématisé sur la figure 1.1, nous considérons ici que le routage du message se fait par l'intermédiaire de routeurs implanté dans chaque nœud. La commutation d'un routeur est le phénomène consistant en la réception d'une adresse de destination, puis au décodage de cette adresse pour déterminer le canal de sortie convenable et à l'envoi de cette adresse sur ce canal. Mais il existe différents mécanismes physiques ou technologiques de commutation utilisés par les routeurs. Ces différences de mécanismes sont en partie dues au progrès et à la maîtrise technologiques acquises par les différents constructeurs au cours du temps. Néanmoins, comme nous le verrons dans la suite de cette thèse, tous les types de commutations ont dans certains cas des avantages et dans d'autres des inconvénients.

On trouve une présentation des diverses techniques de commutation classiques dans [12, 18, 21]. Pour que le lecteur non familier avec le sujet puisse comparer ces différents modes de commutation nous redonnons les différents modes classiques tels que présentés en partie dans le livre *"Rumeur"* [18].

1.2.1 Commutation de messages (*store-and-forward*)

Il s'agit certainement du modèle le plus simple et le plus ancien (modélisant la quasi totalité des machines parallèles jusqu'au début des années 90). Dans ce cas, les communications s'effectuent entre voisins. Les messages avancent dans le réseau vers leur destination en transitant dans les nœuds intermédiaires. Ainsi, le message est `stocké avant d'être reémis`[1] par un nœud intermédiaire. A chaque étape, le canal emprunté est aussitôt libéré. Un défaut de cette technique est de nécessiter une taille de registre importante pour stocker le message sur les processeurs intermédiaires. En fait, ces messages sont généralement stockés en mémoire. Mais les temps d'accès à la mémoire, proportionnels à la taille des messages, ralentissent alors fortement les communications.

EXEMPLES. La Computing Surface de Meiko, le T-node et le Mega-node de Telmat, les Volvox d'Archipel, ainsi que les autres machines à base de *Transputers* utilisent un routage par commutation de messages.

1.2.2 Commutation de paquets (*packet-switching*)

Ce mode dérive du précédent. En effet, cette fois-ci les messages sont découpés en paquets de taille fixe, on bénéficie alors d'un effet *pipeline*. Les paquets sont routés indépendamment les uns des autres, car chacun a en en-tête l'adresse de destination ; ils peuvent même emprunter des chemins différents. Notons que ceci implique, par rapport à la commutation de messages, un surplus de communications engendré par le supplément d'information ajouté à chaque paquet. De plus, la recomposition du message à partir des différents paquets demande également le transport d'un supplément d'information sur chaque paquet. Par contre, l'avantage de ce mode de commutation est de pouvoir utiliser des registres de petite taille, locaux au routeur. La capacité de stockage nécessaire sur un nœud pour un canal se limite à la taille d'un paquet.

1. D'où le terme anglais consacré dans le cadre des machines distribuées : *"Store-and-Forward"*.

1.2.3 Routage par déflexion (*deflection or hot-patato routing*)

Le routage par déflexion est une autre technique permettant de diminuer le surcoût induit par les recopies d'un message dans la mémoire d'un nœud intermédiaire. Ceci est dû au fait que cette technique élimine les files d'attente des messages sur les nœuds intermédiaires. Pour ce faire, un message essaye toujours de transiter vers sa destination, mais cependant si deux messages (ou plus) arrivent via deux liens (ou plus) d'entrée différents sur un nœud intermédiaire et qu'ils cherchent à en sortir sur un seul et même lien, alors le nœud intermédiaire route l'un des deux messages sur ce lien de sortie tandis que l'autre message est routé sur un autre lien de sortie différent. En particulier, avec un tel mécanisme, des messages peuvent temporairement être amenés à transiter dans le réseau en s'éloignant de leur destination. L'analyse du routage par déflexion est considérablement plus difficile que dans le cas de la commutation de messages ou paquets. Il n'y a a priori pas de borne sur le nombre de liens qu'un message pourrait être amené à utiliser avant d'atteindre sa destination. En effet, un message peut être renvoyé vers une région inopinée du réseau, interférant ainsi avec d'autres messages transitant dans cette même région. Nonobstant, ce mécanisme semble bien fonctionner dans la pratique, puisque quelques machines parallèles, comme la “Tera Computer” ou la “HEP mutiprocessor” l'implantent.

1.2.4 Commutation de circuits (*circuit-switching*)

C'est le principe du téléphone. On établit d'abord la liaison (ici, cela consiste à réserver une suite de canaux), la conversation commence ensuite. D'autres communications qui voudraient emprunter une partie d'un circuit déjà établi sont bloquées jusqu'à la libération de la totalité de la liaison.

Exemple. Le *direct-connect*, qui est proposé par Intel sur les hypercubes de la série iPSC/2 et iPSC/860, est un routage par commutation de circuits.

1.2.5 Routage *wormhole*

Sur les machines à mémoire distribuée les plus récentes, le mode de routage par commutation de messages est abandonné au profit du mode de routage *wormhole*. Au contraire du mode commutation de messages, où les messages (ou paquets) sont entièrement stockés dans la mémoire du nœud intermédiaire avant d'être transmis au nœud suivant, dans le mode de routage *wormhole*, les messages progressent dans le réseau de processeurs *flit* par *flit*[2], le premier *flit* contenant l'adresse de destination. La tête du message, c'est-à-dire le premier *flit*, avance d'un canal, chaque fois que cela est possible. Le reste du message suit, libérant la

2. Un *flit* — pour *flow control digit* — est égal à la taille de la queue d'un canal.

queue du dernier canal, qui stocke la fin du message. Ce dernier canal est alors disponible pour un nouveau message.

Remarque. Il convient de bien distinguer ce mode de routage d'un routage par commutation de paquets. En effet, dans ce dernier, chaque paquet contient en en-tête l'adresse de destination et peut donc être routé indépendamment les uns des autres, pas forcément le long d'un même chemin. Au contraire, dans le cas du *wormhole*, seul le premier *flit* contient l'adresse de destination, les autres suivants à la queue leu leu le premier, ils empruntent tous le même chemin.

Dans le mode de routage *wormhole*, les étapes intermédiaires entre la source et la destination consistent en l'établissement d'un *circuit virtuel*. On peut comparer l'avancée d'un message dans ce mode à celle d'un ver qui progresse sous la terre. Un message peut commencer à être reçu avant que l'émission ne soit terminée. De la même façon, si le message est suffisamment court, la source est libérée avant la réception du premier *flit* par le destinataire. Notons qu'une fois que le *flit* de tête a été affecté à un canal, ce canal ne peut transmettre aucun *flit* d'un autre message, tant que le message originel n'est pas passé (on ne peut pas « couper le ver »). Sur un nœud, seul un *flit* est stocké dans la queue du canal : il n'y a pas d'accès à la mémoire (pas de stockage intermédiaire du message, coûteux en temps [19]). De plus, des *flits* de petite taille permettent d'utiliser des queues de taille réduite, ce qui facilite l'implantation en VLSI du routeur. Si l'en-tête est bloqué, c'est-à-dire si le ou les canaux de sortie sont utilisés par d'autres messages, la propagation du message est stoppée et les *flits* restent stockés dans les queues des canaux qu'ils occupent.

Remarque. Une erreur fréquente est de confondre le routage *wormhole* avec le routage par commutation de circuits. Dans ce dernier, le fonctionnement est analogue à celui du téléphone : établissement d'un circuit physique entre les correspondants, discussion, puis destruction du circuit. Dans le cas du *wormhole*, il n'y a pas construction de circuit au sens propre : l'en-tête (le premier *flit*) établit des connexions physiques sur chacun des routeurs intermédiaires, au fur et à mesure de sa progression. Les *flits* intermédiaires suivent, et le dernier *flit* détruit les connexions. Le circuit n'est que virtuel et n'existe qu'entre le premier et le dernier *flit*.

1.2.6 *Virtual-cut-through*

Une technique semblable au *wormhole*, le *virtual-cut-through*, a été étudiée par Kermani et Kleinrock [12]. Le mode de routage *virtual-cut-through* est identique en tout point au *wormhole*, sauf lorsque la propagation du premier *flit* d'un message est impossible, le ou les canaux de sortie étant déjà tous utilisés pour propager d'autres messages. Rappelons que dans le cas du *wormhole*, les *flits* restent stockés localement dans les queues des canaux qu'ils occupent, c'est-à-dire «sur le chemin» . Au contraire, dans le cas du routage *virtual-cut-through*, les *flits* contenant le corps du message continuent à avancer et sont tous stockés sur le nœud où le premier *flit* s'est trouvé bloqué. Le routeur doit donc disposer de queues de tailles arbitrairement grandes, ce qui ne permet pas de l'intégrer facilement sur un nœud. Cette technique n'a donc pas été implantée à notre connaissance, alors que le *wormhole* fait l'objet d'études et, comme nous l'avons déjà dit, de réalisations réelles.

1.2.7 Autres modes

Des recherches plus récentes tendent à montrer les avantages d'une mise en œuvre matérielle de nouveaux mécanismes de routage [16]. Cette prise en charge matérielle passe par l'ajout de diverses fonctionnalités au sein des routeurs. En particulier, nous évoquons ici deux nouveaux modes de routage qui nous semblent intéressants et qui commencent à être étudiés, même s'ils n'ont pas encore été implantés dans les machines :

- Un mode *diffusion*, qui permettrait au routeur de transmettre sur tous ses canaux en sortie un *flit* accepté en entrée. On routerait alors sur des arbres au lieu de router sur des chemins. Cela faciliterait les procédures apparentées à la diffusion d'un message d'un nœud vers tous les autres.

- Un mode *transparence* ou *capacité de réception intermédiaire* [6, 11, 14, 15], dans lequel le processeur de calcul pourrait lire les *flits* qui transitent par le routeur auquel il est rattaché. Ici aussi, cela faciliterait les procédures de diffusion. En effet, en mode *wormhole* par exemple, lorsqu'un message est émis par un nœud x à destination de z et que ce message transite par un nœud y lors du routage, le message n'a pas été stocké dans la mémoire de y, mais seulement dans les registres du routeur. Ainsi, y n'a pas eu connaissance du message.

1.3 Commutation de messages *versus* commutation de circuits

Il semble qu'à l'heure actuelle, le marché des machines parallèles distribuées s'orientent de plus en plus vers un mécanisme de routage du type *wormhole* au détriment de la commutation de messages. Les deux dernières générations de Cray (T3D et T3E), l'IBM SP2 ou la Paragon d'Intel en sont de bons exemples.

Ce changement de situation est en partie dû au fait que la technologie *wormhole* (et ses variantes) est désormais bien maîtrisée par les grand constructeurs de machines, tels que Cray et Silicon Graphics, Intel ou IBM. Une autre raison tient au fait que les performances d'un mécanisme de communication dépendent de la topologie du réseau d'interconnexion utilisé. En effet, très souvent les constructeurs sont astreints à divers contraintes de fabrication (niveau d'intégration VSLI des composants, problèmes de connectivité entre un grand nombre de composants sur une même ou plusieurs cartes, etc), et les réseaux réels proposés ne sont pas ceux susceptibles, du moins théoriquement, de fournir le plus de puissance. Notamment, la distance (en terme de nombre de nœuds intermédiaires) qu'il peut y avoir entre certains couples de nœuds dans les réseaux existant peut souvent être importante. Alors s'il intervient, au cours de l'exécution d'une application parallèle réelle, un grand nombre de communications entre ces nœuds, les performances peuvent en être affectées. C'est pourquoi, plutôt que de mettre au point de nouvelles topologies de réseaux, qui aurait très certainement demandé l'élaboration d'un nouveau «savoir-faire» et donc une augmentation importante des coûts en recherche des entreprises, les constructeurs ont préféré la solution d'implanter un mécanisme de communication moins sensible aux principaux défauts structurels de leurs réseaux. Ainsi, le choix pratique ce tourne de plus en plus vers un routage de type *wormhole* qui est, comme le montre l'encadré 1.3.1, bien moins sensibles aux distances entre les nœuds que ne l'est la commutation de messages.

Encadré 1.3.1 — Comparaison de différentes techniques de commutation

---

Nous comparons (de façon simplifiée) en figure 1.2 le coût de communication d'un message entre les principaux mécanismes de routage (routage par commutation de messages ou paquets, commutation de circuits et *wormhole*) dans un réseau en absence de toute contention.

Le cas du *virtual-cut-through* n'est pas représenté, car en l'absence de contention il est identique au routage *wormhole*. Le temps de propagation sur un lien n'est pas pris en compte ici. Cette figure représente "l'activité" de chaque nœud (axe des ordonnées) en fonction du temps (axe des abscisses) pour un message allant de l'émetteur source S vers son destinataire D via les nœuds intermédiaires $N1$, $N2$ et $N3$.

A la différence de la commutation de messages ou de paquets, la figure 1.2 montre clairement que le routage par commutation de circuits et *wormhole* ont des coûts de communication pratiquement indépendants de la longueur du chemin établi entre la source et la destination. Clairement, cela signifie que le routage par commutation de messages (ou paquets) sera très sensible au diamètre du réseau, alors que les routages du type commutation de circuits ou *wormhole* le seront peu. Cette caractéristique a été confirmée par des mesures sur des machines réelles [17].

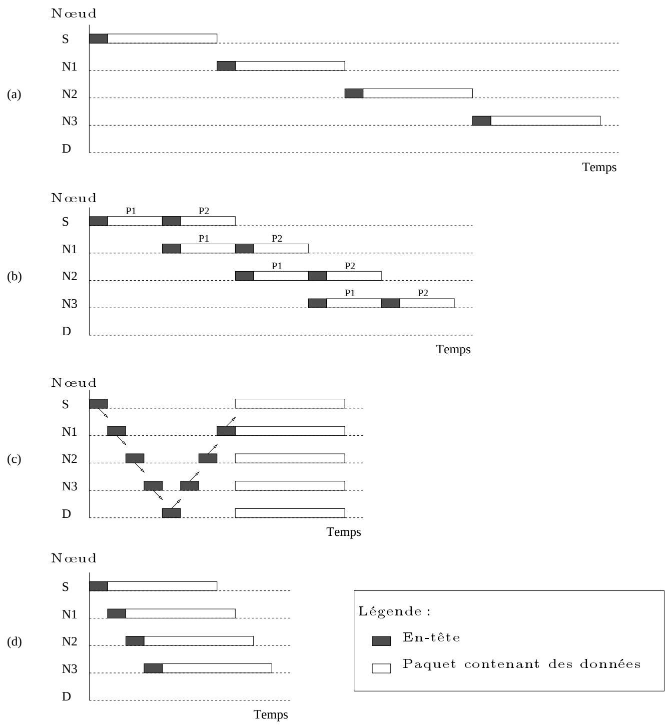

FIG. 1.2 – : *Différentes techniques de commutation : (a) commutation de messages ; (b) commutation de paquets ; (c) commutation de circuits ; (d) wormhole.*

Une grande partie de cette thèse traite des communications dans les réseaux d'interconnexion. Nous verrons, pour le problème particulier des communications globales que nous introduisons au chapitre 2, que nous pourrons regrouper sous un seul et même modèle théorique l'ensemble des mécanismes de routage du type *wormhole* qui sont désormais les plus populaires. Sous cette optique, les chapitres 3 et 4 dressent une synthèse des travaux les plus significatifs dans ce domaine. C'est également dans ces chapitres que s'intègre une part importante de nos travaux sur ces problèmes de communications globales.

1.4 Graphes et réseaux d'interconnexion

La théorie des graphes [4] apparaît être l'un des outils permettant d'obtenir une bonne modélisation des réseaux d'interconnexion [13, 18]. Comme nous en aurons besoin dans toute la suite, nous définissons dans cette section quelques graphes et éléments de théorie des graphes les plus fréquemment utilisés dans cette thèse. Nous renvoyons le lecteur au livre [18] pour les définitions non données ici et les preuves des propriétés des réseaux abordées dans cette section.

Un réseau est en général modélisé par un graphe G orienté. Dans le cas ou celui-ci est symétrique on utilisera souvent la terminologie plus classique de graphe non orienté et c'est la manière dont nous présenterons les réseaux dans cette section.

1.4.1 Constructions classiques

Il existe des méthodes classiques de construction de graphes permettant de générer des familles (ou classes) de graphes ayant des propriétés de régularité. Généralement ces méthodes permettent d'obtenir un résultat immédiat pour tous ces graphes, dès qu'il est prouvé pour la classe à laquelle ils appartiennent.

Remarque. Il existe de nombreuses constructions. Nous n'en présentons ici que deux, celles les plus utilisées dans cette thèse.

Definition. La *somme cartésienne* (*cartesian sum*, souvent appelée *cartesian product*) de deux graphes G et G', que nous noterons $G\square G'$, est le graphe dont les sommets sont tous les couples (x, x') où x est un sommet de G et x' est un sommet de G'. Deux sommets (x, x') et (y, y') de $G\square G'$ sont adjacents si et seulement si $x = y$ et $[x', y']$ est une arête de G' ou si $x' = y'$ et $[x, y]$ est une arête de G.

La même définition s'applique aux graphes orientés, en remplaçant les arêtes par des arcs. Cette opération est associative et commutative.

Definition. Soit un graphe orienté G, on appelle *graphe représentatif des arcs* du graphe G (*line-digraph*), le graphe orienté $L(G)$ dont les sommets représentent les arcs de G et dont les arcs sont définis de la façon suivante : il existe un arc du sommet e vers le sommet f dans $L(G)$, si et seulement si l'arc de G représenté par e a pour extrémité terminale, l'extrémité initiale de l'arc de G représenté par f.

On définit inductivement le $k^{ème}-$itéré du graphe représentatif des arcs de la façon suivante :

$$\begin{aligned} L^0(G) &= G \\ L^k(G) &= L(L^{k-1}(G)) \quad pour \quad k \geq 1 \end{aligned}$$

1.4.2 Réseaux classiques

Nous donnons ici, les définitions de quelques réseaux couramment utilisé dans la suite de cette thèse.

Le cycle

C_N représente le *cycle* d'ordre N. C'est un graphe connexe régulier (tous les sommets ont le même degré) de degré 2, à N sommets.

Le graphe complet

K_N représente le *graphe complet d'ordre N*. C'est un graphe à N sommets dont deux sommets quelconques sont adjacents. La distance entre deux sommets quelconques est 1, donc le diamètre est égal à 1.

Remarque. On utilise aussi K_N^*, *graphe complet orienté symétrique*, dans lequel il existe un arc de tout sommet x vers tout sommet y. Le diamètre de K_N^* est 1. Pour tout sommet x, $d^+(x) = d^-(x) = N - 1$.

L'hypercube

On appelle *hypercube de dimension* n et on note $H(n)$, le graphe dont les sommets sont les mots de longueur n sur un alphabet à deux lettres 0 et 1, et dont deux sommets sont adjacents si et seulement si ils diffèrent en une seule coordonnée. Un sommet, noté $x_1x_2\cdots x_i\cdots x_n$, est donc relié aux sommets $x_1x_2\cdots\overline{x_i}\cdots x_n$, avec $i = 1, 2, \ldots, n$.

$H(n)$ a 2^n sommets et est régulier de degré n. C'est un graphe de Cayley et son diamètre est n. On peut remarquer que $H(1)$ est le graphe isomorphe à une arête, $H(2)$ est isomorphe à C_4 et $H(3)$ peut se voir comme le cube classique. Notons également que $H(n)$ est aussi une somme cartésienne de K_2 :

$$H(n) = K_2 \Box H(n-1) = \underbrace{K_2 \Box K_2 \Box \cdots \Box K_2}_{n \text{ fois}}$$

La grille

Etant donnés d entiers n_i tels que, $\forall i \in \{0, \cdots, d-1\}$, $n_i \geq 2$, la *grille de dimension d*, appelée aussi *grille d-dimensionnelle* et notée $M(n_0, n_1, \ldots, n_{d-1})$, est la somme cartésienne de d chaînes de n_i sommets, soit $P_{n_0} \Box P_{n_1} \Box \ldots \Box P_{n_{d-1}}$. Les sommets de la grille sont les d-uplets $(x_0, x_1, \ldots, x_{d-1})$, avec $x_i \in [0, n_i - 1]$ et $i \in [0, d-1]$. Une arête joint deux sommets qui diffèrent de 1 sur l'une de leurs coordonnées. Enfin, on appelle *grille n-régulière* et on note $M(n)^d$ une grille telle que : $\forall i \in [0, d-1]$, $n_i = n$.

La grille torique

Un **tore de dimension** k est la somme cartésienne de k cycles d'ordre $l_1, l_2, \ldots, l_k$ et sera noté par $TM(l_1, l_2, \ldots, l_k) = C_{l_1} \Box C_{l_2} \Box \cdots \Box C_{l_k}$. Lorsque $l_1 =$

$l_2 = \cdots = l_k = l$, nous utiliserons la notation condensée $TM(l)^k$. Nous appellerons ces graphes des **tores carrés de coté $\boldsymbol{l}$ et de dimension $\boldsymbol{k}$**.

Le graphe de de Bruijn

Le *graphe de de Bruijn* $B(d, D)$ (aussi noté $\mathcal{B}(d, D)$) est le graphe orienté dont les sommets sont les mots de longueur D sur un alphabet de taille d ($d \geq 2$) et dont un sommet, noté $x_1x_2\cdots x_D$, est relié aux sommets $x_2\cdots x_D\lambda$, λ étant une lettre quelconque de l'alphabet. On passe donc d'un sommet à un autre par un décalage à gauche avec adjonction d'une lettre. On notera $x_1x_2\cdots x_Dx_{D+1}$ l'arc reliant le sommet $x_1x_2\cdots x_D$ au sommet $x_2\cdots x_Dx_{D+1}$.

Notons que le graphe de de Bruijn est aussi un graphe représentatif des arcs puisque $B(d, D) = L(B(d, D-1))$.

Le graphe *Butterfly*

Le *graphe Butterfly de demi-degré d et de dimension n*, noté $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d, n)$, est le graphe orienté dont les sommets sont les couples (x, l) où x est un mot de longueur n dont les lettres sont les éléments de $\mathbb{Z}_d$ et l est le numéro de *niveau*, tel que $0 \leq l < n$. Un sommet, noté $(x_{n-1}x_{n-2}\cdots x_l\cdots x_0,\ l)$ est relié par un arc à tout sommet $(x_{n-1}x_{n-2}\cdots x_{l+1}\ (x_l+\alpha)\ x_{l-1}\cdots x_0,\ l+1 \pmod n)$ où α est une lettre prise parmi les éléments de $\mathbb{Z}_d$. Chacun de ces arcs est dit être de *pente* α.

Cette définition construit des graphes orientés d-réguliers d'ordre nd^n et de diamètre $2n-1$. On en obtient une version non orientée, notée $\mathcal{WBF}(d, n)$, en remplaçant les arcs par des arêtes.

Il existe une autre définition de ce graphe, plus courante mais aussi plus réductrice. En effet, on note $BF(n)$ et on appelle *graphe Butterfly de dimension n* le graphe isomorphe à $\mathcal{WBF}(2, n)$. C'est pour cette raison qu'on l'appelle communément *graphe Butterfly binaire* (voir figure 1.3), par opposition à ce que l'on pourrait appeler le *graphe Butterfly généralisé* de la définition précédente.

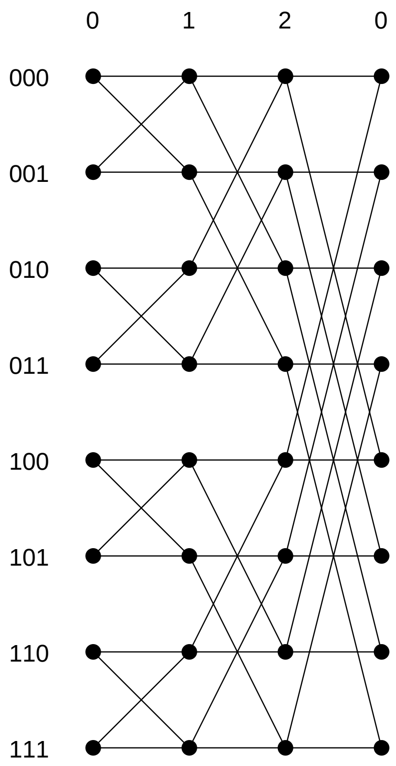

Fig. 1.3 – : *Graphe Butterfly* $\mathcal{WBF}(2, 3)$ *ou* $BF(3)$

Le degré du graphe $BF(n)$ est toujours 4, son ordre est $n2^n$ et son diamètre vaut : $n + \lfloor \frac{n}{2} \rfloor$.

Remarque. De manière à obtenir une figure plus lisible, on représente presque toujours le graphe *Butterfly* en dédoublant le niveau 0, comme sur la figure 1.3. Cependant, ce graphe ne possède bien que n niveaux de d^n sommets. C'est pour cette raison qu'on l'appelle aussi *graphe Butterfly rebouclé*, par opposition au réseau *multiétages* correspondant, qui lui, compte $n + 1$ niveaux (ou *étages*). Ce dernier est noté $\vec{\mathcal{BF}}(d, n)$ en orienté et $\mathcal{BF}(d, n)$ en non orienté. Il est aussi appelé *graphe FFT* car sa structure est calquée sur les permutations de données d'une Transformée de Fourier Rapide. Notons que certains auteurs ou ouvrages (par exemple [18]) représente ce graphe dans le sens opposé, i.e. en dessinant de la gauche vers la droite les niveaux $0, n - 1, n - 2, \ldots, 1, 0$.

Bibliographie

[1] S. G. Akl. *The Design and Analysis of Parallel Computers.* Prentice-Hall, 1989.

[2] G. Almasi and A. Gottlieb. *Highly Parallel Computing.* Benjamin Cummings Publishing Co., 1989.

[3] G. S. Almasi. Overview of parallel processing. *Parallel Computing*, 2:191–203, 1985.

[4] C. Berge. *Graphes.* Gauthiers-Villars, 1983.

[5] J. Dongarra and I. S. Duff. Advanced architecture computers. Technical Report CS-89-90, University of Tennessee, Computer Science Department, November 1989.

[6] E. Fleury. *Communications, routage et architectures des machines à mémoire distribuée - Autour du routage wormhole.* Thèse de doctorat, Université de Lyon, Ecole Normale Supérieure de Lyon, 1996.

[7] M. J. Flynn. Very high speed computing systems. In *IEEE Proceedings*, volume 54, pages 1901–1909, 1966.

[8] P. Fraigniaud. *Vers un principe de localité pour les communications dans les réseaux d'interconnexion.* Thèse d'habilitation, Université de Lyon, Laboratoire de l'Informatique du Parallèlisme - CNRS Ecole Normale Supérieure de Lyon, 1995.

[9] R. W. Hockney and C. R. Jesshope. *Parallel Computers 2, Architecture, Programming and Algorithms.* IOP Publishing ltd, 1988.

[10] K. Hwang and F. A. Briggs. *Computer Architecture and Parallel Processing.* McGraw-Hill, 1984.

[11] D.D. Kandlur and K.G. Shin. Reliable brodcast algorithm for HARTS. *ACM Transactions on Computer Systems*, 9(4):374–398, November 1991.

[12] P. Kermani and L. Kleinrock. Virtual cut-through: a new computer communication switching technique. *Computers Networks*, 3:267–286, 1979.

[13] F. Thomson Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays . Trees . Hypercubes.* Computer Science, Mathematics, Electrical Engineering. Morgan Kaufmann Publishers, 1992.

[14] X. Lin, P.K. McKinley, and L.M. Ni. Deadlock-free multicast wormhole routing in 2D-mesh multicomputers. *IEEE Transactions on Parallel and Distributed Systems*, 5(8):793–804, August 1994.

[15] X. Lin and L.M. Ni. Deadlock-free multicast wormhole routing in multicomputer networks. In *18th Annual International Symposium on Computer Architecture*, pages 116–125. ACM, 1991. (see also Technical Report MSU-CPS-ACS-29, 1990).

[16] L. M. Ni. Should scalable parallel computers support efficient hardware multicast? In *International Conference on Parallel Processing (ICPP '95), Workshop on Challenges for Parallel Processing*, pages 2–7. IEEE, 1995.

[17] L.M. Ni and P.K. McKinley. A survey of wormhole routing techniques in direct networks. *IEEE Computers*, 26(2):62–76, February 1993.

[18] Jean de Rumeur. *Communication dans les réseaux de processeurs.* Collection Etudes et Recherches en Informatique. Masson, Paris, 1994.

[19] C. L. Seitz. Concurrent architectures. In R. Suaya and G. Birtwist, editors, *VLSI and Parallel Computation*, pages 1–84. Morgan Kaufmann, 1990.

[20] H. S. Stone. *High-Performance Computer Architecture.* Addison-Wesley, 1987.

[21] E. Upfal. On the theory of interconnection networks for parallel computers. Research Report CS94-06, Weizmann Institute of Science, Department of Applied Mathematics and Computer Science, April 1994.

Chapitre 2

Introduction aux communications globales en mode commutation de circuits

✔ *La littérature traitant des communications globales est très abondante (voir les articles de synthèse de Fraigniaud et Lazard [17], d'Hedetniemi, Hedetniemi et Liestman [19] et de Hromković, Klasing, Monien et Peine [21]). Ces synthèses n'adressent cependant que le problème des communications globales en mode* «commutation de messages»*. Notre objectif est d'effectuer une synthèse identique, mais dédiée au mode* « commutation de circuits »*. Celle-ci regroupe les chapitres 2, 3 et 4. Dans ce chapitre introductif, nous décrivons notre modélisation des problèmes.*

Dans tout ce chapitre ainsi que dans les chapitres 3 et 4, nous nous placerons dans le cadre de machines parallèles à mémoire ditribuée pour lesquelles les échanges de données entre processeurs s'effectuent par échange de messages sur un réseau d'interconnexion point-à-point statique[1]. De plus, nous ne ferons pas ici de différence entre processeur et routeur et les assimilerons à une seule et même entité.

Les problèmes de routage des communications intervenant au cours d'applications parallèles, au sens large du terme (calculs scientifiques, systèmes d'exploitation, etc) peuvent être regroupés en deux grandes classes : les routages *"temps-réels"*[2] et les routages *"précalculés"*[3] [16, 23].

1. C'est-à-dire non-reconfigurable.
2. *"On-line"* en anglais.
3. *"Off-line"* en anglais.

Le terme *"temps-réel"* est utilisé lorsque, et c'est le cas dans de nombreuses applications, le problème de routage n'est pas connu à l'avance (i.e. non connu avant l'exécution du programme). Cependant, nous pouvons quelquefois connaître par avance des schémas de communications qui interviendront à coup sûr dans l'application. Dans ce cas, il est possible de calculer, avant l'exécution, une solution au routage de tels schémas. Nous utiliserons pour cela le terme *"précalculés"*. Résoudre, avant l'exécution, de tels problèmes peut être particulièrement intéressant dans le cas où ils sont susceptibles d'intervenir un grand nombre de fois au cours de l'exécution d'une même application. Le lecteur trouvera dans [16] une définition plus formelle des problèmes de routage *"temps-réel"* et *"précalculé"*.

Un tel classement est justifié par le fait que les techniques utilisées pour résoudre ces deux types de problèmes sont différentes. En effet, c'est à l'exécution du programme[4] qu'une solution au problème du routage *"temps-réel"* doit être trouvée. Cela signifie que chaque processeur doit décider du routage des messages qu'il reçoit et ne peut pour cela se baser que sur un contrôle local (comme le contenu de différents registres, les connexions établies entre différents ports d'entrée et de sortie, le contenu du message à router, etc). Dans ce cas de figure, les communications peuvent intervenir à tout moment, sans aucune synchronisation entre elles, et on pourra parler de communications "anarchiques". L'étude de cette classe de problèmes passe par l'étude de fonctions de routages qui essayent de garantir au mieux des communications de faibles coûts tout en essayant d'éviter des problèmes de blocage (comme l'interblocage ou le mouvement perpétuel) et fait, pour cela, généralement intervenir des outils théoriques particuliers (graphes de dépendances, canaux virtuels, etc) [5, 6, 7, 9, 10, 11, 12, 14].

Ainsi, au cours d'une application, les processeurs communiquent en échangeant des messages, mais la répartition des communications sur le réseau dépend fortement de l'algorithme utilisé. Heureusement, l'étude de paradigmes classiques montre qu'il existe dans la classe des routages *"précalculés"* un certain nombre de "communications structurées (globales, généralisées, ou collectives)" qui apparaissent très souvent dans de nombreux problèmes de calculs parallèles ou distribués [1, 8, 15], comme par exemple en algèbre linéaire ou non-linéaire [3, 4, 20], en traitement d'images [25], ou bien encore dans les systèmes de bases de données [18, 29]. Dès lors, il sera intéressant d'étudier ces schémas de communications globales, pour éviter qu'ils soient résolus seulement à l'exécution. Alors, lorsqu'un programme fera intervenir au cours de son exécution un tel schéma de communication, il n'aura qu'à appliquer la solution précalculée. C'est cette classe particulière de communications structurées ou globales que nous étudions dans ce chapitre.

4. *"Run-time"* en anglais.

2.1 Communications globales

Il existe de nombreux schémas de communications globales. Nous définissons ici, les schémas les plus fréquemment mis en œuvre.

- **La diffusion** (one to all ou broadcasting) : opération qui consiste à envoyer un message à tous les processeurs à partir d'un initiateur unique.
- **L'échange total** (commérage, all to all, total exchange ou gossiping) : opération qui consiste à effectuer une diffusion à partir de tous les processeurs simultanément.
- **La distribution** (diffusion personnalisée, personalized one to all, distributing ou scattering) : opération qui consiste, pour un initiateur unique, à envoyer un message différent à chacun des autre processeurs. La distribution se différencie de la diffusion par le fait que les messages échangés ne sont pas les mêmes tout au long du processus de communication, i.e. lorsqu'un processeur a reçu le message qui lui était destiné, il n'a plus besoin de le renvoyer aux autres.
- **Le rassemblement** (gathering) : opération qui consiste à récupérer en un processeur des données différentes, provenant de tous les autres. Remarquons, que cette opération est l'inverse de la distribution. Cela signifie que savoir réaliser une distribution, implique que l'on sache réaliser un rassemblement et réciproquement. L'étude de ces deux problèmes est donc identique.
- **La multidistribution** (échange total personnalisé, personalized all to all, complete exchange ou multiscattering) : opération qui consiste à effectuer une distribution simultanément à partir de tous les processeurs, i.e. chaque processeur veut envoyer un message différent à chacun des autres.

Il existe bien évidemment quantité d'autres types de communications globales ou structurées, comme par exemple le *"one to many"* où un processeur envoie le même message à un groupe particulier de processeurs (*"multicasting"*). On peut également considérer toutes les extensions du type *"many to many"* ou bien encore les permutations sur un ensemble de processeurs. Tout schéma de communications structurées peut motiver une étude. Pour notre part, nous aborderons ici seulement trois grands types d'opérations globales (couramment rencontrés [24]) que sont la `diffusion`, l'`échange total` et la `multidistribution`.

2.2 Le modèle de type commutation de circuits

Nous regrouperons sous le terme « modèle de type commutation de circuits », plusieurs modes de commutation. Nous verrons, que même si ces modes de commutation implantent des mécanismes différents, ils peuvent être classés dans un même ensemble pour le problème particulier des communications globales.

Ainsi, nous regrouperons dans le « **modèle de type commutation de circuits** », les routages par *commutation de circuits*, *"wormhole"* [28], *"virtual cut-through"* [22] ou *"direct connect"* [26]. Nous avons vu en section 1.2, que des différences existent entre ces divers techniques (par exemple, présence ou non d'un accusé de réception, faible ou grande taille des tampons d'entrée et sortie des liens, etc). Mais pour l'étude des communications globales, les différences entre ces modes de routage apparaissent comme trop fines et en tout état de cause difficilement modélisables pour être prises en compte dans une étude théorique.

Pour exemple, si l'on voulait étudier des schémas de communications globales utilisant le routage *"wormhole"*, il faudrait prendre en compte le fait qu'un lien, utilisé pour router un message, ne peut router un nouveau message qu'à partir du moment où le dernier *"flit"* du premier message a terminé de traverser ce lien. Ceci implique que le coût d'une telle opération est très fortement lié à la longueur des messages, mais aussi à la taille des tampons [5] des routeurs. En effet, si les messages sont petits, ils n'occuperont les liens de communications que très peu de temps. Au contraire, si les messages sont plutôt longs, les liens sur lesquels ils transitent seront occupés plus longtemps, empêchant un autre message de réutiliser un de ces liens ; ce nouveau message devra alors trouver un autre chemin disjoint afin d'établir sa communication. Ainsi, il est clair que pour un problème de communication globale fixé et pour un réseau donné, les chemins des communications seront totalement différents en fonction de la taille des messages. De plus, avec ce mode de routage le nombre d'étapes d'un algorithme devient une notion plutôt floue, car il n'y a pas forcément de synchronisation entre les communications. Une telle étude apparaît donc comme très difficile d'un point de vue théorique.

Dans le but de pouvoir comparer les divers protocoles proposés et d'évaluer leur qualité, il est nécessaire d'adopter un modèle commun. Ceci a déterminé la plupart des auteurs à se placer dans le cadre du « modèle de type commutation de circuits » qui est désormais le plus communément adopté. Celui-ci consiste à dire que les **protocoles de communications globales s'effectuent comme une succession d'étapes**. Une étape comporte généralement plusieurs communications entre divers couples de sommets. Ces communications, pour éviter des blocages ou des pertes, devront s'effectuer le long de **chemins arc-disjoints**. Dans la terminologie des télécommunications un chemin est appelé circuit dans la mesure ou l'on établit au moins virtuellement une communication de x à y et de y à x. Le coût d'une étape est le maximum du coût des communications ayant lieu au cours de cette étape. Souvent, il est pratique de considérer qu'**une**

5. *"Buffers"* en anglais.

étape ne peut commencer que si toutes les communications de l'étape précédente sont achevées. Dans ce cas, on dit que les protocoles sont « synchrones». Il devient alors possible de comparer les résultats entre eux et de donner des bornes sur les problèmes étudiés. On pourra ainsi parler d'optimalité ou de non-optimalité des protocoles. Un tel modèle théorique est motivé par le fait suivant : l'implémentation d'un protocole optimal ou même quasi-optimal, dans le modèle simplifié théorique, fournira sur une machine réel avec un mécanisme de routage complexe (comme par exemple le *"wormhole"*), un protocole, il est vrai non-optimal, mais à coup sûr efficace. C'est ce fait qui a conduit un grand nombre de chercheurs à étudier de façon théorique les problèmes de communications globales.

Dans la suite, nous regrouperons les routages du « type commutation de circuits » sous le terme `mode commutation de circuits`.

2.3 Modélisation des problèmes

Maintenant que nous avons donné le modèle général de communication que nous utiliserons, il est désormais nécessaire de modéliser avec précision le temps de transfert d'un message entre deux processeurs.

2.3.1 Temps de communication d'un processeur à un autre

Dans le mode commutation de circuits, lorsqu'un processeur envoie un message à un autre processeur, la modélisation du temps de transmission s'exprime comme la somme de trois termes : le premier tient compte d'un délai constant, aussi appelé délai d'initialisation[6], induit par le processus qui envoie le message ; le second est associé au temps de commutation des commutateurs par lesquels transite le message et le troisième mesure le débit de l'information. Ainsi, pour un message de longueur L allant d'un processeur x à un processeur y, le long d'un chemin de longueur l, le temps de communication sera modélisé par $T_{x \to y} = \alpha + l\delta + L\tau$, où α est le délai d'initialisation[7], δ le temps de commutation d'un commutateur intermédiaire[8] et $\frac{1}{\tau}$ la bande passante des liens[9].

6. Ceci inclut, si besoin, les délais nécessaires aux divers accusés de réception.
7. *"Start-up time"* en anglais.
8. *"Hop time"* en anglais.
9. Parfois τ est appelé *"flit-transfer time"* en anglais.

☞ Il est intéressant de remarquer que le modèle théorique commutation de circuits est au moins aussi puissant que le modèle commutation de messages dans lequel un processeur ne peut envoyer et recevoir des messages qu'avec ses voisins directs. En effet, la modélisation du temps de communication d'un processeur à un autre en commutation de messages dépend de deux paramètres : le premier tient compte d'un délai d'initialisation et le second mesure le flot d'information. Ainsi, pour un message de longueur L allant d'un processeur x à un processeur y, le long d'un chemin de longueur l, le temps de communication sera modélisé par $T_{x \to y} = l(\beta + L\tau)$, où β est le délai d'initialisation et $\frac{1}{\tau}$ la bande passante des liens (pour une description plus précise du mode commutation de messages voir [27]). Donc, le mode commutation de circuits peut simuler la commutation de messages, puisqu'il suffit d'envoyer les messages à distance au plus 1. Cela correspond à dire $\alpha + \delta = \beta$. Alors, tout algorithme construit pour la commutation de messages fonctionnera en commutation de circuits.

Il existe diverses simplifications justifiées par la réalité de ce modèle de temps. En effet, les paramètres α, δ et τ diffèrent généralement entre les diverses machines réelles qui implémentent des mécanismes de routage du type commutation de circuits, et certains paramètres négligeables devant les autres dans une machine, ne le sont plus du tout dans une autre machine. Une autre hypothèse simplificatrice très souvent utilisée, revient à considérer des messages de petite longueur pour lesquels $\alpha >> L\tau$ et $l\delta >> L\tau$. Le modèle de temps peut alors se réduire à $T_{x \to y} = \alpha + l\delta$. Le cas des messages longs se traitent généralement avec des techniques très différentes (pipeline, arbres couvrants arc-disjoints, ...) de celles utilisées pour les messages courts [27].

Nous résumons dans le tableau 2.1, les principaux modèles de temps les plus communément utilisés pour un message de longueur L allant d'un processeur x à un processeur y le long d'un chemin de longueur l.

MODÈLE	HYPOTHÈSES
$\boldsymbol{\alpha + l\delta + L\tau}$	Modèle linéaire complet
$\boldsymbol{\alpha + l\delta}$	Petit message (L petit) et/ou $\alpha >> L\tau$ et $l\delta >> L\tau$
$\boldsymbol{\alpha + L\tau}$	Distance négligeable $\alpha >> l\delta$ et $L\tau >> l\delta$
$\boldsymbol{\alpha}$	Modèle temps constant $\alpha >> l\delta$ et $\alpha >> L\tau$

TAB. 2.1 – : *Principales modélisations du temps de communication d'un message de taille L le long d'un chemin de longueur l.*

REMARQUE. Il est à noter que le terme α apparaît dans tous ces modèles. Ceci est justifié par le fait qu'actuellement toutes les machines implémentant un routage du type commutation de circuits ont comme paramètre prépondérant leur délai d'initialisation.

2.3.2 Coût des protocoles de communications globales

Soient G un graphe connexe modélisant un réseau d'interconnexion, et x un sommet de G.

