THE EXPRESSION OF GRAPH PROPERTIES AND GRAPH TRANSFORMATIONS IN MONADIC SECOND-ORDER LOGIC

B. COURCELLE
LABRI (URA CNRS 1304), Bordeaux I University
351, Cours de la Libération, 33405 Talence - France
e-mail: courcell@labri.u-bordeaux.fr

By considering graphs as logical structures, one can express formally their properties by logical formulas. We review the use of monadic second-order logic for expressing graph properties, and also, graph transformations. We review the intimate relationships of monadic second-order logic and context-free graph grammars. We also discuss the definition of classes of graphs by forbidden configurations.

Introduction

By considering graphs as logical structures, one can express formally their properties by logical formulas. One can thus describe classes of graphs by formulas of appropriate logical languages expressing characteristic properties. There are two main motivations for doing this : the first one, originating from the work by Fagin [39], consists in giving logical characterizations of complexity classes ; the second one consists in using logical formulas as finite devices, comparable to grammars or automata, to specify classes of graphs and to establish properties of such classes from their logical descriptions.

We shall only consider here the second of these motivations. The ideal language is in this respect monadic second-order logic, as we shall demonstrate. It is crucial for establishing "easily" results like this one :

the set of planar graphs belonging to a HR set of graphs[a] is HR,

or this one :

the set of Hamiltonian graphs belonging to a HR set of graphs is HR,

by essentially the same proof, using the fact that planarity and Hamiltonicity can both be described by MS (Monadic Second-order) formulas. These two results do not concern logic, but their proofs use logic as a tool. The deep reason why MS logic is so crucial is that it replaces for graphs (and for the development of the theory of context-free graph grammars) the notion of a finite automaton which is very important in the theory of formal languages. It "replaces" because no convenient notion of finite automaton is known for graphs. The notion of a transformation from words or trees to words or trees is also essential in language theory (Berstel [4], Raoult [54]). These transformations are usually defined in terms of finite automata, that produce an output while traversing the given word or tree. Since we have

[a] A HR set of graphs is a set of finite graphs generated by a Hyperedge Replacement graph grammar

no notion of finite graph automaton, we cannot define graph transformations in terms of automata. However, we can define such transformations in terms of MS-formulas. We call them *definable transductions*. "Definable" refers to logic and "transduction" to the way transformations of words and trees are usually named.

MS logic is thus an essential notion in the extension of formal language theory to graphs, hypergraphs and related structures. Another important notion is that of a graph (or hypergraph) operation. By using it, one can define context-free sets of graphs as components of least solutions of systems of equations (without using any graph rewriting rule) and recognizable sets of graphs (without using any notion of graph automaton). Context-free and recognizable sets can thus be defined and investigated in the general framework of Universal Algebra. They instanciate immediately to graphs and hypergraphs of all kinds as soon as appropriate operations on these structures are defined. Furthermore the notion of recognizability establishes the link between Logic and Universal Algebra because every MS-definable set of graphs is recognizable : this result extends the result by Büchi saying that every MS-definable set of finite words is as regular language.

This chapter is organized as follows. Section 1 reviews relational structures, first-order logic, second-order logic and monadic second-order logic. Some basic lemmas helping in the construction of logical formulas are proved. Section 2 introduces several possible representations of graphs, hypergraphs and partial orders by relational structures, and discusses how the choice of a representation affects the expressive power of the three considered logical languages. Section 3 discusses the expressibility in MS logic of the finiteness of a set and of the parity of its cardinality when it is finite. Section 4 introduces definable transductions, reviews their basic properties and their application to the comparison of the different relational structures representing partial orders, graphs and hypergraphs reviewed in Section 2. Section 5 defines the hyperedge replacement (HR) hypergraph grammars and the vertex replacement (VR) graph grammars, in terms of systems of recursive set equations, by means of appropriate operations on hypergraphs and graphs respectively. Logical characterizations of the HR sets of hypergraphs and of the VR sets of graphs in terms of definable transductions are also given. These characterizations yield the stability of the corresponding classes under the relevant definable transductions. Section 6 introduces the recognizability of sets of graphs and hypergraphs. This notion is based on finite congruences relative to the operations on graphs and hypergraphs introduced in Section 5. In general, the intersection of an equational and a recognizable set is an equational set. (An instance of this result is the fact that the intersection of a context-free language and a regular one is context-free). The major result of this section says that a monadic second-order definable set of graphs or hypergraphs is recognizable. This result yields in particular the two results mentioned at the beginning of the introduction concerning HR sets of graphs, planarity and Hamiltonicity. Section 7 deals with the logical aspect of the definition of sets of graphs by forbidden minors. Kuratowski's Theorem stating that a graph is planar iff it does not contain any of the two graphs K_5 and $K_{3,3}$ as a minor is a well-known example of such a definition. By using deep results by Robertson and Seymour, we relate definitions by forbidden minors with definitions by MS formulas and/or by HR grammars.

1 Relational structures and logical languages

In order to express graph properties by logical formulas, we shall represent graphs by relational structures, i.e., by logical structures with relations only (without functions). We shall review first-order logic, second-order logic and monadic second-order logic, which is an extension of the former and a fragment of the latter. We shall review some basic tools that will help in the construction of formulas in forthcoming sections.

1.1 Structures

Let $\mathcal{R}$ be a finite set of relation symbols. Each symbol $R \in \mathcal{R}$ has an associated positive integer called its arity, denoted by $\rho(R)$. An $\mathcal{R}$-structure is a tuple $S =< D_S, (R_S)_{R\in\mathcal{R}} >$ such that D_S is a possibly empty set called the *domain of* S and each R_S is a $\rho(R)$-ary relation on D_S, i.e., a subset of $D_S^{\rho(R)}$. We shall say that "$R(d_1, \cdots, d_n)$ holds in S" iff $(d_1, \cdots, d_n) \in R_S$, where, of course, $d_1, \cdots, d_n \in D_S$. We shall denote by $STR(\mathcal{R})$ the class of $\mathcal{R}$-structures. Structures may have infinite domains.

We give two examples of the use of structures. A word u in A^* is usually defined as a sequence of letters from A, equivalently as a mapping $\{1, \cdots, n\} \to A$ for some $n \in \mathcal{N}$ (with $n = 0$ for the empty word). In order to represents words by relational structures we let $\mathcal{R}_{w,A} := \{suc, lab_a, \cdots, lab_d\}$ where $A = \{a, \cdots, d\}$ (A can have more than 4 letters), suc is binary and $lab_a, \cdots, lab_d$ are unary. For every word $u \in A^*$, we let $\| u \| \in STR(\mathcal{R}_{w,A})$ be the structure S such that :

$D_S = \emptyset$ if u is the empty word ε ; otherwise $D_S = \{1, \cdots, n\}$ if u has length n;

$suc_S = \{(1, 2), (2, 3), \cdots, (n - 1, n)\}$;

$i \in lab_{yS}$ iff y is the i-th letter of u.

In order to represent graphs by relational structures, we let $\mathcal{R}_s = \{edg\}$ where edg is binary. With a directed graph G, we associate the $\mathcal{R}_s$-structure $\mid G \mid_1 = < V_G, edg_G >$ where V_G is the set of vertices of G (and the domain of $\mid G \mid_1$) and $(x, y) \in edg_G$ iff there is in G an edge from x to y. We do the same if G is undirected and we let $(x, y) \in edg_G$ iff there is in G an edge linking x and y : it follows that edg_G is in this case symmetric. The structure $\mid G \mid_1$ does not contain information concerning the multiplicity of edges. It is thus appropriate to represent simple graphs only because two simple graphs G and G' are isomorphic iff the structures $\mid G \mid_1$ and $\mid G' \mid_1$ are isomorphic. (In Section 2, we shall define another representation, denoted by $\mid G \mid_2$, which will be appropriate for graphs with multiple edges). For directed graphs, we shall sometimes use suc instead of edg. We call y a *successor* of x if $(x, y) \in suc_G$.

Relational structures form the basis of the theory of relational databases (Abiteboul et al. [1]). More precisely, a finite $\mathcal{R}$-structure S can be considered as a state of a relational database describing relations between the objects of D_S. The relations R_S for $R \in \mathcal{R}$ are

the various relations of the database. A finite structure S is represented by means of some coding of the objects of D_S and for each $R \in \mathcal{R}$ a list of $\rho(R)$-tuples of "codes" of the elements of D_S. The study of query languages for relational databases has also been a motivation for finite model theory. (See the survey by Kannellakis [48] and the book by Abiteboul et al. [1]).

1.2 First-order logic

We let $\mathcal{X}$ be a countable alphabet of lowercase letters called *individual variables*. Let $\mathcal{R}$ be a finite set of relation symbols. The atomic formulas are $x = y, R(x_1, \cdots, x_n)$ for $x, y, x_1, \cdots, x_n \in \mathcal{X}$, $R \in \mathcal{R}$, $n = \rho(R)$. The first-order formulas are formed from atomic formulas with the propositional connectives $\wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$, and quantifications $\exists x, \forall x$ (for $x \in \mathcal{X}$).

We shall denote by $FO(\mathcal{R}, \mathcal{Y})$ where $\mathcal{Y} \subseteq \mathcal{X}$ the set of first-order formulas over $\mathcal{R}$ with free variables in $\mathcal{Y}$. In order to specify the free variables that may occur in a formula φ we shall write it $\varphi(x_1, \cdots, x_n)$ if $\varphi \in FO(\mathcal{R}, \{x_1, \cdots, x_n\})$. (Some variables in $\{x_1, \cdots, x_n\}$ may have no free occurrence in φ). If $\varphi(x_1, \cdots, x_n)$ is a first-order formula over $\mathcal{R}$, if $S \in STR(\mathcal{R})$, and if $d_1, \cdots, d_n \in D_S$, we write $S \models \varphi(d_1, \cdots, d_n)$ or $(S, d_1, \cdots, d_n) \models \varphi$ to mean that φ is true in S if x_i is given the value d_i for $i = 1, \cdots, n$. If φ has no free variables, i.e., if it is *closed*, then it describes a property of S and not of tuples of elements of S. Here is an example. The formula $\varphi(x)$ over $\mathcal{R}_s$ defined as :

$$\forall y_1, y_2, y_3[edg(x, y_1) \wedge edg(x, y_2) \wedge edg(x, y_3) \Rightarrow y_1 = y_2 \vee y_1 = y_3 \vee y_2 = y_3]$$

expresses that the vertex x of the represented graph has out-degree at most 2. The closed formula $\forall x.\varphi(x)$ expresses thus that the considered graph has outdegree at most 2.

We now give an example concerning words. We let $A = \{a, b, c\}$. The formula θ below is constructed in such a way that, for every word $u \in A^*$:

$$\| u \| \models \theta \text{ iff } u \in ab^*c.$$

Here is θ :

$$\exists x[lab_a(x) \wedge \forall y(\neg suc(y, x))] \wedge \forall x[lab_a(x) \Rightarrow \exists y(suc(x, y) \wedge (lab_b(y) \vee lab_c(y)))]$$

$$\wedge \forall x[lab_b(x) \Rightarrow \exists y(suc(x, y) \wedge (lab_b(y) \vee lab_c(y)))] \wedge \forall x[lab_c(x) \Rightarrow \forall y(\neg suc(x, y))].$$

Let us remark that θ has models that are not representations of words. So, θ characterizes ab^*c as a subset of A^* ; it does not characterize the structures representing the words of ab^*c among those of $STR(\mathcal{R}_{w,A})$. The languages characterized in this way by a first-order formula form a subclass of the class of regular languages, called the class of locally threshold testable languages (see Thomas [63] or the survey by Pin [52]).

1.3 Second-order logic

We now let $\mathcal{X}$ contain individual variables as in Subsection 1.2 and also *relation variables*, denoted by uppercase letters, $X, Y, X_1, \cdots, X_n$. Each relation variable has an arity which is a positive integer ($\rho(X)$ is the arity of X). We let $\mathcal{R}$ be a finite set of relation symbols and we now define the second-order formulas over $\mathcal{R}$. The atomic formulas are : $x = y, R(x_1, \cdots, x_n), X(x_1, \cdots, x_n)$, where $x, y, x_1, \cdots, x_n, X \in \mathcal{X}$, $R \in \mathcal{R}$ and $n = \rho(R) = \rho(X)$. The formulas are constructed from the atomic formulas with the propositional connectives (as in Subsection 1.2) and the quantifications $\exists x, \forall x, \exists X, \forall X$ over individual and relation variables. (We do not give a formal syntax ; see the examples below). We shall denote by $SO(\mathcal{R}, \mathcal{Y})$ the set of second-order formulas over $\mathcal{R}$ with free variables in $\mathcal{Y}$. The notation $\varphi(x, y, z, X_1, \cdots, X_n)$ indicates that the free variables of φ belong to $\{x, y, z, X_1, \cdots, X_n\}$.

Consider a formula $\varphi(x_1, \cdots, x_m, X_1, \cdots, X_n)$. If $S \in STR(\mathcal{R})$ if $d_1, \cdots, d_m \in D_S$, if $E_1, \cdots, E_n$ are relations on D_S of respective arities $\rho(X_1), \cdots, \rho(X_n)$ then the notation $S \models \varphi(d_1, \cdots, d_m, E_1, \cdots, E_n)$ means that φ is true in S for the values $d_1, \cdots, d_m$ of $x_1, \cdots, x_m$ and $E_1, \cdots, E_n$ of $X_1, \cdots, X_n$ respectively. Let $\mathcal{Y} = \{x_1, \cdots, x_m, X_1, \cdots, X_n\}$. A *$\mathcal{Y}$-assignment* in S is a mapping γ with domain $\mathcal{Y}$ such that $\gamma(x_i) \in D_S$ for $i = 1, \cdots, m$ and $\gamma(X_j)$ is a $\rho(X_j)$-ary relation on D_S for each $j = 1, \cdots, n$. We shall also use the notation $(S, \gamma) \models \varphi$ instead of $S \models \varphi(d_1, \cdots, d_m, E_1, \cdots, E_n)$ where $\gamma(x_i) = d_i$ and $\gamma(X_j) = E_j$.

We now give some examples. Let $\beta(X)$ be the formula in $SO(\emptyset, \{X\})$:

$$\forall x, y, z[X(x, y) \wedge X(x, z) \Longrightarrow y = z] \wedge \forall x, y, z[X(x, z) \wedge X(y, z) \Longrightarrow x = y],$$

where X is binary. It expresses that X is a functional relation, and that the corresponding function is injective on its domain. (This domain may be strictly included in the domain of the considered structure). Hence the following formula $\alpha(X)$ of $SO(\mathcal{R}_s, \{X\})$ expresses that X is an automorphism of a simple graph G represented by the structure $|G|_1$ in $STR(\mathcal{R}_s)$. Here is $\alpha(X)$:

$$\beta(X) \wedge \forall x \exists y X(x, y) \wedge \forall x \exists y X(y, x)$$

$$\wedge \forall x, y, x', y'[edg(x, y) \wedge ((X(x, x') \wedge X(y, y')) \vee (X(x', x) \wedge X(y', y))) \Longrightarrow edg(x', y')].$$

Hence, G has a nontrivial automorphism iff $|G|_1$ satisfies the closed formula

$$\exists X[\alpha(X) \wedge \exists x, y(\neg x = y \wedge X(x, y))].$$

Consider now the formula $\gamma(Y_1, Y_2) \in SO(\emptyset, \{Y_1, Y_2\})$:

$$\exists X[\beta(X) \wedge \forall x(Y_1(x) \Longleftrightarrow \exists y X(x, y)) \wedge \forall x(Y_2(x) \Longleftrightarrow \exists y X(y, x))].$$

It expresses that there exists a bijection (namely the binary relation defined by X) between the sets Y_1, Y_2 (handled as unary relations). One can thus characterize by the following second-order formula the nonregular language $\{a^n b^n / n \geq 1\}$:

$\exists x[lab_a(x) \wedge \forall y(\neg suc(y,x))] \wedge \forall x[lab_a(x) \Longrightarrow \exists y(suc(x,y) \wedge (lab_a(y) \vee lab_b(y)))]$

$\wedge \forall x,y[lab_b(x) \wedge suc(x,y) \Longrightarrow lab_b(y)] \wedge$

$\exists Y_1, Y_2[\gamma(Y_1,Y_2) \wedge \forall x[lab_a(x) \Longleftrightarrow Y_1(x)] \wedge \forall x[lab_b(x) \Longleftrightarrow Y_2(x)]].$

The main logical language to be used in this paper is monadic second-order logic which lies inbetween first-order and second-order logic.

1.4 Monadic second-order logic

Let $\mathcal{R}$ be a finite set of relation symbols. Let $\mathcal{X}$ contain individual variables and relation variables of arity one. Since a relation with one argument is nothing but a set, we shall call these variables *set variables*. A monadic second-order formula over $\mathcal{R}$ is a second-order formula written with $\mathcal{R}$ and $\mathcal{X}$: the quantified and free variables are individual or sets variables ; there is no restriction on the arity of the symbols in $\mathcal{R}$. In order to get more readable formulas, we shall write $x \in X$ instead of $X(x)$ where X is a set variable. We denote by $MS(\mathcal{R},\mathcal{Y})$ the set of monadic second-order formulas (MS formulas for short) over $\mathcal{R}$, with free variables in $\mathcal{Y}$.

We give some examples. The MS formula $\delta \in MS(\{suc\},\emptyset)$ below expresses that a word in A^* has an odd length. Since this property does not depend on the letters, the formula δ does not use the relations $lab_x, x \in A$. Here is δ :

$$\exists X,Y[\delta'(X,Y) \wedge \exists x(x \in X \wedge \forall y(\neg suc(x,y)))]$$

where $\delta'(X,Y)$ is

$$\exists x(x \in X \wedge \forall y(\neg suc(y,x))) \wedge \forall x,y(x \in X \wedge suc(x,y) \Longrightarrow y \in Y) \wedge$$
$$\forall x,y(x \in Y \wedge suc(x,y) \Longrightarrow y \in X).$$

For every word $w \in A^*$, there is a unique pair X,Y with $X,Y \subseteq D_{\|w\|}$ satisfying δ': the elements of X are the odd rank positions in w and the elements of Y are the even rank ones. The formula δ expresses that the last position is odd, i.e., that the considered word has odd length. A theorem by Büchi and Elgot [8], [31] (see Thomas [62], Thm 3.2) says that the languages defined by MS formulas are exactly the regular languages.

We now consider an example concerning graphs. The following formula $\tau_3 \in MS(\{\mathcal{R}_s\})$ expresses that the considered graph is 3-vertex colorable (with every two adjacent vertices of different colors ; one may have loops). Here is τ_3 :

$\exists X_1, X_2, X_3[\forall x(x \in X_1 \vee x \in X_2 \vee x \in X_3) \wedge \forall x(x \in X_1 \Longrightarrow \neg x \in X_2 \wedge \neg x \in X_3)$

$\wedge \forall x(x \in X_2 \Longrightarrow \neg x \in X_1 \wedge \neg x \in X_3) \wedge \forall x(x \in X_3 \Longrightarrow \neg x \in X_1 \wedge \neg x \in X_2)$

$\wedge \forall x,y(edg(x,y) \wedge \neg x = y$

$\Longrightarrow \neg(x \in X_1 \wedge y \in X_1) \wedge \neg(x \in X_2 \wedge y \in X_2) \wedge \neg(x \in X_3 \wedge y \in X_3))]$

A triple of sets X_1, X_2, X_3 satisfying this condition is a partition of the set of vertices. Considering i such that $x \in X_i$ as the color of x, the formula expresses that adjacent vertices have different colors. For each positive integer k, one can write a similar formula τ_k expressing that the considered graph is k-vertex colorable.

Let L be any of the languages FO, MS and SO. We write $L(\mathcal{R})$ for $L(\mathcal{R}, \emptyset)$. A property P of structures in $STR(\mathcal{R})$ is $L - expressible$ iff there exists a formula φ in $L(\mathcal{R})$ such that, for every structure $S \in STR(\mathcal{R}), P(S)$ holds iff $S \models \varphi$. A class of structures $\mathcal{C}$ is $L - definable$ iff it is the class of structures satisfying an $L - definable\ property$. These definitions can be relativized to specific classes of structures. Let $\mathcal{T} \subseteq STR(\mathcal{R})$. Then, a property P *of structures in* $\mathcal{T}$ is $L-expressible$ iff there exists a formula φ in $L(\mathcal{R})$ such that, for every S in $\mathcal{T}$, $P(S)$ holds iff $S \models \varphi$. One obtains similarly the notion of a subclass $\mathcal{C}$ of $\mathcal{T}$ that is $L - definable\ with\ respect\ to\ \mathcal{T}$. A property P of tuples $(d_1, \cdots, d_n, E_1, \cdots, E_m)$ in the structures S of a class $\mathcal{T}$ is $L - expressible$ iff there exists a formula $\varphi \in L(\mathcal{R}, \{x_1, \cdots, x_n, X_1, \cdots, X_m\})$ such that for all $S \in \mathcal{T}$, all $(d_1, \cdots, d_n, E_1, \cdots, E_m)$ of appropriate type, $P(S, d_1, \cdots, E_m)$ holds iff $S \models \varphi(d_1, ..., E_m)$.

One obtains thus a hierarchy of graph properties linked with the hierarchy of languages: $FO \subset MS \subset SO$. In Sections 2 and 3, we shall introduce intermediate languages between MS and SO and discuss the strictness of the corresponding hierarchy of graph properties, relativized to various classes of graphs or related objects like words, partial orders and hypergraphs. The languages defined by monadic second-order formulas are the regular languages by a theorem of [8] (see also the survey by Pin [52]).

