ELSEVIER Theoretical Computer Science 160 (1996) 87–143

Theoretical Computer Science

Fundamental Study

The monadic second-order logic of graphs X: Linear orderings☆

Bruno Courcelle*

Université Bordeaux-I, LaBRI[1], 351, Cours de la Libération, 33405 Talence Cedex, France

Received June 1994; revised February 1995
Communicated by W. Thomas

Abstract

Graphs are finite and handled as relational structures. We give some answers to the following general questions: (1) For which classes of graphs $\mathscr{C}$ is it possible to specify a *linear ordering* of the set of vertices of each graph of $\mathscr{C}$ by fixed *monadic second-order formulas*? (2) For which classes of graphs $\mathscr{C}$ does there exist an extension $\mathscr{L}$ of monadic second-order logic such that a subclass L of $\mathscr{C}$ is *recognizable* if and only if it is the class of graphs in $\mathscr{C}$ that satisfy a formula of $\mathscr{L}$? (In this paper, recognizability is understood in an algebraic sense, relative to a finite set of graph operations and basic graphs that generate all graphs of $\mathscr{C}$.) (3) For which classes of graphs $\mathscr{C}$ is it possible to construct, *in every graph of the class*, and by fixed formulas of a suitable extension of monadic second-order logic, its *hierarchichal structure*, i.e., a finite term written with the operations and basic graphs of (2), that defines the considered graph? Applications concern dependency graphs of partially commutative words, partial k-paths, cographs, and graphs, the *modular decomposition* of which uses prime graphs of bounded size.

Contents

☆ Work supported by ESPRIT Basic Research Working Group COMPUGRAPH II and the "Programme de Recherches Coordonnées: Mathématiques et Informatique".

* Email: Courcell@labri.u-bordeaux.fr.

[1] Laboratoire associé au CNRS.

SSDI 0304-3975(95)00083-6

0. Introduction

This article continues the investigation of Monadic Second-Order Logic as a logical language able to express formally graph properties and graph transformations.

The logical definability of linear orders on finite structures has been considered recently in Hella et al. [17]. Their motivation is to obtain logical characterizations of complexity classes. In particular, a class of linearly ordered structures is in the complexity class PTIME if and only if it is characterized by a formula in a certain extension of first-order logic with fixed point operators. It is still open to obtain a similar characterization of polynomial-time classes of finite unordered structures. On the other hand, the class NP and the other classes of the polynomial hierarchy have logical characterizations in terms of second-order logic, without needing to assume a linear ordering of the structures. It is actually straightforward to specify a linear ordering if quantifications on binary relations are allowed. Hence, the definability of linear orders in finite structures has a certain importance. We shall consider the following question:

Question 1 (Courcelle [10]). *For which classes of finite graphs it is possible to specify a linear ordering of the set of vertices of each graph of $\mathscr{C}$ by fixed monadic second-order formulas?*

This is not possible for all finite graphs: take the discrete (edgeless) graphs; they have nontrivial automorphisms, so no linear order can be defined. Even if we choose in a given discrete graph k sets of vertices (by means of k set variables that we shall call "parameters"), we cannot define a linear order if it is "too large" because discrete graphs with at least $2^k + 1$ vertices have nontrivial automorphisms preserving these k subsets. This explains why the discrete graphs cannot all be linearly ordered by

a unique MS-formula (MS will abbreviate "Monadic Second-order"), even with parameters denoting sets of vertices. Hence, we can only hope to order linearly the graphs of specific classes. In [9] we considered the similar question of specifying by MS-formulas an orientation of the edges of an undirected graph.

Our motivations for investigating the MS-definability of linear orders are different from those of [17]. In addition to the intrinsic interest of Question 1, understanding the relationships between recognizability and MS-definability is our main motivation. The notion of a recognizable set of graphs has been introduced in [6]. It is based on graph congruences with finitely many classes and is relative to operations on graphs that, typically, glue two graphs together or extend in some way a given graph. It is known from Büchi and Doner (see [24]) that a set of words (or of binary trees) is recognizable if and only if it is MS-definable. For trees of unbounded degree, a result of this form is proved in [6] where definability is relative to an extension of MS-logic called *Counting Monadic Second-order logic* (CMS in short). Its formulas are written with special "counting modulo q" existential first-order quantifiers: $\exists^{\mathrm{mod}\,q} x\,\varphi(x)$ means that the number of elements x that satisfy φ is a multiple of q. We ask the following general question, already considered in [7, 9, 18, 25]:

Question 2. *For which classes of finite graphs $\mathscr{C}$ does there exist an extension $\mathscr{L}$ of monadic second-order logic such that, for every $L \subseteq \mathscr{C}$, L is recognizable if and only if it is $\mathscr{L}$-definable?*

In [7] we proved that CMS is the appropriate extension for the class of graphs of tree-width at most 2.

We now explain the links between Questions 1 and 2. Let $\mathscr{C}$ be a class of graphs, let $\mathscr{F}$ be the set of graph operations on $\mathscr{C}$ involved in the intended notion of recognizability, let us also assume that every graph in $\mathscr{C}$ is the value of an $\mathscr{F}$-expression, i.e., of a finite term over $\mathscr{F}$. (The set $\mathscr{F}$ is in some sense a parameter: different sets $\mathscr{F}$ may yield different notions of recognizability.) Assume we have a language $\mathscr{L}$ (say an extension of MS like CMS), for which we know that, if a subset of $\mathscr{C}$ is *$\mathscr{L}$-definable*, then it is recognizable. Assume finally that for every graph G in $\mathscr{C}$ we can construct "in G" an $\mathscr{F}$-expression that defines this graph. Then, if L is a recognizable subset of $\mathscr{C}$, there exists a finite tree-automaton recognizing the set of $\mathscr{F}$-expressions, the value of which is in L. Given a graph G we can express that $G \in L$ by means of a formula in $\mathscr{L}$ that works as follows:

(1) It defines in G an $\mathscr{F}$-expression, the value of which is G.

(2) It checks whether the automaton accepts this expression:

the graph G is in L if and only if the automaton accepts the expression, if and only if the MS-formula holds.

So the logical language $\mathscr{L}$ must not be too powerful (we want that every $\mathscr{L}$-definable class of graphs be recognizable) but it must be powerful enough to do two things:

(1) to "parse" the graph (i.e., to define an $\mathscr{F}$-expression for it) and

(2) to simulate the behavior of a finite automaton on the obtained $\mathscr{F}$-expression.

In some cases, a *linear ordering of the given graph helps to parse it by MS-formulas*: this is the link between Questions 1 and 2. We now list the results of the paper.

Section 1 reviews definitions. In Section 2, we show how to define topological sortings of dags (directed acyclic graphs) by MS-formulas. Two constructions are given. The second one is less general but gives a topological sorting characterized as the minimal one with respect to a certain lexicographical order. As a consequence, we can construct by MS-formulas a path-decomposition of any k-connected partial k-path. This is a further step towards the proof of the conjecture made in [7] that one can construct by MS-formulas a tree-decomposition of any partial k-tree (and thus answer to Question 2 for partial k-trees by showing that CMS-definability is equivalent to recognizability).

Section 3 gives a new proof of results by Ochmanski [19] and Thomas [25] concerning recognizable sets of traces and their dependency graphs. Section 4 shows that every graph property that is expressible in MS-logic with the help of an auxiliary "invariant" *linear order* on the vertices is recognizable. All properties expressible in CMS-logic are of this form: a linear ordering is helpful to express that a set X has an even number of elements, because one can divide X into two parts, the elements of even rank and those of odd rank, where ranks are relative to this linear order; X has even cardinality if and only if the last element has even rank; the answer, namely the parity of the cardinality of X, *does not depend on the chosen ordering* of the vertices although it is expressed logically in terms of this order: this is what the term "invariant" means. This extension of MS-logic that we denote by MS($\leqslant$) is our most expressive candidate for logical characterizations of recognizable sets of graphs. For trees, recognizability, CMS-definability and MS($\leqslant$)-definability are equivalent notions, all strictly larger than MS-definability.

In Section 5, we show that with the help of such an auxiliary "invariant" ordering, we can reconstruct the "internal structure" of a tree from its leaves and from a ternary relation on leaves that "projects" the internal structure. We can do the same reconstruction *without any linear order* for the class of trees having a degree bounded by any fixed constant. In Section 6, we apply this to prove that the unique *modular decomposition of any graph* can be defined by MS($\leqslant$)-formulas (i.e., MS-formulas using an auxiliary invariant linear order). We obtain logical characterizations of recognizable sets of cographs and of graphs having modular decompositions with prime graphs of bounded size. Section 7 is a review of results and open questions.

1. Basic definitions

1.1. Graphs

All graphs will be finite, directed, simple (no two edges have the same ordered pair of vertices). A graph will be given as a pair $G = \langle \mathbf{V}_G, \mathbf{edg}_G \rangle$ where $\mathbf{V}_G$ is the set of vertices and $\mathbf{edg}_G \subseteq \mathbf{V}_G \times \mathbf{V}_G$ is the edge relation. Undirected graphs will be defined as

(directed) graphs with a symmetric edge relation. We shall assume that $\mathbf{V}_G$ is always a subset of a fixed countable "universal set of vertices" that we need not specify. It follows that all graphs form a set. (A "class" or a "family" of graphs will always be a set in the sense of set theory.) If $X \subseteq \mathbf{V}_G$ we denote by $G[X]$ the induced subgraph of G with set of vertices X. A *path* is a sequence of pairwise distinct vertices $(x_1, \ldots, x_n)$ such that $(x_i, x_{i+1}) \in \mathbf{edg}_G$ for every $i = 1, \ldots, n-1$. It *connects* x_1 to x_n. It is *empty* if $n = 1$. A *cycle* is like a path except that $x_1 = x_n$ and $n > 1$. A graph *is a path* if its vertices form a path $(x_1, \ldots, x_n)$ and all edges of the graph are in the path , i.e., are of the form (x_i, x_{i+1}) for some i. A *discrete* graph is a graph without edges.

We let $\mathbf{Suc}_G(x) := \{y \mid (x, y) \text{ is an edge}\}$ and we call it the set of *successors* of x. The relation $\mathbf{edg}_G$ will also be called the *successor relation*. We say that x is a *predecessor* of y if y is a successor of x. The *outdegree* of G is the maximal cardinality of the sets $\mathbf{Suc}_G(x)$. A graph without cycles is called a *dag* (directed acyclic graph); a *tree* is a dag such that every vertex is reachable by a unique path from a (necessarily unique) vertex called the *root*. A *forest* is a dag, each connected component of which is a tree; hence, a forest that is not a tree has several roots. A vertex without successors is called a *leaf*. The transitive closure of a graph G is a graph denoted by G^+.

If G is a dag, the relation $\mathbf{edg}_G^*$ (the reflexive and transitive closure of the successor relation) is a partial order on $\mathbf{V}_G$. Two vertices x and y are *comparable* if $x\, \mathbf{edg}_G^*\, y$ or $y\, \mathbf{edg}_G^*\, x$; otherwise they are *incomparable* and we write this as $x \perp_G y$. The *reduction* of a dag G is the least subgraph H of G such that $H^+ = G^+$. It is unique and denoted by $\mathbf{red}(G)$; it is the Hasse diagram of the partial order $\mathbf{edg}_G^*$. A *topological sorting* of a dag G is a linear order $\leqslant$ on its set of vertices vertices such that $x \leqslant y$ for every edge (x, y) of G. We say that a graph G is *linear* if it is a dag and any two vertices are linked by an edge (equivalently, G is an acyclic tournament); its reduction is a path and the order $\mathbf{edg}_G^*$ is linear.

1.2. Relational structures and monadic second-order logic

Let R be a finite set of relation symbols where each element r in R has rank $\rho(r)$ in $\mathbb{N}_+$, which will be the arity of relations denoted by r. An *R-(relational) structure* is a tuple $S = \langle \mathbf{D}_S, (r_S)_{r \in R} \rangle$ where $\mathbf{D}_S$ is a finite (possibly empty) set (also a subset of the "universal set of vertices"), called the *domain* of S, and r_S is a subset of $\mathbf{D}_S^{\rho(r)}$ for each r in R. We shall denote by $\mathscr{S}(R)$ the set of R-structures.

The formulas of *monadic second-order logic*, intended to describe properties of R-structures S, are written with lowercase symbols $x, x', y, \ldots$ called *object variables*, ranging over elements of $\mathbf{D}_S$, and uppercase symbols $X, Y, Y', \ldots$ called *set variables*, ranging over subsets of $\mathbf{D}_S$. The atomic formulas are of the forms $x = y$, $x \in X$, and $r(x_1, \ldots, x_n)$, where r is in R and $n = \rho(r)$, and formulas are formed with propositional connectives and quantifications over the two kinds of variables. For every finite set W of object and set variables, we denote by $\mathscr{L}(R, W)$ the set of all formulas that are written with relational symbols from R and have their free variables in W; we also let $\mathscr{L}(R) := \mathscr{L}(R, \emptyset)$ denote the set of closed formulas. A formula is

first-order if it has no set quantification (it may have set variables). *Counting monadic second-order* logic is the extension of monadic second-order logic with the *counting modulo q* existential first-order quantifiers such that $\exists^{\mathrm{mod}\,q} x\,\varphi(x)$ means that the number of elements x that satisfy φ is a multiple of q. We shall use *MS* for "monadic second-order" and *CMS* for "counting monadic second-order".

Let S be an R-structure, let $\varphi \in \mathscr{L}(R, W)$, and let γ be a W-assignment in S (i.e., $\gamma(X)$ is a subset of $\mathbf{D}_S$ for every set variable X in W, and $\gamma(x) \in \mathbf{D}_S$ for every object variable x in W). We write $(S, \gamma) \models \varphi$ if and only if φ holds in S for γ. We write $S \models \varphi$ in the case where φ has no free variable. A set of R-structures L is MS-*definable* if there is a formula φ in $\mathscr{L}(R)$ such that L is the set all R-structures S such that $S \models \varphi$; it is closed under isomorphism.

A graph is thus an $\{\mathbf{edg}\}$-structure $G = \langle \mathbf{V}_G, \mathbf{edg}_G \rangle$ where **edg** is binary. We shall say that a property P of the graphs G of a class $\mathscr{C}$ is MS-*expressible* if there is an MS-formula φ such that, for every G in $\mathscr{C}$, the property $P(G)$ holds if and only if $G \models \varphi$. Any two isomorphic graphs satisfy the same MS-expressible properties. Similarly, we shall consider CMS-*expressible* properties.

In other articles [6–10], we considered MS- and CMS-formulas where set variables can also denote sets of edges. These extended languages are denoted by MS_2 and CMS_2, respectively. They are still monadic second-order languages, but relative to a different representation of graphs by relational structures: the edges become elements of the domains and appropriate relations express incidence. The language MS_2 is in general strictly more expressive than MS, but for particular types of graphs like those of degree bounded by a fixed constant, they are equally powerful (see [10]). Since we shall not write explicit MS_2- and CMS_2-formulas we need not define formally these languages.

1.3. Definable transductions of structures

The notion of (*monadic second-order*) *definable transduction of structures* is surveyed in [11]. Let R and Q be two finite ranked sets of relation symbols. Let W be a finite set of set variables, called the set of *parameters*. A (Q, R)-*definition scheme* is a tuple of formulas of the form

$$\Delta = (\varphi, \psi_1, \ldots, \psi_k, (\theta_w)_{w \in Q^* k}),$$

where

$$k > 0, \quad Q^*k := \{(q, j) \mid q \in Q, j \in [k]^{\rho(q)}\},$$

$$\varphi \in \mathscr{L}(R, W),$$

$$\psi_i \in \mathscr{L}(R, W \cup \{x_1\}) \quad \text{for } i = 1, \ldots, k,$$

$$\theta_w \in \mathscr{L}(R, W \cup \{x_1, \ldots, x_{\rho(q)}\}) \quad \text{for } w = (q, j) \in Q^*k.$$

(We denote by $[k]$ the integer interval $\{1, \ldots, k\}$.) These formulas are intended to define a structure T is $\mathscr{S}(Q)$ from the structure S in $\mathscr{S}(R)$ and will be used in the following way: T is well-defined only if φ holds true in S; assuming this condition is

fulfilled, the domain of T is the disjoint union of the sets $D_1, \ldots, D_k$ where D_i is the set of elements in the domain of S that satisfy ψ_i; finally, the formulas θ_w for $w = (q, j)$, $j \in [k]^{\rho(q)}$, define the relation q_T. Here are the formal definitions. Let $S \in \mathscr{S}(R)$, let γ be a W-assignment in S. A Q-structure T with domain $\mathbf{D}_T \subseteq \mathbf{D}_S \times [k]$ is *defined in* (S, γ) *by* Δ if:

(i) $(S, \gamma) \models \varphi$,

(ii) $\mathbf{D}_T = \{(d, i) \mid d \in \mathbf{D}_S,\ i \in [k],\ (S, \gamma, d) \models \psi_i\}$,

(iii) for each q in Q: $q_T = \{((d_1, i_1), \ldots, (d_t, i_t)) \in \mathbf{D}_T^t \mid (S, \gamma, d_1, \ldots, d_t) \models \theta_{(q,j)}\}$, where $j = (i_1, \ldots, i_t)$ and $t = \rho(q)$.

(By $(S, \gamma, d_1, \ldots, d_t) \models \theta_{(q,j)}$, we mean $(S, \gamma') \models \theta_{(q,j)}$, where γ' is the assignment extending γ, such that $\gamma'(x_i) = d_i$ for all $i = 1, \ldots, t$; a similar convention is used for $(S, \gamma, d) \models \psi_i$.) Since T is associated in a unique way with S, γ and Δ whenever it is defined, we can use the functional notation $\mathbf{def}_\Delta(S, \gamma)$ for T. The *transduction defined by* Δ is the relation

$$\mathbf{def}_\Delta := \{(S', T') \mid S' \text{ is isomorphic to } S,\ T' \text{ is isomorphic to } T,\ T = \mathbf{def}_\Delta(S, \gamma) \text{ for some } W\text{-assignment } \gamma \text{ in } S\} \subseteq \mathscr{S}(R) \times \mathscr{S}(Q).$$

A transduction $f \subseteq \mathscr{S}(R) \times \mathscr{S}(Q)$ is *definable* if it is equal to $\mathbf{def}_\Delta$ for some (Q, R)-definition scheme Δ.

We shall use the facts concerning definable transductions that are collected in the following proposition [7, Proposition 2.5; 11, Proposition 3.2]:

Proposition 1.1. (1) *The inverse image of an MS-definable set of structures under a definable transduction is an MS-definable set of structures.*

(2) *The inverse image of a CMS-definable set of structures under a definable transduction is a CMS-definable set of structures.*

(3) *The composition of two definable transductions is a definable transduction.*

The proof is based on the immediate observation that if $S = \mathbf{def}_\Delta(T, \mu)$, then an MS-property P of S can be expressed as an MS-property P' of (T, μ), where the translation of an MS-formula expressing P into an MS-formula expressing P' is effective and depends only on Δ. (See [11, Proposition 3.2] for a formal statement.)

1.4. Recognizable sets

Let $\mathscr{S}$ be a possibly infinite set of *sorts*. An $\mathscr{S}$-signature is a set F such that each (function symbol) f in F has a *type* of the form $s_1 \times s_2 \times \cdots \times s_n \to s$ where $s_1, \ldots, s_n, s$ are sorts. An *F-magma* (usually called an *F-algebra*) is an object $M = \langle (M_s)_{s \in \mathscr{S}}, (f_M)_{f \in F} \rangle$, where, for each s in $\mathscr{S}$, M_s is a set called the *domain of sort* s of M, and for each $f \in F$ of type $s_1 \times s_2 \times \cdots \times s_n \to s$, f_M is a total mapping: $M_{s_1} \times M_{s_2} \times \cdots \times M_{s_n} \to M_s$.

We denote by $\mathbf{T}(F)$ the F-magma of *finite terms over* F and by $\mathbf{h}_M$ the unique homomorphism: $\mathbf{T}(F) \to M$ that associates with a term its value. We shall say that t is a term *denoting* $\mathbf{h}_M(t)$.

We now review the notion of a *recognizable set*. Let F and $\mathcal{S}$ be as above. An F-magma A is *locally finite* if each domain A_s $s \in \mathcal{S}$, is finite. Let M be an F-magma and $s \in \mathcal{S}$. A subset B of M_s is *recognizable* if there exists a locally finite F-magma A, a homomorphism $h: M \to A$, and a (finite) subset C of A_s such that $B = h^{-1}(C)$. In order to specify the relevant set of operations or the relevant magma, we shall also say that B is *F-recognizable* or *M-recognizable*.

Proposition 1.2. *Let M and N be two F-magmas, let h be a homomorphism of N onto M: a subset L of M_s is F-recognizable if and only if the subset $h^{-1}(L)$ of N_s is F-recognizable. In particular, if N is $\mathbf{T}(F)$, F is finite and every element of M is the value of some term in $\mathbf{T}(F)$, then L is F-recognizable if and only if $h^{-1}(L)$ is a recognizable set of terms.*

Since recognizable sets of terms can be effectively handled by means of tree-automata (see the monograph [16]), this proposition gives an effective way to describe and manipulate recognizable sets.

Proposition 1.3. *Let M be an F-magma. Let N be a G-magma, the domains of which are domains of M and the operations of which are defined by finite combinations of operations of M. If a subset of a domain of N is F-recognizable then it is G-recognizable.*

Proof. An immediate consequence of the definitions since a homomorphism: $M \to A$ yields a homomorphism: $N \to A'$ where A' is a G-magma with the same domains as A. □

1.5. Graph operations

The notion of a recognizable set of graphs results immediately from the definition of a signature F of graph operations, making the set of graphs into an F-magma. We shall use operations concerning graphs with distinguished vertices called *sources*. We shall use $\mathbb{N}$ as set of *source labels*. *A graph with sources* is a pair $H = \langle G, s \rangle$ consisting of a graph G and a total one-to-one mapping $s: C \to \mathbf{V}_G$ called its *source mapping*, where C is a finite subset of $\mathbb{N}$. We say that $s(C) \subseteq \mathbf{V}_G$ is the *set of sources of H* and that $s(c)$ is its *c-source* where $c \in C$. We shall also say that the vertex $s(c)$ has *source label c*. A vertex that is not a source is an *internal vertex*. The set C is called the *type of H* and is denoted by $\tau(H)$. We shall denote by $\mathbb{G}_C$ the set of all graphs of type C. We shall denote by $\mathbf{G}_C$ the set of *abstract* graphs of type C, i.e., of isomorphism classes of graphs in $\mathbb{G}_C$. The graphs in $\mathbb{G}_C$ will be called *concrete* for emphasizing the distinction with abstract ones. We shall use the set $\mathcal{S}$ of finite subsets of $\mathbb{N}$ as a set of sorts. We now define some operations on concrete and abstract graphs with sources. These operations will form an $\mathcal{S}$-signature.

1. Parallel composition: If $K \in \mathbb{G}_C$ and $K' \in \mathbb{G}_{C'}$, the *parallel composition* of K and K', denoted by $K \|_{C,C'} K'$, is defined if and only if:

(1) For every $c \in C \cap C'$ the c-source of K is equal to the c-source of K'.

(2) Every vertex in $\mathbf{V}_K \cap \mathbf{V}_{K'}$ is of this form for some c in $C \cap C'$.

If these conditions hold, then $K \|_{C,C'} K'$ is the graph H in $\mathbb{G}_{C \cup C'}$ such that $\mathbf{V}_H = \mathbf{V}_K \cup \mathbf{V}_{K'}$, $\mathbf{edg}_H = \mathbf{edg}_K \cup \mathbf{edg}_{K'}$, the c-source of K is the c-source of H for every c in C, and the c-source of K' is the c-source of H for every c in C'. Note that an edge (x, y) belongs to K and to K' if x and y are sources in these two graphs with same labels. It yields a unique edge of H and not a pair of parallel edges. (We only consider simple graphs.)