- **Le temps (ou coût) de diffusion du sommet $\boldsymbol{x}$**, noté $b(x, G)$, est le temps minimum nécessaire pour effectuer la diffusion dans G à partir du sommet x. Dans le modèle linéaire complet, ce temps s'exprime comme la somme de trois termes : $b(x, G) = b_\alpha(x, G)\alpha + b_\delta(x, G)\delta + b_\tau(x, G)L\tau$, où $b_\alpha(x, G)$ représente le nombre d'étapes, $b_\delta(x, G)$ la somme du maximum des distances des communications entre processeurs impliqués à chaque étape et $b_\tau(x, G)$ mesure le flot d'information.

- **Le temps (ou coût) de diffusion du graphe $\boldsymbol{G}$**, noté $b(G)$, est le maximum de tous les temps de diffusion des sommets de G, c'est-à-dire $b(G) = \max_{x \in V(G)} b(x, G)$. Comme précédemment, ce temps de diffusion s'exprime comme la somme de trois termes : $b(G) = b_\alpha(G)\alpha + b_\delta(G)\delta + b_\tau(G)L\tau$.

Des définitions similaires seront utilisées pour le temps d'échange total : $g(G) = g_\alpha(G)\alpha + g_\delta(G)\delta + g_\tau(G)L\tau$; et pour le temps de multidistribution : $m(G) = m_\alpha(G)\alpha + m_\delta(G)\delta + m_\tau(G)L\tau$.

Notation 1 — *Lorsqu'il n'y aura pas d'ambiguïté, nous utiliserons la notation simplifiée suivante : $b(G) = b_\alpha\alpha + b_\delta\delta + b_\tau L\tau$ pour la diffusion. De même, pour l'échange total et la multidistribution nous employerons le même type de notation simplifiée.*

Remarque. Il va de soi que le coût d'une opération globale dépendra du modèle de temps utilisé. Ainsi, par exemple, le coût de la diffusion dans le modèle "petit message" s'exprimera alors uniquement comme la somme de deux termes : $b(G) = b_\alpha(G)\alpha + b_\delta(G)\delta$. Le coût de l'échange total dans le modèle temps constant ne dépendra plus que d'un seul terme : $g(G) = g_\alpha(G)\alpha$.

2.3.3 Choix d'une fonction de routage

Le but recherché dans les problèmes de communications globales est d'obtenir un coût ("réalisable" sur une machine réelle durant l'exécution de vraies applications) de communication le plus faible possible. Ce problème peut en fait être abordé de deux façons différentes, suivant que l'on impose ou non à l'avance une fonction de routage.

En effet, certains auteurs considèrent d'abord le problème de communication en faisant abstraction de la méthode de routage, puis essayent de construire cette dernière, permettant d'obtenir un coût de communication minimum. D'autres considèrent les communications en tenant compte directement d'un couple réseau d'interconnexion - méthode de routage.

L'inconvénient de la première approche est de risquer de ne pas prendre en compte ses défauts structurels, ce qui n'est pas le cas de la seconde. Par contre, l'avantage de la première est que les résultats obtenus seront meilleurs (en terme de coût, ou de bornes inférieures) que pour la seconde.

Cette seconde approche est justifiée par le fait que les machines parallèles existantes, implémentent généralement un type de routage particulier, et doivent réaliser toutes les opérations de communications, qu'elles soient structurées ou non, via leur routage prédéterminé. Citons pour exemple, le cas de la machine Cray-T3D[10] qui implémente une fonction de routage du type XYZ. Cette machine utilise cette dernière pour réaliser toutes les communications intervenant au cours d'une application. En fait, elle dispose pour cela d'une table (fichier) de routage chargée sur les routeurs lors de son initialisation. Il est possible de lui fournir d'autres tables (fichiers) de routage, mais pour qu'un nouveau routage soit pris en compte par les routeurs, il est nécessaire de réinitialiser la machine, ce qui implique l'impossibilité de changer les fonctions de routage dynamiquement au cours d'une application. Mais, même si un tel routage restreint forcément les capacités de communications de la machine (par exemple, avec un routage XYZ sur un tore en 3 dimensions comme le Cray-T3D, l'émission d'un message ne peut se faire en $\Delta-$ports), comme très souvent une communication globale intervient dans un réseau en même temps que d'autres communications globales ou anarchiques, alors la charge moyenne des liens est très souvent proche du maximum. Alors le comportement moyen de la machine dans ce cas de figure tend à atteindre un comportement optimal. Par contre, dans la nouvelle machine Cray-T3E[11], il semble qu'il sera possible de redéterminer dynamiquement le type de routage utilisé par les routeurs.

Pour ces raisons, les deux approches du problème des communications globales avec ou sans routage imposé sont dignes d'intérêt. Dans la suite nous différencierons ces deux approches en utilisant la notation 2.

Notation 2 — *Si la fonction de routage est imposée a priori, nous la noterons* $\mathcal{R}_{\texttt{fonction}}$. *Si elle n'est pas fixée à l'avance, nous la noterons* $\mathcal{R}_{\star}$.

10. La topologie de cette machine est un tore de dimension 3.
11. Cette nouvelle machine implémente également une topologie de tore de dimension 3.

2.3.4 Contraintes dues au réseau

Depuis la création de l'informatique et la mise au point des premières machines parallèles jusqu'à nos jours, les technologies (composants, logiciels, etc) n'ont cessé d'évoluer. Une modélisation unique des diverses machines est impossible. Or, il apparaît important de modéliser correctement le réseau étudié. Pour cela, il faut prendre en compte ces contraintes technologiques. Dans la suite nous présentons les principales contraintes qu'il est nécessaire de considérer pour une bonne modélisation. Elles portent sur :

- la nature des liens de communications,
- les communications,
- les émetteurs et les récepteurs.

Nature des liens de communications

La modélisation du réseau se fera à l'aide d'un graphe orienté dans lequel les processeurs seront représentés par les sommets du graphe. Un lien physique entre un processeur x et y sera représenté dans le graphe orienté par un arc (x, y) si et seulement si dans le réseau le processeur x est capable d'envoyer un message vers y via ce lien. On parle alors de liens unidirectionnels. Cependant, dans la plupart des machines existantes si un processeur x peut communiquer avec un processeur y, alors la réciproque est vraie. On modélise un tel réseau soit par un graphe non orienté soit par un graphe orienté symétrique et on parle de liens bidirectionnels entre x et y. Nous choisirons ici la modélisation par un graphe orienté symétrique[12] qui nous permettra de distinguer deux contraintes, couramment utilisées, sur les liens.

- Si un lien ne peut transmettre qu'une seule information à une étape donnée, donc dans **un seul sens à la fois**, alors le lien est dit fonctionner en mode **"télégraphique"**[13]. Dans le graphe modélisant le réseau, cela signifie que l'on interdit d'établir des circuits de communication de longueur 2.
- Si un lien peut transmettre, à une étape donnée, simultanément deux messages dans **les deux directions opposées**, le lien sera dit fonctionner en mode **"téléphonique"**[14]. Dans le graphe modélisant le réseau, cela signifie que l'on autorise l'utilisation de circuits de communication de longueur 2.

12. Dans ce cas même si nous représentons dans les figures, un lien $[x, y]$ par une arête, cela sous-entend que cette arête figure les arc (x, y) et (y, x).
13. *"Half-duplex"* en anglais.
14. *"Full-duplex"* en anglais.

☞ Jusqu'à présent, toutes les machines parallèles implémentent des liens de communications homogènes, c'est-à-dire de même nature. C'est pour cela que nous ne prenons pas en compte la modélisation d'une machine qui implémenterait des liens hétérogènes, c'est-à-dire ayant une partie de ses liens en mode télégraphique, une autre en mode téléphonique.

Conditions sur les communications

Dans une opération synchrone de communications globales, les processeurs d'un réseau communiquent par étapes. La contrainte physique consiste à réserver un canal de communication, à une étape donnée, pour un seul message. Ceci conduit à imposer la condition suivante sur les communications.

« A `chaque étape` d'un protocole d'opération synchrone globale, les communications devront nécessairement s'établir le long de `chemins arc-disjoints` dans le graphe modélisant le réseau. »

Conditions sur les récepteurs et émetteurs

Il faut également prendre en compte les contraintes technologiques sur les capacités des routeurs à émettre et transmettre simultanément plusieurs messages sur leurs ports d'entrée et de sortie. Si un nœud (sommet, processeur ou routeur) ne peut gérer à un instant donné qu'un seul de ses ports en entrée et en sortie, on parlera de modèle **1−port**. S'il peut en gérer simultanément k, on parlera de modèle ***k*−ports**. Et enfin s'il est capable de gérer tous ses liens simultanément nous parlerons de modèle **Δ−ports**.

Nous résumons, dans le tableau 2.2, les contraintes physiques des liens et des routeurs, en utilisant la notation standardisée de [17, 27].

- *Contraintes technologiques* -			
NATURE DES LIENS	CAPACITÉ DES ROUTEURS		
	`1-port`	k-`ports`	Δ-`ports`
`Télégraphe`	H_1	H_k	H_*
`Téléphone`	F_1	F_k	F_*

TAB. 2.2 – : *Notations des divers contraintes technologiques nécessaires à la modélisation d'un problème de communications globales.*

2.4 Notation complète

Comme le coût d'un protocole de communications globales dépendra des hypothèses utilisées (contraintes technologiques, fonction de routage), nous devrons faire apparaître ces hypothèses dans la notation du coût. Pour cela, nous reprenons les notations fixées dans [17, 27] et qui sont aussi reprises dans [13].

Notation 3 — *Etant donné un graphe G et un sommet initiateur u. Le temps de diffusion du sommet u sous la contrainte $\mathcal{M} \in \{H_1, H_k, H_*, F_1, F_k, F_*\}$, en employant une fonction de routage fixée ou non à l'avance $\mathcal{R} \in \{\mathcal{R}_{\texttt{fonction}}, \mathcal{R}_\star\}$ sera noté par $b_{\mathcal{M}}^{\mathcal{R}}(u, G)$. De même, le temps de diffusion dans le graphe G sera noté par $b_{\mathcal{M}}^{\mathcal{R}}(G) = \max_{x \in V(G)} b_{\mathcal{M}}^{\mathcal{R}}(x, G)$.*

Remarque. Le même symbolisme sera utilisé pour l'échange total et la multi-distribution, i.e. $g_{\mathcal{M}}^{\mathcal{R}}(G)$ et $m_{\mathcal{M}}^{\mathcal{R}}(G)$.

Dans un esprit de synthèse, nous représenterons systématiquement les principaux résultats évoqués dans toute la suite de ce chapitre, sous la forme d'un tableau du type :

$b_{\mathcal{M}}^{\mathcal{R}}(G)$		
b_α	b_δ	b_τ/L

signifiant : « *il existe un protocole de diffusion tel que* $b_{\mathcal{M}}^{\mathcal{R}}(G) \leq b_\alpha(G)\alpha + b_\delta(G)\delta + b_\tau(G)L\tau$ ». Le même symbolisme sera utilisé pour l'échange total et la multidistribution.

Enfin rappelons que $D(G)$ (également noté D, lorsqu'il n'y aura pas d'ambiguïté) représente le diamètre du graphe G (orienté ou non selon le cas).

2.5 Taxinomie des problèmes de communications globales

La classification des problèmes de communications globales dépend donc de plusieurs paramètres. Il faut prendre en compte :

- la topologie du réseau (`choix du graphe`),
- les caractéristiques physiques des liens (`télégraphiques ou téléphoniques`),
- les contraintes des routeurs (`1,` k `ou` Δ`−ports`),
- la fonction de routage (`imposée ou non à l'avance`),

– et bien entendu, le problème de communication globale étudié (`diffusion, échange total ou multidistribution`).

Toute combinaison de ces divers paramètres est susceptible d'engendrer un problème particulier. Tout choix de classification a ses avantages et inconvénients. Nonobstant, certains paramètres ont de nombreux points en commun. Tout d'abord, le type de problème posé paraît être un discriminant important. En effet, même s'il existe des similitudes entre plusieurs types de problèmes (comme par exemple la diffusion et l'échange total), souvent ils apparaissent comme différents sur des points fondamentaux (par exemple, les bornes inférieures diffèrent). Nous choisirons donc tout d'abord de séparer les problèmes entre eux. Ensuite, l'ordre du choix des autres paramètres de classification est semble-t-il plus ou moins important. Cependant, au vu des résultats existants, nous choisirons tout de même de classifier les problèmes en fonction de l'utilisation ou non d'une fonction de routage imposée à l'avance (dans l'un ou l'autre cas les bornes inférieures peuvent différer). Nous continuerons cette classification en prenant en compte les contraintes technologiques puis enfin la topologie du réseau.

Ainsi, en continuité directe d'un premier travail de synthèse effectué par E. Fleury dans [13], nous nous proposons dans la suite de regrouper les travaux qui nous paraissent être les plus significatifs sur les problèmes de diffusion, d'échange total et de multidistribution utilisant le modèle de « type commutation de circuits». Néanmoins, nous adressons essentiellement dans cette synthèse les travaux essayant de minimiser en premier lieu le nombre d'étapes d'un protocole de communications globales. Ce travail ainsi que l'étude préliminaire d'E. Fleury, devrait donner lieu à un article de synthèse [2] actuellement en cours de préparation.

Dans le seul but de garder un équilibre entre les divers chapitres de cette thèse, nous avons décidé de scinder toute la partie traitant des communications globales en plusieurs chapitres. Les deux chapitres faisant suite à cette partie introductive portent, l'un sur la diffusion, tandis que l'autre regroupe l'échange total et la multidistribution. Nonobstant ce découpage, il est clair que ces chapitres 2, 3 et 4 sont à regrouper sous une seule et même partie.

Bibliographie

[1] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Prentice–Hall, Englewood Cliffs, NJ, 1989.

[2] Jean de Casanice. An overview of wormhole routing. Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis CNRS URA 1376 - Equipe : *"Communication Algorithms at Sophia Antipolis - Nice"* (J-C. Bermond, E. Darrot, O. Delmas, E. Fleury, S. Perennes, M. Syska), En préparation.

[3] M. Cosnard and P. Fraigniaud. Finding the roots of a polynomial on an mimd multicomputer. *Parallel Computing*, 15:75–85, 1990.

[4] M. Cosnard and D. Trystram. *Algorithmes et architectures parallèles.* Informatique Intelligence Artificielle. InterEditions, 1993.

[5] W.J. Dally. Virtual-channel flow control. *IEEE Transactions on Parallel and Distributed Systems*, 3(2):194–205, March 1992.

[6] W.J. Dally and H. Aoki. Deadlock-free adaptative routing in multicomputer network using virtual channels. *IEEE Transactions on Parallel and Distributed Systems*, 4(4):466–475, April 1993.

[7] W.J. Dally and C.L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transsaction on Computers*, C-36(5):547–553, May 1987.

[8] J.J. Dongarra and D.W. Walker. Software libraries for linear algebra computation on high performances computers. *SIAM Review*, 37:151–180, 1995.

[9] J. Duato. A new theory of deadlock-free adaptive routing in wormhole networks. *IEEE Transactions on Parallel and Distributed Systems*, 4(12):1320–1331, 1993.

[10] J. Duato. On the design of deadlock-free adaptative multicast routing algorithms. *Parallel Processing Letters*, 3(4):321–333, December 1993. Special Issue on Algorithmic and Structural Aspects of Interconnection Networks.

[11] J. Duato. A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks. *IEEE Transactions on Parallel and Distributed Systems*, 6(10), October 1995.

[12] J. Duato. A theory of deadlock-free adaptive multicast routing in wormhole networks. *IEEE Transactions on Parallel and Distributed Systems*, 6(9), September 1995.

[13] E. Fleury. *Communications, routage et architectures des machines à mémoire distribuée - Autour du routage wormhole.* Thèse de doctorat, Université de Lyon, Ecole Normale Supérieure de Lyon, 1996.

[14] E. Fleury and P. Fraigniaud. Deadlocks in adaptive wormhole routing. Research Report 94-09, Laboratoire de l'Informatique du Parallélisme, LIP, École Normale Supérieure de Lyon, 69364 Lyon Cedex 07, France, March 1994. Revised December 94 - Submitted to the 6th Symposium IEEE on Parallel and Distributed Processing.

[15] G. Fox, M Johnsson, G. Lyzenga, S Otto, J. Salmon, and D Walker. *Solving Problems on Concurrent Processors, Volume I.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[16] P. Fraigniaud. *Vers un principe de localité pour les communications dans les réseaux d'interconnexion.* Thèse d'habilitation, Université de Lyon, Laboratoire de l'Informatique du Parallèlisme - CNRS Ecole Normale Supérieure de Lyon, 1995.

[17] P. Fraigniaud and E. Lazard. Methods and problems of communication in usual networks. *Discrete Applied Mathematics*, 53:79–133, 1994. Special volume proceedings international workshop on broadcasting and gossiping 1990.

[18] L. Gargano and A.A. Rescigno. Communication complexity of fault-tolerant information diffusion. In *Proceeding of the 5th IEEE Symposium on Parallel and Distributed Computing (SPDP 93)*, pages 564–571, Dallas, TX, 1993.

[19] S.M. Hedetniemi, S.T. Hedetniemi, and A.L. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 18:319–349, 1986.

[20] C.T. Ho and S.L. Johnsson. Optimum broadcasting and personalized communication in hypercubes. *IEEE Transactions on Computers*, 38(9):1249–1268, 1989.

[21] J. Hromkovic, R. Klasing, B. Monien, and R. Peine. *Combinatorial Network Theory*, chapter Dissemination of information in interconnection networks (broadcasting and gossiping), pages 125–212. Kluwer Academic, 1995.

[22] P. Kermani and L. Kleinrock. Virtual cut-through: a new computer communication switching technique. *Computers Networks*, 3:267–286, 1979.

[23] F. Thomson Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays . Trees . Hypercubes.* Computer Science, Mathematics, Electrical Engineering. Morgan Kaufmann Publishers, 1992.

[24] P. K. McKinley, Y-J. Tsai, and D. F Robinson. A survey of collective communication in wormhole-routed massively parallel computers. Technical Report MSU-CPS-95-35, Michigan State University, East Lansing, Michigan 48824, June 1994.

[25] S. Miguet. *Programmation dynamique et traitement d'images sur machines parallèles à mémoire distribuée.* PhD thesis, Université de Lyon, Ecole Normale Supérieure de Lyon, 1990.

[26] S.F. Nugent. The iPSC/2 direct-connect technology. In G.C. Fox, editor, *Proceedings of 3rd Conference on Hypercube Concurrent Computers and Applications*, pages 51–60. ACM, 1988.

[27] Jean de Rumeur. *Communication dans les réseaux de processeurs.* Collection Etudes et Recherches en Informatique. Masson, Paris, 1994.

[28] C.L. Seitz. Concurrent architectures. In R. Suaya and G. Birtwist, editors, *VLSI and Parallel Computation*, pages 1–84. Morgan Kaufmann, 1990.

[29] O. Wolfson and A. Segall. The communication complexity of atomic commitment and gossiping. *SIAM Journal on Computing*, 20:423–450, 1991.

Chapitre 3

La diffusion en mode commutation de circuits

✔ *Ce chapitre, partie intégrante du précédent, dresse une synthèse des travaux qui nous paraissent les plus significatifs sur le problème de la diffusion, tout du moins lorsque l'on cherche en premier lieu à minimiser le nombre d'étapes.*

Rappelons que dans tout ce chapitre, nous considérons des protocoles de diffusion synchrones, c'est-à-dire se déroulant comme une succession d'étapes et que les graphes considérés sont, sauf indication contraire, orientés symétriques.

3.1 Généralités

3.1.1 Equivalence des modes téléphoniques et télégraphiques pour le nombre d'étapes

Si l'on ne considère que le nombre d'étapes d'un protocole de diffusion dans lequel on ne découpe pas l'unique message à transmettre, alors les modes télégraphiques et téléphoniques sont équivalents.

En effet, supposons qu'au cours d'une étape donnée dans un protocole de diffusion utilisant le mode téléphonique, un sommet x envoie le message à diffuser à y via le chemin $(x, u), (u, v), (v, y)$ et qu'un sommet z envoie le message à diffuser à w via le chemin $(z, v), (v, u), (u, w)$. Ces deux chemins sont bien arc-disjoints mais empruntent un même lien de communication dans les deux directions opposées, i.e. les arcs (u, v) et (v, u) sur l'exemple de la figure 3.1-(a), alors il est possible de transformer ce protocole pour le faire fonctionner en mode télégraphique, sans pour cela augmenter le nombre d'étapes, en établissant cette fois-ci des chemins arc-disjoints, i.e. les chemins (x, u), (u, w) et (z, v), (v, y) sur l'exemple de la figure 3.1-(b), n'empruntant jamais le même lien de communication. Dans l'exemple de la figure 3.1-(b), le sommet y sera alors informé par z et le sommet w par x.

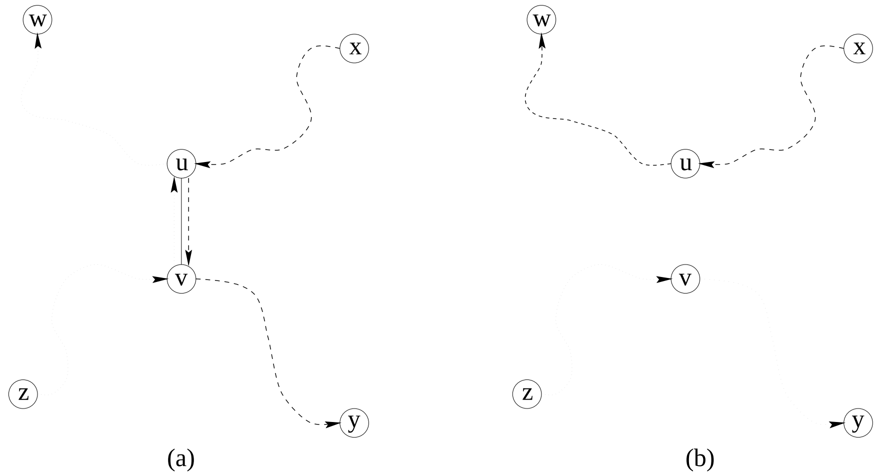

FIG. 3.1 – : *Equivalence des modes téléphoniques et télégraphiques pour le nombre d'étapes d'un protocole de diffusion n'autorisant pas le découpage du message.*

Clairement, cette transformation n'est plus valide si l'on cherche également à minimiser le maximum des longueurs des chemins utilisés à chaque étape.

Cette remarque apparaît nécessaire, car comme nous le verrons dans la suite, des études portant sur la diffusion (Cf. section 3.3.8-iii ou 3.5.3-i) adoptent ces hypothèses (pas de découpage du message à diffuser, longueur de chemins non prise en compte).

Par contre dans la section suivante, nous considérons au contraire des protocoles de diffusion pour lesquels nous autorisons le découpage des messages. Nous cherchons alors à étudier le comportement du paramètre $b_\tau(G)$ notamment lorsqu'un algorithme ne s'effectue pas en un nombre optimal d'étapes.

3.1.2 Compromis entre le nombre d'étapes et le flot d'information

Il apparaît souvent contradictoire de vouloir optimiser plusieurs paramètres simultanément, notamment le nombre d'étapes et le flot d'information. Ainsi, très souvent, les algorithmes utilisés pour un réseau et des contraintes physiques fixées, dépendent fortement des paramètres que l'on cherche à optimiser.

Souvent les auteurs proposent des algorithmes de diffusion optimaux pour le nombre d'étapes. Ces algorithmes sont efficaces lorsque l'on considère que la taille du message initial est petite ou lorsque l'on ne s'autorise pas à découper le message puisque jusqu'à présent le paramètre prépondérant dans les machines existantes est le temps d'initialisation α. Dans ce type d'algorithme, le flot d'information est souvent très élevé. Pour le diminuer, en s'autorisant le découpage du message en plusieurs paquets, les auteurs proposent des algorithmes du type pipeline ou arbres couvrants disjoints, utilisant souvent le mode commutation de paquets, qui diminuent toujours le flot d'information au détriment du nombre d'étapes. Même si généralement pour ce type de problème, les auteurs ne considèrent pas les temps nécessaires aux divers découpages, concaténations ou réorganisations des paquets en mémoire des routeurs, les algorithmes proposés restent néanmoins très efficaces pour de long messages. Pour exemple, citons les algorithmes pipelines ou arbres couvrants disjoints développés pour les grilles [3, 12] ou pour l'hypercube [11, 16, 31]. Pour de plus amples informations sur ce sujet, nous renvoyons le lecteur à l'article de synthèse [13] ou le livre [28].

Ainsi, il apparaît obligatoire de faire un compromis[1] entre ces deux paramètres. Pour étudier ce compromis, il faudrait établir une borne inférieure exprimant le nombre minimum d'étapes $b_\alpha(G)$ en fonction du flot d'information $b_\tau(G)$. Comme nous n'avons trouvé aucune référence traitant de cette question (sauf dans [4], où les auteurs établissent quelques relations sur le compromis nécessaire, mais dans le cas d'une opération de multidistribution), nous détaillons ici notre approche du problème.

Les algorithmes en mode commutation de circuits ou bien de messages obéissent en général à la règle suivante : l'optimalité en nombre d'étapes et l'optimalité en terme d'utilisation de la bande passante sont antinomiques. Ceci est dû au fait suivant. La bande passante utilisable par des messages issus du sommet émetteur croît de manière géométrique à partir de la date initiale d'émission. Afin de formaliser ce comportement nous considérons que le modèle de temps ne dépend que du temps d'initialisation α et de l'inverse de la bande passante des liens τ, i.e. nous ne prenons pas en compte δ. Ainsi, dans cette section le modèle de temps utilisé est :

Hypothèse 1 (Modèle de temps linéaire) — *Le temps nécessaire pour qu'un message de longueur L transite d'un sommet x vers un sommet y est :*

$$T_{x \to y} \;=\; \alpha + L\tau.$$

Dans la suite, nous étudions la diffusion en mode $k-$ports, sous notre modèle de temps, dans le graphe complet K_N et par souci de simplicité nous ferons l'hypothèse suivante :

Hypothèse 2 (Ordre du graphe complet) — *Le nombre de sommets du graphe complet K_N est $N = (k+1)^T$ avec T entier supérieur ou égal à 1.*

1. *"trade-off"* en américain.

Remarque. Comme, dans un tel graphe, tout couple de sommets peut communiquer, alors clairement les bornes inférieures qui y seront établies seront valides en commutation de circuits car on ne considère pas la longueur des chemins, ou bien en commutation de messages dans un graphe G quelconque de degré maximal $\Delta(G) = k$.

Remarque. Nous savons que le nombre minimal d'étapes nécessaire pour effectuer une diffusion $k-$ports dans le graphe complet avec $N = (k+1)^T$ est $b_\alpha(K_N) = T$.

A ce niveau nous introduisons la notation suivante :

Notation 4 — *Le nombre d'étapes optimal pour la diffusion sur le graphe complet K_N est T. Lorsqu'un protocole de diffusion sur K_N s'effectue en un nombre d'étapes strictement plus grand que l'optimal, nous dirons que ce protocole s'exécute en $T + r$ étapes, avec $r > 1$. Dans cette notation, r représente le surcoût en nombre d'étapes par rapport à l'optimalité.*

D'ores et déjà, à partir de notre modèle de temps, nous avons la propriété suivante.

Propriete. Le coût minimal d'une diffusion $k-$ports s'effectuant en exactement $T + r$ étapes dans le graphe complet est de la forme :

$$b_{F_k}^{\mathcal{R}_\star}(K_N) = (T + r)\alpha + f_T(r)L\tau$$

où $N = (k+1)^T$, $r \geq 0$ et $f_T(r)$ appelé coût de transmission, est une fonction ne dépendant que du surcoût en étapes par rapport à l'optimalité.

Preuve. Formellement nous devrions écrire le coût comme $b_{F_k}^{\mathcal{R}_\star}(K_N) \geq (T + r)\alpha + f_T(r, L)\tau$. Cependant, il est clair que $f_T(r, L)$ s'exprime simplement : $f_T(r)L$. En effet, le coût de transmission à une étape donnée des messages étant linéaire nous avons $f_T(r, \gamma L) = \gamma \cdot f_T(r, L)$.

En effet, supposons que l'on dispose d'un protocole de diffusion $\mathcal{P}_1$ pour un message de longueur L, utilisant un découpage en n blocs de longueur $L_1, L_2, \ldots,$

L_{n-1}, L_n et ayant un coût de tranmission total $\mathcal{C}_1$ (voir figure 3.2-a).

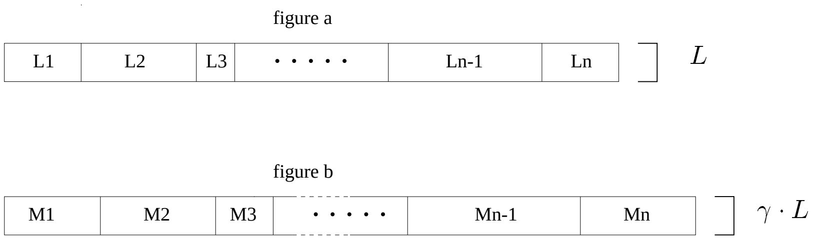

FIG. 3.2 – : *La figure-a représente un message de longueur L découpé en n blocs qui ne sont pas forcément de même taille. Idem pour la figure-b, mais avec un message de longueur $\gamma \cdot L$, où γ n'est pas forcément un entier.*

Alors nous pouvons déduire de $\mathcal{P}_1$ un protocole de diffusion $\mathcal{P}_2$ pour un message de longueur $\gamma \cdot L$ (avec γ un réel strictement positif) ayant un coût de transmission total $\mathcal{C}_2$. En effet, en utilisant un protocole similaire à $\mathcal{P}_1$ pour le message de longueur $\gamma \cdot L$ nous obtenons $\mathcal{C}_2 \leq \gamma \cdot \mathcal{C}_1$. Pour cela, il suffit de découper (voir figure 3.2-b) le message de longueur $\gamma \cdot L$ en n blocs de taille $M_1, \ldots, M_n$, avec $M_1 = \gamma \cdot L_1, \ldots, M_n = \gamma \cdot L_n$, alors à l'étape t au lieu d'envoyer un message de taille L_t, on envoie un message de taille $M_t = \gamma \cdot L_t$, le coût de transmission à cette étape est donc γ fois le coût de transmission de cette même étape dans le protocole $\mathcal{P}_1$, puisque par l'hypothèse 1 le coût de transmission à une étape donnée est linéaire. De même, en inversant le raisonnement, c'est-à-dire en utilisant sur le message de taille L un protocole cette fois-ci similaire à $\mathcal{P}_2$ (i.e. $\frac{M_1}{\gamma} = L_1, \ldots, \frac{M_n}{\gamma} = L_n$), où à l'étape t au lieu d'envoyer un message de taille M_t, on envoie un message de taille $L_t = \frac{M_t}{\gamma}$, alors nous obtenons toujours par l'hypothèse 1 de linéarité que $\mathcal{C}_1 \leq \frac{\mathcal{C}_2}{\gamma}$. Ces deux raisonnements impliquent l'égalité, $\mathcal{C}_1 = \gamma \cdot \mathcal{C}_2$ et donc la linéarité du coût de transmission. □

REMARQUE. Comme f_T est une fonction linéaire en L, nous pourrons sans perte de généralité nous restreindre à l'étude du cas $L = 1$.

Nous cherchons ici à évaluer $f_T(r)$, ce qui permet de comprendre dans quelle mesure l'ajout d'étapes supplémentaires au protocole implique une meilleure utilisation de la bande passante du réseau.

i) Diffusion avec $r = 0$

Dans ce cas la diffusion est optimale pour le nombre d'étapes.

Proposition 3.1.1 —

$$f_T(0) = T.$$

Preuve. La preuve est claire, en remarquant simplement que le nombre d'étapes étant optimal, chaque sommet n'a le temps de recevoir qu'une et une seule information. Pour que la diffusion soit effectuée, il est donc nécessaire de ne pas couper le message. Ainsi, à chaque étape la quantité d'information circulant sur un lien est de 1. □

Avant d'établir la proposition du compromis nécessaire en s'autorisant des étapes supplémentaires par rapport à l'optimalité, nous introduisons les définitions suivantes.

Definition.

- Nous noterons L_t, avec $1 \leq t \leq T+r$, la taille maximale des messages échangés au cours de l'étape t de l'algorithme de diffusion et $\mathcal{L}_t = \sum_{i=1}^{i=t} L_i$.
- Nous noterons I_t, avec $1 \leq t \leq T+r$, la quantité totale d'information (i.e. la somme des tailles de chaque message) circulant sur le réseau au cours de l'étape t; $\mathcal{A}_t$ désignera $\sum_{i=1}^{t} I_i$.

Notons que pour un algorithme optimal, $f_T(1) = \sum_{i=1}^{i=T+1} L_i$.

ii) Diffusion avec $r = 1$

Nous commençons par établir une borne supérieure.

Proposition 3.1.2 —

$$f_T(1) \leq \frac{1}{k+1}(T+1).$$

Preuve. Cette preuve est donnée par l'algorithme suivant :

Nous représentons l'ensemble des sommets du graphe complet par une matrice de $k+1$ lignes et $(k+1)^{T-1}$ colonnes.

$$V(K_N) = \begin{pmatrix} (1,1) & (1,2) & \cdots & (1,(k+1)^{T-1}) \\ (2,1) & (2,2) & \cdots & (2,(k+1)^{T-1}) \\ \vdots & \vdots & \ddots & \vdots \\ (k+1,1) & (k+1,2) & \cdots & (k+1,(k+1)^{T-1}) \end{pmatrix}$$

Le sommet $(1,1)$ est l'initiateur. Il découpe le message en $k+1$ paquets, notés m_i, avec $1 \leq i \leq k+1$. Chaque paquet à une taille égale à $\frac{1}{k+1}$. A partir de ce découpage, on applique l'algorithme suivant :

Algorithme 3.1.1 — **Algorithme de diffusion dans K_N**

- A la première étape, le sommet source $(1, 1)$ envoie en parallèle le message m_i au sommet $(i, 1)$ avec $2 \leq i \leq k + 1$. Le coût en bande passante est $\frac{1}{k+1}\tau$.
- Durant les $T - 1$ étapes suivantes, on effectue une diffusion en parallèle dans chaque ligne.

  Pour cela, chaque sommet $(i, 1)$ avec $1 \leq i \leq k+1$ diffuse (de façon optimale pour le nombre d'étapes) le message m_i sur les sommets (i, j) avec $2 \leq j \leq (k+1)^{T-1}$. Le coût en bande passante est $\frac{T-1}{k+1}\tau$.
- Enfin, durant la dernière étape, comme la totalité de l'information est répartie sur chaque colonne, on effectue un échange total entre les sommets d'une même colonne. Le coût en bande passante est $\frac{1}{k+1}\tau$.

Le coût global de cet algorithme est $(T + 1)\alpha + \frac{T+1}{k+1}\tau$.

□

Remarque. Nous verrons, en section 3.2.2-ii, un autre exemple sur la chaîne en mode 1−port qui atteint cette borne.

Nous établissons maintenant une borne inférieure.

Proposition 3.1.3 —

$$f_T(1) \;\; \geq \;\; \frac{N-1}{N(k+1)-1}(T+1).$$

Preuve. Tout d'abord, pour $\mathcal{L}_2 = (L_1 + L_2)$ fixé, nous cherchons à maximiser la quantité $\mathcal{A}_2 = I_1 + I_2$. Pour cela nous établissons les relations suivantes :

- A la première étape, le sommet initiateur peut envoyer des messages sur au plus k liens, ainsi $I_1 \leq kL_1$ (a);
- à la fin de l'étape 1, il y a au plus $k + 1$ sommets qui possèdent de l'information, donc $I_2 \leq k(k + 1)L_2$ (b);
- l'information transmise à l'étape 2 par les sommets autres que l'initiateur ne peut excéder kI_1, car chaque sommet ne peut renvoyer que k fois (i.e. sur k liens) les messages qu'il a déja reçus, or la quantité totale de messages reçue auparavant (par les sommets autres que l'émetteur) est I_1. Par ailleurs, le sommet émetteur transmet au plus kL_2 informations. Ainsi, $I_2 \leq kI_1 + kL_2$ (c).

Si on optimise la quantité $\mathcal{A}_2$ pour $\mathcal{L}_2$ fixé, on trouve la condition $L_1 = L_2 = \frac{\mathcal{L}_2}{2}$. En effet, les inégalités (a) et (b) donnent :

$$\begin{aligned} \mathcal{A}_2 \leq kL_1 + k(k+1)L_2 &= k(\mathcal{L}_2 - L_2) + k(k+1)L_2 \\ &= k\mathcal{L}_2 + k^2 L_2 \end{aligned}$$

D'autre part, les inégalités (a) et (c) donnent :

$$\mathcal{A}_2 \quad \leq \quad kL_1 + kI_1 + kL_2 \leq kL_1 + k^2L_1 + kL_2 = k\mathcal{L}_2 + k^2 L_1$$

Ainsi, $\mathcal{A}_2 \leq k\mathcal{L}_2 + k^2 \min(L_1, L_2)$. Or, $\min(L_1, L_2) \leq \frac{\mathcal{L}_2}{2}$ et la valeur maximale de $\mathcal{A}_2$ pour $\mathcal{L}_2$ fixé est $\frac{\mathcal{L}_2}{2}(k^2 + 2k)$ et ne peut être atteinte que si $L_1 = L_2$.

Par itération du même raisonnement, nous allons montrer que $\mathcal{A}_t$ avec $3 \leq t \leq T+1$ ne peut atteindre une valeur maximale, pour $\mathcal{L}_t$ fixé égale à $\frac{\mathcal{L}_t}{t}[(k+1)^t - 1]$ que si tous les L_i avec $1 \leq i \leq t$ sont égaux, valant $\frac{\mathcal{L}_t}{t}$. Nous venons de le montrer pour $t = 2$. Supposons maintenant la propriété vraie pour un t compris entre 2 et T et montrons qu'elle reste vraie pour $t+1$.

Pour les mêmes raisons que précédemment, nous avons :

- d'une part, $I_{t+1} \leq k\mathcal{A}_t + kL_{t+1}$ et
- d'autre part, $I_{t+1} \leq k(k+1)^t L_{t+1}$.

Comme $\mathcal{A}_{t+1} = \mathcal{A}_t + I_{t+1}$, ces deux inégalités donnent par hypothèse :

$$\begin{aligned} \mathcal{A}_{t+1} &\leq (k+1)\mathcal{A}_t + kL_{t+1} &&\leq \tfrac{\mathcal{L}_t}{t}(k+1)\left[(k+1)^t - 1\right] + kL_{t+1} \\ \mathcal{A}_{t+1} &\leq \mathcal{A}_t + k(k+1)^t L_{t+1} &&\leq \tfrac{\mathcal{L}_t}{t}\left[(k+1)^t - 1\right] + k(k+1)^t L_{t+1} \end{aligned}$$

En utilisant le fait que $\mathcal{L}_t = \mathcal{L}_{t+1} - L_{t+1}$ nous obtenons :

$$\begin{aligned} \mathcal{A}_{t+1} &\leq \tfrac{\mathcal{L}_{t+1}}{t}(k+1)\left[(k+1)^t - 1\right] + L_{t+1}\left[k - \tfrac{k+1}{t}\left[(k+1)^t - 1\right]\right] &&= g_1(L_{t+1}) \\ \mathcal{A}_{t+1} &\leq \tfrac{\mathcal{L}_{t+1}}{t}\left[(k+1)^t - 1\right] + L_{t+1}\left[k(k+1)^t - \tfrac{1}{t}\left[(k+1)^t - 1\right]\right] &&= g_2(L_{t+1}) \end{aligned}$$

$\mathcal{L}_{t+1}$ étant supposé fixé, g_1 et g_2 sont deux fonctions linéaires en L_{t+1}. Au vu des cœfficients respectifs de L_{t+1}, on observe que g_1 est décroissante en fonction de L_{t+1} alors que g_2 est croissante. Cette borne supérieure de $\mathcal{A}_{t+1}$ ne peut être atteinte que si $L_{t+1} = \frac{\mathcal{L}_{t+1}}{t+1}$ et si la valeur maximale de $\mathcal{A}_t$ pour $\mathcal{L}_t = \mathcal{L}_{t+1} - L_{t+1} = \frac{t\mathcal{L}_{t+1}}{t+1}$ est également atteinte, autrement dit si tous les L_i avec $1 \leq i \leq t+1$ sont égaux, ce qui achève la démonstration de l'itération.

Ainsi une valeur maximale de $\mathcal{A}_{T+1}$ pour $\mathcal{L}_{T+1}$ fixé est $\frac{\mathcal{L}_{T+1}}{T+1}[(k+1)^{T+1} - 1]$. D'autre part, comme tous les sommets du graphe, excepté l'initiateur, doivent recevoir l'information complète, on a $\mathcal{A}_{T+1} \geq N-1 = (k+1)^T - 1$. On obtient alors,

$$\frac{\mathcal{L}_{T+1}}{T+1} \geq \frac{N-1}{N(k+1)-1}$$

Ce qui montre que quel que soit l'algorithme,

$$\sum_{i=1}^{T+1} L_i \geq \frac{N-1}{N(k+1)-1}(T+1)$$

□

Ces deux dernières propositions conduisent à l'encadrement suivant.

Corollaire 1 —

$$\frac{N-1}{N(k+1)-1}(T+1) \leq f_T(1) \leq \frac{1}{k+1}(T+1).$$

Remarque. L'encadrement obtenu est presque exact. Notons aussi que notre système pourrait être affiné en utilisant le fait suivant: à l'étape $T+1$ nous avons utilisé la majoration : $I_{T+1} \leq (k(k+1)^T)L_{T+1} = kNL_{T+1}$ or il n'est pas possible d'utiliser k arcs entrants sur chaque noeud durant cette étape. En effet, ceux entrants sur le sommet initiateur sont inutiles. La contrainte exacte est donc $I_{T+1} \leq (k(k+1)^T - k)L_{T+1}$. Mais nous n'avons pas pu prouver que le deuxième type de contrainte sur I_{T+1} peut être affiné de manière analogue. Cependant si lors de notre optimisation nous estimions I_{T+1} par $(k(k+1)^T - k)L_0$ nous obtiendrions la borne exacte : $f_T(1) = (T+1)\frac{1}{k+1}$.