1.5 Decidability questions

Let $\mathcal{R}$ be a finite set of relation symbols. Let $\mathcal{L}$ be a class of closed formulas expressing properties of the structures in $STR(\mathcal{R})$. Let $\mathcal{C} \subseteq STR(\mathcal{R})$. The $\mathcal{L} - theory\ of\ \mathcal{C}$ is the set :

$$Th_{\mathcal{L}}(\mathcal{C}) := \{\varphi \in \mathcal{L} / S \models \varphi\ for\ every\ S \in \mathcal{C}\}.$$

We say that the $\mathcal{L}$-theory of $\mathcal{C}$ is *decidable* if this set is recursive. The *$\mathcal{L}$-satisfiability problem for $\mathcal{C}$* is the problem of deciding whether a given formula belongs to the set

$$Sat_{\mathcal{L}}(\mathcal{C}) := \{\varphi \in \mathcal{L} /\ \ S \models \varphi\ for\ some\ S \in \mathcal{C}\}.$$

If $\mathcal{L}$ is closed under negation, the $\mathcal{L}$-theory of $\mathcal{C}$ is decidable iff its $\mathcal{L}$-satisfiability problem is decidable. We shall consider some conditions on a class of graphs $\mathcal{C}$ ensuring that its *monadic* theory (i.e., the MS-theory) is decidable.

If $\mathcal{C} = \{S\}$ where S is a finite (and effectively given) structure then the property $S \models \varphi$ is decidable for every formula φ of the languages $\mathcal{L}$ we have considered, or we will consider. The $\mathcal{L}$-theory of $\mathcal{C}$ is thus (trivially) decidable. An interesting question is thus the complexity of this problem. For an example, the property $S \models \varphi$ is polynomial in *size*(S) if φ is a fixed first-order formula and S is the input ; it is NP if φ is existential second-order

(Fagin [39]), where again φ is fixed and S is the input. Conversely, every NP property can be described in this way. More generally, a property belongs to the polynomial hierarchy iff it can be described by a second-order formula (see Stockmeyer [60], Immerman [46] or Abiteboul et al. [1]).

If $\mathcal{C} = \{S\}$ where S is infinite, then the $\mathcal{L}$-theory of $\mathcal{C}$ may be undecidable. However the study of the theories of infinite structures and especially of their monadic (second-order) theories is a very rich topic that we cannot even touch here. We refer the reader to survey papers like those of Gurevich [44], Courcelle [10], Thomas [62].

Proposition 1.1 (Trakhtenbrot [64]) *: The first-order theory of the class of finite graphs is undecidable.*

So are a fortiori its monadic and its second-order theories. The following result is a basic tool for obtaining undecidability results. It concerns square grids. A *grid* is a directed graph isomorphic to $G_{n,m}$ for some $n, m \in \mathcal{N}_+$ where $V_{G_{n,m}} = [n] \times [m]$ and $E_{G_{n,m}} = \{((i,j),(i',j'))/1 \leq i \leq i' \leq n, 1 \leq j \leq j' \leq m$ and, either $i' = i+1$ and $j' = j$, or $i' = i$ and $j' = j+1\}$. ($\mathcal{N}_+$ denotes the set of positive integers and for $n \in \mathcal{N}_+$; $[n]$ denotes $\{1, 2, \cdots, n\}$). A *square grid* is a graph of the above form where $m = n$.

Proposition 1.2 *The monadic (second-order) satisfiability problem of every class of graphs $\mathcal{C}$ containing graphs isomorphic to $G_{n,n}$ for infinitely many integers n is undecidable.*

Proof sketch : One first constructs an MS formula γ that defines the square grids among the finite simple loop-free directed graphs (see Subsection 1.9). Consider now a Turing machine M with total alphabet A (input letters, states, end markers) and an initial configuration w_M. Let $A = \{a_1, \cdots, a_m\}$. If $(X_1, \cdots, X_m)$ is a partition of the set of vertices of a square grid $G_{n,n}$ then one can consider that this partition defines a sequence of n words of length n in A^*. (Each line of the grid represents a word where a line of $G_{n,n}$ is a subgraph with set of vertices $\{i\} \times [n]$ for some i ; the first line represents the configuration w_M). One can construct an MS-formula $\varphi_M(X_1, \cdots, X_m)$ such that, for every $X_1, \cdots, X_m \subseteq V_{G_{n,n}}$

$$G_{n,n} \models \varphi_M(X_1, \cdots, X_m)$$

iff $(X_1, \cdots, X_m)$ is a partition of $V_{G_{n,n}}$ which defines the sequence of configurations of a terminating computation of M. (Configurations of length smaller than n can be extended to the right by a special symbol). It follows that if $\mathcal{C}$ contains infinitely many square grids, then M terminates iff some graph G in $\mathcal{C}$ satisfies the formula

$$\gamma \wedge \exists X_1, \cdots, X_m . \varphi_M \tag{1}$$

Hence the halting problem of Turing machine reduces to the monadic satisfiability problem of any class $\mathcal{C}$ containing infinitely many square grids. This latter problem is thus undecidable. □

Remark : In this construction, one may assume that the machine M is deterministic and one can write φ_M in such a way that the square grid on which its (unique) computation is encoded is minimal. It follows that there exists at most one graph satisfying formula (1). Hence, even if we know that a MS-formula φ has only finitely many finite models (up to isomorphism), we cannot construct this set by an algorithm taking φ as input.

1.6 Some tools for constructing formulas

Two formulas $\varphi, \varphi' \in SO(\mathcal{R}, \{x_1, \cdots, x_n, X_1, \cdots, X_m\})$ are *equivalent* if for every $S \in STR(\mathcal{R})$, for all $d_1, \cdots, d_n \in D_S$, for all relations $E_1, \cdots, E_m$ on D_S of respective arities $\rho(X_1), \cdots, \rho(X_m)$ we have :

$$S \models \varphi(d_1, \cdots, d_n, E_1, \cdots, E_m) \text{ iff } S \models \varphi'(d_1, \cdots, d_n, E_1, \cdots, E_m).$$

Clearly, two equivalent formulas express the same properties of the relevant tuples $(d_1, \cdots, d_n, E_1, \cdots, E_m)$ in every structure $S \in STR(\mathcal{R})$.

In some cases equivalence is relativized to a subclass $\mathcal{S}$ of $STR(\mathcal{R})$: equivalent formulas express the same properties of the relevant tuples in the structures of $\mathcal{S}$, not in all structures.

Lemma 1.3 *Let $\varphi \in SO(\mathcal{R}, \mathcal{Y})$ and $\mathcal{Z}$ be a finite set of variables. One can transform φ into an equivalent formula φ' in $SO(\mathcal{R}, \mathcal{Y})$ in which no variable of $\mathcal{Z}$ is bound. If φ is MS, then φ' is MS ; if φ is FO, then φ' is FO.*

Proof : One simply renames the bound variables belonging to $\mathcal{Z}$. Of course, one renames a variable into one of same type (individual or relational) and arity (in case of a relation variable). This is possible because our "universal" set of variables contains countably many variables of each type and arity. □

For readibility, one usually writes a formula by choosing bound variables that are not free in the formula. This is always possible without loss of generality.

We now recall the definition of first-order substitutions in formulas. Let φ be a second-order formula ; let $x_1, \cdots, x_n$ be pairwise distinct variables ; let $y_1, \cdots, y_n$ be variables, not necessarily pairwise distinct. (In order to avoid double subscripts, we do not index the variable in $\mathcal{X}$; hence $x_1, \cdots, x_n$ are metavariables denoting variables of $\mathcal{X}$; they are not *the* variables $x_1, \cdots, x_n$; in the generic such list, we may have two variables equal). We denote by $\varphi[y_1/x_1, \cdots, y_n/x_n]$ the formula obtained as follows :

- using Lemma 1.3, one takes φ' equivalent to φ with no bound variable in $\{y_1, \cdots, y_n\}$;
- then one substitutes in φ the variable y_i for each occurrence of $x_i, i = 1, \cdots, n$.

Finally, if φ has been previously described as a formula $\varphi(x_1, \cdots, x_n, X_1, \cdots, X_m)$ (which indicates that its free variables are among $x_1, \cdots, x_n, X_1, \cdots, X_m$) then, $\varphi(y_1, \cdots, y_n, X_1, \cdots,$

X_m) will be another notation for $\varphi[y_1/x_1, \cdots, y_n/x_n]$. With this notation, we have the following lemma, giving the semantics of substitution.

Lemma 1.4 *Let $\varphi \in SO(\mathcal{R}, \{x_1, \cdots, x_n, x_{n+1}, \cdots, x_p\})$ with $x_1, \cdots, x_p$ pairwise distinct. Let $y_1, \cdots, y_n$ be variables. Let $\{z_1, \cdots, z_q\}$ be an enumeration of the set of variables $\{y_1, \cdots, y_n, x_{n+1}, \cdots, x_p\}$, so that $\varphi[y_1/x_1, \cdots, y_n/x_n] \in SO(\mathcal{R}, \{z_1, \cdots, z_q\})$. For every $S \in STR(\mathcal{R})$, for every $d_1, \cdots, d_q \in D_S$ we have*

$$S \models \varphi[y_1/x_1, \cdots, y_n/x_n](d_1, \cdots, d_q) \text{ iff } S \models \varphi(d'_1, \cdots, d'_p)$$

where $d'_i = d_j$ iff $1 \leq i \leq n$ and $z_j = y_i$ or $n+1 \leq i \leq p$ and $z_j = x_i$.

Proof : The sequence $d'_1, \cdots, d'_p$ is well-defined because $\{z_1, \cdots, z_q\}$ is an enumeration of the set $\{y_1, \cdots, y_n, x_{n+1}, \cdots, x_p\}$. The verification is easy. □

We now define second-order substitutions, by which relation symbols can be replaced by formulas, intended to define the corresponding relations. Let $\varphi \in SO(\mathcal{R}, \{x_1, \cdots, x_m, X_1, \cdots, X_p\})$. For each $i = 1, \cdots, p$, we let $\psi_i \in SO(\mathcal{R}, \{x_1, \cdots, x_m, y_1, \cdots, y_{\rho(X_i)}\}$ where the listed variables are pairwise distinct. We let $\varphi[\psi_1/X_1, \cdots, \psi_p/X_p]$ be the formula in $SO(\mathcal{R}, \{x_1, \cdots, x_m\})$ constructed as follows :

- by using Lemma 1.3 one first constructs φ' equivalent to φ where no variable of $\{x_1, \cdots, x_m\}$ is bound ;
- then one replaces every atomic formula $X_i(u_1, \cdots, u_{\rho(X_i)})$ of φ' (where the variables $u_1, \cdots, u_{\rho(X_i)}$ need not be pairwise distinct) by the formula $\psi_i[u_1/y_1, \cdots, u_{\rho(X_i)}/y_{\rho(X_i)}]$.

With this notation, we have the following lemma.

Lemma 1.5 *Let $S \in STR(\mathcal{R})$, let $d_1, \cdots, d_m \in D_S$. For each $i = 1, \cdots, p$ let T_i be the $\rho(X_i)$-ary relation on D_S defined by $(a_1, \cdots, a_{\rho(X_i)}) \in T_i$ iff $S \models \psi_i(d_1, \cdots, d_m, a_1, \cdots, a_{\rho(X_i)})$. Then*

$$S \models \varphi[\psi_1/X_1, \cdots, \psi_p/X_p](d_1, \cdots, d_m) \text{ iff } S \models \varphi(d_1, \cdots, d_m, T_1, \cdots, T_p).$$

Proof : Straightforward verification from the definition, by using an induction on the structure of φ. □

Let $S \in STR(\mathcal{R})$ and E be a subset of D_S. We denote by $S[E]$ the *restriction of* S to E, i.e., the structure $S' \in STR(\mathcal{R})$ with domain E and such that $R_{S'} = R_S \cap E^{\rho(R)}$ for every R. (Since our structures are relational the restriction of a structure to a subset of its domain is always well-defined : this would not be the case if we had to ensure that some functions have a well-defined image.) If S represents a graph G, i.e., $S = \mid G \mid_1$, then $S[E]$ represents the induced subgraph of S with set of vertices E which will be denoted by $G[E]$.

Lemma 1.6 *Let $\varphi \in SO(\mathcal{R}, \{x_1, \cdots, x_n\})$; one can construct a formula $\varphi' \in SO(\mathcal{R}, \{x_1, \cdots, x_n, X\})$ such that the following holds for every $S \in STR(\mathcal{R})$. For every $E \subseteq D_S$, for every $d_1, \cdots, d_n \in E$:*

$$S \models \varphi'(d_1, \cdots, d_n, E) \text{ iff } S[E] \models \varphi(d_1, \cdots, d_n)$$

If φ is MS or FO then φ' is MS or FO respectively.

Proof : We can assume (by Lemma 1.3) that X does not occur in φ. We let φ' be associated with φ by the following inductive definition, where φ may have free variables of all types (relation variables as well as individual variables).

$(\forall Y.\psi)' = \forall Y.\psi'$

$(\exists Y.\psi)' = \exists Y.\psi'$

$(\forall x.\psi)' = \forall x.[x \in X \Longrightarrow \psi']$

$(\exists x.\psi)' = \exists x[x \in X \wedge \psi']$

$(\psi_1 op \psi_2)' = \psi_1' op \psi_2'$ where $op \in \{\wedge, \vee, \Longrightarrow, \Longleftrightarrow\}$

$(\neg\psi)' = \neg\psi'$

$\psi' = \psi$ for every atomic formula ψ. □

The formula φ' is called the *relativization of φ to X* and will be denoted by $\varphi\lceil X$.

1.7 Transitive closure and path properties

We denote by T^+ the transitive closure of a binary relation T.

Lemma 1.7 *Let S be an $\{R\}$-structure where R is binary. The transitive closure of R_S is defined in S by the following formula φ_0 of $MS(\{R\}, \{x_1, x_2\})$:*

$$\forall X[\{\forall y, z(y \in X \wedge R(y, z) \Longrightarrow z \in X) \wedge \forall y(R(x_1, y) \Longrightarrow y \in X)\} \Longrightarrow x_2 \in X]$$

Proof : Let $d_1, d_2 \in D_S$ such that $S \models \varphi_0(d_1, d_2)$. The set $D = \{d / R_S^+(d_1, d)\}$ satisfies the property $\forall y, z(y \in X \wedge R(y, z) \Longrightarrow z \in X) \wedge \forall y(R(x_1, y) \Longrightarrow y \in X)$. Hence $d_2 \in D$, i.e. $R_S^+(d_1, d_2)$. Conversely, if $R_S^+(d_1, d_2)$ then d_2 must belong to every set D such that $R_S(d_1, d)$ for all $d \in D$ and that is *closed under* R_S, i.e., is such that $d \in D$ and $R_S(d, d')$ implies $d' \in D$. This shows that $S \models \varphi_0(d_1, d_2)$.□

We now discuss some applications of this lemma to path[b] properties in graphs. We shall write formulas relative to $\mathcal{R}_s$.

Lemma 1.8 *One can write MS formulas expressing in $\mid G \mid_1$ the following properties of vertices x, y and sets of vertices X, where G is an arbitrary directed graph, represented by the structure $\mid G \mid_1$ in $STR(\mathcal{R}_s)$.*

P1 : $x = y$ or there is a nonempty path from x to y,

P2 : G is strongly connected,

P3 : G is connected,

P4 : $x \neq y$ and there is a path from x to y, all vertices of which belong to the set X,

P5 : X is a connected component of G,

P6 : G has no circuit,

P7 : G is a directed tree,

P8 : G has no circuit, $x \neq y$ and X is the set of vertices of a path from x to y,

P9 : G is planar.

Proof: We shall denote by $\varphi_i, i = 1, \cdots, 9$ the formula expressing property Pi, in a structure $S \in STR(\mathcal{R}_s)$.

(1) φ_1 is the formula $x = y \vee \varphi_0(x, y)$ where φ_0 is constructed in Lemma 1.7 with $R = edg$.

(2) φ_2 is the formula $\forall x, y[\varphi_1(x, y)]$.

(3) φ_3 is the formula $\varphi_2[\theta/edg]$ where $\theta(x_1, x_2)$ is $edg(x_1, x_2) \vee edg(x_2, x_1)$.

(4) φ_4 is the formula $\neg x = y \wedge x \in X \wedge y \in X \wedge \varphi_1\lceil X$ hence $\varphi_4(x, y, X)$ says that x, y are vertices of $G[X]$ and that there is a path in $G[X]$ from x to y; this is equivalent to the property of the statement. (We recall that we denote by $G[X]$ the induced subgraph of G with set of vertices X.)

(5) $\varphi_5(X)$ is the formula $\varphi_3\lceil X \wedge \forall Y[X \subseteq Y \wedge \varphi_3\lceil Y \Longrightarrow Y \subseteq X]$; (note that $\varphi_3\lceil X$ expresses that $G[X]$ is connected) ; the subformula $X \subseteq Y$ stands for : $\forall u[u \in X \Longrightarrow u \in Y]$.

(6) φ_6 is the formula $\forall x, y[edg(x, y) \Longrightarrow \neg\varphi_1(y, x)]$.

(7) φ_7 is the formula $\varphi_6 \wedge \exists x \forall y[\varphi_1(x, y)] \wedge \forall x, y, z[edg(x, z) \wedge edg(y, z) \Longrightarrow x = y]$ expressing that G has no circuit, that every vertex is reachable from some vertex (the root) by a directed path and every vertex is of indegree at most 1 : we consider directed trees with edges directed *from* the root, towards the leaves.

[b] A nonempty path in G is here a sequence of vertices $(x_1, x_2, \cdots, x_n)$ with $n \geq 2$ such that $(x_i, x_{i+1}) \in edg_G$ for all i and $x_i = x_j \Rightarrow i = j$ or $\{i, j\} = \{1, n\}$; if $x_1 = x_n$, this path is a *circuit*. We consider (x) as an empty path for every vertex x.

(8) The construction of φ_8 is more complex. We let $\varphi'_8(X)$ express that X is linearly ordered by the relation $edg^*_{G[X]}$ where G is assumed to have no circuit, i.e. $\varphi'_8(X)$ is

$$\forall x, y[x \in X \wedge y \in X \Longrightarrow x = y \vee \varphi_4(x, y, X) \vee \varphi_4(y, x, X)].$$

Let $\varphi''_8(x, y, X)$ be the formula defined as

$$\varphi_6 \wedge \varphi'_8(X) \wedge x \in X \wedge y \in X \wedge \neg x = y \wedge \forall z \in X[(\varphi_4(x, z, X) \vee x = z) \wedge (\varphi_4(z, y, X) \vee z = y)].$$

It says that the considered graph has no circuit, that X is linearly ordered by $edg^*_{G[X]}$ with first element x and last element y with $y \neq x$. If X is finite, this is enough to express that X is the set of vertices of a path from x to y. But this is not true if X is infinite. (Take for example G with set of vertices $\mathcal{N}$ and set of edges $\{(i, j)/i \neq 0, j \in \{0, i + 1\}\}$, take $X = \mathcal{N}$, $x_1 = 1$ and $x_2 = 0$.) We let $\theta(x_1, x_2, X)$ be the following formula :

$$\varphi_6 \wedge \varphi'_8(X) \wedge \varphi_4(x_1, x_2, X) \wedge \forall w[w \in X \wedge \varphi_4(x_1, w, X) \Longrightarrow x_2 = w \vee \varphi_4(x_2, w, X)]$$

expressing that $x_1, x_2 \in X$ and x_2 is the successor of x_1 in X with respect to the linear order $edg^*_{G[X]}$. Let E be the binary relation on X defined by θ. Then X is the set of vertices of a path from x to y iff $(x, y) \in E^+$. Hence the following formula $\varphi_8(x, y, X)$ expresses the desired property :

$$\varphi''_8 \wedge \varphi_0[\theta/R](x, y, X)$$

where φ_0 is from Lemma 1.7.

(9) This will be proved in Section 7 by using Kuratowski's theorem. □

The condition that G be acyclic is important in $P8$: we shall prove in the next proposition that one cannot express by an MS formula that, in a finite directed graph, a given set X is the set of vertices of a path from x to y.

We now review some properties that are provably not MS-expressible. A directed graph is Hamiltonian if it has at least 2 vertices and a circuit passing through all vertices.

Proposition 1.9 *The following properties are not MS-expressible :*

1. *two sets X and Y have equal cardinality,*
2. *a directed graph is Hamiltonian,*
3. *in a directed graph a set X of vertices is the set of vertices of a path from x to y,*
4. *a graph has a nontrivial automorphism.*

Proof : We shall use the following result of Büchi and Elgot (see Thomas [62], Thm 3.2): if $L \subseteq \{a,b\}^*$ is the set of words u such that $\| u \| \models \varphi$ where φ is a closed MS-formula, then L is a regular language.

(1) Assume that we have a formula $\psi \in MS(\emptyset, \{X,Y\})$ such that for every set S, for every $V, W \subseteq S$:

$$S \models \psi(V,W) \text{ if and only if } Card(V) = Card(W).$$

Then the MS-formula $\psi[lab_a(x_1)/X, lab_b(x_1)/Y]$ of $MS(\mathcal{R}_{w,\{a,b\}})$ would characterize (as a subset of $\{a,b\}^*$) the language L of words having as many $a's$ as $b's$. This language is not regular, so we get a contradiction.

(2) With every word $w \in \{a,b\}^+$ represented by the structure $\| w \| = < \{1,\cdots,n\}, suc_{\|w\|}, lab_{a\|w\|}, lab_{b\|w\|} >$ we associate the graph K_w with set of vertices $\{1,\cdots,n\}$ and an edge from i to j iff $i \in lab_{a\|w\|}$ and $j \in lab_{b\|w\|}$ or vice-versa. Hence K_w is a complete bipartite directed graph. It is Hamiltonian iff w belongs to the language L already used in (1).