If $G \in \mathbf{G}_C$ and $G' \in \mathbf{G}_{C'}$, their *parallel composition* $H = G \|_{C,C'} G'$ (in $\mathbf{G}_{C \cup C'}$) is defined as the isomorphism class of $K \|_{C,C'} K'$ for suitable graphs K and K' respectively isomorphic to G and G'. It is well-defined since graph isomorphism is a congruence for parallel composition of concrete graphs.

For concrete as well as for abstract graphs G and G', we shall use the overloaded notation $G \| G'$ when C and C' are known from the context or are irrelevant.

2. Source renaming: For every one-to-one mapping $h: C \to C'$, we let $\mathbf{ren}_h: \mathbb{G}_{C'} \to \mathbb{G}_C$ be the total mapping such that $\mathbf{ren}_h(\langle G, s\rangle) = \langle G, s \circ h\rangle$. In other words, the c-source of $\mathbf{ren}_h(\langle G, s\rangle)$ is defined as the $h(c)$-source of G. If $c \in C' - h(C)$ the c-source of $\langle G, s\rangle$ is an internal vertex in $\mathbf{ren}_h(\langle G, s\rangle)$. We shall say that $\mathbf{ren}_h(\langle G, s\rangle)$ is the *source renaming* of $\langle G, s\rangle$ defined by h. Since the mapping $\mathbf{ren}_h$ commutes with graph isomorphisms, we have a total mapping $\mathbf{ren}_h: \mathbf{G}_{C'} \to \mathbf{G}_C$. If h is the identity mapping: $C \to C \cup P$, where $C \cap P = \emptyset$, we shall denote $\mathbf{ren}_h$ by $\mathbf{fg}_P$: we say that $\mathbf{fg}_P$ "forgets" the p-sources for p in P.

A set of abstract graphs is *recognizable* if it is with respect to the operations $\|_{C,C'}, \mathbf{ren}_h$. A subset of $\mathbb{G}_C$ is *recognizable* if the set of isomorphism classes of its members is recognizable.

The notion of recognizability is thus associated with certain graph operations. It is robust in the sense that small variations on the definitions of the operations do not modify it (this is shown in [12]). However, in Section 6, we shall introduce completely different graph operations, based on substitutions of graphs for vertices in graphs, yielding a different notion of recognizability.

It is proved by Courcelle [6] that every CMS-definable set of graphs is recognizable. Obtaining logical characterizations of recognizability has been considered in [7] and is also one of the motivations of the present article.

2. Monadic second-order definitions of linear orders

Certain graphs are linearly ordered in a natural way: so are typically the paths and the acyclic tournaments. Others are not, for instance the discrete (edgeless) graphs. Is it possible to specify a linear order on every graph by fixed MS-formulas? The answer is "no" and the counterexample is given by the discrete graphs.

We shall thus ask for which classes of graphs this is possible. Here is a precise definition.

Let $\mathscr{S}$ be a class of R-structures (that may represent the graphs of a fixed class). We say that a *linear order on the structures of* $\mathscr{S}$ *is MS-definable* if there exist two MS-formulas $\varphi(X_1, \ldots, X_n)$ and $\theta(x, y, X_1, \ldots, X_n)$ such that for every S in $\mathscr{S}$:

(1) $S \models \exists X_1, \ldots, X_n \varphi$.

(2) For all sets $D_1, \ldots, D_n \subseteq \mathbf{D}_S$, if $(S, D_1, \ldots, D_n) \models \varphi$ the binary relation P such that

$$(u, v) \in P \Leftrightarrow (S, u, v, D_1, \ldots, D_n) \models \theta$$

is a linear order on $\mathbf{D}_S$.

Note that the linear order is defined "uniformly", by the same formulas for all structures of the class, and in terms of parameters $D_1, \ldots, D_n$. In other words, there exists a definable transduction mapping any structure S in $\mathscr{S}$ into a structure S' consisting of S equipped with a linear order of its domain (this linear order will be denoted by a new binary relation symbol, usually $\leqslant$). This does not mean that *every* linear order on the domain of S is obtained in this way, by some choice of sets $D_1, \ldots, D_n$.

In this section, we shall give two constructions of linear orders on dags, both of them being MS-definable. The first one concerns dags for which one already knows a linear ordering of the set of successors of each vertex. From these "local orderings" we shall construct, using depth-first search, a certain topological sorting of the given dag. The second one concerns dags with bounded antichains and constructs topological sortings that are, in a certain sense, lexicographically minimal.

2.1. Locally ordered dags

A vertex of a dag G having no predecessor is called a *root* and $\mathbf{Root}_G$ denotes the set of roots of G. (Since graphs are finite, if a dag G is nonempty, then $\mathbf{Root}_G$ is nonempty.) A partial order α on $\mathbf{V}_G$ *locally orders* G if the sets $\mathbf{Root}_G$ and $\mathbf{Suc}_G(x)$ for every $x \in \mathbf{V}_G$ are linearly ordered by α; we let $\mathbf{Paths}(G)$ denote the set of paths in G starting from a root; $\mathbf{Paths}(G)$ is linearly ordered by $\leqslant_\alpha$ where $\leqslant_\alpha$ is the lexicographical order of sequences of vertices associated with α. For each $x \in \mathbf{V}_G$, we let $\pi(x)$ denote the unique $\leqslant_\alpha$-minimal path from a root to x. For $x, y \in \mathbf{V}_G$, we let $x \leqslant_\alpha y$ if and only if $\pi(x) \leqslant_\alpha \pi(y)$. Hence, $\leqslant_\alpha$ is a linear order on $\mathbf{V}_G$. The enumeration of $\mathbf{V}_G$ in increasing order with respect to $\leqslant_\alpha$ is called the *α-depth-first traversal* of G. It is nothing but the order in which the vertices of G are visited during the depth-first search of G obtained from the following rules:

(1) Start at the α-smallest root,

(2) Whenever the current vertex has unvisited vertices, visit next the α-smallest one,

(3) Whenever the current vertex is a root, all successors of which have been visited, visit next the α-smallest unvisited root; if there is no unvisited root, then the search is finished.

We let P be the binary relation on $\mathbf{V}_G$ such that

$(x, y) \in P \Leftrightarrow x$ is just before y on the path $\pi(y)$.

The graph $\mathbf{F}(G, \alpha) := \langle \mathbf{V}_G, P \rangle$ is the *α-depth-first spanning forest*. See [2] for details. In particular, since G is a dag, an edge (x, y) of G can be only of 3 types, by [2, Lemma 5.6]:

(1) either it is a *tree edge*, i.e., an edge of $\mathbf{F}(G, \alpha)$,

(2) or it is a *forward edge*, i.e., x is an ancestor of y in some tree of $\mathbf{F}(G, \alpha)$, but (x, y) is not an edge of $\mathbf{F}(G, \alpha)$,

(3) or it is a *cross edge*, i.e., x and y are incomparable in $\mathbf{F}(G, \alpha)$ and $y <_\alpha x$.

Finally, we define the *α-canonical traversal* of G as the α-depth-first traversal of $\mathbf{F}(G, \alpha^{-1})$ where α^{-1} is the opposite order of α (i.e., $(x, y) \in \alpha^{-1}$ if and only if $(y, x) \in \alpha$). It orders G locally if and only if α does.

Theorem 2.1. *Let G be a dag locally ordered by α. The α-canonical traversal of G is a topological sorting that is MS-definable in $\langle \mathbf{V}_G, \mathbf{edg}_G, \alpha \rangle$.*

Proof. Consider any edge (x, y) of G. If it is a tree edge or a forward edge (in $\mathbf{F}(G, \alpha^{-1})$), then x is before y in the α-depth-first traversal of $\mathbf{F}(G, \alpha^{-1})$. Otherwise it is a cross-edge, hence $x >_{\alpha^{-1}} y$, and $x <_\alpha y$ where $<_\alpha$ is defined in terms of the paths in $\mathbf{F}(G, \alpha^{-1})$; hence, x is before y in the α-depth-first traversal of $\mathbf{F}(G, \alpha^{-1})$. Hence this traversal is a topological sorting of G. We now formalize its definition in MS.

Claim. *If G is a forest locally ordered by α then its α-depth-first traversal is MS-definable.*

Proof. The relation $x \leqslant_\alpha y$ defined by $\pi(x) \leqslant_\alpha \pi(y)$ can be written as follows (where $x\,\mathbf{edg}_G\,y$ is another notation for $(x, y) \in \mathbf{edg}_G$): $x \leqslant_\alpha y$ if and only if

(1) either $x'\,\mathbf{edg}_G^*\,x, y'\,\mathbf{edg}_G^*\,y$ for some roots x' and y' such that $(x', y') \in \alpha$, $x' \neq y'$,

(2) or $x\,\mathbf{edg}_G^*\,y$,

(3) or $z\,\mathbf{edg}_G\,x''\,\mathbf{edg}_G^*\,x$ and $z\,\mathbf{edg}_G\,y''\,\mathbf{edg}_G^*\,y$ for some z, x'' and y'' such that $x'' \neq y''$ and $(x'', y'') \in \alpha$.

This can be written in MS-logic. □

It is thus enough to show that $\mathbf{F}(G, \alpha^{-1})$ is MS-definable in $\langle \mathbf{V}_G, \mathbf{edg}_G, \alpha \rangle$, because having defined this forest (namely, its edges), we can MS-define its α-depth-first traversal by the claim. We shall do the proof for $\mathbf{F}(G, \alpha)$ in order to simplify the notation. The result will follow since α^{-1} is definable from α. We shall basically translate into MS-formulas the various notions involved in the definition of $\mathbf{F}(G, \alpha)$. The reduction $\mathbf{red}(H)$ of any dag H is the graph K with the same vertices as H and edges defined as follows:

$(x, y) \in \mathbf{edg}_K \Leftrightarrow (x, y) \in \mathbf{edg}_H$ and there is no $z \neq y$ such that $x\,\mathbf{edg}_H\,z$ and $z\,\mathbf{edg}_H^+\,y$.

We claim the existence of formulas $\varphi_1, \ldots, \varphi_7$ with the semantics given below, in a dag G locally ordered by α. The letter X denotes sets of vertices; letters $x, y, z \ldots$ denote vertices of the considered dag G.

$\varphi_1(x, y, X) \Leftrightarrow$ $x, y \in X$, $x \neq y$, and there is a directed path in $G[X]$ (the induced subgraph of G with set of vertices X) from x to y.

$\varphi_2(x, y, X) \Leftrightarrow$ $x, y \in X$, (x, y) is an edge of the graph $\mathbf{red}(G[X])$. (Write that $(x, y) \in \mathbf{edg}_G$ and that there is no z in $X - \{x, y\}$ such that $x\, \mathbf{edg}_G\, z$ and $\varphi_1(z, y, X')$ holds where $X' = X - \{x\}$.)

$\varphi_3(x, y) \Leftrightarrow$ $x \neq y$ and G is a path from x to y. (Write that $\varphi_1(x, y, X)$ holds for $X = \mathbf{V}_G$ and that it does not if X is any proper subset of $\mathbf{V}_G$ containing x and y.)

$\varphi_4(x, y, X) \Leftrightarrow$ $x \neq y$ and X is the set of vertices of a directed path from x to y. (Write that the graph $\mathbf{red}(G[X])$ is a path from x to y).

$\varphi_5(x, y, z, t, X) \Leftrightarrow$ $\varphi_4(x, y, X)$ holds, $z, t \in X$ and z is the predecessor of t on the path $\mathbf{red}(G[X])$ from x to y. (Write that $\varphi_4(x, y, X)$ holds and (z, t) is an edge of $\mathbf{red}(G[X])$.

$\varphi_6(x, y, X) \Leftrightarrow$ $x \neq y$ and X is the set of vertices of the minimal path from x to y. (Write that $\varphi_4(x, y, X)$ holds and that there does not exist $z, t, t' \in \mathbf{V}_G$ such that $\varphi_5(x, y, z, t, X)$ holds, $(z, t') \in \mathbf{edg}_G$, $t'\, \mathbf{edg}_G^*\, y$ and t' is strictly smaller than t with respect to α.)

$\varphi_7(x, y) \Leftrightarrow$ there exist x and z such that z is a root of G, $\varphi_6(z, y, X)$ and $\varphi_5(z, y, x, y, X)$ hold.

Hence $\varphi_7(x, y)$ holds if and only if (x, y) is an edge of $\mathbf{F}(G, \alpha)$. This concludes the proof. □

In the following result, we need not use a given local ordering, because we can define one.

Corollary 2.2. *For each $d \in \mathbb{N}$, some depth-first traversal of trees of outdegree at most d is MS-definable.*

Proof. Let $T = \langle \mathbf{V}_T, \mathbf{edg}_T \rangle$ be a tree of outdegree at most d. A partition $V_1, \ldots, V_d$ of $\mathbf{V}_T$ is a *good partition* if no two successors of a same vertex belong to the same set V_i. Clearly there exists a good partition of $\mathbf{V}_T$ in at most d classes. A partial order on $\mathbf{V}_T$ can be obtained from $(V_1, \ldots, V_d)$ as follows:

$(u, v) \in \alpha$ if and only if either $u = v$ or $u \in V_i$, $v \in V_j$ for some $1 \leqslant i < j \leqslant d$.

One can thus construct two MS-formulas $\varphi(X_1, \ldots, X_d)$ and $\theta(x, y, X_1, \ldots, X_d)$ such that, for every tree T of outdegree at most d, for every $V_1, \ldots, V_d \subseteq \mathbf{V}_T$:

(1) $(T, V_1, \ldots, V_d) \models \varphi$ if and only if $(V_1, \ldots, V_d)$ is a good partition.

(2) If $(V_1, \ldots, V_d)$ satisfies φ, then the relation P on $\mathbf{V}_T$ such that

$$(u, v) \in P \text{ if and only if } (T, u, v, V_1, \ldots, V_d) \models \theta$$

is the α-depth-first traversal of T, where α is associated with $V_1, \ldots, V_d$.

Hence P is a depth-first traversal, defined by the formula θ in terms of sets $V_1, \ldots, V_d$ provided these sets satisfy φ. Since for every tree of outdegree at most d one can find $V_1, \ldots, V_d$ satisfying φ, one obtains in this way a depth-first traversal of every such tree. □

In Theorem 2.1, we showed how to construct *some* topological sorting. In certain applications (in particular to traces, see Section 3), one may wish to construct *a specific one.*

Let us assume that α is a linear ordering of $\mathbf{V}_G$ where G is a dag. There exists a $\leqslant_\alpha$-minimal topological sorting of G where $\leqslant_\alpha$ is the lexicographic ordering on topological sortings considered as sequences of vertices associated with α. The topological sorting of a dag G defined in Theorem 2.1 is *not* this one. Take for example the tree with vertices a, b, c, d, e, with root a and edges (a, b), (a, c), (a, e), (b, d). Consider the ordering α: $a < b < c < d < e$. The minimal topological sorting is a, b, c, d, e and the α-depth-first traversal is a, b, d, c, e, which is different. (Anticipating on Section 3, we may note that this tree is the dependency graph of a trace.)

Open question 2.3. *Let T be a tree given with a linear order α of its vertices. Can one MS-define its $\leqslant_\alpha$-minimal topological sorting?*

2.2. Dags with bounded antichains

We shall now give an MS-definition of the $\leqslant_\alpha$-minimal topological sorting of a dag with antichains of size at most k where α is defined from a given covering of the dag by chains.

Let G be a dag. A *chain* in G is a set of vertices that is linearly ordered by $\mathbf{edg}_G^*$. An *antichain* in G is a set of pairwise incomparable vertices. We denote of $\mathscr{A}_k$ the set of dags having no antichain of cardinality more than k. By Dilworth's Theorem a dag G is in $\mathscr{A}_k$ if and only if every vertex of G belongs to one of k (not necessarily disjoint) paths $P_1, P_2, \ldots, P_k$ in G. A *chain partition* of $\mathbf{V}_G$ is a partition $(X_1, \ldots, X_k)$ such that each set X_i is a chain. From paths P_i as above, one gets a chain partition by taking $X_1 = P_1$, $X_i = P_i - (P_1 \cup \cdots \cup P_{i-1})$ for $i = 2, 3, \ldots, k$ (we denote here by P_i the set of vertices of the path P_i). Conversely, if a chain partition $X_1, \ldots, X_k$ is given, then for each i one can find a path P_i that contains X_i. Hence G is vertex covered by $P_1 \cup \cdots \cup P_k$, a subgraph of G that is the union of k paths. From $X_1, \ldots, X_k$ we define as follows a linear order on

$\mathbf{V}_G$, denoted by $\alpha(X_1, \ldots, X_k)$:

$$(x, y) \in \alpha(X_1, \ldots, X_k) \text{ if and only if either } x \in X_i,\ y \in X_j \text{ and } i < j \text{ or } x, y \in X_i \text{ and } x\ \mathbf{edg}_G^*\ y \text{ for some } i, j \in \{1, \ldots, k\}.$$

We shall denote by $\mathbf{S}(G, X_1, \ldots, X_k)$ the $\leqslant_\alpha$-minimal topological sorting of G where $\alpha = \alpha(X_1, \ldots, X_k)$ and $(X_1, \ldots, X_k)$ is a chain partition of G.

Theorem 2.4. *For every fixed k, $\mathbf{S}(G, X_1, \ldots, X_k)$ is MS-definable in the structure $\langle \mathbf{V}_G, \mathbf{edg}_G, X_1, \ldots, X_k \rangle$ if G is a dag and $(X_1, \ldots, X_k)$ is a chain partition of $\mathbf{V}_G$.*

We need some notation and lemmas for the proof of this theorem. If S is a linear order on a set V, we denote by $\uparrow S$ the enumeration of V in increasing order for S. Concatenation of sequences is denoted by a big dot $\bullet$. If G is a dag and α is a linear order on a superset of $\mathbf{V}_G$ (i.e., a set containing $\mathbf{V}_G$), we let $\mathbf{S}(G, \alpha)$ denote the $\leqslant_\alpha$-minimal topological sorting. If x is a vertex, we denote by $G - x$ the graph $G[\mathbf{V}_G - \{x\}]$.

Lemma 2.5. *Let G be a dag and α be a linear order on a superset of $\mathbf{V}_G$. Then $\uparrow\mathbf{S}(G, \alpha) = b \bullet \uparrow\mathbf{S}(G - b, \alpha)$ where b is the α-smallest root of G.*

Proof. The sequence $b \bullet \uparrow\mathbf{S}(G - b, \alpha)$ is a topological sorting of G. (To see this, note that $\uparrow\mathbf{S}(G - b, \alpha)$ is a topological sorting of $G - b$; we need only consider edges between b and the other vertices; since b is a root, they all go in the desired direction.) Every topological sorting of G must begin with a root. It follows that $\uparrow\mathbf{S}(G, \alpha)$ begins with b. Let S be the sequence such that $\uparrow\mathbf{S}(G, \alpha) = b \bullet S$. It is a topological sorting of $G - b$. If another one, say S', is smaller than S with respect to $\leqslant_\alpha$ then $b \bullet S'$ is also a topological sorting of G and is strictly smaller than $\uparrow\mathbf{S}(G, \alpha) = b \bullet S$. It is a topological sorting of G and is strictly smaller than $\uparrow\mathbf{S}(G, \alpha)$. Hence S is $\leqslant_\alpha$-minimal and $\uparrow\mathbf{S}(G, \alpha) = b \bullet \uparrow\mathbf{S}(G - b, \alpha)$. □

Our proof of Theorem 2.4 will be an induction on k. The following lemma will give the inductive step.

Lemma 2.6. *Let G be a dag, let (Y_1, Y_2) be a partition of $\mathbf{V}_G$ such that Y_2 is linearly ordered by $\mathbf{edg}_G^*$. We let α_2 be this linear order on Y_2 and α_1 be any linear order on Y_1. Then $\mathbf{S}(G, \alpha_1 \bullet \alpha_2) = \mathbf{S}(G \cup \alpha_1', \alpha_1' \bullet \alpha_2)$ where $\alpha_1' = \mathbf{S}(G^+[Y_1], \alpha_1)$.*

We denote by $\alpha_1 \bullet \alpha_2$ the linear ordering on $\mathbf{V}_G$ such that $(x, y) \in \alpha_1 \bullet \alpha_2$ if and only if either $(x, y) \in \alpha_i$ for some i or $x \in Y_1$, $y \in Y_2$. (Hence $\uparrow\alpha_1 \bullet \alpha_2 = \uparrow\alpha_1 \bullet \uparrow\alpha_2$.) This lemma means that one can construct $\mathbf{S}(G, \alpha_1 \bullet \alpha_2)$ in three steps: one first constructs $\alpha_1' = \mathbf{S}(G^+[Y_1], \alpha_1)$, namely the $\leqslant_{\alpha_1}$-minimal topological sorting of the restriction to Y_1 of the transitive closure of G; then one lets G' consist of G augmented by the edges (x, y), $x \neq y$, such that $(x, y) \in \alpha_1'$: we shall verify that G' is indeed acyclic; and finally one constructs $\mathbf{S}(G', \alpha_1' \bullet \alpha_2)$ giving the desired $\mathbf{S}(G, \alpha_1 \bullet \alpha_2)$.

Proof. We first verify that G' defined above is acyclic. Assume that G' has a cycle. We shall shorten it into a strictly smaller one, and this will give a contradiction. Since $G'[Y_1]$ is acyclic (its edges are in $\mathbf{S}(G^+[Y_1], \alpha_1)$) and $G'[Y_2] = G[Y_2]$ is also acyclic by the hypothesis on Y_2, this cycle cannot be totally within $G'[Y_1]$ or $G'[Y_2]$. It must contain vertices in Y_1 and vertices in Y_2. Hence it can be written as $(x_1, y_1, \ldots, y_n, x_2, x_3, \ldots, x_m, x_1)$ with $x_1, x_2 \in Y_1, y_1, y_2, \ldots, y_n \in Y_2$, $n \geqslant 1$, $x_3, \ldots, x_m \in Y_1 \cup Y_2$. We have $(y_1, y_2), (y_2, y_3), \ldots, (y_{n-1}, y_n) \in \mathbf{edg}_G$. Also we have $(x_1, y_1) \in \mathbf{edg}_G$, $(y_n, x_2) \in \mathbf{edg}_G$, hence $(x_1, x_2) \in \mathbf{edg}_{G'}$ since $G^+[Y_1] \subseteq G'$. Hence $(x_1, x_2, x_3, \ldots, x_m, x_1)$ is a strictly shorter cycle in G'.