Afin d'obtenir une borne exacte, une idée est d'essayer d'affiner l'analyse comme suit. Si $\mathcal{L}_T$ est fixé nous savons que la transmission de message sera optimale jusqu'au temps T si $L_t = \frac{\mathcal{L}_T}{T} = L_1$ pour $t \leq T$. Cette transmission est alors égale à $((k+1)^T - 1)L_1 = (N-1)L_1$. Ceci implique qu'il existe au temps T au moins un sommet autre que l'émetteur qui n'a reçu qu'au plus L_1 informations. A la dernière étape ce sommet devra donc recevoir au moins $1 - L_1$ informations, ce qui implique $kL_{T+1} \geq 1 - L_1$. Le coût total en transmission de l'algorithme sera au moins:

$$f_T(1) \geq \frac{1-L_1}{k} + TL_1$$

Comme $f_T(1) \leq \frac{T+1}{k+1}$, on obtient $L_1 \leq \frac{1}{k+1}$ et $L_{T+1} \geq \frac{1}{k+1}$. De plus, un algorithme ayant un coût en transmission strictement inférieur à $\frac{T+1}{k+1}$ devrait nécessairement envoyer des messages de taille strictement inférieure à $\frac{1}{k+1}$ pendant les T premières étapes et au moins un message de taille strictement supérieure à $\frac{1}{k+1}$ lors de lors de la $(T+1)^{i\grave{e}me}$ étape. *Nous pensons qu'un tel algorithme n'existe pas et nous émettons la conjecture suivante.*

Conjecture 3.1.4 —

$$f_T(1) = \frac{T+1}{k+1}.$$

iii) Diffusion avec $r > 1$

Nous commençons par généraliser la borne supérieure initialement donnée pour le cas $r = 1$ en proposition 3.1.2.

Proposition 3.1.5 — *Pour $r \leq T$,*

$$f_T(r) \quad \leq \quad \frac{T-r}{(k+1)^r} \quad + \quad \frac{2}{k}\left(1 - \frac{1}{(k+1)^r}\right)$$

PREUVE. Nous définissons l'algorithme par induction sur r. Pour $r = 1$ le résultat est prouvé dans 3.1.2. Considérons les $N = (k+1)^T$ sommets comme une matrice de $k+1$ lignes et $(k+1)^{T-1}$ colonnes. A la première étape le message à diffuser est découpé en $k+1$ messages $m_0, m_1, \cdots, m_k$ de longueur $\frac{1}{k+1}$, le message m_i étant transmis au sommet de la $i^{\grave{e}me}$ ligne première colonne. Ensuite durant $T+r-2$ étapes, dans la ligne i , le sommet de la première colonne diffuse le message m_i aux noeuds de sa ligne. Durant cette phase dans chaque ligne il s'effectue une diffusion d'un message de longueur $\frac{1}{k+1}$ vers $(k+1)^{T-1}$ sommets en $T+r-2$ étapes. Par définition ce type de protocole aura un coût en bande passante de :

$$\frac{1}{k+1} \cdot f_{T-1}(r-1)$$

A la fin de cette phase il suffit d'effectuer un échange total dans chaque colonne, des messages m_i. Ceci s'effectue en une étape en échangeant des messages de longueur $\frac{1}{k+1}$.

Globalement nous avons montré que :

$$f_T(r) \quad \leq \quad \frac{2}{k+1} \quad + \quad \frac{f_{T-1}(r-1)}{k+1}$$

ce qui implique par itérations successives,

$$f_T(r) \quad \leq \quad 2\left(\frac{1}{k+1} + \frac{1}{(k+1)^2} + \cdots + \frac{1}{(k+1)^r}\right) \quad + \quad \frac{1}{(k+1)^r} \cdot f_{T-r}(0)$$

ce qui donne le résultat de la proposition. $\square$

Pour les bornes inférieures, nous émettons seulement une conjecture pour le cas $r = 2$. Dans ce cas, nous verrons que la borne inférieure donnée en conjecture 3.1.6 est de la même forme que celle de la borne supérieure obtenue à la proposition 3.1.5 lorsque $r = 2$, i.e. $f_T(2) \leq \frac{T}{(k+1)^2} + \frac{2}{k+1}$.

Conjecture 3.1.6 — *Pour k fixé et T grand, alors*

$$f_T(2) \quad \geq \quad \frac{T}{(k+1)^2} \quad + \quad \frac{2}{k+1} \quad - \quad \frac{k}{(k+1)^3} \quad + \quad \Theta\left(\frac{1}{T}\right)$$

Nous donnons ci-dessous un certains nombre d'arguments en faveur de cette conjecture et nous ramenons le problème à un problème d'optimisation convexe que nous espérons résoudre ultérieurement.

Afin de généraliser le résultat précédent nous devons affiner quelque peu l'analyse. Pour cela nous allons considérer deux type de paquets :

- ceux qui transportent de l'information émise par l'initiateur lors de la première étape, et
- ceux qui transportent ce qu'il reste de l'information initiale.

Remarquons alors que la longueur du message qui n'a pas quitté le sommet initial après la première étape est au moins $\max(1 - kL_1, 0)$. Nous effectuons deux estimations opposées qui permettent de déterminer assez précisément le comportement de $f_T(2)$.

a) Tout d'abord notons que L_1 ne doit pas être trop faible. En effet, après la première étape il reste à diffuser $1-kL_1$ informations en $T+1$ étapes. Or la valeur minimale de ce coût est donné en proposition 3.1.3. Comme nous travaillons à k fixé et T variant pour des valeurs relativement grandes nous avons :

$$\frac{N-1}{N(k+1)-1} \quad = \quad \frac{(k+1)^T - 1}{(k+1)^{T+1} - 1} \quad = \quad \frac{1}{k+1} \quad + \quad \Theta\left(\frac{1}{(k+1)^{T+1}}\right) \tag{3.1}$$

Soit au total :

$$f_T(2) \quad \geq \quad L_1 \quad + \quad (T+1) \cdot \frac{1-kL_1}{k+1}\left(1 + \Theta\left(\frac{1}{(k+1)^{T+1}}\right)\right) \tag{3.2}$$

Cette fonction de L_1 étant décroissante, l'estimation montre que L_1 doit être relativement grand.

b) En revanche, la première étape transmet très peu de messages. Celle-ci ne doit donc pas avoir un coût excessif, ce qui revient à dire que L_1 ne doit pas être trop grand.

Nous considérons $\mathcal{A}_2$ fixé, et cherchons à maximiser la valeur $\mathcal{A}_{T+2}$ pour un coût $\mathcal{L}_t = \sum_{i=1}^{i=t} L_i$ supposé également fixé. Remarquons alors qu'à l'étape t la quantité maximale de messages qui peut être transmise est majorée par les deux estimations suivantes :

$$\begin{aligned} I_t &\leq k\mathcal{A}_{t-1} + kL_t \\ I_t &\leq N_{t-1}kL_t \end{aligned}$$

où N_{t-1} désigne le nombre maximal de sommets possèdant de l'information au temps $t-1$.

A ce niveau, nous émettons la conjecture suivante qui se ramène à un problème d'optimisation convexe :

En utilisant des outils de programmation linéaire convexe, on peut montrer que les points extrémaux sont atteints lorsque les valeurs de L_t vérifient :

$$k\mathcal{A}_{t-1} + kL_t = N_{t-1}kL_t$$

Ce qui implique

$$L_t = \frac{\mathcal{A}_{t-1}}{N_{t-1} - 1} \tag{3.3}$$

`b-1) Cas` $t < T + 2$

Pour $2 \leq t < T + 2$, N_{t-1} vaut $(k+1)^{t-1}$. La solution optimale est donc caractérisée par :

$$L_t = \frac{\mathcal{A}_{t-1}}{(k+1)^{t-1} - 1} \tag{3.4}$$

$$I_t = k(k+1)^{t-1}L_t \tag{3.5}$$

Nous montrons alors par itérations successives que pour $2 < t < T + 2$,

$$\mathcal{A}_t = ((k+1)^t - 1)A = k\sum_{i=0}^{t-1}(k+1)^i A \tag{3.6}$$

où A est déterminé par la condition initale

$$\mathcal{A}_2 = ((k+1)^2 - 1)A \tag{3.7}$$

En effet si l'hypothèse est vraie pour t nous trouvons $L_{t+1} = \frac{\mathcal{A}_t}{(k+1)^t - 1} = A$. Il vient alors $I_{t+1} = k(k+1)^t A$ et $\mathcal{A}_{t+1} = \mathcal{A}_t + I_{t+1} = k\sum_{i=0}^{t-1}(k+1)^i + k(k+1)^t = ((k+1)^{t+1} - 1)A$.

b-2) Cas $t = T+2$

Le système est un peu différent lorsque $t = T+2$, puisque cette fois-ci le nombre de sommets possédant de l'information au temps $T+1$ est $N_{T+1} = N = (k+1)^T$ et non pas $(k+1)^{T+1}$ comme lors des étapes précédentes, car il est impossible d'informer plus de sommets que l'ordre du graphe.

A partir des relations (3.3) et (3.6) nous déduisons une relation sur L_{T+2}, et la relation sur I_{T+2} se déduit de (3.5).

$$L_{T+2} = \frac{(k+1)^{T+1}-1}{(k+1)^T-1}A \tag{3.8}$$

$$I_{T+2} = k(k+1)^T L_{T+2} \tag{3.9}$$

Comme $\mathcal{A}_{T+2} = \mathcal{A}_{T+1} + I_{T+2}$, à partir de (3.6), (3.8) et (3.9) nous obtenons,

$$\mathcal{A}_{T+2} = A((k+1)^{T+1}-1)\cdot\left(1+\frac{k(k+1)^T}{(k+1)^T-1}\right)$$

Comme $\mathcal{A}_{T+2} \geq (k+1)^T - 1$ c'est-à-dire $\mathcal{A}_{T+2} \geq N-1$, nous déduisons une minoration de A :

$$A \geq \left(\frac{(k+1)^T-1}{(k+1)^{T+1}-1}\right)^2 \tag{3.10}$$

$$\text{soit,}\quad A \geq \frac{1}{(k+1)^2} + \Theta\left(\frac{1}{(k+1)^{2T+4}}\right) \tag{3.11}$$

Concernant le coût de ce protocole optimisant la transmission des messages nous avons grâce aux relations (3.4) et (3.6) :

$$f_T(2) = L_1 + L_2 + (T-1)A + L_{T+2}$$

En utilisant la relation (3.8) et le fait que $((k+1)^2-1)A = \mathcal{A}_2 \leq kL_1 + k(k+1)L_2$, nous trouvons une minoration sur L_{T+2}. Quant à la relation sur L_2, elle se déduit directement de (3.7).

$$L_{T+2} \geq \frac{(k+1)^T-1}{(k+1)^{T+1}-1}$$

$$L_2 \geq (1+\frac{1}{k+1})A - \frac{L_1}{k+1}$$

Nous obtenons le coût du meilleur algorithme transmettant suffisamment de messages, et utilisant à la première étape des messages de longueur L_1 :

$$f_T(2) = \frac{k}{k+1}L_1 + \frac{A}{k+1} + TA + L_{T+2}$$

$$f_T(2) = \frac{k}{k+1}L_1 + \frac{1}{(k+1)^3} + \frac{T}{(k+1)^2} + \frac{1}{k+1} + \Theta\left(\frac{1}{(k+1)^{T+1}}\right) \tag{3.12}$$

c) Nous avons donc obtenu deux estimations du coût, l'une donnée par l'équation (3.2) et l'autre par l'équation (3.12). Comme l'une des fonction est croissante tandis que l'autre est décroissante, nous pouvons minorer le coût par la valeur commune de ces fonctions en leur intersection. Ceci donne :

$$L_1 \;=\; \frac{1}{k+1} - \frac{k^2}{((T+1)k-1)(1+k)^2} + \Theta\left(\frac{T}{(k+1)^{T+1}}\right)$$

En réinjectant cette valeur de L_1 dans l'équation (3.12), nous obtenons :

$$\begin{aligned} f_T(2) \;\geq\; & \frac{k}{(k+1)^2} + \frac{T}{(k+1)^2} + \frac{1}{k+1} + \frac{1}{(k+1)^3} \\ & - \frac{k^2}{(k+1)^3(T+1)k-1} + \Theta\left(\frac{T}{(k+1)^{T+1}}\right) \end{aligned}$$

soit,

$$\begin{aligned} f_T(2) \;\geq\; & \frac{T}{(k+1)^2} + \frac{2}{k+1} - \frac{1}{(k+1)^2} + \frac{1}{(k+1)^3} \\ & - \frac{k^2}{(k+1)^3(T+1)k-1} + \Theta\left(\frac{T}{(k+1)^{T+1}}\right) \end{aligned}$$

ce qui donne finalement,

$$f_T(2) \;\geq\; \frac{T}{(k+1)^2} + \frac{2}{k+1} - \frac{1}{(k+1)^2} + \frac{1}{(k+1)^3} + \Theta\left(\frac{1}{T}\right)$$

Remarque. Lorsque T est grand et $r < T$ nous avons vu apparaitre un phénomème de décroissance exponentielle de $f_T(r)$ lorsque r croît, de fait $f_T(r)$ est de la forme $\frac{T}{(k+1)^r}$. En revanche quand r est supérieur à T un phénomène de type pipeline semble apparaître.

3.2 La diffusion sans fonction de routage imposée dans le modèle 1−port

3.2.1 Bornes inférieures générales dans le modèle 1−port

Signalons que le nombre d'étapes dans le modèle 1-port est résolu puisque dans [5] les auteurs montrent principalement qu'il est non seulement possible de diffuser en $\lceil \log_2 N \rceil$ étapes (résultat prouvé par Farley [9]), mais aussi que toutes les communications réalisées à chaque étape de la diffusion peuvent s'effectuer sur des plus courts chemins. Ce résultat s'exprime comme le théorème suivant.

Théorème 3.2.1 ([5]) — *Pour tout graphe $G = (V, E)$ de N sommets et pour tout sommet u de G, on peut construire en temps polynômial un schéma de diffusion à partir du sommet u en $\lceil \log_2 N \rceil$ étapes tel que*

1. *toutes les communications s'effectuent sur des plus courts chemins;*
2. *à chaque étape, la somme des longueurs des chemins utilisés est minimum.*

☞ Notons que, dans la preuve de ce théorème, le schéma de diffusion à partir du sommet u ne minimise pas nécessairement la somme pour i allant de 1 à $\lceil \log_2 N \rceil$ de la somme des longueurs des chemins utilisés à l'étape i. En fait, il est montré dans [5] que minimiser ce paramètre est un problème $NP-$complet.

Les bornes inférieures sur les autres paramètres de la diffusion sont données par la proposition suivante.

Proposition 3.2.2 — *Soit un graphe G d'ordre N et de diamètre D, alors sous la contrainte $1-$ports, les deux autres paramètres de la diffusion sont minorés par :*

$$b_\delta(G) \geq \max(D, \log_2 N), \quad \textit{et } b_\tau(G) \geq L$$

Ceci conduit à une minoration du coût globale de la diffusion.

Proposition 3.2.3 — *Dans un graphe G d'ordre N et de diamètre D, le coût de la diffusion sous la contrainte $1-$ports, est minorée par :*

$$b_{F_1}^{\mathcal{R}_\star}(G) \geq \lceil \log_2 N \rceil \alpha + \max(D, \log_2 N)\delta + L\tau$$

Ainsi, la diffusion $1-$port apparaît être particuliere puisque la recherche de l'optimalité sur le nombre d'étapes est désormais un problème clos. Il reste cependant, pour un nombre d'étapes optimal, à construire des algorithmes minimisant la longueur des chemins utilisés. En effet, si on considère le graphe en étoile avec $N = n + 1$ sommets qui est de diamètre 2, alors tout protocole de diffusion en $\lceil \log_2 N \rceil$ étapes a un cœfficient en δ d'au moins 1 à la première étape et 2 à chaque autre étape, donc au moins $2\lceil \log_2 N \rceil - 1$. Si le réseau est assez dense, en particulier s'il contient un hypercube $H(n)$, alors le protocole classique utilisant la commutation de messages (i.e. des chemins de longueur 1 à chaque étapes), permet d'obtenir un coefficient $b_\delta(H(n))$ optimal puisque nous savons que dans un tel graphe le coût est :

$b_{F_1}^{\mathcal{R}_\star}(H(n))$		
b_α	b_δ	b_τ/L
$\log_2 N$	$\log_2 N$	b_α

Pour le flot d'information, nous avons vu, en section 3.1.2, que l'amélioration du ceofficient en τ nécessitait une augmentation du nombre d'étapes et l'utilisation de techniques du type pipeline ou arbres couvrants disjoints. Dans la suite, nous donnons les protocoles étudiés dans la littérature pour des réseaux réguliers et peu denses : chaîne, cycle, grille, tore.

3.2.2 La chaîne dans le modèle F_1

i) Protocole optimal en nombre d'étapes

Nous redonnons ici un protocol trivial et optimal sur la chaîne. Montrons par induction que l'on peut en $\lceil \log_2 N \rceil$ étapes réaliser un protocole de diffusion sur la chaîne P_N vérifiant $b_\delta(G) = N - 1 = D$.

Tout d'abord découpons la chaîne P_N en 2 parties, l'une de cardinalité $\left\lceil \frac{N}{2} \right\rceil$, l'autre de $\left\lfloor \frac{N}{2} \right\rfloor$. Alors, l'initiateur envoie son message au sommet, dans l'autre partie, qui est le plus proche de lui, donc à distance inférieure ou égale à $\left\lfloor \frac{N}{2} \right\rfloor$. Ensuite, chacun des deux sommets ayant l'information diffuse, de façon disjointe dans sa partie, son message en utilisant le même algorithme récursivement. Ainsi, deux diffusions disjointes s'effectueront sur chacune des deux chaînes, l'une de cardinalité $\left\lceil \frac{N}{2} \right\rceil$, l'autre de $\left\lfloor \frac{N}{2} \right\rfloor$. La figure 3.3 fournit un exemple d'une telle diffusion dans la chaîne à 8 sommets.

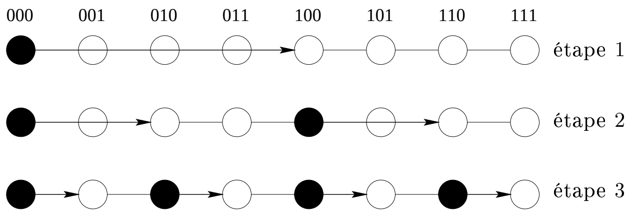

FIG. 3.3 – : *Diffusion dans une chaîne de 8 sommets, à partir du sommet 0 qui est le cas le plus défavorable pour la longueur des chemins.*

L'équation de récurrence qui en découle est :

$$b_\delta(P_N) \;\leq\; \left\lfloor \frac{N}{2} \right\rfloor + b_\delta(P_{\lceil \frac{N}{2} \rceil})$$

Donc par induction pour $N' < N$, on a $b_\delta(P'_N) = N' - 1$, soit

$$\begin{aligned} b_\delta(P_N) &\leq \left\lfloor \frac{N}{2} \right\rfloor + \left\lceil \frac{N}{2} \right\rceil - 1 \\ &= N - 1 \end{aligned}$$

Soit un coût global de :

$b_{F_1}^{\mathcal{R}_\star}(P_N)$		
b_α	b_δ	b_τ/L
$\log_2 N$	$N-1=D$	b_α

REMARQUE. Ce protocole est aussi décrit dans [1] au moins pour N puissance de 2, car les auteurs le réutilisent pour la grille.

ii) Protocole non optimal en nombre d'étapes

Seidel avait déjà utilisé en partie le fait que le mode de commutation est peu sensible à la distance des chemins. Il propose dans [29] un algorithme de diffusion bidirectionnel, qui ne cherche pas à obtenir l'optimalité sur le nombre d'étapes mais à diminuer de moitié le facteur proportionnel à la longueur du message L, en s'autorisant une étape supplémentaire par rapport à l'optimalité.

L'idée est de diffuser non plus sur un seul arbre couvrant, mais sur deux arbres couvrants. La source commence par envoyer dans un premier temps la moitié du message à la racine du second arbre. Les deux racines diffusent ensuite leur moitié de message (voir figure 3.4).

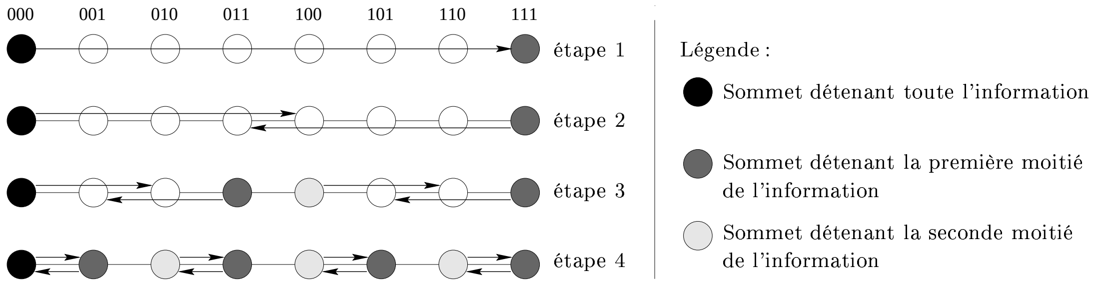

FIG. 3.4 – : *Diffusion dans une chaîne de 8 sommets, à partir du sommet 0 qui est le cas le plus dévavorable pour la longueur des chemins. L'algorithme suit le plongement dynamique de deux arbres recouvrants.*

Le temps de cet algorithme est $b_{F_1}^{\mathcal{R}_\star}(P_{2^i}) \leq \alpha + \delta(2^i-1) + \frac{L}{2}\tau + \sum_{j=1}^{i}\left(2^{i-j}\delta + \alpha + \frac{L}{2}\tau\right)$, soit :

$b_{F_1}^{\mathcal{R}_\star}(P_N)$ avec $N=2^i$		
b_α	b_δ	b_τ/L
$1+\log_2 N = 1+i$	$2D$	$\frac{b_\alpha}{2}$

Seidel montre que cet algorithme est sans conflit et qu'il est valide quel que soit le sommet initiateur.

Remarque. D'après la proposition 3.1.3, ce protocole est optimal pour le coefficient b_τ/L avec $k = 1$.

3.2.3 Le cycle dans le modèle F_1

Fraigniaud et Peters ont étudié [14] la diffusion dans les cycles C_N, avec $N = 2^i$. Le coût de leur protocole est :

$b_{F_1}^{\mathcal{R}_\star}(C_N)$ avec $N = 2^i$		
b_α	b_δ	b_τ/L
$\log_2 N = i$	D	b_α

A la première étape, l'émetteur envoie le message à un de ses voisins. A l'étape j, $2 \le j \le \log_2 N$, chaque sommet informé envoie un message à un processeur à distance $\frac{N}{2^j}$.

Remarque. On peut aisément étendre ce résultat en utilisant le protocole pour la chaîne (Cf. section 3.2.2-i). A la première étape l'initiateur envoie son message à l'un de ses voisins directs. Puis, chacun des deux sommets effectue simultanément le protocole vu pour la chaîne qui est de longueur au plus $\left\lceil \frac{N}{2} \right\rceil$. Ce protocole utilise $\lceil \log_2 N \rceil$ étapes avec un coefficient $b_\delta(C_N) = \left\lceil \frac{N}{2} \right\rceil$ qui est le meilleur possible et égal au diamètre lorsque N est pair. En effet, si à la première étape, l'émetteur (représenté par le sommet 0) informe le sommet j, il faudra pour les autres étapes informer le sommet $\frac{N-j+1}{2}$ donc en tout un coefficient $b_\delta(C_N) \ge \left\lceil \frac{N-j+1}{2} \right\rceil + j$, soit $b_\delta(C_N) \ge \left\lceil \frac{N}{2} \right\rceil$. Le minimum est atteint lorsque $j = 1$.

3.2.4 Le tore de dimension k dans le modèle F_1

Fraignaud et Peters généralisent leur résultat sur le cycle (Cf. section 3.2.3), au cas du tore de dimension k. Pour cela, ils appliquent directement l'algorithme sur les cycles dimension après dimension.

Ce résultat s'étend aisément, en utilisant le résultat sur le cycle (Cf. section 3.2.3), au cas du tore de coté pair. Le coût de ce protocole est :

$b_{F_1}^{\mathcal{R}_\star}(TM(2p)^k)$		
b_α	b_δ	b_τ/L
$\lceil \log_2 N \rceil$	D	b_α

Cet algorithme optimal pour le nombre d'étapes, l'est également pour la longueur des chemins utilisés.

3.2.5 La grille de dimension k dans le mode F_1

i) Protocole optimal pour le nombre d'étapes

Barnette, Payne et van de Geijn généralisent très facilement leur algorithme sur la chaîne (Cf. section 3.2.2-i), au cas des grilles $M(2^{k_1}, \ldots, 2^{k_n}) = P_{2^{k_1}} \Box P_{2^{k_2}} \Box \ldots \Box P_{2^{k_n}}$, de dimension n. Il suffit d'appliquer l'algorithme de diffusion dans une chaîne, dimension par dimension. Le sommet source va dans un premier temps informer tous les sommets de sa ligne, puis dans un deuxième temps, tous les sommets de la ligne informée vont diffuser l'information dans les colonnes, et ainsi de suite. On obtient comme temps de diffusion :

$$\begin{aligned} b_{F_1}^{\mathcal{R}\star}(M(2^{k_1}, \ldots, 2^{k_n})) &\leq \sum_{j=1}^{n} \sum_{i=1}^{k_j} \left(\alpha + 2^{k_j - i}\delta + L\tau\right) \\ &\leq \alpha \sum_{j=1}^{n} k_j + \delta \sum_{j=1}^{n} (2^{k_j} - 1) + L\tau \sum_{j=1}^{n} k_j \end{aligned}$$

Soit un coût de :

$b_{F_1}^{\mathcal{R}\star}(M(2^{k_1}, \ldots, 2^{k_n}))$		
b_α	b_δ	b_τ / L
$\log_2 N$	D	b_α

REMARQUE. L'algorithme présenté ici, valable pour tout sommet emetteur, est optimal à la fois sur le nombre d'étapes et la longueur des chemins utilisés.

☞ Contrairement à l'algorithme développé pour les chaînes, cette généralisation aux grilles $M(p_1, \ldots, p_n)$ de dimension n n'est pas optimale en nombre d'étapes lorsque le nombre de sommets dans chaque dimension n'est pas une puissance de deux, puisqu'elle requiert $\sum_{i=1}^{n} \lceil \log_2 p_i \rceil$ étapes, ce qui, dans le cas général, est supérieur à la borne inférieure égale à $\lceil \log_2 N \rceil$ étapes (d'un facteur additif au plus égal à $n - 1$).

Nous avons donné ici un protocole optimal pour le nombre d'étapes. Nous décrivons dans les deux points suivants des algorithmes non optimaux pour le nombre d'étapes, mais diminuant le flot d'information.

ii) Algorithme de Seidel

Dans [29], Seidel généralise son algorithme sur les chaînes (Cf. section 3.2.2-ii), aux grilles de dimension 2, avec 2^{k_i} sommets par dimension, dans le cas où l'initiateur est le sommet $(0, 0)$. Il ne cherche pas à obtenir un nombre d'étapes minimum, mais à diminuer le flot d'information en s'autorisant 2 ou 3 étapes supplémentaires par rapport à l'optimalité.

Pour cela, il considère que la grille $M(2^{k_1}, 2^{k_2})$ est constituée de 4 grilles $M(2^{k_1-1}, 2^{k_2-1})$ entrelacées (voir la grille de gauche dans la phase d'échange de la figure 3.5 où chacune des sous-grilles est représentée avec une couleur différente Q_1, Q_2, Q_3 et Q_4). Une première phase de 2 étapes consiste à envoyer une partie du message (de taille $\frac{L}{4}$) à un sommet de chaque sous-grille (`phase de distribution`). On utilise ensuite l'algorithme de l'arbre couvrant sur chacune des sous-grilles (`phase de diffusion`), puis une phase d'échange entre processeurs voisins suivant les 2 dimensions (`phase d'échange`). La figure 3.5 présente l'algorithme sur une grille 4×4.

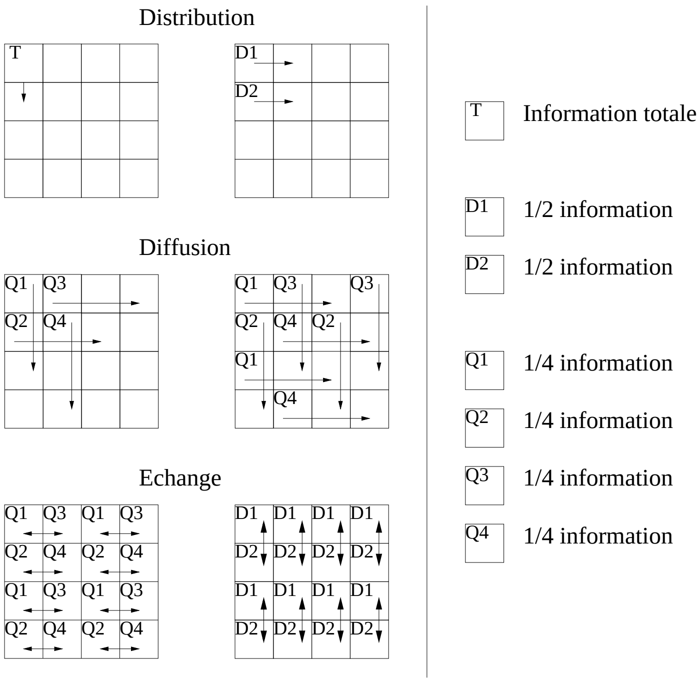

FIG. 3.5 – : *Diffusion suivant le plongement dynamique d'un arbre de recouvrement sur une grille $M(4, 4)$.*

Si l'on note $k = \max(k_1, k_2)$, le temps de cet algorithme est :

$$\begin{aligned} b_{F_1}^{\mathcal{R}_\star}((0,0), M(2^{k_1}, 2^{k_2})) &\leq 2\sum_{i=1}^{2}\left(\alpha+\delta+\frac{L}{2^i}\tau\right) + 2\sum_{i=1}^{k-1}\left(\alpha+\frac{\tau}{4}L\right) + \sum_{j=1}^{2}\sum_{i=1}^{k_j} 2^{k_j-i}\delta \\ &\leq (2+2k)\,\alpha + (2+\sum_{j=1}^{2}2^{k_j})\delta + \left(\frac{2\,(k-1)-2}{4}+2\right)L\tau \end{aligned}$$

Si $k_1 = k_2 = k$, le coût est :

$b_{F_1}^{\mathcal{R}_\star}((0,0), M(2^k)^2)$		
b_α	b_δ	b_τ/L
$2+\log_2 N$	$D+4$	$\frac{b_\alpha+2}{4} = 1+\frac{\log_2 N}{4}$

Seidel [29] améliore l'algorithme précédent en s'inspirant de l'algorithme de diffusion sur un arbre recouvrant bidirectionnel utilisé sur le chemin. Le message est découpé en 8 paquets. Le message est, lors d'une première phase, distribué parmi les 8 sommets de deux carrés de quatre sommets (l'un en haut à gauche contenant la source $(0,0)$ et l'autre en bas à droite contenant le sommet diamétralement opposé à la source). A la fin de cette première phase, chaque sommet détient un paquet de taille $\frac{L}{8}$. Ensuite, les 8 sommets diffusent la partie du message qu'ils détiennent sur 8 sous grilles $M(2^{k_1-1}, 2^{k_2-1})$ entrelacées de façon similaire à l'algorithme précédent. À la fin de cette phase, il suffit de faire un échange entre sommets voisins suivant chaque dimension pour que chaque sommet prenne connaissance de la totalité du message. La figure 3.6 illustre cet algorithme sur une grille 8×8.

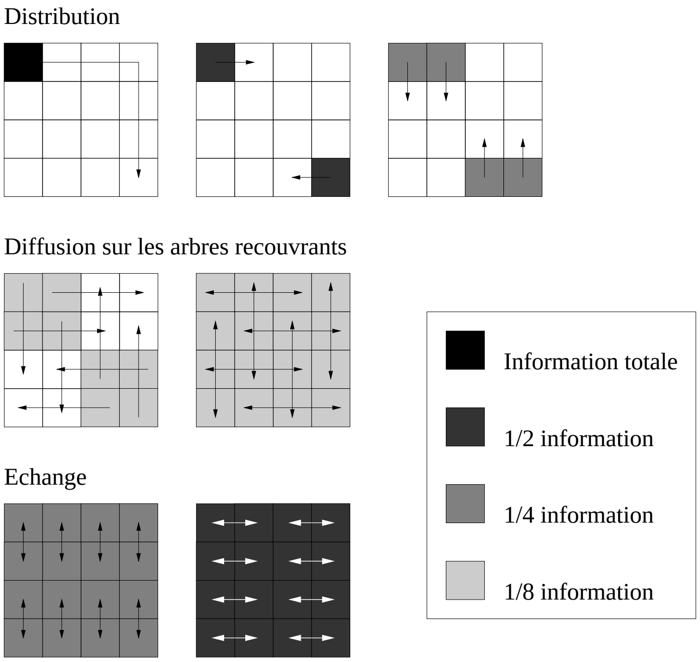

FIG. 3.6 – : *Diffusion suivant le plongement dynamique de deux arbres de recouvrement sur une grille* $M(4,4)$.

Si l'on note $k = \max(k_1, k_2)$. En appliquant cet algorithme on obtient :

$$\begin{aligned} b_{F_1}^{\mathcal{R}_\star}((0,0), M(2^{k_1}, 2^{k_2})) \quad \leq \quad & \alpha + D\delta + \frac{L}{2}\tau + \sum_{i=1}^{2}\left(\alpha + \delta + \frac{L}{2^{i+1}}\tau\right) \\ & + \sum_{i=0}^{1}\left(\alpha + \delta + \frac{L}{2^{2-i}}\tau\right) \\ & + 2\sum_{i=1}^{k-1}\left(\alpha + \frac{L}{8}\tau\right) + \sum_{j=1}^{2}\sum_{i=1}^{k_j-1} 2^{k-i}\delta \end{aligned}$$

$$\leq \quad (2k+3)\,\alpha + (2D+2)\delta + \left(\frac{2k+3}{8}+1\right) L\tau$$

Si $k_1 = k_2 = k$, le coût est :

$b_{F_1}^{\mathcal{R}_\star}((0,0), M(2^k)^2)$		
b_α	b_δ	b_τ/L
$3+\log_2 N$	$2(D+1)$	$1+\frac{b_\alpha}{8}$

Les algorithmes présentés par Seidel permettent, en fonction de la taille des messages et des différents paramètres, d'ajuster le coût d'une diffusion. Cependant, ces algorithmes ne sont présentés que dans le cas où le sommet initiateur est le sommet $(0,0)$. Seidel conjecture dans [29] qu'ils restent valides pour tout sommet initiateur.

☞ E. Fleury conjecture [10] que le premier algorithme de Seidel, présenté ici pour la grille de dimension 2, peut se généraliser à des grilles de dimension n, notamment dans le cas où la source est le sommet $(0,\ldots,0)$. On applique la même stratégie mais en envoyant les messages de façon cyclique suivant les dimensions. Quant au second algorithme, il pense également qu'il peut se généraliser. La source (sommet $(0,\ldots,0)$) envoie lors de la première phase la moitié du message au sommet qui lui est antipodal. Puis ces deux sommets distribuent le message dans les hypercubes de dimension n auquel ils appartiennent. La diffusion se fait de façon similaire et de façon cyclique suivant les n dimensions.

iii) Algorithme de Barnett, Payne, van de Geijn et Watts

Notons, pour finir, l'algorithme basé sur une méthode de pipeline (utilisant des techniques classiques de la commutation de messages) suivant le plongement dynamique de deux arbres couvrants proposé par Barnett, Payne, van de Geijn et Watts [2, 37]. Les sommets de la grille $M(l_1, l_2)$ sont considérés comme étant de couleur blanche ou noire (comme sur un échiquier). Les sommets noirs transmettent l'information vers l'Est lors des étapes paires, et vers le Sud lors des étapes impaires, tandis que les sommets blancs transmettent l'information vers le Sud lors des étapes paires, et vers l'Est lors des étapes impaires. Pour que tout sommet puisse jouer le rôle de la source, la grille est rebouclée de façon logique pour former un tore (*i.e.*, des messages circulent vers l'Ouest et vers le Nord). La figure 3.7 illustre cet algorithme.

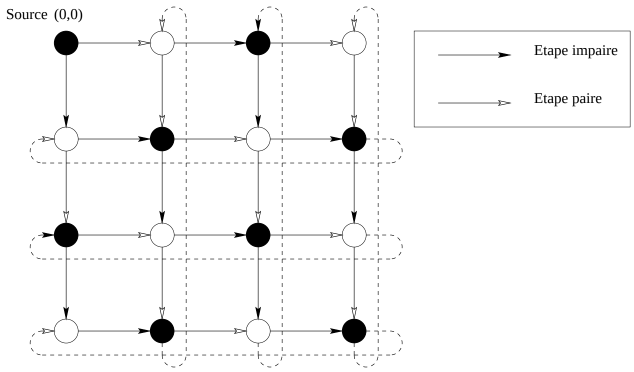

FIG. 3.7 – : *Plongement dynamique de deux arbres recouvrants arc-disjoints dans la grille $M(4)^2$.*

En appliquant cet algorithme, et en supposant que le message de longueur L est découpé en p paquets de même longueur, on obtient :

$$b_{F_1}(M(l_1, l_2)) \quad \leq \quad (p + l_1 + l_2)(\beta + \frac{L}{p}\tau)$$

où $\beta = \alpha + (\max(l_1, l_2) - 1)\delta$. Ainsi, le coût est :

$b_{F_1}^{\mathcal{R}_\star}(M(l_1, l_2))$		
b_α	b_δ	b_τ/L
$p + l_1 + l_2$	$(p + l_1 + l_2)(\max(p1, p2) - 1)$	$\frac{b_\alpha}{p} = 1 + \frac{l_1+l_2}{p}$

REMARQUE. Cet algorithme se comporte en $L\tau$ pour le flot d'information, uniquement lorsque le rapport $\frac{l_1+l_2}{p}$ est négligeable devant 1. Clairement cela signifie que pour l_1 et l_2 fixés, il faut avoir $p >> l_1 + l_2$, ce qui implique une augmentation non négligeable du nombre d'étapes de l'algorithme.

3.3 La diffusion sans fonction de routage imposée dans le modèle k et $\Delta-$ports

3.3.1 Bornes inférieure générales dans le modèle $k-$ports

Nous donnons ici, des bornes inférieures génériques au sens ou elle ne dépende pas de la topologie, mais seulement des contraintes physiques des liens et routeurs.

Proposition 3.3.1 — *Soit un graphe G d'ordre N de degré Δ, de diamètre D et d'arête connectivité λ, alors sous la contrainte $k-$ports, les trois paramètres de la diffusion sont minorés par :*

$$b_\alpha(G) \geq \left\lceil \log_{k+1} N \right\rceil, \quad b_\delta(G) \geq D, \quad et\ b_\tau(G) \geq \frac{L}{\min(\lambda,k)}$$

Une preuve de cette proposition est développée, pour le cas $k = \Delta$, dans [26, 28]. Il est aisé de généraliser ces preuves pour $1 \leq k \leq \Delta$.

3.3.2 Diffusion dans le modèle $\boldsymbol{\Delta}-$ports - Méthodologie générale

Au vu des résultats existants sur la diffusion $\Delta-$ports, il semble ressortir une méthodologie générale que nous tentons de synthétiser dans cette section. En effet, comme nous le verrons dans la suite, bien que de nombreux auteurs utilisent généralement des notations et définitions différentes, très souvent leur méthode de diffusion utilise un même principe commun qui est très proche de celui que nous décrivons dans [8]. Cette méthodologie générale est la suivante.

Nous considérons un protocole de diffusion s'effectuant en T étapes.

Notation 5 — $\boldsymbol{S_t}$ *représente l'ensemble des sommets qui connaissent l'information* **après l'étape $\boldsymbol{t}$**.

Nous définissons ensuite quelques propriétés sur ces ensembles.

Propriétés 1 — Soit G un graphe de degré Δ. Alors, pour $0 \leq t < T$,

- S_0 est le sommet émetteur,
- S_T est l'ensemble de tous les sommets du graphe,
- $S_t \subset S_{t+1}$,
- S_t peut prévenir S_{t+1} en une étape le long de chemins arc-disjoints.
- $|S_{t+1}| \leq \Delta \cdot |S_t|$,

Si on veut réaliser un protocole atteignant la borne inférieure T dans un graphe d'ordre $N = (\Delta + 1)^T$, il faudra avoir la propriété $|S_{t+1}| = \Delta \cdot |S_t|$. Dans ce cas les propriétés 1 sont **nécessaires et suffisantes**.

Souvent on cherchera un ensemble S_{T-1} tel que chaque sommet de G appartienne à S_{T-1} ou bien soit voisin d'un unique élément de l'ensemble S_{T-1}. Dit autrement, les boules de rayon 1, centrées sur chaque sommet de S_{T-1}, forment une partition des sommets de G. La dernière étape qui a priori semble la plus difficile, car c'est au cours de celle-ci qu'intervient généralement le plus grand nombre de communications, est dans ce cas aisée car les chemins sont tous de longueur 1. En fait, il apparait que les cas optimaux correspondent à des ensembles S_t associés à des codes correcteurs (Cf. l'exemple de la section 3.3.6-ii).

3.3.3 Le cycle dans le modèle H_*

Pour un message petit, nous avons la proposition suivante:

Proposition 3.3.2 — *Soit C_N le cycle d'ordre N. Pour $N = 3^i$, il existe un protocole de diffusion tel que:*

$b_{H_*}^{\mathcal{R}_\star}(C_N)$ *avec* $N = 3^i$		
b_α	b_δ	b_τ/L
$\log_3 N = i$	$\frac{1}{2}(N-1) = D$	b_α

Une preuve de cette proposition figure dans [28] ou [30].

La méthodologie générale de la section 3.3.2 s'applique ici en prenant :

$$S_t = k3^{i-t} \quad \text{avec } 0 \leq k \leq 3^t, \quad \text{donc } |S_t| = 3^t.$$

3.3.4 Les graphes hamiltoniens dans le modèle H_*

Une conséquence directe du résultat sur le cycle, implique les propositions suivantes :

Proposition 3.3.3 — *Si G est un graphe hamiltonien d'ordre N, alors il existe un protocole de diffusion tel que :*

$b_{H_2}^{\mathcal{R}_\star}(G)$		
b_α	b_δ	b_τ/L
$\lceil \log_3 N \rceil$	$\frac{1}{2}(N-1)$	b_α

Proposition 3.3.4 — *Soit G un graphe d'ordre N admettant une décomposable hamiltonienne arc-disjoints, alors il existe un protocole de diffusion en mode $\Delta-$ports tel que :*

$b_{H_*}^{\mathcal{R}_\star}(G)$		
b_α	b_δ	b_τ
$\lceil \log_3 N \rceil$	$\frac{1}{2}(N-1)$	$\frac{b_\alpha}{\Delta}$

Il suffit pour cela de couper le message en Δ blocs de longueur $\frac{L}{\Delta}$ et de diffuser simultanément un bloc sur chaque cycle hamiltonien.

3.3.5 Le tore de dimension 2 dans le modèle H_* et F_*

☞ Pour un nombre fixé d'étapes T dans un protocole de diffusion, le nombre maximum potentiel de sommets susceptible de recevoir le message est majoré par $(\Delta + 1)^T$. Dans le tore $TM(l)^2$ cette valeur vaut 5^T. Ainsi, dans cette partie Peters et Syska considèrent le problème critique de la diffusion sur les tores $TM(5^i)^2$.