Let us now assume the existence of a formula η in $MS(\mathcal{R}_s, \emptyset)$ that would characterize the Hamiltonian graphs among finite graphs. Let $\mu(x_1,x_2)$ be the FO formula defined as :

$$(lab_a(x_1) \wedge lab_b(x_2)) \vee (lab_b(x_1) \wedge lab_a(x_2))$$

This formula defines in $\| w \|$ the edges of K_w (note that $D_{\|w\|} = V_{K_w}$). It follows that for every word $w \in \{a,b\}^*$ of length $n \geq 2$ we have

$$w \in L \text{ iff } K_w \text{ is Hamiltonian iff } \| w \| \models \eta[\mu/edg],$$

and the set of words in L of length at least 2 is regular, a contradiction.

(3) Assume we have a formula $\varphi \in MS(\mathcal{R}_s, \{x,y,X\})$ expressing that in a directed graph, X is the set of vertices of a directed path from x to y. Then the MS formula :

$$\exists X[\forall u(u \in X) \wedge \exists x,y(\varphi(x,y,X) \wedge \neg x = y \wedge edg(y,x))]$$

expresses that the considered graph has at least two vertices and is Hamiltonian. This is not possible by (2) hence no such formula φ does exist.

(4) The proof is very much like that of (2). With every word of the form a^nbca^m with $n \geq 2, m \geq 2$ one associates the graph $H_{n,m}$ with set of vertices $\{-n, -n+1, \cdots, -1, 0, 1, 2, \cdots, m\} \cup \{0'\}$ and undirected edges between 0 and $0'$ and between i and $i+1$ for every $i, -n \leq i < m$. This graph has a nontrivial automorphism iff $n = m$ iff the given word belongs to the non-regular language $\{a^nbca^n / n \geq 2\}$. Hence, no MS formula can express that a graph has a nontrivial automorphism. We omit details. □

Remark : The structure representing $H_{n,m}$ can be obtained from the structure $\| a^nbca^m \|$ by a definable transduction using quantifier-free formulas (to be introduced in Section 4 below).

1.8 Monadic second-order logic without individual variables

This technical subsection may be skipped on first reading. We define a syntactical variant of MS logic where all variables denote sets. The corresponding formulas are less readable than those used up to now, but the proofs concerning them and the properties they define will be easier, because the syntax is limited.

We let $\mathcal{R}$ be as before ; we shall only use set variables. We define $MS'(\mathcal{R}, \{X_1, \cdots, X_n\})$ by taking atomic formulas of the forms :

$X \subseteq Y$ where X, Y are variables,

$R(Y_1, \cdots, Y_n)$ where $R \in \mathcal{R}$, $n = \rho(R)$ and $Y_1, \cdots, Y_n$ are variables.

The formulas are constructed with the usual propositional connectives and set-quantifications $\exists X$ and $\forall X$. We denote by $MS'(\mathcal{R}, \{X_1, \cdots, X_n\})$ the set of such formulas with free variables among $X_1, \cdots, X_n$.

Let $S \in STR(\mathcal{R})$. The meaning of $X \subseteq Y$ is set inclusion. If $D_1, \cdots, D_n$ are sets denoted by $Y_1, \cdots, Y_n$ then $R(Y_1, \cdots, Y_n)$ is true in S iff $(d_1, \cdots, d_n) \in R_S$ for some $d_1 \in D_1, \cdots, d_n \in D_n$. The validity of formulas in $MS'(\mathcal{R}, \{X_1, \cdots, X_n\})$ in a structure S and for an assignment $\gamma : \{X_1, \cdots, X_n\} \to S$ follows immediately.

Lemma 1.10 *For every formula $\varphi \in MS'(\mathcal{R}, \{X_1, \cdots, X_n\})$ one can construct an equivalent formula φ' in $MS(\mathcal{R}, \{X_1, \cdots, X_n\})$.*

Proof : One replaces $X \subseteq Y$ by $\forall x[x \in X \Longrightarrow x \in Y]$ and $R(Y_1, \cdots, Y_n)$ by $\exists y_1, \cdots, y_n[y_1 \in Y_1 \wedge \cdots \wedge y_n \in Y_n \wedge R(y_1, \cdots, y_n)]$. We omit details.□

Lemma 1.11 *For every formula $\varphi \in MS(\mathcal{R},\{x_1, \cdots, x_n, Y_1, \cdots, Y_m\})$ one can construct a formula $\varphi' \in MS'(\mathcal{R},\{X_1, \cdots, X_n, Y_1, \cdots, Y_m\})$ such that for every $S \in STR(\mathcal{R})$ for every assignment $\gamma : \{x_1, \cdots, x_n, Y_1, \cdots, Y_m\} \longrightarrow S$ then $(S, \gamma) \models \varphi$ iff $(S, \gamma') \models \varphi'$ where $\gamma'(X_i) = \{\gamma(x_i)\}$ and $\gamma'(Y_i) = \gamma(Y_i)$.*

Proof : We first let $X = \emptyset$ be the formula $\forall Y[X \subseteq Y]$ in $MS'(\emptyset, \{X\})$ which characterizes the empty set. We let also $Sing(X) \in MS'(\emptyset, \{X\})$ be the formula $\forall Y[Y \subseteq X \Longrightarrow X \subseteq Y \vee Y = \emptyset]$ which characterizes the singleton sets.

We now translate φ into φ'. For each individual variable x, we let X denote a new set variable. ("New" means distinct of any variable in φ). We obtain φ' from φ by the following inductive construction :

$(x = y)' = X \subseteq Y \wedge Y \subseteq X$,

$(R(y_1, \cdots, y_n))' = R(Y_1, \cdots, Y_n)$,

$(x \in Y)' = X \subseteq Y$,

$(\varphi_1 op \varphi_2)' = \varphi_1' op \varphi_2'$

for every binary connective op equal to $\vee, \wedge, \Longrightarrow$ or $\Longleftrightarrow$,

$(\neg\varphi_1)' = \neg\varphi_1'$,

$(\forall X.\varphi_1)' = \forall X.\varphi_1'$,

$(\exists X.\varphi_1)' = \exists X.\varphi_1'$,

$(\forall x.\varphi_1)' = \forall X[Sing(X) \Longrightarrow \varphi_1']$,

$(\exists x.\varphi_1)' = \exists X[Sing(X) \wedge \varphi_1']$.

We omit details. □

1.9 A worked example : the definition of square grids in MS logic

The *rectangular grid* $G_{m,n}$ is the graph with set of vertices $V = \{0, 1, \cdots, m-1\} \times \{0, 1, 2, \cdots, n-1\}$ and set of edges $E = \{((i,j),(i',j'))/(i,j),\ (i',j') \in V$ and either $i' = i$ and $j' = j+1$ or $i' = i+1$ and $j' = j\}$. Its *north-, west-, south,* and *east-borders* are the sets of vertices :

$X_n = \{0, \cdots, m-1\} \times \{n-1\}$

$X_w = \{0\} \times \{0, \cdots, n-1\}$

$X_s = \{0, \cdots, m-1\} \times \{0\}$

$X_e = \{m-1\} \times \{0, \cdots, n-1\}$.

Its *well-coloring* is the 4-tuple of sets of vertices Y_0, Y_1, Y_2, Y_3 such that :

$(i,j) \in Y_k$ iff $k = mod(i) + 2(mod(j))$

where $mod(i)$ is the remaining (in $\{0,1\}$) of the integer division of i by 2.

We claim that for every positive integer m there exists an MS_1 formula θ with free variables $Y_0, Y_1, Y_2, Y_3, X_n, X_w, X_s, X_e$ that characterizes (among the finite directed simple graphs) those isomorphic to $G_{2m,2m}$ where, in addition, Y_0, Y_1, Y_2, Y_3 form a well-coloring and X_n, X_w, X_s, X_e are the four borders.

We let θ express the following conditions concerning a simple graph G given with sets of vertices $Y_0, Y_1, Y_2, Y_3, X_n, X_e, X_s, X_w$:

(1) G has no circuit;

(2) Y_0, Y_1, Y_2, Y_3 form a partition of V_G ; assuming this we shall call i-vertex, i-successor of y, i-predecessor of y a vertex, or a successor of y, or a predecessor of y that belongs to Y_i ;

(3) every vertex has at most one i-successor and at most one i-predecessor for each i;

(4) a 0- or a 3-vertex has no 0- or 3-successors ; a 1- or a 2-vertex has no 1- or 2-successor;

(5) $G[X_n]$ is a path consisting alternatively of 2- and 3-vertices ; its origin is a 2-vertex and its end is a 3-vertex ; the origin is the unique element of $X_n \cap X_w$ and the end is the unique element of $X_n \cap X_e$;

(5') $G[X_e]$ is a path consisting alternatively of 1- and 3- vertices ; its origin is a 1-vertex which is the unique element of $X_s \cap X_e$; its end is a 3-vertex which is the unique element of $X_n \cap X_e$;

(5") and (5"') state similar conditions on the sets X_s and X_w;

(6) each 2-vertex in X_n has a 3-successor and no 0-successor ; each 3-vertex in $X_n - X_e$ has a 2-successor and no 1-successor;

(6') (6") (6"') state similar properties of X_e, X_s, X_w;

(7) each vertex in $V_G - (X_n \cup X_e \cup X_s \cup X_w)$ has two predecessors and two successors;

(8) there exists a path from the vertex in $X_s \cap X_w$ to the one in $X_n \cap X_e$ the vertices of which have the colors $0, 1, 3, 2, 0, 1, 3, 2, \cdots 0, 1, 3$ in this order.

It is not hard to see that conditions (1) to (7) characterize the well-colored grids of the form $G_{2n,2m}$ for $n \geq 1, m \geq 1$. Condition (8) implies furthermore that $m = n$. It is not hard to modify this construction in order to characterize the grids of the form G_{2n+1}, G_{2n+1} for $n \geq 1$. □

In view of the proof of Proposition 1.2, a line of G given with $Y_0, Y_1, \cdots, X_w$ as above is a set $L \subseteq V_G$ such that

(1) $G[L]$ is a path from a vertex in X_w to a vertex in X_e

(2) either $L \subseteq Y_0 \cup Y_1$ or $L \subseteq Y_2 \cup Y_3$.

2 Representations of partial orders, graphs and hypergraphs by relational structures

In this section we discuss more in detail the *various possibilities* of representing a partial order, a graph or a hypergraph by a relational structure. The choice of the representation is important for the possibility of expressing a given property by some formula of the three logical languages we have introduced.

2.1 Partial orders

The simplest way to represent a partial order $\leq_D$ on a set D is by the structure $< D, \leq_D >$. If D is finite, one can also represent $< D, \leq_D >$ by a graph $< D, S >$ where S is binary, such that $\leq_D$ is the reflexive and transitive closure of S. There is even a unique minimal such relation (minimal for inclusion) that we shall denote by suc_D (where suc means successor) and the corresponding graph $< D, suc_D >$ is the classical Hasse-diagram of the partial order $< D, \leq_D >$. A property of partial orders representable by a formula φ with respect to the representation $< D, \leq_D >$ will be representable by another (perhaps more complex) formula ψ with respect to the representation $< D, suc_D >$.

Proposition 2.1 *(1) A property of a finite partial orders is MS with respect to the representation $< D, \leq_D >$ iff it is MS with respect to the representation $< D, suc_D >$.*
(2) Furthermore, it is FO with respect to the representation $< D, \leq_D >$ if it is FO with respect to the representation $< D, suc_D >$.

If a property is FO with respect to the representation $< D, \leq_D >$ we can only conclude (by (1)) that it is MS with respect to the representation $< D, suc_D >$. We have actually the proper hierarchy :

$$FO_{suc} \subset FO_{\leq} \subset MS_{suc} = MS_{\leq},$$

where the subscripts indicate the considered representation. The languages defined by first-order formulas in terms of $\leq$ (as opposed to in terms of the successor relation) are the star-free languages. This class is a proper subclass of the class of regular languages (definable by MS_{Suc} or $MS_{\leq}$ formulas) and contains properly the class of locally threshold testable languages (definable by FO_{suc} formulas) (see Pin [52]). From these strict inclusions concerning classes of languages follow the corresponding strict inclusions for partial orders ($FO_{suc} \subset FO_{\leq} \subset MS_{suc}$).

Proof : We let suc be a binary relation symbol. Since $\leq_D$ is the reflexive and transitive closure of suc_D, it follows that $\leq_D$ is defined in $< D, suc_D >$ by the MS formula $\mu(y_1, y_2)$:

$$y_1 = y_2 \vee \varphi_0[suc(x_1, x_2)/R]$$

where φ_0 is from Lemma 1.7. Hence if a property is expressed by an MS or a FO formula φ with respect to $< D, \leq_D >$ it can be expressed by the MS formula $\varphi[\mu(y_1, y_2)/\leq]$.

Now suc_D is defined in $< D, \leq_D >$ by the FO formula : $\mu'(y_1, y_2)$

$$\neg y_1 = y_2 \wedge y_1 \leq y_2 \wedge \forall z[y_1 \leq z \wedge z \leq y_2 \Longrightarrow y_1 = z \vee z = y_2].$$

Hence a property expressed in $< D, suc_D >$ by an MS or FO formula φ' can be expressed by the formula $\varphi'[\mu'/suc]$ which is MS or FO respectively. □

2.2 Edge set quantifications

As already noted the representation of a graph by a relational structure that we have used up to now is not convenient to express properties of graphs that depend on the multiplic-

ity of edges. We shall define another representation, where the edges are elements of the domain, which is more natural for expressing logically the properties of multigraphs and which, furthermore, makes it possible to express more properties of simple graphs by MS formulas.

We let $\mathcal{R}_m = \{inc\}$ where inc is ternary (inc stands for "incidence"). For every graph G, directed or not, simple or not, we denote by V_G its set of vertices and by E_G its set of edges. The incidence relation between vertices and edges is represented by the ternary relation inc_G such that : $(x, y, z) \in inc_G$ iff x is an edge and either this edge is directed and it links y to z or it is undirected and it links y and z. If x is an undirected edge, we have : $(x, y, z) \in inc_G$ iff $(x, z, y) \in inc_G$.

We let $D_G := V_G \cup E_G$ (we always assume that $V_G \cap E_G = \emptyset$) and we let $\mid G \mid_2$ be the structure $< D_G, inc_G > \in STR(\mathcal{R}_m)$. If G has several edges linking y to z then, they are represented by distinct elements of D_G.

In a structure $S = < D_S, inc_S >$ representing a graph, the edges are the elements x of D_S that satisfy the formula $\exists y, z[inc(x, y, z)]$. The structures in $STR(\mathcal{R}_m)$ representing directed graphs are exactly those which satisfy the following conditions :

(1) $\forall x, y, z[inc(x, y, z) \Longrightarrow \neg\exists u, v(inc(y, u, v) \vee inc(z, u, v))]$, and

(2) $\forall x, y, z, y', z'[inc(x, y, z) \wedge inc(x, y', z') \Longrightarrow y = y' \wedge z = z']$.

In any such structure, if we let $E = \{x \in D_S \ / \ S \models \exists y, z.inc(x, y, z)\}$ and $V = D_S - E$ then $inc_S \subseteq E \times V \times V$ and there exists a unique directed graph G with $V_G = V, E_G = E$ and $\mid G \mid_2 = S$. Similarly, the structures in $STR(\mathcal{R}_m)$ representing undirected graphs are exactly those that satisfy (1) above together with :

(2') $\forall x, y, z, y', z'[inc(x, y, z) \wedge inc(x, y', z') \Longrightarrow (y = y' \wedge z = z') \vee (y = z' \wedge z = y')]$

(3') $\forall x, y, z[inc(x, y, z) \Longrightarrow inc(x, z, y)]$.

Graph properties can be expressed logically, either via the representation of a graph G by $\mid G \mid_2$, or via the initially defined representation $\mid G \mid_1 := < V_G, edg_G >$. The representation $\mid G \mid_1$ only allows quantification on vertices, sets of vertices, relations on vertices (according to the language we consider), whereas the representation $\mid G \mid_2$ also allows quantifications on edges, sets of edges and relations on edges. We shall distinguish the MS_1-definable classes (which are nothing but the MS-definable ones we have considered up to now), from the MS_2-definable ones, which use formulas in $MS(\{inc\}, \emptyset)$ and the representation of a graph G by $\mid G \mid_2$. Similarly, we have FO_1 -, FO_2 -, SO_1- and SO_2-definable classes of graphs.We shall also speak of FO_i - or MS_i - or SO_i-expressible graph properties, where $i = 1$ or 2.

Since a set of edges in a graph can be considered as a binary relation on its set of vertices it is quite clear that every MS_2-expressible property is SO_1-expressible (we shall prove that later). However it is not always MS_1-expressible.

Proposition 2.2 *The following properties of a directed graph G can be expressed by MS_2-formulas.*
(1) X is a set of edges (resp. of vertices) forming a path from x to y, where $x \neq y$.
(2) G is Hamiltonian.

Proof : (1) For every subset X of E_G we denote by $G[[X]]$ the subgraph H of G such that V_H is the set of vertices incident to some edge in X and $E_H = X$. The desired condition is thus that $G[[X]]$ is a path from x to y. This can be expressed by a formula ψ'_1 in $MS(\mathcal{R}_m, \{x, y, X\})$ constructed with the help of Lemma 1.6 and the formula φ_8 of Lemma 1.8. The desired formula ψ_1 is thus

$$\neg x = y \wedge \forall z[z \in X \Longrightarrow \exists u, v(inc(z, u, v))] \wedge \psi'_1.$$

If we want to express that Y is the set of vertices of a path from x to y we take $\psi_2(x, y, Y)$ defined as

$$\exists X[\psi_1(x, y, X) \wedge \forall z[z \in Y \Longleftrightarrow \exists u, v(u \in X \wedge (inc(u, z, v) \vee inc(u, v, z)))]].$$

The quantification on sets of edges is thus crucial in ψ_2 since we have proved that no formula in $MS(\mathcal{R}_s, \{x, y, Y\})$ equivalent to ψ_2 does exist (see Proposition 1.9).
(2) follows from (1) : see the proof of Proposition 1.9, assertion (3).□

Proposition 2.3 *Let $\mathcal{C}$ be a class of simple, directed or undirected graphs :*

(1) $\mathcal{C}$ is FO_1-definable iff $\mathcal{C}$ is FO_1-definable

(2) $\mathcal{C}$ is SO_2-definable iff $\mathcal{C}$ is SO_1-definable

(3) $\mathcal{C}$ is MS_2-definable if $\mathcal{C}$ is MS_1-definable and the converse does not hold.

Proof : The relation edg_G is definable from inc_G by the formula $\exists u.inc(u, x_1, x_2)$. It follows that for each $L \in \{\text{FO, SO, MS}\}$, $\mathcal{C}$ is L_2-definable if it is L_1-definable.

That the converse does not hold for MS follows from Propositions 1.9 and 2.2 : the class of simple Hamiltonian directed graphs (with at least 2 vertices) is MS_2-definable but is not MS_1-definable relatively to the class of simple graphs. It holds for FO because a quantification of the form "there exists an edge e..." can be replaced by a quantification of the form "there exist vertices x and y that form an edge such that...". The proof is similar for SO : a quantification over n-ary edge relations is replaced by a quantification over $2n$-ary vertex relations. We omit details. □

The next theorem reviews some classes of graphs on which MS_1 and MS_2 are equally expressive. (Tree-width will be defined in Section 5).

Theorem 2.4 *Let $\mathcal{C}$ be the class of planar directed graphs, or of directed graphs of degree at most k (for any fixed k) or finite directed graphs of tree-width at most k (for any fixed k). A property of graphs in $\mathcal{C}$ is MS_2-expressible iff it is MS_1-expressible. The same holds for the corresponding classes of undirected graphs.*

We do not reproduce the full proof of this theorem which is quite long (see Courcelle [14]). We only give the basic lemma and some consequences.

Let G be a directed graph. A *semistrong $k-$coloring* of G is a mapping $\gamma : V_G \to \{1, 2, \cdots, k\}$ such that, for every two vertices $v, v' \neq v$:

1. if v, v' are adjacent, then $\gamma(v) \neq \gamma(v')$,
2. if there are edges linking v to w and v' to w, where w is a third vertex, then $\gamma(v) \neq \gamma(v')$.

The following lemma will be proved in Section 4. Let $\mathcal{C}_k$ be the class of finite or infinite simple directed loop-free graphs having a semistrong k-coloring.

Lemma 2.5 *Let $k \in \mathcal{N}$. A property of graphs in $\mathcal{C}_k$ is MS_1-expressible if it is MS_2-expressible.*

The class $\mathcal{C}_2$ contains the directed trees. (Edges are directed in such a way that there is a unique path from the root to each vertex). Hence the languages MS_1 and MS_2 are equally powerful for expressing properties of directed trees. Let $d \in \mathcal{N}$ and $k = d^2 + 1$. Let us prove that $\mathcal{C}_k$ contains the simple directed loop-free graphs of degree at most d. Let G be such a graph. Let G' be the graph obtained by adding an edge between any two vertices at distance 2. This graph has degree at most $d + d(d-1) = d^2$. Hence G' has a k-vertex coloring (in the usual sense, where one requires that any two adjacent vertices have different colors). Such a coloring is a semistrong k-coloring of G. Hence we have proved that Theorem 2.4 holds for the classes of directed trees and of simple directed loop-free graphs of degree at most any fixed d. The complete proof of Theorem 2.4 is based on Lemma 2.5 and constructions of appropriate colorings.