Hence $\mathbf{S}(G', \alpha_1' \bullet \alpha_2)$ is well-defined and we need only verify that it equals $\mathbf{S}(G, \alpha_1 \bullet \alpha_2)$. The proof is by induction on $\mathbf{card}(\mathbf{V}_G)$, using Lemma 2.5. Let b be the $(\alpha_1 \bullet \alpha_2)$-smallest root of G. If $b \in Y_2$ then G has no root in Y_1. Then G' has no root in Y_1 either, because $G \subseteq G'$. Hence b is also the smallest $(\alpha_1' \bullet \alpha_2)$-root of G'. If $b \in Y_1$ then b is the α_1-smallest root of $G^+[Y_1]$, hence the first element of α_1'. Hence b is also the $(\alpha_1' \bullet \alpha_2)$-smallest root of G'. Hence $\uparrow \mathbf{S}(G, \alpha_1 \bullet \alpha_2) = b \bullet \uparrow \mathbf{S}(G - b, \alpha_1 \bullet \alpha_2)$ and $\uparrow \mathbf{S}(G', \alpha_1' \bullet \alpha_2) = b \bullet \uparrow \mathbf{S}(G' - b, \alpha_1' \bullet \alpha_2)$ by Lemma 2.5. We know by the induction hypothesis that $\uparrow \mathbf{S}(G - b, \alpha_1 \bullet \alpha_2) = \uparrow \mathbf{S}(G'', \alpha_1'' \bullet \alpha_2)$ where $\alpha_1'' = \mathbf{S}((G - b)^+[Y_1 - \{b\}], \alpha_1)$ and $G'' = (G - b) \cup \alpha_1''$. We need only prove that

$$\uparrow \mathbf{S}(G' - b, \alpha_1' \bullet \alpha_2) = \uparrow \mathbf{S}(G'', \alpha_1'' \bullet \alpha_2). \tag{1}$$

Since b is a root of G we have

$$(G - b)^+[Y_1 - \{b\}] = G^+[Y_1 - \{b\}] = G^+[Y_1] - b$$

because b cannot be an intermediate vertex on any directed path in G^+ connecting two vertices of $G - b$. Hence $\alpha_1'' = \mathbf{S}(G^+[Y_1] - b, \alpha_1)$ and is the restriction of α_1' to $Y_1 - \{b\}$, and we have $G' - b = (G \cup \alpha_1') - b = (G - b) \cup \alpha_1'' = G''$. Equality (1) follows. □

Remark. Given a partition (Y_1, Y_2) of $\mathbf{V}_G$ and two linear orders α_1 on Y_1, and α_2 on Y_2, one cannot compute separately $\alpha_1' = \mathbf{S}(G^+[Y_1], \alpha_1)$ and $\alpha_2' = \mathbf{S}(G^+[Y_2], \alpha_2)$ and simply get $\mathbf{S}(G, \alpha_1 \bullet \alpha_2)$ as $\mathbf{S}(G \cup \alpha_1' \cup \alpha_2', \alpha_1' \bullet \alpha_2')$ because the graph $G \cup \alpha_1' \cup \alpha_2'$ may have cycles. Here is an example. Take $\mathbf{V}_G = Y_1 \cup Y_2$ with $Y_1 = \{x, y\}$, $Y_2 = \{z, t\}$, $\mathbf{edg}_G = \{(y, z), (t, x)\}$, $(x, y) \in \alpha_1$ and $(z, t) \in \alpha_2$. Then $\alpha_1' = \alpha_1$, $\alpha_2' = \alpha_2$ and $G \cup \alpha_1' \cup \alpha_2'$ has the cycle (x, y, z, t, x). However $G' = G \cup \alpha_1'$ has the edges (t, x), (x, y) and (y, z). It is linearly ordered and $\mathbf{S}(G, \alpha_1 \bullet \alpha_2) = G'$.

Proof of Theorem 2.4. We use an induction on k. Let us recall that G is a dag and $(X_1, \ldots, X_k)$ a chain partition of $\mathbf{V}_G$. The case $k = 1$ is trivial. We consider the case $k = 2$.

Claim. *The relation* $\mathbf{S}(G, X_1, X_2)$ *is equal to the relation* S *such that*

$(x, y) \in S$ *if and only if either* $x\, \mathbf{edg}_G^*\, y$ *or* $x \perp_G y$, $x \in X_1$, $y \in X_2$.

Proof of Claim. Clearly S is reflexive. We check that S is symmetric. Let (x,y) and $(y,x) \in S$. We cannot have $x \perp_G y$ because $X_1 \cap X_2 = \emptyset$. Hence $x\,\mathbf{edg}_G^*\,y\,\mathbf{edg}_G^*\,x$ and $x = y$ since G is acyclic.

We now check transitivity. Let (x,y) and $(y,z) \in S$. This happens if $x\,\mathbf{edg}_G^*\,y\,\mathbf{edg}_G^*\,z$ which gives $(x,z) \in S$ as desired. The case where $x \perp_G y \perp_G z$ cannot happen because y cannot be both in X_1 and X_2. Let us consider the case $x\,\mathbf{edg}_G^*\,y \perp_G z$, with $y \in X_1$, $z \in X_2$:

- If $x\,\mathbf{edg}_G^*\,z$ then $(x,z) \in S$ as desired.
- If $z\,\mathbf{edg}_G^*\,x$ then $z\,\mathbf{edg}_G^*\,y$ and this contradicts $y \perp_G z$.
- If $x \perp_G z$ then we cannot have x and z both in the same set X_2 (because we started with a chain partition); hence $x \in X_1$ and $(x,z) \in S$.

The case $x \perp_G y\,\mathbf{edg}_G^*\,z$ is similar.

Then S is a linear order (because incomparable elements must be in different sets X_i) and furthermore a topological sorting of G.

It remains to check that $\mathbf{S}(G, X_1, X_2) = S$. We shall use Lemma 2.5. Let b be the $\alpha(X_1, X_2)$-smallest root of G. Then $\uparrow\mathbf{S}(G, X_1, X_2) = b \bullet \uparrow\mathbf{S}(G - b, X_1', X_2')$ where $X_i' = X_i - \{b\}$. It is easy to check that $(b,x) \in S$ for every $x \in \mathbf{V}_G$. Hence $\uparrow S = b \bullet \uparrow S'$ for some linear order S' on $\mathbf{V}_{G-b}$. We claim that, for $x, y \in \mathbf{V}_G - \{b\}$,

$$(x,y) \in S' \Leftrightarrow \text{either } x\,\mathbf{edg}_{G-b}^*\,y \text{ or } x \perp_{G-b} y \text{ and } x \in X_1, y \in X_2.$$

But this is clear from the fact that b is a root, and we get $\uparrow S' = \uparrow\mathbf{S}(G - b, X_1', X_2')$, whence the result. □

The definition of S is expressible by an MS-formula (because transitive closure is MS-expressible), and this completes the proof of the case $k = 2$. We now consider the case $k > 2$. We let $Y = X_1 \cup \cdots \cup X_{k-1}$. We have, letting β_i be the linear order equal to the restriction of $\mathbf{edg}_G^*$ to X_i,

$$\begin{aligned}\mathbf{S}(G, X_1, \ldots, X_k) &= \mathbf{S}(G, \beta_1 \bullet \beta_2 \bullet \cdots \bullet \beta_k) \quad \text{by definitions}\\ &= \mathbf{S}(G \cup \beta', \beta' \bullet \beta_k) \quad \text{by Lemma 2.6,}\end{aligned}$$

where $\beta = \beta_1 \bullet \beta_2 \bullet \cdots \bullet \beta_{k-1}$ and $\beta' = \mathbf{S}(G^+[Y], \beta)$.

By induction $\mathbf{S}(G^+[Y], \beta)$ is definable by an MS-formula in terms of $X_1, \ldots, X_{k-1}$. By the case $k = 2$, another formula can define $\mathbf{S}(G \cup \beta', \beta' \bullet \beta_k)$, i.e., the desired $\mathbf{S}(G, X_1, \ldots, X_k)$. □

2.3. General graphs

Can one extend Theorem 2.1 to directed graphs with cycles? One cannot of course define a topological sorting but one may want to define a linear order. Let us assume that G is directed and has an *origin*, namely a vertex r from which every vertex is reachable by a directed path; let us also assume that α is a partial order on $\mathbf{V}_G$ that orders linearly each set $\mathbf{Suc}_G(x)$. For every x there is a $\leqslant_\alpha$-smallest path in G from r to

x denoted by $\pi(x)$ and the definition of $\mathbf{F}(G,\alpha)$ extends. The α-depth-first traversal of the tree $\mathbf{F}(G,\alpha)$ is a linear order of G (but not always a topological sorting). However, we do not know how to define it in MS. The reduction $\mathbf{red}(H)$ is not uniquely defined for graphs H with cycles (take for example a complete directed graph with 3 vertices) and we do not know how to define an alternative formula with the same meaning as $\varphi_4(x,y,X)$. Actually, no such formula does exist; otherwise one could express in MS that a directed graph has a Hamiltonian cycle and this is not expressible (see [10, p. 125]).

Open question 2.7. *Is it possible to MS-define a linear order on every locally ordered directed graph having an origin?*

The answers to this question is "yes" if, instead of MS, we use the more powerful language MS_2 where quantified variables can denote sets of edges. One can easily express in MS_2 that a set of edges forms the spanning tree $\mathbf{F}(G,\alpha)$ and the proof continues using this tree and the claim of Theorem 2.1. However, by the results of [10], MS_2-formulas can be translated into equivalent MS-formulas for graphs of bounded degree, and for graphs excluding a fixed graph as a minor (whence in particular for planar graphs and also for graphs of bounded tree-width). Hence the answer to this question is also "yes" for graphs that are restricted in any of these two ways (and by using MS-formulas).

2.4. Recognizability versus MS-definability

We now consider sets of graphs, the recognizable subsets of which are MS-definable. Applications to traces will be given in the next section.

We let $\mathscr{C}_k \subseteq \mathscr{A}_k$ be the class of dags G such that there exists a chain partition $(X_1,\ldots,X_k)$ of $\mathbf{V}_G$ and a topological sorting S such that:

> for each edge (x,y) of G, if $x \in X_i$ then x is the last vertex in X_i that precedes y in the linear order S. (2)

We shall see later that the dependency graphs of traces over an alphabet with k letters are in $\mathscr{C}_k$.

Lemma 2.8. *The class $\mathscr{C}_k$ is MS-definable and MS-formulas can define in every graph G of $\mathscr{C}_k$ a chain partition and a topological sorting witnessing that G is in $\mathscr{C}_k$.*

Proof. An MS-formula $\psi_1(X_1,\ldots,X_k)$ can express that $(X_1,\ldots,X_k)$ is a chain partition of the considered dag. From any chain partition $(X_1,\ldots,X_k)$ we define a graph G' by adding to G an edge (x,y) if and only if:

> there exist $z \in \mathbf{V}_G$ and $i \in [k]$ such that $z,y \in X_i$, $(z,x) \in \mathbf{edg}_G$ and y is the successor of z in $G^+[X_i]$.

Claim. *A topological sorting S of G satisfies condition (2) if and only if it is a topological sorting of G'.*

Proof of Claim. Let S satisfy (2). Let $(x, y), i, z$ be as in the definition of G'. Then (z, x) and (z, y) belong to S. Since z is the S-largest element of X_i before x and $z, y \in X_i$, we must have $(x, y) \in S$. Hence S is a topological sorting of G'.

Let us conversely assume that S is a topological sorting of G'. It is thus one of G. If S does not satisfy (2), we have $(z, x) \in \mathbf{edg}_G$ such that $z\,\mathbf{edg}_{G^+}\,y$ and $(y, x) \in S$ with $z, y \in X_i$, $y \neq x$. Let y' be the successor of z in $G^+[X_i]$. We still have $z\,\mathbf{edg}_{G^+}\,y'\,\mathbf{edg}_{G^+}\,y$ hence $(y', y) \in S$, and also $(y', x) \in S$ since $(y, x) \in S$. Hence there is in G' an edge (x, y'). But S is not a topological sorting of G'. □

The edges of G' that are not in G can be defined by an MS-formula $\psi_2(x, y, X_1, \ldots, X_k)$. We can thus construct an MS-formula $\psi_3(X_1, \ldots, X_k)$ that expresses that G' constructed from $X_1, \ldots, X_k$ is acyclic. If it is, then any topological sorting S of G' is a topological sorting of G satisfying condition (2). By Theorem 2.4, one can define one by an MS-formula $\psi_4(x, y, X_1, \ldots, X_k)$ since G' also belongs to $\mathscr{A}_k$. Hence, the formula $\psi_3(X_1, \ldots, X_k)$ expresses that $(X_1, \ldots, X_k)$ is a chain partition for which a topological sorting satisfying (2) does exist and $\psi_4(x, y, X_1, \ldots, X_k)$ defines such a topological sorting. The class $\mathscr{C}_k$ is defined by the formula: $\exists X_1, \ldots, X_k \psi_3$. □

Example 2.9. We construct graphs showing that the class $\mathscr{A}_2$ is not included in any class $\mathscr{C}_k$. For each $n \in \mathbb{N}$, we let G_n be the dag in $\mathscr{A}_2$ shown in Fig. 1. We shall prove that if $n \geq k + 1$ then $G_n \notin \mathscr{C}_k$.

Let us assume by contradiction that $G = G_n \in \mathscr{C}_k$, $n \geq k + 1$; we let $(X_1, \ldots, X_k)$ be a chain partition of G. Some chain, say X_5, must contain two vertices of $\{y_1, \ldots, y_n\}$, say y_3 and y_8, where y_8 is the successor of y_3 on $G^+[X_5]$. Another one, say X_2, must contain two vertices of $\{t_1, \ldots, t_n\}$, say t_4 and t_{11}, where t_{11} is the successor of t_4 on $G^+[X_2]$. Let us now consider G' as in the proof of Lemma 2.8. It must contain the edges (z_3, y_8) and (x_4, t_{11}), hence it has the cycle $(z_3, y_8, y_9, \ldots, x_4, t_{11}, t_{12}, \ldots, z_1, z_2, z_3)$. A contradiction.

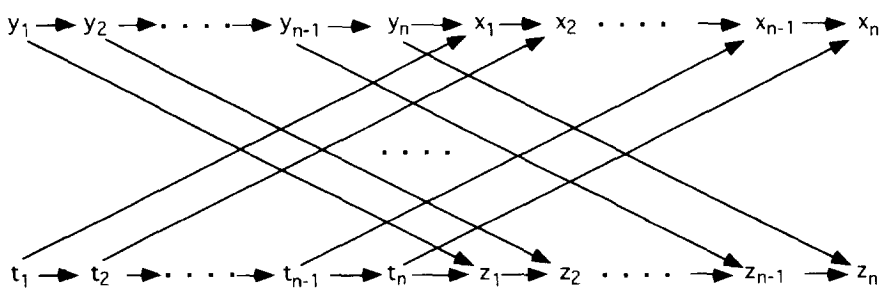

Fig. 1.

Theorem 2.10. *Let $k \in \mathbb{N}$. Every recognizable subset of $\mathscr{C}_k$ is MS-definable.*

Proof. By the methodology described in [7], it is enough to construct a finite signature of graph operations, a definable transduction that associates with every graph in $\mathscr{C}_k$, a finite term over these operations that denotes this graph. By Lemma 2.8, we can assume that a graph G in $\mathscr{C}_k$ is given as a structure containing $(X_1, \ldots, X_k)$ and S witnessing that G belongs to $\mathscr{C}_k$ since these objects can be MS-defined in G. Let $v_1, \ldots, v_m$ be an enumeration of $\mathbf{V}_G$ in increasing order for S. For $j = 1, \ldots, m$, we let $G_j = G[\{v_1, \ldots, v_j\}]$. We let H_j be the sourced graph $\langle G_j, s\rangle$ where s is a source mapping with $s(i) = x$ if and only if x is the S-largest element of $\{v_1, \ldots, v_j\} \cap X_i$. It follows that $\tau(H_j) = \{i \in [k] \mid \{v_1, \ldots, v_j\} \cap X_i \neq \emptyset\}$. Note that $G = \mathbf{fg}_{[k]}(H_m)$, where $\mathbf{fg}_C$ is the graph operation that "forgets" the c-sources for all c in C. (This operation is a special case of the source renaming recalled in Section 1.)

We need only describe unary graph operations $f_2, f_3, \ldots, f_m \in F$ where F is a fixed finite set such that $G = \mathbf{fg}_{[k]}(f_m(f_{m-1}(\ldots f_3(f_2(H_1))\ldots)))$ and such that for every $g \in F$ there is an MS-formula $\theta_g \in \mathscr{L}(\{\mathbf{edg}, \leqslant\}, \{X_1, \ldots, X_k, x\})$ such that

$$(G, S, X_1, \ldots, X_k, x) \models \theta_g \text{ if and only if } g = f_j \text{ where } j \in \{2, \ldots, m\} \text{ is such that } x = v_j.$$

For $i, i' \in [k]$, with $i \neq i'$, we denote by $\mathbf{e}(i, i')$ the graph consisting of one directed edge linking the i-source to the i'-source. We now define f_j such that $H_j = f_j(H_{j-1})$. There are two cases.

Case 1: $\tau(H_j) = \tau(H_{j-1}) \cup \{p\}$. We let x be the unique vertex in H_j that is not in H_{j-1}. It is the p-source of H_j. We let $\{i_1, \ldots, i_q\} \subseteq \tau(H_{j-1})$ be the set of integers i_r such that there is in H_j an edge from the i_r-source of H_{j-1} (which is also the i_r-source of H_j) to the vertex x. We can take for f_j the mapping

$$f_j(u) := \mathbf{e}\{i_1, p) \| \mathbf{e}(i_2, p) \| \cdots \| \mathbf{e}(i_q, p) \| u,$$

where $\|$ is the parallel composition defined in Section 1 and u is a variable denoting graphs of type $\tau(H_{j-1})$. Hence f_j is of type $\tau(H_{j-1}) \to \tau(H_j)$, and we have $H_j = f_j(H_{j-1})$. Note that f_j can be determined by an MS-formula from x, the order S (from which we get the set of vertices before x) and the sets $X_1, \ldots, X_k$.

Case 2: $\tau(H_j) = \tau(H_{j-1})$. The vertex x (added to H_{j-1} to form H_j) is the p-source of H_j: so the p-source of H_{j-1} is internal in H_j. We let $\{i_1, \ldots, i_q\}$ be as above and we let

$$f_j(u) = \mathbf{ren}_{k+1 \to p}(\mathbf{fg}_{\{p\}}(\mathbf{e}(i_1, k+1) \| \cdots \| \mathbf{e}(i_q, k+1) \| u)).$$

Here f_j is of type $\tau(H_{j-1}) \to \tau(H_{j-1})\,(= \tau(H_j))$ and again $H_j = f_j(H_{j-1})$. (We denote by $\mathbf{ren}_{r \to p}$ the source renaming operation (see Section 1) that makes the r-source into the p-source.) Again f_j can be determined from $x, X_1, \ldots, X_k$ and S by an MS-formula. It follows that the expression $\mathbf{fg}_{[k]}(f_m(f_{m-1}(\ldots(f_2(H_1))\ldots)))$ denotes G and can be constructed as the result of a definable transduction. This completes the proof. □

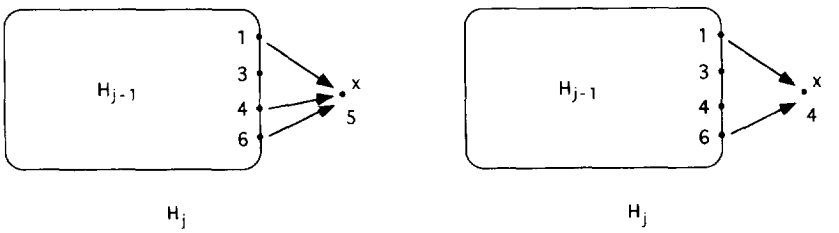

Fig. 2.

It follows from this construction that graphs in $\mathscr{C}_k$ have path-width at most k. We shall use this fact shortly.

Example 2.11. We illustrate the two cases of the construction of f_j.

Case 1: See the left part of Fig. 2. We have $\tau(H_{j-1}) = \{1,3,4,6\}$, $\tau(H_j) = \tau(H_{j-1}) \cup \{5\}$, $f_j(u) = \mathbf{e}(1,5) \parallel \mathbf{e}(4,5) \parallel \mathbf{e}(6,5) \parallel u$.

Case 2: See the right part of Fig. 2. We have $\tau(H_{j-1}) = \tau(H_j) = \{1,3,4,6\}$. $f_j(u) = \mathbf{ren}_{7\to 4}(\mathbf{fg}_{\{4\}}(\mathbf{e}(1,7) \parallel \mathbf{e}(6,7) \parallel u))$. The absence of an edge between the 4-source of H_{j-1} and that of H_j does not contradict the hypothesis that X_4 should be linearly ordered under $\mathbf{edg}^*_G$: we can have in H_{j-1} an edge from the 4-source to the 6-source.

We let $\mathbf{und}(G)$ denote the simple undirected graph obtained from G by letting $\mathbf{edg}_{\mathbf{und}(G)} = \mathbf{edg}_G \cup (\mathbf{edg}_G)^{-1}$.

Corollary 2.12. *Let $k \in \mathbb{N}$. A subset of $\mathscr{C}_k$ is recognizable if and only if it is MS-definable. The same holds for a subset of* $\mathbf{und}(\mathscr{C}_k)$.

Proof. The "if" directions of the two assertions follow from the main theorem of [6]. The "only if" direction of the first assertion is Theorem 2.10. We now consider the "only if" direction of the second assertion. Let L be a recognizable subset of $\mathbf{und}(\mathscr{C}_k)$. Since the mapping **und** is a homomorphism for the operations of parallel composition and source renaming, it follows from Proposition 1.2 that $\mathbf{und}^{-1}(L)$ is a recognizable set of graphs; the set $\mathscr{C}_k$ is recognizable since it is MS-definable (Lemma 2.8) hence $K = \mathbf{und}^{-1}(L) \cap \mathscr{C}_k$ is recognizable, and K is MS-definable by Theorem 2.10. By Theorem 3.11 of [9], $L = \mathbf{und}(K)$ is MS_2-definable. We have observed that the graphs in $\mathscr{C}_k$, hence also those in $\mathbf{und}(\mathscr{C}_k)$, have path-width at most k. By the results of [10], an MS_2-definable set of graphs of path-width at most k is MS-definable. Hence, L is MS-definable. □

2.5. *Partial k-paths*

We conclude with an application to a "classical" class of undirected graphs. A *partial k-path* [21] is a simple undirected graph G such that, for some n, we have

$\mathbf{V}_G = V_1 \cup V_2 \cup \cdots \cup V_n$ and the following holds:

(1) the two ends of every edge belong to some set V_i,

(2) a vertex in $V_i \cap V_j$ belongs to V_ℓ for every ℓ between i and j,

(3) $\mathbf{card}(V_i) = k + 1$ for every $i = 1, \ldots, n$,

(4) $\mathbf{card}(V_i \cap V_{i+1}) = k$ for every $i = 1, \ldots, n-1$,

(5) $V_i \cap V_{i+1} \neq V_{i+1} \cap V_{i+2}$ for $i = 1, \ldots, n-2$.

A graph is *k-connected* if it has at least $k + 2$ vertices and there is no set of vertices with at most $k - 1$ vertices, the deletion of which disconnects the graph.

The graph operations upon which recognizability is defined also apply to undirected graphs. Hence, the notion of a recognizable set of undirected graphs is well-defined.

Corollary 2.13. *Let $k \in \mathbb{N}$. Every k-connected partial k-path belongs to* $\mathbf{und}(\mathscr{C}_k)$. *A set of k-connected partial k-paths is recognizable if and only if it is MS-definable.*

Proof. For every k-connected partial k-path G we shall define an orientation G', a topological sorting S and a partition $(X_1, \ldots, X_k)$ of $\mathbf{V}_G$ making G' into a member of $\mathscr{C}_k$. We let $(V_1, \ldots, V_n)$ be as in the definition of a partial k-path; since G is k-connected, n is at least 2.

We first define S. Its first element is the unique vertex in $V_1 - V_2$. The next k elements are those of $V_1 \cap V_2$ in any order. Then comes the unique element of $V_2 - V_1$, then that of $V_3 - V_2$, etc., the last one being the unique element of $V_n - V_{n-1}$.

There is a unique orientation G' of G for which S is a topological sorting. We now define $X_1, \ldots, X_k$. We put in X_1 the unique vertex in $V_1 - V_2$. We put each of the k vertices of $V_1 \cap V_2$ in exactly one of the sets $X_1, \ldots, X_k$, and we put in X_1 the S-largest of these vertices. We consider then, in turn, the vertices in $V_i - V_{i-1}$ for $i = 2, 3, \ldots, n-1$. Let x_i be the unique vertex in $V_i - V_{i-1}$. By condition (5) of the definition of a partial k-path, some vertex y in $V_i \cap V_{i-1}$ does not belong to $V_i \cap V_{i+1}$. Assume that this y has been put in X_j, then we also put x_i in X_j. The last vertex (the unique one in $V_n - V_{n-1}$) is put in any of the sets $X_1, \ldots, X_k$. It remains to check that $(X_1, \ldots, X_k)$ is a chain partition of G and that condition (2) of the definition of $\mathscr{C}_k$ holds.