Pour un petit message, J. Peters et M. Syska [26] ont établit la proposition suivante:

Proposition 3.3.5 ([26]) — *Dans le tore $TM(5^i)^2$, il existe un protocole de diffusion tel que :*

$b_{H_*}^{\mathcal{R}_\star}(TM(5^i)^2)$		
b_α	b_δ	b_τ/L
$\log_5(5^{2i}) = 2i$	$5^i - 1 = D$	b_α

Leur algorithme appelé *circuit-switched tiling* est basé sur un pavage récursif du tore de dimension 2.

La figure 3.8 décrit le protocole de diffusion utilisé dans le tore $TM(5)^2$. Pour ne pas surcharger la figure, nous avons omis de représenter tous les liens. La figure 3.8 - a représente les chemins de communication utilisés au cours de la première étape du protocole. Le sommet initiateur informe 4 nouveaux sommets le long de chemins arc-disjoints de longueur 3. La figure 3.8 - b représente les chemins de la seconde étape. Sur cette figure, en noir, sont représentés les sommets connaissant l'information après la première étape. Ces sommets noirs ont été choisis, comme indiqués par la méthodologie générale en section 3.3.2, de telle sorte qu'en envoyant leur information à l'ensemble de leurs voisins directs, tout le tore est informé. Ces communications se font sur des chemins arc-disjoints de longueur 1.

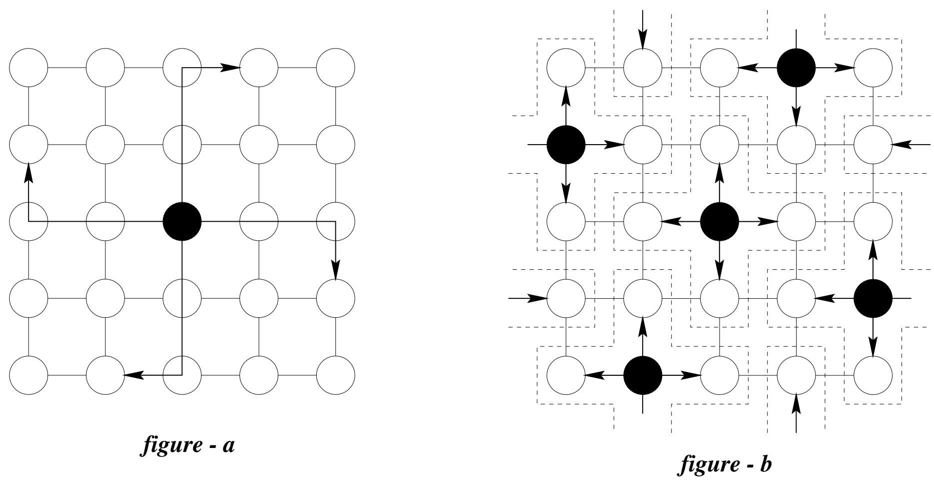

FIG. 3.8 – : *Diffusion dans le tore $TM(5)^2$.*

Les auteurs généralisent ce protocole de façon récursive aux tores dont le coté est une puissance de 5. La figure 3.9 représente les quatre étapes du protocole de diffusion dans le tore $TM(5^2)^2$. Comme précédemment, pour ne pas surcharger le dessin, nous avons omis de représenter les liens. Les points noirs représentent les sommets connaissant l'information au début d'une étape. La figure 3.9 - a décrit les chemins utilisés pour la première étape. En fait ces chemins sont les même que sur la figure 3.8 - a mais avec un facteur de dilatation de 5. La longueur de ces chemins arc-disjoints est de $5 \times 3 = 15$. La deuxième étape, représentée par la figure 3.9 - b, utilise elle aussi des chemins similaires à ceux de la figure 3.8 - b, avec toujours un facteur de dilatation de 5. La longueur de ces chemins arc-disjoints est de $5 \times 1 = 5$. En fait, au cours de cette étape chaque sommets noir envoie son information au centre de chaque bloc de 5×5 sommets voisin de son propre bloc 5×5. Enfin, les figures 3.9 - c et d représentent les deux dernières étapes du protocole. Ces deux étapes consistent simplement à effectuer le protocole de la figure 3.8, sur chaque bloc 5×5 de façon simultanée. La longueur totale des chemins est $15 + 5 + 3 + 1 = 24 = D(TM(5^2)^2)$. Il est possible de poursuivre la récursion de cet algorithme aux tore $TM(5^i)^2$.

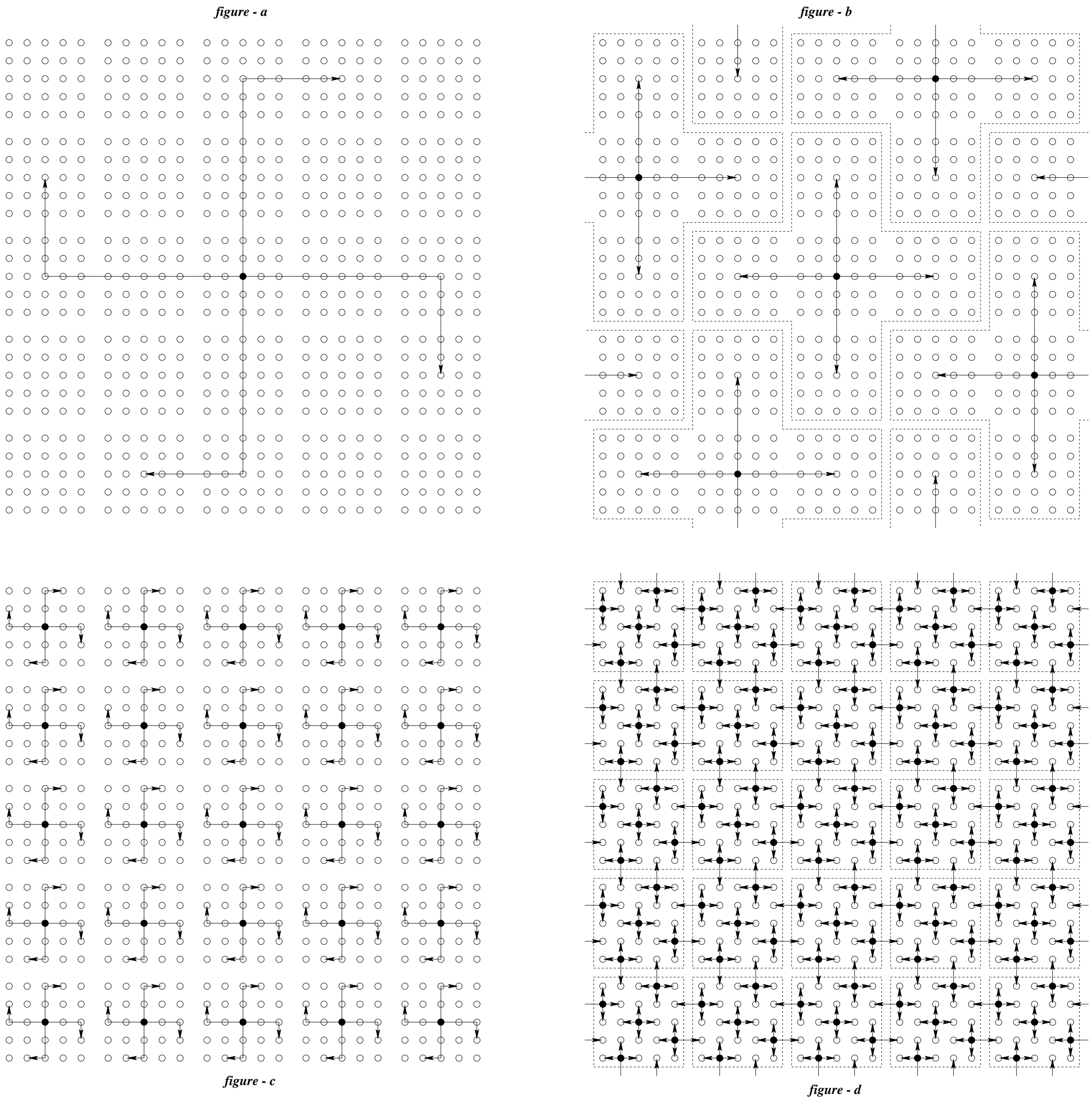

FIG. 3.9 – : *Diffusion dans le tore* $TM(5^2)^2$.

REMARQUE. Les pavages peuvent être adaptés à d'autres tailles, et donner ainsi des protocoles de diffusion optimaux en nombre d'étapes.

Dans le cas d'un long message, des techniques pipe-line ou arbres couvrants disjoints seront plus appropriés. Néanmoins ces techniques simulant celles employées dans le mode commutation de messages se font au détriment du nombre d'étapes. Dans [26], J. Peters et M. Syska, ont développé deux autres algorithme appelé *hybrides*, qui combine le mode commutation de circuit avec les techniques de pipe-line et d'arbres couvrants disjoints utilisés habituellement dans le mode commutation de messages. En effet, lorsque le temps de propagation du message à travers le réseau devient important, leurs algorithmes hybrides permettent d'effectuer la diffusion de façon efficace pour les trois paramètre à la fois. Ce sont les premiers algorithmes qui utilisant le mode commutation de circuit permettent d'obtenir de bonnes performance sur les trois paramètres simultanément. Pour cela, nous donnons une idée de leurs algorithmes. Ces algorithmes permettent de diminuer le paramètre $b_\tau(G)$ sans pour cela augmenter de façon trop importante les deux autres paramètres $b_\alpha(G)$ et $b_\delta(G)$.

L'idée sur laquelle repose ces algorithmes hybrides est la suivante. Un protocole de diffusion forme un arbre couvrant orienté où la racine est le sommet émetteur. Dans le mode commutation de message les arcs de cet arbre représentent un lien de communication du réseau. Dans le mode commutation de circuits, chaque arcs de l'arbre représente un chemin dans le réseau. Dans les deux cas les arcs de l'arbre de diffusion doivent être disjoints pour un instant donné. Dans le mode commutation de message, le message doit être complètement reçu avant de pouvoir être reémis par un nœud du réseau. Si le message est long, le délai entre le début et la fin de la réception du message peut-être très long. La technique du pipe-line réduit ce délai en partitionnant le message en paquets et en les envoyant les uns après les autres le long d'un même chemin. Ainsi, un nœud peut commencer à reémettre le message (dès qu'il a reçu le premier paquet), avant même d'avoir reçu tout le message. Ainsi, le délai est réduit par recouvrement des phases de l'algorithme. Cette même technique est applicable à la commutation de circuit si les chemins permettant le recouvrement des phases sont arc-disjoints.

Comme l'algorithme précédent, ne peut pas utiliser une méthode de recouvrement de phases, les auteurs ont développé dans [26] deux algorithmes hybrides appelés *circuit-switched D&C* et *circuit-switched H-trees*. Nous donnons ici une idée de l'algorithme *circuit-switched H-trees* sur les tores $TM(2^i-1)^2$.

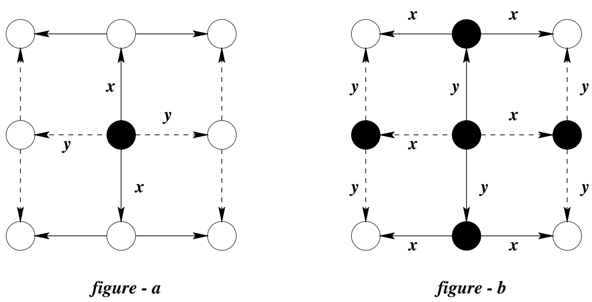

FIG. 3.10 – : *H-tree dans le tore* $TM(3)^2$.

La figure 3.10 montre comment réaliser une diffusion dans le tore $TM(3)^2$ en utilisant deux arbres de diffusion arc-disjoints. Le premier est représenté par les arcs solides, et le second par les arcs en pointillé (le second est obtenu par rotation de 90 degrés du premier). Ces arbres sont appelé H-tree pour des raisons évidentes. L'algorithme utilise deux phases. La figure montre que le message est partionné en deux paquets X et Y. Dans une première phase (figure 3.10 - a) le sommet émetteur envoie le paquet X sur les liens verticaux et Y sur les horizontaux. Dans l'autre phase (figure 3.10 - b), le schéma à partir du sommet source est inversé et les quatre sommets informés après la première étape réémettent le paquet qu'il ont déjà reçu vers leurs deux voisins dans le H-tree.

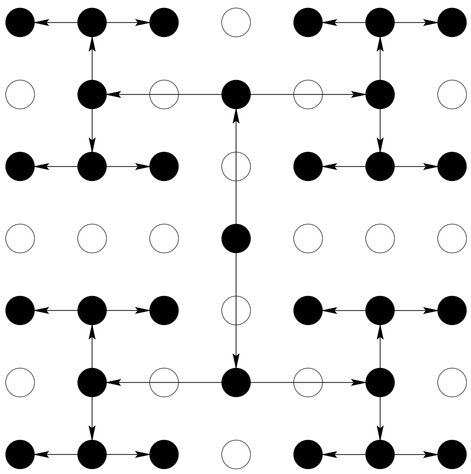

FIG. 3.11 – : *H-tree dans le tore* $TM(7)^2$.

La figure 3.11 montre un H-tree dans le tore $TM(7)^2$. Ce schéma de communications se généralise facilement en utilisant une définition récursive de l'algorithme de diffusion utilisant des H-trees. De plus des techniques de pipe-line peuvent être utilisé sur ce schéma de communication.

En supposant que le message initial de longueur L est partitionné en p paquets (ou blocs) de même taille, nous résumons dans les tableaux ci-dessous, le coût de leurs algorithmes hybrides.

$b_{F_*}^{\mathcal{R}\star}(TM(2^i)^2)$			
- Algorithme -	b_α	b_δ	b_τ/L
Circuit-switched D&C	$i+p-1$	$2^{i-1}(p+1)$	$\frac{b_\alpha}{p} = 1 + \frac{i-1}{p}$

Le flot d'information de cet algorithme se comporte asymptotiquement en $L\tau$ lorsque $p >> i-1$. Mais dans ce cas cela augmente de façon non négligeable le nombre d'étapes.

$b_{F_*}^{\mathcal{R}\star}(TM(2^i-1)^2)$			
- Algorithme -	b_α	b_δ	b_τ/L
Circuit-switched H-tree	$2i + \frac{p}{2} - 2$	$2^{i-2}(\frac{p}{2}+3) - 1$	$\frac{b_\alpha}{p} = \frac{1}{2} + \frac{2(i-1)}{p}$

Le flot d'information de cet algorithme se comporte asymptotiquement en $\frac{L}{2}\tau$ lorsque $p >> 2(i-1)$. Mais dans ce cas cela augmente de façon non négligeable le nombre d'étapes.

3.3.6 Le tore de dimension k dans le modèle H_*

i) Algorithme de Park et Choi

Park et Choi ont étudié [24, 25] un algorithme de diffusion, pour un message petit, optimal en nombre d'étapes sur les tores $TM((2k+1)^i)^k$. Elles obtiennent en particulier :

$b_{H_*}^{\mathcal{R}_*}(TM((2k+1)^i)^k)$	
b_α	b_τ/L
$\log_{\Delta+1} N = \log_{2k+1}(2k+1)^{ki} = ki$	b_α

Leur algorithme est basé sur un partitionnement du tore en sous-grilles, le sommet émetteur allant informer le centre de chaque sous-grille. Néanmoins, l'algorithme conçu pour un petit message, ne prend pas en compte le paramètre $b_\delta(TM((2k+1)^i)^k)$. Il ne cherche pas à minimiser ce paramètre. Par exemple, lors de la dernière étape de l'algorithme, les chemins utilisés sont de longueur supérieure à 1.

ii) Algorithme de Delmas et Perennes

Indépendamment des travaux de Park et Choi, nous avons étudié dans [7, 8], pour un petit message, la diffusion dans les tores $TM((2k+1)^i)^k$ de dimension k . Nous donnons des protocoles de diffusion optimaux en nombre d'étapes et quasi-optimaux pour les longueurs des chemins. Ceci généralise le résultat [26] obtenu pour $k=2$.

Peut mieux aborder cette section, le lecteur pourra également se reporter à l'article en annexe A. Dans [7] nous montrons que notre technique s'appuie essentiellement sur deux méthodes combinant des outils d'algèbre linéaire et de théorie des codes. En effet, nos algorithmes sont basés sur une décomposition du tore en domaines, qui est associée à un pavage répliqué. Nous commençons par redonner avec notre approche le résultat obtenu par Peters et Syska (Cf. section 3.3.5) sur les tore de dimension 2. Nous verrons que cette approche permet de généraliser aux tores de dimension supérieure.

Ainsi, si on note $B_k = \{e_1, e_2, \cdots, e_k, 0, -e_1, -e_2, \cdots, -e_k\}$ le vecteur nul et l'ensemble des vecteurs de norme 1 de l'espace $\mathbb{Z}_{2k+1}^k$ on remarque les propriétés suivantes:

- pour $k=2$ et $i=1$, $\mathbb{Z}_5^2 = (M_2 + I_d)B_2$ où $M_2 = \begin{pmatrix} 1 & 2 \\ 2 & -1 \end{pmatrix}$,
- plus généralement pour $k=2$ et $\forall i$, $\mathbb{Z}_{5^i}^2 = (M_2^{2i-1} + M_2^{2i-2} + \cdots + M_2 + I_d)B_2$.

L'algorithme de diffusion est alors le suivant :

Algorithme 3.3.1 — Algorithme de diffusion dans $TM(5^i)^2$

- `Pour` tous les temps t allant de 1 à $2i$ `faire`
  - Au temps t tout sommet x informé transmet son information aux sommets $x + M_2^{2i-t}B_2$.
- `fin faire`

On montre qu'il est possible de réaliser les communications de cet algorithme le long de chemins arcs disjoints. Par induction les sommets informés après l'étape t sont les éléments de l'ensemble $S_t = (M_2^{2i-1} + M_2^{2i-2} + \cdots + M_2^{2i-t})B_2$. Ainsi l'algorithme dure $2i$ étapes et une analyse plus détaillée montre qu'il utilise des chemins de longueur optimale. Le temps de diffusion est donc $b_{H_*}^{\mathcal{R}_\star}(TM(5^i)^2) = 2i\alpha + D(TM(5^i)^2)\delta + 2iL\tau$.

Nous avons généralisé cette technique aux tores $TM(7^i)^3$ de dimension 3. En utilisant la matrice :

$$M_3 = \begin{pmatrix} 0 & 1 & 0 \\ -2 & 3 & 1 \\ -1 & 0 & -3 \end{pmatrix}$$

nous obtenons :

$b_{H_*}^{\mathcal{R}_\star}(TM(7^i)^3)$		
b_α	b_δ	b_τ/L
$\log_{\Delta+1} N = \log_7(7^{3i}) = 3i$	$\frac{10}{9}D$	b_α

En dimension 4, sur une remarque de C. Delorme, il est préférable d'utiliser le tore $TM(3)^4$. Ainsi, en utilisant la matrice :

$$M_4 = \begin{pmatrix} 0 & 1 & 1 & 1 \\ -1 & 0 & 1 & -1 \\ -1 & -1 & 0 & 1 \\ -1 & 1 & -1 & 0 \end{pmatrix}$$

on démontre de manière analogue que pour $TM(3^i)^4$ nous avons :

$b_{H_*}^{\mathcal{R}_\star}(TM(3^i)^4)$		
b_α	b_δ	b_τ/L
$\log_9(3^{4i}) = 2i$	D	b_α

Plus généralement, nous montrons dans [8] (Cf. annexe A) que le nombre d'étapes de la diffusion dans le tore $TM((2k+1)^i)^k$ est optimale, le long de chemins de longueur quasi-optimale. Ainsi, nous avons pu établir les faits suivantes.

Proposition 3.3.6 — *Soit $G_i^k = TM((2k+1)^i)^k$. Il existe un protocole de diffusion dans G_i^k tel que $b_\alpha(G_i^k) \le b_\alpha(G_1^k)i$ et $b_\delta(G_i^k) \le \frac{b_\delta(G_1^k)}{D(G_1^k)}D(G_i^k)$.*

☞ Il est remarquable que du fait des équations de récurrence le rapport $\frac{b_\delta(G_i^k)}{D(G_i^k)}$ reste au plus fixe, égal à celui obtenu pour le premier tore G_1^k.

Corollaire 3.3.7 — *Il existe un protocole de diffusion sur le tore $TM(2k+1)^k$ tel que $b_\alpha(TM(2k+1)^k) = \log_{2k+1}(2k+1)^k = k$ et $b_\delta(TM(2k+1)^k) \leq k^2 + 2k + k\ln(k^2+1)$.*

Ce qui conduit au corollaire suivant :

Corollaire 3.3.8 — *Il existe un protocole de diffusion dans le tore $TM((2k+1)^i)^k$ tel que :*

$b_{H_*}^{\mathcal{R}_\star}(TM((2k+1)^i)^k)$		
b_α	b_δ	b_τ/L
$\log_{\Delta+1} N = \log_{2k+1}((2k+1)^{i^k}) = ki$	$(1+\frac{2}{k}+\ln(k+1))D$	b_α

Pour aboutir à ces résultats, nous avons suivit la métodologie générale décrite en section 3.3.2 afin d'effectuer une diffusion optimale en nombre d'étapes sur le tore $TM(2k+1)^k$.

Dans le tore $TM(2k+1)^k$ le nombre potentiel minimum d'étapes est $\log_{2k+1}(2k+1)^k = k$. Effectuer une diffusion optimale en k étapes revient à chercher des ensembles S_t ayant les propriétés 1, mais vérifiant également la condition $|S_{t+1}| = (2k+1)\cdot|S_t|$, alors ces propriétés sont nécessaires et suffisantes.

Afin de diffuser dans le tore $TM(2k+1)^k$, nous devons définir les ensembles S_t. Pour obtenir une diffusion en un nombre optimum d'étapes, il est nécessaire que $S_k = \mathbb{Z}_{2k+1}^k$. Nous chercherons ensuite à définir l'ensemble S_{k-1} à partir de la remarque suivante.

Dans un protocole de diffusion utilisant les ensembles S_t, l'étape dans laquelle est impliqué le plus grand nombre de communications est la dernière, c'est-à-dire lorsque les sommets de S_{k-1} informent ceux de S_k. Comme remarqué en section 3.3.2, il est intéressant de choisir un ensemble S_{k-1} réparti de telle sorte que l'on sache réaliser aisément les communications de cette étape. C'est le cas si l'ensemble S_{k-1} vérifie la définition :

Definition. Les sommets S_{k-1} sont dit être les éléments d'un **code parfait**, s'il vérifie la propriété suivante: chaque sommet de S_k est soit un élément de S_{k-1} soit un voisin direct (i.e. à distance 1) d'exactement un et un seul élément de S_{k-1}.

En effet, si S_{k-1} est un code couvrant, alors l'union des éléments de S_{k-1} et de tous leurs voisins directs correspond à S_k. La dernière étape se réduit alors à l'envoi par chaque sommets de S_{k-1} du message à chacun de ses voisins, les chemins sont alors tous de longueur 1 et clairement arc-disjoints.

REMARQUE. La distance dans le tore correspond à la distance de Lee sur $\mathbb{Z}_{2k+1}^k$ (voir [23]). La propriété que nous recherchons sur S_{k-1} se traduit [23] en théorie des codes ainsi: "*S_{k-1} est un code de Lee de rayon de recouvrement 1 de cardinalité $(2k+1)^{k-1}$, c'est donc un code parfait de rayon 1*".

De tels codes parfaits de rayon 1 sont bien connus, nous choisirons d'utiliser $S_{k-1} = \{x \in S_k \mid x_1 + 2x_2 + \cdots + kx_k = 0\ (2k+1)\}$.

Nous définissons alors les ensembles S_t inductivement comme suit: pour $k-2 \leq t \leq 0$ $S_t = \{x \in S_{t+1} \mid x_{t+2} = 0\}$. Les définitions des ensembles S_t se résument par:

$$\left\{\begin{array}{lcl} S_k & = & \mathbb{Z}_{2k+1}^k, \\ S_{k-1} & = & \{x \in S_k \mid \sum_{t=1}^{t=k} tx_t = 0\ (2k+1)\}, \\ S_t & = & \{x \in S_{t+1} \mid x_{t+2} = 0\}, \text{ pour } k-2 \leq t \leq 0. \end{array}\right.$$

Nous montrons dans [8] que de tels ensembles S_t vérifient les propriétés 1.

3.3.7 La grille de dimension k dans le mode H_*

Park et Choi [24] ont étudié le problème de la diffusion dans les grilles en considérant le cas d'un petit message. Elles montrent que, dans le cas d'une grille $M(2k+1)^k$ de dimension k, on ne peut pas atteindre la borne inférieure sur le nombre d'étapes.

Proposition 3.3.9 ([24]) — *Sur la grille $M(2k+1)^k$ de dimension k, le nombre d'étapes nécessaire pour effectuer une diffusion est:*

$$\begin{array}{rcl} b_\alpha(M(2k+1)^k) & \geq & \log_{2k+1}(2k+1)^k + 1 \\ & \geq & k+1 \end{array}$$

En modifiant leur algorithme initialement construit pour le tore de dimension k, elles obtiennent un protocole dont le coût est :

$b_{H_*}^{\mathcal{R}_\star}(M((2k+1)^i)^k)$	
b_α	b_τ/L
$\log_{2k+1}((2k+1)^{ik}) + i + k - 1 = ki + i + k - 1$	b_α

Dans cette étude les auteurs ne considèrent pas la longueur des chemins.

☞ Peu de références existent sur la diffusion en mode de commutation wormhole dans les grilles sous la contrainte F_*. Du fait que tous les sommets n'ont pas le même degré, les algorithmes n'atteignent pas la borne inférieure sur le nombre d'étapes.

3.3.8 L'hypercube dans le modèle F_*

La diffusion d'un petit message dans l'hypercube paraît être un problème difficile si l'on cherche un protocole optimal en nombre d'étapes. Pour cette raison et comme nous le verrons dans cette section, souvent les auteurs ne considère pas la longueur des chemins, car cela conduirait à aboutir à un problème encore plus ardu. Le coût de la diffusion, dans l'hypercube de dimension n, est donc réduit ici à $b_{F_*}^{\mathcal{R}_*}(H(n)) = b_\alpha(H(n))\alpha + b_\tau(H(n))L\tau$. Les deux premiers protocoles que nous proposons utilisent le mode classique de commutation de messages.

i) Algorithme de l'arbre binômial recouvrant

Une première idée est d'avoir recours à un arbre de recouvrement binômial et de diffuser le message sur les n ports de sortie. Cependant, cet algorithme, nommé SBT pour *Spanning Binômial Tree* (Cf. section 3.4.3), se révèle ne pas être une bonne approche comme l'illustre la figure 3.12. Certes, certains sommets vont recevoir le message plus tôt que dans la version 1-port (voir figure 3.20), mais le nombre d'étapes sera toujours n.

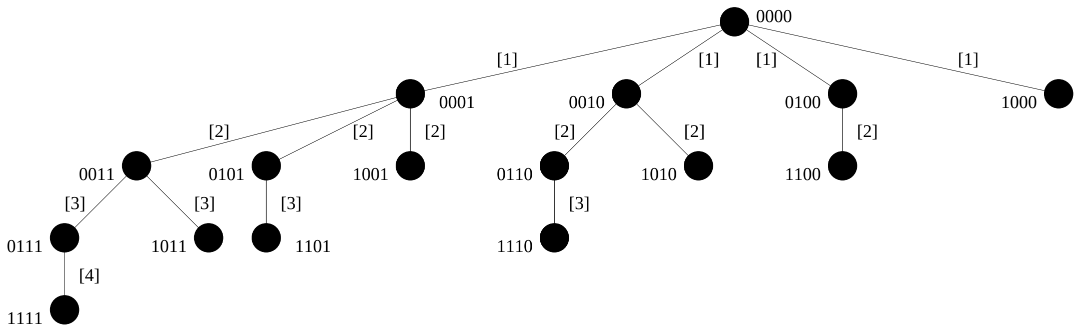

FIG. 3.12 – : *Algorithme de diffusion SBT dans le mode F_* sur un hypercube $H(4)$.*

Ainsi, la complexité de cet algorithme est identique à celle du mode F_1 présenté en section 3.4.3.

ii) Algorithme de l'arbre double

McKinley et Trefftz [19] ont modifié légèrement l'algorithme SBT et ont proposé un algorithme nommé *DT* (*Double Tree*) afin d'améliorer le nombre d'étapes. L'idée est d'effectuer, cette fois-ci, la diffusion le long de deux arbres de recouvrement dont les sources sont respectivement s et $\bar{s}$.

Dans cet algorithme, le sommet émetteur, noté s, effectue la première étape de l'algorithme SBT en mode Δ ports, sauf que le message envoyé dans la dimension 1, n'est pas à destination du voisin de s dans cette dimension, mais au sommet qui lui est antipodal (i.e. $\bar{s}$). Ensuite, les deux sommets s et $\bar{s}$ se comportent comme les racines de deux arbres couvrants partiels. Dans le premier arbre, le routage s'effectue en changeant les 0 en 1, et dans le second arbre, le routage s'effectue en changeant les 1 en 0. Un exemple de diffusion dans $H(5)$ est illustré par la figure 3.13.

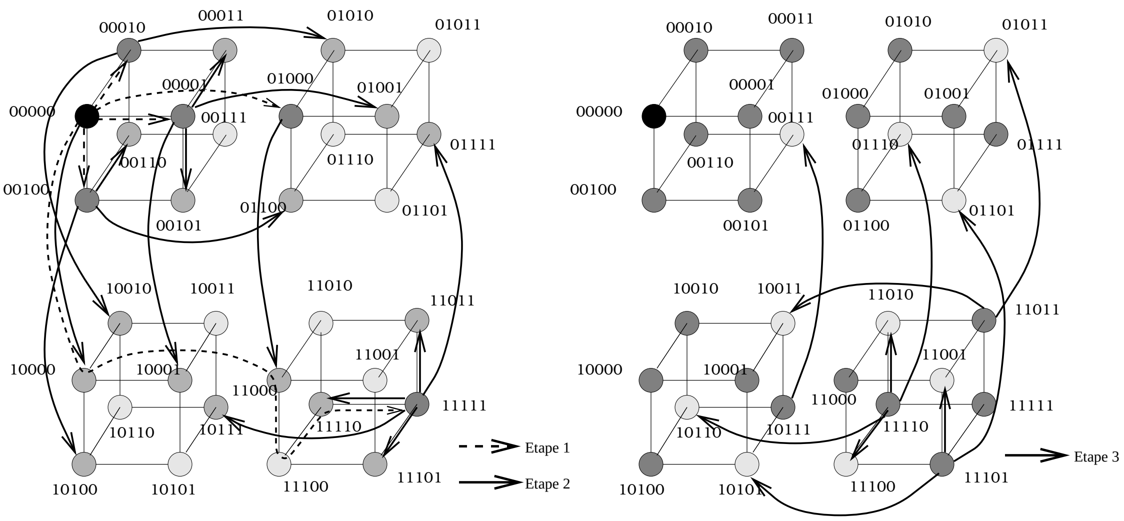

FIG. 3.13 – : *Algorithme DT dans l'hypercube $H(5)$.*

En appliquant cet algorithme on obtient $b_{F_*}^{\mathcal{R}\star}(H(n)) \leq \alpha + D\delta + L\tau + \left\lfloor \frac{n-1}{2} \right\rfloor (\alpha + \delta + L\tau)$, soit :

$b_{F_*}^{\mathcal{R}\star}(H(n))$		
b_α	b_δ	b_τ / L
$\left\lceil \frac{\log_2 N}{2} \right\rceil$	$\frac{3}{2}D$	b_α

Cet algorithme est n'est pas optimal en nombre d'étapes.

iii) Protocole utilisant des sous hypercubes disjoints

Le principe de cet algorithme, qui se trouve aussi dans [18], a été donné par Perennes (communication privée) et est à la base de l'article de Ho et Kao [17] qui ont obtenu des résultats similaires de façon indépendante (Cf. section 3.5.3-i). Il construit l'ensemble S_{t+1} à partir de S_t de manière récursive.

On essaie de partitionner l'hypercube $H(n)$ en $d+1$ sous hypercubes (deux à deux sommets disjoints) $H_0, H_1, \cdots, H_d$, de sorte que le sommet initiateur envoie son message à un représentant de chaque sous hypercube selon des chemins deux à deux arc-disjoints. A l'étape suivante, on diffuse en parallèle dans chacun des sous hypercubes et on obtient la relation suivante :

$$b_\alpha(H(n)) = 1 + \max b_\alpha(H_i).$$

L'idée est de choisir les dimensions des sous-hypercubes H_i de sorte que le maximum de leurs dimensions soit le plus petit possible. Si la dimension de H_i

est n_i, on a $\sum_{i=0}^{d} 2^{n_i} = 2^n$. Le meilleur choix consiste à prendre tous les n_i égaux ou quasi égaux, c'est-à-dire $n_i = m$, pour tout i, $0 \leq i \leq d$, ou $n_i = m$ ou $m+1$ pour tout i. Ceci implique $(d+1)2^m \leq 2^n$ et $(d+1)2^{m+1} > 2^n$ d'où $m = n - \lfloor \log_2(d+1) \rfloor$.

On doit donc choisir d le plus grand possible. Les chemins entre le sommet initiateur et les représentants des $d+1$ sous hypercubes doivent être deux à deux arc-disjoints. On peut choisir le sommet initiateur comme représentant du sous hypercube auquel il appartient. Comme le degré de $H(n)$ est n, on pourra au plus trouver n autres chemins arc-disjoints et donc $d+1 \leq n+1$ soit $d \leq n$. On est donc amené à décomposer, si possible, $H(n)$ en $n+1$ sous hypercubes de dimension m ou $m+1$ avec $m = n - \lfloor \log_2(n+1) \rfloor$.

De telles décompositions existent en grand nombre. Une fois la décomposition trouvée, soit H_0 le sous hypercube contenant l'initiateur. On peut choisir n'importe quel sommet x_i comme représentant de H_i. En effet, l'hypercube étant n-connexe, il existe n chemins deux à deux sommet (et donc arc)-disjoints entre l'émetteur et les x_i, $1 \leq i \leq n$.

Pour la décomposition, une manière simple est de prendre comme sous hypercubes de dimension $n-m$ (resp. $n-m+1$) celui contenant les sommets ayant m (resp. $m+1$) coordonnées fixées. Deux tels sous hypercubes sont disjoints si les valeurs fixées ne sont pas identiques. Par exemple, si $n = 2^k - 1$, alors $n - k = m$. On peut considérer les 2^k possibilités obtenues en fixant les k premières coordonnées (voir exemple avec $n = 7$). Si $2^k - 1 < n < 2^{k+1} - 1$, on a $m = n - k$ et on fixe les k premières coordonnées pour $2^{k+1} - n - 1$ sous hypercubes et les $k+1$ premières pour les autres (voir l'exemple ci-dessous pour $n = 5$).

Exemple. Exemples de décomposition. Les huit sous hypercubes (de dimension 4) pour $n = 7$ et les six sous hypercubes (deux de dimension 4 et quatre de dimension 3) pour $n = 5$.

000□□□□	
001□□□□	00□□□
010□□□□	01□□□
011□□□□	100□□
100□□□□	101□□
101□□□□	110□□
110□□□□	111□□
111□□□□	

Une autre manière consiste à procéder par récurrence.

Considérons le cas $n = 2^k - 1$. On divise les sous hypercubes cherchés en deux catégories : ceux dont la dernière coordonnée vaut 0 et les $k-1$ autres coordonnées fixées sont choisies parmi les dimensions $1, 2, \cdots, 2^{k-1}-1$ et ceux dont la dernière coordonnée vaut 1 et les $k-1$ autres coordonnées fixées sont choisies parmi $2^{k-1}, 2^{k-1}+2, \cdots, 2^k-1$. On répète le processus k fois (voir l'exemple ci-dessous).

EXEMPLE. Exemples de décomposition avec $n = 3$ et $n = 7$.

		0□0□□□0	H_0
		1□0□□□0	H_1
0□0	H_0	□11□□□0	H_2
1□0	H_1	□01□□□0	H_3
□11	H_2	□□□1□01	H_4
□01	H_3	□□□□111	H_5
		□□□□011	H_6
		□□□0□01	H_7

Nous verrons dans la section3.5.3-i que cette méthode à l'avantage de pouvoir réaliser les étapes de communications le long de chemins arc-disjoints en utilisant une fonction de routage classique de l'hypercube.

En résumé, nous avons :

Théorème 3.3.10 —

$$b_\alpha(H(n)) \leq b_\alpha(H(n - \log_2(n+1))) + 1$$

Une application répétée de ce théorème fournit un protocole asymptotiquement optimal.

$b_{F_*}^{\mathcal{R}_\star}(H(n))$
$b_\alpha(H(n))$
$\Theta\left(\frac{n}{\log_2(n+1)}\right) = \Theta\left(\log_{\Delta+1} N\right)$

Sans fonction de routage imposée, on peut effectuer la diffusion en deux étapes sur $H(5)$, ce qui est optimal.

Proposition 3.3.11 — *Pour un hypercube de dimension* 5, $b_\alpha(H(5)) = 2$.

PREUVE. L'idée est de choisir 5 sommets relativement éloignés comme ensemble S_1 pour que la diffusion s'effectue approximativement entre des sommets voisins à la deuxième étape.
Les 5 sommets choisis pour S_1 sont $00111, 10101, 01110, 11011$ et 11100. Il est aisé de trouver des chemins disjoints pour cette étape. Ensuite, comme le montre la figure 3.14, ces sommets diffusent le message à tous les sommets non informés selon des chemins de longueur 1 excepté dans quatre cas où l'on a besoin de chemins de longueur 2 ou 3. □

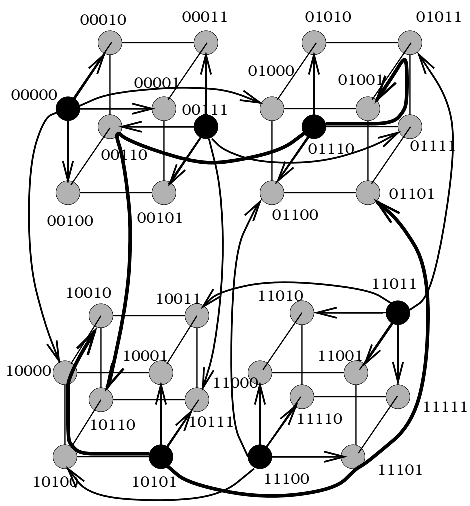

FIG. 3.14 – : *Diffusion dans le mode F_* sur $H(5)$.*

REMARQUE. La valeur obtenue pour $H(5)$ permet d'améliorer d'autres valeurs par le théorème 3.3.10 comme par exemple $n = 8$ pour lequel $b_\alpha(H(8)) = 3$.

iv) Protocole basé sur les codes

Ce protocole est présenté dans la thèse de Kodate [18]. Il utilise pour $n = 2^m - 1$ l'existence de codes linéaires cycliques très particuliers. Ainsi, l'ensemble S_{t-1} correspond à l'existence de codes parfaits de Hamming (Cf. [23]). En effet, un sommet de l'hypercube peut être considéré comme un mot binaire d'un code, la distance dans l'hypercube correspond à la distance de Hamming entre 2 mots du code.

v) Résumé des résultats sur l'hypercube

Le tableau suivant montre la borne inférieure et les nombres d'étapes nécessaires avec différents algorithmes proposés. MT désigne l'algorithme de McKinley et Trefftz (section 3.3.8-ii), HK celui de Ho et Kao (Cf. section 3.5.3-i), HK' celui présenté en section 3.3.8-iii et $Codes$ celui obtenu par l'utilisation de codes correcteurs d'erreur et présenté en section 3.3.8-iv.

3.3.9 Le Butterfly dans le modèle F_*

Nous étudions ici la diffusion sur le graphe Butterfly orienté de degré (entrant et sortant) $d = 2$, noté $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2, n)$. Comme nous l'avons vu dans le chapitre 1.4, nous utiliserons, dans le but de simplifier la représentation graphique où le niveau

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	$\cdots$	31
Borne inf.	2	2	2	2	3	3	3	3	3	4	4	4	4	4	4		5
MT	2	2	2	3	3	4	4	5	5	6	6	7	7	8	8		16
HK	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	$\cdots$	10
HK'	2	2	2	2	3	3	3	4	4	4	5	5	5	5	6		9
$Codes$		2				3		3			4			4			5

TAB. 3.1 – : *Nombres d'étapes nécessaires avec différents algorithmes de diffusion dans l'hypercube $H(n)$.*

0 est dédoublé . Dans cette étude, nous ne prendrons pas en compte la longueur des chemins. La borne inférieure sur le nombre d'étapes est $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)) = \lceil \log_{d+1}(n2^n) \rceil = \lceil n \log_{d+1} 2 + \log_{d+1} n \rceil$. Il semble difficile de trouver un protocole atteignant cette borne inférieure. Nous donnons donc ici un protocole efficace, mais non optimal.

Comme le graphe *Butterfly* est un graphe de Cayley, nous pourront supposer dans la suite, sans perdre de généralité, que le sommet émetteur est $(0,0)$. Alors nous cherchons à construire un algorithme de diffusion utilisant deux phases[2]. Dans la première phase, le sommet émetteur envoie son information à tous les sommets du niveau 0. Puis dans la seconde phase, l'ensemble des sommets informés diffuse l'information au reste du réseau. Ainsi, le nombre d'étapes nécessaires à un tel protocole sera la somme du nombre d'étapes de la phase 1 et de la phase 2, et nous le noterons :

$$b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)) \quad = \quad b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)) + b_\alpha^2(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n))$$

Dans la suite nous aurons besoin de la définition suivante.

DEFINITION. Soit (x,l), avec $x \in \mathbb{Z}_2^n$ et $l \in \mathbb{Z}_n$, les sommets du graphe orienté $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$, alors on appelle $ligne(x)$ le circuit $(x,0),(x,1),\cdots,(x,n-1),(x,0)$.

Nous commençons par résoudre la seconde phase du protocole.