2.3 Hypergraphs

We define directed, hyperedge labelled hypergraphs. We let A be a finite ranked set : each symbol $a \in A$ has an associated rank, a nonnegative integer denoted by $\tau(a)$. A hypergraph H has a set of vertices V_H and a set of hyperedges E_H ; each hyperedge e has a label $lab_H(e)$ in A and a sequence of vertices of length $\tau(lab_H(e))$. The label of e may have rank 0 and in this case, e has no vertex. We always assume that $V_H \cap E_H = \emptyset$. As for graphs we define two representations of hypergraphs by relational structures.

We let $\mathcal{R}_s(A) := \{edg_a / a \in A\}$ and $\mathcal{R}_m(A) := \{inc_a / a \in A\}$ where edg_a is $\tau(a)$-ary and inc_a is $(\tau(a)+1)$-ary. With a hypergraph H we will associate the structures $\mid H \mid_1 := < V_H, (edg_{aH})_{a \in A} > \in STR(\mathcal{R}_s(A))$ and $\mid H \mid_2 := < V_H \cup E_H, (inc_{aH})_{a \in A} > \in STR(\mathcal{R}_m(A))$ where :

(1) $inc_{aH}(x, y_1, \cdots, y_n)$ holds iff $x \in E_H, y_1, \cdots, y_n \in V_H, lab_H(x) = a, n = \tau(a)$ and $(y_1, \cdots, y_n)$ is the sequence of vertices of x, and

(2) $edg_{aH}(y_1, \cdots, y_n)$ holds iff $inc_{aH}(x, y_1, \cdots, y_n)$ holds for some $x \in E_H$.

The structure $| H |_1$ contains no information on the hyperedges of type 0, and no information either on the number of hyperedges having the same label and the same sequence of vertices. A hypergraph is *simple* if it has no hyperedge of type 0 and if no two hyperedges have the same label and the same sequence of vertices. The structures $| H |_2$ are appropriate for representing all hypergraphs H whereas the structures $| H |_1$ are only appropriate for representing simple hypergraphs or for expressing logically properties of hypergraphs that are independent of the multiplicity of hyperedges and of the existence of hyperedges of type 0.

We shall refer by MS_i to MS logic relative to the representation of a hypergraph H by the structure $| H |_i$ where $i = 1, 2$. The results of Proposition 2.3 hold for hypergraphs built over a fixed finite set A as well as for graphs.

Proposition 2.6 *Let $k \in \mathcal{N}$ and A be a finite ranked alphabet. The same properties of finite simple hypergraphs over A of tree-width at most k are expressible in MS_1 and in MS_2.*

Proof : See Courcelle and Engelfriet [23]. □

In certain cases, hypergraphs are equipped with distinguished vertices called sources or ports (see Section 5). These vertices will be represented in relational structures by means of additional unary relations. For instance, if a hypergraph H is given with k sets of distinguished vertices then the structure $| H |_1$ contains k unary relations $P_{1H}, \cdots, P_{kH}$ where $P_1, \cdots, P_k$ are additional unary symbols.

3 The expressive powers of monadic-second order languages

In the preceding section, we have seen that the choice of a representing structure for an object like a partial order, a graph or a hypergraph may affect the first-order or the monadic second-order expressibility of properties of these objects. Here, we shall consider the extension of MS logic by cardinality predicates. In some cases this extension is just a syntactic shorthand, and in others we shall obtain a real extension of expressive power.

3.1 Cardinality predicates

We first extend MS logic by constructing formulas with the help of the new atomic formulas of the form $Fin(X)$ where X is a set variable. Such a formula is valid iff X denotes a finite set. We shall denote by $MS^f(\mathcal{R}, \mathcal{Y})$ the corresponding sets of formulas. We have $MS(\mathcal{R}, \mathcal{Y}) \subseteq MS^f(\mathcal{R}, \mathcal{Y})$.

This extension is of interest only in the case where we consider possibly infinite structures. In finite structures, $Fin(X)$ is always true, and every formula in $MS^f(\mathcal{R}, \mathcal{Y})$ can be simplified into one in $MS(\mathcal{R}, \mathcal{Y})$, that is equivalent in finite structures. Otherwise, we obtain a real increase of expressive power because the finiteness of a set is not, in general, MS-expressible. See Corollary 3.3 below. We wrote "in general" because in certain structures like binary directed trees, it is, as we shall see.

We now introduce an extension of MS logic called *counting monadic second-order logic* and denoted by CMS. For every integer $p \geq 2$, for every set variable X, we let $Card_p(X)$ be a new atomic formula expressing that the set denoted by X is finite and that its cardinality is a (possibly null) multiple of p. We denote by $CMS(\mathcal{R}, \mathcal{Y})$ the extension of $MS^f(\mathcal{R}, \mathcal{Y})$ with atomic formulas of the forms $Card_p(X)$ for $p \geq 2$, where X is a set variable. We have thus a hierarchy of languages $MS \subset MS^f \subset CMS$. We shall discuss cases where the corresponding hierarchy of graph properties is also strict.

Lemma 3.1 *For each $k \in \mathcal{N}$, there is a first-order formula in $FO(\emptyset, \{X\})$ expressing that the set denoted by X has cardinality k.*

Proof : We only give the formula for $k = 3$:
$\exists x_1, x_2, x_3[x_1 \in X \wedge x_2 \in X \wedge x_3 \in X \wedge \neg x_1 = x_2 \wedge \neg x_1 = x_3 \wedge \neg x_2 = x_3 \wedge$
$\forall y(y \in X \Longrightarrow x_1 = y \vee x_2 = y \vee x_3 = y)]$. □

It follows that one can express in CMS that a set X is finite and has a cardinality of the form $q + \lambda p$ for some $\lambda \in \mathcal{N}$ where $1 \leq q < p$: it suffices to write that for some Y and Z, $X = Y \cup Z$, $Y \cap Z = \emptyset$, $Card(Y) = q$ and $Card(Z)$ is a multiple of p. We now recall a result from Courcelle [11].

Proposition 3.2 *Every formula $\varphi \in MS(\emptyset, \{X_1, \cdots, X_n\})$ is equivalent to a finite disjunction of conjunctions of conditions of the forms $Card(Y_1 \cap Y_2 \cap \cdots \cap Y_n) = m$ or $Card(Y_1 \cap Y_2 \cap \cdots \cap Y_n) > m$ where $m \in \mathcal{N}$ and, for each $i = 1, \cdots, n, Y_i$ is either X_i or $D - X_i$, where D is the domain of the considered structure.*

By Lemma 3.1 the conditions $Card(Y_1 \cap Y_2 \cap \cdots \cap Y_n) = m$ and $Card\ (Y_1 \cap Y_2 \cap \cdots \cap Y_n) > m$ can be expressed by formulas in $FO(\emptyset, \{Y_1, \cdots, Y_n\})$. This result shows that MS logic is equivalent to FO logic for "pure" finite or infinite sets.

Corollary 3.3 *CMS is strictly more expressive than MS on finite sets. MS^f is strictly more expressive than MS on infinite sets.*

Proof : Let us assume that a formula $\varphi \in MS(\emptyset, \{X\})$ can express, in every finite set D that a subset X of D has even cardinality. By Proposition 3.2, φ can be expressed as a disjunction of conditions of the forms :

(1) $Card(X) = m$ and $Card(D - X) = m'$,

(2) $Card(X) = m$ and $Card(D - X) > m'$,

(3) $Card(X) > m$ and $Card(D - X) = m'$,

(4) $Card(X) > m$ and $Card(D - X) > m'$.

We let M be the maximum integer m or m' occurring in these conditions. No condition of the form (3) or (4) can appear because it can be valid for sets X with large enough odd

cardinality. The remaining conditions imply that $Card(X) \leq M$. But then φ is not valid for certain sets X with even cardinality larger than M. Contradiction.

If we now assume that φ as above expresses that X is finite we also get a contradiction by the same case analysis. $\square$

This proves that for finite structures, we have the hierarchy (where $\subset$ indicates a proper increase of expressive power and $\equiv$ indicates an equivalent one)

$$MS \equiv MS^f \subset CMS$$

whereas for general (possibly infinite) structures we have

$$MS \subset MS^f \subset CMS.$$

We shall now investigate classes for which these proper inclusions become equivalences.

3.2 Linearly ordered structures

We let $\mathcal{L}$ be the class of finite linear orders ; we shall represent them by structures of the form $< D, \leq_D >$ as explained in Subsection 2.1.

Lemma 3.4 *For every $p \geq 2$ one can construct a formula $\gamma_p(X) \in MS(\{\leq\}, \{X\})$ expressing in every $D \in \mathcal{L}$ that X has a cardinality equal to 0 modulo p.*

Proof : One can easily write a formula γ_p expressing the following :
either X is empty or there exist sets $Y_1, \cdots, Y_p$ forming a partition of X and such that

1. the first element of X is in Y_1,
2. the last element of X is in Y_p,
3. for every two elements x, y of X such that y is the successor of x for the restriction to X of the order $\leq_D$, then, $y \in Y_{i+1}$ if $x \in Y_i$, for some $i = 1, \cdots, p$, and $y \in Y_1$ if $x \in Y_p$. $\square$

This lemma extends to any class $\mathcal{C}$ of finite structures on which a linear order is definable by MS-formulas. Let us define precisely this notion.

Let $\mathcal{C} \subseteq STR(\mathcal{R})$ for some finite set $\mathcal{R}$ of relation symbols. Let $\theta_0 \in MS(\mathcal{R}, \{X_1, \cdots, X_n\})$ and $\theta_1 \in MS(\mathcal{R}, \{x_1, x_2, X_1, \cdots, X_n\})$. We say that (θ_0, θ_1) *defines a linear order on the structures of* $\mathcal{C}$ if the following conditions hold, for every $S \in \mathcal{C}$:

1. $S \models \exists X_1, \cdots, X_n.\theta_0$
2. $S \models \forall X_1, \cdots, X_n[\theta_0 \Longrightarrow \forall x, y, z\{\theta_1(x, x) \wedge (\theta_1(x, y) \wedge \theta_1(y, x) \Longrightarrow x = y)$

$\wedge(\theta_1(x,y) \wedge \theta_1(y,z) \Longrightarrow \theta_1(x,z)) \wedge (\theta_1(x,y) \vee \theta_1(y,x))\}]$.

In words this means that for every choice of sets $D_1, \cdots, D_n$ satisfying θ_0, the binary relation defined by θ_1 where $X_1, \cdots, X_n$ take respectively the values $D_1, \cdots, D_n$ is a linear order on D_S, and that for every $S \in \mathcal{C}$, there is at least one such tuple from which a linear order is definable by θ_1.

Proposition 3.5 *Let $\mathcal{C}$ be a class of finite structures on which a linear order is MS-definable. The CMS-expressible properties of the structures in $\mathcal{C}$ are MS-expressible.*

Proof : We let $\mathcal{R}$ be such that $\mathcal{C} \subseteq STR(\mathcal{R})$. Let $(\theta_0(X_1, \cdots, X_n), \theta_1(x, y, X_1, \cdots, X_n))$ be a pair of formulas that defines a linear order on every structure of $\mathcal{C}$. Let $\varphi \in CMS(\mathcal{R}, \emptyset)$ express some property of the structures of $\mathcal{C}$. We can assume that the variables $X_1, \cdots, X_n$ have no occurrence in φ. For each p, we let $\gamma'_p = \gamma_p[\theta_1/\leq] \in MS(\mathcal{R}, \{X_1, \cdots, X_n\})$ where γ_p is from Lemma 3.4. For every $S \in \mathcal{C}$, for every $D_1, \cdots, D_n \subseteq D_S$ satisfying θ_0, for every $D \subseteq D_S$ then

$$S \models \gamma'_p(D, D_1, \cdots, D_n)$$

iff $Card(D)$ is a multiple of p.

We let φ' be obtained from φ by the replacement of every atomic formula $Card_p(Y)$ by $\gamma'_p[Y/X]$. We let then φ" be the formula of $MS(\mathcal{R},\{X\})$:

$$\exists X_1, \cdots, X_n[\theta_0(X_1, \cdots, X_n) \wedge \varphi'(X, X_1, \cdots, X_n)].$$

If $S \models \varphi"(D)$ then $S \models \theta_0(D_1, \cdots, D_n)$ for some $D_1, \cdots, D_n$ and $S \models \varphi'(D, D_1, \cdots, D_n)$ hence $S \models \varphi(D)$ by the construction of φ. Conversely, if $S \models \varphi(D)$ then there exists $(D_1, \cdots, D_n)$ satisfying θ_0 hence $S \models \varphi'(D, D_1, \cdots, D_n)$ and $S \models \varphi"(D)$. Hence φ is equivalent to φ" in every structure S in $\mathcal{C}$. □

A *directed tree* is a directed graph, every vertex of which is reachable from some vertex (called the *root*) by one and only one path. (In particular it is connected and has no circuit).

Proposition 3.6 *Let $d \in \mathcal{N}$. One can define by means of MS formulas a linear order on (possibly infinite) directed trees of degree at most d, and more generally on forests consisting of at most d trees of degree at most d.*

Proof : Let T be a directed tree given by the structure $< V_T, suc_T >$. We denote by $\leq_T$ the reflexive and transitive closure of suc_T. A partition $(V_1, V_2, \cdots, V_d)$ of V_T is *good* if no two successors of a vertex belong to a same set V_i. If T has degree at most d (i.e., if each vertex has at most d successors) then V_T has a good partition in d sets. From a good partition $(V_1, \cdots, V_d)$ of V_T, we can define the following linear order :

$x \leq y$ iff either $x \leq_T y$ or there exist z, x', y' such that $zsuc_T x' \leq_T x, zsuc_T y' \leq_T y, x' \in V_i, y' \in V_j$ and $i < j$.

It is not hard to see that $\leq$ is a linear order on V_T. An FO_1-formula $\theta_0(X_1, \cdots, X_d)$ can express that a given d-tuple $(V_1, V_2, \cdots, V_d)$ of subsets of V_T is good and an MS_1-formula $\theta_1(x, y, X_1, \cdots, X_d)$ can express that $x \leq y$ holds where X_i takes the value V_i (for $i = 1, \cdots, d$) and $(V_1, \cdots, V_d)$ is the good partition from which $\leq$ is defined. (One can write θ_1 with the help of the formula φ_1 of Lemma 1.8.)

Hence (θ_0, θ_1) defines a linear order of V_T (and even a topological sorting) for every tree T of degree at most d. The proof for forests is similar except that in the definition of a good partition, we require that the roots of two trees of the forest do no belong to the same set V_i. $\square$

We refer the reader to Courcelle [18] for extensions of this result to other types of graphs than forests.

3.3 Finiteness

We know that the finiteness of a set is not MS-expressible in general. However it is in certain structures. Let $< D, \leq_D >$ be a linear order isomorphic to $\mathcal{N}$. A subset X of D is finite iff it has a maximal element, which is FO-expressible. We shall use this observation in order to define classes of graphs in which finiteness is MS-expressible. For these classes of graphs the languages MS and MS^f are thus of equal expressive power.

Proposition 3.7 *Let $\mathcal{T}$ be the class of directed trees, each vertex of which has finite degree. The finiteness of a set of vertices of a tree in $\mathcal{T}$ is MS-expressible.*

Proof : We represent a directed tree T by the structure $T = < V_T, suc_T >$. We denote by $root_T$ the root of T. Every vertex is accessible from the root by a directed path. The $\leq_T$-maximal elements of V_T are the leaves and $root_T$ is the unique $\leq_T$-minimal element.

For each nonempty $X \subseteq V_T$ we let $I(X) := \{y \in V_T \ / \ y \leq_T x, x \in X\}$ be the *ideal* generated by X. The graph $T[I(X)]$ is a directed tree and its root is $root_T$.

Claim : X is infinite iff the tree $T[I(X)]$ has an infinite branch.

Proof : Let $(y_0, y_1, y_2, \cdots, y_i, \cdots)$ be an infinite branch in $T[I(X)]$. We construct as follows an infinite sequence in X, proving thus that X is infinite :

1. we let $x_0 \in X$ be an element such that $y_0 \leq_T x_0$;
2. having defined x_j, we define x_{j+1} as follows : we let i be such that y_i is at a larger distance to the root than x_j. We let then x_{j+1} be any element of X such that $y_i \leq_T x_{j+1}$.

The elements $x_0, x_1, \cdots, x_j, \cdots$ of X are at strictly increasing distances to the root. Hence they are pairwise distinct and X is infinite.

Conversely, if X is infinite then $I(X)$ is infinite (since $X \subseteq I(X)$) hence the tree $T[I(X)]$ has an infinite branch by Koenig's lemma. □

We now complete the proof of the proposition. One can construct an MS formula $\theta(X, Y)$ expressing that, in a directed tree T :

Y is linearly ordered by $\leq_T$, and for every $y \in Y$ there is $z \in Y$ such that $y <_T z$ and $z \leq_T x$ for some $x \in X$.

It follows from the claim that a set X is finite iff $\theta(X, Y)$ does not hold for any set Y. □

Corollary 3.3 and Propositions 3.6 and 3.7 yield the following result (we omit details) :

Corollary 3.8 *(1) Let $d \in \mathcal{N}$. The languages MS, MS^f and CMS are equally powerful for expressing properties of trees of degree at most d.*

(2) The languages MS and MS^f are equally powerful for expressing properties of trees of finite degree, and the language CMS is strictly more powerful than them.

4 Monadic second-order definable transductions

In this section, we use MS formulas in order to define certain graph transformations, that are as important in the theory of context-free graph grammars as are rational transductions in language theory.

A binary relation $R \subseteq A \times B$ where A and B are sets, typically of words or of trees, can be considered as a multivalued partial mapping associating with certain elements of A one or more elements of B. It is called in this case a *transduction* : $A \to B$. Transductions of words and trees, defined in terms of finite-state automata with output are essential in Formal Language Theory, especially in constructions concerning context-free grammars. (See Berstel [4]). Does there exist an analogous notion for graphs ? Since there is no convenient notion of graph automaton, there is no "machine-based" notion of graph transduction. However, by using a classical technique of logic called "interpretation", by which a structure is defined in another one, one can define transformations of structures, whence of graphs and hypergraphs via their representations by structures. The formulas defining a structure T inside another one S (or rather : inside the union of k disjoint copies of S) will be MS formulas.

4.1 Transductions of relational structures

We first define *(monadic second-order) definable transductions of relational structures.* Let $\mathcal{R}$ and $\mathcal{Q}$ be two finite ranked sets of relation symbols. Let $\mathcal{W}$ be a finite set of set variables, called here the set of *parameters.* (It is not a loss of generality to assume that all parameters are set variables.) A $(\mathcal{Q},\mathcal{R})$-*definition scheme* is a tuple of formulas of the form :

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

where $k > 0$, $\mathcal{Q}^*k := \{(q,\vec{j}) \mid q \in \mathcal{Q}, \vec{j} \in [k]^{\rho(q)}\}$, $\varphi \in MS(\mathcal{R},\mathcal{W})$, $\psi_i \in MS(\mathcal{R},\mathcal{W}\cup\{x_1\})$ for $i = 1,\cdots,k$, $\theta_w \in MS(\mathcal{R},\mathcal{W}\cup\{x_1,\cdots,x_{\rho(q)}\})$, for $w = (q,\vec{j}) \in \mathcal{Q}^*k$.

These formulas are intended to define a structure T in $STR(\mathcal{Q})$ from a structure S in $STR(\mathcal{R})$ and will be used in the following way. The formula φ defines the domain of the corresponding transduction ; namely, T is defined only if φ holds true in S for some assignment of values to the parameters. Assuming this condition fulfilled, the formulas $\psi_1,\cdots,\psi_k$, define the domain of T as the disjoint union of the sets $D_1,\cdots,D_k$, where D_i is the set of elements in the domain of S that satisfy ψ_i for the considered assignment. Finally, the formulas θ_w for $w = (q,\vec{j}), \vec{j} \in [k]^{\rho(q)}$ define the relation q_T. Here are the formal definitions.

Let $S \in STR(\mathcal{R})$, let γ be a $\mathcal{W}$-assignment in S. A $\mathcal{Q}$-structure T with domain $D_T \subseteq D_S \times [k]$ is *defined in* (S,γ) by Δ if :

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

(ii) $D_T = \{(d,i) \mid d \in D_S, i \in [k], (S,\gamma,d) \models \psi_i\}$

(iii) for each q in $\mathcal{Q}$:

$q_T = \{((d_1,i_1),\cdots,(d_t,i_t)) \in D_T^t \mid (S,\gamma,d_1,\cdots,d_t) \models \theta_{(q,\vec{j})}\}$,

where $\vec{j} = (i_1,\cdots,i_t)$ and $t = \rho(q)$.

(By $(S,\gamma,d_1,\cdots,d_t) \models \theta_{(q,\vec{j})}$, we mean $(S,\gamma') \models \theta_{(q,\vec{j})}$, where γ' is the assignment extending γ, such that $\gamma'(x_i) = d_i$ for all $i = 1,\cdots,t$; 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, i.e., whenever $(S,\gamma) \models \varphi$, we can use the functional notation $def_\Delta(S,\gamma)$ for T.

The *transduction defined by* Δ is the relation $def_\Delta := \{(S,T) \mid T = def_\Delta(S,\gamma)$ for some $\mathcal{W}$-assignment γ in $S\} \subseteq STR(\mathcal{R})\times STR(\mathcal{Q})$. A transduction $f \subseteq STR(\mathcal{R})\times STR(\mathcal{Q})$ is *definable* if it is equal to def_Δ for some $(\mathcal{Q},\mathcal{R})$-definition scheme Δ. In the case where $\mathcal{W} = \emptyset$, we say that f is *definable without parameters* (note that it is functional). We shall refer to the integer k by saying that def_Δ is k-copying ; if $k = 1$ we say that it is *noncopying* and we can write more simply Δ as $(\varphi,\psi,(\theta_q)_{q\in\mathcal{Q}})$. In this case :

$D_T = \{d \in D_S \mid (S,\gamma,d) \models \psi\}$ and for each q in $\mathcal{Q}$

$q_T = \{(d_1,\cdots d_t) \in D_T^t \mid (S,\gamma,d_1,\cdots d_t) \models \theta_q\}$, where $t = \rho(q)$.