The vertex x in $V_1 - V_2$ is linked to all vertices of $V_1 \cap V_2$, otherwise G would not be k-connected (a proper subset of $V_1 \cap V_2$ would separate x from the vertex in $V_2 - V_1$). Hence, the first two vertices of X_1 are linked by an edge. Consider now the step in the above algorithm where we put x_i in X_j. We claim that (y, x_i) is an edge of G. If this is not the case then $V_i \cap V_{i-1} - \{y\}$ would separate x_i from y, contradicting the k-connectedness. Finally, the last vertex is the target of edges having as origins each of the k vertices in $V_{n-1} - V_{n-2}$. Hence, each set X_i induces a path in G'.

Finally, one can observe that every vertex x_i, $i = 2, \ldots, n$, has incoming edges from $V_i \cap V_{i-1}$ only, and that each set $V_i \cap V_{i-1}$ contains one and only one vertex of each set $X_1, \ldots, X_k$. This ensures that for every edge (x, y) the vertex x is the last one before y in the same set X_j. Hence, G' is in $\mathscr{C}_k$.

The second assertion follows from Corollary 2.12. □

It is conjectured in [7] that a set of graphs of tree-width at most k for any fixed k is recognizable if and only if it is CMS-definable. This result is a further step towards the proof of this conjecture since partial k-path have tree-width at most k.

Example 2.14. We show in Fig. 3 an example of the construction of G' where $k = 3$.

In this example we have $n = 6$. The boxes show the sets $V_1, \ldots, V_6$. The vertices are numbered from 1 to 9 in the order of S. The vertices of X_1 are dots, those of X_2 are hearts and those of X_3 are diamonds. Let us review the construction of X_1, X_2 and X_3. The vertex 1 is placed in X_1, 2, 3, 4 are placed respectively in X_2, X_3 and X_1. Then 5 is the unique vertex in $V_2 - V_1$, and 4, the vertex in $V_2 - V_3$, belongs to X_1. Hence 5 is placed in X_1. Then 6 is the unique vertex in $V_3 - V_2$, and 2, the vertex in $V_3 - V_4$, belongs to X_2. Hence 6 is placed in X_2. We continue in this way. Finally, we redraw the graph. The vertices in a same set X_i are on the same horizontal line. See Fig. 4.

The classes of undirected graphs we have considered can be compared as follows:

k-connected partial k-path $\subseteq$ **und**$(\mathscr{C}_k) \subseteq$ **und**$(\mathscr{A}_k)$,

und$(\mathscr{C}_k) \subseteq$ partial k-paths,

and the class of partial k-paths is incomparable with **und**$(\mathscr{A}_k)$.

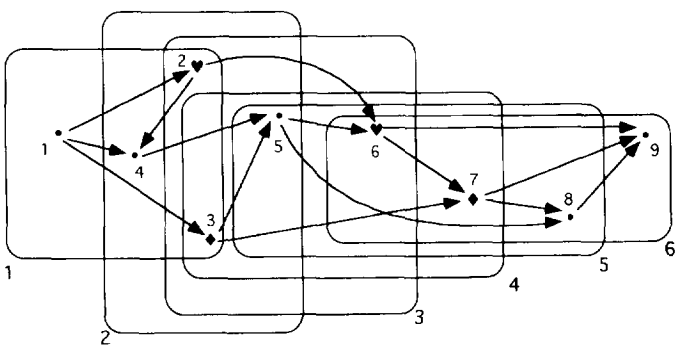

Fig. 3.

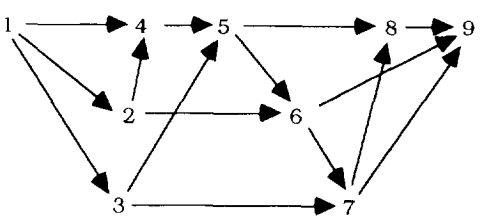

Fig. 4.

3. Traces

As an application of Theorem 2.4, we give a new proof of a result of Ochmanski [19] on recognizable sets of partially commutative words, also called *traces*. We recall a few definitions and we refer the reader to [1] for more details.

A *partially commutative alphabet* is a pair (A, C) where A is a finite alphabet and C a set of unordered pairs of letters of A that are said to commute. We let $\equiv$ denote the least congruence on A^* such that $ab \equiv ba$ for every $\{a,b\} \in C$. An element of $A^*/\equiv$ is called a *trace*. The quotient monoid $\mathbf{M}(A,C) := A^*/\equiv$ is called the *trace monoid defined by* (A,C).

If $L \subseteq A^*/\equiv$, we let $\langle L \rangle \subseteq A^*$ be the union of the sets t, for $t \in L$. (Each trace t is an equivalence class, hence a subset of A^*.)

The notion of a recognizable subset of $\mathbf{M}(A,C)$ follows immediately from the monoid structure. However, we shall use the following more concrete characterization, that will be our definition:

$L \subseteq A^*/\equiv$ is recognizable if and only if $\langle L \rangle$ is a regular language (i.e., a recognizable subset of the free monoid A^*).

Let us enumerate A in a fixed way as $\{a_1, \ldots, a_k\}$. Every trace t contains a unique $\leqslant$-minimal element (where $\leqslant$ is the lexicographic order associated with the chosen enumeration of A) that we shall denote by $\mathbf{min}(t)$. Ochmanski has proved that a set $L \subseteq \mathbf{M}(A,C)$ is recognizable if and only if $\mathbf{Min}(L)$ (defined as $\{\mathbf{min}(t) \mid t \in L\}$) is a regular language. We shall give a new proof of this result, based on graphs and monadic second-order logic.

Every trace can be represented by a dag with vertices labelled by the letters, called its *dependency graph*. We fix $A = \{a_1, \ldots, a_k\}$ and C. With every word $u = b_1 b_2 \cdots b_n \in A^*$ (where $b_1, \ldots, b_n \in A$) we associate a graph G constructed as follows:

- $\mathbf{V}_G = \{1, 2, \ldots, n\}$ (if $n = 0$ then G is empty),
- vertex i has the label b_i,
- there is a directed edge from i to j if and only if $i < j$ and $\{b_i, b_j\} \notin C$ (this is the case in particular if $b_i = b_j$).

Finally, we let H be the reduction of G; it will be denoted by $\mathbf{dep}(u)$. It will be handled as a relational structure $\langle \mathbf{V}_H, \mathbf{edg}_H, (\mathbf{lab}_{aH})_{a \in H} \rangle$ where $\mathbf{lab}_{aH}(x)$ holds if and only if x has label a.

Proposition 3.1 (Aalbersberg and Rozenberg [1]). *For any two words $u, v \in A^*$, $u \equiv v$ if and only if $\mathbf{dep}(u)$ and $\mathbf{dep}(v)$ are isomorphic, if and only if v is a topological sorting of $\mathbf{dep}(u)$.*

It follows that the graph H as above is actually associated with the equivalence class of u, i.e., with a trace t. We shall denote by $\mathbf{dep}(t)$ the abstract graph that is the isomorphism class of $\mathbf{dep}(u)$ where u is any member of t. (The numbering of the vertices of $\mathbf{dep}(u)$ depends on u but is irrelevant in $\mathbf{dep}(t)$.) If L is a set of traces, we let $\mathbf{Dep}(L) := \{\mathbf{dep}(t) \mid t \in L\}$.

Lemma 3.2. *The mapping* **dep** *from words to graphs is a definable transduction.*

Proof. This is immediate from the definition and the fact that reduction is a definable transduction. □

Proposition 3.3. *Let* (A, C) *be a partially commutative alphabet.*

(1) *Every dependency graph satisfies the following properties: it is acyclic, reduced, any two adjacent vertices are labelled by noncommuting letters and any two vertices labelled by noncommuting letters (in particular any two vertices labelled by the same letter) are comparable.*

(2) *Every directed graph with vertices labelled in A that satisfies the above conditions is a dependency graph.*

In particular every dependency graph G belongs to the class of dags $\mathscr{C}_k$ defined in Section 2 where k is the size of the alphabet: the sets of vertices with a same label form an appropriate chain partition and the topological sorting S is the natural order on the letter position of a word with dependency graph G. For such a graph G, we let

$$\begin{aligned}\mathbf{min}(G) &:= \mathbf{min}(\mathbf{dep}^{-1}(G))\\ &:= \text{the unique } \leqslant\text{-minimal word in the trace } \mathbf{dep}^{-1}(G) \subseteq A^*.\end{aligned}$$

Proposition 3.4. *The mapping* **min** *is a definable transduction from dependency graphs to words.*

Proof. Let $A = \{a_1, \ldots, a_k\}$. Let G be a dependency graph. Let $X_i = \{x \in \mathbf{V}_G \mid \mathbf{lab}_G(x) = a_i\}$. Then $\mathbf{V}_G = X_1 \cup \cdots \cup X_k$ where $(X_1, \ldots, X_k)$ is a chain partition. The word $\mathbf{min}(G)$ is nothing but the reduction of the linear graph $\mathbf{S}(G, X_1, \ldots, X_k)$, hence **min** is definable since it is the composition of two definable transductions: the one defining $\mathbf{S}(G, X_1, \ldots, X_k)$ (by Theorem 2.4) and the reduction. □

Theorem 3.5. *Let* (A, C) *be a partially commutative alphabet. The following properties of a subset* X *of* A^* *are equivalent:*

(1) $\langle X \rangle$ *is a regular language,*

(2) $\mathbf{Min}(X)$ *is a regular language,*

(3) $\mathbf{Dep}(X)$ *is an MS-definable set of graphs,*

(4) $\mathbf{Dep}(X)$ *is a recognizable set of graphs.*

Before proving the result, we make a few observations. Since one can express in MS (by Proposition 3.3) that a directed vertex-labelled graph is a dependency graph, it is equivalent to saying that a set of dependency graphs is MS-definable *among dependency graphs* (i.e., is the set of dependency graphs satisfying an MS-formula) or is MS-definable *among all graphs*. Hence (3) can be read in two equivalent ways. Also,

since traces are graphs of bounded path-width, the language MS_2 is no more powerful than MS in order to express their properties (by [10]). Hence, (3) can be read in total in four different equivalent ways.

Proof. (1) $\Rightarrow$ (2): By Proposition 3.4, one can construct an MS-formula $\theta(x, y)$ such that, for every word $u \in A^*$, this formula defines in the relational structure representing u a linear order that corresponds to $\mathbf{min}(u)$. One can build a closed MS-formula θ' that verifies that this order coincides with the natural order on the letters of u, i.e., that $\mathbf{min}(u) = u$. The set $\mathbf{Min}(A^*)$ of minimal words is MS-definable, hence regular, by Büchi's Theorem which says that a language is regular if and only if it is MS-definable (see [24]). Since $\mathbf{Min}(X) = \langle X \rangle \cap \mathbf{Min}(A^*)$, we get the desired implication.

(2) $\Rightarrow$ (3): The transduction **min** from dependency graphs to minimal words is definable. Since $\mathbf{Dep}(X) = \mathbf{min}^{-1}(\mathbf{Min}(X))$ we obtain that $\mathbf{Dep}(X)$ is MS-definable if $\mathbf{Min}(X)$ is, which is the case by Büchi's Theorem since $\mathbf{Min}(X)$ is assumed to be regular.

(3) $\Rightarrow$ (1): The transduction **dep** that maps a word $u \in A^*$ to the corresponding dependency graph is definable (see Lemma 3.2). Note also that $\langle X \rangle = \mathbf{dep}^{-1}(\mathbf{Dep}(X))$. Hence, so is the language $\mathbf{dep}^{-1}(\mathbf{Dep}(X))$ since $\mathbf{Dep}(X)$ is MS-definable. The language $\langle X \rangle$ is thus regular by Büchi's Theorem.

(3) $\Rightarrow$ (4): is a consequence of the result by Courcelle [6] saying that every MS-definable set of graphs is recognizable.

(4) $\Rightarrow$ (3): We have observed that dependency graphs belong to the class $\mathscr{C}_k$ where $k = \mathbf{card}(A)$. The result follows from Theorem 2.10 saying that recognizable subsets of $\mathscr{C}_k$ are MS-definable. (The proof was done for unlabelled graphs, but it is straightforward to adapt it to labelled graphs.) □

The equivalence of (1) and (2) in this theorem is also proved in [19] by a more complicated method using rational expressions for representing the two considered languages. The equivalence of (1) and (3) in this theorem is also proved by Thomas [25], by using the asynchronous cellular automata [3] and the difficult result stating that these automata define exactly the recognizable sets of traces; this equivalence is also proved for certain infinite traces (whence for traces as a subcase) by Ebinger and Muscholl [15] by means of rational expressions. Our proof does not use such complex tools: it uses only MS-logic, and regular languages are handled through MS-logic by Büchi's Theorem.

4. Order-invariant MS-definable graph properties

In this section, we introduce an extension of MS-logic that is based on the use of auxiliary linear orderings. This extension, denoted by $MS(\leqslant)$ (read as *MS logic with linear order*) subsumes CMS. It is useful only in structures where *no linear ordering is MS-definable*, because whenever a linear order is definable its expressive power is no

larger than that of MS. The main result of this section states that the MS($\leqslant$)-definable sets of graphs are recognizable. MS($\leqslant$)-logic is thus a good candidate for the logical characterizations of recognizability we are looking for. In Section 5, it will prove useful for characterizing recognizable sets of cographs. Results are stated and proofs are done in terms of relational structures. They are thus applicable to graphs of several types and also to hypergraphs.

4.1. Order-invariant properties of ordered graphs and relational structures

An *ordered graph* is a pair consisting of a graph G and a linear order P of its set of vertices. Note that G may be ordered itself in a natural way, for instance if it is a word: however, P can be *any* linear order of $\mathbf{V}_G$. If $\mathcal{G}$ is a class of graphs, we let $\mathcal{G}(\leqslant)$ denote the class of all ordered graphs of the form $\langle G, P\rangle$ for $G \in \mathcal{G}$. A property φ of ordered graphs is *order-invariant* if, for every $G \in \mathcal{G}$, for any two linear orders P and P' on $\mathbf{V}_G$:

$$\varphi(\langle G, P\rangle) \Leftrightarrow \varphi(\langle G, P'\rangle).$$

Graphs are defined as $\{\mathbf{edg}\}$-structures. We let $\leqslant$ be a new binary relation symbol. A property φ of graphs of a class $\mathcal{G}$ is MS($\leqslant$)-*expressible* if and only if it is an order-invariant MS-property of the graphs in $\mathcal{G}(\leqslant)$, hence, if and only if there exists a formula φ in $\mathcal{L}(\{\mathbf{edg}, \leqslant\})$ such that

(1) for every $G \in \mathcal{G}$, for any two linear orders P and P' on $\mathbf{V}_G$:
$\langle G, P\rangle \models \varphi$ if and only if $\langle G, P'\rangle \models \varphi$ (where P and P' are values of $\leqslant$);

(2) for every $G \in \mathbf{G}$:
$\varphi(G)$ holds if and only if $\langle G, P\rangle \models \varphi$ for some linear order P on $\mathbf{V}_G$ (equivalently, if and only if $\langle G, P\rangle \models \varphi$ for all linear orders P on $\mathbf{V}_G$).

A set of graphs is MS($\leqslant$)-*definable* if and only if it is the set of graphs satisfying an MS($\leqslant$)-expressible property.

As an example, we consider the property of a graph G: "$\mathbf{card}(\mathbf{V}_G)$ is even". This property is not MS-expressible if $\mathcal{G}$ is the class of discrete graphs (see [6]). However it is MS($\leqslant$)-expressible. The appropriate formula in $\mathcal{L}(\{\leqslant\})$ is the one that says the following:

there exist two sets X_0, X_1 that form a partition of the set of vertices and such that the $\leqslant$-smallest element is in X_0, the $\leqslant$-successor of every element of X_0 is in X_1, the $\leqslant$-successor of every element of X_1 is in X_0, and the $\leqslant$-largest element is in X_1.

This formula holds if and only if $\mathbf{card}(\mathbf{V}_G)$ is even and this property does not depend on the considered linear order. More generally, every condition of the form "$\mathbf{card}(X) \equiv 0$ modulo p" (for fixed p) is an MS($\leqslant$)-property. It follows that every CMS-definable property is MS($\leqslant$)-definable. (We recall that CMS refers to *Counting Monadic Second-order* logic, i.e., to an extension of MS in which one can say that a set has cardinality $\equiv 0$ modulo p, for any fixed p.) We have the following hierarchy of families of sets of graphs:

MS-definable $\subseteq$ CMS-definable $\subseteq$ MS($\leqslant$)-definable $\subseteq$ Recognizable.

where the last inclusion will be proved in Theorem 4.1 in the more general case of relational structures. The first and third inclusion are strict in general but we shall consider classes of graphs for which they are equalities. Whether the second one is strict is an open problem. For classes of graphs having an MS-definable ordering, like the classes $\mathscr{A}_k$ of Section 2, the first two inclusions are equalities. For the class of discrete graphs or the class of trees, the first inequality is strict and the last two are equalities. A table in Section 7 will summarize the known results regarding these comparisons.

We shall compare MS($\leqslant$)-definability and recognizability in the general framework of [8], which deals with slightly more general relational structures than those used up to now. Let R be a finite set of relation symbols as before. Let C be a finite set of nullary symbols called *constants*. An (R, C)-*structure* is an object of the form $S = \langle \mathbf{D}_S, (r_S)_{r\in R}, (c_S)_{c\in C}\rangle$ where $\langle \mathbf{D}_S, (r_S)_{r\in R}\rangle$ is an R-structure, and c_S belongs to $\mathbf{D}_S$ for each c in C. (One may have $c_S = c'_S$ with $c \neq c'$.) We denote by $\mathscr{S}(R, C)$ the set of (R, C)-structures. The nullary symbols are convenient to represent the sources of graphs: each source label c is turned into a constant and its value in the structure representing a graph is its c-source. We denote by $\mathscr{L}(R, C, W)$ the set of MS-formulas written with R, the nullary symbols in C, and having their free variables in W: they are constructed from atomic formulas of the forms $v_1 = v_2$, $v_1 \in X$ and $r(v_1, \ldots, v_n)$ where each v_i is an object variable or a constant in C and X is a set variable.

By an *ordered structure*, we mean a structure equipped with a linear order on its domain, usually denoted by the binary symbol $\leqslant$. The notions of an *order-invariant* MS-property of structures and of an MS($\leqslant$)-*definable* set of (R, C)-structures can be defined similarly as for graphs.

The language MS($\leqslant$) is somewhat ineffective because it is not possible to decide whether a given MS-formula defines an order-invariant property or not. However, we shall write formulas that will define, by construction, order-invariant properties, as is the case above for the formula expressing parity.

In order to prove this undecidability result we consider, for a Turing machine M given with some initial configuration, the set $L(M)$ consisting of the graphs that either are complete (i.e., that have one edge from any vertex to any other one) or are a square grid on which a terminating computation of M can be encoded (in the usual way: the configurations of the computation are encoded on the successive lines of the grid; we assume that the edges of these grids are directed and that there is an edge from x to y for each edge from y to x). The set $L(M)$ is MS-definable and a defining MS-formula ψ_M can be constructed from M. Let us now consider the property of an ordered graph saying that the sequence of its vertices listed in increasing order is a Hamiltonian path. This property is expressible by a formula θ in $\mathscr{L}(\{\mathbf{edg}, \leqslant\})$. Now the property defined by the formula $\psi_M \wedge \theta$ is satisfied by infinitely many graphs (in particular by every complete graph, whichever linear ordering is given on it); it is thus nontrivial; it is order-invariant if and only if M does not halt (because the validity of θ on an ordered square grid depends on the linear order). Hence, the order-invariance of an MS-property of ordered graphs is undecidable.

In order to define the notion of a recognizable set of (R, C)-structures, we define some operations on structures. We shall distinguish the *concrete* structures belonging to the sets $\mathscr{S}(R, C)$ from the *abstract* structures, i.e., the isomorphism classes of concrete structures which form the corresponding sets $\mathbf{S}(R, C)$.

4.2. The parallel composition of structures

Let $S \in \mathscr{S}(R, C)$ and $S' \in \mathscr{S}(R', C')$ be disjoint structures, i.e., structures such that $\mathbf{D}_S \cap \mathbf{D}_{S'} = \emptyset$. We let $S \| S'$ be the structure T defined as follows. We let $D := \mathbf{D}_S \cup \mathbf{D}_{S'}$, we let $\sim$ be the least equivalence relation on D such that $c_S \sim c_{S'}$ for every $c \in C \cap C'$. We let $\mathbf{D}_T := D/\sim$ and we denote by $[d]$ the equivalence class of an element d in D. We also let:

$c_T := [c_S]$ if $c \in C$,

$c_T := [c_{S'}]$ if $c \in C'$ (we have $[c_S] = [c_{S'}]$ if $c \in C \cap C'$),

$r_T([d_1], \ldots, [d_n])$ holds if $r_S(d'_1, \ldots, d'_n)$ holds or if

$r_{S'}(d'_1, \ldots, d'_n)$ holds for some $d'_1 \in [d_1], \ldots, d'_n \in [d_n]$.

Note that $S \| S' \in \mathscr{S}(R \cup R', C \cup C')$. If $C \cap C' = \emptyset$, then $\mathbf{D}_{S \| S'} = \mathbf{D}_S \cup \mathbf{D}_{S'}$ and $S \| S'$ is the union of S and S'. If $S \in \mathbf{S}(R, C)$ and $S' \in \mathbf{S}(R', C')$, then $S \| S'$ is the isomorphism class of $T \| T'$ where T and T' are disjoint and respectively isomorphic to S and S'. Clearly, $\|$ is a total operation on abstract structures. This operation extends the parallel composition of sourced graphs: if G is a sourced graph of type C considered as a structure $G = \langle \mathbf{V}_G, \mathbf{edg}_G, (c_G)_{c \in C} \rangle$ where c_G is the c-source of G, if G' is a sourced graph of type C' defined similarly as an $(\{\mathbf{edg}\}, C')$-structure, then the structure $G \| G'$ represents the abstract graph $G \|_{C,C'} G'$.

4.3. Quantifier-free definable operations

We denote by $QF(R, C, \mathscr{X})$ the set of quantifier-free formulas written with R, C, and the variables of $\mathscr{X}$ (where $\mathscr{X}$ is a set object variables). Our purpose is to specify by quantifier-free formulas total mappings: $\mathscr{S}(R, C) \to \mathscr{S}(Q, D)$. We let Δ be a tuple of the form $\langle \delta, (\theta_q)_{q \in Q}, (\tau_d)_{d \in D} \rangle$ such that:

- $\delta \in QF(R, C, \{x_1\})$ and is of the form

$$\delta' \vee \bigvee_{d \in D} \bigvee \{x_1 = \tau_d\}$$

for some formula δ' in $QF(R, C, \{x_1\})$,

- $\theta_q \in QF(R, C, \{x_1, \ldots, x_n\})$ where $n = \rho(q)$,
- $\tau_d \in C$ for each $d \in D$.

With every such tuple of formulas Δ, we associate the total mapping $\mathbf{def}_\Delta : \mathscr{S}(R, C) \to \mathscr{S}(Q, D)$ such that, for every S in $\mathscr{S}(R, C)$, $T = \mathbf{def}_\Delta(S)$ is the structure in $\mathscr{S}(Q, D)$ defined as follows:

$\mathbf{D}_T := \{x \in \mathbf{D}_S \mid S \models \delta(x)\}$,

$d_T := (\tau_d)_S$ for each $d \in D$,

$q_T(x_1, \ldots, x_n)$ holds if and only if $x_1, \ldots, x_n \in \mathbf{D}_S$ and

$$S \models \delta(x_1) \wedge \cdots \wedge \delta(x_n) \wedge \theta_q(x_1, \ldots, x_n)$$

(i.e., $x_1, \ldots, x_n \in \mathbf{D}_T$ and $S \models \theta_q(x_1, \ldots, x_n)$).

The domain of T is the set of elements of $\mathbf{D}_S$ that satisfy δ: this formula has been taken of such a form that the elements $(\tau_d)_S$, $d \in D$, that are needed as values of d in T, are indeed in the domain of T. Each formula θ_q specifies q_T in terms of the constants and relations of S. A mapping: $\mathscr{S}(R, C) \to \mathscr{S}(Q, D)$ is *quantifier-free definable* (*qfd*) if and only if it is of the form $\mathbf{def}_\Delta$ for some Δ as above. The extension into a mapping: $\mathbf{S}(R, C) \to \mathbf{S}(Q, D)$ also denoted by $\mathbf{def}_\Delta$ is straightforward. Note that these operations are definable transductions of a special form.