Proposition 3.3.12 — *Si dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ les 2^n sommets d'un même niveau connaissent l'information, alors il est possible de diffuser l'information sur les $(n-1)2^n$ autres sommets en $\lceil \log_3 n \rceil$ étapes.*

PREUVE. Comme tous les niveaux jouent le même rôle, nous supposerons sans perdre de généralité que tous les sommets du niveau 0 connaissent l'information. Alors les sommets détenant l'information sont tous sur une $ligne(x)$ différente. Pour être optimal en nombre d'étapes, il faut simuler la diffusion $d-$ports optimale du cycle sur chaque ligne. Comme le graphe est orienté, toujours dans le

2. Une phase peut contenir plusieurs étapes

même sens, il n'est pas possible de simuler la diffusion du cycle en n'utilisant que les arcs de pente[3] 0. Cependant, comme le graphe est 2-régulier, chaque sommet à un degré 2 entrant et sortant. Ainsi, si chaque sommet du niveau 0 envoie son information selon deux chemins arc-disjoints, l'un n'utilisant que les pentes 0, l'autre que les pentes 1, alors il est possible en une première étape d'informer 2 nouveaux sommets sur chacune des lignes. L'un reçoit son information du sommet du niveau 0 de sa propre ligne (en utilisant que les pentes 0), l'autre reçoit l'information d'un sommet de niveau 0 en provenance d'une autre ligne (en utilisant que des pentes 1). Comme toutes les lignes utilisent simultanément le même protocole et comme les chemins utilisent soit que des arcs de pente 0 ou bien que de pente 1, alors les chemins sont arc-disjoints. En appliquant le même type de stratégie, pour les étapes suivantes (i.e. un sommet connaissant l'information la renvoie selon deux chemins, l'un n'utilisant que les pentes 0, l'autre que les pentes 1), et comme le protocole est le même sur chaque ligne, alors il est possible de simuler la diffusion optimale en nombre d'étapes du cycle le long de chemins arc-disjoints sur chacune des lignes. □

Nous nous intéressons maintenant à la première phase du protocole. Pour cela il faut étudier la diffusion sur tous les sommets du niveau 0. Pour ce problème, la borne inférieure sur le nombre d'étape est $\lceil n \log_{d+1} 2 \rceil$.

Nous étudions dans un premier temps la diffusion dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$. La figure 3.15, montre comment le sommet $(000,0)$ informe tous les sommets du niveau 0 en deux étapes.

3. Cf. section 1.4.2 pour une définition de "pente".

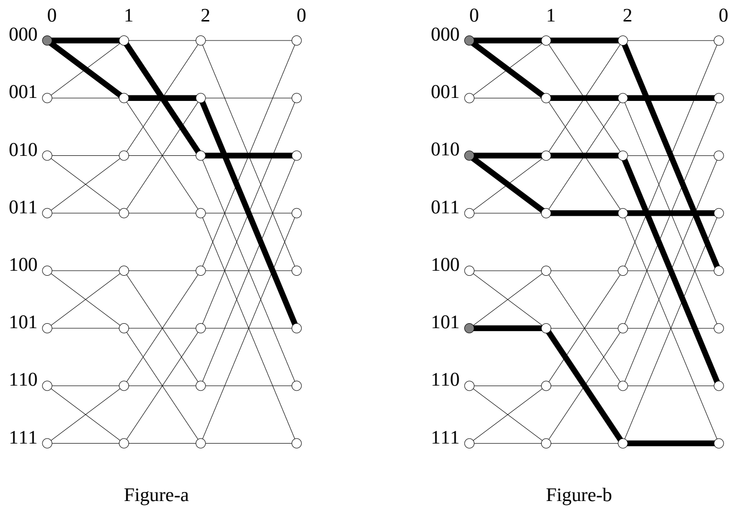

FIG. 3.15 – : *Diffusion dans* $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$ *sur les sommets de niveau 0. Au cours de la première étape (figure-a), l'initiateur informe les sommets* $(010,0)$ *et* $(101,0)$ *par des chemins arc-disjoints. Durant la seconde étape (figure-b) les trois sommets détenant l'information informent tous les sommets du niveau 0 par des chemins arc-disjoints.*

REMARQUE. Notons qu'en utilisant la proposition 3.3.12 et l'algorithme de diffusion sur le cycle, nous pouvons informer les sommets restants en une étape, c'est-à-dire effectuer de façon optimale la seconde phase du protocole. Les chemins utilisés sont :

$$\begin{cases} \forall x \in \mathbb{Z}_2^3, & (x,0) \rightarrow (x,1) \rightarrow (x,2) \quad \text{et} \\ \forall x \in \mathbb{Z}_2^3, & (x,0) \rightarrow (x+1,1) \end{cases}$$

Le coût global de la diffusion est donc $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)) = 2 + \lceil \log_3 3 \rceil = 3$.

Nous pousuivons l'étude sur le $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,6)$. Nous montrons que nous pouvons induire un protocole de diffusion à partir de celui décrit pour $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$. Le protocole consiste à informer en deux étapes les sommets $(y\ 000, 0)$ $\forall y \in \mathbb{Z}_2^3$. La figure 3.16-a montre comment le sommet $(0,0)$ informe dans une première étape les sommets $(010\ 000, 0)$ et $(101\ 000, 0)$. La figure 3.16-b montre comment les sommets $(y\ 000, 0)$ avec $y \in \mathbb{Z}_2^3$ reçoivent l'information au cours de la seconde étape. Remarquons que ces deux première étapes correspondent aux deux premières étapes de la diffusion dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$ lorsque l'on fixe les trois bits de poids faibles des sommets à 000 (i.e. en considérant les sommets $(y\ 000, 0)$ avec $y \in \mathbb{Z}_2^3$).

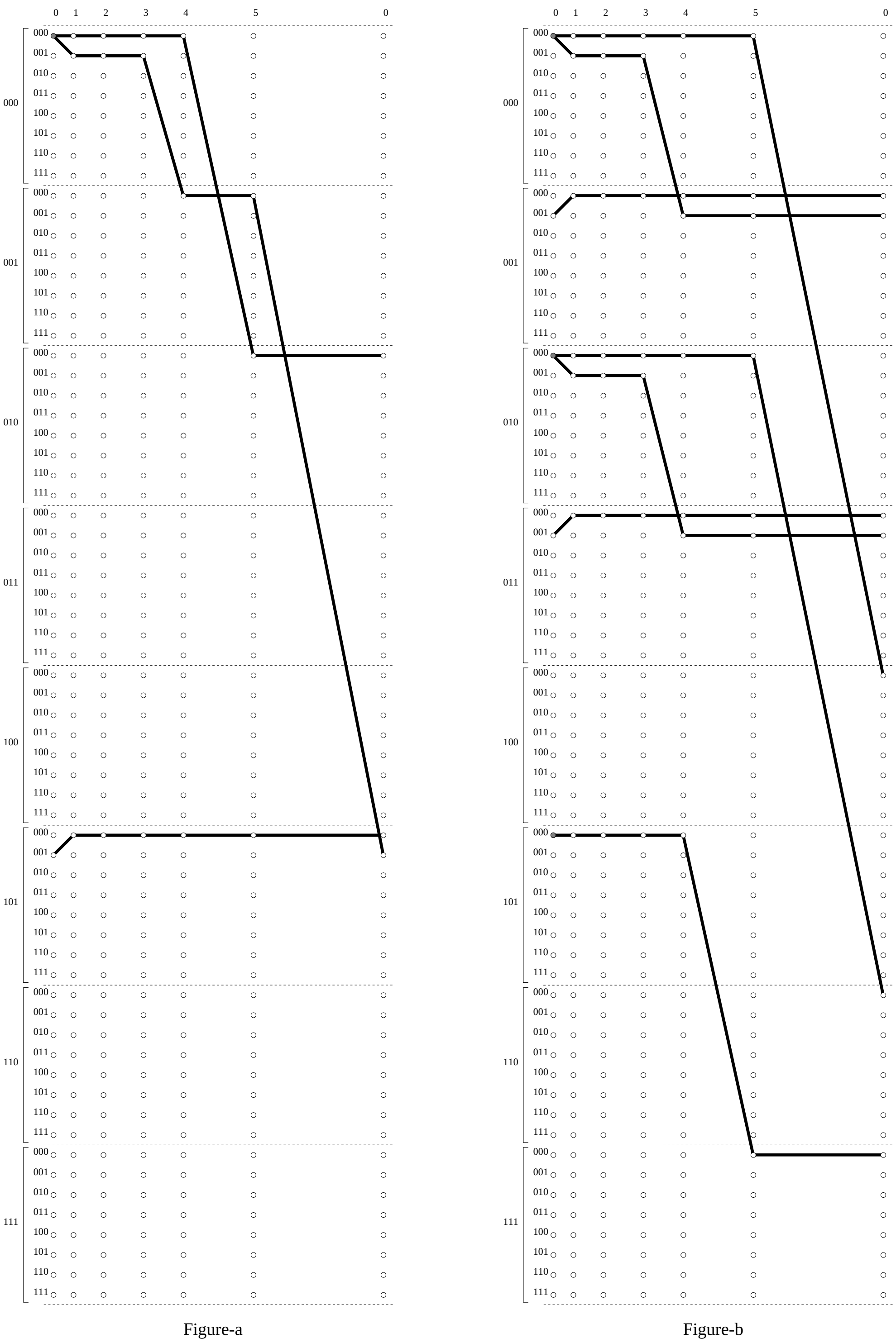

FIG. 3.16 – : *Représentation des deux premières étapes de la diffusion dans* $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,6)$. *Par souci de clarté, nous ne représentons pas les liens du graphe, mais seulement les chemins arc-disjoints utilisés aux cours des deux premières étapes.*

Ainsi après la seconde étape les sommets $(y\ 000, 0)\quad \forall y \in \mathbb{Z}_2^3$, i.e. tous les sommets 000 de chaque copie $y \in \mathbb{Z}_2^3$ du niveau 0, ont reçu l'information. Il est alors possible d'informer en deux étapes tous les sommets du niveau 0. Pour cela il suffit d'appliquer simultanément le protocole du $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$ dans chaque copie y du $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,6)$. Les figure 3.17-a et figure 3.17-b montre comment appliquer ce protocole dans une copie y. Comme le $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$ n'a que 3 niveau, les chemins traversant les niveaux 3,4 et 5 utilisent toujours des arcs de pentes 0.

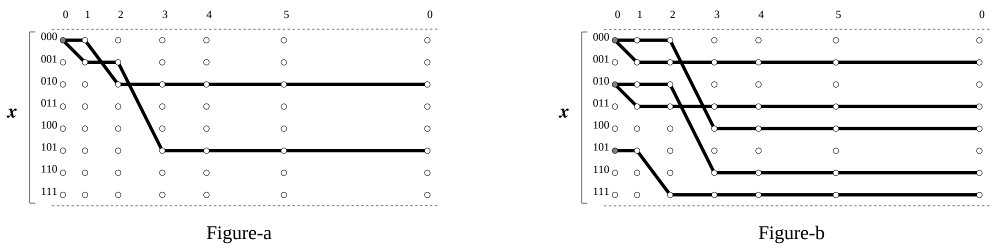

Fig. 3.17 – : *Représentation des étapes 3 (figyre-a) et 4 (figure-b) de la diffusion dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,6)$. On applique simultanément sur chaque copie $x \in \mathbb{Z}_2^3$ les deux premières étapes de la diffusion dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$.*

Remarque. Après ces 4 premières étapes, tous les sommets du niveau 0 connaissent l'information. Il est alors possible d'achever le protocole en $\lceil \log_3 6 \rceil = 2$ étapes grâce à la proposition 3.3.12 et le protocole sur le cycle. Le coût global de la diffusion est $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)) = 6$.

En fait, il est possible d'induire un tel protocole sur le Butterfly $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3i)$. Pour cela il faut remarquer que le graphe Butterfly $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ possède une décomposition récursive qui est détaillée en annexe D. Cette décomposition montre que les $n+1$ premiers niveaux de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n+1)$ forment d sous-graphes sommets disjoints chacun isomorphique à $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$. Cette relation reste bien entendue valable lorsque $d = 2$. A partir de cette propriété, on établit la relation de récurrence suivante :

Proposition 3.3.13 — *Le nombre d'étapes nécessaires pour effectuer la première phase de la diffusion dans le graphe $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,p+q)$ est :*

$$b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,p+q)) = b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,p)) + b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,q)).$$

En particulier, nous avons :

$$\begin{aligned} b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3i)) &= b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3(i-1))) + b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)) \\ &= i \cdot b_\alpha^1(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)) = 2i \end{aligned}$$

Coût du protocole lorsque $n = 3i$

Nous analysons ici le coût de notre protocole sur le graphe $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ avec $n = 3i$. Sur un tel réseau, le nombre optimal d'étapes est $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3i)) = \lceil \log_3(3i) + 3i\log_3 2\rceil$. Comme nous l'avons déjà fait remarquer, il semble difficile de trouver un protocole atteignant cette borne. C'est pourquoi nous avons proposé un protocole récursif non optimal mais efficace. Rappelons que l'idée est d'effectuer la diffusion en deux phases. La première consiste à informer les sommets du niveau 0. Puis durant une seconde phase, on achève le protocole en effectuant des simulations de diffusion optimale dans le cycle (ligne du Butterfly). Le nombre minimum d'étapes d'un tel protocole est $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3i)) = \lceil 3i\log_3 2\rceil + \lceil \log_3(3i)\rceil$. Or le protocole récursif que nous proposons à un coût de $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3i)) = i\lceil 3\log_3 2\rceil + \lceil \log_3(3i)\rceil = 2i + \lceil \log_3(3i)\rceil = \frac{2}{3}n + \lceil \log_3(3i)\rceil$. Comme $\frac{2}{3} \approx 0,666$ et que $\log_3 2 \approx 0,630$, notre protocole est donc proche de l'optimal.

Coût du protocole pour n quelconque

Dans le graphe $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ pour n quelconque, on effectue la division euclidienne de n par 3, i.e. $n = 3p + r$ avec $0 \leq r < 3$, nous savons alors par induction que nous pouvons informer le premier niveau en $2(p+1) = 2\left\lceil \frac{n}{3}\right\rceil$ étapes, tandis que la diffusion dans les cycles s'effectue de façon optimale. Le nombre total d'étapes pour effectuer les deux phases de notre protocole est donc $b_\alpha(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)) = 2\left\lceil \frac{n}{3}\right\rceil + \lceil \log_3 n\rceil$.

Remarque. Considérons la classe des algorithmes récursifs fonctionnant en deux phases : la première pour informer tous les sommets d'un niveau, l'autre pour simuler la diffusion optimale sur les cycles. L'algorithme que nous avons présenté ici pour le $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ fait partie de cette classe. Comme nous savons toujours réaliser la seconde phase de façon optimale, nous examinons ici uniquement la première phase. Notre induction commençant sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$ nous obtenons pour la première phase un coefficient de $\frac{2}{3}n$ lorsque n est un multiple de 3, pour le nombre d'étapes. Nous avons vu que cette valeur est relativement bonne car proche de $\log_3 2$. Néanmoins on peut se demander s'il n'aurait pas été plus judicieux de commencer l'induction avec une autre valeur de n afin d'obtenir un coefficient encore plus proche du $\log_3 2$. On peut montrer qu'il est toujours possible de s'approcher de plus en plus de $\log_3 2$, mais que cela nécessite de démarrer l'induction avec un n de plus en plus grand. Supposons que l'on démarre l'induction sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,p)$, alors le nombre d'étapes nécessaire pour effectuer la première phase sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ avec $n = ip$ sera noté $f(p)\cdot n$. Nous donnons dans le tableau 3.2 les premières valeurs de la fonction $f(p)$.

Ce tableau montre que la première valeur, meilleure que $f(3)$, est $f(11)$. Cela signifie, qu'il faudrait effectuer la première phase de diffusion de façon optimale

p	2	3	4	5	6
$f(p)$	1	$\frac{2}{3} \approx 0,666$	$\frac{3}{4} = 0,75$	$\frac{4}{5} = 0,8$	$\frac{4}{6} = \frac{2}{3}$
p	7	8	9	10	**11**
$f(p)$	$\frac{5}{7} \approx 0,714$	$\frac{6}{8} = 0,75$	$\frac{6}{9} = \frac{2}{3}$	$\frac{7}{10} = 0,7$	$\boldsymbol{\frac{7}{11} \approx 0,636}$
p	12	13	14	15	16
$f(p)$	$\frac{8}{12} = \frac{2}{3}$	$\frac{9}{13} \approx 0,692$	$\frac{9}{14} \approx 0,642$	$\frac{10}{15} = \frac{2}{3}$	$\frac{11}{16} \approx 0,687$

TAB. 3.2 – : *Tableau des premières valeurs de $f(p)$. En supposant que l'induction commence sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,p)$, alors le nombre d'étapes nécessaire pour effectuer la première phase dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ avec $n = ip$ est de la forme $f(p) \cdot n$.*

sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,11)$, afin d'induire un algorithme récursif ayant un meilleur coût que le notre. Le calcul montre qu'il faut aller jusqu'à $f(30) \approx 0,633$ pour obtenir la première valeur meilleure que $f(11)$.

Raffinement de notre protocole

Nous avons vu que pour appliquer notre protocole sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$ avec n quelconque, il fallait commencer par effectuer la division euclidienne de n par 3, soit $n = 3p + r$ avec $0 \leq r < 3$. Nous allons voir comment améliorer de façon simple le nombre d'étapes de la première phase de notre protocole pour certaine valeurs de n. Pour cela utilisons l'exemple du $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,7)$. Sur ce graphe, la première phase de notre protocole nécessite 6 étapes. Or, en remarquant que le nombre d'étapes sur le $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,6)$ est 4, nous montrons qu'il est possible de construire simplement un protocole n'utilisant que 5 étapes sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,7)$. Le protocole utilisé est le suivant : comme sur chaque niveau de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,7)$ il y a $2 \times 64 = 128$ sommets, nous commençons par couper en deux parties $\mathcal{P}_1$ et $\mathcal{P}_2$ le graphe $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,7)$ où $\mathcal{P}_1$ (resp. $\mathcal{P}_2$) est le sous-graphe engendré par les sommets (x,l) avec $0 \leq x \leq 63$ (resp. $64 \leq x \leq 127$) et $l \in \mathbb{Z}_n$. Alors, nous commençons par informer en 4 étapes les 64 sommets du niveau 0 de $\mathcal{P}_1$. Pour cela, nous appliquons simplement l'algorithme de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,6)$ sur $\mathcal{P}_1$. Alors, il est clair qu'en une étape, les 64 sommets du niveau 0 de $\mathcal{P}_1$ peuvent informer les 64 autres sommets de niveau 0 de $\mathcal{P}_2$ le long de chemins arc-disjoints. Il suffit par exemple que le sommet $(0,0)$ trouve un chemin quelconque vers un sommet du niveau 0 de $\mathcal{P}_2$, et que chaque autre sommet de $\mathcal{P}_1$ utilise un chemin parallèle à celui de $(0,0)$.

Ainsi, il est clair que pour $n = 3p+1$, le nombre d'étapes de la première phase est égal à $2\frac{n-1}{3} + 1 = 2p + 1$. Le coût global de notre protocole est donc :

$b_{F_*}^{\mathcal{R}_\star}(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n))$ avec $n=3p$ ou $n=3p+2$	
b_α	b_τ/L
$2\left\lceil\frac{n}{3}\right\rceil+\lceil\log_3 n\rceil$	b_α
$b_{F_*}^{\mathcal{R}_\star}(\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n))$ avec $n=3p+1$	
b_α	b_τ/L
$\frac{2}{3}(n-1)+1+\lceil\log_3 n\rceil$	b_α

REMARQUE. Une autre approche du problème, consiste à essayer de construire là encore un algorithme fonctionnant en deux phases, mais qui ne soit plus récursif. Une tel protocole permettrait d'être optimal pour la première phase, c'est-à-dire d'atteindre $\lceil n\log_3 2\rceil$ étapes sur $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,n)$. Nous pensons que dans ce type d'approche la méthodologie à utiliser est celle décrite en section 3.3.2.

☞ Il semble possible qu'une stratégie semblable à la notre puisse être utilisé dans la version non orienté du Butterfly $\mathcal{WBF}(2,n)$. En particulier, la seconde phase consistant à diffuser sur les cycles s'effectuera en $\lceil\log_{2d+1} n\rceil$. Pour cela, il suffit d'appliquer notre protocole de la version orienté simultanément dans les deux directions dans le graphe non orienté. Il resterait alors à déterminer un protocole efficace pour la première phase, consistant à informer tous les sommets d'un même niveau. En induisant, de manière semblable à la notre, un algorithme récursif à partir du protocole optimal pour la première phase sur $\mathcal{WBF}(2,n)$ décrit en figure 3.18, nous obtiendions un coût en $\lceil\frac{n}{2}\rceil$ étapes.

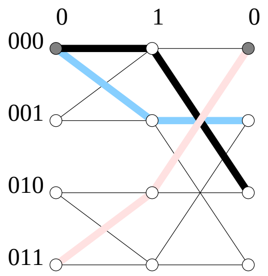

FIG. 3.18 – : *Représentation de la première phase du protocole de diffusion sur le Butterfly $\mathcal{BF}(2,2)$ non orienté à partir du sommet $(0,0)$. Cette phase qui consiste à informer tous les sommets de niveau 0 s'effectue en une étape (donc de façon optimale).*

Cette valeur est a comparer au $\log_5 2 = 0,430$. Le calcul montre que pour obtenir un meilleur coût que $\lceil\frac{n}{2}\rceil$ étapes, il faudrait commencer l'induction sur $\mathcal{WBF}(2,9)$, le coût de la première phase serait alors de $4\lceil\frac{n}{9}\rceil$.

3.4 La diffusion avec fonction de routage imposée dans le modèle $1-$port

3.4.1 Le tore de dimension 2 dans le modèle F_1

En étudiant le problème de la diffusion partielle, McKinley, Xu, Esfahanian et Ni [21] ont proposé une méthode utilisant une fonction de routage de type e-cube (correspondant à un routage du type XY sur le tore et que l'on notera $\mathcal{R}_{XY}$) et permettant de "diffuser" un message à un ensemble de N processeurs en $\log_2(N+1)$ étapes. Bien que cette méthode soit initialement prévue pour effectuer des diffusions partielles, elle peut être appliquée au cas d'une diffusion [20, 22].

L'idée est basée sur le tri des destinations, i.e. l'ensemble des processeurs dans le cas d'une diffusion, suivant l'ordre lexicographique des dimensions pour former une séquence Φ. Le processeur source scinde successivement en deux la séquence Φ. Si la source se trouve dans la demie partie inférieure (respectivement supérieure) de Φ, elle envoie un message au plus petit processeur de la demie partie supérieure (respectivement inférieure). Ce processeur va alors à son tour jouer le rôle de source et retransmettre le message aux autres sommets en utilisant la même stratégie. A chaque message est associée la liste des sommets destinations qu'il doit atteindre.

Remarque. Cette stratégie est aussi applicable aux grilles et aussi aux hypercubes et génère des chemins arc-disjoints.

Dans le cas des tores, il faut légèrement changer la façon dont est divisée la séquence Φ en effectuant une rotation des adresses de telle sorte que l'adresse de la source apparaisse en première position dans la liste Φ [27]. En appliquant cet algorithme on obtient :

$b_{F_1}^{\mathcal{R}_{XY}}(TM(l_1,l_2))$		
b_α	b_δ	b_τ/L
$\lceil\log_2(N+1)\rceil$	$\lceil\log_2(N+1)\rceil D$	b_α

3.4.2 La grille de dimension 2 dans le modèle F_1

Barnett, Payne, van de Geijn et Watts [2] ont généralisé leur algorithme de diffusion, présenté en section 3.2.5, au cas des grilles $M(p_1,p_2)$ dont le nombre de processeurs par dimension n'est pas une puissance de deux. La grille est numérotée de façon classique, c'est-à-dire, soit par colonne, soit par ligne, soit en serpentin. Leur algorithme de diffusion s'exécute en fait sur un chemin virtuel de $N=p_1p_2$ processeurs, numérotés de 0 à $N-1$ de la gauche vers la droite (voir figure 3.19). Ce chemin est plongé dans la grille en utilisant le plongement induit par la numérotation des sommets de la grille et du chemin. Leur algorithme divise les processeurs du chemin virtuel en deux partitions de taille approximativement égale. La racine émet le message vers le processeur le plus à droite (respectivement à gauche) de la nouvelle partition ne contenant pas la racine, si la nouvelle partition est à gauche (respectivement à droite) de la racine. L'algorithme s'exécute de façon récursive dans la partition contenant la racine. Dans l'autre partition, le sommet venant de recevoir le message devient la racine, et envoie le message uniquement vers la gauche (respectivement la droite). La figure 3.19 illustre cet algorithme sur une grille 3×3, numérotée par ligne.

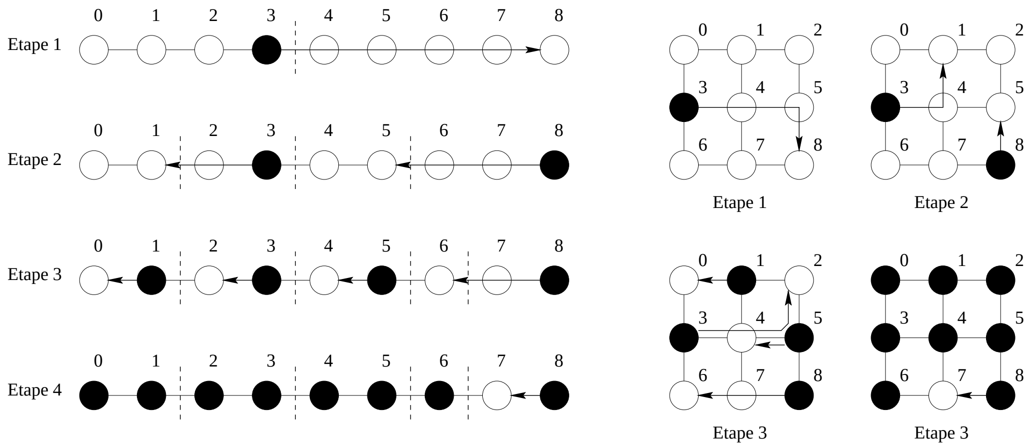

FIG. 3.19 – : *Illustration d'une diffusion dans une grille* 3×3. *L'algorithme de diffusion opère en fait sur une chaîne de longueur* 9.

Cet algorithme requiert $\lceil \log_2 N \rceil$ étapes. Les auteurs montrent que chaque étape est exempte de conflits si un routage XY, que l'on notera $\mathcal{R}_{XY}$, est employé. On obtient comme temps de diffusion :

$b_{F_1}^{\mathcal{R}_{XY}}(M(p_1,p_2))$		
b_α	b_δ	b_τ
$\lceil \log_2 N \rceil$	$\lceil \log_2 N \rceil D$	b_α

3.4.3 L'hypercube dans le modèle F_1

Comme nous l'avons déjà signalé, l'algorithme de diffusion classique nommé SBT (pour *Spanning Binomial Tree*) [13, 15] développé en mode commutation de messages atteind les bornes inférieures en commutation de circuits (Cf. section 3.2.1), sur le nombre d'étapes et les longueurs de chemins utilisés. Cet algorithme utilise un routage e-cube que l'on notera $\mathcal{R}_{e-cube}$. La figure 3.20 illustre cet algorithme pour un hypercube de dimension 4.

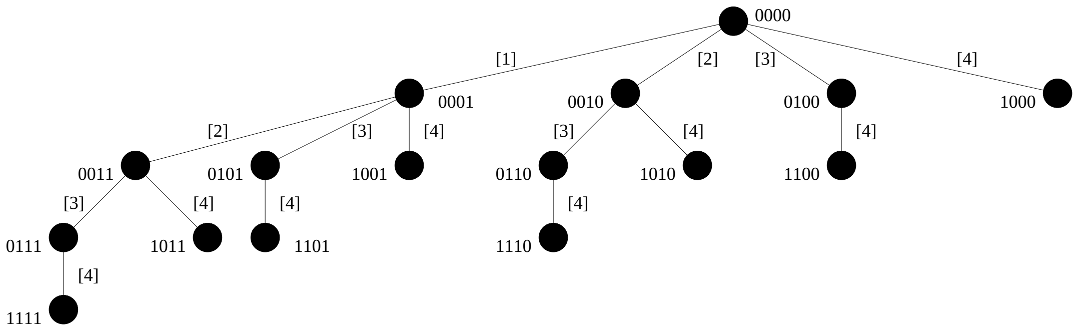

FIG. 3.20 – : *Algorithme de diffusion SBT dans le mode F_1 sur un hypercube $H(4)$.*

La complexité de cet algorithme est :

$b_{F_1}^{\mathcal{R}_{e-cube}}(H(n))$		
b_α	b_δ	b_τ/L
$\log_2 N$	$\log_2 N$	b_α

3.5 La diffusion avec fonction de routage imposée dans le modèle $\Delta-$ports

3.5.1 Le tore de dimension 2 et 3 dans le modèle F_*

Tsai et McKinley [36] ont mis en œuvre un algorithme pour les tores basé sur la notion d'ensemble dominant étendu (voir la section 3.5.2 sous le même type de contrainte). Leur algorithme utilise une fonction de routage déterministe de type XYZ, que l'on notera $\mathcal{R}_{XY}$, avec 3 types de canaux virtuel par dimension.

Pour le tore de dimension 2, de coté en puissance paires de 2, les auteurs obtiennent $b_{F_*}^{\mathcal{R}_{XY}}(TM(2^{2k})^2) \leq 2k(\alpha + L\tau) + \frac{4}{3}(2^{2k} - 1)\delta$, soit :

$b_{F_*}^{\mathcal{R}_{XY}}(TM(2^{2k})^2)$		
b_α	b_δ	b_τ/L
$\log_\Delta N$	$\frac{4}{3}(D-1)$	b_α

Pour le tore de dimension 2, de coté en puissance impaires de 2, ils obtiennent $b_{F_*}^{\mathcal{R}_{XY}}(TM(2^{2k+1})^2) \leq (2k+1)(\alpha + L\tau) + (2^{2k+2} - 2^{k+2} + 2)\delta$, soit :

$b_{F_*}^{\mathcal{R}_{XY}}(TM(2^{2k+1})^2)$		
b_α	b_δ	b_τ/L
$\log_\Delta N$	$2D - 4\sqrt{\frac{D}{2}} + 2)$	b_α

Enfin pour le tore de dimension 3, avec $7 \times 6^m + 1 \leq z \leq 7 \times 6^{m+1}$, le coût de leur protocole est $b_{F_*}^{\mathcal{R}_{XY}}(TM(2^{k+2}, 2^{k+2}, z)) \leq ((k+2) + (m+2))(\alpha + D\delta + L\tau)$, soit :

$b_{F_*}^{\mathcal{R}_{XY}}(TM(2^{k+2}, 2^{k+2}, z))$ avec $7 \times 6^m + 1 \leq z \leq 7 \times 6^{m+1}$		
b_α	b_δ	b_τ/L
$O(\log_{\Delta-2} N)$	$O(\log_{\Delta-2} N)D$	b_α

REMARQUE. Les algorithmes développés par Tsai et McKinley n'atteignent pas les bornes inférieures en nombre d'étapes. Cela est principalement dû au fait que la fonction de routage utilisée est déterministe et ne permet pas d'employer tous les chemins possibles pour connecter deux sommets distants.

3.5.2 La grille de dimension 2 et 3 dans le modèle F_*

Dans le cas des grilles de dimension 2, Tsai et McKinley [34, 35] ont proposé une méthode basée sur les ensembles dominants (*dominating sets*) et ensembles dominants étendus (*extended dominating sets*).

DEFINITION. Etant donné un graphe $G = (V, E)$, un sous-ensemble $V' \subseteq V$ est un *ensemble dominant* si tout sommet $v \in V$ est soit dans V', soit connecté à au moins un sommet de V' par une arête de E.

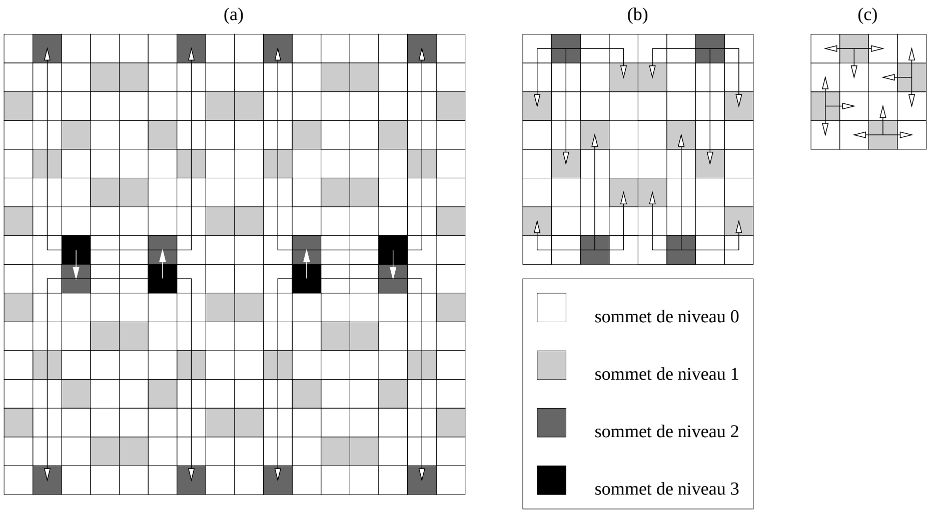

FIG. 3.21 – : *Ensembles dominants étendus dans une grille* 16×16. (a) *Les 4 sommets de niveau 3 informent les sommets de niveau 2 dans la grille* 16×16. (b) *Au sein de chaque sous-grille* 8×8, *les 4 sommets de niveau 2 informent les sommets de niveau 1.* (c) *Au sein de chaque sous-grille* 4×4, *les 4 sommets de niveau 1 informent les sommets de niveau 0.*

Tsai et McKinley généralisent cette notion d'ensemble dominant et introduisent la notion d'ensemble dominant étendu en permettant que les sommets de $V - V'$ soient dominés par les sommets de V' par des chemins arc-disjoints deux à deux. Si les sommets de V' connaissent l'information, alors ils peuvent informer tous les sommets de $V - V'$ en une étape (voir figure 3.21). Le but est donc de trouver une suite d'ensembles dominants étendus telle que les sommets d'un ensemble dominant de niveau i dominent les sommets de l'ensemble dominant de niveau $i-1$. Leur stratégie consiste à trouver des schémas d'ensembles dominants récursifs qui forment des motifs réguliers. La figure 3.21 illustre un exemple sur une grille 16×16. Dans le cas d'une grille $M(2^k, 2^k)$, cet algorithme nécessite $k+2$ étapes. Le facteur additif 2 étant dû à deux étapes d'initialisation nécessaires dans le cas où la source n'appartient pas à un ensemble dominant. En appliquant leur algorithme on obtient :

$$\begin{aligned} b_{F_*}^{\mathcal{R}_{XY}}(M(2^k)^2) &\leq 2(\alpha + (2^{k+1}-2)\delta + L\tau) + \sum_{i=1}^{k} \Big(\alpha + (2^{i+1}-2)\delta + L\tau\Big) \\ &\leq (k+2)\alpha + \Big(2^{k+3} - 2k - 8\Big)\,\delta + (k+2)L\tau \end{aligned}$$

Soit,

$b_{F_*}^{\mathcal{R}_{XY}}(M(2^k)^2)$		
b_α	b_δ	b_τ/L
$2 + \log_\Delta N$	$4D - 2k$	b_α

Tsai et McKinley ont appliqué cette notion au cas des grilles de dimension 3 [33]. Le nombre d'étapes nécessaires à leur algorithme dans le cas d'une grille $M(2^{k+2}, 2^{k+2}, 4.3^m)$ est $k+m+4$ ce qui donne comme coût :

$b_{F_*}^{\mathcal{R}_{XY}}(M(2^{k+2}, 2^{k+2}, 4.3^m))$		
b_α	b_δ	b_τ/L
$1 + \log_{\Delta-2} N$	$(1 + \log_{\Delta-2} N)D$	b_α

3.5.3 L'hypercube dans le modèle F_*

Dans toute la partie traitant de la diffusion dans l'hypercube, nous supposerons que l'initiateur dans $H(n)$ est le sommet 0^n. Ceci ne restreint en rien la généralité du problème, car l'hypercube est un graphe de Cayley et donc sommet transitif.

Nous dirons également, en utilisant la terminologie de [32], qu'un routage est **ascendant** (resp. **descendant**) dans l'hypercube, si pour toute paire de sommet (x, y), le chemin de x à y est un routage des **plus courts chemins** et si les coordonnées sont changées dans l'ordre **croissant** (resp. **décroissant**) des dimensions. Un tel routage **ascendant** (resp. **descendant**) est appelé routage **"e-cube"** et sera noté $\mathcal{R}_{e-cube}$.

Exemple. Si le sommet 000 envoie un message au sommet 111, alors

- le chemin $(000,100),(100,110),(110,111)$ correspond à un routage e-cube ascendant,
- le chemin $(000,001),(001,011),(011,111)$, correspond à un routage e-cube descendant,
- par contre le chemin $(000,010),(010,110),(110,111)$ ne correspond pas à un routage e-cube.

Remarque. Il a été montré par Dally et Seitz [6] que le routage e-cube est sans interblocage. C'est pour cela qu'une telle fonction de routage est très souvent utilisée dans les algorithmes de communications de l'hypercube [32].

Tout d'abord signalons que les deux algorithmes, présentés en section 3.3.8-i et ii, s'appliquent aussi avec une fonction de routage e-cube.

i) Protocole utilisant des hypercubes disjoints

Dans la thèse de Kodate [18], il est montré que l'approche récursive (Cf. section 3.3.8-i) est avantageuse, car il est facile d'exhiber les chemins arc-disjoints utilisés à chaque étape. En effet, l'avantage de ce découpage est que l'on peut trouver une bijection entre les d dimensions et les d sous hypercubes (autre que celui où les k coordonnées fixées valent 0) de sorte que l'hypercube H_i contienne son premier 1 dans la dimension i. Si on prend comme représentant x_i de H_i, le sommet dont les coordonnées non fixées valent 0 ; et si P_i est le plus court chemin de 0^n à x_i obtenu en changeant d'abord la dimension i puis les autres dans l'ordre croissant, alors les chemins P_i sont deux à deux arc-disjoints. Notons que leur longueur est k et en fait ils correspondent à un routage ascendant de type e-cube. Ainsi, avec un tel routage le résultat est identique qu'en section 3.3.8-i, c'est-à-dire :

$b_{F_*}^{\mathcal{R}_\star}(H(n))$	
b_α	b_τ/L
$\Theta\left(\frac{n}{\log_2(n+1)}\right) = \Theta\left(\log_{\Delta+1} N\right)$	b_α

Ce résultat est aussi décrit dans [17] où Ho et Kao donnent une manière très particulière d'exhiber les $n+1$ sous hypercubes H_i et leurs représentants x_i de sorte que les chemins entre x et x_i soient ceux du routage e-cube. Le coût de la diffusion est lui aussi asymptotiquement optimal (identique à celui de la partie iii précédente).

Remarque. Ce résultat n'est optimal qu'asymptotiquement. Ho et Kao montrent de plus, que pour $n = 5$, si l'on impose un routage e-cube, alors tout protocole nécessite au moins trois étapes, alors que l'on a vu (Cf. section 3.3.8-i) que si l'on n'impose pas de condition sur le routage, on peut en fait effectuer la diffusion de façon optimale en deux étapes.

Bibliographie

[1] M. Barnett, D. G. Payne, and R. van de Geijn. Optimal broadcasting in mesh-connected architectures. Technical Report TR-91-38, Department of Computer Sciences, University of Texas, December 1991.

[2] M. Barnett, D.G. Payne, R.A. van de Geijn, and J. Watts. Broadcasting on Meshes with Wormhole Routing. *Journal of Parallel and Distributed Computing*, 35(2):111–122, June 1996.

[3] J-C. Bermond, P. Michallon, and D. Trystram. Broadcasting in wraparound meshes with parallel monodirectional links. *Parallel Computing*, 18:639–648, 1992.

[4] J. Bruck, C-T. Ho, S. Kipnis, and D. Weathersby. Efficient algorithms for all-to-all communications in multi-port message-passing systems. In *Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA '94)*, pages 298–309, Cape May, New Jersey, June 1994. ACM.

[5] J. Cohen, P. Fraigniaud, J-C. König, and A. Raspaud. Complexité de la diffusion en mode commutation de circuit. In R. Castanet and J. Roman, editors, *Renpar'8*, pages 49–52, Bordeaux, France, Mai 1996.

[6] W.J. Dally and C.L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transsaction on Computers*, C-36(5):547–553, May 1987.

[7] O. Delmas and S. Perennes. Diffusion en mode commutation de circuits. In R. Castanet and J. Roman, editors, *Proceedings of the 8th RenPar Conference*, pages 53–56, Bordeaux, France, May 1996.

[8] O. Delmas and S. Perennes. Diffusion en mode commutation de circuits dans les tores de dimension k. Laboratoire I3S - CNRS URA 1376 - Accepté avec révisions à la revue Technique et Science Informatiques. Hermès, AFCET, Paris., 1997.

[9] A.M. Farley. Minimum-time line broadcast networks. *Networks*, 10:59–70, 1980.

[10] E. Fleury. *Communications, routage et architectures des machines à mémoire distribuée - Autour du routage wormhole.* Thèse de doctorat, Université de Lyon, Ecole Normale Supérieure de Lyon, 1996.

[11] P. Fraigniaud. Performance analysis of broadcasting hypercubes with restricted communication capabilities. *Journal of Parallel and Distributed Computing*, 16:15–26, 1992.

[12] P. Fraigniaud and C. Laforest. Disjoint spanning trees of small depth. In G.R. Joubert, D. Trystram, F.J. Peters, and D.J. Evans, editors, *ParCo '93*, pages 105–112, 1993.

[13] P. Fraigniaud and E. Lazard. Methods and problems of communication in usual networks. *Discrete Applied Mathematics*, 53:79–133, 1994. Special volume proceedings international workshop on broadcasting and gossiping 1990.

[14] P. Fraigniaud and J. Peters. Structured communication in torus networks. In IEEE, editor, *28th Annual Hawaii International conference on system sciences*, pages 584–593, Hawaii, 1995.

[15] C-T. Ho and M. T. Raghunath. Efficient communication primitives on circuit-switched hypercubes. In *6th Distributed Memory Computing Conference*, pages 390–397. IEEE, April 1991.

[16] C.T. Ho and S.L. Johnsson. Optimum broadcasting and personalized communication in hypercubes. *IEEE Transactions on Computers*, 38(9):1249–1268, 1989.

[17] C.T. Ho and M.Y. Kao. Optimal broadcast in all-port wormhole-routed hypercube. *IEEE Transactions on Parallel and Distributed Systems*, 6(2):200–204, February 1995.

[18] T. Kodate. *Communications stucturées dans les réseaux d'interconnexion*. Thèse de doctorat, Université de Nice - Sophia Antipolis, Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis CNRS URA 1376, 1996.

[19] P. K McKinley and C Trefftz. Efficient broadcast in all-port wormhole-routed hypercube. In *International Conference on Parallel Processing (ICPP '93)*, volume II, pages 288–291, St. Charles, Illinois, August 1993. (see also Technical Report MSU-CPS-ACS-6).

[20] P. K. McKinley, Y-J. Tsai, and D. F Robinson. A survey of collective communication in wormhole-routed massively parallel computers. Technical Report MSU-CPS-95-35, Michigan State University, East Lansing, Michigan 48824, June 1994.

[21] P. K. McKinley, H. Xu, A. H. Esfahanian, and L. M. Ni. Unicast-based multicast communication in wormhole-routed networks. *IEEE Transactions on Parallel and Distributed Systems*, 5(12):1252–1265, December 1994.