Before applying these definitions to general hypergraphs, we give three examples. Our first example is the transduction that associates with a graph the set of its connected components. A graph G is represented by $\mid G \mid_2$ (see Section 2).

The definition scheme Δ given below uses a parameter X. It is constructed in such a way that $def_\Delta(\mid G \mid_2, \{x\})$ is the structure $\mid G' \mid_2$ where G' is the connected component of G containing the vertex x. We let $(\varphi, \psi, \theta_{inc})$ be the noncopying definition scheme where :

φ is a formula with free variable X expressing that X consists of a unique vertex,

ψ is a formula with free variables X and x expressing that either x is a vertex linked by a path (where edges can be traversed in either direction) to the vertex in X, or x is an edge, one end of which is linked by such a path to the vertex in X,

θ_{inc} is the formula $inc(x_1, x_2, x_3)$.

It is straightforward to verify that Δ is as desired.

Our second example is the functional transduction that maps a word u in $\{a, b\}^+$ to the word u^3. (We denote by $\{a, b\}^+$ the set of nonempty words written with a and b.) In order to define it as a transduction of structures, we represent a word u in $\{a, b\}^+$ by the structure $\parallel u \parallel$ defined in Subsection 1.1. We let Δ be the 3-copying definition scheme without parameter $(\varphi, \psi_1, \psi_2, \psi_3, (\theta_{(suc,i,j)})_{i,j=1,2,3}, (\theta_{(lab_a,i)})_{i=1,2,3}, (\theta_{(lab_b,i)})_{i=1,2,3})$ such that :

φ expresses that the input structure indeed represents a word in $\{a, b\}^+$,

ψ_1, ψ_2, ψ_3 are identical to the Boolean constant *true*,

$\theta_{(suc,i,j)}(x_1, x_2)$ is $suc(x_1, x_2)$ if $i = j$,

$\theta_{(suc,i,j)}(x_1, x_2)$ expresses that x_1 is the last position and that x_2 is the first one if $i = 1$ and $j = 2$, or if $i = 2$ and $j = 3$,

$\theta_{(suc,i,j)}(x_1, x_2)$ is the constant *false* if $i \neq j$ and if $(i, j) \notin \{(1, 2), (2, 3)\}$,

$\theta_{(lab_a,i)}(x_1)$ is $lab_a(x_1)$ for $i = 1, 2, 3$,

$\theta_{(lab_b,i)}(x_1)$ is $lab_b(x_1)$ for $i = 1, 2, 3$.

We claim that $def_\Delta(S) = T$ if and only if S is a structure of the form $\parallel u \parallel$ and T is the corresponding structure $\parallel u^3 \parallel$. To take a simple example, if $S = < \{1, 2, 3, 4\}, suc, lab_a, lab_b >$ representing the word *abba*, then $def_\Delta(S)$ is the structure

$$T = < \{1_1, 2_1, 3_1, 4_1, 1_2, 2_2, 3_2, 4_2, 1_3, 2_3, 3_3, 4_3\}, suc', lab'_a, lab'_b >$$

where we write i_j instead of (i, j), so that i_j is the jth copy if i for $j = 1, 2, 3$ and

$suc'(i_j, k_m)$ holds if and only if k_m is the successor of i_j in the above enumeration of the domain of T,

$lab'_a(i_j)$ holds if $i \in \{1,4\}, j \in \{1,2,3\}$ and

$lab'_b(i_j)$ holds otherwise.

This is an example of a definable transduction from words to words that is not a rational one. (This transduction will also be used as a counterexample in Proposition 4.6 below).

Our last example is the product of a finite-state automaton $\mathcal{A}$ by a *fixed* finite-state automaton $\mathcal{B}$. A finite-state automaton is defined as a 5-tuple $\mathcal{A} = < A, Q, M, I, F >$ where A is the input alphabet, (there we take $A = \{a,b\}$), Q is the set of states, M is the transition relation which is here a subset of $Q \times A \times Q$ (because we consider nondeterministic automata without ε-transitions), I is the set of initial states and F is that of final states. The language recognized by $\mathcal{A}$ is denoted by $L(\mathcal{A})$. The automaton $\mathcal{A}$ is represented by the relational structure : $\mid \mathcal{A}\mid = < Q, trans_a, trans_b, I, F >$ where $trans_a$ and $trans_b$ are binary relations and :

$trans_a(p,q)$ holds if and only if $(p,a,q) \in M$,

$trans_b(p,q)$ holds if and only if $(p,b,q) \in M$.

Let $\mathcal{B} = < A, Q', M', I', F' >$ be a similar automaton, and $\mathcal{A} \times \mathcal{B} = < A, Q \times Q', M'', I \times I', F \times F' >$ be the product automaton intended to define the language $L(\mathcal{A}) \cap L(\mathcal{B})$. We let Q' be $\{1, \cdots, k\}$ (let us recall that $\mathcal{B}$ is fixed). We shall define a definition scheme Δ such that $def_\Delta(\mid \mathcal{A}\mid) = \mid \mathcal{A} \times \mathcal{B}\mid$; it will be k-copying since the set of states of $\mathcal{A} \times \mathcal{B}$ is $Q \times \{1, \cdots, k\}$. We let $\Delta = (\varphi, \psi_1, \cdots, \psi_k, (\theta_w)_{w \in R^* k})$, where $R = \{trans_a, trans_b, I, F\}$ and:

φ is the constant *true* (because every structure in $STR(\mathcal{R})$ represents an automaton ; this automaton may have inaccessible states and useless transitions),

$\psi_1, \cdots, \psi_k$ are the constant *true*,

$\theta_{(trans_a,i,j)}(x_1,x_2)$ is the formula $trans_a(x_1,x_2)$ if (i,a,j) is a transition of $\mathcal{B}$ and is the constant *false* otherwise,

$\theta_{(trans_b,i,j)}$ is defined similarly,

$\theta_{(I,i)}(x_1)$ is the formula $I(x_1)$ if i is an initial state of $\mathcal{B}$ and is *false* otherwise,

$\theta_{(F,i)}(x_1)$ is defined similarly.

Note that the language defined by an automaton $\mathcal{A}$ is nonempty if and only if there is a path in $\mathcal{A}$ from some initial state to some final state. This latter property is expressible in monadic second-order logic. Hence it follows from Proposition 4.3 below that, for a fixed rational language K, the set of structures representing an automaton $\mathcal{A}$ such that $L(\mathcal{A}) \cap K$ is nonempty is definable. This construction is used systematically in Courcelle [17].

The definitions concerning definable transductions of structures apply to hypergraphs via their representation by relational structures as explained above. However, since we

have two representations of hypergraphs by logical structures, we must be more precise. We say that a *hypergraph transduction*[c], i.e., a binary relation f on hypergraphs is *(i,j)-definable*, where i and j belong to $\{1,2\}$ if and only if the transduction of structures $\{(\mid G \mid_i, \mid G' \mid_j) \ / \ (G,G') \in f\}$ is definable. We shall also use transductions from trees to hypergraphs. Since a tree t is a graph, it can be represented either by $\mid t \mid_1$ or by $\mid t \mid_2$. However, both structures are equally powerful for expressing monadic second-order properties of trees and definable transductions from trees to trees and from trees to graphs. (This follows from the proof of Theorem 2.4). When we specify a transduction involving trees (or words which are special trees) as input or output we shall use the symbol * instead of the integers 1 and 2, in order to recall that the choice of representation is not important in these cases. We shall use *definable* for (*,*)-*definable*.

Here are a few facts concerning definable transductions of structures.

4.1.1. Fact : *If f is a definable transduction, there exists an integer k such that $Card(D_T) \leq kCard(D_S)$ whenever T belongs to $f(S)$.*

4.1.2. Fact : *The domain of a definable transduction is MS-definable.*

Proof : Let Δ be a definition scheme as in the general definition with $\mathcal{W} = \{X_1, \cdots, X_n\}$. Then $Dom(def_\Delta) = \{S \ / \ S \models \exists X_1, \cdots, X_n \varphi\}$. □

The next two propositions list examples of transductions of words, trees and graphs that are definable.

Proposition 4.1 *The following mappings are definable transductions : (1) word homomorphisms, (2) inverse nonerasing word homomorphisms, (3) gsm mappings, (4) the mirror-image mapping on words, (5) the mapping $\lambda u.[u^n]$ where u is a word and n a fixed integer, (6) the mapping that maps a derivation tree relative to a fixed context-free grammar to the generated word, (7) linear root-to-frontier or frontier-to-root tree transductions.*

The proofs are easy to do. Let us recall that a *gsm mapping* is a transduction from words to words defined by a *generalized sequential machine*, i.e., a (possibly nondeterministic) transducer that reads at least one input symbol and outputs and a (possibly empty) word on each move. A special case of (5) has been constructed in detail in our second example before Fact 4.1.1. See Gecseg and Steinby [42] or the survey by Raoult [54] for tree transductions. □

Fact 4.1.1 which limits the sizes of the output structures, shows that certain transductions are *not definable*. This is the case of inverse erasing word homomorphisms and of ground tree transducers except in degenerated cases (see Dauchet *et al.* [28] or [54] on ground tree transducers).

[c] Graphs are special hypergraphs. We shall speak of graph transduction if appropriate.

Proposition 4.2 *The transductions that associate with a graph G : (1) its spanning forests, (2) its connected components, (3) its subgraphs satisfying some fixed 2-definable property, (4) its maximal subgraphs satisfying some fixed MS_2-definable property (maximal for subgraph inclusion), (5) the graph consisting of the union of two disjoint copies of G, (6) its minors, are all (2,2)-definable. The mapping associating with a graph its line graph is (2,1)-definable but not (2,2)-definable.*

We recall that the *line graph* of a graph G has E_G as set of vertices and has undirected edges between any two vertices representing edges sharing a vertex (in G).

Proof : Assertions (1)-(6) are easy consequences of the existence of an MS formula expressing that two given vertices are linked by some path. See Section 1. Assertion (6) is proved in Courcelle [12]. In the last assertion, the (2,1)-definability is an easy consequence of the definition of a line graph. See [23] for the negative part of this assertion. □

4.2 The fundamental property of definable transductions

The following proposition is the basic fact behind the notion of semantic interpretation ([53]). It says that if $T = def_\Delta(S, \mu)$ i.e., if T is defined in (S, μ) by Δ, then the monadic second-order properties of T can be expressed as monadic second-order properties of (S, μ). The usefulness of definable transductions is based on this proposition.

Let $\Delta = (\varphi, \psi_1, \cdots, \psi_k, (\theta_w)_{w \in Q^* k})$ be a $(\mathcal{Q}, \mathcal{R})$-definition scheme, written with a set of parameters $\mathcal{W}$. Let $\mathcal{V}$ be a set of set variables disjoint from $\mathcal{W}$. For every variable X in $\mathcal{V}$, for every $i = 1, \cdots, k$, we let X_i be a new variable. We let $\mathcal{V}' := \{X_i / X \in \mathcal{V}, i = 1, \cdots, k\}$. For every mapping $\eta : \mathcal{V}' \to \mathcal{P}(D)$, we let $\eta^k : \mathcal{V} \to \mathcal{P}(D \times [k])$ be defined by $\eta^k(X) = \eta(X_1) \times \{1\} \cup \cdots \cup \eta(X_k) \times \{k\}$. With these notations we can state :

Proposition 4.3 *For every formula β in $MS(Q, \mathcal{V})$ one can construct a formula β' in $MS(\mathcal{R}, \mathcal{V}' \cup \mathcal{W})$ such that, for every S in $STR(\mathcal{R})$, for every assignment $\mu : \mathcal{W} \to S$ for every assignment $\eta : \mathcal{V} \to S$, we have :*

$def_\Delta(S, \mu)$ is defined (if it is, we denote it by T), η^k is a $\mathcal{V}$-assignment in T, and $(T, \eta^k) \models \beta$
if and only if $(S, \eta \cup \mu) \models \beta'$.

Note that, even if T is well-defined, the mapping η^k is not necessarily a $\mathcal{V}$-assignment in T, because $\eta^k(X)$ is not necessarily a subset of the domain of T which is a possibly proper subset of $D_S \times [k]$.

Proof sketch : Let us first consider the case where def_Δ is noncopying. In order to transform β into β', one replaces every atomic formula $q(u_1, \cdots, u_n)$ by the formula $\theta_q(u_1, \cdots, u_n)$ which defines it in terms of the relations of S (See Lemma 1.5). One also restricts quantifications to the domain of T, that is, one replaces $\exists x[\mu]$ by $\exists x[\psi(x) \wedge \mu]$ and $\exists X[\mu]$ by

$\exists X[\forall x\{x \in X \Longrightarrow \psi(x)\} \wedge \mu]$. (See Lemma 1.6). In the case where $k > 1$, one replaces $\exists X[\mu]$ by a formula of the form $\exists X_1, \exists X_2, \cdots, \exists X_k[\mu']$ where μ' is an appropriate transformation of μ based on the fact that $X = X_1 \times \{1\} \cup \cdots \cup X_k \times \{k\}$. The reader will find a complete construction in [13], Proposition 2.5, p. 166. □

From this proposition, we get easily :

Proposition 4.4 *(1) The inverse image of an MS-definable class of structures under a definable transduction is MS-definable.*
(2) The composition of two definable transductions is definable.

Proof : (1) Let $L \subseteq STR(\mathcal{Q})$ be defined by a closed formula β and def_Δ be a transduction as in Proposition 4.3. Then $def_\Delta^{-1}(L) \subseteq STR(\mathcal{R})$ is defined by the formula $\exists Y_1, \cdots, Y_n[\beta']$ where $Y_1, \cdots, Y_n$ are the parameters and β' is constructed from β as in Proposition 4.3.

(2) Let $\Delta = (\varphi, \psi_1, \cdots, \psi_k, (\theta_w)_{w \in \mathcal{Q}_{*k}})$ be a k-copying definition scheme and $\Delta' = (\varphi', \psi'_1, \cdots, \psi'_{k'}, (\theta'_w)_{w \in \mathcal{P}*k'})$ be a k'-copying such that def_Δ is a transduction from $STR(\mathcal{R})$ to $STR(\mathcal{Q})$ and $def_{\Delta'}$ is a transduction from $STR(\mathcal{Q})$ to $STR(\mathcal{P})$. Let f be the transduction $def_{\Delta'} o def_\Delta$ from $STR(\mathcal{R})$ to $STR(\mathcal{P})$: we shall construct a definition scheme Δ" for it. Just to simplify the notation we shall assume that the parameters of Δ are Y and Y' and that those of Δ' are Z and Z'. We shall also assume that the relations of $\mathcal{P}$ are all binary. The general case will be an obvious extension.

In order to describe Δ" we shall denote by S an $\mathcal{R}$-structure, we shall denote by T the $\mathcal{Q}$-structure $def_\Delta(S, Y, Y')$ where Y and Y' are subsets of D_S and we denote by U the $\mathcal{P}$-structure $def_{\Delta'}(T, Z, Z')$ where Z and Z' are subsets of D_T. Hence D_T is a subset of $D_S \times [k]$, and D_U is a subset of $D_S \times [k] \times [k']$ which is canonically isomorphic to a subset of $D_S \times [kk']$. Hence Δ" will be kk'-copying. The parameters Z and Z' represent sets of the respective forms $Z = Z_1 \times \{1\} \cup \cdots \cup Z_k \times \{k\}$ and $Z' = Z'_1 \times \{1\} \cup \cdots \cup Z'_k \times \{k\}$. Hence the definition scheme Δ" will be written in terms of parameters $Y, Y', Z_1, \cdots, Z_k, Z'_1, \cdots, Z'_k$. It will be of the form :

$$(\varphi", (\psi"_{i,j})_{i \in [k], j \in [k']}, (\theta"_{(p,(i,i'),(j,j'))})_{p \in \mathcal{P}, i,i' \in [k] jj' \in [k']})$$

so that the domain of D_U will be handled as a subset of $D_S \times [k] \times [k']$ and not of $D_S \times [kk']$. The formulas forming Δ" will be obtained from those forming Δ' by the transformation of Proposition 4.3. We first consider φ" which should express that $def_\Delta(S, Y, Y')$ is defined, i.e., that $(S, Y, Y') \models \varphi$, and that if $Z = Z_1 \times \{1\} \cup \cdots \cup Z_k \times \{k\}$ and $Z' = Z'_1 \times \{1\} \cup \cdots \cup Z'_k \times \{k\}$, then $def_\Delta(T, Z, Z')$ is defined, i.e., that $(T, Z, Z') \models \varphi$ which is equivalent to :

$$(S, Y, Y', Z_1, \cdots, Z_k, Z'_1, \cdots, Z'_k) \models \tau$$

where τ is obtained from φ' by the transformation of Proposition 4.3. Hence φ'' is the conjunction of φ and τ. We omit the constructions of the other formulas because they are quite similar. □

We could define more powerful transductions by which a structure T would be constructed "inside" $S \times S$ instead of "inside" a structure formed of a fixed number of disjoint copies of S (like in [40]. However, with this variant, one could construct a *second-order* formula β' as in Proposition 4.3 (with quantifications on binary relations), but not a *monadic* second-order one (at least in general). We wish to avoid non monadic second-order logic because most constructions and decidability results (like those of [11], [12]) break down. In the third example given before Fact 4.1.1, we have shown that the transduction associating the automaton $\mathcal{A} \times \mathcal{B}$ with an automaton $\mathcal{A}$ is definable (via the chosen representation of finite-state automata by relational structures) *for fixed* $\mathcal{B}$.

Here are some other closure properties of the class of definable transductions.

Proposition 4.5 *The intersection of a definable transduction with a transduction of the form $A \times B$ where A and B are MS-definable classes of structures, is a definable transduction.*

Proof : Straightforward. See [13] □

Proposition 4.6 *(1) The image of an MS-definable class of structures under a definable transduction is not MS-definable in general.*

(2) The inverse of a definable transduction is a transduction that is not definable in general.

(3) The intersection of two definable transductions is a transduction that is not definable in general.

Proof : (1) The transduction of words that maps $a^n b$ to $a^n b a^n b a^n b$ for $n \geq 0$ is definable (this follows from Proposition 4.5 and the second example given before Fact 4.1.1). The image of the definable language a^*b is a language that is not regular (and even not context-free), hence not definable by the result of Büchi and Elgot ([62], Thm 3.2) saying that a set of words is MS-definable iff it is regular.
(2) The inverse of this transduction is not definable since, if it were, its domain is would be definable (by Fact 4.1.2), hence regular, which is not the case.
(3) The intersection of the definable transductions of words that map $a^n b^m$ to c^n, and $a^n b^m$ to c^m is the one that maps $a^n b^n$ to c^n. It is not definable because its domain is not a an MS-definable (regular) language. □

4.3 Comparisons of representations of partial orders, graphs and hypergraphs by relational structures

We shall formulate in terms of definable transductions some transformations between several relational structures representing the same object .

4.3.1 Fact : *The transduction of $< D, \leq_D >$ into $< D, suc_D >$ where $< D, \leq_D >$ is a partial order is definable, and so is its inverse. The transduction of $\mid G \mid_2$ into $\mid G \mid_1$ where G is a*

graph or a hypergraph is definable.

It follows in particular from Proposition 4.4 that a class of finite partial orders is MS-definable w.r.t. one of the representations $< D, \leq_D >$ or $< D, suc_D >$ iff it is w.r.t. the other (this has been already proved in Proposition 2.1). It follows also that a class of graphs or hypergraphs is MS_2-definable if it is MS_1-definable. We shall prove the converse for any subclass of $\mathcal{C}_k$, the class of simple directed graphs having a semi-strong k-coloring, where k is any (fixed) integer (see Subsection 2.2).

Lemma 4.7 *Let $k \in \mathcal{N}$. One can construct a definable transduction δ_k that associates with $\mid G \mid_1$ a structure T isomorphic to $\mid G \mid_2$, for every graph G in $\mathcal{C}_k$.*

Proof : Let G be a simple directed graph with a semi-strong coloring $\gamma : V_G \to \{1, \cdots, k\}$. A mapping $\gamma : V_G \to \{1, \cdots, k\}$ can be specified by the k-tuple $(V_1, \cdots, V_k)$ of subsets of V_G such that $V_i = \gamma^{-1}(i)$ for every $i = 1, \cdots, k$. Thus a formula $\pi \in MS(\mathcal{R}_s, \{X_1, \cdots, X_k\})$ can express that a given tuple $(V_1, \cdots, V_k)$ represents a semi-strong k-coloring. It follows that the class $\mathcal{C}_k$ is MS-definable (the corresponding formula is π' defined as $\exists X_1, \ldots, X_k.\pi$).