A set of abstract (R, C)-structures is *recognizable* if and only if it is with respect to the magma $\mathbf{S}$ with domains $\mathbf{S}(R, C)$ for all pairs (R, C) and equipped with the operations $\|$ which map $\mathbf{S}(R, C) \times \mathbf{S}(R', C')$ into $\mathbf{S}(R \cup R', C \cup C')$ and the qfd operations $\mathbf{def}_\Delta : \mathbf{S}(R, C) \to \mathbf{S}(Q, D)$. (Let us recall that $\|$ is an overloaded symbol denoting infinitely many operations.) A set of (R, C)-structures is *recognizable* if and only if the set of its isomorphism classes is recognizable.

Now we consider the relation between MS($\leqslant$)-definability and recognizability. It was proved in [8] that every MS-definable set of (R, C)-structures is recognizable. We extend this result to MS($\leqslant$)-definable sets.

Theorem 4.1. *Let R be a finite set of relation symbols, and C a finite set of constants. Every MS($\leqslant$)-definable set of (R, C)-structures is recognizable.*

Proof. We shall denote by $\mathbf{S}_\leqslant(R, C)$ (a subset of $\mathbf{S}(R \cup \{\leqslant\}, C)$) the set of abstract ordered (R, C)-structures. We let $\mathbf{uno} : \mathbf{S}_\leqslant(R, C) \to \mathbf{S}(R, C)$ be the mapping that "forgets the order" (read "**uno**rder"). The recognizability of a set L of abstract (R, C)-structures is defined with respect to the parallel composition $\|$ and to the qfd operations. For each of these operations, we define an "ordered" version, say $\|_\leqslant : \mathbf{S}_\leqslant(R, C) \times \mathbf{S}_\leqslant(R', C') \to \mathbf{S}_\leqslant(R \cup R', C \cup C')$ or $\mathbf{def}_{\Delta\leqslant} : \mathbf{S}_\leqslant(R, C) \to \mathbf{S}_\leqslant(Q, D)$, such that **uno** behaves homomorphically, namely such that:

$$\mathbf{uno}(S \|_\leqslant S') = \mathbf{uno}(S) \| \mathbf{uno}(S'), \quad \mathbf{uno}(\mathbf{def}_{\Delta\leqslant}(S)) = \mathbf{def}_\Delta(\mathbf{uno}(S)).$$

We begin with $\mathbf{def}_{\Delta\leqslant} : \mathbf{S}_\leqslant(R, C) \to \mathbf{S}_\leqslant(Q, D)$ where $\Delta = \langle \psi, (\theta_r)_{r \in Q}, (\tau_d)_{d \in D} \rangle$. We let $\Delta_\leqslant$ consist of Δ augmented with the formula $\theta_\leqslant$ defined as $x_1 \leqslant x_2$. Then $\mathbf{def}_{\Delta\leqslant}$ is a qfd operation: $\mathbf{S}(R \cup \{\leqslant\}, C) \to \mathbf{S}(Q \cup \{\leqslant\}, D)$ which maps $\mathbf{S}_\leqslant(R, C)$ into $\mathbf{S}_\leqslant(Q, D)$ because, if $T = \mathbf{def}_{\Delta\leqslant}(S)$, then $\leqslant_T$ is the restriction of the order $\leqslant_S$ to the domain $\mathbf{D}_T$. It is clear that $\mathbf{uno}(\mathbf{def}_{\Delta\leqslant}(S)) = \mathbf{def}_\Delta(\mathbf{uno}(S))$.

The definition of $\|_\leqslant$ needs more notation. We let p and p' be new unary relation symbols. We let $\mathbf{mk} : S(R \cup \{\leqslant\}, C) \to \mathbf{S}(R \cup \{\leqslant, p\}, C)$ (read "**mark**") be the transformation of a structure $S = \langle \mathbf{D}_S, (r_S)_{r \in R}, \leqslant_S, (c_S)_{c \in C} \rangle$ into the structure $T = \langle \mathbf{D}_S, (r_S)_{r \in R}, \leqslant_S, p_T, (c_S)_{c \in C} \rangle$ where $p_T(x)$ holds if and only if x is not the value of a constant in C. (This is indeed a qfd operation for each pair (R, C).) We let

$\mathbf{mk}' : \mathbf{S}(R \cup \{\leqslant\}, C') \to \mathbf{S}(R \cup \{\leqslant, p'\}, C')$ be the qfd operation defined similarly with respect to C'. We shall define later a qfd operation **conc** (for "**conc**atenation") and let

$$S \|_{\leqslant} S' = \mathbf{conc}(\mathbf{mk}(S) \| \mathbf{mk}'(S')). \tag{1}$$

The idea is that $S \|_{\leqslant} S'$ is ordered as follows: first all elements which are values of constants, in a certain order fixed from the constants, then the elements of $\mathbf{D}_S$ which are not values of constants, in the order of S, and finally the elements of $\mathbf{D}_{S'}$ which are not values of constants in the order of S'. The qfd operation **conc** is defined by $\Gamma = \langle \psi, (\theta_r)_{r \in R \cup R' \cup \{\leqslant\}}, (\tau_c)_{c \in C \cup C'} \rangle$ where:

ψ is the formula **true**,

τ_c is c for every $c \in C \cup C'$,

θ_r is $r(x_1, \ldots, x_n)$ for $r \in R \cup R'$ of arity n,

$\theta_{\leqslant}$ is the formula

$$\begin{aligned}
&(p(x_1) \wedge p'(x_2)) \\
&\quad \vee (p(x_1) \wedge p(x_2) \wedge x_1 \leqslant x_2) \vee (p'(x_1) \wedge p'(x_2) \wedge x_1 \leqslant x_2) \\
&\quad \vee (\neg p(x_1) \wedge \neg p'(x_1) \wedge p(x_2)) \vee (\neg p(x_1) \wedge \neg p'(x_1) \wedge p'(x_2)) \\
&\quad \vee (\neg p(x_1) \wedge \neg p'(x_1) \wedge \neg p(x_2) \wedge \neg p'(x_2) \wedge \theta'(x_1, x_2)).
\end{aligned}$$

In this definition, $\theta'(x_1, x_2)$ is a quantifier-free formula that compares x_1 and x_2 defined by constants as follows:

$\theta'(x_1, x_2)$ holds if and only if $x_1 = x_2$ or $\mathbf{c}(x_1) < \mathbf{c}(x_2)$

where $\mathbf{c}(x)$ is the $<$-smallest constant c in $C \cup C'$ equal to x and $<$ is some fixed linear order on $C \cup C'$.

For every $S \in \mathscr{S}_{\leqslant}(R, C)$ and $S' \in \mathscr{S}_{\leqslant}(R', C')$ the structure defined by (1) is linearly ordered and, furthermore,

$$\mathbf{uno}(S \|_{\leqslant} S') = \mathbf{uno}(S) \| \mathbf{uno}(S').$$

Note that $\|_{\leqslant}$ is not commutative, even in the case where $R = R'$, $C = C'$ although $\|$ is commutative: $\mathbf{S}(R, C)^2 \to \mathbf{S}(R, C)$.

Let $L \subseteq \mathbf{S}(R, C)$ be MS($\leqslant$)-definable. This means that $L = \mathbf{uno}(L')$ where L' is the set of structures in $\mathbf{S}_{\leqslant}(R, C)$ satisfying an order-invariant MS-definable property. It follows that L' is recognizable as a subset of $\mathbf{S}(R \cup \{\leqslant\}, C)$. It is thus recognizable with respect to the operations of the form $\mathbf{def}_{A \leqslant}$ (which are qfd) and $\|_{\leqslant}$ which are operations formed as finite combinations of $\|$ and qfd operations (see Proposition 1.3). Hence L' is a recognizable subset of $\mathbf{S}_{\leqslant}(R, C)$ with respect to these "ordered" operations. Since $L' = \mathbf{uno}^{-1}(\mathbf{uno}(L'))$ by the order-invariance of its defining property, its image under **uno** is recognizable (by Proposition 1.2). □

Corollary 4.2. *Every MS($\leqslant$)-definable set of graphs is recognizable.*

Proof. This is an immediate consequence of Theorem 4.1 and Proposition 1.3 since the operation of parallel composition of sourced graphs is nothing but the parallel composition of the corresponding structures, as already observed, and since every source renaming operation is a qfd operation on the representing structures. □

Corollary 4.3. *For every set L of forests, the following properties are equivalent:*

(1) *L is CMS-definable,*

(2) *L is $MS(\leqslant)$-definable,*

(3) *L is recognizable.*

The cases of a set L of discrete graphs or of trees are two special cases of this result.

Proof. Immediate consequence of Corollary 4.2 and the result form [6] that such a set is recognizable if and only if it is CMS-definable. □

Proposition 4.4. *The inverse image of an $MS(\leqslant)$-definable set of structures by a definable transduction is $MS(\leqslant)$-definable.*

We had in Proposition 1.1 similar statements for MS- and CMS-definable sets.

Proof. Let τ be a definable transduction of structures, let L and L' be two sets such that $L = \tau^{-1}(L')$ and L' is MS($\leqslant$)-definable. One can "enrich" τ in order to make it into a transduction τ' that defines a linear order on the output structures from any linear order on the input structure. The set $L'(\leqslant)$ is MS-definable by the definition. Hence the set $\tau'^{-1}(L'(\leqslant))$ is MS-definable by Proposition 1.2.

Let us prove that $L(\leqslant) = \tau'^{-1}(L'(\leqslant))$. Let $(S, \leqslant)$ belong to $L(\leqslant)$. Some image $(S', \leqslant')$ of $(S, \leqslant)$ under τ' belongs to $L'(\leqslant)$. Hence $(S, \leqslant)$ belongs to $\tau'^{-1}(L'(\leqslant))$. If, conversely, $(S, \leqslant)$ belongs to $\tau'^{-1}(L'(\leqslant))$, then S belongs to $\tau^{-1}(L') = L$, hence $(S, \leqslant)$ belongs to $L(\leqslant)$. Hence, $L(\leqslant)$ is MS-definable and L is MS($\leqslant$)-definable.

Let us sketch the construction of τ'. Assuming that τ is a transduction from $\mathscr{S}(R, C)$ into $\mathscr{S}(Q, D)$ that defines the domain of an object structure T as $D_1 \times \{1\} \cup D_2 \times \{2\} \cup \cdots \cup D_k \times \{k\}$ where $D_1, D_2, \ldots, D_k$ are MS-definable subsets of the domain of the input structure S, we let $\leqslant$ be a new binary symbol. In order to get a definable transduction τ' from $\mathscr{S}(\{R, \leqslant\}, C)$ into $\mathscr{S}(\{Q, \leqslant\}, D)$, we only add to the definition scheme of τ the formulas $(\theta_w)_{w \in \{\leqslant\}^* k}$ (see Section 1), in such a way that they define in T the linear order such that: $(d, i) \leqslant_T (d', j)$ if and only if either $i < j$, or $i = j$ and $d \leqslant_S d'$. □

A *bidefinable coding* is a transduction between two classes of abstract structures which is bijective, definable and such that its inverse is also definable.

Corollary 4.5. *Let $\mathscr{C}$ and $\mathscr{D}$ be two classes of graphs in bijection by a bidefinable coding. If every CMS-definable subset of $\mathscr{C}$ is MS-definable, the same holds for every*

MS-definable subset of $\mathscr{D}$. If every MS($\leqslant$)-definable subset of $\mathscr{C}$ is CMS-definable, the same holds for every MS($\leqslant$)-definable subset of $\mathscr{D}$.

Proof. Immediate consequence of Propositions 1.1 and 4.4. □

Remark 4.6. In the definition of an MS($\leqslant$)-definable set, the condition that the MS-property be order-invariant is essential if we want to have Theorem 4.1. Sets of structures defined by a condition of the form: "there exists a linear order on S such that $(S, \leqslant)$ satisfies P" where P is an MS-property, are not always recognizable. As a counterexample, one can take the sets with a unary predicate such that the number of elements that satisfy this predicate is equal to the number of those that do not.

5. The reconstruction of a tree from its leaves

In this section, we consider whether and how it is possible to reconstruct a tree from a ternary relation on its leaves derived from its internal structure, in a way that is definable by MS- or by MS($\leqslant$)-formulas. We shall give a construction using MS($\leqslant$)-formulas that works for proper trees and, for each integer d, another one using MS-formulas only, that works for trees of degree at most d. This is an example of the use of MS($\leqslant$)-formulas and will be applied in the next section to the logical characterization of recognizable sets of cographs.

In this section and in the following one, the vertices of trees will be called *nodes*. This is useful in complicated situations where we deal with trees representing graphs. The set of nodes of a tree T is denoted by $\mathbf{N}_T$. A node that is not a leaf is *internal*. We shall denote by $\mathbf{L}_T$ the set of leaves of T. We shall denote by $\leqslant_T$ (or $\leqslant$ if the context makes T clear) the partial order $\mathbf{edg}_T^*$ on $\mathbf{N}_T$. Every two nodes x and y have a greatest lower bound for $\leqslant_T$ that we shall denote by $x \wedge y$ (or by $x \wedge_T y$ if necessary). Finally, for every triple of leaves of T, we let $\mathbf{R}_T(x, y, z)$ hold if and only if $x \wedge y \leqslant z$. We let $\lambda(T) = \langle \mathbf{L}_T, \mathbf{R}_T \rangle$ (λ stands for "leaves").

A tree T can be given either as the structure $\langle \mathbf{N}_T, \mathbf{edg}_T \rangle$ or as the structure $\langle \mathbf{N}_T, \leqslant_T \rangle$. These two representations are equivalent for MS-logic because each of $\mathbf{edg}_T$ and $\leqslant_T$ is MS-definable in terms of the other.

A tree is *proper* if it has at least two leaves and no node has exactly one successor. The number of nodes of a proper tree is at most twice the number of its leaves.

Proposition 5.1. *Given a finite set L of cardinality at least 2 and a ternary relation R on L, there is at most one proper tree T such that $\lambda(T) = \langle L, R \rangle$.*

Proof. Assume that there exists a proper tree T with $\mathbf{L}_T = L$ and $\mathbf{R}_T = R$. Let $\prec$ be the binary relation on $L \times L$ such that

$$(x, y) \prec (z, t) \text{ if and only if } R(x, y, z) \text{ and } R(x, y, t). \tag{1}$$

It is clear that $(x, y) \prec (z, t)$ if and only if $x \wedge_T y \leqslant z \wedge_T t$, hence that $\prec$ is a quasi-order. For any two leaves x and y, we have $(x, x) \prec (y, y)$ if and only if $x = y$ because $x \wedge_T x = x$, $y \wedge_T y = y$. Since T is proper, every internal node of T is of the form $x \wedge y$ where x and y are distinct leaves. It follows that $\langle \mathbf{N}_T, \leqslant_T \rangle$ is isomorphic to the structure $\langle L \times L/\equiv, \prec/\equiv \rangle$ where $(x, y) \equiv (z, t)$ if and only if $(x, y) \prec (z, t)$ and $(z, t) \prec (x, y)$. In this isomorphism, the leaves x correspond to the pairs (x, x) (which are singletons in their equivalence classes). This gives a definition of T in terms of L and R. Hence T is the unique tree such that $\mathbf{L}_T = L$ and $\mathbf{R}_T = R$. □

Remarks. (1) If $\langle L, R \rangle = \lambda(T)$ where T is not proper but $\mathbf{card}(L) \geqslant 2$, we obtain from this construction a proper tree T' such that $\lambda(T') = \langle L, R \rangle$.

(2) The class of structures $\langle L, R \rangle$ of the form $\lambda(T)$ for some tree is characterized by a first-order formula. (To see this note that a structure $\langle D, \leqslant \rangle$ is a tree if and only if $\leqslant$ is a partial order having a least element (the root) and such that every set of the form $\{x \mid u \leqslant x \leqslant v\}$ is linearly ordered; since we shall not use this fact, we omit the details.)

Corollary 5.2. *For every first-order formula $\varphi \in \mathscr{L}(\{\leqslant\}, \{X_1, \ldots, X_k\})$ one can construct a first-order formula $\varphi' \in \mathscr{L}(\{R\}, \{X_1, \ldots, X_k\})$ such that, for every proper tree T, for all sets $L_1, \ldots, L_k \subseteq \mathbf{L}_T$:*

$$\langle \mathbf{L}_T, \mathbf{R}_T, L_1, \ldots, L_k \rangle \models \varphi'$$

if and only if

$$\langle \mathbf{N}_T, \leqslant_T, L_1, \ldots, L_k \rangle \models \varphi.$$

Proof (*sketch*). The formula φ has no set quantification. Every object variable x of φ will be represented in φ' by a pair of variables (x', x''). We construct φ' by induction on the structure of φ. We begin with the atomic formulas:

$x \leqslant y$ translates into $\theta(x', x'', y', y'')$ where $\theta(x_1, x_2, x_3, x_4)$ is the formula:

$$R(x_1, x_2, x_3) \wedge R(x_1, x_2, x_4),$$

$x = y$ translates into $\theta(x', x'', y', y'') \wedge \theta(y', y'', x', x'')$,

$x \in X_i$ translates into $x' = x'' \wedge x' \in X_i$.

A formula of the form $\neg\varphi$ or $\varphi \wedge \psi$ translates into $\neg\varphi'$ or $\varphi' \wedge \psi'$, respectively. A formula of the form $\exists x \varphi$ translates into $\exists x', x'' \varphi'$. □

We would like to extend this corollary to MS-formulas. This could be possible if the transduction λ^{-1} (i.e., the reconstruction of T from $\langle \mathbf{L}_T, \mathbf{R}_T \rangle$) would be a definable transduction, which does not follow from the construction given in the proof of Corollary 5.2 (and we conjecture that there is no alternative construction that would make λ^{-1} definable). However, we shall prove that λ^{-1} is indeed definable in two special cases: when $\langle \mathbf{L}_T, \mathbf{R}_T \rangle$ is equipped with a linear ordering and when the tree T to be reconstructed has degree bounded by some fixed number.

Theorem 5.3. *The transduction of relational structures* $\{(\langle L, R, P\rangle, \langle N, suc\rangle) \mid P$ *is a linear order on* $L, \langle N, suc\rangle$ *is a proper tree* T *and* $\langle L, R\rangle = \lambda(T)\}$ *is definable. In other words, the transduction* λ^{-1} *is definable provided the input structure (namely* $\langle L, R\rangle$*) is ordered.*

Proof. Let T be proper, let $\langle L, R\rangle = \lambda(T)$, let P be a linear order on L. We derive from P a strict linear order on the successors of the nodes of T:

> if y, z are two successors of x, we let $y <_P z$ if and only if the P-smallest leaf below y is strictly P-smaller than the P-smallest leaf below z.

Since P is proper, every internal node x has a first successor and a second successor (with respect to the strict order $<_P$), denoted respectively by $\mathbf{suc1}(x)$ and $\mathbf{suc2}(x)$. Every internal node x has a *representative leaf*, denoted by $\mathbf{rep}(x)$, that we define by: $\mathbf{rep}(x) = \mathbf{suc2}^*(\mathbf{suc1}(x))$ where for every node y, $\mathbf{suc2}^*(y) = \mathbf{such2}^n(y)$ and n is the unique integer ($n \geqslant 0$) such that $\mathbf{suc2}^n(y)$ is a leaf. The mapping **rep** is thus a bijection of $\mathbf{N}_T - \mathbf{L}_T$ onto a subset of $\mathbf{L}_T$.

Our next purpose is to find an MS-formula $\theta \in \mathscr{L}(\{R, P\}, \{x, y, z\})$ such that, for every x, y, z in L,

$$\langle L, R, P\rangle \models \theta(x, y, z) \text{ if and only if } z = \mathbf{rep}(x \wedge y).$$

By Corollary 5.2, one can construct first-order formulas $\varphi_1, \ldots, \varphi_4$ such that, for all $u, v, u', v' \in L$:

$\langle L, R\rangle \models \varphi_1(u, v, u', v')$ if and only if $u \wedge v \leqslant_T u' \wedge v'$,

$\langle L, R\rangle \models \varphi_2(u, v, u', v')$ if and only if $u' \wedge v'$ is a successor of $u \wedge v$,

$\langle L, R, P\rangle \models \varphi_3(u, v, u', v')$ if and only if $u' \wedge v'$ is the first successor of $u \wedge v$,

$\langle L, R, P\rangle \models \varphi_4(u, v, u', v')$ if and only if $u' \wedge v'$ is the second successor of $u \wedge v$.

We observe that for all $x, y, z \in L$, $z = \mathbf{rep}(x \wedge y)$ if and only if there exists a nonempty sequence $x_1, x_2, \ldots, x_n$ of leaves such that:

$z \wedge x_1$ is the first successor of $x \wedge y$,

$z \wedge x_i$ is the second successor of $z \wedge x_{i-1}$ $(2 \leqslant i \leqslant n)$, $z = x_n$.

These conditions can also be written as follows:

$z = \mathbf{rep}(x \wedge y)$ if and only if there exists a set of leaves X and a leaf $x' \in X$ such that:

(i) $z \wedge x'$ is the first successor of $x \wedge y$,

(ii) the graph $\langle X, \rightarrow\rangle$ where $u \rightarrow v$ is defined by

"$z \wedge v$ is the second successor of $z \wedge u$"

is a path with first element x' and last element z.

From this latter formulation, the construction of an MS-formula $\theta(x, y, z)$ defining the relation $z = \mathbf{rep}(x \wedge y)$ follows immediately. We now construct from $\langle L, R, P\rangle$ the following structure $\langle N, suc\rangle$:

- $N = L \times \{1\} \cup L' \times \{2\}$ where $L' = \{\mathbf{rep}(x \wedge y) \mid x \neq y,\ x, y \in L\}$.
- $((z, i), (z', j)) \in suc$ if and only if.

either $j = 1$ and $i = 2$ and there exist u, v with $u \neq v$, such that z' is a successor or $u \wedge v$ and $z = \mathbf{rep}(u \wedge v)$ (this can be expressed by $\varphi_2(u, v, z', z') \wedge \theta(u, v, z)$)
or $i = j = 2$ and there exist u, v, u', v' with $u \neq v$, $u' \neq v'$ such that $z = \mathbf{rep}(u \wedge v)$, $z' = \mathbf{rep}(u' \wedge v')$ and $u' \wedge v'$ is a successor of $u \wedge v$ (this can be expressed by the formula $\varphi_2(u, v, u', v') \wedge \theta(u, v, z) \wedge \theta(u', v', z')$).

By using the formulas $\varphi_1, \ldots, \varphi_4$, θ one can construct a definition scheme and prove thus that the transformation $\tau = \langle L, R, P \rangle \mapsto \langle N, suc \rangle$ is definable.

If P is a linear order on L and $\langle L, R \rangle = \lambda(T)$ then $\langle N, suc \rangle$ is isomorphic to T. Note that $\langle N, suc \rangle = \tau(\langle L, R, P \rangle)$ may be well-defined, even if $\langle L, R \rangle$ is not of the form $\lambda(T)$. An additional MS-formula ψ can be written such that, for every $\langle L, R, P \rangle$, we have $\langle L, R, P \rangle \models \psi$ if and only if P is a linear order on L and the structure $\tau(\langle L, R, P \rangle) = \langle N, suc \rangle$ is a tree, the leaves of which are the elements of N of the form $(x, 1)$ and such that for every x, y, z in L we have:

(1) $(x, x, z) \in R$ if and only if $x = z$ and

(2) if $x \neq y$, then $(x, y, z) \in R$ if and only if there exists $u \in L$ such that $\theta(x, y, u)$ holds, and $(u, 2)$ is an ancestor of $(z, 1)$ in the tree $\langle N, suc \rangle$.

The restriction of τ to the structures that satisfy ψ is thus the desired transduction. □

This representation of an internal node of a proper ordered tree by a leaf is also used in [20] for proving that every MS-formula describing proper ordered trees of bounded degree can be translated into an MS-formula where set quantifications are restricted to sets of leaves.