[22] P.K. McKinley, Y-J. Tsai, and D.F. Robinson. Collective Communication in Wormhole-routed Massively Parallel Computers. *IEEE Computer*, 28(12):39–50, December 1995. Revised as MSU-CPS-95-6, "Collective Communication Trees in Wormhole-Routed Massively Parallel Computers,", 1995.

[23] F.J. McWilliams and N.J.A. Sloane. *The theory of Error-Correcting Codes*. North-Holland, 1977.

[24] J-Y. L. Park and H-A. Choi. Circuit-switched broadcasting in torus and mesh networks. *IEEE Transactions on parallel and distributed systems*, 7(2):184–190, February 1996.

[25] J-Y. L. Park, S-K. Lee, and H-A. Choi. Circuit-switched broadcasting in $d-$dimensional tori and meshes. In *Proceedings Eighth Int'l Parallel Processing Symposium*, Cancun, Mexico, April 26-29 1994.

[26] J.G. Peters and M. Syska. Circuit-Switched Broadcasting in Torus Networks. *IEEE Transactions on Parallel and Distributed Systems*, 7(3), March 1996.

[27] D. F. Robinson, P. K. McKinley, and B. H. C. Cheng. Optimal multicast communication in wormhole-routed torus networks. *IEEE Transactions on Parallel and Distributed Systems*, 6(10):1029–1041, October 1995.

[28] Jean de Rumeur. *Communication dans les réseaux de processeurs*. Collection Etudes et Recherches en Informatique. Masson, Paris, 1994.

[29] S. R. Seidel. Broadcasting on linear arrays and meshes. Technical Report ORNL/TM-12356, Oak Ridge National Laboratory, March 1993.

[30] C. Spencer. *Circuit-switched structured communications on toroidal meshes*. PhD thesis, Simon Fraser University, February 1994.

[31] Q. F. Stout and B. Wagar. Intensive hypercube communication, prearranged communication in link-bound machines. *Journal of Parallel and Distributed Computing*, 10:167–181, 1990.

[32] H. Sullivan and T.R. Bashkow. A large scale homogeneous, fully distributed parallel machine. In *4th Annual Symposium Comput. Architecture*, pages 105–124, March 1977.

[33] Y-J. Tsai and P. K. McKinley. Broadcast in all-port wormhole-routed $3D$ mesh networks using extended dominating sets. In *International Conference on Parallel and Distributed Systems*, pages 120–127, Hsinchu, Taiwan, December 1994.

[34] Y-J. Tsai and P. K. McKinley. A dominating set model for broadcast in all-port wormhole-routed $2D$ mesh networks. In *Eighth International Conference on Supercomputing*, Manchester, England, July 1994.

[35] Y-J. Tsai and P. K. McKinley. A extented dominating node approach to collective communication in all-port wormhole-routed $2D$ meshes. In *Scalable Hight Performance Computing Conference (SHPCC '94)*, Knoxville, Tennessee, May 1994. IEEE.

[36] Y-J. Tsai and P. K. McKinley. A broadcast algorithm for all-port wormhole-routed torus networks. In *FRONTIERS '95*, McLean, Virginia, February 1995. IEEE.

[37] J. Watts and R. van de Geijn. A pipelined broadcast for multidimensional meshes. *Parallel Processing Letters*, 5(2):281–292, 1995.

Chapitre 4

L'échange total et la multidistribution en mode commutation de circuits

✔ *Ce chapitre, partie intégrante du chapitre 2, dresse une synthèse des travaux qui nous paraissent les plus significatifs sur les problèmes de l'échange total et de la multidistribution, tout du moins lorsque l'on cherche en premier lieu à minimiser le nombre d'étapes.*

Rappelons, comme précédemment, que dans tout ce chapitre, nous considérons des protocoles de diffusion synchrones, c'est-à-dire se déroulant comme une succession d'étapes.

4.1 L'échange total – Hypothèses supplémentaires

A ce niveau, les hypothèses utilisées pour le problème de la diffusion ne sont plus suffisantes. En effet, l'échange total nécessite de décrire plus finement les contraintes technologiques dues aux routeurs. Du début jusqu'à la fin d'une diffusion, le message qui circule sur le réseau est le même, et donc, conserve toujours la même taille L. Dans le problème de l'échange total, chaque sommet du réseau doit diffuser sa propre information, i.e. le message initial de chaque sommet est différent. Par simplicité, nous utiliserons une hypothèse supplémentaire sur la taille des messages initiaux.

Hypothèse 3 — Les longueurs initiales des messages intervenant dans une opération d'échange total sont toutes identiques (de taille L).

En outre, en mode k-ports, un sommet est susceptible de recevoir k messages au cours d'une étape (par exemple, après la première étape un sommet peut avoir reçu k messages différents, chacun de longueur L). Lors de l'étape suivante, ce même sommet peut vouloir retransmettre, en parallèle sur k de ses ports, l'ensemble des nouvelles informations qu'il a reçues, plus eventuellement la sienne. Or, cette information peut avoir une taille bien supérieure à L. Il est donc nécessaire de connaître les contraintes sur les tailles des messages susceptibles de transiter sur les liens en une étape. Si cette taille est supérieure à L, il faudrait aussi prendre en compte le temps de concaténation des messages à l'intérieur des routeurs. Par souci de simplicité, nous utiliserons l'hypothèse supplémentaire suivante :

Hypothèse 4 — La taille des messages pouvant circuler sur un lien de communication est indéfinie. Le temps de concaténation de messages à l'intérieur d'un routeur est négligeable et ne sera pas pris en compte.

Remarque. Cette seconde hypothèse est réaliste lorsque les messages devant être échangés sont initialement de petites tailles. Ainsi, dans la suite de cette partie, nous nous intéresserons avant tout au problème de l'échange total de messages de petites tailles, et nous évoquerons essentiellement les protocoles essayant de minimiser le nombre d'étapes.

4.2 L'échange total sans fonction de routage imposée

4.2.1 Bornes inférieures générales

Proposition 4.2.1 — *Soit G, un graphe d'ordre N et de diamètre D, alors le temps d'échange total sous la contrainte k-ports dans G est minoré par :*

$$g_{F_k}^{\mathcal{R}^\star}(G) \geq \max(\lceil \log_{k+1} N \rceil \alpha, D\delta, \frac{N-1}{k} L\tau)$$

Nous donnons dans [12, 13], une borne inférieure non triviale sur le nombre d'étapes nécessaire à l'échange total.

Théorème 4.2.2 ([13]) — *Soit G un graphe orienté symétrique d'ordre N, de degré (entrant ou sortant) maximum Δ et d'arc-indice de transmission*[1] $\pi(G)$ *et*

1. Cf. [17] pour une définition de l'arête ou arc-indice de transmission.

soit $t_0 = \lceil \log_{\Delta+1} N \rceil$*, alors le nombre d'étapes de l'échange total sous la contrainte* F_* *est minoré par :*

$$g_\alpha(G) \geq t_0 + \log_{\Delta+1}\left(\frac{\pi(G)}{N}\right) - O(\log_{\Delta+1}\log_{\Delta+1} N)$$

Preuve. Cf. annexe B. □

4.2.2 Le cycle dans le modèle F_*

Dans le modèle F_*, il est plus pratique de considérer le circuit orienté symétrique que le cycle. Ainsi, dans cette partie nous noterons par C_N le circuit orienté symétrique d'ordre N.

Suivant le paramètre que l'on cherche à optimiser, il y a deux approches triviales, chacune favorable à l'un des paramètres de l'échange total.

1. Initialement, chaque sommet transmet son message à ses deux voisins directs. Puis chaque sommet renvoie sur son voisin de droite (resp. de gauche) le message reçu à la fin de l'étape précédente de son voisin de gauche (resp. de droite). Le coût de ce protocole est :

$g_{F_*}^{\mathcal{R}_\star}(C_N)$		
g_α	g_δ	g_τ/L
$\lfloor \frac{N}{2} \rfloor$	$\lfloor \frac{N}{2} \rfloor$	$\lfloor \frac{N}{2} \rfloor$

2. Dans un premier temps on effectue l'inverse de l'opération de diffusion, décrite en section 3.3.3, pour obtenir une concentration de tous les messages sur un seul sommet. Dans un second temps, le sommet en possession de l'ensemble de l'information effectue la diffusion de la section 3.3.3. Le coût de ce protocole est :

$g_{F_*}^{\mathcal{R}_\star}(C_N)$			
	g_α	g_δ	g_τ/L
`Phase de concentration`	$\log_3 N$	$\frac{N-1}{2}$	$\frac{N-1}{2}$
`Phase de diffusion`	$\log_3 N$	$\frac{N-1}{2}$	$N \log_3 N$
`Somme des deux phases`	$2\log_3 N$	$N-1$	$N\log_3 N + \frac{N-1}{2}$

L'approche 1 est favorable aux cas de longs messages, tandis que l'approche 2 est favorable aux petits messages.

4.2.3 Le tore de dimension k dans le modèle F_*

Avant de passer en revue les principaux travaux effectués sur le tore, nous donnons la borne inférieure sur le nombre d'étapes dans la cas d'un tore de dimension k. En appliquant le théorème 4.2.2 avec la proposition suivante :

Proposition 4.2.3 (M.C. Heydemann, J.C. Meyer, D. Sotteau [17]) — *L'arc indice de transmission du graphe orienté symétrique $TM(n)^k$ est*

$$\pi(TM(n)^k) = \frac{n^{k-1}}{2}\left\lfloor \frac{n^2}{4} \right\rfloor,$$

nous obtenons dans [12, 13] une borne inférieure sur le nombre d'étapes nécessaire à l'échange total dans le tore carré de coté n et de dimension k.

Corollaire 4.2.4 — *Soit le graphe orienté symétrique $TM(n)^k$, de degré (entrant ou sortant) $\Delta = 2k$, alors le nombre d'étapes de l'échange total sous la contrainte F_* est minoré par :*

$$g_\alpha(TM(n)^k) \geq (k+1)\log_{2k+1}(n) - O(\log_{2k+1}\log_{2k+1} n)$$

☞ Il existe une analogie entre l'arc-indice de transmission et la bissection sommets, puisque dans [27], Klasing établit une borne inférieure du même type, en utilisant la bissection sommets (mais dans un modèle qu'il appelle *"two-way vertex-disjoint paths mode"*).

De plus, en appliquant les algorithmes classiques d'échange total développés dans le cadre du mode commutation de paquets [11, 14, 28, 31], il est possible de donner une borne supérieure du coût de l'échange total dans le tore carré de coté n de dimension 2 (d'ordre $N = n^2$).

$$g_{F_*}^{\mathcal{R}_\star}(TM(n)^2) \leq D(TM(n)^2)(\alpha+\delta) + \left\lceil \frac{(N-1)L}{4} \right\rceil \tau$$

i) Le tore de dimension 2 - Protocole ne minimisant pas le nombre d'étapes

Dans [30], Peters et Syska proposent une adaptation simple de leur algorithme de diffusion (Cf. section 3.3.5) pour effectuer un échange total. L'idée est de concentrer dans une première phase l'ensemble de l'information sur un seul sommet (en appliquant dans le sens inverse l'algorithme de diffusion). Puis dans une seconde phase, le sommet connaissant toute l'information effectue une diffusion. Le coût de cette stratégie dans le tore $TM(5^i)^2$ d'ordre N est :

$g_{F_*}^{\mathcal{R}_\star}(TM(5^i)^2)$		
g_α	g_δ	g_τ/L
$2\log_{\Delta+1} N$	$2D$	$\frac{N}{4} + N\log_{\Delta+1} N$

Cette stratégie n'est pas optimale en nombre d'étapes.

ii) Le tore de dimension 2 - Protocole minimisant le nombre d'étapes

Calvin, Perennes et Trystram [9, 10] ont développé un algorithme optimal en nombre d'étapes. Leur algorithme procède récursivement et est construit de façon similaire à l'algorithme de diffusion de Peters et Syska (Cf. section 3.3.5). Ils utilisent un pavage du tore avec des tuiles en forme de croix. La figure 4.1 illustre cet algorithme sur un tore 5×5.

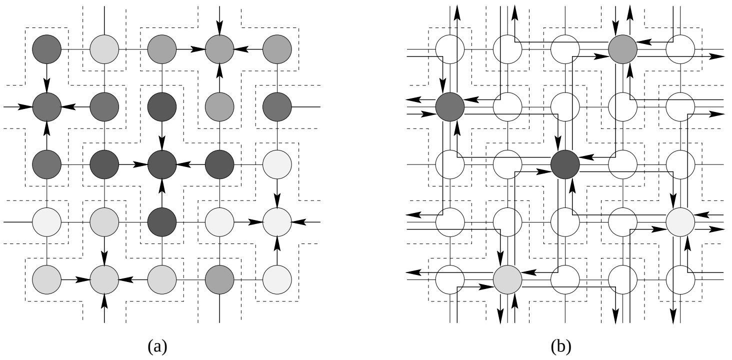

FIG. 4.1 – : *Echange total dans le tore $TM(5)^2$ en trois étapes : (a) concentration de l'information des sommets de chaque « croix » au centre de la croix. La taille des messages échangés est L et la longueur des chemins est 1. (b) échange total entre les sommets centres des cinq croix. La taille des messages échangés est $5L$ et la longueur des chemins est 3. (c) diffusion de l'information à partir des sommets centres des croix sur leur voisins directs. La taille des messages échangés est $25L$ et la longueur des chemins est 1.*

Leur algorithme, qui procède de façon récursive sur le tore $TM(5^i)^2$ d'ordre $N = 5^{2i}$, a un coût de :

$g_{F_*}^{\mathcal{R}_\star}(TM(5^i)^2)$		
g_α	g_δ	g_τ/L
$\frac{3}{2}\log_{\Delta+1} N$	$\frac{5}{4}D$	$\frac{3}{2}N - \frac{5\sqrt{N}+1}{4}$

Cet algorithme optimal en nombre d'étapes est approprié pour de petits messages. Dans le cas où la taille des messages est plus longue, les auteurs proposent une amélioration du facteur de transmission au détriment du nombre d'étapes.

$g_{F_*}^{\mathcal{R}_\star}(TM(5^i)^2)$		
g_α	g_δ	g_τ/L
$2\log_{\Delta+1} N$	$\frac{3}{2}D$	$\frac{1}{2}N + \frac{3\sqrt{N}-1}{4}$

iii) Le tore de dimension 3 - Protocole minimisant le nombre d'étapes

Dans [12, 13] nous présentons, pour de petits messages, un algorithme optimal en nombre d'étapes sur les tores $TM(7^i)^3$ de dimension 3. Les principes sont semblables à ceux développés en section 4.2.3-ii et reposent sur une technique générale que nous pouvons résumer ainsi :

Algorithme 4.2.1 — Algorithme générique pour un tore $TM(n^i)^k$

- si $i = 1$
  1. Concentrer l'information sur un ensemble de représentants.
  2. Effectuer un échange total des représentants.
  3. Distribuer l'information au reste du réseau
- sinon:
  1. Concentrer l'information sur un ensemble de représentants.
  2. Partitionner l'ensemble des représentants en familles de façon à ce que chaque famille "forme un tore" $TM(n^{i-1})^k$.
  3. Effectuer en parallèle un échange total au sein de chaque famille.
  4. Effectuer un échange total entre les familles.
  5. Distribuer l'information au reste du réseau

L'algorithme 4.2.1 fonctionne à condition de déterminer un ensemble de représentants et de familles vérifiant certaines propriétés. Il faut aussi définir un échange total des représentants lorsque $i = 1$. C'est ce que nous avons fait dans [12, 13] pour les tores de dimension 3. Pour cela, nous utilisons comme ensemble de représentants les sommets appartenant à des *"codes de Lee parfaits [29]"*. Nous montrons qu'il est possible, sur le tore de dimension 3, de réaliser à chaque étape de l'algorithme des communications le long de chemins arc-disjoints. En particulier, lorsque $i = 1$ nous obtenons sur le tore $TM(7)^3$ (i.e. $n = 7$, $k = 3$ et $i = 1$) un protocole dont le coût est :

$g_{F_*}^{\mathcal{R}_\star}(TM(7)^3)$		
g_α	g_δ	g_τ/L
$(k+1)\log_{2k+1} n = 4$	$\frac{12}{9}D = 12$	$1 + 7 + 7^2 + 7^3$

ce qui est optimal pour le nombre d'étapes et quasi-optimal pour la longueur des chemins. Grâce aux propriétés de l'ensemble des représentants choisi (celui défini comme un code de Lee parfait), nous montrons qu'il est possible d'appliquer l'algorithme 4.2.1 sur les tores $TM(7^i)^3$ (i.e. $n = 7^i$, $k = 3$, $i \geq 1$ et $N = 7^{3i}$). Dans ce cas, le coût de notre protocole est :

$g_{F_*}^{\mathcal{R}_\star}(TM(7^i)^3)$		
g_α	g_δ	g_τ/L
$(k+1)\log_{2k+1} n = 4i$	$\frac{12}{9}D$	$\left(\frac{57}{49}(7^{i-1} - 1) + 7^{3-2i} - 7^{-3i}\right)\frac{N}{6}$

Dans [13], nous examinons aussi le cas où la longueur des messages est plus importante. Nous montrons que l'on peut, à condition d'augmenter le nombre d'étapes, utiliser plus efficacement la bande passante du réseau. Le coût de ce protocole, sur le tore $TM(7^i)^3$ (i.e. $n = 7^i$, $k = 3$, $i \geq 1$ et $N = 7^{3i}$), est :

$g_{F_*}^{\mathcal{R}_\star}(TM(7^i)^3)$		
g_α	g_δ	g_τ/L
$5i$	$\frac{13}{9}D$	$\left(\frac{22}{49}(7^{i-1}-1)+19\cdot 7^{1-2i}-7^{-3i}\right)\frac{N}{6}$

iv) Le tore de dimension k - Protocole ne minimisant pas le nombre d'étapes

Dans [26] Juurlink, Rao et Sibeyn proposent plusieurs algorithmes d'échange total dans les grilles et tores de dimension k. Il est important de noter que les auteurs ne cherche pas ici à optimiser le nombre d'étapes. Ils se placent dans le cadre d'un message de longueur moyenne (i.e. ni de très grande taille, ni de très petite taille), et proposent des algorithmes ayant un bon compromis entre le nombre d'étapes et le flot d'information. Dans cette étude les auteurs ne considèrent pas les longueurs des chemins.

L'idée générale est basée sur le fait suivant : comme il existe des protocoles très efficaces pour des petits messages mais très mauvais dans le cas où les messages sont longs et qu'il existe des protocoles très efficaces pour de longs messages mais très mauvais dans le cas où les messages sont petits, alors un protocole hybride mélangeant ces deux approches devrait être efficace pour des messages de taille moyenne. Pour illustrer cette idée, les auteurs donnent dans [26] le protocole utilisé dans le cas du cycle (tore de dimension 1).

Nous avons vu à la section 4.2.2, qu'il existe deux approches triviales pour effectuer un échange total dans le cycle, l'une favorable aux longs messages (approche 1), l'autre favorable aux messages de petite tailles (approche 2). A partir de ces deux méthodes ils construisent l'algorithme 4.2.2. Pour cela on considère que chaque sommet du cycle C_N a une donnée de taille 1.

Algorithme 4.2.2 — Algorithme hybride d'échange total dans le cycle

Soit deux entiers positifs a et b.

1. Concentrer $\frac{N}{a}$ informations sur a sommets équitablement espacés en utilisant l'aproche 2. Ces sommets sont appelés `points de concentration`.
2. Durant $\lfloor \frac{a}{2} \rfloor$ étapes les points de concentration s'échangent leurs donnée de taille $\frac{N}{a}$ en utilisant l'approche 1.
3. Durant $\lceil \log_a N - 1 \rceil$ phases (une phase comprend plusieurs étapes), on augmente le nombre de points de concentration de a jusqu'à N. L'information est envoyée à $a-1$ nouveaux points entre deux points de concentration définis après l'étape 1, en $b \geq \lfloor \frac{a}{2} \rfloor$ étapes avec des paquets de taille $\frac{N}{2b-a+2}$.

Remarque. L'algorithme 4.2.2 est identique à celui de l'approche 1 lorsque $a = N$.

La figure 4.2 donne un exemple de cet algorithme pour $N = 9$, $a = 3$ et $b = 2$.

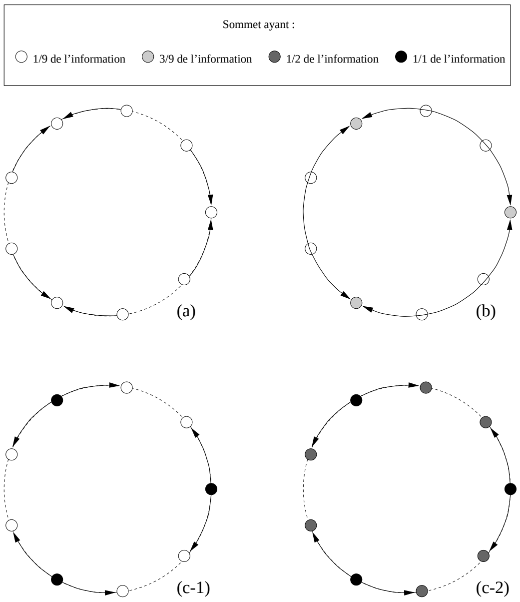

Fig. 4.2 – : *Algorithme hybride d'échange total dans le cycle. Les paramètres utilisés sont $N = 9$, $a = 3$ et $b = 2$. Au cours de la première étape (figure-a) l'information est distribuée sur 3 points de concentration. La seconde étape (figure-b) correspond à l'échange des informations entre chaque point de concentration. Enfin les figures $c-1$ et $c-2$ montrent comment les points de concentrations qui connaissent toute l'information, la renvoient, en coupant ici le message en 2 paquets, sur les autres sommets.*

Le coût de de leur protocole dépend donc des deux paramètres a et b.

	$g_{F_*}^{\mathcal{R}_\star}(C_N)$	
	g_α	g_τ/L
`Phase 1`	$\log_3 \frac{N}{a}$	$\frac{N}{2a}$
`Phase 2`	$\lfloor \frac{a}{2} \rfloor$	$\lfloor \frac{a}{2} \rfloor \frac{N}{a}$
`Phase 3`	$(\log_a(N) - 1)b$	$(\log_a(N) - 1)b \left\lceil \frac{N}{2b-a+2} \right\rceil$

Le coût global pour b_α et $\frac{b_\tau}{L}$ est la somme des coûts de chacune des trois phases. Bien entendu les meilleurs choix de a et b dépendent des autres paramètres de la machine (i.e. α, τ et L). Pour certaines valeurs fixées de ces paramètres, les auteurs trouvent par un calcul sur ordinateur les meilleurs choix de a et b. L'étude du comportement de leur algorithme montre qu'il est mieux approprié au cas de messages de longueur moyenne par rapport à l'approche 1 ou 2, ce qui est bien entendu tout à fait normal.

Les auteurs généralisent ce type d'approche aux cas des tores de dimension 2 et supérieure. Ils montrent là aussi que cette approche hybride garde un bon comportement même sur des tores de grande dimension dans le cas de messages de longueur moyenne.

Remarque. Moyennant quelques modifications mineures (nécessitant notamment d'adapter le protocole d'échange total du cycle à celui de la chaîne), les auteurs sont en mesure d'adapter l'ensemble de leurs protocoles au cas des grilles de dimension k.

4.2.4 L'hypercube dans le modèle F_*

Signalons enfin, le récent résultat de Fujita et Yamashita, puisque dans [16], les auteurs donnent un protocole asymptotiquement optimal pour le nombre d'étapes dans l'hypercube $H(n)$. En fait, ils étudient un problème plus général appelé *"group-gossiping problem"* qui correspond à effectuer un échange total entre un sous-ensemble de sommet $U \subseteq V$ de l'hypercube $H(n) = (V, E)$, et construisent pour cela un algorithme nécessitant

$$\frac{\log_2 |U|}{\log_2 n} + o\left(\frac{\log_2 |U|}{\log_2 n}\right)$$

étapes, ce qui est asymptotiquement optimal. Ce résultat reste bien entendu valable pour l'échange total, c'est-à-dire lorsque $U = V$. Le coût global est alors :

$g_{F_*}^{\mathcal{R}_\star}(H(n))$
g_α
$\frac{\log_2 N}{\log_2 n} + o\left(\frac{\log_2 N}{\log_2 n}\right)$

4.3 L'échange total avec fonction de routage imposée

4.3.1 L'hypercube dans le modèle F_*

Les algorithmes développés en mode commutation de paquets [14, 21, 34] saturant complètement les liens, atteignent les bornes inférieures sur la longueur des chemins et le flot d'information. En utilisant un routage e-cube, le coût du protocole sur l'hypercube de dimension n d'ordre $N = 2^n$ est :

$g_{F_*}^{\mathcal{R}_{e-cube}}(H(n))$		
g_α	g_δ	g_τ/L
$\log_2 N$	D	$\frac{N-1}{\Delta}$

4.3.2 L'hypercube dans le modèle F_1

Seidel [33] propose un algorithme très simple d'échange total où chaque processeur envoie une copie de son message aux $2^n - 1$ autres sommets. Il montre qu'en utilisant une fonction de routage e-cube cet algorithme se révèle être sans conflit. Le coût de cet algorithme est $g_{F_1}^{\mathcal{R}_{e-cube}}(H(n)) \leq \sum_{i=1}^{n} \binom{n}{i} (\alpha + i\delta + L\tau)$, soit :

$g_{F_1}^{\mathcal{R}_{e-cube}}(H(n))$		
g_α	g_δ	g_τ/L
$N-1$	$D2^{D-1}$	$N-1$

Cet algorithme est optimal en temps de propagation, mais pas en temps d'initialisation ni en distance de chemin emprunté.

Cependant, l'algorithme développé en mode commutation de circuit, consistant à échanger toute l'information détenue par un sommet, dimension par dimension (en utilisant un routage e-cube), obtient de meilleures performances. Le coût d'un tel protocole est $g_{F_1}^{\mathcal{R}_{e-cube}}(H(n)) \leq \sum_{i=1}^{n} (\alpha + \delta + 2^i L\tau)$, soit :

$g_{F_1}^{\mathcal{R}_{e-cube}}(H(n))$		
g_α	g_δ	g_τ/L
$\log_2 N$	D	$N-1$

4.4 La multidistribution – Notations et hypothèses supplémentaires

On trouve dans la littérature deux grandes classes d'algorithmes de multidistribution. Une première approche consiste à envoyer les messages de façon *directe*, c'est-à-dire sans étape intermédiaire entre la source et la destination des messages. La seconde approche utilise des envois de messages regroupés, et donc des retransmissions. Dans le premier cas, nous parlerons d'algorithmes **directs** et dans le second d'algorithmes **indirects**. Dans le but de bien différencier ces deux approches, nous utiliserons la notation suivante.

Notation 6 — *Le coût d'un protocole de multidistribution dans un graphe* G*, sera noté par* $m_{\mathcal{M}}^{\mathcal{R},Direct}(G)$ *dans le cas d'un algorithme de type direct et* $m_{\mathcal{M}}^{\mathcal{R}}(G)$ *dans le cas indirect.* $\mathcal{M}$ *et* $\mathcal{R}$ *ont les mêmes significations que celles définies en notation 3.*

Dans le cas d' algorithmes indirects, nous ferons l'hypothèse supplémentaire suivante :

Hypothèse 5 — Dans un algorithme de multidistribution du type indirect, les divers temps de recopies mémoire et réarrangement de messages au sein des routeurs sont négligeables et ne seront pas pris en compte.

Comme dans le cas de l'échange total, cette hypothèse est réaliste lorsque les messages sont de petite taille. De plus, dans tous les cas nous adopterons l'hypothèse 3 sur la longueur initials des messages.

4.5 La multidistribution sans fonction de routage imposée

4.5.1 Bornes inférieures générales

Proposition 4.5.1 — *Soit* G *un graphe d'ordre* N *de diamètre* $D(G)$*. Le coût de la multidistribution sous la contrainte* 1*-port est minoré par :*

$$m_{F_1}^{\mathcal{R}\star}(G) \quad \geq \quad \max\left(\lceil \log_2 N \rceil \alpha, D(G)\delta, L(N-1)\tau\right)$$

La borne inférieure sur la multidistribution dans un graphe G proposée par Fraigniaud et Lazard [14] s'applique au mode commutation de circuits sous la contrainte F_*. Ils appliquent le principe de globalité, i.e. en comparant la quantité d'information devant circuler dans le réseau par rapport à la capacité de ce dernier.

Proposition 4.5.2 ([14]) — *Soit $|\Gamma_u^i|$ le nombre de sommets à distance i d'un sommet u d'un graphe G dont la bande passante totale est $\mathcal{B}_\mathcal{G}$. On a :*

$$m_{F_*}^{\mathcal{R}_\star}(G) \quad \geq \quad \frac{\left(\sum_{u\in V}\sum_{i=1}^{D} i|\Gamma_u^i|\right)}{\mathcal{B}_\mathcal{G}} L\tau$$

De plus, les bornes inférieures sur le temps de la multidistribution dans un modèle Δ-ports sont identiques au mode de commutation de paquets.

Proposition 4.5.3 — *Soit G un graphe d'ordre N de degré Δ et de diamètre $D(G)$.*

$$m_{U_*}^{\mathcal{R}_\star}(G) \quad \geq \quad \max\left(\left\lceil \log_{\Delta+1} N \right\rceil \alpha, D(G)\delta\right)$$

4.5.2 Compromis entre le nombre d'étapes et le flot d'information

Bruck, Ho, Kipnis et Weathersby [8] ont étudié la multidistribution dans le cadre d'une machine parallèle k-ports, où tout processeur peut communiquer avec tout autre processeur. Ils s'intéressent donc seulement aux deux paramètres $m_\alpha(G)$ et $m_\tau(G)$. Leur approche suppose que les performances d'une communication point-à-point entre deux processeurs ne dépend pas de la paire de processeurs considérée, donc que la topologie sous-jacente est le graphe complet.

Les auteurs prouvent que l'optimisation simultanée des paramètres $m_\alpha(G)$ et $m_\tau(G)$ est incompatible. Pour cela, ils fournissent des bornes inférieures exprimant le nombre minimum d'étapes $m_\alpha(G)$ en fonction de la quantité de données échangées $m_\tau(G)$, c'est-à-dire le compromis[2] qu'il est nécessaire de faire entre ces deux paramètres. Clairement, ceci signifie que diminuer le flot d'information implique d'augmenter le nombre d'étapes (et inversement) d'un protocole de multidistribution (Cf. section 3.1.2). Les bornes obtenues dans le graphe complet sont valables sur un graphe G quelconque en mode commutation de circuits.

Théorème 4.5.4 ([8]) — *Soit G un graphe d'ordre N. Dans tout algorithme de multidistribution, sous la contrainte F_1,*

$$\textit{si } m_\alpha(G) = \log_2 N \textit{ alor } m_\tau(G) = \Omega(N \log_2(N) L).$$

Théorème 4.5.5 ([8]) — *Soit $d \geq 0$ et G un graphe de degré Δ et d'ordre $N = (\Delta+1)^d$. Dans tout algorithme de multidistribution, sous la contrainte F_*,*

$$\textit{si } m_\alpha(G) = \log_{\Delta+1} N \textit{ alor } m_\tau(G) \geq \frac{N \log_{\Delta+1} N}{\Delta+1} L.$$

2. "Trade-off" en américain.

Proposition 4.5.6 ([8]) — *Soit G un graphe d'ordre N de degré Δ. Dans tout algorithme de multidistribution, sous la contrainte F_*, si chaque sommet transmet exactement $\frac{N-1}{\Delta}L$ données, alors :*

$$m_\alpha(G) \geq \frac{N-1}{\Delta}.$$

4.5.3 La chaîne dans le modèle F_1

i) Algorithme de Scott

Scott étudie dans [32] des algorithmes d'échange total personnalisé du type direct dans les chaînes. En considérant le nombre de messages qui doivent être échangés entre la partie gauche et la partie droite dans un chaîne P_N, Scott obtient comme borne inférieure :

$$m_\alpha(P_N) \geq \left\lfloor \tfrac{N}{2} \right\rfloor \left\lceil \tfrac{N}{2} \right\rceil, \text{ pour les algorithmes de type direct.}$$

Il propose un algorithme d'échange total personnalisé dans un chemin P_N qui atteint la borne inférieure en nombre d'étapes. A chaque étape, quatre processeurs participent et envoient leur information suivant un cycle. Les quatre processeurs (notés A, B, C et D) sont choisis de tel sorte qu'il n'y ait pas de conflit. Pour tout processeur A et B de la première moitié de P_N, les deux autres processeurs C et D sont les processeurs symétriques de A et B respectivement (voir figure 4.3). Toutes les paires source-destination existent sauf celles entre processeurs symétriques (il y en a $\frac{N}{2}$) pour lesquelles il faut réaliser des étapes spéciales d'échange entre deux processeurs. Le cas où le nombre de processeurs est impair est similaire, sauf pour les étapes spéciales pour lesquelles trois processeurs envoient des données (voir figure 4.3).

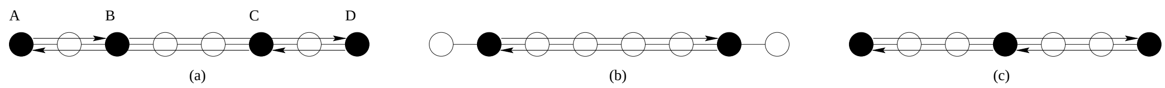

FIG. 4.3 – : *Différentes phases au cours de l'échange total personnalisé dans un chemin P_N. (a) étape normale d'échange entre 4 processeurs (N pair ou impair). (b) étape spéciale dans le cas où N est pair. (c) étape spéciale dans le cas où N est impair.*

Ainsi, lorsque $N = 2n$ est pair :

$$m_{F_1}^{\mathcal{R}_\star,Direct}(P_N) \quad \leq \quad \sum_{i=1}^{\frac{N}{2}} \Big((2i-1)(\alpha + L\tau) + 2\delta \sum_{j=1}^{i-1} 2i - 1 - j \Big)$$

soit un coût de :

$m_{F_1}^{\mathcal{R}_\star,Direct}(P_N)$ avec $N = 2n$		
m_α	m_δ	m_τ/L
$\frac{N^2}{4}$	$\frac{N^3}{8}$	$\frac{N^2}{4}$

Lorsque $N = 2n + 1$ est impair :

$$m_{F_1}^{\mathcal{R}_\star,Direct}(P_N) \leq \sum_{i=1}^{\frac{N-1}{2}} \Big(2i\,(\alpha + L\tau) + 2\delta \sum_{j=1}^{i} 2i - j + 1\Big)$$

soit un coût de :

$m_{F_1}^{\mathcal{R}_\star,Direct}(P_N)$ avec $N = 2n+1$		
m_α	m_δ	m_τ/L
$\frac{N^2-1}{4}$	$\frac{N^3+N^2-N-1}{8}$	$\frac{N^2-1}{4}$

ii) Algorithme de Takkella et Seidel

Dans [38] Takkella et Seidel développent un algorithme de multidistribution pour un chemin P_N qui s'exécute en deux phases. Soit $h = \left\lceil \frac{N-1}{2} \right\rceil$. Lors de la première étape de la première phase, chaque processeur groupe les messages destinés aux h processeurs situés à sa droite (de façon cyclique) et envoie ces h messages vers son voisin de droite. Les processeurs gardent les messages dont ils sont destinataires et retransmettent le reste. Cette opération de décalage cyclique est répétée h fois. La seconde phase est similaire à la première mais dans la direction opposée et comporte $N - h - 1$ étapes. Le temps de cet algorithme est :

$m_{F_1}^{\mathcal{R}_\star}(P_N)$		
m_α	m_δ	m_τ/L
$N-1$	D^2	$\left\lceil \frac{N}{2} \right\rceil \left\lfloor \frac{N}{2} \right\rfloor$

4.5.4 La grille de dimension 2 dans le modèle F_1

i) Algorithmes de Bokhari et Berryman

Bokhari et Berryman [4] proposent de simuler sur une grille $M(2^k)^2$, l'algorithme SBT de multidistribution développé sur hypercube (Cf. section 4.6.2). Pour cela, la grille est successivement partitionnée en deux suivant l'axe des x et l'axe des y, et les échanges de messages se font de part et d'autre de la frontière. Après la phase 1, tous les messages ont rejoint la moitié de la grille dans laquelle se trouve leur destination. Après la phase 2, ils sont dans le bon quadrant, etc. Au cours de la phase i, les échanges doivent s'effectuer en 2^{i-1} étapes pour éviter les conflits. Le temps de cet algorithme est :

$$\begin{aligned} m_{F_1}(M(2^k, 2^k)) &\leq \sum_{i=1}^{k} 2\Big(2^{i-1}\left(\alpha + 2^{2k-1}L\tau\right) + 2^{2(i-1)}\delta\Big) \\ &\leq \left(2^{k+1} - 2\right)\alpha + \frac{2}{3}(2^{2k} - 1)\delta + \left(2^{3k} - 2^{2k}\right)L\tau \end{aligned}$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2)$		
m_α	m_δ	m_τ/L
$2(\sqrt{N}-1)$	$\frac{1}{6}(D^2+4D)$	$N(\sqrt{N}-1)$

Les mêmes auteurs proposent un autre algorithme nommé *quadrant exchange* qui exécute des séquences de 3 types d'échanges, illustrées par la figure 4.4 -(a). Ces trois types d'échanges permettent à quatre sommets d'échanger entre eux leur données en trois étapes et sans conflit. A la première phase, la grille est divisée en quadrants, et chaque sommet échange ses données avec les sommets symétriques (par rapport à x, par rapport à y et par rapport au centre) dans les autres quadrants en effectuant une séquence de trois échanges. Afin d'éviter les conflits, seuls certains sommets sont autorisés à communiquer, et une phase s'effectue donc en plusieurs séquences. A la fin de la première phase, toutes les données ont transité vers leur bon quadrant. L'algorithme s'exécute alors récursivement au sein de chacun des quadrants. La figure 4.4 -(b) illustre cet algorithme sur une grille 4×4.

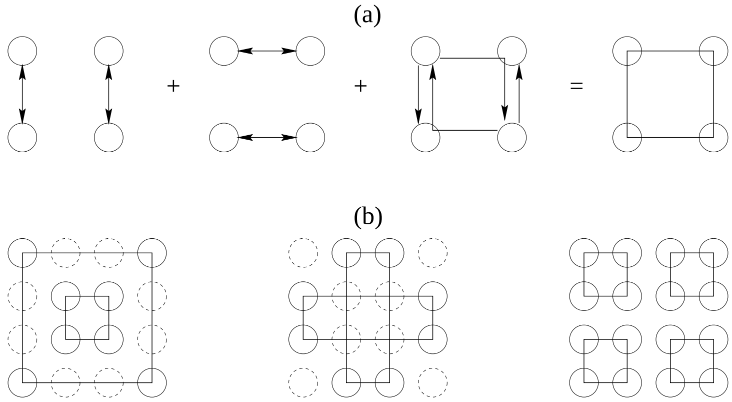

FIG. 4.4 – : *(a) La séquence de trois types d'échanges appliquée récursivement au cours de l'algorithme par quadrant. (b) Algorithme par quadrant sur une grille 4×4. La première phase s'exécute en deux étapes. La deuxième phase s'exécute en une étape.*

Le temps de cet algorithme est :

$$\begin{aligned} m_{F_1}(M(2^k,2^k)) &\leq 3\sum_{i=1}^{k} 2^{i-1}\left(\alpha + 2^{2\,k-2}L\tau + 4\left(2^i-1\right)\delta\right) \\ &\leq 3(2^k-1)\alpha + \left(8\cdot 4^k - 12\cdot 2^k + 4\right)\delta + \frac{3}{4}(2^{3k}-2^{2k})L\tau \end{aligned}$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2)$		
m_α	m_δ	m_τ/L
$3(\sqrt{N}-1)$	$2(D^2+D)$	$\frac{3}{4}N(\sqrt{N}-1)$

ii) Algorithmes de Takkella et Seidel

Takkella et Seidel [38] modifient la séquence d'échanges qui était employée dans l'algorithme de Bokhari et Berryman décrit en section 4.5.4-i en l'augmentant d'une étape mais en permettant de couvrir plus de sommets, ce qui réduit globalement le nombre d'étapes. Le temps de leur algorithme est :

$$m_{F_1}(M(2^k, 2^k)) \quad \leq \quad 4 \sum_{i=1}^{k} 2^{i-2} \left(\alpha + 2^{2\,k-2} L\tau + 6 \left(2^i - 1\right) \delta\right)$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2)$		
m_α	m_δ	m_τ/L
$2(\sqrt{N}-1)$	$2(D^2+D)$	$\frac{1}{2}N(\sqrt{N}-1)$

De plus, les auteurs proposent d'adapter leur algorithme initalement développé pour les chaînes (Cf. section 4.5.3-ii) aux grilles $M(p_1, p_2)$ en l'utilisant d'abord sur les lignes pour amener les messages dans la bonne colonne, puis sur les colonnes pour les amener à leur destination finale. Le coût de cet algorithme pour une grille $M(p_1, p_2)$ est :

$m_{F_1}^{\mathcal{R}_\star}(M(p_1, p_2))$		
m_α	m_δ	m_τ/L
$p_1 + p_2 - 2$	$(p_1-1)^2 + (p_2-1)^2$	$\left\lceil \frac{p_1}{2} \right\rceil \left\lfloor \frac{p_1}{2} \right\rfloor p_2 + \left\lceil \frac{p_2}{2} \right\rceil \left\lfloor \frac{p_2}{2} \right\rfloor p_1$

ce qui donne, dans le cas d'une grille $M(2^k)^2$:

$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2)$		
m_α	m_δ	m_τ/L
$2(\sqrt{N}-1)$	$\frac{D^2}{2}$	$\frac{1}{2}N\sqrt{N}$

iii) Algorithmes de Sundar, Jayasimha, Panda et Sadayappan.

Dans [35], Sundar, Jayasimha, Panda et Sadayappan proposent un algorithme nommé *cyclic exchange* (CE). Cet algorithme est de type indirect, i.e. il regroupe un certain nombre de données que doit envoyer un processeur à un sous-ensemble de processeurs de la grille, et les envoie à un représentant de ce sous-ensemble. Cet algorithme est illustré par la figure 4.5 dans le cas d'une grille $M(8)^2$.