Let $G \in \mathcal{C}_k$ and $(V_1, \cdots, V_k)$ represent a semi-strong k-coloring. We let $rep : E_G \to E \subseteq V_G \times \{1, \cdots, k\}$ be the bijection such that : $rep(e) = (v, i)$ iff e links w to v where w is a vertex in V_i. It is one-to-one since $(V_1, \cdots, V_k)$ represents a semi-strong coloring (the origins of two edges with same target have different colors). We shall thus construct T isomorphic to $\mid G \mid_2$ with domain $V_G \times \{0\} \cup E \subseteq V_G \times \{0, 1, \cdots, k\}$ as image of $\mid G \mid_1$ by a definable transduction. We let Δ be the following definition scheme with parameters $X_1, \cdots, X_k$:

$$\Delta = (\pi', \psi_0, \psi_1, \cdots, \psi_k, (\theta_{(inc,j)})_{j \in [0,k]^3}),$$

where π' has already been defined, and :

ψ_0 is : *true*

ψ_i is : $\exists w[w \in X_i \wedge edg(w, x_1)]$, for $i = 1, \cdots, k$

$\theta_{(inc,i,0,0)}$ is : $x_1 = x_3 \wedge edg(x_2, x_3) \wedge x_2 \in X_i$ for $i = 1, \cdots, k$ and

$\theta_{(inc,n,m,p)}$ is : *false* if $n = 0$ or $m \neq 0$ or $p \neq 0$.

Claim : *For every graph G in $\mathcal{C}_k$, for every* k*-tuple $V_1, \cdots, V_k$ representing a semi-strong* k*-coloring of G, the structure $def_\Delta(\mid G \mid_1, V_1, \cdots, V_k)$ is isomorphic to $\mid G \mid_2$.*

Proof of Claim : Let $T = def_\Delta(\mid G \mid_1, V_1, \cdots, V_k)$. Then $D_T = V_G \times \{0\} \cup E$ from the definitions. From the definition of the formulas $\theta_{(inc,j)}$ we have :
$((x_1, n), (x_2, m), (x_3, p)) \in inc_T$ iff
$n \neq 0$ and $m = p = 0$ and $edg(x_2, x_3)$ and $x_2 \in X_n$ and $x_1 = x_3$ iff
$n \neq 0$ and $m = p = 0$ and $rep(e) = (x_1, n)$ where e is the edge of G that links x_2 to x_3.

It follows that $\mid G \mid_2$ is isomorphic to T by the isomorphism mapping $v \in V_G$ onto $(v, 0)$ and $e \in E_G$ onto $rep(e)$. □

The lemma follows immediately. □

From Lemma 4.7, one obtains immediately Lemma 2.5 because any two isomorphic structures satisfy the same formulas.

5 Equational sets of graphs and hypergraphs

The easiest way to define context-free sets of graphs is by systems of recursive equations. These equations are written with set union and the set extensions of operations on graphs that generalize the concatenation of words. By these operations one can generate all finite graphs from elementary graphs more or less as one can generate finite nonempty words from the letters, by means of concatenation. This idea also applies to hypergraphs. In this section, we first present systems of equations in a general algebraic setting. Then we review the operations on graphs and hypergraphs that yield algebraic presentations of the VR sets of graphs and of the HR sets of hypergraphs in terms of systems of recursive set equations. The grammatical definitions of these sets, by context-free graph grammars of various types are investigated in detail in other chapters of this book. The relevant operations are presented in a uniform way in terms of operations on structures. This will be helpful for obtaining theorems relating MS logic and context-free graph grammars.

We first review equational sets in a general algebraic setting ; then we define operations on graphs yielding VR sets of graphs ; then we define operations on hypergraphs yielding the HR sets of hypergraphs. We also state logical characterizations of the VR sets of graphs and of the HR sets of hypergraphs from which follow closure properties under definable transductions.

5.1 Equational sets

As in many other works, we shall use the term *magma* borrowed from Bourbaki [7] for what is usually called an *abstract algebra* or an *algebra*. The words "algebra" and "algebraic" are used in many different contexts with different meanings. We prefer to avoid them completely and use fresh words. Many-sorted notions are studied in detail by Ehrig and Mahr [32], Wirsing [66] and Wechler [65]. We mainly review the notation. We shall use *infinite* sets of sorts and *infinite* signatures, which is not usual.

F-magmas

Let $\mathcal{S}$ be a set called the set of *sorts*. An $\mathcal{S}$-*signature* is a set F given with two mappings $\alpha : F \to seq(\mathcal{S})$ (the set of finite sequences of elements of $\mathcal{S}$), called the *arity* mapping, and $\sigma : F \to \mathcal{S}$, called the *sort* mapping. The length of $\alpha(f)$ is called the *rank* of f, and is denoted by $\rho(f)$. The *type* of f in F is the pair $(\alpha(f), \sigma(f))$ that we shall rather write $s_1 \times s_2 \times \cdots \times s_n \to s$ where $\alpha(f) = (s_1, \cdots, s_n)$ and $\sigma(f) = s$. (We may have $n = 0$; in

this case f is a constant). If $\mathcal{S}$ has only one sort, we say that F is a *ranked alphabet*, and the arity of a symbol is completely defined by its rank.

An *F-magma*, i.e. an *F-algebra* ([32], [66], [65]), is an object $M = < (M_s)_{s\in\mathcal{S}}, (f_M)_{f\in F} >$, where for each s in $\mathcal{S}$, M_s is a nonempty set, called the *domain of sort* s of M, and for each $f \in F$, the object f_M is a total mapping : $M_{\alpha(f)} \to M_{\sigma(f)}$. These mappings are called the *operations* of M. (For a nonempty sequence of sorts $\mu = (s_1, \cdots, s_n)$, we let $M_\mu := M_{s_1} \times M_{s_2} \times \cdots \times M_{s_n}$). We assume that $M_s \cap M_{s'} = \emptyset$ for $s \neq s'$. We let M also denote $\cup\{M_s / s \in \mathcal{S}\}$, and for $d \in M$, we let $\sigma(d)$ denote the unique s such that $d \in M_s$.

If M and M' are two F-magmas, a homomorphism $h : M \to M'$ is a mapping h that maps M_s into M'_s for each sort s, and commutes with the operations of F. We shall call it an *F-homomorphism* if it is useful to specify the signature F. We denote by $T(F)$ the *initial F-magma*, and by $T(F)_s$ its domain of sort s. This set can be identified with the set of well-formed ground terms over F of sorts s. If M is an F-magma, we denote by h_M the unique homomorphism : $T(F) \to M$. If $t \in T(F)_s$, then the image of t under h_M is an element of M_s, also denoted by t_M. One can consider t as a term *denoting* t_M, and t_M as *the value of* t in M. We say that F *generates* a subset M' of M if M' is the set of values of the terms in $T(F)$. We say that M' is *finitely generated* if it is the set of values of terms in $T(F')$ where F' is a finite subset of F.

An *$\mathcal{S}$-sorted set of variables* is a pair (X, σ) consisting of a set X, and a *sort mapping* $\sigma : X \to \mathcal{S}$. It will be more simply denoted by X, unless the sort mapping must be specified. We shall denote by $T(F, X)$ the set of well-formed terms written with $F \cup X$ and by $T(F, X)_s$ the subset of those of sort s. Hence, $T(F, X) = T(F \cup X)$ and $T(F, X)_s = T(F \cup X)_s$. However, the notations $T(F, X)$ and $T(F, X)_s$ are more precise because they specify the variables among the nullary symbols of $F \cup X$. Let $\mathcal{X}$ be a finite sequence of pairwise distinct variables from X. We shall denote by $T(F, \mathcal{X})_s$, the set of terms of $T(F, X)_s$ having all their variables in the list $\mathcal{X}$. If $t \in T(F, \mathcal{X})_s$, we denote by $t_{M,\mathcal{X}}$ the mapping : $M_{\sigma(\mathcal{X})} \to M_s$ associated with t in the obvious way, by letting a symbol f from F denote f_M, (where $\sigma(\mathcal{X})$ denotes the sequence of sorts of the elements of the sequence $\mathcal{X}$). We call $t_{M,\mathcal{X}}$ a *derived operation of* M. If $\mathcal{X}$ is known from the context, we write t_M instead of $t_{M,\mathcal{X}}$. This is the case in particular if t is defined as a member of $T(F, \{x_1, \cdots, x_k\})_s$: the sequence $\mathcal{X}$ is implicitely $(x_1, \cdots, x_k)$.

Power-set magmas and polynomial systems

Let F be an $\mathcal{S}$-signature. We enlarge it into F_+ by adding, for every sort s in $\mathcal{S}$, a new symbol $+_s$ of type : $s \times s \to s$, and a new constant Ω_s of sort s. With an F-magma M we associate its *power-set magma* :

$\mathcal{P}(M) :=< (\mathcal{P}(M_s))_{s\in\mathcal{S}}, (f_{\mathcal{P}(M)})_{f\in F_+} >$

where

$\Omega_{s\mathcal{P}(M)} = \emptyset$

$A_1 +_{s\mathcal{P}(M)} A_2 := A_1 \cup A_2$ (for $A_1, A_2 \subseteq M_s$),

$f_{\mathcal{P}(M)}(A_1, \cdots, A_k) :=^{\mathcal{P}} f_M(A_1, \cdots, A_k)$, i.e. $:= \{f_M(a_1, \cdots, a_k)/a_1 \in A_1, \cdots, a_k \in A_k\}$

for $A_1 \subseteq M_{s_1}, \cdots, A_k \subset M_{s_k}$ where $\alpha(f) = (s_1, \cdots, s_k)$. (We denote by ${}^{\mathcal{P}}g$ the set extension of a mapping g.)

Hence $\mathcal{P}(M)$ is an F_+-magma. A *polynomial system* over F is a sequence of equations $S = < u_1 = p_1, ..., u_n = p_n >$, where $U = \{u_1, \cdots, u_n\}$ is an $\mathcal{S}$-sorted set of variables called the set of *unknowns* of S. Each term p_i is a *polynomial*, i.e., a term of the form Ω_s or $t_1 +_s t_2 +_s \cdots +_s t_m$, where the terms t_j are monomials of sort $s = \sigma(u_i)$. A *monomial* is a term in $T(F \cup U)$. The subscript s is usually omitted in $+_s$ and in Ω_s. A mapping $S_{\mathcal{P}(M)}$ of $\mathcal{P}(M_{\sigma(u_1)}) \times \cdots \times \mathcal{P}(M_{\sigma(u_n)})$ into itself is associated with S and M as follows : for $A_1 \subseteq M_{\sigma(u_1)}, \cdots, A_n \subseteq M_{\sigma(u_n)}$, we let

$$S_{\mathcal{P}(M)}(A_1, \cdots, A_n) = (p_{1\mathcal{P}(M)}(A_1, \cdots, A_n), \cdots, p_{n\mathcal{P}(M)}(A_1, \cdots, A_n)).$$

A *solution* of S in $\mathcal{P}(M)$ is an n-tuple $(A_1, \cdots, A_n)$ such that $A_i \subseteq M_{\sigma(u_i)}$ for each $i = 1, \cdots, n$ and $(A_1, \cdots, A_n) = S_{\mathcal{P}(M)}(A_1, \cdots, A_n)$, i.e.,

$$A_i = p_{i\mathcal{P}(M)}(A_1, \cdots, A_n), \textit{for every } i = 1, \cdots, n. \tag{2}$$

A solution of S is also called a fixed-point of $S_{\mathcal{P}(M)}$. Every such system S has a least solution in $\mathcal{P}(M)$ denoted by $(L((S, M), u_1), \cdots, L((S, M), u_n))$. ("Least" is understood with respect to set inclusion). This n-tuple can be concretely described as follows :

$$L((S, M), u_i) = \cup\{A_i^j / j \geq 0\}$$

where $A_i^0 = \emptyset$ for all $i = 1, \cdots, n$ and $(A_1^{j+1}, \cdots, A_n^{j+1}) = S_{\mathcal{P}_{(M)}}(A_1^j, \cdots, A_n^j)$

The components of the least solutions in $\mathcal{P}(M)$ of polynomial systems are the *M-equational* sets. We denote by $Equat(M)$ the family of M-equational sets and by $Equat(M)_s$ the subfamily of those included in M_s.

We review the classical example of context-free grammars. Let A be a finite alphabet, say $A = \{a_1, \cdots, a_n\}$. Let $F_A = A \cup \{., \varepsilon\}$ be the ranked alphabet where $\rho(a_i) = 0$ for all $i, \rho(\varepsilon) = 0, \rho(.) = 2$. We denote by $\mathcal{W}_A$ the F_A-magma $< A^*, ., \varepsilon, a_1, \cdots, a_n >$ where A^* is the set of words over A, . is the concatenation, ε is the empty word, $a_1, \cdots, a_n$ denote themselves as words. It is a monoid (with a binary associative operation having a unit) augmented with constants. The set $Equat(\mathcal{W}_A)$ is the set of *context-free languages* over A. This follows from the theorem of Ginsburg and Rice [43] that characterizes these languages as the components of the least solutions of systems of recursive equations written with the nullary symbols $\varepsilon, a_1, \cdots, a_n$, the concatenation and, of course, set union (denoted here by $+$). Take for example, the context-free grammar $G = \{u \to auuv, u \to avb, v \to avb, v \to ab\}$ with nonterminal symbols u and v and terminal symbols a and b. The corresponding system of equations is :

$$S = < u = a.(u.(u.v)) + a.(v.b), v = a.(v.b) + a.b > .$$

We now give an example concerning trees. By a *tree* T we mean a finite connected undirected graph without multiple edges and cycles. The set of trees is denoted by $\mathcal{T}$. A *rooted tree* is a pair $R = (T, r)$ consisting of a tree T and a distinguished node r called the *root*. The set of rooted trees is denoted by $\mathcal{R}$. The set of nodes of a rooted or unrooted tree T is denoted by N_T. Any two isomorphic trees are considered as equal. (The set of trees is actually a quotient set w.r.t. isomorphism but we shall not detail this technical point).

We now define a few operations on trees and rooted trees. The types of these operations will be given in terms of two sorts, t and r, namely, the sort t of trees and the sort r of rooted trees. The first operation is the *root gluing* $// : r \times r \to r$. For S and T in $\mathcal{R}$, we let $S//T$ be the rooted tree obtained by fusing the roots of S and T (or rather, of two disjoint isomorphic copies of S and T). The second operation is the *extension* denoted by $ext : r \to r$. For T in $\mathcal{R}$, we let $ext(T)$ be the rooted tree obtained from T by the addition of a new node that becomes the root of $ext(T)$, linked by a new edge to the root of T. We denote by 1 the rooted tree consisting of a single node (the root). Finally, we let $fg : r \to t$ be the mapping that "forgets" the root of a rooted tree R. Formally, $fg(R) = T$ where $R = (T, r) \in \mathcal{R}$.

Hence, we have an $\{r, t\}$-sorted signature $F := \{//, ext, 1, fg\}$ and a many-sorted F-magma $\mathcal{TRECE}$ having $\mathcal{R}$ as domain of sort $r, \mathcal{T}$ as a domain of sort t, and the operations defined above. Hence, $\mathcal{TRECE} = < \mathcal{R}, \mathcal{T}, //, ext, 1, fg >$.

The set of trees of odd degree, i.e., such that the degree of every node is odd, is equal to $L((S, \mathcal{TRECE}), u)$ where S is the system

$$S = \begin{cases} u = & fg(ext(v)) \\ v = & w//w + v//w//w \\ w = & ext(v) + ext(1) \end{cases}$$

The sorts of u, v and w are t, r and r. It is not hard to see that $L((S, \mathcal{TRECE}), u)$ is the set of (finite) trees of odd dregree. As a hint observe that $L((S, \mathcal{TRECE}), v)$ is the set of rooted trees different from 1, all nodes of which except the root have odd degree, and that $L((S, \mathcal{TRECE}), w)$ is the set of rooted trees such that all nodes have odd degree and the root has degree one.

The following result is due to Mezei and Wright [50].

Proposition 5.1 *Let M and M' be F-magmas. If $h : M \to M'$ is an F-homomorphism, if S is a polynomial system over F, then $L((S, M'), u) = h(L((S, M), u))$ for every unknown u.*

Proof: The homomorphism $h : M \to M'$ extends into an F_+-homomorphism ${}^{\mathcal{P}}h : \mathcal{P}(M) \to \mathcal{P}(M')$ defined by ${}^{\mathcal{P}}h(A) = \{h(a)/a \in A\}$ for $A \subseteq M_s, s \in \mathcal{S}$. It is easy to verify that for every $j \in \mathcal{N}$:

$$S^j_{\mathcal{P}(M')}(\emptyset, \cdots, \emptyset) =^{\mathcal{P}} h(S^j_{\mathcal{P}(M)}(\emptyset, \cdots, \emptyset))$$

where of course ${}^{\mathcal{P}}h(A_1, \cdots, A_n) = ({}^{\mathcal{P}}h(A_1), \cdots, {}^{\mathcal{P}}h(A_n))$ for every $A_1, \cdots, A_n \subseteq M$. (The proof is by induction on j, using the fact that ${}^{\mathcal{P}}h$ is an F_+-homomorphism). The result follows immediately. □

Here is a consequence of Proposition 5.1. (We recall that h_M denotes the unique homomorphism $T(F) \to M$).

Corollary 5.2 *For every polynomial system S and for every unknown u of S we have $L((S, M), u) = h_M(L((S, T(F)), u))$. The emptiness of $L((S, M), u)$ depends only on S and u and is decidable.*

This means that an equational set, defined as a component of the least solution of a polynomial system S, is the image of the corresponding component of the least solution of S in the F-magma of terms $T(F)$, under the canonical homomorphism. Hence the study of the equational subsets of M can, in a sense, be reduced to the study of the equational sets of terms and the homomorphism h_M. In particular, the emptiness of $L((S, T(F)), u)$ can be decided by tree automata techniques (see below) ; by the first assertion this set is nonempty iff $L((S, M), u)$ is nonempty, where M is *any* F-magma.

Sets of terms can be defined in several ways, by grammars of various types, by automata (usually called "tree automata"), or in terms of congruences on the magmas of terms $T(F)$. Of special importance is the class of recognizable subsets of $T(F)$ characterized in terms of finite-state tree automata, of regular tree-grammars, of congruences having finitely many classes and also of monadic-second order formula. (See Courcelle [22]). We do not recall these characterizations here ; we only refer the reader to the book [42] for automata, grammars and congruences and to [62], Thm 3.2 for the characterization in terms of MS formulas.

In this section, we shall use the following result by Mezei and Wright [50], [42].

Proposition 5.3 *: Let F be a finite signature. A subset of $T(F)$ is equational iff it is recognizable.*

The main part of its proof is the determinization of finite-state frontier-to-root tree-automata.

Corollary 5.4 *: Let M be an F-magma (where F may be infinite). A set L is M-equational iff $L = h_M(K)$ for a recognizable subset of $T(F')$ where F' is a finite subsignature of F.*

If $t \in K$ and $m = h_M(t)$ then t is a syntactic description of m, m is the value of t in M. If $m \in L$ is given, the parsing problem relative to the system of equations (equivalently the grammar) from which K comes, consists in finding t in K such that $m = h_M(t)$. If such t does not exist, then $m \notin L$. Otherwise the found t is the desired syntactic analysis of m.

5.2 Graphs with ports and VR sets of graphs

Graphs will be simple and directed, as in Section 1. They will have distinguished vertices called *ports* and designated by special labels. We let $\mathcal{C}$ be a countable set called the set of *port labels*. A *graph with ports* is a pair $G = < H, P >$ where H is a graph and $P \subseteq V_H \times C$ for some finite subset C of $\mathcal{C}$. We denote P by $port_G, V_H$ by V_G, E_H by E_G, inc_H by inc_G and edg_H by edg_G. We let also G^o denote H. A vertex v is a *p-port* if $(v,p) \in port_G$. We consider p as a label attached to v and marking it as a p-port. A vertex may have several port labels. We let $\tau(G)$ be the set of labels of ports of G and we call this set the *type* of G.

We denote by $\mathcal{GP}(C)$ the class of graphs with ports G such that $\tau(G) \subseteq C$ where C is a subset of $\mathcal{C}$. We let $\mathcal{GP}$ denote the class of all graphs with ports. We now define some operations on $\mathcal{GP}$. If $G, G' \in \mathcal{GP}$ and G, G' are disjoint (i.e., $V_G \cap V'_G = \emptyset$) then $G \oplus G'$ is the *disjoint union* of G and G', and is defined as $< G^o \cup G'^o,\ port_G \cup port_{G'} >$. Clearly $\tau(G \oplus G') = \tau(G) \cup \tau(G')$.

The next operation adds edges to a given graph with ports. If $p, q \in \mathcal{C}$ and $G \in \mathcal{GP}$, we let $add_{p,q}(G) =< H, port_G >$ where H is G^o augmented with the edges (x,y) such that :

x is a p-port, y is a q-port, $x \neq y, (x,y)$ is not already an edge of G^o.

Hence H is simple and the operation $add_{p,q}$ does not add loops to G. The type of $add_{p,q}(G)$ is equal to that of G and $add_{p,q}(G) = G$ if G has no p-port or no q-port. Let us note finally that the number of edges added to G by $add_{p,q}$ depends on the numbers of p-ports and of q-ports and is not uniformly bounded.

A third operation will be useful, in order to modify port labels. Let P be a finite subset of $\mathcal{C} \times \mathcal{C}$. For $G \in \mathcal{GP}$ we let : $mdf_P(G) = < G^o, port_G.P >$. A vertex v is a q-port of $mdf_P(G)$ iff it is a p-port of G for some p such that $(p,q) \in P$. We have $\tau(mdf_P(G)) = \{q \ / \ (p,q) \in P \text{ for some } p \in \tau(G)\}$. Two special cases of this operation will be useful and deserve special notation : if C is a finite subset of $\mathcal{C}$, we let for $G \in \mathcal{GP}(D)$, $fg_C(G) = mdf_P(G)$ where $P = \{(d,d) \ / \ d \in D - C\}$: this operation "forgets" the p-ports for $p \in C$. It simply "removes" the label p on the p-ports for each $p \in C$; hence $\tau(fg_C(G)) = \tau(G) - C$. If $p_1, p_2, \cdots, p_k, q_1, \cdots, q_k$ are labels in $\mathcal{C}$ with $p_1, \cdots, p_k$ pairwise distinct, we let for $G \in \mathcal{GP}(D)$, $ren_{p_1 \to q_1, p_2 \to q_2, \cdots, p_k \to q_k}(G) = mdf_P(G)$ where $P = \{(d,d) \ / \ d \in D - \{p_1, \cdots, p_k\}\} \cup \{(p_1, q_1), \cdots, (p_k, q_k)\}$. This operation "renames" the port labels p_i into q_i, for all $i = 1, \cdots, k$. Hence $\tau(ren_{p_1 \to q_1, \cdots, p_k \to q_k}(G)) = (\tau(G) - \{p_1, \cdots, p_k\}) \cup \{q_i \ / \ p_i \in \tau(G), 1 \leq i \leq k\}$.