Proposition 5.4. *One can define by MS-formulas a linear ordering on every structure $\lambda(T)$ such that T is a tree of outdegree at most 2.*

Proof. If $\langle L, R \rangle = \lambda(T)$ where $\mathbf{card}(L) \geqslant 2$ and T has outdegree at most 2, then there exist by Proposition 5.1 and Corollary 5.2 a proper binary tree T' (all internal nodes have outdegree 2) such that $\lambda(T') = \langle L, R \rangle$. Hence we can restrict our attention to proper binary trees. Let $T = \langle \mathbf{N}_T, \mathbf{suc}_T \rangle$ be such a tree. We let $\gamma : \mathbf{N}_T \to \{a, b, c\}$ be a node-coloring of T such that:

for every internal node x, if y and z are the successors of x then
$\gamma(x) \neq \gamma(y) \neq \gamma(z) \neq \gamma(x)$. (2)

There exists γ satisfying these conditions; let us fix one and define

$$L_w = \mathbf{L}_T \cap \gamma^{-1}(w) \quad \text{for } w \in \{a, b, c\}. \tag{3}$$

Our objective is to MS-define in the structure $\langle \mathbf{L}_T, \mathbf{R}_T, L_a, L_b, L_c \rangle$ the coloring γ of $\mathbf{N}_T$. From γ it will be easy to obtain a linear order on $\mathbf{L}_T$ and to apply the construction of Theorem 5.3.

We need some definitions. If x is an internal node of T we let $B(x)$ be the smallest (for inclusion) subset of $\mathbf{N}_T$ that contains x and such that, for each $y \in B(x)$, if

$z \in \mathbf{Suc}(\mathbf{Suc}(y))$ and $\gamma(z) = \gamma(y)$ then $z \in B(x)$. (There are either 2, 1, or 0 such nodes z associated in this way with any node y.) We let $B'(x)$ be the set of leaves that are successors of nodes in $B(x)$. We let $B''(x) = B(x) \cap \mathbf{L}_T$.

Claim 1. (1) $\gamma(y) = \gamma(x)$ *for every* $y \in B(x)$.
(2) $\gamma(y) \neq \gamma(x)$ *for every* $y \in B'(x)$.
(3) $B(x)$ *is closed under* $\wedge$.

Proof. (1) is clear from the definition of $B(x)$.
(2) follows from (1) since if $y \in \mathbf{Suc}(z)$ then $\gamma(y) \neq \gamma(z)$.
(3) Let us construct $B(x)$ by starting from x and adding nodes as required by the definition; whenever one adds z_1 and z_2 (or just z_1) from y with $z_1, z_2 \in \mathbf{Suc}(\mathbf{Suc}(y))$, then

$$z_1 \wedge z_2 = y, \quad z_1 \wedge y = y, \quad z_2 \wedge y = y, \quad z_1 \wedge w = y \wedge w = z_2 \wedge w$$

for any already existing node w other than y, z_1, z_2. Hence, the closure under $\wedge$ is preserved at each step of the construction of $B(x)$. □

For every node $x \in T$ we let $\mathbf{pred}(x)$ be its predecessor (i.e., its father). If $X \subseteq \mathbf{N}_T$, we let $\mathbf{Pred}(X) = \{\mathbf{pred}(x) \mid x \in X\}$. For $X, Y \subseteq \mathbf{L}_T$ we define

$$N(X, Y) = \{x \wedge y \mid x, y \in X \cup \mathbf{Pred}(Y)\}.$$

Note that $X \cup \mathbf{Pred}(Y) \subseteq N(X, Y)$ since $z \wedge z = z$ for every node z.

Claim 2. $B(x) = N(B''(x), B'(x))$ *for every internal node* x *of* T.

Proof. We prove that every $x' \in B(x)$ belongs to $N(B''(x), B'(x))$. We let $\ell(x')$ be the largest distance of x' to a leaf below it. We use an induction on $\ell(x')$
If $\ell(x') = 0$, then x' is a leaf, hence $x' \in B''(x)$, so $x' \in N(B''(x), B'(x))$.
If $\ell(x') = 1$, then the two successors of x' are leaves, hence they belong to $B'(x)$ and $x' \in N(B''(x), B'(x))$ since $\mathbf{Pred}(B'(x)) \subseteq N(B''(x), B'(x))$.
If $\ell(x') \geqslant 2$, then we distinguish two cases.
Case 1: $\mathbf{Suc}(x') = \{y_1, y_2\}$, y_1 is a leaf and y_2 is not. Then $y_1 \in B'(x)$ and $x' \in N(B''(x), B'(x))$ since $\mathbf{Pred}(B'(x)) \subseteq N(B''(x), B'(x))$.
Case 2: $\mathbf{Suc}(\mathbf{Suc}(x')) = \{y_1, y_2, y_3, y_4\}$. Two of them, say y_1 and y_2, are in $B(x)$ (by the way T is colored and the set $B(x)$ is defined) and $x' = y_1 \wedge y_2$. By the induction hypothesis y_1 and y_2 belong to $N(B''(x), B'(x))$. Hence we have $y_1 = z_1 \wedge z_2$ and $y_2 = z_3 \wedge z_4$ for $z_1, z_2, z_3, z_4 \in B''(x) \cup \mathbf{Pred}(B'(x))$. Since $x' = y_1 \wedge y_2$, we have also $x' = z_1 \wedge z_3$, hence $x' \in N(B''(x), B'(x))$.
We also have $N(B''(x), B'(x)) \subseteq B(x)$ since $B(x)$ is closed under $\wedge$, $B''(x) \subseteq B(x)$ and $\mathbf{Pred}(B'(x)) \subseteq B(x)$. □

We make a break for an example.

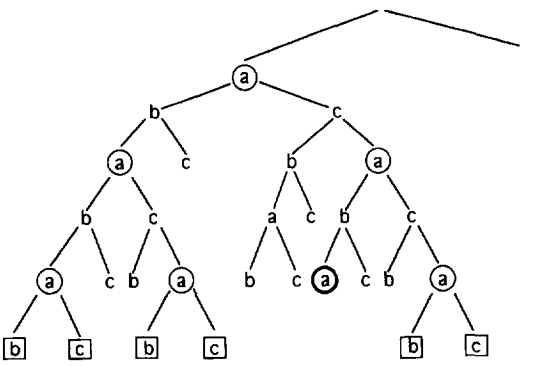

Fig. 5.

Example. Fig. 5 shows a tree, with the coloring γ of a node x and of its descendents (x has color a). The elements of $B(x) - B''(x)$ are circled, the unique element in $B''(x)$ is circled in bold, those of $B'(x)$ are boxed. Note that not all nodes colored by a are in $B(x)$.

We now continue the proof. If $x, y \in \mathbf{L}_T$, if $X, Y \subseteq \mathbf{L}_T$, we say that (X, Y) is a *good* pair for (x, y) if:

(1) $X \cap Y = \emptyset$ and $x \wedge y \in N(X, Y)$,

(2) for every $z \in N(X, Y)$ and $t \in \mathbf{Suc}(z)$ we have $t \notin N(X, Y)$,

(3) for every $z \in N(X, Y)$, either z has two successors in Y, or z has one successor in Y and another one, t, such that $\mathbf{Suc}(t) \cap N(X, Y)$ is singleton, or z has two successors t, t' such that $\mathbf{Suc}(t) \cap N(X, Y)$ and $\mathbf{Suc}(t') \cap N(X, Y)$ are both singletons.

Claim 3. *For every internal node x of T, if y, z are leaves such that $x = y \wedge z$, then $(B''(x), B'(x))$ is good for (y, z).*

Proof. Easy verification from the definitions of $B'(x)$ and $B''(x)$. □

Claim 4. *For every two distinct leaves y, z of T, $\gamma(y \wedge z) = a$ if and only if there exists a good pair (X, Y) for (y, z) such that $X \subseteq L_a$ and $Y \subseteq L_b \cup L_c$.*

Proof. The "only if" part follows from Claims 1–3: one takes $X = B''(y \wedge z)$ and $Y = B'(y \wedge z)$. Then (X, Y) is good for (y, z) by Claim 3; $X \subseteq L_a$ and $Y \subseteq L_b \cup L_c$ by Claim 1; X and Y are disjoint because of the colors.

Let, conversely, (X, Y) be a good pair for (y, z) such that $X \subseteq L_a$ and $Y \subseteq L_b \cup L_c$. We let $x = y \wedge z$. It is enough to prove that every $w \in N(X, Y)$ has color a since $y \wedge z \in N(X, Y)$ by the definition of a good pair. We prove this by induction on $\ell(w)$ (see Claim 2 for ℓ). If $\ell(w) = 0$ then w is a leaf and $w \in X$. Since $X \subseteq L_a$, we get the result. If $\ell(w) = 1$ then w has two successors w_1 and w_2 which are leaves, hence must be in Y since (X, Y) is good. Hence they have colors b or c, and not both the same. Hence w has color a by the coloring rule of T. If $\ell(w) \geqslant 2$ then there are two cases.

Case 1: $\mathbf{Suc}(\mathbf{Suc}(w)) = \{y_1, y_2, y_3, y_4\}$. By the definition of a good pair, two of these nodes, say y_1 and y_2, belong to $N(X, Y)$. They have both color a by the induction hypothesis. Let y_3 be the brother of y_1 and y_4 be that of y_2. Then y_3 has color b or c. Say b. Then y_4 cannot have colors a (as brother of y_2 with color a) or color b (because then $y_1 \wedge y_3$ and $y_2 \wedge y_4$ would be two brothers with the same color c). Hence y_4 has color c, $y_1 \wedge y_3$ and $y_2 \wedge y_4$ have color c and b, respectively, and w has color a.

Case 2: $\mathbf{Suc}(w) = \{y_1, y_2\}$, y_1 is a leaf, $\mathbf{Suc}(y_2) = \{y_3, y_4\}$. Then because (X, Y) is good, one of y_3, y_4, say y_3, is in $N(X, Y)$, hence has color a by induction hypothesis.

Subcase 1: $y_1 \in Y$. Then y_1 has color b or c, say b. Then y_2 cannot have color a (because of y_3) or b (as brother of y_1 with color b). Hence it has color c. Hence w has color a.

Subcase 2: $y_1 \in N(X, Y)$; this contradicts the definition of a good pair (Clause 2).

Hence we have proved that w has color a as desired. □

Claim 5. *One can construct MS-formulas θ and φ_w in $\mathcal{L}(\{R\}, \{x, y, W_a, W_b, W_c\})$ for every $w \in \{a, b, c\}$ (where W_a, W_b, W_c are set variables) such that, for every proper binary tree T, for every $z, t \in \mathbf{L}_T$ we have:*

(i) *$\lambda(T) \models \varphi_w(z, t, L_a, L_b, L_c)$ if and only if $\gamma(z \wedge t) = w$, where γ is a node-coloring and L_a, L_b, L_c are sets of leaves that satisfy* (2) *and* (3), *and*

(ii) *the binary relation $<$ such that*

$$z < t \text{ if and only if } \lambda(T) \models \theta(z, t, L_a, L_b, L_c)$$

is a linear order on $\mathbf{L}_T$.

Proof. (i) The construction of φ_a is based on Claim 4. The property that a pair of sets of leaves (X, Y) is good for a pair of leaves (x, y) is first-order in the structure $\langle \mathbf{N}_T, \leqslant_T \rangle$ (see the definition; note that $x_1 \wedge_T x_2 = x_3$ is first-order definable from $\leqslant_T$). Hence, it is first-order in $\langle \mathbf{L}_T, \mathbf{R}_T \rangle$ by Corollary 5.2. In order to express that $\gamma(z \wedge t) = a$, it is enough to write:

there exist X, Y such that $X \subseteq L_a$, $Y \subseteq L_b \cup L_c$ and (X, Y) is good for (z, t).

This is MS-expressible in $\langle \mathbf{L}_T, \mathbf{R}_T \rangle$ with the help of L_a, L_b, L_c and gives φ_a. By permuting the letters a, b, c one obtains φ_b and φ_c.

(ii) We now define for $z, t \in \mathbf{L}_T$: $z < t$ if and only if there exist $z', t' \in \mathbf{L}_T$ such that $\mathbf{Suc}_T(z \wedge t) = \{z \wedge z', t \wedge t'\}$, $\gamma(z \wedge z') <_{abc} \gamma(t \wedge t')$ (where $<_{abc}$ is the order on colors: $a <_{abc} b <_{abc} c$). This relation is MS-expressible with the help of the formulas $\varphi_a, \varphi_b, \varphi_c$. □

We can now finish the proof the proposition. We let W_a, W_b, W_c be set variables. For all sets $L'_a, L'_b, L'_c \subseteq \mathbf{L}_T$, the formula $\theta(x, y, W_a, W_b, W_c)$ defines a binary relation $S(L'_a, L'_b, L'_c)$ on $\mathbf{L}_T$, where W_p takes value L'_p, for $p = a, b, c$. (If the sets L'_a, L'_b, L'_c are not associated with a coloring γ satisfying (2) and (3), $S(L'_a, L'_b, L'_c)$ may not be a linear order.) One can construct an MS-formula $\delta \in \mathcal{L}(\{R\}, \{W_a, W_b, W_c\})$ such that for all $L'_a, L'_b, L'_c \subseteq \mathbf{L}_T$:

$(\mathbf{L}_T, \mathbf{R}_T, L'_a, L'_b, L'_c) \models \delta$ if and only if (L'_a, L'_b, L'_c) is a partition of $\mathbf{L}_T$ and the binary relation $S(L'_a, L'_b, L'_c)$ is a linear order on $\mathbf{L}_T$.

The pair of formulas (δ, θ) specifies a linear order on every structure $\langle L, R\rangle$ which is of the form $\lambda(T)$ for some proper binary tree T, because, if $\langle L, R\rangle = \lambda(T)$, the sets L_a, L_b, L_c associated with a coloring of T (according to (2) and (3)) satisfy δ, and $S(L_a, L_b, L_c)$, defined by θ, is a linear order. (Of course δ and θ may also specify a linear order on structures $\langle L, R\rangle$ not of the form $\lambda(T)$ but we need not care.) □

Proposition 5.5. *Let $d \in \mathbb{N}$, $d > 2$. One can MS-define a linear ordering of every structure $\lambda(T)$ where T is a tree of outdegree at most d.*

Proof. The proof will develop the tools introduced for the proof of Proposition 5.4. As in Proposition 5.4 we need only consider proper trees.

Let T be a proper tree of outdegree at most d. We shall say that two nodes are at *distance* n if the undirected path (where edges can be traversed in either direction) linking them has n edges. Let $D = (d+1)^4$. For every $x \in \mathbf{N}_T$, there are at most $(d+1)(d^3 + d^2 + d + 1)$ nodes $y \neq x$ at distance at most 4 of x. The number is less than D. We define $A := \{a, b, c\} \times \{1, \ldots, D\}$. Each subset $\{a, b, c\} \times \{i\}$ of A is called a *color class*. An *A-coloring* of T is a mapping $\gamma : \mathbf{N}_T \to A$. We define an A-coloring γ of T by the following algorithm, using an arbitrary linear ordering α of $\mathbf{N}_T$:

1. Define $\gamma(\mathbf{root}_T) := (a, 1)$.
2. **Until** all nodes are colored **repeat**:
   - 2.1. Let x be the $\leqslant_\alpha$-smallest colored node having uncolored successors (where $\leqslant_\alpha$ is the lexicographic ordering associated with α, see Section 2) and let $(w, i) = \gamma(x)$.
   - 2.2. Let $y_1, y_2, \ldots, y_n$ be the successors of x enumerated by increasing order for α; {**fact**: none of them is colored}.
   - 2.3. Define $\gamma(y_1) := (w', i)$, $\gamma(y_2) := (w'', i)$, where $\{w', w''\} = \{a, b, c\} - \{w\}$ and $w' <_{abc} w''$.
   - 2.4. **For each** $j = 3, \ldots, n$ **do**:
     define $\gamma(y_j) := (w, i')$, where i' is the smallest element of $\{1, \ldots, D\}$ that is not the second component of the color of any colored node at distance at most 4 of y_j; {**fact:** i' does exist because D is large enough}.

See Fig. 6 for an example.

We shall write $c \approx c'$ to indicate that two colors c and c' are in the same color class, and $x \to y$ to indicate an edge from x to y.

Claim 1. (0) *No two adjacent vertices have the same color.*

Let x, y, z, t, u be nodes of T.

(1) *If $y \leftarrow x \to z$ and $\gamma(y) \approx \gamma(z)$ then $\gamma(x) \approx \gamma(y)$ and $\gamma(y) \neq \gamma(z)$.*

(2) *If $x \to y \to z$ and $\gamma(x) \approx \gamma(z)$ then $\gamma(y) \approx \gamma(x)$.*

(3) *If $y \leftarrow x \to z \to t$ and $\gamma(y) \approx \gamma(t)$ then $\gamma(x) \approx \gamma(z) \approx \gamma(y)$.*

(4) *If $y \leftarrow z \leftarrow x \to t \to u$ and $\gamma(y) \approx \gamma(u)$ then $\gamma(z) \approx \gamma(x) \approx \gamma(t) \approx \gamma(y)$.*

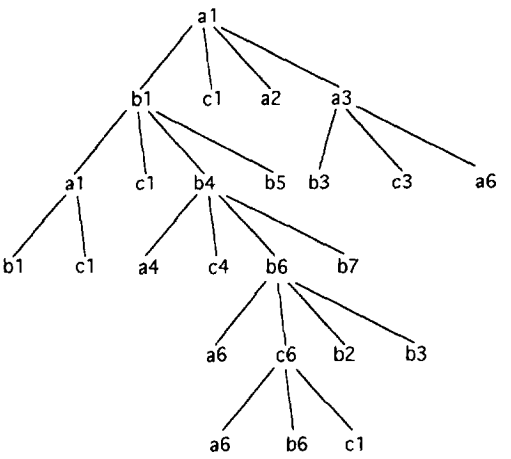

Fig. 6.

Proof. (0) That no two adjacent vertices have the same color is clear from the definition of γ.

(1) The algorithm above defines simultaneously $\gamma(y)$ and $\gamma(z)$: from Clauses 2.3 and 2.4, the result follows.

(2) Consider the step where $\gamma(z)$ is defined. This cannot be by Clause 2.4 because then $\gamma(z)$ would not be in the same class as $\gamma(x)$ which has been defined earlier and z is at distance 2 of x. This must be by Clause 2.3 which gives $\gamma(z) \approx \gamma(y)$.

(3) The last of nodes x, y, z, t, the color of which is defined, must be t. If $\gamma(t)$ is defined by Clause 2.3 then $\gamma(t) \approx \gamma(z)$ and the result follows from (2). It cannot be by Clause 2.4 because t is at distance 3 of y and we assumed that $\gamma(t) \approx \gamma(y)$. (This would contradict Clause 2.4)

(4) The last node, the color of which is defined, is either y or u. Say y, without loss of generality. If $\gamma(y)$ is defined by Clause 2.3 then $\gamma(y) \approx \gamma(z)$ and we conclude by (3). It cannot be by Clause 2.4 since y and u are at distance 4. (This would contradict Clause 2.4.) □

For $p \in A$ we let $L_p = \{x \in \mathbf{L}_T \mid \gamma(x) = p\}$. As in Claim 4, we shall prove that for $x, y \in \mathbf{L}_T$, $\gamma(x \wedge y)$ can be determined from the sets L_p. We redefine a few notions from the proof of Proposition 5.4. For $x \in \mathbf{N}_T - \mathbf{L}_T$ we let $B(x)$ and $B''(x)$ be as in this proof. We let $B'(x)$ be the set of leaves z such that $z \in \mathbf{Suc}(y)$ for some $y \in B(x)$ and $\gamma(z)$ and $\gamma(y)$ are in the same color class. Claim 1 holds, but, in addition, $\gamma(y)$ and $\gamma(x)$ are in the same color class for every $y \in B'(x)$. For $X, Y \subseteq \mathbf{L}_T$ we define $N(X, Y)$ exactly as in the proof of Proposition 5.4 and Claim 2 holds. (The proof is essentially the same, we omit details.) The notion of a good pair is as in the proof of Proposition 5.4. With these definitions, if y and z are distinct leaves then $(B''(y \wedge z), B'(y \wedge z))$ is good for (y, z) (i.e., Claim 3 holds). The verification is straightforward.

Claim 2. *If (X, Y) is a good pair for (y, z), if $X \subseteq L_{(a,i)}$ and $Y \subseteq L_{(b,i)} \cup L_{(c,i)}$ then $\gamma(y \wedge z) = (a, i)$.*

Proof. We prove by induction on $\ell(w)$ that $\gamma(w) = (a, i)$ for every $w \in N(X, Y)$, where X, Y is good for (y, z). If $\ell(w) = 0$ then w is a leaf, $w \in X$ and $\gamma(w) = (a, i)$. If $\ell(w) = 1$, the successors of w are leaves. Two of them must be in Y, hence have colors (b, i) or (c, i). From the way γ is defined, it follows that $\gamma(w)$ cannot be anything else than (a, i) by Claim 1. If $\ell(w) \geqslant 2$ we may have several cases, arising from the definition of a good pair.

Case 1: w has two successors in Y and we get $\gamma(w) = (a, i)$ as above.

Case 2: w has one successor t in Y and one successor t' with $t'' \in \mathbf{Suc}(t') \cap N(X, Y)$. Hence $\gamma(t)$ is (b, i) or (c, i). Say $\gamma(t) = (b, i)$. And $\gamma(t'') = (a, i)$ by induction. It follows from Claim 1(3) that $\gamma(t') \approx \gamma(t)$. Hence $\gamma(t') = (c, i)$ (it cannot be (a, i) because of $\gamma(t'') = (a, i)$, and it cannot be (b, i) either because of $\gamma(t) = (b, i)$). Hence $\gamma(w) = (a, i)$.

Case 3: w has two successors t, t' such that there exist $y \in \mathbf{Suc}(t) \cap N(X, Y)$ and $y' \in \mathbf{Suc}(t') \cap N(X, Y)$. By induction $\gamma(y) = \gamma(y') = (a, i)$. Assertion (4) of Claim 1 gives $\gamma(w) \approx \gamma(t) \approx \gamma(t')$. As in the proof of Claim 4 of the proof of Proposition 5.4, we obtain $\gamma(w) = (a, i)$. □

We obtain thus, as in Claim 4 of the proof of Proposition 5.4, that $\gamma(y \wedge z) = (a, i)$ if and only if there exists a good pair (X, Y) for (y, z) such that $X \subseteq L_{(a,i)}$ and $Y \subseteq L_{(b,i)} \cup L_{(c,i)}$. Then one finishes the proof as for Proposition 5.4. □

Theorem 5.6. *Let $n \in \mathbb{N}$, $n \geqslant 2$. The transduction*

$$\{(\langle L, R\rangle, \langle N, suc\rangle) \mid \langle L, R\rangle = \lambda(T),\ T = \langle N, suc\rangle \text{ and } T \text{ is a proper tree of outdegree at most } n\} \text{ is definable.}$$

Proof. The case $n = 2$ follows from Theorem 5.3 and Proposition 5.4 because, given $\langle L, R\rangle$ (of the appropriate form), one can MS-define a linear order on L (by Proposition 5.4) and from this order, one can define $T = \lambda^{-1}(\langle L, R\rangle)$ by Theorem 5.3. The case $n > 2$ follows similarly from Theorem 5.3 and Proposition 5.5. □

We now recall the definitions of some fragments of MS-logic on trees. A *chain variable* is a set variable, the interpretation of which is restricted to *chains*, i.e., to sets of nodes linearly ordered by the order of descendence. An *antichain variable* is, similarly, a set variable, the interpretation of which is restricted to *antichains*, i.e., to sets of pairwise incomparable nodes. We shall consider definability by formulas such as: $\forall X^a \exists Y^c \forall Z^a \varphi \ldots$, where X^a, Z^a are antichain variables and Y^c is a chain variable. It is easy to translate such a formula into an ordinary one where the set variables have no restricted range. (One replaces typically a formula $\exists Z^a \varphi$ by the formula $\exists Z$ ["Z is an antichain" $\wedge \varphi'$] where φ' translates inductively φ.) An *antichain formula*

(a *chain–antichain formula*) is a formula where all set variables are antichain variables (resp. are chain or antichain variables). We get thus the notions of *antichain* (*chain–antichain*) definability, which are restricted forms of MS-definability.