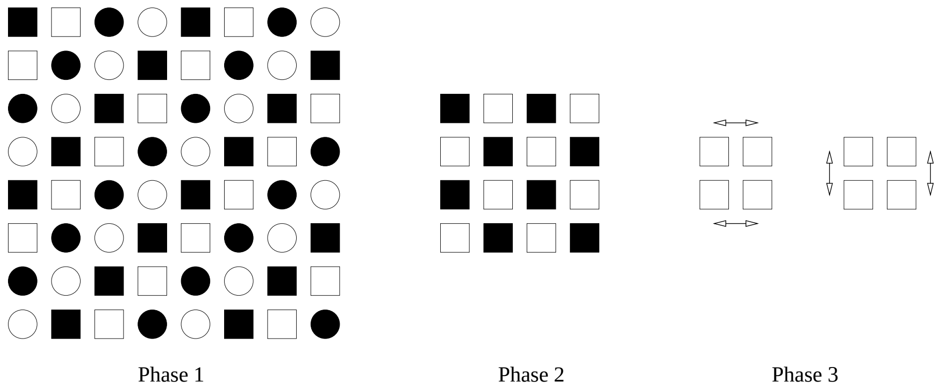

FIG. 4.5 – : *Algorithme* cyclique exchange *(CE) sur une grille* 8×8.

Sur cet exemple, l'algorithme requiert trois phases. Au cours de chaque phase i, avec $i = 0, \ldots, k-1$, les processeurs communiquent avec deux autres processeurs qui sont à distance 2^i, un sur la même ligne, et l'autre sur la même colonne. Les processeurs sont coloriés comme un échiquier de façon à entrelacer les communications pour qu'il n'y ait pas de conflit. Au cours de chaque phase i, avec $i = 1, \ldots, k-1$, une première moitié des processeurs communiquent horizontalement (par exemple les blancs sur la figure 4.5), et l'autre moitié communique verticalement (les noirs) ce qui requiert 2^{i-1} étapes. Cet ordre est ensuite inversé. La dernière phase $i = 0$, s'effectue en deux étapes. Le temps de cet algorithme est :

$$m_{F_1}(M(2^k, 2^k)) \leq \sum_{i=1}^{k-1} 2^i(\alpha + 2^i\delta + \frac{2^{2k}}{2}L\tau) + 2(\alpha + \delta + \frac{2^{2k}}{2}L\tau)$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2)$		
m_α	m_δ	m_τ/L
$\sqrt{N}$	$\frac{1}{3}\left(\frac{D^2}{4} + D + 3\right)$	$\frac{1}{2}N\sqrt{N}$

Les même auteurs présentent dans [36] un algorithme hybride construit à partir de l'algorithme CE (Cf. section 4.5.4-iii) et de l'algorithme DE (décrit en section 4.6.1). Cette approche est très similaire à celle employée dans le cas des hypercubes [3, 19, 20] (voir section 4.6.2-ii sous la même contrainte). Ils remplacent les K dernières phases de l'algorithme CE par une multidistribution réalisée à l'aide de l'algorithme DE ($K = 0$ correspond à l'algorithme CE, et $K = N$ correspond à l'algorithme DE). Le temps de cet algorithme, pour $K > 0$ est :

$$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2) \leq 2^{3K-2}(\alpha + 2^{K-1}\delta + 2^{2k-2K}L\tau) + \sum_{i=1}^{k-K} 2^i(\alpha + 2^i\delta + \frac{2^{2k}}{2}L\tau)$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(M(2^k)^2)$		
m_α	m_δ	m_τ/L
$2^{k-K+1}+2^{3K-2}-2$	$\frac{4^{k-K+1}}{3}+\frac{2^{4K}}{8}-\frac{4}{3}$	$2^k\left(2^{k-K}+2^{K-2}-1\right)$

☞ Sundar et al. remarquent dans leur article [36] que pour des grilles et/ou des messages de taille importante, il peut être avantageux, lors de la mise en œuvre des algorithmes, d'insérer des barrières de synchronisation entre les phases, voire entre chaque étape, afin de réduire les conflits qui peuvent survenir au sein d'une machine parallèle du fait de l'asynchronisme des processeurs. De plus, ils considèrent que les temps d'initialisation des différentes étapes au sein d'une même phase sont « recouverts ». Cela permet de diminuer avantageusement le facteur d'initialisation, mais nous n'en avons pas tenu compte dans notre modèle.

Remarque. Par rapport aux algorithmes de multidistribution indirects que nous venons de présenter, l'algorithme CE de Sundar et al. est celui qui obtient les meilleures performances que cela soit en nombre d'étapes ($\sqrt{N}$) ou en temps de propagation $\frac{1}{2}N\sqrt{N}L\tau$.

Par contre nous verrons en section 4.6.1 que les algorithmes de type direct présentés par Scott obtiennent les meilleures performances en terme de temps de propagation $\frac{1}{4}N\sqrt{N}L\tau$ mais requièrent un nombre important d'étapes. Cependant, par définition, ces algorithmes ne nécessitent pas de recopies mémoire ni de réarrangement de messages. Le temps nécessaire à ces divers mouvement de données au sein de la mémoire n'est pas pris en compte dans les complexités des algorithmes indirects et peuvent accroître le temps de ces algorithmes.

Afin de réduire le nombre d'étapes, Thakur et Choudhary [39], ainsi que Fraigniaud et Peters [15] présentent des algorithmes d'échange total personnalisé qui ne sont pas sans conflit. Leur modèle suppose que les messages se partagent la bande passante des liens de communication. Il est difficile de pouvoir comparer leurs algorithmes étant donné que notre modèle suppose l'absence de conflit. Néanmoins leur approche est intéressante dans la mesure où sur les machines actuelles, plusieurs messages peuvent être multiplexés sur un lien.

4.5.5 Le tore de dimension 2 dans le modèle F_1

i) Algorithme de Hinrischs, Kosak, O'Hallaron, Striker et Take et de Horie et Hayashi.

Hinrischs, Kosak, O'Hallaron, Striker et Take [18] mettent en œuvre des algorithmes de multidistribution directe en modifiant les routeurs iWARP [6, 7] pour qu'ils se synchronisent sur les phases des algorithmes de multidistribution, évitant les phases de synchronisation globale nécessaires pour que les algorithmes soient effectivement sans conflit.

Les auteurs généralisent l'algorithme de Scott [32] développé sur un chemin P_N (Cf. section 4.5.3-i) au cas des anneaux C_N, et applique une méthode similaire, à celle des grilles, pour l'étendre au cas des tores. Une borne inférieure sur le nombre d'étapes dans le cas d'un tore de dimension n est :

$$m_\alpha(TM(p)^n) \geq \frac{p^{n+1}}{8}, \text{ pour les algorithmes de type direct.}$$

Le temps de leur algorithme est :

$m_{F_1}^{\mathcal{R}_\star,Direct}(TM(p)^2)$		
m_α	m_δ	m_τ/L
$\frac{N\sqrt{N}}{8}$	$\frac{N\sqrt{N}}{8}D$	m_α

Ce protocole est optimal en nombre d'étape dans le cas d'algorithmes directs.

Remarque. Notons que Horie et Hayashi [22] présentent des algorithmes directs sur des tores utilisant le même genre de stratégie.

ii) Algorithme de Tseng, Gupta et Panda

Dans [40], Tseng, Gupta et Panda proposent un algorithme d'échange total personnalisé indirect dans un tore $TM(2^k)^2$. Les processeurs du tore sont divisés en quatre groupes qui forment un pavage du tore suivant ses diagonales, i.e., le groupe i est composé des processeurs dont les coordonnées x, y vérifient : $(x - y) \bmod 4 = i$. Au cours de chaque étape, les processeurs ne communiquent qu'avec un autre processeur du même groupe, chaque groupe ayant une direction particulière afin d'éviter les conflits (voir figure 4.6).

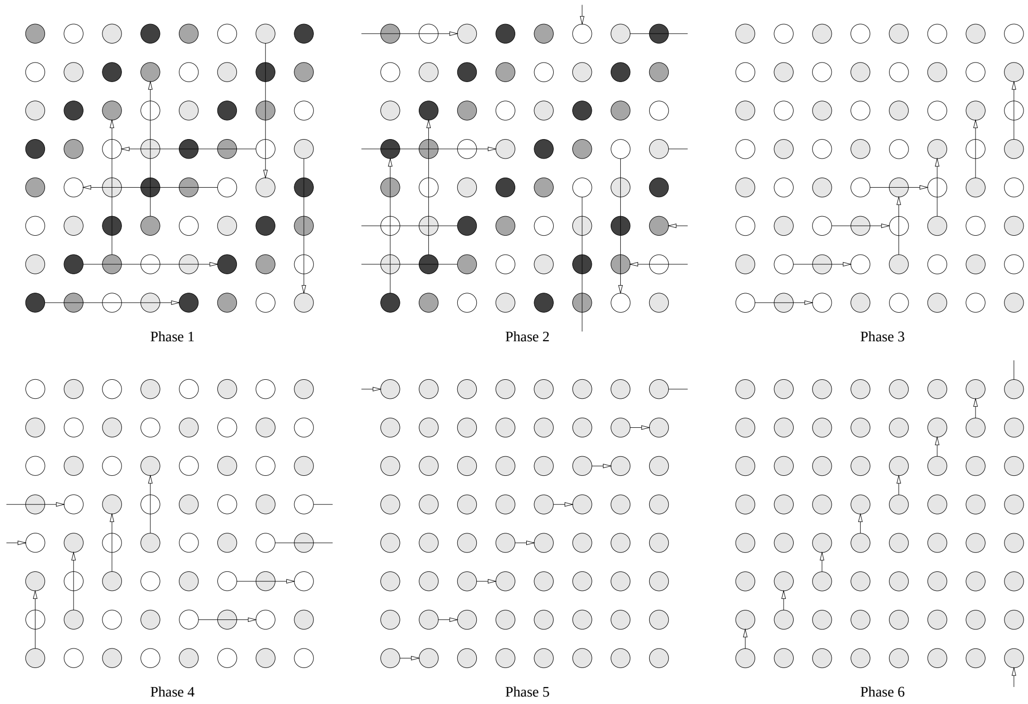

FIG. 4.6 – : *Exemple de propagation suivant la diagonale sur un tore 8×8. Seul quelques chemins représentatifs ont été tracés sur cette figure. Tous les sommets de même couleur communiquent avec un autre sommet à une position similaire.*

Le temps de cet algorithme est :

$$\begin{aligned} m_{F_1}(T(2^k, 2^k)) &\leq 4(\alpha + 2^{2\,n-1} L\tau) + 6\,\delta + \sum_{i=2}^{n-1} 2^{i-1} \left(\alpha + 2^i \delta + 2^{2\,n-1} L\tau\right) \\ &\leq \left(2^{k-1} + 2\right)\alpha + \left(\frac{10}{3} + \frac{4^k}{6}\right)\delta + \left(2^{2\,k} + \frac{2^{3\,k}}{4}\right) L\tau \end{aligned}$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(TM(2^k)^2)$		
m_α	m_δ	m_τ / L
$\frac{\sqrt{N}}{2} + 2$	$\frac{1}{3}(D^2 + 10)$	$N\left(1 + \frac{\sqrt{N}}{4}\right)$

4.5.6 Le tore de dimension 3 dans le modèle F_1

Tseng, Lin, Gupta et Panda ont proposé, dans [41], une version pour le tore de dimension 3. Le temps de leur algorithme est :

$$\begin{aligned} m_{F_1}(T(2^k, 2^k, 2^k)) &\leq 4 \sum_{i=3}^{n-1} 2^{i-3} \left(\alpha + 2^i \delta + 2^{3\,n-1} L\tau\right) + 9\,\alpha + 21\,\delta + 9\,2^{3\,n-1} L\tau \\ &\leq \left(2^{k-1} + 5\right)\alpha + \left(\frac{4^k}{6} + \frac{31}{3}\right)\delta + \left(\frac{5}{2} 2^{3k} + \frac{2^{4k}}{4}\right) L\tau \end{aligned}$$

Soit,

$m_{F_1}^{\mathcal{R}_\star}(TM(2^k)^3)$		
m_α	m_δ	m_τ/L
$\frac{\sqrt[3]{N}}{2}+5$	$\frac{1}{3}\left(2\left(\frac{D}{3}\right)^2+31\right)$	$N\left(\frac{5}{2}+\frac{\sqrt{N}}{4}\right)$

4.6 La multidistribution avec fonction de routage imposée

4.6.1 La grille de dimension 2 dans le modèle F_1

Dans le cas d'une grille $M(p)^2$ où le routage se fait de façon XY, Scott [32] donne une borne inférieure sur le nombre d'étapes nécessaires pour effectuer une multidistribution directe :

$$m_\alpha(M(p)^2) \geq \frac{p^3}{4}, \text{ pour les algorithmes de type direct.}$$

Il propose un algorithme, appelé DE, dans une grille $M(4p)^2$ d'ordre N, atteignant cette borne inférieure en nombres d'étapes. Les étapes sont obtenues en effectuant le produit cartésien des étapes effectuées dans un chemin P_{4p} en utilisant le protocole décrit en section 4.5.3-i (voir un exemple sur la figure 4.7).

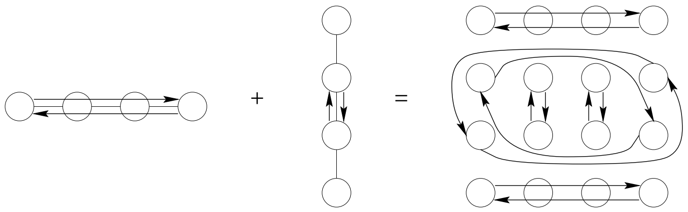

FIG. 4.7 – : *Etape au cours de la multidistribution dans une grille* 4×4 *obtenue à partir du produit cartésien entre deux étapes effectuées sur un chemin* P_4.

Le temps de cet algorithme est :

$m_{F_1}^{\mathcal{R}_{XY},Direct}(M(4p)^2)$		
m_α	m_δ	m_τ/L
$\frac{N\sqrt{N}}{4}$	$\frac{N\sqrt{N}}{4}D$	$\frac{N\sqrt{N}}{4}$

4.6.2 L'hypercube dans le modèle F_1

Rappelons la complexité de l'algorithme nommé SBT utilisé dans le mode commutation de paquet qui s'exécute en n étapes [21, 19, 20, 34] sur l'hypercube $H(n)$:

$m_{F_1}^{\mathcal{R}_{e-cube}}(H(n))$		
m_α	m_δ	m_τ
$\log_2 N$	D	$\frac{N\log_2 N}{2}$

Remarque. Notons que cet algorithme entraîne de nombreux mouvements et réordonnancements de données au sein même de la mémoire des processeurs, temps qui n'est pas pris en compte dans l'estimation de la complexité ci-dessus.

i) Algorithme de Seidel

Dans [33], Seidel propose un algorithme dans lequel tous les processeurs échangent leur information initiale avec tous les autres processeurs. Il montre qu'en utilisant une fonction de routage e-cube, cet algorithme (aussi appelé *DR direct-route*) est sans conflit :

$m_{F_1}^{\mathcal{R}_{e-cube}}(H(n))$		
m_α	m_δ	m_τ/L
$N-1$	$D2^{D-1}$	m_α

Remarque. Cet algorithme est repris dans divers articles [5, 19, 20, 32] et apparaît initialement dans [37] mais en version japonaise !!

ii) Algorithme de Ho et Raghunath et de Bokhari

Notons que Ho et Raghunath [19, 20] proposent une stratégie hybride construite à partir des deux approches précédentes (SBT et DR) afin de trouver un compromis entre le nombre d'étapes (n pour SBT et 2^n-1 pour DR) et le temps de transmission (il existe un rapport $\frac{n}{2}$ entre DR et SBT). Cette approche est aussi employée par Bokhari [3]. En effet, l'algorithme standard est plus efficace pour des tailles « petites » de messages. Donc, en combinant les deux et en effectuant des échanges partiels simultanément au sein de sous-cubes de dimension n_1, $n_1 < n$, puis sur les sous-cubes de dimension $n - n_1$, on peut obtenir un algorithme plus performant que les deux algorithmes précédents pris séparément. Pour chaque partition possible, $\mathcal{D} = \{n_1, \ldots, n_d\}$ de n tel que $\sum_{i=1}^d n_i = n$, le meilleur algorithme est employé à chaque phase. Le nombre de partitions d'un entier n est exponentiel et est de l'ordre de $\phi(n) \sim \frac{1}{4n\sqrt{3}} e^{\pi\sqrt{2/3}\sqrt{n}}$, mais pour des valeurs de n raisonnables, les valeurs de $\phi(n)$ sont faibles ($\phi(10) = 42$ et $\phi(20) = 627$). Le temps correspondant à cet algorithme étant donnée une partition $\mathcal{D} = \{n_1, \ldots, n_d\}$ de n est :

$m_{F_1}^{\mathcal{R}_{e-cube}}(H(n))$		
m_α	m_δ	m_τ/L
$\sum_{i=1}^d (2^{n_i-1}-1)$	$\sum_{i=1}^d (2^{n_i-1}-1)\left(\frac{n_i 2^{n_i-1}}{2^{n_i}-1}\right)$	$\sum_{i=1}^d (2^{n_i-1}-1)$

REMARQUE. Notons que la complexité en temps de transmission de l'algorithme - ii - de Seidel est meilleure que la borne inférieure sur le temps de transmission calculée dans le mode de commutation par paquets sous la contrainte 1-port qui est égale à $\frac{N\log(N)}{2}L\tau$. Cela est dû au fait que, dans le mode de commutation wormhole, plusieurs messages peuvent transiter par un sommet alors que ce dernier est lui même source et/ou destination d'une communication ce qui est impossible dans le mode commutation par paquets sous la contraint 1-port. Le mode commutation wormhole permet une meilleure utilisation de la bande passante globale offerte par le réseau.

4.6.3 L'hypercube dans le modèle F_*

Seidel adapte son algorithme présenté en section 4.6.2-i dans le cas 1-port au mode Δ-ports. A chaque phase j, $0 \leq j \leq n/2$, chaque processeur échange son information avec un processeur à distance j et un processeur à distance $n-j$, cela pour tous les processeurs qui sont à distance j. Cet algorithme n'utilise en fait que 2-ports de communication simultanément. Il montre qu'en utilisant une fonction de routage e-cube, les chemins sont arc-disjoints au cours de chaque étape.

$$\begin{aligned}
m_{F_*}^{\mathcal{R}_{e-cube}}(H(n)) &\leq \sum_{i=0}^{\lfloor\frac{n-1}{2}\rfloor}\binom{n}{i}(\alpha+(n-i)\delta+L\tau)\\
&\quad +(1-n \bmod 2)\frac{1}{2}\binom{n}{\frac{n}{2}}(\alpha+\frac{n}{2}\delta+L\tau)\\
&\leq 2^{n-1}\alpha+n2^{n-1}\delta+2^{n-1}L\tau\\
&\quad +(1-n \bmod 2)\frac{2^{n-1}}{\sqrt{\pi n}}(\alpha+\frac{n}{2}\delta+L\tau)
\end{aligned}$$

Donc, à partir d'un certain rang, on a :

$m_{F_*}^{\mathcal{R}_{e-cube}}(H(n))$		
m_α	m_δ	m_τ/L
$\frac{N}{2}$	$D2^{D-1}$	m_α

Cet algorithme atteint la borne inférieure en temps de transmission, mais conserve un temps d'initialisation important.

Notons que l'algorithme SBT [21, 34] obtient un meilleur temps :

$m_{F_*}^{\mathcal{R}_{e-cube}}(H(n))$		
$m_\alpha(H(n))$	$m_\delta(H(n))$	$m_\tau(H(n))$
$\log_2 N$	$D(H(n))$	$\frac{N}{2}L$

Cependant, comme il n'est pas de type direct, il requiert un grand nombre de copies mémoires au sein des processeurs afin de réorganiser les messages, temps qui n'est pas pris en compte dans le coût.

4.7 Conclusion des chapitres 2, 3 et 4

Nous avons tout au long de ces trois chapitres cherché à synthétiser les travaux principaux pourtant sur la diffusion, l'échange total et la multidistribution essentiellement lorsque l'on cherche en premier lieu à minimiser le nombre d'étapes de protocoles utilisant le modèle de « type commutation de circuits synchrone ». Néanmoins cette synthèse n'est bien évidemment pas exhaustive.

- En premier lieu, nous n'avons pas considérer ici, tous les modèles (portant sur les contraintes technologiques des routeurs) existant dans la littérature. Citons par exemple le modèle appelé « sommets disjoints » [24, 25, 23, 27] dans lequel l'ensemble des chemins réalisés (modélisant l'établissement des communications) au cours d'une étape donné doit être sommets disjoints, i.e. un sommet ne être traversé que un seul chemin à une étape donnée. Bien entendu, un tel modèle peut lui aussi se décliner comme de coutume en fonction de la nature des liens (télégraphique ou téléphonique).

- En second lieu, nous avons regroupé les travaux portant sur les réseaux les plus usuellement étudiés. Mais, d'autres graphes tout aussi intéressant même si moins classiques ont fait l'objet d'études (voir par exemple [1, 2]).

De plus comme nous l'avons déjà précisé dans l'introduction de cette partie, il existe d'autres schémas de communications globales.

Bibliographie

[1] T. Armitage. Broadcasting on torus-like chordal rings. Master's thesis, Simon Fraser University, School of Computing Science, August 1996.

[2] T. Armitage and J.G. Peters. Broadcasting on torus-like chordal rings. IWIN 95, Luminy, France, July 1995.

[3] S.H. Bokhari. Multiphase complete exchange on circuit switched hypercubes. In C-L. Wu, editor, *1991 International Conference on Parallel Processing (ICPP '91)*, volume I, pages 525–529, August 1991.

[4] S.H. Bokhari and H. Berryman. Complete exchange on a circuit switched mesh. In *Scalable High Performance Computing Conference (SHPCC'92)*, pages 300–306, Williamsburg, Virginia, April 1992. IEEE.

[5] R. Boppana and C. Raghavendra. All-to-all personalized communication on circuit-switched hypercubes. Technical report, Dept. EE-Systems, University of Southern California, Los-Angeles, October 1990.

[6] S.B. Borkar, R. Cohn, G. Cox, S. Gleason, T. Gross, H.T. Kung, M. Lam, B. Moore, C. Peterson, J. Pieper, L. Rankin, P.S. Tseng, J. Sutton, J. Urbanski, and J. Webb. iWARP: An integrated solution to high-speed parallel computing. In *Supercomputing'88*, pages 330–339, Kissimmee, Florida, November 1988. IEEE.

[7] S.B. Borkar, R. Cohn, G. Cox, T. Gross, H.T. Kung, M. Lam, M. Levine, B. Moore, W. Moore, C. Peterson, J. Susman, J. Sutton, J. Urbanski, and J. Webb. Supporting systolic and memory communication in iWARP. In *Proceeding of the 17th Annual International Symposium on Computer Architecture*, pages 70–81, Seatle, Washington, May 1990.

[8] J. Bruck, C-T. Ho, S. Kipnis, and D. Weathersby. Efficient algorithms for all-to-all communications in multi-port message-passing systems. In *Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA '94)*, pages 298–309, Cape May, New Jersey, June 1994. ACM.

[9] C. Calvin, S. Perennes, and D. Trystram. Gossiping in torus with circuit-switched routing. Technical Report APACHE 9, Institut d'Informatique et de Mathématiques Appliquées de Grenoble, April 1994. (Submitted to the "Journal of Computer and Software Engineering - Special issue on Parallel Algorithms and Architectures"), http://www-apache.imag.fr/apache.

[10] C. Calvin, S. Perennes, and D. Trystram. Gossiping in torus with wormhole-like routing. In *7-th IEEE Symposium on Parallel and Distributed Processing (SPDP '95)*, San Antonio, USA, October 1995.

[11] M. Cosnard and D. Trystram. *Algorithmes et architectures parallèles.* Informatique Intelligence Artificielle. InterEditions, 1993.

[12] O. Delmas and S. Perennes. Circuit-switched gossiping in 3-dimensional torus networks. In L. Bougé, P. Fraigniaud, A. Mignotte, and Y. Robert, editors, *Proceedings of the Euro-Par'96 Parallel Processing / Second International EURO-PAR Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 370–373, Lyon, France, August 1996. Springer Verlag.

[13] O. Delmas and S. Perennes. Circuit-Switched Gossiping in 3-Dimensional Torus Networks. Rapport de Recherche 2930, Institut National de Recherche en Informatique et en Automatique, Unité de recherche INRIA Sophia Antipolis - France, July 1996. (Version étendue de [12]).

[14] P. Fraigniaud and E. Lazard. Methods and problems of communication in usual networks. *Discrete Applied Mathematics*, 53:79–133, 1994. Special volume proceedings international workshop on broadcasting and gossiping 1990.

[15] P. Fraigniaud and J. Peters. Structured communication in torus networks. In IEEE, editor, *28th Annual Hawaii International conference on system sciences*, pages 584–593, Hawaii, 1995.

[16] S. Fujita and M. Yamashita. Optimal group gossiping in hypercubes under a circuit-switching model. *SIAM Journal on Computing*, 25(5):1045–1060, October 1996.

[17] M-C. Heydemann, J-C. Meyer, and D. Sotteau. On forwarding indices of networks. *Discrete Applied Mathematics*, 23:103–123, 1989.

[18] S. Hinrischs, C. Kosak, D. R. O'Hallaron, Striker T. M., and R. Take. An architecture for optimal all-to-all personalized communication. In *Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA '94)*, Cape May, New Jersey, June 1994. ACM. (see also Technical Report CMU-CS-94-10, Carnegie Mellon University, September 1994).

[19] C-T. Ho and M. T. Raghunath. Efficient communication primitives on circuit-switched hypercubes. In *6th Distributed Memory Computing Conference*, pages 390–397. IEEE, April 1991.

[20] C-T. Ho and M.T. Raghunath. Efficient communication primitives on hypercubes. *Concurrency: Practice and experience*, 4(6):427–457, September 1992.

[21] C.T. Ho and S.L. Johnsson. Optimum broadcasting and personalized communication in hypercubes. *IEEE Transactions on Computers*, 38(9):1249–1268, 1989.

[22] T. Horie and K. Hayashi. All-to-all personalized communication on wrap-around mesh. In *2nd Fujitsu-ANU CAP Workshop*, November 1991. ftp://fcapwide.fujitsu.co.jp/ap1000/english/pcw/capws91/.

[23] J. Hromkovič, R. Klasing, and E.A. Stöhr. Gossiping in vertex-disjoint paths mode in interconnection networks. In Springer Verlag, editor, *Proc. 19th Int. Workshop on Graph-Theoretic Concepts in Computer Science (WG'93)*. LNCS, 1993. to appear.

[24] J. Hromkovič, R. Klasing, E.A. Stöhr, and H. Wagener. Gossiping in vertex-disjoint paths mode in $d-$dimensional grids and planar graphs. In Springer Verlag, editor, *Proc. First Annual European Symposium on Algorithms (ESA'93)*, pages 200–211. LNCS 726, 1993.

[25] J. Hromkovič, R. Klasing, W. Unger, and H. Wagener. Optimal algorithms for broadcast and gossip in the edge-disjoint paths modes. In Springer Verlag, editor, *Proc. 4th Scandinavian Workshop on Algorithm Theory (SWAT'94)*. LNCS, 1994. to appear.

[26] B.H.H. Juurlink, P.S. Rao, and J.F. Sibeyn. Whorm-hole gossiping on meshes. In L. Bougé, P. Fraigniaud, A. Mignotte, and Y. Robert, editors, *Proceedings of the Euro-Par'96 Parallel Processing / Second International EURO-PAR Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 361–369, Lyon, France, August 1996. Springer Verlag.

[27] R. Klasing. The relationship between gossiping in vertex-disjoint paths mode and bisection width. In *Proc. 19th Int. Symp. on Mathematical Foundations of Computer Science*, 1994.

[28] J-C. König, P.S. Rao, and D. Trystram. Analysis of gossiping algorithms in torus with restricted bufferization capabilities. Technical Report APACHE 13, Institut d'Informatique et de Mathématiques Appliquées de Grenoble, November 1994. http://www-apache.imag.fr/apache.

[29] F.J. McWilliams and N.J.A. Sloane. *The theory of Error-Correcting Codes*. North-Holland, 1977.

[30] J.G. Peters and M. Syska. Circuit-Switched Broadcasting in Torus Networks. *IEEE Transactions on Parallel and Distributed Systems*, 7(3), March 1996.

[31] Jean de Rumeur. *Communication dans les réseaux de processeurs*. Collection Etudes et Recherches en Informatique. Masson, Paris, 1994.

[32] D. S. Scott. Efficient all-to-all communication patterns in hypercube and mesh topologies. In *6th Distributed Memory Computing Conference*, pages 398–403, 1991.

[33] S.R. Seidel. Circuit-switched vs. store-and-forward solutions to symmetric communication problems. In *The Fourth Conference on Hypercubes, Concurrent Computers, and Applications*, pages 253–255, March 1989.

[34] Q. F. Stout and B. Wagar. Intensive hypercube communication, prearranged communication in link-bound machines. *Journal of Parallel and Distributed Computing*, 10:167–181, 1990.

[35] N.S. Sundar, D.N. Jayasimha, D.K. Panda, and P. Sadayappan. Complete exchange in 2D meshes. In *Scalable Hight Performance Computing Conference (SHPCC '94)*, pages 406–413, Knoxville, Tennessee, May 1994. IEEE.

[36] N.S. Sundar, D.N. Jayasimha, D.K. Panda, and P. Sadayappan. Hybrid algorithms for complete exchange in 2D-meshes. In *International Conference on Supercomputing (ICS '96)*, pages 181–188, Philadelphia, Pennsylvania, USA, May 1996. ACM.

[37] R. Take. An optimal routing method of all-to-all communication on hypercube networks. In *35th Information Processing Society of Japan*, 1987. (In Japanese dixit Ho).

[38] S. Takkella and S. Seidel. Complete exchange and broadcast algorithms for meshes. In *Scalable Hight Performance Computing Conference (SHPCC '94)*, pages 422–428, Knoxville, Tennessee, May 1994. IEEE.

[39] R. Thakur and A. Choudhary. All-to-all communication on meshes with wormhole routing. In *Eighth International Parallel Processing Symposium (IPPS '94)*, pages 561–565, Cancun, Mexico, April 1994.

[40] Y-C. Tseng, S.K. Gupta, and D.K. Panda. An efficient scheme for complete exchange in 2D tori. In *International Parallel Processing Symposium (IPPS '95)*, pages 532–536. IEEE, April 1995.

[41] Y-C. Tseng, T-H. Lin, S.K. Gupta, and D.K. Panda. Bandwidth-optimal complete exchange on wormhole-routed 2D/3Dtorus networks: A diagonal-propagation approach. Technical Report OSU-CISRC-3/96-TR14, Department of Computer and Information Science, Ohio State University, Columbus, 1996. http://www.cis.ohio-state.edu/Research.html.

Chapitre 5

Propriétés et construction de graphes

✔ *Nous abordons dans ce chapitre, deux des propriétés de réseau couramment étudiés dans la littérature. En premier lieu, nous résumons nos travaux sur la décomposition hamiltonienne du graphe Butterfly généralisé, puis nous abordons le problème des larges* $(\Delta, D)-$*graphes.*

Les propriétés nécessaires au bon fonctionnement d'un réseau dépendent bien entendu de l'utilisation de ce dernier, ce qui implique de trouver des algorithmes de communications efficaces. Cette étude a fait l'objet des chapitres précédents. Cependant, certaines propriétés dans la plupart des réseaux sont exigées, par exemple on veut que le nombre de liaisons à utiliser ne soit pas trop grand, ce qui correspond à respecter les contraintes technologiques sur le nombre d'entrées/sorties dans la fabrication d'un processeur. Souvent, les constructeurs ne peuvent gérer qu'un petit nombre de ces entrées/sorties par composants. D'autres propriétés peuvent être intéressantes. Même si elles ne sont pas encore toujours utilisées par les constructeurs, elles peuvent devenir, dans un avenir proche, essentielles à la construction d'un réseau efficace, par exemple comment connecter un maximum de processeurs pour un degré petit fixé et un diamètre faible également fixé. Ainsi, en plus de l'étude d'algorithmes efficaces il est important de trouver ou d'étudier de « bon » réseaux. Mais définir le terme « bon » n'est pas toujours chose facile. Nonobstant, quelques paramètres et propriétés, considérés comme importants et en tout cas révélateurs des qualités ou défauts d'un réseau sont particulièrement étudiés. Le nombre de ces propriétés est important et pour notre part nous n'en aborderons que deux dans ce chapitre.

La première partie résume la recherche menée sur le sujet de la décomposition hamiltonienne d'un graphe. Après en avoir montré son intérêt, nous résumons notre contribution à la résolution de ce problème dans le cas du graphe *Butterfly* généralisé.

La seconde partie concerne le problème désormais classique de recherche de larges graphes (Δ, D). Après avoir rappelé quelques résultats fondamentaux, nous évoquons nos travaux sur ce sujet.

5.1 Décomposition hamiltonienne du réseau Butterfly généralisé

5.1.1 Motivation

Tout d'abord, fixons la terminologie employée :

DEFINITION. Lorsqu'un graphe G admet p cycles (resp. circuits) hamiltoniens deux-à-deux arête-disjoints (resp. arc-disjoints), nous dirons que G admet p cycles (resp. circuits) hamiltoniens *compatibles*.

Cette terminologie vient de celle en vigueur pour les cycles (resp. circuits) eulériens. Rappelons que deux cycles eulériens sont dits *compatibles* si toute paire d'arêtes consécutives dans un cycle ne se retrouve pas dans l'autre cycle. De plus pour les graphes représentatifs des arcs, nous avons :

Théorème 5.1.1 — *Si G est un graphe orienté fortement connexe, alors $L(G)$ est hamiltonien si et seulement si G est eulérien.*

Ce résultat vient du fait qu'un circuit hamiltonien de $L(G)$ est un circuit eulérien de G. La démonstration, que l'on peut trouver dans les livres [8] et [30], est algorithmique, un circuit eulérien dans G se calculant en temps $\Theta(N)$. Notons que l'on peut aisément montrer la généralisation suivante du théorème 5.1.1 :

Théorème 5.1.2 — *$L(G)$ admet p cycles (resp. circuits) hamiltoniens compatibles si et seulement si G admet p cycles (resp. circuits) eulériens compatibles.*

Le cas particulier où les cycles (resp. circuits) recouvrent toutes les arêtes (resp. arcs) a été particulièrement étudié.

DEFINITION. G est décomposable (ou se décompose) en cycles (resp. circuits) hamiltoniens si on peut partitionner les arêtes (resp. arcs) de G en cycles (resp. circuits) hamiltoniens.

Cette définition implique que si G est non orienté et décomposable en p cycles hamiltoniens, alors G est régulier de degré pair $2p$ et admet p cycles hamiltoniens compatibles. De même, dans le cas orienté, si G est décomposable en p circuits hamiltoniens, alors cela implique que G est fortement régulier (i.e. pour tout sommet x, $d^+(x) = d^-(x) = p$) et admet p circuits compatibles.

REMARQUE. Notons que dans la synthèse [1], une définition plus générale de «décomposable» est donnée dans le cas non orienté. Pour traiter le cas des graphes réguliers de degré impair, on dit alors qu'un graphe $2p+1$ régulier est décomposable en cycles hamiltoniens s'il admet p cycles hamiltoniens compatibles. Ceci revient à dire qu'il est possible de partitionner ses arêtes en p cycles hamiltoniens plus un couplage parfait.

L'existence de plusieurs cycles ou circuits hamiltoniens est importante en algorithmique distribuée. En effet, si avec un cycle hamiltonien on peut exécuter tous les algorithmes écrits pour l'anneau sans perte de performance, avec p cycles, on peut exécuter p instances de ces algorithmes (pas forcément les mêmes) de façon concurrente. Par exemple, si un graphe G est hamiltonien, on obtient une borne supérieure immédiate sur le temps de diffusion ou d'échange total en appliquant l'algorithme de communication de l'anneau. Or, si G admet p cycles hamiltoniens, ces temps sont améliorés puisque l'on dispose d'une bande passante p fois plus grande : au lieu de diffuser un message de longueur L sur un cycle hamiltonien, on en diffuse p portions de longueur $\frac{L}{p}$ (on peut aussi imaginer une adaptation de l'algorithme d'échange total). De plus, cela renforce sa capacité à résister aux pannes, puisqu'il faudra au moins p liens en panne pour casser tous les cycles et donc empêcher l'exécution de l'algorithme.

5.1.2 Etat de l'art

Nous rappellons ici quelques résultats et conjectures connus en matière de décomposition hamiltonienne pour des constructions présentées dans la section 1.4.1. Ces constructions donnent des graphes en général très réguliers, donc susceptibles d'être décomposables en cycles ou en circuits hamiltoniens. Pour plus de détails, on pourra consulter la synthèse [1], dans laquelle le lecteur pourra notamment trouver la preuve de la proposition suivante :

Proposition 5.1.3 — *Le graphe complet K_N se décompose en $\frac{N-1}{2}$ cycles hamiltoniens si N est impair, et admet $\frac{N-2}{2}$ cycles hamiltoniens compatibles si N est pair.*

Le cas du graphe complet orienté symétrique a été résolu par Tillson et sert de base aux récurrences des principaux résultats des sections qui suivent.

Théorème 5.1.4 (Tillson, 1980 [32]) — *Si $d \notin \{4, 6\}$, $\mathcal{K}_d^*$ est décomposable en $d-1$ circuits hamiltoniens. Si $d = 4$ ou 6, $\mathcal{K}_d^*$ contient $d-2$ circuits hamiltoniens compatibles et un 1-difacteur formé d'arcs doubles.*

Remarque. Rappelons qu'un $1-$difacteur est un graphe orienté régulier avec $d^+(x) = d^-(x) = 1$ pour tout sommet x.

5.1.2.1 Graphes de Cayley et sommes cartésiennes

Il est connu que tout graphe de Cayley défini sur un groupe fini abélien (c'est-à-dire commutatif) est hamiltonien. La conjecture suivante est plus forte :

Conjecture 5.1.5 (Alspach, 1990 [1]) — *Tout graphe de Cayley défini sur un groupe abélien se décompose en cycles hamiltoniens.*

Cette conjecture a été vérifiée par Bermond, Favaron et Mahéo [15] pour les graphes de degré 4. Depuis, des progrès ont été faits par Liu dans le cas des graphes de Cayley définis sur un groupe abélien d'ordre impair [24].

La conjecture a aussi été montrée pour des graphes particuliers tels que la grille torique de dimension k notée $TM(l)^k$ ou l'hypercube $H(2k)$ de dimension $2k$ (qui sont décomposables en k cycles hamiltoniens). Le résultat sur la grille torique résulte du fait que la somme cartésienne de deux cycles est décomposable en deux cycles hamiltoniens. Pour l'hypercube, cela résulte aussi du résultat suivant dû à Aubert et Schneider [3], à savoir que la somme cartésienne d'un cycle et du graphe G formé de deux cycles hamiltoniens est décomposable en trois cycles hamiltoniens.

Plus généralement, la conjecture suivante est toujours ouverte, même si de nombreux cas ont déjà été prouvés par Stong [31]. Récemment, Mellendorf l'a démontré dans le cas de "multicycles" [25].

Conjecture 5.1.6 (Bermond, 1978 [9]) — *Si G et G' sont décomposables en cycles hamiltoniens, alors la somme cartésienne $G\Box G'$ l'est aussi.*

5.1.2.2 Graphes représentatifs des arêtes ou des arcs

Dans [10], Bermond a émis la conjecture suivante concernant le graphe représentatif des arêtes $L(G)$ d'un graphe non orienté G.

Conjecture 5.1.7 (Bermond, 1988 [10]) — *Si G se décompose en cycles hamiltoniens, alors $L(G)$ aussi.*

Cette conjecture a été quasiment résolue par Muthusamy et Paulraja dans [26] et Zhan dans [33]. Ils ont montré que si G est décomposable en un nombre pair (resp. impair) de cycles hamiltoniens, alors $L(G)$ est décomposable en cycles hamiltoniens (resp. en cycles hamiltoniens et un 2-facteur). Ils ont aussi montré que si G est seulement hamiltonien et de degré $2r$, alors $L(G)$ admet $2r - 2$

cycles hamiltoniens compatibles. De plus, Heinrich et Verrall ont récemment montré dans [23] que $L(K_{2k+1})$ était décomposable en cycles hamiltoniens, en construisant $2k$ cycles eulériens compatibles dans K_{2k+1}.

En ce qui concerne le cas orienté, Bond et Muthusamy ont exhibé des familles de graphes décomposables en cycles hamiltoniens dont les graphes représentatifs des arcs ne sont pas décomposables [16]. Néanmoins, Fleischner et Jackson ont prouvé dans [22] que tout graphe représentatif des arcs d'un graphe d-régulier admet $\lfloor \frac{d}{2} \rfloor$ circuits hamiltoniens compatibles. De plus, Balakrishman, Bermond et Paulraja ont montré dans [4] que les graphes sous-jacents de graphes représentatifs des arcs d-réguliers admettent presque toujours $d-1$ cycles hamiltoniens, résultat quasi-optimal.

Concernant les graphes de de Bruijn orientés qui sont des itérés de graphes représentatifs des arcs de $\mathcal{K}_d^+$, Bond a émis la conjecture suivante :

Conjecture 5.1.8 (Bond) — *$\mathcal{B}(d,n)$ admet $d-1$ circuits hamiltoniens compatibles.*

Notons que cette conjecture est vérifiée par le théorème 5.1.4 pour $n=1$ et $d \notin \{4,6\}$. Dans [6], Barth, Bond, et Raspaud prouvent cette conjecture pour $n=2$ et d premier. Elle est alors énoncée dans les termes suivants :

Conjecture 5.1.9 — *$\mathcal{K}_d^+$ admet un ensemble de $d-1$ circuits eulériens compatibles.*

Rowley et Bose ont ensuite montré dans [29] que lorsque d est une puissance de nombre premier, le graphe de de Bruijn orienté admet une décomposition en d 1-difacteurs isomorphes, tous constitués d'un circuit passant par tous les sommets sauf un et d'une boucle. Leur démonstration repose sur des propriétés des récurences linéaires sur les corps finis. En modifiant et en croisant les structures obtenues, les auteurs prouvent la conjecture de Bond lorsque $d=2^i$ et démontrent que si d est une puissance de premier ($d=p^i$), $\mathcal{B}(d,n)$ admet $\lfloor \frac{d-1}{2} \rfloor$ circuits hamiltoniens compatibles.

Pour le cas général, la meilleure borne obtenue se déduit du résultat de Fleischner et Jackson.

5.1.3 Décomposition hamiltonienne du graphe Butterfly

Le graphe *Butterfly* est à la fois un graphe de Cayley et un graphe représentatif des arcs. Barth et Raspaud ont démontré dans [7] que le graphe *butterfly* non orienté binaire $\mathcal{WBF}(2,n)$ est décomposable en 2 cycles hamiltoniens. Ils conjecturaient plus généralement :

Conjecture 5.1.10 — *$\mathcal{WBF}(d,n)$ se décompose en d cycles hamiltoniens.*

En revanche, à propos du cas orienté, Barth conjecturait dans [5] :

Conjecture 5.1.11 — *$\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ admet $d-1$ circuits hamiltoniens deux à deux arc-disjoints.*

La preuve de ces conjectures fait l'objet des articles [11, 12] donnés en annexe C pour le cas orienté et en annexe D pour le cas non orienté. Les articles étant en anglais, les sections suivantes ont pour but de résumer en français les résultats et les idées principales.