Equational sets of graphs with ports

Having defined operations on $\mathcal{GP}$ we can now write systems of equations intended to define equational subsets of $\mathcal{GP}$. Here is an example. The equation

$$u = p + ren_{q \to p}(add_{p,q}(u \oplus q))$$

where p denotes the graph with a single vertex that is a p-port, and similarly for q, is intended to define (in the sense of Subsection 5.1) the set of finite tournaments without circuits, all vertices of which are p-ports. However we are facing a difficulty : the operation $\oplus$ is partial because $G \oplus G'$ is defined if and only if G and G' are disjoint. It follows for instance that $G \oplus G$ is undefined. We can make $\oplus$ into a total operation by letting G' be "replaced in $G \oplus G'$ by an isomorphic copy G'_1 disjoint with G". But now comes another difficulty : we may have $G \oplus (G' \oplus G'_1)$ not equal to $(G \oplus G') \oplus G'_1$ (depending on how the isomorphic copies of the second arguments of the operations $\oplus$ are chosen) which is clearly unsatisfactory. However they are isomorphic and, actually, we are interested in properties of graphs that are invariant under isomorphism. Let us call *concrete* a graph in $\mathcal{GP}$ and *abstract* the isomorphism class of such a graph. We shall denote by GP the class of abstract graphs with ports. We have thus a magma structure on GP because isomorphism is a congruence for the operations $\oplus, mdf_P$ and $add_{p,q}$. Graph grammars and systems of equations define subsets of GP. Logical formulas do the same because any two isomorphic relational structures satisfy the same formulas of any of the logical languages we have considered. However, in order to simplify the presentation we shall omit the distinction between $\mathcal{GP}$ and GP, and we shall do as if $\oplus$ was total on $\mathcal{GP}$.

We let F_{VR} denote the set of all operations $\oplus, mdf_P, fg_C, ren_{p_1 \to q_1, \cdots, p_k \to q_k}, add_{p,q}$ together with the constants p and p^ℓ for $p \in \mathcal{C}$. If $K \subseteq \mathcal{C}$, we denote by $F_{VR}(K)$ the subset of F_{VR} consisting of the above operations such that $P \subseteq K \times K, C \subseteq K, p_1, q_1, \cdots, p_k, q_k, p, q \in K$. The constant p^ℓ denotes a graph consisting of one loop, the unique vertex of which is a p-source. A term in $T(F_{VR})$ will be called a VR-*(graph) expression*. It denotes a graph with ports (actually an element of GP to be precise). We shall denote this graph by $val(t)$ for $t \in T(F_{VR})$.

Proposition 5.5 *: Every finite graph with port G is the value of some VR-expression in $T(F_{VR}(K))$ where $Card(K) \leq Max\{Card(V_G), Card(\tau(G))\}$.*

Proof sketch : For each vertex v of G one chooses a port label $p(v)$ (a different one for each vertex). One defines then G by :

$$G = mdf_P(...add_{p(v),p(v')}(...(... \oplus p(v) \oplus ...)))$$

where the sum $... \oplus p(v) \oplus ...$ extends to all $v \in V_G$, the operations $add_{p(v),p(v')}$ are inserted for all edges from v to v' and P is the set $\{(p(v), q)/(v, q) \in port_G\}$. One uses $p(v)^\ell$ instead of $p(v)$ if v has a loop. We omit further details. □

A complexity measure on graphs follows from this definition :

$$c(G) = Min\{Card(K)/G = val(t), t \in T(F_{VR}(K))\}$$

This complexity measure is investigated in Courcelle and Olariu [26]. In particular it is compared with tree-width (defined below in Subsection 5.3)

An equational subset of $\mathcal{GP}$ will be called a VR *set of graphs*. Here VR stands for "Vertex-Replacement" and refers to the generation of the same sets by certain context-free graph grammars. These grammars are considered in another chapter of this book ([38]). (The equivalence between equational systems on $\mathcal{GP}$ and these grammars was first proved in Courcelle et al. [24]).

A logical characterization of the VR sets of graphs

We obtain immediately from Corollary 5.4 that :

a set of graphs with ports L is VR iff it is the set of values of the VR-expressions of a recognizable set $K \subseteq T(F_{VR}(C))$ for some finite set $C \subseteq \mathcal{C}$.

We have already recalled the theorem of Doner, Thatcher and Wright stating that a subset of $T(F)$ (where F is finite) is recognizable iff it is MS-definable (see Thomas [62], thm 11.1). We shall now prove that the mapping $val : T(F_{VR}(C)) \to \mathcal{GP}$ is a definable transduction (see Section 4). More precisely :

Proposition 5.6 *: Let C be a finite subset of $\mathcal{C}$. The mapping $val : T(F_{VR}(C)) \to \mathcal{GP}(C)$ is (*, 1)-definable.*

Proof : A term $t \in T(F_{VR}(C))$ is considered as a finite directed tree the nodes of which are labelled in $F = F_{VR}(C)$ and represented by the structure $\mid t \mid = < N_t, suc_1, suc_2, (lab_f)_{f \in F} >$ where

N_t is the set of nodes,

$suc_1(x, y) :\Longleftrightarrow y$ is the first (leftmost) successor of x,

$suc_2(x, y) :\Longleftrightarrow y$ is the second successor of x,

$lab_f(x) :\Longleftrightarrow f$ is the label of x.

The nodes have at most two successors since the symbols in F have arity at most 2 ($\oplus$ has arity two and all others have arity 1 or 0). A graph G in $\mathcal{GP}(C)$ is represented by the structure $\mid G \mid_1 = < V_G, edg_G, (pt_{cG})_{c \in C} >$ where V_G is the set of vertices, edg_G is the set of edges ($edg_G \subseteq V_G \times V_G$), pt_{cG} is the set of c-ports for $c \in C$. We let $\mathcal{R}_s(C) = \{edg\} \cup \{pt_c / c \in C\}$.

Let us now consider $t \in T(F_{VR}(C))$ and $G = val(t) \in \mathcal{GP}(C)$. We shall define V_G as a subset of N_t, edg_G as a binary relation on N_t (such that $edg_G \subseteq V_G \times V_G$), the sets pt_{cG} as subsets of V_G (hence of N_t), all this by MS-formulas. We take V_G equal to the set of leaves of t (i.e., of nodes without successors). If t has no symbol mdf_P or $ren_{p \to q, \ldots}$ then we can define :

$edg_G = \{(x, y) \in N_t / x, y$ are leaves, x has label p, y has label q, there is a node z in N_t labelled by $add_{p,q}$, a path from z to x on which there is no symbol fg_C with $p \in C$ and a

path from z to y on which there is no symbol fg_C with $q \in C$.}

Let us take the following example :

$$t = add_{p,r}(add_{p,q}[p \oplus q \oplus fg_{\{p\}}(add_{q,p}(p \oplus q))] \oplus add_{r,q}[q \oplus r]).$$

There are 6 leaves, $x_1, x_2, \cdots, x_6$ corresponding respectively to the constants p, q, p, q, q, r in t. Because of the "$add_{p,q}$" operation there is an edge from x_1 to x_2 but there is no edge from x_3 to x_2 because of the "$fg_{\{p\}}$" operation which "kills" the label p of x_3.

The general construction must handle the fact that the operations mdf_P (and $ren_{p\to q,\ldots}$) can modify certain port labels. Hence if x is a leaf of t with label p, if y is an ancestor of x then x is a vertex of the graph $val(t/y)$ (where we denote by t/y the subterm of t corresponding to the subtree issued from node y) but it may not be a p-port : it may be a q-port with $q \neq p$ or be not a port. In the last example w is not a p-port of $val(t)$. The construction can be completed with the help of the following lemma. We shall omit all further details. □

Lemma 5.7 *: We fix C as in Proposition 5.6. For each $p \in C$,one can construct a formula $\varphi_p(x, y)$ such that, for every $t \in T(F_{VR}(C))$ and $x, y \in N_t$:*

$$| \, t \, | \quad \models \varphi_p(x, y)$$

iff x is a leaf, y is an ancestor of x in the tree t and x is a p-port of $val(t/y)$.

Proof : Without loss of generality we let $C = \{1, \cdots, n\}$ and $p = 1$. Let us take $t \in T(F_{VR}(C))$; let x be a leaf of t and y an ancestor of x. There exists a unique n tuple of sets $X_1, \cdots, X_n$ satisfying the following conditions :

(1) $X_1 \cup \cdots \cup X_n$ is the set of nodes of t of the path from y to x,

(2) for every $i = 1, \cdots, n : x \in X_i$ iff x has label i (in t),

(3) for every z with $y \leq_T z <_T x$ (see Section 2 for notation) for every $i = 1, \cdots, n$, letting z' be the successor of z such that $z' \leq_T x$ we have : $z \in X_i$ iff

(3.1) either z has label $\oplus$ or $add_{j,j'}$ for some j, j' and $z' \in X_i$
(3.2) or z has label mdf_P for some $P \subseteq C \times C$ and $(j, i) \in P$ for some j and $z' \in X_j$.

(Since fg_C and $ren_{\ldots}$ are special cases of mdf_P this last case also applies to them). It is clear that for every z with $y \leq_T z \leq_T x, x$ is an i-port in $val(t/z)$ iff $z \in X_i$.

Conditions (1) (2) (3) above can be expressed by an MS formula $\psi(x, y, X_1, \cdots, X_n)$. The desired formula is thus $\varphi_p(x, y)$ defined as

$$\theta(x, y) \wedge \exists X_1, \cdots, X_n[\psi(x, y, X_1, \cdots, X_n) \wedge y \in X_p]$$

where $\theta(x, y)$ expresses that x is a leaf and y is an ancestor of x. □

We denote by $\mathcal{B}$ the set of finite binary directed trees. Formally (and more precisely) $\mathcal{B} = T(\{f, a\})$ where f is a binary function symbol and a is nullary.

Theorem 5.8 ([34], [15]) *: Let C be a finite subset of $\mathcal{C}$. A subset of $\mathcal{GP}(C)$ is VR iff it is the image of $\mathcal{B}$ under a $(*, 1)$-definable transduction.*

Proof : "Only if". From Corollary (5.4), one obtains that if L is a VR set then $L = val(K)$ where K is a recognizable subset of $T(F_{VR}(C))$. By Doner et al.'s theorem, K is MS-definable and val is $(*, 1)$-definable. It follows that L is the image of $T(F_{VR}(C))$ by a $(*, 1)$-definable transduction (namely the restriction of val to K, see Proposition 4.5). The classical encoding of terms by binary trees can be adapted and yields a bijection k of K onto a definable subset K' of $\mathcal{B}$ such that k and k^{-1} are both $(*, *)$-definable (See [23], Lemma 2.4 for details). It follows that L is the image of $\mathcal{B}$ under a $(*, 1)$-definable transduction, namely the restriction of val o k^{-1} to K'.

"If". The proof is quite complicated and we refer the reader to [34] or [15], Corollary 4.9. □

Corollary 5.9 *: The family of VR sets of graphs is closed under (1,1)-definable transductions.*

Proof : Immediate consequence of Theorem 5.8 and the fact that the composition of two (1,1)-definable transduction is (1,1)-definable. □

It follows for instance that the set of connected components of the graphs of a VR set is a VR set.

5.3 Hypergraphs with sources and HR sets of hypergraphs

This section is fully analogous to the preceding one. We present operations on hypergraphs upon which the (context-free) HR ("Hyperedge Replacement") hypergraph grammars can be defined as systems of equations.

Hypergraphs have been defined in Section 2.3. We assume that the ranked set A of hyperedge labels is fixed and we shall not specify it in notation. We shall use operations on hypergraphs needing to distinguish certain vertices by means of labels. We let $\mathcal{C}$ be a fixed countable set of *source labels*, not to be confused with the set of hyperedge labels A.

A *hypergraph with sources* is a pair $H =< G, s >$ consisting of a hypergraph G and a total mapping $s : C \to V_G$ called its *source mapping*, where C is a finite subset of $\mathcal{C}$. We say that $s(C) \subseteq V_G$ 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 $\tau(H)$. The source mapping of H is also denoted by src_H. This mapping is not necessarily injective.

We shall denote by $\mathcal{HS}(C)$ the class of all hypergraphs of type C and by $\mathcal{HS}_d(C)$ the subclass of those having *distinct sources*, i.e., that have an injective source mapping.

We shall use the set $\mathcal{S}$ of finite subsets of $\mathcal{C}$ as a set of sorts. We now define some operations on hypergraphs with sources. These operations will form an $\mathcal{S}$-signature. Note that $\mathcal{S}$ is infinite. The definitions will be given here directly in terms of *abstract hypergraphs* (i.e., of isomorphism classes of hypergraphs).

(1) Parallel composition

If $G \in \mathcal{HS}$ and $G' \in \mathcal{HS}(C')$ we let $H = G//_{C,C'}G'$ be the hypergraph in $\mathcal{HS}(C \cup C')$ obtained as follows : one first constructs a hypergraph H' by taking the union of two disjoint hypergraphs K and K' respectively isomorphic to G and G', and by fusing any two vertices v and v' that are respectively the c-source of K and the c-source of K' for some $c \in C \cap C'$; the hypergraph H is defined as the isomorphism class of H'. Clearly, H does not depend on the choices of K and K'. We shall use the simplified (overloaded) notation $G//G'$ when C and C' are known from the context or are irrelevant. We say that $G//G'$ is the *parallel composition* of G and G'. In the case where G and G' have distinct sources, we have the following characterization :

a hypergraph H of type $\tau(G) \cup \tau(G')$ is isomorphic to $G//G'$ if and only if H has subhypergraphs K and K' respectively isomorphic to G and G' such that : $V_K \cup V_{K'} = V_H, E_K \cup E_{K'} = E_H, E_K \cap E_{K'} = \emptyset$ and $V_K \cap V_{K'}$ is the set of sources of H that have label c where $c \in C \cap C'$.

(2) Source renaming

For every mapping $h : C \to C'$, we let $ren_h : \mathcal{HS}(C') \to \mathcal{HS}(C)$ be the mapping such that $ren_h(< G, s >) =< G, s \circ h >$. In other words, the c-source of $ren_h(G)$ is defined as the $h(c)$-source of G. If a vertex v of G is a c-source with $c \in C' - h(C)$ and is not a c'-source for any c' in $h(C)$, then it is an internal vertex in $ren_h(G)$. We shall say that $ren_h(G)$ is the *source renaming* of G defined by h. Note that when we write ren_h, we assume that h is given together with the sets C and C', so that the type of ren_h (namely $C' \to C$) is defined in a unique way. Nevertheless, we shall use the overloaded notation $ren_\emptyset$ for ren_h when h is the *empty mapping* : $\emptyset \to C$ and C need not be specified (or is known from the context).

(3) Source fusion

For every set B, we denote by $Eq(B)$ the set of equivalence relations on B. Let C be a subset of $\mathcal{C}$. For every $\delta \in Eq(C)$ we let $fuse_\delta$ be the mapping $\mathcal{HS}(C) \to \mathcal{HS}(C)$ such that:

$H = fuse_\delta(G)$ if and only if

$V_H = V_G/\sim$, where $\sim$ is the equivalence relation on V_G generated by the set of pairs $\{(src_G(i), src_G(j)) \mid (i,j) \in \delta\}$,

$E_H = E_G$,

$vert_H(e,i) = [vert_G(e,i)]$, where $[v]$ denotes the equivalence class of v with respect to $\sim$,

$lab_H = lab_G$,

$[v]$ is a c-source of H whenever some $v' \sim v$ is a c-source of G.

Intuitively, H is obtained from G by fusing its c- and c'-sources for all c, c' for which c' is δ-equivalent to c. We shall say that H is obtained from G by a *source fusion.*

For every $p \in C$, we denote by p the graph with a single vertex which is the p-source. Hence $p \in \mathcal{HS}(\{p\})$. For every $a \in A$, for every $p_1, \cdots, p_n \in C$ where $n = \tau(a)$, we denote by $a(p_1, \cdots, p_n)$ the hypergraph consisting of a single hyperedge with label a and a sequence of vertices $(x_1, \cdots, x_n)$ such that x_i is the p_i-source for every $i = 1, \cdots, n$. Note that we have $x_i = x_j$ iff $p_i = p_j$. Clearly $a(p_1, \cdots, p_n) \in \mathcal{HS}(\{p_1, \cdots, p_n\})$.

We let F_{HR} be the $\mathcal{S}$-signature consisting of $//_{C,C'}, fuse_\delta, ren_h, p, a(p_1, \cdots, p_n)$ for all relevant $C, C', \delta, h, p, a, p_1, \cdots, p_n$. For $K \subseteq \mathcal{C}$, we denote by $F_{HR}(K)$ the subsignature consisting of the above symbols with $C, C' \subseteq K, p, p_1, \cdots, p_n \in K$ etc... We obtain thus an F_{HR}-magma $\mathcal{HS}$.The terms in $T(F_{HR})$ are called the *HR(hypergraph)-expressions.* Each of them, say t, denotes a hypergraph $val(t)$ in $\mathcal{HS}$ called its value. (Hypergraph expressions based on different operations were first introduced in [3]).

Proposition 5.10 *: Every finite hypergraph in $\mathcal{HS}(C)$ is the value of some HR-expression in $T(F_{HR})_C$.*

Proof : Quite similar to that of Proposition 5.4. One takes a source label $p(v)$ for each $v \in V_G$. One defines G by an HR-expression of the form

$$ren_h(\cdots //a(p_1, \cdots, p_n)// \cdots //p(v_1)// \cdots //p(v_m))$$

where $V_G = \{v_1, \cdots, v_m\}$ and the expression contains one factor $a(p_1, \cdots, p_n)$ for each hyperedge. We omit the details.□

The minimum cardinality of $C \subseteq \mathcal{C}$ such that $G = val(t)$ for some $t \in T(F_{HR}(C))$ is connected with an important complexity measure on graphs and hypergraphs called *treewidth.* We refer the reader to [55] and to [20], Theorem 1.1 for more details. We only indicate that the minimum cardinality of C such that $G = val(t), t \in T(F_{HR}(C))$ is $Max\{Card(\tau(G)), twd(G) + 1\}$ where $twd(G)$ denotes the tree-width of G.

A subset of $\mathcal{HS}(C)$ is called an HR *set of hypergraphs* iff it is $\mathcal{HS}$-equational. The terminology HR refers to an equivalent characterization in terms of "Hyperedge Replacement grammars" studied in another chapter of this book ([29]).

We give here the example of series-parallel graphs. We shall use the source labels 1, 2, 3, and one edge label a of type 2. We define from the operations of $F_{HR}(\{1,2,3\})$ the following operations :

$G//G' = G//_{C,C}G'$ where $C = \{1,2\}$ and $G, G' \in \mathcal{HS}(C)$
$G.G' = ren_h(G//_{\{1,2\},\{2,3\}}ren_{h'}(G'))$

where $h : \{1,2\} \to \{1,3\}$ maps $1 \mapsto 1$ and $2 \mapsto 3$, and $h' : \{2,3\} \to \{1,2\}$ maps $3 \mapsto 2$ and $2 \mapsto 1$. The equation

$$u = u//u + u.u' + a(1,2)$$

defines the class SP of series-parallel graphs. (These graphs are directed and all their edges are labelled by a).

A hypergraph $H \in \mathcal{HS}(C)$ will be represented by the relational structure $\mid H \mid_2 = < V_H \cup E_H, (inc_{aH})_{a\in A}, (ps_{cH})_{c\in C} >$ where inc_{aH} has been defined in Section 2.3 and : $ps_{cH}(x)$ is true iff x is the c-source. Hence $\mid H \mid_2 \ \in STR(\mathcal{R}_m(A,C))$ where $\mathcal{R}_m(A,C) = \mathcal{R}_m(A) \cup \{ps_c / c \in C\}$ and each ps_c is unary. Clearly H and H' are isomorphic iff $\mid H \mid_2$ and $\mid H' \mid_2$ are isomorphic.

Proposition 5.11 *: Let K be a finite subset of $\mathcal{C}$ and $C \subseteq K$. The mapping val : $T(F_{HR}(K))_C \to \mathcal{HS}(C)$ is a $(*,2)$-definable transduction.*

Proof : We refer the reader to Courcelle [13], Lemma 4.3, p. 177. □

Theorem 5.12 *: Let C be a finite subset of $\mathcal{C}$. A subset of $\mathcal{HS}(C)$ is HR iff it is the image of the set of finite binary trees $\mathcal{B}$ under a $(*,2)$-definable transduction.*

Proof : The "only if" direction follows from the preceding proposition like that of Theorem 5.8 from Proposition 5.6. The "if" direction is quite difficult to prove and we refer the reader to Courcelle and Engelfriet [23]. □

Corollary 5.13 *: The family of HR sets of hypergraphs with sources is closed under $(2,2)$-definable transductions.*

Proof : Similar to that of Corollary 5.9. □

Here are some examples of $(2,2)$-definable transductions that are meaningful in graph theory : the transduction from a graph to its spanning trees, or of a hypergraph to its

connected components, or of a graph to its maximal planar subgraphs. (See Proposition 4.2 for other examples).