If T is a tree, locally ordered by α, we let $\lambda(\langle T, \alpha\rangle) = \langle \mathbf{L}_T, \mathbf{R}_T, \leqslant_\alpha\rangle$ where $\leqslant_\alpha$ is the lexicographical order on leaves associated with α. If K is a set of trees, either locally ordered or not, we let $\Lambda(K)$ be the set of structures $\lambda(T)$ for T in K.

Corollary 5.7. *Let K be a class of proper trees. Then*

(1) *K is MS-definable*

if and only if

(2) *$\Lambda(K)$ is $MS(\leqslant)$-definable.*

The following conditions are equivalent:

(3) *K is chain–antichain definable,*

(4) *K is antichain definable,*

(5) *$\Lambda(K)$ is MS-definable.*

If K is a class of locally ordered proper trees or of proper trees of bounded degree, then these five assertions are all equivalent.

Proof. (1) $\Rightarrow$ (2) follows from Theorem 5.3 and Proposition 1.1.

(2) $\Rightarrow$ (1) follows from Proposition 1.1 and the remark that λ is definable.

(3) $\Rightarrow$ (5): Let T be a proper tree. We first show that its chains can be represented by leaves and sets of leaves.

For every $x \in \mathbf{L}_T$ and $X \subseteq \mathbf{L}_T$, the set $C(x, X) = \{x \wedge y \mid y \in X\}$ is a chain. Every chain is of this form (because T is proper). Hence a chain variable on T can be replaced by a pair (x, X) of an object variable x and a set variable X ranging, respectively, on leaves and sets of leaves. The membership $y \wedge z \in C(x, X)$ for $y, z \in \mathbf{L}_T$ is definable in terms of x, y, z, X.

We now represent antichains similarly. If $X \subseteq \mathbf{L}_T$ we let $A(X)$ be the set of $\leqslant_T$-maximal (closest to leaves) elements of $\{x \wedge y \mid x \neq y, x, y \in X\}$. It is an antichain of internal nodes of T. Conversely, every antichain of internal nodes is of this form. An arbitrary antichain Z of T can be described as $Z = Y \cup A(X)$ for some subsets Y and X of $\mathbf{L}_T$. It follows that antichain variables on T can be replaced by pairs of set variables ranging over subsets of $\mathbf{L}_T$. This gives (3) $\Rightarrow$ (5), and (5) $\Rightarrow$ (4) is clear because $\mathbf{L}_T$ is an MS-definable antichain of T.

(4) $\Rightarrow$ (3) is trivial.

Let us finally assume that T is locally ordered or of degree at most d for d fixed. Then a linear order of $\lambda(T)$ is either given in this structure in the first case, or MS-definable by Proposition 5.5 in the second. Thus, we get (2) $\Rightarrow$ (5). Since (5) $\Rightarrow$ (2) holds in all cases, we obtain the final assertion. □

Remarks 5.8. The hypothesis that trees are proper is essential in Corollary 5.7. For arbitrary trees of bounded degree it is known from Thomas [23] that antichain

definability is weaker than MS-definability. It follows of Theorem 5.6 that antichain definability and MS-definability are equivalent for proper trees that are *either* locally ordered *or* of bounded degree. This result is proved in [20] for proper trees that are locally ordered *and* of bounded degree.

6. Modular decompositions

Every directed or undirected graph can be represented in a unique hierarchical way by means of its *modular decomposition*. The modular decomposition can be seen as an algebraic expression evaluating to the considered graph, but based on other operations than those of Section 1.

By using the results of Section 5, we prove that the transduction from an ordered graph to its modular decomposition is definable. The resulting modular decomposition *does not depend* on the linear order on the given graph. We obtain thus that every recognizable set of graphs, the modular decomposition of which uses finitely many prime graphs, is MS($\leqslant$)-definable. We shall first consider the special case of cographs.

An undirected graph is a directed graph such that every edge (x, y) has an opposite edge (y, x). We let $\mathbb{U}$ be the set of finite simple undirected loop-free graphs and $\mathbf{U}$ be the set of isomorphism classes of graphs in $\mathbb{U}$. We define two graph operations. The (*concrete*) *disjoint union* $\oplus$ transforms a pair of disjoint graphs G_1, G_2 from $\mathbb{U}$ into the graph $G_1 \oplus G_2$ which is simply their union. The (*concrete*) *product* of two disjoint graphs G_1 and G_2 in $\mathbb{U}$ is the graph $G_1 \otimes G_2$ obtained from $G_1 \oplus G_2$ by the addition of an edge between any vertex of G_1 and any vertex of G_2. The (*abstract*) *disjoint union* $\oplus$ transforms a pair of (abstract) graphs G_1, G_2 from $\mathbf{U}$ into the isomorphism class of $K_1 \oplus K_2$ where K_1 and K_2 are disjoint concrete graphs respectively isomorphic to G_1 and G_2. The abstract version of the product is defined similarly. The operations $\oplus$ and $\otimes$ are total on $\mathbf{U}$. The graphs with a unique vertex form an isomorphism class denoted by $\mathbb{1}$.

The set of (concrete) *cographs* is the least subset of $\mathbb{U}$ containing the single vertex graphs and closed under $\oplus$ and $\otimes$. (Equivalent definitions of cographs are given in [4].) An abstract cograph is thus the value of a term built with $\oplus$, $\otimes$ and the nullary symbol $\mathbb{1}$. The operations $\oplus$ and $\otimes$ are associative and commutative and two terms define the same abstract cograph if and only if they can be transformed into each other by using these algebraic laws. The operations $\oplus$ and $\otimes$ will be handled as operations of variable arity, without any relevant order on their arguments. Every abstract cograph is the value of a canonical expression called its *cotree* and that we now define.

We need the notion of module, to be used also later. Let G be a directed graph (not necessarily in $\mathbb{U}$). A *module* in G is a subset X of $\mathbf{V}_G$ such that every vertex $y \in \mathbf{V}_G$-X "sees all vertices of X in the same way". Formally, X is a module if and only if for every $y \in \mathbf{V}_G - X$:

either $(x, y) \in \mathbf{edg}_G$ for all x in X

or $(x, y) \in \mathbf{edg}_G$ for no x in X

and

either $(y,x) \in \mathbf{edg}_G$ for all $x \in X$

or $(y,x) \in \mathbf{edg}_G$ for no $x \in X$.

We say that X is a *prime module* if for every module Y:

either $X \subseteq Y$ or $Y \subseteq X$ or $X \cap Y = \emptyset$.

It follows that $\emptyset$, $\mathbf{V}_G$ and all singletons are prime modules. The *modular tree* of G is the tree $\mathbf{mt}(G) = \langle N, suc \rangle$ defined as follows:

- its set of nodes N is the set of nonempty prime modules,
- Y is a successor of X if and only if $Y \subseteq X$ and there is no prime module Z with

$$Y \subseteq Z \subseteq X \quad \text{and} \quad Y \neq Z \neq X.$$

This tree will be used below to construct the modular decomposition of a general graph G. For the moment, we go back to cographs. If G is a concrete cograph, if X is a prime module of G and if $Y_1, \ldots, Y_k$ are it successors in $\mathbf{mt}(G)$ then there are two cases:

$$\text{either } G[X] = G[Y_1] \oplus \cdots \oplus G[Y_k] \tag{1}$$

$$\text{or} \quad G[X] = G[Y_1] \otimes \cdots \otimes G[Y_k]. \tag{2}$$

(At most one of these cases holds: condition (1) implies that $G[X]$ is not connected and condition (2) that it is. If $G[X]$ is not connected, each of its connected components is a prime module of $G[X]$ and of G. We get equality (1). If it is not, one replaces G by its edge complement: it remains a cograph because the operations $\oplus$ and $\otimes$ are exchanged but modules are not changed. One then uses case (1).)

We use here the concrete versions of $\oplus$ and $\otimes$. We obtain the cotree of G by equipping $\mathbf{mt}(G)$ with the following labelling: the label of a prime module X of G is $\mathbb{1}$ if X is a leaf of $\mathbf{mt}(G)$ (equivalently: is a singleton), it is $\oplus$ if X satisfies (1) and it is $\otimes$ if X satisfies (2). We obtain in this way the cotree of G denoted by $\mathbf{cotree}(G)$. It can be seen as the tree representation of a term that evaluates to G.

Example 6.1. A cograph G is shown in Fig. 7. Its vertices are $a, b, \ldots, g$. The boxes represent the prime modules that are not singletons, and they are numbered from 1 to 4. Its modular tree is shown in Fig. 8 with its labelling. The corresponding expression is $(\mathbb{1} \oplus \mathbb{1} \oplus \mathbb{1}) \otimes (\mathbb{1} \oplus (\mathbb{1} \otimes \mathbb{1})) \otimes \mathbb{1}$.

Proposition 6.2. *The transduction from an ordered cograph to its cotree is definable.*

Proof. Let G be a cograph. Letting apart the special case where G has a single vertex, its cotree T is proper. By Theorem 5.3 we need only define in G, by an MS-formula, the ternary relation $\mathbf{R}_T$. It is clear that for any two leaves $\{x\}$ and $\{y\}$ of T (x and y are vertices of G) the node $\{x\} \wedge \{y\}$ of T is the $\subseteq$-minimal prime module containing x and y. It follows that $\mathbf{R}_T(\{x\}, \{y\}, \{z\})$ (which is equivalent to $\{x\} \wedge \{y\} \leqslant_T \{z\}$) can be expressed as:

every prime module containing x and y also contains z. (3)

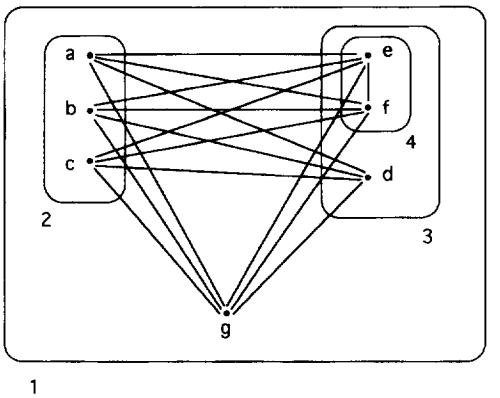

Fig. 7.

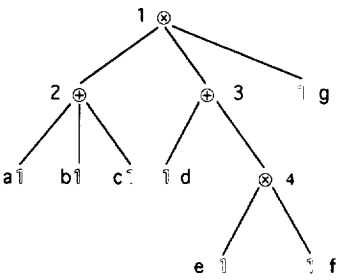

Fig. 8.

Since the notion of a prime module is expressible by an MS-formula, the property $\mathbf{R}_T(\{x\},\{y\},\{z\})$ is MS-expressible. By using Theorem 5.3, we obtain $\langle N, suc\rangle$ from $\langle \mathbf{V}_G, \mathbf{edg}_G, P\rangle$ by an MS-definable transduction, where P is an arbitrary linear order on $\mathbf{V}_G$. It remains to MS-define the labelling of the nodes of the tree $\langle N, suc\rangle$. A leaf is labelled by $\mathbb{1}$. An internal node X of the tree $\langle N, suc\rangle$ is labelled $\oplus$ if and only if X has two distinct successors Y and Z and there are two vertices y and z such that $Y \leqslant_T \{y\}$, $Z \leqslant_T \{z\}$ and y and z are not linked in G. It is labelled $\otimes$ otherwise. Provided G is a cograph, the preliminary results on cotrees recalled previously ensure that $\langle N, suc, lab\rangle$ is the cotree of G where *lab* is the above defined labelling. Since an undirected graph is a cograph if and only if it has no induced P_4, i.e., no induced path with 4 vertices, the condition that G is a cograph is MS-definable. This completes the proof. □

It follows from this result that a set of cographs K is $\{\oplus, \otimes\}$-recognizable if and only if it is MS($\leqslant$)-definable. This consequence will be proved later as a corollary of a more general result.

We now consider modular decompositions of graphs. We shall only consider the case of directed graphs. Since an undirected graph is a directed graph having the edge

(y, x) whenever it has an edge (x, y), the case of undirected graphs is just a special case of that of directed graphs. The modular decomposition is based on the *substitution* of graphs for vertices of graphs. We recall this notion (called the X-join in [22]). As in Section 1, we distinguish the set $\mathbb{G}$ of (concrete) directed graphs from the set $\mathbf{G}$ of their isomorphism classes (all of them without sources).

Let H be a graph with $\mathbf{V}_H = \{v_1, \ldots, v_k\}$. For pairwise disjoint graphs $G_1, \ldots, G_k$, we let $H\langle G_1, \ldots, G_k\rangle$ be obtained in the following way. One takes the union of $G_1, \ldots, G_k$ and one adds an edge (x, y) whenever $x \in \mathbf{V}_{G_i}$, $y \in \mathbf{V}_{G_j}$, $i \neq j$, and (v_i, v_j) is an edge of H. Hence, we get a partial k-ary operation on $\mathbb{G}$. It yields a total operation on $\mathbf{G}$ (as the other operations considered previously). We shall use in particular the operations

$$G_1 \oplus G_2 = H\langle G_1, G_2\rangle \quad \text{where } H = \{\bullet v_1 \ \ \bullet v_2\},$$

$$G_1 \otimes G_2 = H\langle G_1, G_2\rangle \quad \text{where } H = \{v_1 \bullet \rightleftarrows \bullet v_2\},$$

$$G_1 \vec{\otimes} G_2 = H\langle G_1, G_2\rangle \quad \text{where } H = \{v_1 \bullet \longrightarrow \bullet v_2\}.$$

The first two (that we have already defined for undirected graphs) are associative and commutative, the last one is only associative.

A graph G is *prime* if it cannot be written as $H\langle G_1, \ldots, G_k\rangle$ except in a trivial way with $H = G$ and G_i has a unique vertex for each i. This is equivalent to the condition that all modules are empty, singletons or equal to the set of all vertices. (The above three operations are associated with the three graphs with two vertices, which are all prime.)

The modular decomposition of G is the labelled tree $\mathbf{mdec}(G) = \langle \mathbf{mt}(G), lab\rangle$ where we define *lab* as follows. (The nodes of $\mathbf{mt}(G)$ are the nonempty prime modules of G.) For every prime module X that is not a singleton we have exactly one of the following 4 cases, where $Y_1, \ldots, Y_k$ are the successors of X in $\mathbf{mt}(G)$ (they are prime modules and can be singletons):

Case 1: $G[X] = G[Y_1] \oplus \cdots \oplus G[Y_k]$.

Case 2: $G[X] = G[Y_1] \otimes \cdots \otimes G[Y_k]$.

Case 3: $G[X] = G[Y_1] \vec{\otimes} \cdots \vec{\otimes} G[Y_k]$ (for some appropriate numbering of the successors of X, let us recall that the operation $\vec{\otimes}$ is not commutative).

Case 4: $G[X] = H\langle G[Y_1], \ldots, G[Y_k]\rangle$ for some prime graph H with at least 3 vertices; this graph H is obtained from $G[X]$ by the fusion of any two vertices in a same set Y_i, the deletion of the resulting loops and the fusion of the resulting multiple edges having the same direction.

(We use here operations on concrete graphs.) The label of X is defined by: $lab(X) := \oplus$ in Case 1, $lab(X) := \otimes$ in Case 2, $lab(X) := \vec{\otimes}$ in Case 3, and $lab(X) := H$ in Case 4.

The modular decomposition of a graph G can be seen as the tree representing a term denoting G and constructed with substitution operations. See [14] for a linear algorithm computing the modular decomposition of a directed or undirected graph. Here is an example showing that $\mathbf{mdec}(G)$ can be considered as a certain expression evaluating to G.

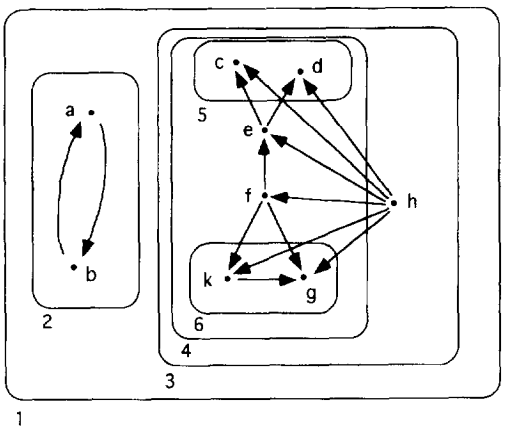

Fig. 9.

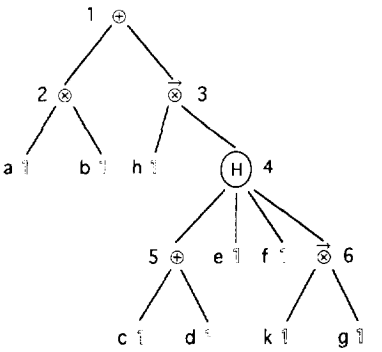

Fig. 10.

Example 6.3. Fig. 9 shows a directed graph, with vertices $a, b, \ldots, k$; the boxes show its prime modules. Fig. 10 shows its modular decomposition. The operation labelling node 4 of the modular decomposition is the substitution associated with the graph H defined as $v_1{}^{\bullet} \leftarrow v_2 \leftarrow v_3{}^{\bullet} \rightarrow {}^{\bullet}v_4$. Note the relevant ordering of the vertices of H.

Modular decompositions defined in this way can be considered as terms, built over an infinite set of operations because each prime graph is turned into a graph operation and there are infinitely many prime graphs. This is not convenient for our purposes since we want to consider all modular decompositions as relational structures using a same finite set of relation symbols. We shall redefine them as graphs by means of the notion of *graph expansion* already used in [13].

An *ε-graph* is a directed graph K, some edges of which (called the ε-edges) are labelled by ε (the other being unlabelled), and such that the subgraph of K consisting of the ε-edges is acyclic. An ε-graph K will be represented by the relational structure $\langle \mathbf{V}_K, \mathbf{edg}_K, \mathbf{edg}_K^{\varepsilon} \rangle$ where $\mathbf{edg}_K^{\varepsilon}$ represents the ε-edges and $\mathbf{edg}_K$ the other ones. A graph

G called the *expansion* of K can be associated with an ε-graph K. We let $\mathbf{V}_K$ be the set of vertices of K that are not the source of any ε-edge. We let (x, y) be an edge of G if and only if $x \neq y$ and there exists an edge (x', y') in K such that there is an ε-path in K from x' to x and one from y' to y. (An *ε-path* is a directed path consisting of ε-edges.) We denote G by $\mathbf{exp}(K)$.

For every graph G, we let $\mathbf{gdec}(G)$ (read "the **g**raph representation of the modular **dec**omposition of G") be the ε-graph K defined as follows from the modular tree $\mathbf{mt}(G)$:

- $\mathbf{V}_K$ is the set of nodes of $\mathbf{mt}(G)$.
- Its ε-edges are the edges of $\mathbf{mt}(G)$.
- For every node X of $\mathbf{mt}(G)$, i.e., every prime module of G, we put edges between the successors $Y_1, \ldots, Y_k$ of X in $\mathbf{mt}(G)$ according to the four cases considered in the definition of $\mathbf{mdec}(G)$:
  in Case 1, we put no edge (X is a $\oplus$-*node*);
  in Case 2, we put an edge from Y_i to Y_j for every $i, j \neq i$ (X is a $\otimes$-*node*);
  in Case 3 we put an edge from Y_i to Y_j for every i, j with $1 \leqslant i < j \leqslant k$ (we assume that the successors $Y_1, \ldots, Y_k$ are labelled in such a way that $G[X] = G[Y_1] \vec{\otimes} \cdots \vec{\otimes} G[Y_k]$; X is a $\vec{\otimes}$-node);
  in Case 4 we have $G[X] = H\langle G[Y_1], \ldots, G[Y_k]\rangle$ where H is prime with at least 3 vertices, and we put an edge from Y_i to Y_j if and only if there is an edge in G from a vertex of Y_i to one of Y_j and $i \neq j$ (X is an H-node).

Example (*continuation of Example* 6.3). The ε-graph $\mathbf{gdec}(G)$, where G is the graph of Example 6.3, is shown in Fig. 11. The ε-edges are light and oblique; the others are bold and horizontal.

Proposition 6.5. *For every graph G, $\mathbf{gdec}(G)$ is an ε-graph and $G = \mathbf{exp}(\mathbf{gdec}(G))$.*

Proof. Easy verification by induction on the structure of the tree $\mathbf{mt}(G)$. □

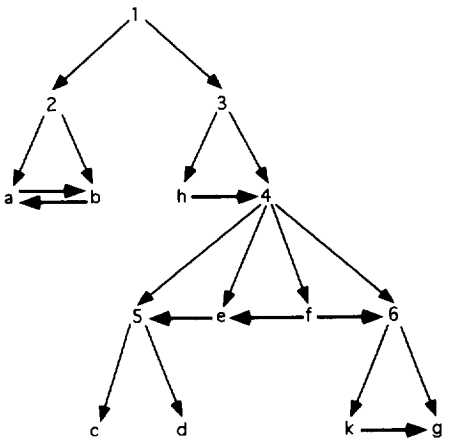

Fig. 11.

Proposition 6.6. *The transduction that associates with an ordered graph G its modular decomposition* $\mathbf{gdec}(G)$ *is definable.*

Proof. The proof is a straightforward extension of the one given for cographs (Proposition 6.2). As in this proof, we can construct the tree $T = \mathbf{mt}(G)$ by a definable transduction. More precisely, we can obtain the structure $\langle \mathbf{N}_T, \mathbf{suc}_T, \mathbf{edg}_G \rangle$ from $\langle \mathbf{V}_G, \mathbf{edg}_G, P \rangle$, where P is a linear order on $\mathbf{V}_G$, by a definable transduction. (Since the leaves of T are the vertices of G, the relation $\mathbf{edg}_G$ is a binary relation on $\mathbf{N}_T$.) The relation $\mathbf{suc}_T$ gives the ε-edges of the graph $\mathbf{gdec}(G)$. By the definition of $\mathbf{gdec}(G)$ its other edges can be defined as follows by an MS-formula over $\langle \mathbf{N}_T, \mathbf{suc}_T, \mathbf{edg}_G \rangle$ (or a first-order formula over $\langle \mathbf{N}_T, \leqslant_T, \mathbf{edg}_G \rangle$):

for $x, y \neq x$ in $\mathbf{N}_T$, there is in $\mathbf{gdec}(G)$ an edge from x to y if and only if:

(1) x and y are both successors of some node z of T, and

(2) there are leaves u, v of T with u below x and v below y such that (u, v) is an edge of G.

This completes the proof. □

Corollary 6.7. (1) *Every graph property that is expressible as an MS-property of the modular decomposition* $\mathbf{gdec}(G)$ *of the considered graph G is $MS(\leqslant)$-expressible.*

(2) *Every graph property that is expressible as a first-order property of the modular decomposition* $\mathbf{gdec}(G)$ *of the considered graph G is MS-expressible.*

Proof. (1) is an immediate consequence of Propositions 6.6 and 1.1.

(2) follows from Corollary 5.2, the remark that $\mathbf{gdec}(G)$ is first-order definable from $\langle \mathbf{N}_T, \leqslant_T, \mathbf{edg}_G \rangle$ and the fact that $\mathbf{R}_T$ is MS-definable from G. □

Let us define the *modular width* of a graph G as the maximal number of vertices of a prime graph H appearing in an H-node in the modular decomposition of G. We shall denote it by $\mathbf{mwd}(G)$. If G is prime then $\mathbf{mwd}(G) = \mathbf{card}(\mathbf{V}_G)$. The modular width of a cograph is 0. The modular width of an undirected graph is either 0 or at least 4 because the smallest prime undirected graph with at least 3 vertices is P_4 (the undirected path with 4 vertices). The modular width of a directed graph is either 0 or at least 3 (the directed graph: $\bullet \leftarrow \bullet \leftarrow \bullet$ is prime).