5.1.3.1 Cas orienté

Une des idées clefs est de ramener l'étude des cycles hamiltoniens dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ à l'existence de permutations sur $\mathbb{Z}_d^n$. Nous dirons qu'une permutation de $\mathbb{Z}_d^n$ est cyclique si pour un élément quelconque $x \in \mathbb{Z}_d^n$, les éléments $\pi^i(x)$ avec $0 \leq i < d^n$ sont tous distincts. En fait à un cycle hamiltonien de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ on peut associer dans le *Butterfly* multiétages $\vec{\mathcal{B}}\mathcal{F}(d,n)$ un ensemble de chemins deux à deux arc-disjoints reliant les sommets du niveau 0 aux sommets du niveau n qui correspondent dans la terminologie des réseaux multiétages, à une permutation dite réalisable. Un cycle hamiltonien de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ correspond à une permutation cyclique et réalisable dans $\vec{\mathcal{B}}\mathcal{F}(d,n)$.

De plus, à un ensemble de p circuits hamiltoniens compatibles de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$, on associe p permutations cycliques réalisables qu'on dira compatibles dans la mesure où les chemins qui leur correspondent dans le réseau multiétages $\vec{\mathcal{B}}\mathcal{F}(d,n)$ sont arc-disjoints.

Cette caractérisation énoncée, nous montrons qu'il existe p permutations cycliques compatibles et réalisables dans $\vec{\mathcal{B}}\mathcal{F}(d,n+1)$, dès qu'il en existe p dans $\vec{\mathcal{B}}\mathcal{F}(d,n)$. Pour cela, nous utilisons la construction récursive de $\vec{\mathcal{B}}\mathcal{F}(d,n+1)$ à partir de d copies isomorphes à $\vec{\mathcal{B}}\mathcal{F}(d,n)$. Ce résultat peut s'énoncer de la manière suivante :

Proposition 5.1.12 — *Le nombre de circuits hamiltoniens compatibles du graphe Butterfly $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ ne décroît pas lorsque n croît.*

En utilisant ce résultat et le fait que $\mathcal{K}_d^*$ se décompose en circuits hamiltoniens pour $d \notin \{4,6\}$ et que $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(4,2)$ et $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(6,2)$ contiennent respectivement 4 et 6 circuits hamiltoniens compatibles (voir le tableau suivant), nous prouvons la conjecture de Barth :

Proposition 5.1.13 — *$\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ contient $d-1$ circuits hamiltoniens compatibles, sauf $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(4,1)$, isomorphe à $\mathcal{K}_4^*$, et $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(6,1)$, isomorphe à $\mathcal{K}_6^*$.*

Nous avons alors mené une recherche exhaustive sur ordinateur ; celle-ci est présentée en détails dans [19]. Elle montre que le résultat précédent n'est pas optimal. En effet, le tableau suivant donne le nombre de circuits hamiltoniens compatibles dans $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$, en fonction des premières valeurs des paramètres d et n :

n \ d	**2**	**3**	**4**	**5**	**6**	**7**
1	1	2	2	4	4	6
2	1	2	4	5	6	7
3	1	3	4	5	6	7
4	2	3	4	5	6	7

Ces résultats nous ont conduit à modifier la conjecture de Barth :

Conjecture 5.1.14 — *$\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ est décomposable en circuits hamiltoniens, sauf pour $n = 1$, et pour $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,2)$, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(2,3)$ et $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(3,2)$.*

Quand $n = 1$, on est en présence du graphe complet orienté symétrique, cas résolu par Tillson (proposition 5.1.4). La recherche exhaustive, utilisée comme point de départ de l'induction de la proposition 5.1.13, répond à la conjecture pour $d \leq 7$. Il reste donc à prouver que $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,2) = L(\mathcal{K}_{d,d})$ est décomposable en circuits hamiltoniens pour $d > 3$.

Pour parfaire la réduction du problème, nous cherchons alors à nous ramener au cas d premier. Pour ce faire, nous utilisons une technique de conjonction dont nous rappelons la définition ci-dessous.

Definition. Etant donnés deux graphes G et G', la conjonction de G et G', notée $G \cdot G'$ est définie par :

- L'ensemble des sommets de $G \cdot G'$ est le produit cartésien $V(G) \times V(G')$, un sommet de $G \cdot G'$ étant noté (x, x') avec $x \in V(G)$ et $x' \in V(G')$.
- Dans $G \cdot G'$, il existe une arête entre les sommets (x, x') et (y, y') si et seulement si $[x, y]$ est une arête de G et $[x', y']$ est une arête de G'.

On obtient une définition similaire pour les graphes orientés en remplaçant les arêtes par des arcs.

Remarque. La conjonction commute avec l'opération de passage au graphe représentatif des arcs : $L(G \cdot G') = L(G) \cdot L(G')$

Tout d'abord, nous exprimons une propriété couramment citée du graphe *Butterfly* qui établit que le graphe de de Bruijn est un graphe quotient du graphe

Butterfly. Si l'on note $L(G)$ le graphe représentatif des arcs de G et $L^p(G)$ le graphe obtenu en itérant p fois l'opération $L^p(G) = L(L^{p-1}(G))$, alors ce résultat de graphe quotient, souvent expliqué de manière complexe (par exemple dans [2]), s'exprime très clairement de la façon suivante :

Proposition 5.1.15 — *On a les égalités :*

$$\mathcal{B}(d,n) = L^{n-1}(\mathcal{K}_d^+) \tag{5.1}$$

$$\mathcal{B}(d_1d_2,n) = \mathcal{B}(d_1,n) \cdot \mathcal{B}(d_2,n) \tag{5.2}$$

$$\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n) = \mathcal{B}(d,n) \cdot \vec{C}_n = L^{n-1}(\mathcal{K}_d^+ \cdot \vec{C}_n) \tag{5.3}$$

$$\vec{\mathcal{B}}\mathcal{F}(d,n) = \mathcal{B}(d,n) \cdot \vec{P}_n \tag{5.4}$$

$$\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1d_2,n) = \mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1,n) \cdot \mathcal{B}(d_2,n) \tag{5.5}$$

Nous prouvons alors que :

Proposition 5.1.16 — *$\mathcal{B}(d,2) \cdot \vec{C}_n$ est décomposable en circuits hamiltoniens dès que $n \geq 3$.*

Mais il semble assez difficile de conclure lorsque $n = 2$ (cas où l'on a $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,2) = \mathcal{B}(d,2) \cdot \vec{C}_2 = L(\mathcal{K}_{d,d}^*)$).

Les propriétés usuelles de la conjonction vis-à-vis de la décomposition en graphes partiels induisent alors, plus généralement :

Théorème 5.1.17 — *Si G, possédant au moins 3 sommets, est décomposable en circuits hamiltoniens, alors $\mathcal{B}(d,2) \cdot G$ l'est aussi.*

Ce résultat implique clairement que si $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1,2)$ (avec $d_1 > 1$) est décomposable en circuits hamiltoniens, alors $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1d_2,2)$ l'est aussi :

- si $d_2 = 1$, c'est une tautologie ;
- si $d_2 > 1$, alors $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1d_2,2) = \mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1,2) \cdot \mathcal{B}(d_2,2)$; comme $d_1 > 1$, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d_1,2)$ contient plus de 3 sommets et le théorème précédent permet alors de conclure.

Utilisant alors le fait que $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(4,2)$, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(6,2)$ et $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(9,2)$ sont décomposables, la conjecture 5.1.14 se ramène à prouver que pour tout p premier strictement supérieur à 3, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(p,2) = L(\mathcal{K}_{p,p}^*)$ se décompose en circuits hamiltoniens.

La recherche exhaustive devenant rapidement impossible (pour $p > 9$), nous nous sommes alors limités à des solutions très régulières, telles que les circuits soient obtenus par isomorphisme. Après quelques essais infructueux, il est apparu une famille de solutions très restreinte (pour un p fixé, le nombre d'éléments de la famille est p^2) qui permet de résoudre le problème pour tous les premiers inférieurs à 12000. Nous conjecturons qu'il existe toujours une telle solution dès que p est supérieur ou égal à 7 :

Conjecture 5.1.18 — *Pour tout p premier, supérieur à 5, il existe $\alpha \notin \{0,1\}$ et $\beta \neq 0$, tels que la permutation π de $(\mathbb{Z}_p)^2$ définie par*

$$\begin{array}{rcll} \pi(ab) & = & (a+\alpha b+1)(\alpha b) & \quad (a \neq 0) \\ \pi(0b) & = & (\beta+\alpha b+1)(\beta+\alpha b) & \end{array}$$

soit cyclique.

Répondre à cette conjecture de théorie des nombres cloturerait le problème.

Pour conclure, nous pouvons résumer les résultats obtenus :

- Si d admet un diviseur strict compris entre 4 et 12000, alors $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ est décomposable en circuits hamiltoniens dès que $n \geq 2$.
- Si d admet un diviseur strict inférieur à 12000, alors $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ est décomposable en circuits hamiltoniens pour $n \geq 4$.
- Si la conjecture 5.1.18 est résolue, alors le problème de la décomposition du graphe *Butterfly* en circuits hamiltoniens est clos :

	2	**3**	**4**	**5**	**6**	$\mathbf{d \geq 7}$	
1	1	2	2	4	4	$d-1$	$\}\ \mathcal{K}_d^*$
2	1	2	4	5	6	d	$\}\ L(\mathcal{K}_{d,d}^*)$
3	1	3	4	5	6	d	
4	2	3	4	5	6	d	
$\mathbf{n \geq 5}$	2	3	4	5	6	d	

Remarque. La méthode utilisée implique certains résultats pour le graphe de de Bruijn, en particulier :

Proposition 5.1.19 — *Si p est le plus grand facteur premier de d, alors $\mathcal{B}(d,2)$ admet $\frac{p-1}{p}d$ circuits hamiltoniens compatibles.*

Proposition 5.1.20 — *$\mathcal{B}(2^i q, 2)$ admet $(2^i-1)q$ circuits hamiltoniens compatibles.*

Ces résultats sont des conséquences de la proposition 5.1.16 (et des contructions précédentes). Nous pensons pouvoir les généraliser et savoir prouver que la conjonction $\mathcal{B}(d,n) \cdot \vec{C}_{n'}$ est décomposable en circuits hamiltoniens, dès que n' est strictement supérieur à n (le cas du graphe *Butterfly* correspond à $n' = n$).

Une question analogue est de déterminer à partir de quelle valeur de $f_d(n)$ la conjonction $\mathcal{B}(d,n) \cdot \vec{C}_{f_d(n)}$ est décomposable en circuits hamiltoniens.

5.1.3.2 Cas non orienté

Pour trouver une proposition analogue à 5.1.12 et démontrer la conjecture 5.1.10 de Barth et Raspaud, nous construisons une décomposition de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ en circuits ou en cycles orientés qui ont une propriété particulière que nous appelons *"l−crossing"* (voir annexe D).

De plus, notre preuve utilise une décomposition de $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ en deux graphes partiels : l'un contenant les arcs de type ω_α, avec $\alpha \in \{0,1\}$, c'est-à-dire ceux de pente 0 ou 1 ; l'autre comprenant les arcs restants, c'est-à-dire ceux de type ω_α, avec $\alpha \in [2, d-1]$. Nous prouvons alors que le premier graphe se décompose en 2 cycles hamiltoniens, alors que le second admet presque toujours $d-2$ circuits hamiltoniens compatibles. Certains cas apparaissant comme des exceptions, nous obtenons :

- Pour $n \geq 2$ et $d \notin \{3,4,6\}$, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ se décompose en $d-2$ circuits hamiltoniens et 2 cycles hamiltoniens.
- Pour $d \in \{4,6\}$, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(d,n)$ est décomposable en circuits hamiltoniens.
- Pour $n \geq 3$, $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(3,n)$ est décomposable en 3 circuits hamiltoniens.
- $\mathcal{W}\vec{\mathcal{B}}\mathcal{F}(3,2)$ est décomposable en 3 cycles hamiltoniens.

Ainsi, appliqué au cas non orienté ceci prouve :

Théorème 5.1.21 — *Quelque soit d et $n \geq 2$, $\mathcal{WBF}(d,n)$ est décomposable en cycles hamiltoniens.*

5.2 Problème $(\boldsymbol{\Delta}, \boldsymbol{D})$

Le but d'un réseau, quelque soit sa nature (informatique ou non), est de connecter des composants de telle façon que chacun d'entre eux puisse joindre tous les autres (i.e. on cherche à construire des réseaux connexes). Dans la majorité des cas, et a fortiori dans le monde informatique, on désire que le nombre de composants connectés soit le plus grand possible (par exemple, plus une machine parallèle comporte de processeurs, plus sa puissance de calcul potentielle est grande). Cependant, on est bien souvent limité par des contraintes physiques de réalisation, exprimées sous la forme de paramètres. Dans la problématique qui nous intéresse, on en retient que deux : le degré maximum Δ, qui limite le nombre de voisins possibles pour un des composants du réseau (représentés par les sommets du graphe) et le diamètre D qui est la distance maximale qui peut exister entre deux composants du réseau (en effet, l'efficacité globale du réseau est dépendante de cette quantité qui détermine la vitesse des échanges entre les composants). Ce problème a été posé en 1964 par Elspas [21] sous la forme suivante : «quel est le nombre maximum de sommets d'un (Δ, D)-graphe, un (Δ, D)-graphe étant un graphe de degré maximum Δ et de diamètre D?». On note $N(\Delta, D)$ le nombre maximum de sommets d'un (Δ, D)-graphe.

Nous rappellons ci-après les résultats fondamentaux dont on peut trouver les preuves dans [30].

La borne obtenue sur le nombre maximum de sommets d'un (Δ, D)-graphe est appelée *borne de Moore* (du nom de l'auteur qui est à l'origine des résultats qui vont suivre). Les *graphes de Moore* sont les (Δ, D)-graphes atteignant cette borne.

Proposition 5.2.1 (E. F. Moore, 1958) — *On a l'inégalité suivante :*

$$N(\Delta, D) \leq 1 + \Delta + \Delta(\Delta-1) + \cdots + \Delta(\Delta-1)^{D-1}$$

ce qui donne :

$$\begin{aligned} N(\Delta, D) &\leq \frac{\Delta(\Delta-1)^D - 2}{\Delta - 2} \quad \textit{pour} \quad \Delta \geq 3 \\ \textit{et} \quad N(2, D) &\leq 2D + 1 \end{aligned}$$

Le résultat suivant montre qu'en fait il existe très peu de graphes de Moore.

Théorème 5.2.2 — *Les graphes de Moore existent seulement pour :*

- $\Delta = 2$, *ce sont les cycles* C_{2D+1}.
- $D = 1$, *ce sont les graphes complets* $K_{\Delta+1}$.
- $D = 2$ *et* $\Delta = 3$, *c'est le graphe de* Petersen (voir [30] pour sa définition).

- *$D = 2$ et $\Delta = 7$, c'est le graphe de* Hoffman-Singleton [17].
- *Peut-être pour $D = 2$ et $\Delta = 57$* [14].

Le lecteur trouvera dans les synthèses [13, 14] un état de l'art sur le problème des (Δ, D)-graphes. De plus dans [14], on peut trouver une table des (Δ, D)-graphes ayant le plus grand nombre de sommets connus. Cependant, ce problème a suscité beaucoup d'émulation dans le monde de la recherche en théorie des graphes et de nombreux chercheurs ont essayé d'exhiber les plus grands graphes pour Δ et D fixés. Certains résultats présentés dans les articles de synthèse que nous venons de citer sont peut-être aujourd'hui obsolètes. Aussi, signalons l'interessante initiative de l'équipe de graphes d'Espagne de l'*Universitat Politècnica de Catalunya - Department of Applied Mathematics and Telematics* qui maintiennent à jour des tables accessibles via le réseau Internet par l'URL : `http://www_ mat.upc.es/grup_ de_ grafs/`.

Remarque. Il existe aussi des résultats dans le cas de graphes orientés (voir par exemple [18, 27, 28]). La borne obtenue sur N est alors appelée *borne de Moore orientée*. Le lecteur trouvera pour ce type de problème des références dans [28, 30].

5.2.1 Notre approche du problème

L'une des approches possible pour trouver de larges $(\Delta, D)-$graphes consiste à effectuer des recherches ou constructions sur ordinateur(s) [20]. Comme ce type de recherche procède toujours de façon quasiment exhaustive, elle est très coûteuse en temps et requiert très souvent de forte puissance de calcul. Ainsi, les recherches effectuer jusqu'à présent se sont faites pour de faible degré ($\Delta \leq 16$) et diamètre ($D \leq 10$) [14, 20].

Actuellement, beaucoup des meilleurs $(\Delta, D)-$graphes connus qui ont été obtenu par ordinateurs sont des graphes de Cayley. L'une des explication à cette dominance tient au fait que de tels graphes sont sommets transitifs. Cette propriété implique que le diamètre du graphe est obtenu simplement en calculant l'excenticité d'un sommet quelconque du graphe. Un des groupes souvent utilisé [20] pour construire de tels graphes de Cayley est le « produit semidirect » de deux groupes dans le cas particulier où les groupes concercnés sont $\mathbb{Z}_n$ et $\mathbb{Z}_m$ (où $\mathbb{Z}_q$ représente l'ensemble des entiers modulo q) :

Definition. Soit a un élément de $\mathbb{Z}_n$ dont l'ordre divise m, alors le **produit semidirect** de $\mathbb{Z}_m$ avec $\mathbb{Z}_n$ est défini par :

$$[x, y][u, v] \quad = \quad [x + u \bmod m, ya^u + v \bmod n].$$

Ces groupes sont définis en termes de générateurs et de relations par :

$$m \times_a n = \langle x, y | x^m = y^n = x^{-1}yxy^{-a} = 1 \rangle.$$

L'un des avantages du produit semidirect est qu'il est possible de le programmer simplement de façon efficace. Ainsi, en effectuant une recherche par ordinateur sur des graphes de Cayley définis par un produit semidirect, nous avons pu établir la proposition suivante :

Proposition 5.2.3 — $N(4, 10) \geq 17525$.

PREUVE. Cette borne inférieure est obtenue par un graphe de Cayley définit à partir du produit semidirect de $\mathbb{Z}_m$ avec $\mathbb{Z}_n$ suivant :

$N(4,10)$	Groupe	Générateurs	Inverses
17525	$25 \times_{13} 701$	[3 12] [1 0]	[22 261] [13 701]

□

La précédente meilleure valeur connue était 16555 (voir [20]).

REMARQUE. Signalons que nous avons aussi trouvé un $(4, 10)-$graphe d'ordre intermédiaire, qui possède 17185 sommets. Comme précédemment c'est un graphe de Cayley définit à partir du produit semidirect de $\mathbb{Z}_m$ avec $\mathbb{Z}_n$ suivant :

$N(4,10)$	Groupe	Générateurs	Inverses
17185	$35 \times_{12} 491$	[3 12] [1 0]	[32 310] [23 491]

Notons également que notre recherche par ordinateur nous a aussi permis de retrouver certains $(\Delta, D)-$graphes des meilleurs connus (mais non isomorphes) pour de faibles valeurs de Δ et D. En effet, la faible puissance de calcul dont nous disposions (4 ou 5 Sun Sparc Station, sur lesquels nous n'étions pas les seuls a travailler, et les 16 processeurs $i860$ en réseau de la « *Meïko Machine* ») ne nous a pas permis d'aller au delà de $\Delta = 4$ et $D = 10$ ou $\Delta = 5$ et $D = 9$.

Description de notre algorithme de recherche

L'idée de base de notre algorithme (Cf. alorithme 5.2.1) est classique, puisque pour les valeurs m, n et a fixés ainsi que des valeurs fixés de générateurs, on effectue la fermeture transitive du graphe à partir d'un sommet initial.

Algorithme 5.2.1 — Algorithme de fermeture transitive avec $m \times_a n$

`Initialisation` : Les générateurs sont fixés - Le graphe G ne possède qu'un seul sommet (N:=1) et le diamètre $D := 0$ - Positionner la variable booléenne FIN à FAUX.

- `Répéter`
  Appliquer les générateurs à tous les sommets de G,
  - `Si` il existe $p > 0$ nouveaux sommets n'appartenant pas encore à G `alors` `début faire`

    Insérer tous les nouveaux sommets dans G ;
    $D := D + 1$;
    $N := N + p$;

    `fin faire`
  - `Sinon` $FIN :=$ VRAI.
- `Tant que` FIN n'est pas VRAI.

`En sortie` : D est le diamètre et N l'ordre du graphe.

Amélioration de notre algorithme

L'algorithme 5.2.1 dépend en entrée des valeurs des paramètres m, n, a et des générateurs. Son exécution répétée est clairement très coûteuse en temps et nous avons cherché à diminuer ce coût en essayant au mieux d'éviter de prendre en compte de mauvais graphes.

Pour ce faire, comme nous avons cherché des graphes de degré et diamètre D fixés par avance, notre algorithme s'arrêtait systématiquement lorsque la fermeture transitive d'un graphe ne s'était toujours pas achevé au diamètre maximum D autorisé, c'est-à-dire dès que le graphe avait un diamètre supérieur à D. Dans ce cas l'algorithme recommençait avec de nouveaux paramètres d'entrée. De plus, comme les calculs se font sur les entiers modulo, nous précalculons systématiquement avant le lancement de l'algorithme, une table des inverses. Ceci pour éviter au processeur, d'effecter ce calcul a chaque fois qu'il en aurait besoin. En effet, le temps pour chercher une valeur dans une table (petite) est moindre par raport au temps nécessaire au calcul de l'inverse d'un entier.

Introduction de paramètres de coupure

Dans la suite on suppose $\Delta > 2$. Nous aurons ici besoin des définitions suivantes :

Definition.

- Soit v un sommet d'un graphe G et d'excentricité $ecc(v)$. On apelle *boule stricte de rayon* i, avec $0 \leq i \leq ecc(v)$ et que l'on note $B_i(G)$, l'ensemble des sommets à distance exactement i de v.
- Nous dirons que la boule $B_i(G)$ est maximum si $|B_i(G)| = b_i^{\max}(\Delta) = \Delta(\Delta-1)^{i-1}$.

Par définition $B_0(G) = \{v\}$ et $B_0(G) \bigcup B_1(G) \bigcup \cdots \bigcup B_{ecc(v)}(G) = V(G)$. Nous pouvons remarquer que l'algorithme 5.2.1 procède en fait en construisant systématiquement chacune des boules $B_i(G)$ pour i allant de 0 à $D = ecc(v)$. De plus, $b_i^{\max}(\Delta)$ correspond au nombre maximum de sommets se trouvant exactement à distance i d'un sommet initial dans un graphe de Moore.

Lorsque $|B_0(G)|+|B_1(G)|+\cdots+|B_i(G)|$ atteind la borne de moore du graphe de degré Δ et de diamètre i, alors les boules $B_j(G)$ pour $0 \leq j \leq i$ sont toutes maximales.

Or l'expérience nous a montré que tout les «bons[1]» $(\Delta, D)-$graphes que nous avons exhiber par notre algorithme avaient leurs premières boules maximums. Par exemple pour les « bons » graphes de degré $\Delta = 4$ et de diamètre $D = 10$, les boules $B_1(G)$, $B_2(G)$ et $B_3(G)$ étaient toutes les trois maximums. Lorsqu'une de ces trois premières boules n'était pas maximum, nous obtenions un mauvais graphe. Une explication à ce phénomène consiste à remarquer que si, par exemple, $B_1(G)$ n'est pas maximum alors c'est qu'il y a dès le début de la construction du graphe un petit cycle de longeur 2. Par la suite, et par applications successives des générateurs, ce cycle va se propager et augmenter en nombre dans les autres boules. Ceci entraine alors une perte importante de sommets et le graphe a alors peu de chance d'être « bon ».

Ainsi, dans le but de "repérer" le plus tôt possible les mauvais graphes afin d'éviter d'effectuer une fermeture transitive dessus coûteuse en temps, nous avons introduit des paramètres de coupures C_i sur chaque boule $B_i(G)$. Pour une table de coupure contenant les valeurs de $C_i \in [0,1]$ pour $1 \leq i \leq D$, nous modifions l'algorithme 5.2.1 en introduisant la procédure suivante :

Algorithme 5.2.2 — Procédure de coupure

A l'étape i, calculer $B_i(G)$.
<u>`Si`</u> $|B_i(G)| \geq C_i \cdot b_i^{\max}(\Delta)$ <u>`alors`</u> continuer la recherche <u>`sinon`</u> arrêter la recherche.

1. C'est-à-dire ayant un nombre de sommets proche du meilleur connu.

Typiquement si l'on désire contraindre fortement la recherche sur des premières boules prochent des maximales, il faut alors initialiser les C_i à des valeurs proches de 1. La valeur $C_i = 1$ signifie que la boule i doit être maximum, $C_i = 0$ signifie qu'il n'y a aucune contrainte. Par la suite pour affiner la recherche, il est possible d'initialiser certaines coupures à des valeurs comprises entre 0 et 1.

Automatisation du choix des paramètres de coupure

Le choix initial des paramètres de coupure s'est avéré être une tâche hardue. En effet, ils doivent être initialisés de façon empirique, et pour avoir une idée correcte de leur estimation il s'avère nécessaire d'étudier au préalable le comportement et l'évolution des boules sur de nombreux graphes. Pour nous éviter une telle pré-étude (également coûteuse en temps), nous avons décidé d'automatiser le choix de ces paramètres directement à l'intérieur de notre algorithme.

Notre procédure d'automatisation procède de la façon suivante : initialement tous les paramètres C_i sont fixés à 0, c'est-à-dire que l'on impose aucune contrainte. Dès que l'algorithme trouve un graphe (même mauvais) G_1 atteignant le diamètre D fixé, on initialise chaque C_i par la valeur de $\frac{|B_i(G_1)|}{b_i^{\max}(\Delta)}$, pour $1 \leq i \leq D$. Puis l'algorithme poursuit sa recherche sur un graphe G_2 muni des coupures $C_i = B_i(G_1)$. Ces paramètres de coupure ne changent pas tant que l'algorithme n'a pas trouvé un meilleur graphe que G_1. S'il trouve un graphe G_k meilleur que G_1, alors il réinitialise chaque C_i par la valeur de $\frac{|B_i(G_k)|}{b_i^{\max}(\Delta)}$, pour $1 \leq i \leq D$, et ainsi de suite.

Une telle technique permet de démarrer sans aucune contrainte, c'est-à-dire lorsque l'on n'a aucune connaissance du comportement des graphes. Puis au fur et à mesure du déroulement de l'algorithme, les paramètres de coupure "s'affinent" de plus en plus en fonction du meilleur graphe trouvé.

Même si cette méthode d'automatisation des valeurs de coupure semblait initialement efficace, car devant permettre de très vite réfuter les mauvais graphes, elle n'a pas donné les résultats escomptés. Cela est du au fait suivant : soit les graphes G_1 et G_2, alors si G_2 est meilleur que G_1 cela implique que $\sum_{i=0}^{D} |B_i(G_2)| > \sum_{i=0}^{D} |B_i(G_1)|$, mais en aucun cas que $|B_i(G_2)| > |B_i(G_1)|$ pour tout $0 \leq i \leq D$. En d'autres termes, cela signifie que notre méthode d'automatisation peut être amené à trop contraindre la recherche et ainsi refuter des graphes qui auraient pu être très bons. Pour cette raison nous avons été amenés à modifier cette méthode en introduisant un paramètre de tolérance.

Introduction de paramètres de tolérance sur les coupures

Nous avons introduit à ce niveau, une table supplémentaire de tolérance composé des élements $T_i \in [0, 1]$ avec $1 \leq i \leq D$. Cette table initialisé par avance manuellement sert à pondérer les valeurs de coupures. Ainsi, l'automatisation des

i	C_i	T_i	i	C_i	T_i
1	1.00	0.0	6	0.98	0.1
2	1.00	0.0	7	0.91	0.1
3	1.00	0.0	8	0.70	0.1
4	1.00	0.0	9	0.38	0.1
5	1.00	0.1	10	0.14	0.1

TAB. 5.1 – : *Paramètres de coupure C_i et de tolérance T_i, qui ont permis d'obtenir le graphe de la proposition 5.2.3 ($\Delta = 4$, $D = 10$) possédant 17525 sommets.*

valeurs de coupures s'effectuent exactement comme décrit juste au-dessus, seul la procédure change. Elle devient :

Algorithme 5.2.3 — Procédure de coupure avec tolérance

A l'étape i, calculer $B_i(G)$.
<u>`Si`</u> $|B_i(G)| \geq T_i \cdot C_i \cdot b_i^{\max}(\Delta)$ <u>`alors`</u> continuer la recherche <u>`sinon`</u> arrêter la recherche.

En quelque sorte, cette technique permet de fournir un coefficient de pondération T_i sur chaque coupure C_i reflétant l'importance accordé à la coupure. Si $T_i = 0$, la coupure C_i ne joue aucun rôle, au contraire si $T_i = 1$ alors la coupure n'est pas minorée et doit permettre de contraindre la recherche.

L'expérience acquise sur ce type de recherche tend à montrer qu'il est nécessaire, pour réfuter le plus possible de mauvais graphes, de contraindre fortement les premières coupures. Par contre, pour éviter de réfuter de bon graphe, il est aussi nécessaire de peu contraindre les valeurs centrales des coupures. Cela est du au simple fait que pour être bon un graphe doit au moins débuter avec des boules B_i prochent des boules maxima (sinon, cela signifie qu'il y a présence de petit cycle qui se propageront dans la suite de la construction du graphe, et donc qu'il y aura une perte importante de sommets). Par contre, le comportement des boules $B_i(G)$, n'est pas bien précis. Un graphe ayant quelques boules centrales de mauvaise valeur peut tout de même s'avérer être très bon en fin de compte.

C'est cette dernière technique de recherche qui nous a permis d'obtenir de « bons » graphes, et notamment celui de la proposition 5.2.3. Le tableau 5.1 donne les valeurs finale des C_i ainsi que les valeurs initiales des tolérances T_i utilisées. On remarque sur la table 5.1 que ce graphe possède des boules maxima jusqu'au rayon 5.

Bibliographie

[1] B. Alspach, J-C. Bermond, and D. Sotteau. Decomposition into cycles I: Hamilton decompositions. In G. Hahn et al., editors, *Cycles and Rays*, volume 301 of *NATO ASI Series C: Mathematical and Physical Sciences*, pages 9–18. Kluwer Academic Publishers, Dordrecht, 1990. Proceedings of the NATO Advance Research Workshop on Cycles and Rays: Basic Structures in Finite and Infinite Graphs, Montréal, May 3-9, 1987.

[2] F. S. Annextein, M. Baumslag, and A. L. Rosenberg. Group action graphs and parallel architectures. *SIAM Journal on Computing*, 19:544–569, 1990.

[3] J. Aubert and B. Schneider. Décomposition de la somme cartésienne d'un cycle et de l'union de deux cycles en cycles hamiltoniens. *Discrete Mathematics*, 38:7–16, 1982.

[4] R. Balakrishnan, J-C. Bermond, and P. Paulraja. On the decomposition of de Bruijn graphs into hamiltonian cycles. Manuscript.

[5] D. Barth. *Réseaux d'interconnection : structures et communications.* PhD thesis, Université de Bordeaux I, January 1994.

[6] D. Barth, J. Bond, and A. Raspaud. Compatible eulerian circuits in K_n^{**}. *Discrete Applied Mathematics*, 56:127–136, 1995.

[7] D. Barth and A. Raspaud. Two edge-disjoint hamiltonian cycles in the Butterfly graph. *Information Processing Letters*, 51:175–179, 1994.

[8] C. Berge. *Graphes.* Gauthiers-Villars, 1983.

[9] J-C. Bermond. Hamiltonian decomposition of graphs, directed graphs and hypergraphs. *Advances in Graph Theory, Annals of Discrete Mathematics*, 3:21–28, 1978.

[10] J-C. Bermond. Problem 97. *Discrete Mathematics*, 71:275–276, 1988.

[11] J-C. Bermond, E. Darrot, O. Delmas, and S. Perennes. Hamilton circuits in the directed Butterfly network. Research Report 2925 — Thème 1, INRIA Sophia Antipolis, July 1996. A paraître dans *Discrete Applied Mathematics.*

[12] J-C. Bermond, E. Darrot, O. Delmas, and S. Perennes. Hamilton cycle decomposition of the Butterfly network. Research Report 2920 — Thème 1, INRIA Sophia Antipolis, June 1996. A paraître dans *Parallel Processing Letters.*

[13] J-C. Bermond, C. Delorme, and J-J. Quisquater. Strategies for interconnexion networks: Some methods from graph theory. *Journal of Parallel and Distributed Computing*, 3:433–449, 1986.

[14] J-C. Bermond, C. Delorme, and J-J. Quisquater. Table of large (Δ, D)-graphs. *Discrete Applied Mathematics*, 37/38:575–577, 1992.

[15] J-C. Bermond, O. Favaron, and M. Maheo. Hamiltonian decomposition of Cayley graphs of degree 4. *Journal of Combinatorial Theory*, 46(2):142–153, 1989.

[16] J. Bond and A. Muthusamy. Nonexistence of a Pair of Compatible Euler Tours in Regular Digraphs. Manuscript.

[17] J. A. Bondy and U. S. R. Murty. *Graph theory with applications.* McMillan Press, 1976.

[18] W. G. Bridges and S. Toueg. On the impossibility of directed Moore graphs. *Journal of Combinatorial Theory*, 29:339–341, 1980.

[19] E. Darrot. *Communications par commutation de messages dans les réseaux d'interconnexion.* Thèse de doctorat, Université de Nice - Sophia Antipolis, Laboratoire I3S-CNRS URA 1376, 1996.

[20] M.J. Dinneen and P. Hafner. New results for the degree/diameter problem. *Networks*, 24:359–367, 1994.

[21] B. Elspas. Topological constraints on interconnection limited logic. *Switching circuits theory and Logical design*, 5:133–147, 1964.

[22] H. Fleischner and B. Jackson. Compatible Euler tours in Eulerian digraphs. In Hahn et al. [1], pages 95–100. Proceedings of the NATO Advance Research Workshop on Cycles and Rays: Basic Structures in Finite and Infinite Graphs, Montréal, May 3-9, 1987.

[23] K. Heinrich and H. Verrall. A construction of a Perfect Set of Euler Tours of K_{2k+1}. Manuscript.

[24] J. Liu. Hamiltonian decompositions of Cayley graphs on abelian groups of odd order. *Journal of Combinatorial Theory*, 66:75–86, 1996.

[25] S. Mellendorf. Hamilton decompositions of cartesian products of multicycles. *Journal of Graph Theory*, 24(1):85–115, January 1997.

[26] A. Muthusamy and P. Paulraja. Hamilton Cycle Decompositions of Line Graphs and a Conjecture of Bermond. *Journal of Combinatorial Theory*, 64:1–16, 1995.

[27] S. Perennes. A proof of Jean de Rumeur's conjecture. In [28]. A paraître dans *Discrete Applied Mathematics*.

[28] S. Perennes. *Communications dans les réseaux d'interconnexion*. Thèse de doctorat, Université de Nice - Sophia Antipolis, Laboratoire I3S-CNRS URA 1376, 1996.

[29] R. Rowley and B. Bose. On the number of arc-disjoint hamiltonian circuits in the de Bruijn graph. *Parallel Processing Letters*, 3(4):375–380, 1993.

[30] Jean de Rumeur. *Communication dans les réseaux de processeurs*. Collection Etudes et Recherches en Informatique. Masson, Paris, 1994.

[31] F. Stong. Hamiltonian decompositions of Cartesian products of graphs. *Discrete Mathematics*, 90:169–190, 1991.

[32] T. Tillson. A Hamiltonian decomposition of K^*_{2m}, $2m \geq 8$. *Journal of Combinatorial Theory*, 29:68–74, 1980.

[33] S. Zhan. *Circuits and Cycle Decompositions*. PhD Thesis, Simon Fraser University, Canada, 1992.

Conclusion et perspectives

Un certain nombre de problèmes apparus tout au long de cette thèse, notamment sur les communications et les propriétés des réseaux devraient continuer d'inspirer notre recherche.

Tout d'abord, le fait d'avoir synthétisé dans les chapitres 2 , 3 et 4 des travaux significatifs sur des problèmes de communications globales utilisant le routage de type commutation de circuits, nous a permis d'exhiber de nombreux points encore non résolus ou demandant à être améliorés. Nous souhaitons ainsi poursuivre l'étude des questions suivantes :

- Déterminer des bornes inférieures générales sur le compromis nécessaire entre le nombre d'étapes et le flot d'information. En effet, en section 3.1.2, nous avons résolu le cas où l'on s'autorise jusqu'à deux étapes supplémentaires par rapport à l'optimalité dans un protocole de diffusion. Il semble que pour plus de deux étapes, les phénomènes intervenant soient relativement complexes. Généraliser ce problème permettrait très certainement de bien comprendre les comportements optimaux des réseaux face à ce type de problème.

- Essayer d'appliquer la méthodologie générale de diffusion $\Delta-$ports développée en section 3.3.2 à d'autres réseaux. Par exemple, nous voudrions essayer de l'appliquer au réseau *Butterfly*, afin d'obtenir un résultat encore plus proche de l'optimalité que celui présenté en section 3.3.9.

- Dans le cas où l'on travaille avec une fonction de routage imposée, nous pensons également qu'il est possible d'améliorer des résultats existant notamment dans les grilles ou les tores et d'obtenir des protocoles quasi-optimaux pour le nombre d'étapes.

- Chercher à construire d'autres protocoles essayant de minimiser deux paramètres simultanément, comme par exemple le nombre d'étapes et la longueur des chemins utilisés.

De plus, même si les routages de «type commutation de circuits» sont désormais communément adoptés par les constructeurs de machines parallèles, il apparaît déjà nécessaire d'étudier d'autres techniques de routages comme le mode «diffusion» ou «transparence» décrit en section 1.2.7.

Il semble également, que les réseaux de stations (locaux ou même de taille très large) soient l'une des solutions préssenties pour permettre de réaliser à faible coût des applications à calculs intensifs. Dans le but d'obtenir des bandes passantes efficaces pour ce type d'environnement parallèle, il semble inévitable de faire intervenir des technologies opto-électroniques ou même tout optiques. Cependant, introduire des systèmes d'interconnexion optiques signifie l'émergence de nouveaux problèmes spécifiques encore non résolus. Pour ce type d'approche, l'un des modèles souvent adopté est celui appelé *"wavelength-division multiplexing (WDM)"*. Ce modèle peut être considérer comme une généralisation du routage de «type commutation de circuits». En effet, le mode WDM peut tout du moins dans un premier temps être vu comme de la commutation de circuits colorés, car on peut associer cette fois-ci une couleur (correspondant en fait à une longueur d'onde particulière) à un message. La contrainte du routage correspond alors au fait que sur un lien ne peuvent pas passer simultanément deux messages de même couleur. Néanmoins, plusieurs messages peuvent emprunter simultanément un même lien physique s'ils ont tous des couleurs différentes. Lorsque l'on autorise l'utilisation que d'une seule et unique couleur (i.e. longueur d'onde), on retrouve alors le modèle classique commutation de circuits étudié dans cette thèse. Nous espérons ainsi, pouvoir utiliser efficacement nos connaisances et compétences dans ce domaine, pour les généraliser au modèle optique WDM.

En ce qui concerne l'étude des réseaux, nous souhaitons :

- Etendre les techniques développées en section 5.1.3 afin d'améliorer les résultats concernant la décomposition en circuits hamiltoniens des réseaux de de Bruijn.
- Poursuivre l'étude commencée en section 5.2.1, sur les problèmes des larges $(\Delta, D)-$graphes. Celle-ci devrait être favorisée par le fait que nous avons depuis peu de temps de forte puissance de calcul à notre disposition (16 PentiumPro en réseau, et 4 Alpha 300 Mhz aussi en réseau).

Enfin, dans le cadre du projet commun Cnrs/Inria/Unsa, Sloop, le domaine des réseaux distribués nous apparaît comme le plus propice à valider pratiquement les compétences que nous avons acquises dans le domaine des communications, et des propriétés des graphes. En effet, jusqu'à présent nous avons considéré des problèmes réguliers et précis, et nous souhaiterions aborder l'analyse de réseaux distribués irréguliers. Ceci impliquerait bien entendu de se familiariser avec de nouveaux outils tels que les graphes aléatoires ou l'analyse probabiliste, afin de les associer avec les techniques plus classiques que nous avons acquises.

Annexe A

Diffusion en mode commutation de circuits dans les tores de dimension $\boldsymbol{k}$

Annexe B

Circuit-switched gossiping in the **3**−dimensional torus networks

Annexe C

Hamilton circuits in the directed Butterfly network

Annexe D

Hamilton cycle decomposition of the Butterfly network

Les résultats obtenus dans cette thèse portent principalement sur l'**étude des communications dans les architectures parallèles, distribuées ou réseaux d'interconnexion**.

Dans le **chapitre 1** nous présentons brièvement une rapide classification des machines parallèles. Puis nous décrivons en détails les principaux mécanismes de routage des messages existant à l'heure actuelle dans de telles machines. Nous détaillons en particulier, les nouveaux mécanismes de routage du type «*wormhole*». Ce chapitre contient également un bref rappel des principales notions de théorie des graphes utilisées pour la modélisation des machines parallèles à mémoire distribuée.

Les **chapitres 2, 3 et 4** dressent une synthèse des travaux qui nous paraissent les plus significatifs sur quelques principaux problèmes de communications globales (diffusion, échange total et multidistribution) par commutation de circuits, tout au moins lorsque l'on cherche essentiellement à minimiser le nombre d'étapes des protocoles.

Dans le **chapitre 5** nous résumons en premier lieu nos travaux sur la décomposition hamiltonienne du réseau *Butterfly* généralisé, puis en second lieu nous donnons notre approche au problème des larges graphes à degré et diamètre fixés.

Mots clés: `réseau d'interconnexion, routage par commutation de circuits, communications globales, diffusion, échange total, multidistribution, réseau Butterfly, décomposition hamiltonienne, larges graphes` (Δ, D).

This thesis deals essentially with **communication in interconnection network**.

In **chapter 1**, we present briefly a classification of parallel and distributed computers. We detail the main routing mechanisms in such machines. In particular, we talk about the new routing mechanisms such as «*wormhole*» routing. This chapter also gives an abstract of the main notions of the graph theory that we use for a theorical study of the parallel and distributed computers.

Chapter 2, 3 and 4 are an overview of circuit-switched routing protocols for the basic collective communication problems (one to one, all to all and personalized all to all), but we consider mainly the studies which try to minimize the number of rounds of such protocols.

In **chapter 5**, first we talk about our results on the Hamilton decomposition of the generalized *Butterfly* network and, secondly we give our approach for finding large graphs with fixed degree and diameter.

Key-words: `interconnection network, circuit-switched routing, global communications, broadcasting, gossiping, personalized all to all, Butterfly network, Hamilton decomposition, large` (Δ, D)`-graphs`.