Proposition 6.8. *For every $n \in \mathbb{N}$, it can be expressed in MS-logic that the modular width of a graph is at most n.*

Proof. We check that the property $\mathbf{mwd}(G) \leqslant n$ is expressible as a first-order property of the ε-graph $\mathbf{gdec}(G)$. We need only express that the integer k appearing in Case 4 of the definition of edges linking the successors of any node of $\mathbf{mt}(G)$ is at most n. This means that we can separate the nodes of type 1, 2, 3 that may have more than n successors from those of type 4. Nodes of type 1 are those such that there is no edge between any two distinct successors. Nodes of type 2 are those for which there is an

edge between any two successors. The only remaining case concerns the identification of nodes of type 3 by a first-order formula. Let us first remark that a graph is linear (see Section 1) if and only if:

- it is transitive (i.e., (x, y) is an edge if (x, z) and (z, y) are edges),
- it has no pair of opposite edges,
- any two vertices are linked by an edge.

These conditions are expressible in first-order logic. And a node x of $\mathbf{mt}(G)$ is of type 3 if and only if the induced subgraph $\mathbf{gdec}(G)[X]$ of $\mathbf{gdec}(G)$, where X is the set of successors of x in $\mathbf{mt}(G)$, is linear. This can be checked by a first-order formula. □

For each $n \in \mathbb{N}$ we let PR_n be the set of prime graphs with at most n vertices and at least 3 vertices. For each $H \in PR_n$, given with a fixed enumeration $v_1, \ldots, v_k$ of its vertices ($k \leqslant n$), we define a function symbol $\mathbf{sub}_H$ intended to represent the operation that associates $H\langle G_1, \ldots, G_k\rangle$ with graphs $G_1, \ldots, G_k$. We let $\mathscr{F}_n = \{\oplus, \otimes, \vec{\otimes}, \mathbb{1}\} \cup \{\mathbf{sub}_H \mid H \in PR_n\}$ and $\mathscr{F}_\infty = \bigcup \{\mathscr{F}_n \mid n \geqslant 0\}$. The finite terms built over $\mathscr{F}_n$ (we denote their set by $\mathbf{T}(\mathscr{F}_n)$) define the graphs of modular width at most n.

For manipulating graphs of bounded modular width, such terms handled as trees are convenient. However, the definition of **mdec** as a labelled tree must be refined: the order of the successors of the $\vec{\otimes}$-nodes and of the H-nodes must be specified because the corresponding operations, namely $\vec{\otimes}$ and $\mathbf{sub}_H$, are not commutative. We shall do that by means of an integer that marks the position of a node among its brothers in these two cases. Furthermore, the operation $\vec{\otimes}$ being associative, we can forbid that a $\vec{\otimes}$-node be a first successor of a $\vec{\otimes}$-node. It follows that $\vec{\otimes}$-nodes will always have two successors. The operations $\oplus$ and $\otimes$ will be handled as associative and commutative operations and $\oplus$- and $\otimes$-nodes will have an unordered set of at least two successors (all marked by 0 to indicate that there is no relevant order among these successors).

We now define $\mathbf{mdec}'(G)$ from the labelled tree $\mathbf{mdec}(G)$. We fix $n \in \mathbb{N}$, $n \geqslant 3$, and we let $A = \mathscr{F}_n \times \{0, 1, \ldots, n\}$: this set will be used to label the nodes of $\mathbf{mdec}'(G)$.

First step: we create new $\vec{\otimes}$-nodes. Let X be a node of $\mathbf{mt}(G)$, i.e., a prime module of G. If $G[X] = G[Y_1] \vec{\otimes} \cdots \vec{\otimes} G[Y_k]$ and $k \geqslant 3$ (this corresponds to Case 3 of the definition of the labelling of $\mathbf{mdec}(G)$), we introduce new $\vec{\otimes}$-nodes $z_1, z_2, \ldots, z_{k-1}$ and we link them as follows: Y_1 is the first successor of X, and z_1 the second one; for $i = 1, \ldots, k-3$, Y_{i+1} is the first successor of Y_i and z_{i+1} is the second one; Y_{k-1} is the first successor of z_{k-1} and Y_k the second one.

Second step: For every node X of $\mathbf{mdec}(G)$ labelled by f in $\mathscr{F}_n$, we relabel it into (f, i) according to the following rules:

If X is the root or a successor of a $\oplus$- or a $\otimes$-node, we let $i = 0$.

If X is the jth successor of a $\vec{\otimes}$-node, $j = 1$ or 2, we let $i = j$.

If X is the jth successor of an H-node, $j \in \{1, \ldots, n\}$, we let $i = j$. This corresponds to Case 4 of the definition of the labelling of the father Z of X in $\mathbf{mdec}(G)$, where $G[Z] = H\langle G[Y_1], \ldots, G[Y_k]\rangle$, H is a prime graph with at least 3 vertices and $X = Y_j$.

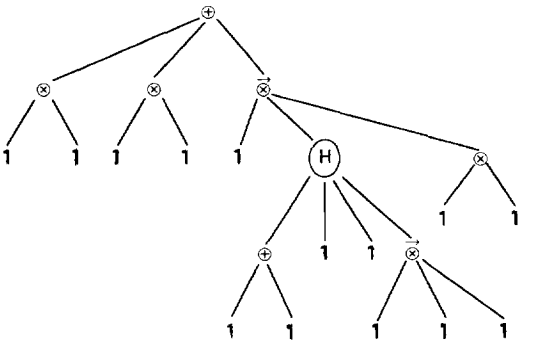

Fig. 12.

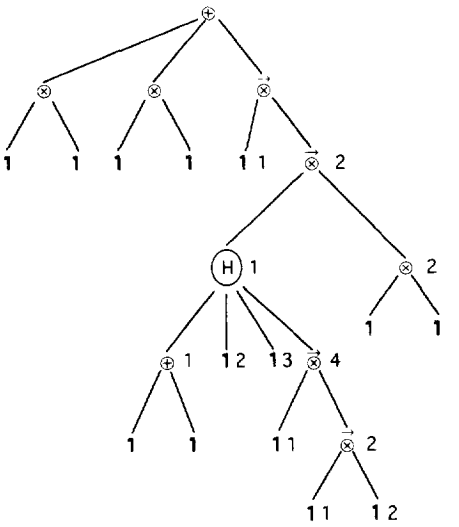

Fig. 13.

Example. Fig. 12 shows a tree **mdec**(G) and Fig. 13 shows the corresponding labelled tree **mdec**$'(G)$. In order to have a more readable drawing, the "0" 's is labels $(f,0)$ are omitted.

It is easy to see that for each n, there is a bidefinable coding (see the end of Section 4 for the definition) between the structures **gdec**(G) and **mdec**$'(G)$, for every G of modular width at most n.

Corollary 6.9. *For every $n \in \mathbb{N}$, one can construct definable transductions that associate with every ordered graph $(G, \leqslant)$ of modular width at most n the labelled tree* **mdec**$'(G)$ *and a term* $\mathbf{t}(G, \leqslant)$ *in* $\mathbf{T}(\mathcal{F}_n)$ *denoting G. The same holds for graphs G with a modular tree* **mt**(G) *of degree at most n, without any order on G.*

Proof. For $\mathbf{mdec}'(G)$ the result follows from Propositions 6.6 and 6.8.

In order to get a term in $\mathbf{T}(\mathscr{F}_n)$ denoting G from $\mathbf{mdec}'(G)$, one need only replace the $\oplus$- and $\otimes$-nodes having at least 3 successors by sequences of, respectively, $\oplus$- and $\otimes$-nodes having 2 successors, exactly as we did for the $\vec{\otimes}$-nodes in the first step of the construction of $\mathbf{mdec}'(G)$. From a given ordering of the vertices of G, one gets a linear order on the successors of any node in the tree (cf. $<_P$ in the proof of Theorem 5.3). This ordering makes it possible to transform the *set of successors* of a node into a *sequence*, as needed. The obtained term is denoted by $\mathbf{t}(G, \leqslant)$. It depends on the linear order $\leqslant$, but any two such terms associated with different linear orders define the same graph, namely G. These transformations of $\mathbf{mdec}'(G)$ into $\mathbf{t}(G, \leqslant)$ can be done by a definable transduction; hence, the transduction $\mathbf{t}$ is definable, as composition of two definable transductions (Proposition 1.1).

If G has a modular tree $\mathbf{mt}(G)$ of degree at most n, then Theorem 5.6 and the proof of Proposition 6.2 show that one can MS-define a linear order on the leaves of this tree, i.e., on the vertices of G. Then, one applies the first statement. □

A set of abstract graphs L is *$\mathscr{F}_\infty$-recognizable* if it is recognizable with respect to the $\mathscr{F}_\infty$-magma $\mathbf{G}$. Since every abstract graph is expressible as a finite combination of operations in $\mathscr{F}_\infty$ and since whenever $H = K\langle H_1, \ldots, H_n\rangle$ we have $\mathbf{sub}_H = \mathbf{sub}_K \circ (\mathbf{sub}_{H1}, \ldots, \mathbf{sub}_{Hn})$, it follows from Proposition 1.3 that L is $\mathscr{F}_\infty$-recognizable if and only if it is with respect to the operations $\mathbf{sub}_H$ associated with all not necessarily prime graphs H. Equivalently, this means that L is saturated for an equivalence relation $\sim$ on $\mathbf{G}$ having finitely many classes and such that, for all graphs H, $G_1, \ldots, G_k$, $G'_1, \ldots, G'_k$ (where H has k vertices),

$$G_1 \sim G'_1, \ldots, G_k \sim G'_k \Rightarrow H\langle G_1, \ldots, G_k\rangle \sim H(G'_1, \ldots, G'_k\rangle.$$

Proposition 6.10. *Every $MS(\leqslant)$-definable subset of $\mathbf{G}$ is $\mathscr{F}_\infty$-recognizable.*

Proof. This follows from Proposition 1.3 and Theorem 4.1 because the operations of $\mathscr{F}_\infty$ can be built as combinations of the parallel composition $\|$ and of qfd operations on the structures representing graphs. We omit details. (This extends the result of [5] saying that the CMS-definable subsets of $\mathbf{G}$ are $\mathscr{F}_\infty$-recognizable.) □

The converse does not hold because every set L of prime graphs is $\mathscr{F}_\infty$-recognizable (take the equivalence relation $H \sim H'$ if and only if H and H' are both in L or both not in L). Hence there are uncountably many $\mathscr{F}_\infty$-recognizable sets of graphs so that they cannot be characterized by the countably many formulas of any logical language using finite formulas and countably many symbols. We are thus obliged to restrict our attention to special classes of graphs (as we did in [7]) for the notion of recognizability recalled in Section 1 in order to get logical characterizations of recognizability.

If L is a set of graphs, we denote by $\mathbf{Gdec}(L)$ the set of graphs $\mathbf{gdec}(G)$ for G in L and we denote by $\mathbf{Mdec}'(L)$ the set of trees $\mathbf{mdec}'(G)$ for G in L.

Theorem 6.11. *Let $n \geqslant 0$ and L be a set of graphs of modular width $\leqslant n$. The following are equivalent:*

(1) *L is MS($\leqslant$)-definable,*
(2) *L is $\mathscr{F}_\infty$-recognizable,*
(3) *L is $\mathscr{F}_n$-recognizable,*
(4) **Gdec**(L) *is CMS-definable,*
(5) **Mdec**$'(L)$ *is CMS-definable.*

Proof. (1) $\Rightarrow$ (2) follows from Proposition 6.10.

(2) $\Rightarrow$ (3) is trivial.

(3) $\Rightarrow$ (1): Let G be a graph. By Proposition 6.8, an MS-formula θ can verify that G has modular width at most n. Assume this is the case and let $\leqslant$ be any linear order on G. By Corollary 6.9 one can MS-define $\mathbf{t}(G, \leqslant)$ from $(G, \leqslant)$. Since L is assumed $\mathscr{F}_n$-recognizable, the set L' of terms in $\mathbf{T}(\mathscr{F}_n)$ that denote elements of L is recognizable (Proposition 1.3) and hence MS-definable by Doner's Theorem (which says that a set of terms over a finite alphabet is recognizable if and only if the set of labelled trees representing its elements is MS-definable; see [24]). Finally, we get by Proposition 1.1 an MS-formula ψ expressing in $(G, \leqslant)$ that $\mathbf{t}(G, \leqslant)$ belongs to L', i.e., that G is in L. Hence, the conjunction of θ and ψ expresses that $(G, \leqslant)$ belongs to $L(\leqslant)$. Hence, L is MS($\leqslant$)-definable.

(1) $\Rightarrow$ (5): The transduction mapping **mdec**$'(G)$ to G is definable as the composition of the bidefinable coding of **mdec**$'(G)$ onto **gdec**(G) and the definable transduction **exp**. Let L be MS($\leqslant$)-definable; then, by Proposition 4.4, **Mdec**$'(L)$ is MS($\leqslant$)-definable. Hence, it is CMS-definable by Corollary 4.3, since its elements are trees.

(5) $\Rightarrow$ (4) because **mdec**$'(G)$ and **gdec**(G) are related by a bidefinable coding and by Proposition 1.1.

(4) $\Rightarrow$ (1): Let L be such that $L' =$ **Gdec**(L) is CMS-definable. Then $L(\leqslant)$ is the inverse image of the CMS-definable set L' by a definable transduction (Corollary 6.6). Hence, it is CMS-definable by Proposition 1.1. But the structures in $L(\leqslant)$ are ordered, hence a CMS-formula on them can be replaced by an MS-formula. Hence, $L(\leqslant)$ is MS-definable, and L is MS($\leqslant$)-definable. □

Corollary 6.12. *Let L be a set of graphs with modular trees of degree at most some integer n. One can MS-define a linear order on these graphs. The statements of Theorem* 6.11 *holds with MS instead of MS($\leqslant$) and CMS.*

Proof. The definability of a linear order follows from the proofs of Propositions 6.2 and 6.6 with the help of Theorem 5.6. The remaining follows from the fact that CMS and MS are equivalent for trees of bounded degree (see [6]), and that MS($\leqslant$) and MS are equivalent on linearly orderable structures. □

If L is a set of cographs, we denote by **Cotree**(L) the set of cotrees of the cographs in G in L and we recall that each cotree is a labelled tree, hence a graph, and also a member of $\mathbf{T}(\{\oplus, \otimes\})/\cong$ where $\cong$ is the congruence generated by the equational axioms expressing that $\oplus$ and $\otimes$ are associative and commutative. Note that **cotree**(G) = **mdec**$'(G)$ for every cograph G. The following result is just a special case of Theorem 6.11 and Corollary 6.12.

Corollary 6.13. *Let L be a set of cographs. The following are equivalent:*

(1) *L is $MS(\leqslant)$-definable,*

(2) *L is $\mathscr{F}_{\infty}$-recognizable,*

(3) *L is $\{\oplus, \otimes\}$-recognizable,*

(4) **Cotree**(L) *is CMS-definable.*

If L is a set of cographs with cotrees of degree at most some integer n, then these statements hold with MS instead of $MS(\leqslant)$ and CMS.

7. Summary and open questions

In Table 1 we review some answers to Question 1 of the introduction: these results concern the possibility of defining linear orders on graphs by MS-formulas.

Table 1

Class of graphs	Linear order	Topological sorting	Topological sorting from a local ordering	Minimal topological ordering
Directed and connected	**No** (Stars; see the remarks before Lemma 2.8)	Irrelevant (in general)	Irrelevant (in general)	Irrelevant (in general)
Directed, connected and of bounded degree	**Yes** (remarks after Open question 2.7)	Irrelevant (in general)	Irrelevant (in general)	Irrelevant (in general)
Cographs with cotree of bounded degree	**Yes** by Corollary 6.12	Irrelevant	Irrelevant	Irrelevant
Dags	**No** (Stars)	**No** (Stars)	**Yes** by Theorem 2.1	**?** see Open question 2.3
Trees of bounded degree, dags in $\mathscr{A}_k$	**Yes** by Corollary 2.2	**Yes** by Corollary 2.2	**Yes** by Corollary 2.2	**?** see Open question 2.3
Dags in $\mathscr{A}_k$, traces	**Yes** by Corolary 2.2	**Yes** by Corollary 2.2	**Yes** by Corollary 2.2	**Yes** by Theorem 2.4

By "traces" we mean "dependency graphs of traces" in order to simplify. In each box we give the reference to the proof, to a counterexample or to a discussion. *Stars* are trees with all leaves at distance one of the root. Stars are in bijection by a bidefinable coding with the nonempty discrete graphs. Hence they cannot be linearly ordered by CMS-formulas (by the proof of Corollary 4.3).

Table 2 reviews the relationships between the classes of MS-, CMS-, MS($\leqslant$)-definable and recognizable sets of graphs. These results give some answers to Question 2 of the introduction. By REC we mean the class of recognizable sets of graphs as defined in Section 1. By $\mathscr{F}_\infty$-REC we mean the class of $\mathscr{F}_\infty$-recognizable sets as defined in Section 6. The stars and the discrete graphs establish the strictness of the inclusions MS $\subset$ CMS of this table. The equality CMS = REC is conjectured in [7] for graphs of bounded tree-width, and proved for those of tree-width at most 2. For connected directed graphs of bounded degree and for dags in $\mathscr{A}_k$ the equalities MS = CMS = MS($\leqslant$) follow from the MS-definability of a linear order (see Table 1).

We now review some open questions that we state as conjectures:

Conjecture 7.1. *It is not possible to reconstruct a proper tree T from $\lambda(T)$ by a definable transduction.*

Conjecture 7.2. *Antichain definability is strictly weaker than MS-definability for proper trees.*

Conjecture 7.3. *CMS-definability is strictly weaker than MS($\leqslant$)-definability for general graphs.*

Table 2

Class of graphs	
All graphs	MS $\subset$ CMS $\subseteq$ MS($\leqslant$) $\subset$ REC $= ?$
Graphs of bounded tree-width	MS $\subset$ CMS $\subseteq$ MS($\leqslant$) $\subseteq$ REC $= ?$ $= ?$
Cographs, graphs of bounded modular width	MS $\subset$ CMS $\subseteq$ MS($\leqslant$) $=$ $\mathscr{F}_\infty$-REC $= ?$ by Corollaries 6.13 and 6.12
Discrete graphs, trees, graphs of tree-width at most 2	MS $\subset$ CMS = MS($\leqslant$) = REC by Corollary 4.3 and [7]
Connected directed graphs of bounded degree, dags in $\mathscr{A}_k$	MS = CMS = MS($\leqslant$) $\subset$ REC
Words, trees of bounded degree, traces, dags in $\mathscr{C}_k$	MS = CMS = MS($\leqslant$) = REC by Büchi and Doner (see [24]), Proposition 3.4 and Lemma 2.8
Cographs with cotrees of bounded degree	MS = CMS = MS($\leqslant$) = $\mathscr{F}_\infty$-REC Corollary 6.13

We suggest an example for proving it. Let $E(G)$ be the graph property saying: "G has an even number of prime modules" or, equivalently, "the modular tree $\mathbf{mt}(G)$ has an even number of nodes". It follows from Theorem 5.3 that property E is MS($\leqslant$)-expressible.

Corollary 7.4. *The property that a cograph has an even number of prime modules is not CMS-definable.*

We have the following implications:

Conjecture 7.4 $\Rightarrow$ Conjecture 7.3,

Conjecture 7.2 $\Rightarrow$ Conjecture 7.1 (see 5.10),

Conjecture 7.4 $\Rightarrow$ Conjecture 7.1 (because the property $E(G)$ is a CMS-property of $\mathbf{mt}(G)$).

Finally if CMS-definability is weaker than MS($\leqslant$)-definability for cographs, then Conjecture 7.1 also holds.

Acknowledgements

I thank A. Arnold, Y. Métivier, W. Thomas and W. Zielonka for helpful comments.

References

[1] I. Aalbersberg and G. Rozenberg, Theory of traces, *Theoret. Comput. Sci.* **60** (1988) 1–82.

[2] A. Aho, J. Hopcroft and J. Ullman, *The Design and Analysis of Computer Algorithms* (Addison-Wesley, Reading, MA, 1974).

[3] R. Cori, Y. Metivier and W. Zielonka, Asynchronous mappings and asynchronous cellular automata, *Inform. and Comput.* **106** (1993) 159–203.

[4] D. Corneil, H. Lerchs and L. Stewart, Complement reducible graphs, *Discrete Appl. Math.* **3** (1981) 163–174.

[5] B. Courcelle, An axiomatic definition of context-free rewriting and its application to NLC graph grammars, *Theoret. Comput. Sci.* **55** (1987) 141–181.

[6] B. Courcelle, The monadic second-order logic of graphs I: recognizable sets of finite graphs. *Inform. and Comput.* **85** (1990) 12–75.

[7] B. Courcelle, The monadic second-order logic of graphs V: on closing the gap between definability and recognizability, *Theoret. Comput. Sci.* **80** (1991) 153–202.

[8] B. Courcelle, The monadic second order logic of graphs VII: graphs as relational structures, *Theoret. Comput. Sci.* **101** (1992) 3–33.

[9] B. Courcelle, The monadic second-order logic of graphs VIII: orientations, *Ann. Pure Appl. Logic* **72** (1995) 103–143.

[10] B. Courcelle, The monadic second-order logic of graphs VI: on several representations of graphs by relational structures, *Discrete Appl. Math.* **54** (1994) 117–149.

[11] B. Courcelle, Monadic second-order definable graph transductions: a survey, *Theoret. Comput. Sci.* **126** (1994) 53–75.

[12] B. Courcelle, Recognizable sets of graphs: equivalent definitions and closure properties, *Math. Struct. Comput. Sci.* **4** (1994) 1–32.

[13] B. Courcelle, Structural properties of context-free sets of graphs generated by vertex-replacement, *Inform. and Comput.* **116** (1995) 275–293.

[14] A. Cournier and M. Habib, A new linear algorithm for modular decomposition, in: *Proc. CAAP '94*, Lecture Notes in Computer Science, Vol. 787 (Springer, Berlin, 1994) 68–94.
[15] W. Ebinger and A. Muscholl, Logical definability of infinite traces, in: *Proc. ICALP '93*, Lecture Notes in Computer Science, Vol. 700 (Springer, Berlin, 1993) 335–346.
[16] F. Gecseg and M. Steinby, *Tree Automata* (Academiai Kiado, Budapest, 1984).
[17] L. Hella, P. Kolaitis and K. Luosto, How to define a linear order on finite models, in: *Proc. IEEE Symp. Logic in Computer Science* (1994).
[18] H. Hoogeboom and P. ten Pas, Recognizable text languages, in: *MFCS 1994*, Lecture Notes in Computer Science, Vol. 841 (Springer, Berlin, 1994) 413–422.
[19] E. Ochmanski, Regular behaviour of concurrent systems, *Bull. EATCS* **27** (1985) 56–67.
[20] A. Potthof and W. Thomas, Regular tree languages without unary symbols are star-free, in: *Proc. FCT '93*, Lecture Notes in Computer Science, Vol. 710 (Springer, Berlin, 1993) 396–405.
[21] A. Proskurowski, Separating subgraphs in k-trees: cables and caterpillars, *Discrete Math.* **49** (1984) 275–285.
[22] D. Sumner, Graphs indecomposable with respect to the X-join, *Discrete Math.* **6** (1973) 281–298.
[23] W. Thomas, Logical aspects of the study of tree languages, in: B. Courcelle, ed., *Proc. 9th Colloq. on Trees in Algebra and Programming (CAAP)* (Cambridge Univ. Press, Cambridge, 1984) 31–49.
[24] W. Thomas, Automata on infinite objects, in: J. Van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, Amsterdam, 1990) 133–192.
[25] W. Thomas, On logical definability of trace languages, in: V. Diekert, ed., Proceedings of a workshop held in Kochel in October 1989, *Report of Technische Universität München I-9002* (1990) 172–182.