Corollary 5.14 *: The set of line graphs of the graphs of an HR set is a VR set of graphs.*

Proof : Immediate consequence of Theorems 5.8 and 5.12 and the last assertion of Proposition 4.2. □

We collect below some properties, the proofs of which are too long to be included in this chapter. The reader will note that the statements do not concern logic but only graph properties. However monadic second-order logic is an essential tool for the proofs. We denote by $und(G)$ the undirected graph obtained from G by forgetting the orientations of edges.

Theorem 5.15 *Every HR set of simple directed graphs is VR. Conversely, for a VR set L of directed graphs, the following conditions are equivalent :*

1. *L is HR*
2. *there exists an integer k such that $Card(E_G) \leq k.Card(V_G)$ for every $G \in L$,*
3. *there exists an integer n, such that the complete bipartite graph $K_{n,n}$ is not a subgraph of $und(G)$ for any $G \in L$.*

Proof hints : The first assertion is an easy consequence of Theorems 5.8 and 5.12 because the transduction $\{(\mid G \mid_2, \mid G \mid_1) \ / \ G$ is a graph$\}$ is definable (one can also say that the identity on simple directed graphs is $(2,1)$-definable ; see Fact 4.3.1).

Implications (1) $\Rightarrow$ (2) and (1) $\Rightarrow$ (3) follow easily from general properties of HR sets of graphs (see chapter [29] in this book). The implications (2) $\Rightarrow$ (1) and (3) $\Rightarrow$ (1) are difficult : see Courcelle [19]. □

One can extend Theorem 5.15 as follows. A set $L \subseteq \mathcal{H}(A)$ of simple hypergraphs (without sources) is said to be VR iff it is the image of $\mathcal{B}$ under a $(*,1)$-definable transduction. Hence we use here the characterization of Theorem 5.8 of VR sets of graphs and we make it into a definition of *VR sets of hypergraphs.* (Equivalent characterizations in terms of systems of equations can be found in Courcelle [15], Thm 4.6 but no equivalent context-free grammar, based on appropriate rewriting rules has yet be defined.)

For every hypergraph H we denote by $K(H)$ the undirected graph obtained by substituting a complete undirected graph K_m for every hyperedge of type m.

Theorem 5.16 *[19] : Every HR set of simple hypergraphs is VR. Conversely, for every VR set L of directed graphs, the following conditions are equivalent :*

1. *L is HR,*

2. *there exists an integer* k *such that* $Card(E_H) \leq k.Card(V_H)$ *for every* $H \in L$,

3. *there exists an integer* n *such that* $K_{n,n}$ *is not a subgraph of* $K(H)$ *for any* $H \in L$.

Again, the only difficult implications are (2) $\Rightarrow$ (1) and (3) $\Rightarrow$ (1).

6 Inductive computations and recognizability

The notion of a recognizable set is due to Mezei and Wright [50] ; it extends the notion of a regular language like the notion of an equational set extends that of a context-free one. It was originally defined for one-sort structures, and we adapt it to many-sorted ones, possibly with infinitely many sorts. We begin with a study of recognizability in a general algebraic framework. Then we state the basic result that MS-definable sets of graphs are recognizable. Finally we state a generalized version of Parikh's theorem based on inductive computations, (the fundamental notion behind recognizability) and give some applications to VR and HR grammars.

6.1 Inductive sets of predicates and recognizable sets

Let F be an $\mathcal{S}$-signature. An F-magma A is *locally finite* if each domain A_s is finite. Let M be an F-magma and $s \in \mathcal{S}$. A subset B of M_s is *M-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 $B = h^{-1}(C)$. We denote by $Rec(M)_s$ the family of M-recognizable subsets of M_s.

If F is a finite signature, the recognizable subsets of terms over F, i.e., the $T(F)$-recognizable sets can be characterized by finite-state *tree*-automata (see Gecseg and Steinby [42]). The classical identification of terms with finite ordered ranked trees explains the qualification of "tree"-automaton.

By a *predicate* on a set E, we mean a unary relation, i.e., a mapping $E \to \{true, false\}$. If M is a many-sorted F-magma with set of sorts $\mathcal{S}$, a *family of predicates* on M is an indexed set $\{\hat{p}/p \in P\}$, given with a mapping $\sigma : P \to S$ such that each $\hat{p}$ is a predicate on $M_{\sigma(p)}$. We call $\sigma(p)$ the *sort* of p. Such a family will also be denoted by P. For $p \in P$, we let $L_p = \{d \in M_{\sigma(p)} \ / \ \hat{p}(d) = true\}$. The family P is *locally finite* if, for each $s \in \mathcal{S}$ the set $\{p \in P \ / \ \sigma(p) = s\}$ is finite. We say that P is *f-inductive* where f is an operation in F, if for every $p \in P$ of sort $s = \sigma(f)$ there exist $m_1, \cdots, m_n$ in $\mathcal{N}$, (where n is the rank of f), an $(m_1 + \cdots + m_n)$-place Boolean expression B, and a sequence of $(m_1 + \cdots + m_n)$ elements of $P, (p_{1,1}, \cdots, p_{1,m_1}, p_{2,1}, \cdots, p_{2,m_2}, \cdots, p_{n,m_n})$, such that, if the type of f is $s_1 \times s_2 \times \cdots \times s_n \to s$ we have :

1. $\sigma(p_{i,j}) = s_i$ for all $j = 1, \cdots, m_i$ and $i = 1, \cdots, n$,

2. for all $d_1 \in M_{s_1}, \cdots, d_n \in M_{s_n}$

$$\hat{p}(f_M(d_1, \cdots, d_n)) = B[\hat{p}_{1,1}(d_1), \cdots, \hat{p}_{1,m_1}(d_1), \hat{p}_{2,m_2}(d_2), \cdots, \hat{p}_{n,m_n}(d_n)].$$

The sequence $(B, p_{1,1}, \cdots, p_{2,1}, \cdots, p_{n,m_n})$ is called a *decomposition of p relative to f*. In words, the existence of such a decomposition means that the validity of p for any object of the form $f_M(d_1, \cdots, d_n)$ can be computed from the truth values of finitely many predicates of P for the objects $d_1, \cdots, d_n$. This computation can be done by a Boolean expression that depends only on p and f. We say that P is *F-inductive* if it is f-inductive for every f in F.

Proposition 6.1 *: Let M be an F-magma. For every $s \in \mathcal{S}$ a subset L of M_s is recognizable if and only if $L = L_{p_0}$ for some predicate p_0 belonging to a locally finite F-inductive family of predicates on M.*

Proof : "Only if". Let $L = h^{-1}(C) \subseteq M_s$, for some homomorphism $h : M \to A$ where A is locally finite and $C \subseteq A_s$. We let $P = \cup\{A_t / t \in \mathcal{S}\} \cup \{p_0\}$. Each element a of A_t is of sort t (considered as a member of P). The domains of A are pairwise disjoint and p_0 is of sort s. For $d \in M_t$, and $a \in A_t$, we let :

$\hat{a}(d) \quad = true \quad$ if $h(d) = a$,
$\quad\quad\;\; = false \quad$ otherwise,

For $d \in M_s$, we let

$\hat{p}_0(d) \quad = true \quad$ if $h(d) \in C$,
$\quad\quad\;\; = false \quad$ otherwise.

It is clear that P is locally finite. It is F-inductive, and, clearly, $L = L_{p_0}$.

"If". Let P be a locally finite F-inductive family of predicates. Let $L = L_{p_0}$ for some $p_0 \in P$. For every $t \in \mathcal{S}$ we let P_t be the set of predicates in P of sort t. We let A_t be the set of mappings : $P_t \to \{true, false\}$. We let h be the mapping : $\cup\{M_t \;/\; t \in S\} \to \cup\{A_t \;/\; t \in S\}$ such that, for every $t \in S$ and $m \in M_t, h(m)$ is the mapping : $P_t \to \{true, false\}$ such that $h(m)(p) = \hat{p}(m)$. We want to find operations on the sets A_t such that $A = \; < (A_t)_{t \in \mathcal{S}}, (f_A)_{f \in F} >$ is an F-magma and $h : M \to A$ defined above is a homomorphism. We need to define f_A where f is of type $s_1 \times \cdots \times s_n \to t$ in such a way that for all $(m_1, \cdots, m_m) \in M_{s_1} \times \cdots \times M_{s_n}$ we have :

$$h(f_M(m_1, \cdots, m_n)) = f_A(h(m_1), \cdots, h(m_n)).$$

But it is possible to find such f_A by using the decompositions of the predicates in P_t relative to f. Hence, $L = L_{p_0} = h^{-1}(C)$ where $C = \{\theta \in A_t \;/\; \theta(p_0) = true\}$. It follows that L is M-recognizable since, by construction, A is locally finite.□

Example : Let L be the set of rooted trees with a number of nodes that is not a multiple of 3. Let p be the corresponding predicate on $\mathcal{R}$. (We use the notation and definitions of Subsection 5.1.) Let us consider the following predicates : for $i = 0, 1, 2$ we let $q_i(t)$ hold iff the number of nodes of t is of the form $3k + i$ for some k. It is easy to check that $P = \{p, q_0, q_1, q_2\}$ is inductive with respect to the operations ext and $//$ on rooted trees ; this check uses in particular the following facts which hold for all rooted trees t and t' :

$q_0(t//t') = (q_0(t) \wedge q_1(t')) \vee (q_1(t) \wedge q_0(t')) \vee (q_2(t) \wedge q_2(t'))$,

$q_1(t//t') = (q_1(t) \wedge q_1(t')) \vee (q_0(t) \wedge q_2(t')) \vee (q_2(t) \wedge q_0(t'))$,

$q_2(t//t') = (q_1(t) \wedge q_2(t')) \vee (q_2(t) \wedge q_1(t')) \vee (q_0(t) \wedge q_0(t'))$,

$q_0(ext(t)) = q_2(t)$,

$q_1(ext(t)) = q_0(t)$,

$q_2(ext(t)) = q_1(t)$.

Since p is equivalent to $q_1 \vee q_2$, we have :

$p(ext(t)) = q_0(t) \vee q_1(t)$

and one can easily write a similar definition of $p(t//t')$. Hence $L = L_p$ is recognizable. Note that $\{q_0, q_1\}$ is inductive because q_2 is definable as $\neg q_0 \wedge \neg q_1$. □

Proposition 6.2 *: Let M be generated by F and t be a sort. A subset L of M_t is M-recognizable iff $h_M^{-1}(L)$ is $T(F)$-recognizable.*

Proof : We prove the "only if" direction. If $L = h^{-1}(C)$ for some homomorphism $h : M \to A$, where A is locally finite, then $h_M^{-1}(L) = (hoh_M)^{-1}(C)$, and, since hoh_M is a homomorphism : $T(F) \to A$, the set $h_M^{-1}(L)$ is $T(F)$-recognizable. For the "if" direction, see Courcelle [22], Proposition 4.4. □

This proposition means that, in order to decide whether an element m of M belongs to L, it suffices to take *any* term t in $T(F)$ denoting m and to decide whether it belongs to the recognizable set $h_M^{-1}(L)$ (for instance by running an automaton on this term). The key point is that the answer is the same for any term t denoting m. This should be contrasted with the characterization of equational sets of Corollary 5.4, which says that, if L is equational, then it is of the form $h_M(K)$ for some recognizable set of terms K : in this case, in order to establish that m belongs to L, one must find a specific term t denoting m, and the decision cannot be made from an arbitrary term in $h_M^{-1}(m)$.

Relationships between equational sets and recognizable sets.

We recall that we denote by $Equat(M)_s$ the family of M-equational subsets of M_s.

Theorem 6.3 *: If $K \in Rec(M)_s$ and $L \in Equat(M)_s$ then $L \cap K \in Equat(M)_s$.*

Proof : We can assume that $L = L((S, M), u_0)$ for some $u_0 \in U$ where S is a uniform polynomial system over F with set of unknowns U. A polynomial system is *uniform* if its equations are of the form $u = t_1 + t_2 + \cdots + t_m$, where each t_i is of the form $f(u_1, u_2, \cdots, u_k)$ for some $f \in F$, some unknowns $u_1, \cdots, u_k \in U$; the transformation of an arbitrary system

into an equivalent one which is uniform is the essence of the transformation of an arbitrary context-free grammar into one in Chomsky normal form. An example will illustrate this step later. See Courcelle [22], Proposition 5.19 for the proof.

Let $F' \subseteq F$ be the finite set of symbols occurring in S, and let $\mathcal{S}' \subseteq \mathcal{S}$ be the finite set of sorts of these symbols together with those of the unknowns of S. Hence F' is an $\mathcal{S}'$-signature. Let $h : M \to A$ be an F'-homomorphism (with A locally finite), such that $K = h^{-1}(C)$ for some $C \subseteq A_s$.

For every $u \in U$, we let $L_u := L((S, M), u)$. Let W be the new set of unknowns $\{[u, a]/u \in U, a \in A_{\sigma(u)}\}$. It is finite. We shall define a system S', with set of unknowns W, such that :

$$L((S', M), [u, a]) = L_u \cap h^{-1}(a)$$

for every $[u, a] \in W$. Let $u \in U$ and $a \in A_{\sigma(u)}$. Let us assume that the defining equation of u in S is of the form $u = t_1 + \cdots + t_k$. Consider one of the monomials, say t_i. Let us assume that it is of the form $f(u_1, \cdots, u_n)$ for some unknowns $u_1, \cdots, u_n$. For every $a \in A_{\sigma(u)}, a_1 \in A_{\sigma(u_1)}, \cdots, a_n \in A_{\sigma(u_n)}$ such that $f_A(a_1, \cdots, a_n) = a$, we form the monomial $f([u_1, a_1], \cdots, [u_n, a_n])$, and we let $\hat{t}_i$ denote the sum of these monomials. If no such n-tuple $(a_1, \cdots, a_n)$ exists, then $\hat{t}_i$ is defined as Ω. The defining equation of $[u, a]$ in S' is taken as :

$$[u, a] = \hat{t}_1 + \hat{t}_2 + \cdots + \hat{t}_k.$$

It is clear from this construction that the W-indexed family of sets $(L_u \cap h^{-1}(a))_{[u,a]\in W}$ is a solution of S' in $\mathcal{P}(M)$. Hence $L_u \cap h^{-1}(a) \supseteq L_{u,a}$, where $(L_{u,a})_{[u,a]\in W}$ denotes the least solution of S' in $\mathcal{P}(M)$.

In order to establish the opposite inclusion, we define from $L_{u,a}$ the sets $L'_u = \cup\{L_{u,a}/a \in A_{\sigma(u)}\}$ for $u \in U$. Clearly $(L'_u)_{u\in U}$ is a solution of S in $\mathcal{P}(M)$. Hence $L_u \subseteq L'_u$ for all u. For all $a \in A_{\sigma(u)}$, we have :

$$L_u \cap h^{-1}(a) \subseteq L'_u \cap h^{-1}(a) = (\cup\{L_{u,b}/b \in A\}) \cap h^{-1}(a).$$

The latter set is equal to $L_{u,a} \cap h^{-1}(a)$ since $L_{u,a} \subseteq L_u \cap h^{-1}(a)$ and, $h^{-1}(b) \cap h^{-1}(b') = \emptyset$ for all b, b' with $b \neq b'$. Hence $L_u \cap h^{-1}(a) \subseteq L_{u,a}$. By the first part of the proof, we have an equality, and $(L_u \cap h^{-1}(a))_{[u,a]\in W}$ is the least solution of S' in $\mathcal{P}(M)$. We have :

$$\begin{aligned} L \cap K &= L_{u_0} \cap h^{-1}(C) \\ &= \cup\{L((S', M), [u_0, a])/a \in C\}. \end{aligned}$$

Finally, we get

$$L \cap K = L((S'', M), w)$$

where S'' is S' augmented with the equation

$$w = [u_0, a_1] + [u_0, a_2] + \cdots + [u_0, a_r],$$

w is a new unknown and $C = \{a_1, a_2, \cdots, a_r\}$. Hence $L \cap K$ is M-equational.□

This theorem extends the classical result saying that the intersection of a context-free language and a regular one is context-free. The construction is effective if K is effectively given (i.e., if h is computable, if the finite sets A_s are computable, etc...) and L is defined by a given system. Hence, since the emptiness of an equational set (defined by a system of equations) is decidable, we have the following corollary.

Corollary 6.4 *: If K is an effectively given M-recognizable set, and if L is an M-equational set defined by a given system, one can test whether $L \cap K = \emptyset$.*

Example : We consider the set $\mathcal{R}$ of rooted trees with the operations $//, ext$ and 1 defined in Subsection 5.1. We consider the following two predicates on $\mathcal{R}$:

$p(t) :\Longleftrightarrow$ all branches of t are of even length,

$q(t) :\Longleftrightarrow$ all branches of t are of odd length.

The following clauses give a precise definition of p and q and show that $\{p, q\}$ is $\{//, ext\}$-inductive. For all $t, t' \in \mathcal{R}$ we have :

$p(t//t') = p(t) \wedge p(t')$,

$q(t//t') = q(t) \wedge q(t')$,

$p(ext(t)) = q(t)$,

$q(ext(t)) = p(t)$,

$p(1) = false$,

$q(1) = true$.

Let us perform on this example the construction of the proof of the "if" part of Proposition 6.1. We take $A = \{true, false\}^2$ where in $(x, y) \in A$ the component x is the truth value of p and y is that of q. Actually, since p and q cannot be both $true$, we can restrict ourselves to $A = \{(t, f), (f, t), (f, f)\}$ where t stands for $true$ and f for $false$. We let $1_A = (f, t)$. The mappings $//_A$ and ext_A are defined by the following tables :

$//_A$	(t,f)	(f,t)	(f,f)
(t,f)	(t,f)	(f,f)	(f,f)
(f,t)	(f,f)	(f,t)	(f,f)
(f,f)	(f,f)	(f,f)	(f,f)

ext_A	
(t,f)	(f,t)
(f,t)	(t,f)
(f,f)	(f,f)

Take for example the following system S which is not uniform :

$$\begin{cases} u = & 1 + ext(ext(v)) \\ v = & u + ext(v)//ext(v). \end{cases}$$

We transform it into the following uniform system S' :

$$\begin{cases} u = & 1 + ext(w) \\ v = & 1 + ext(w) + w//w \\ w = & ext(v) \end{cases}$$

and clearly $L((S', \mathcal{R}), u) = L((S, \mathcal{R}), u)$. In order to construct $L' = L((S', \mathcal{R}), u) \cap L_p$ we introduce the unknowns $[x, y, z]$ with $x \in \{u, v, w\}, y, z \in \{t, f\}, y$ and z not both *true*. We obtain the following equations forming a new system S" :

$$\begin{array}{lcl} [u,t,f] & = & ext([w,f,t]) \\ [u,f,t] & = & 1 + ext([w,t,f]) \\ [u,f,f] & = & ext([w,f,f]) \\ [v,t,f] & = & ext([w,f,t]) + [w,t,f]//[w,t,f] \\ [v,f,t] & = & 1 + ext([w,t,f]) + [w,f,t]//[w,f,t] \\ [v,f,f] & = & ... \\ [w,t,f] & = & ext([v,f,t]) \\ [w,f,t] & = & ext([v,t,f]) \\ [w,f,f] & = & ext([w,f,f]) \end{array}$$

We are interested in $L' = L((S", \mathcal{R}), [u, t, f])$ and it is not hard to see that the unknowns of the form $[x, f, f]$, $x = u, v, w$ are useless for the generation of the elements of L'. So is $[u, f, t]$. We can thus delete the corresponding equations. Letting x_1 replace $[x, t, f]$ and x_2 replace $[x, f, t]$ for $x \in \{u, v, w\}$ we obtain the more readable system T :

$$\begin{cases} u_1 & = & ext(w_2) \\ v_1 & = & ext(w_2) + w_1//w_1 \\ v_2 & = & 1 + ext(w_1) + w_2//w_2 \\ w_1 & = & ext(v_2) \\ w_2 & = & ext(v_1) \end{cases}$$

and $L' = L((T, \mathcal{R}), u_1)$. This system can be reduced into the following one, call it T'.

$$\begin{cases} u_1 & = & ext(ext(v_1)) \\ v_1 & = & ext(ext(v_1)) + ext(v_2)//ext(v_2) \\ v_2 & = & 1 + ext(ext(v_2)) + ext(v_1)//ext(v_1) \end{cases}$$

Finally, we have $L((T', \mathcal{R}), u_1) = L((S, \mathcal{R}), u) \cap L_p$. □

6.2 Inductivity of monadic second-order predicates

In this section, we present the fundamental result saying that, roughly speaking, every monadic second-order definable set of graphs or hypergraphs is recognizable. However, in order to get a sensible statement, we must specify two things :

1. the representation of graphs and hypergraphs we are using (because we have defined two of them, yielding two different notions of monadic second-order definability),

2. the relevant operations on graphs and hypergraphs (we have defined two signatures of operations denoted by F_{VR} and F_{HR}, defining respectively the magma $\mathcal{GP}$ of simple graphs with ports and the magma $\mathcal{HS}$ of hypergraphs with sources ; see Subsections 5.2 and 5.3.

We recall that if C is a set of port labels, we denote by $F_{VR}(C)$ the subsignature of F_{VR} consisting of the operations involving only port labels from C. We denote by $\mathcal{GP}(C)$ the $F_{VR}(C)$-magma with domain $\mathcal{GP}(C)$ and operations of $F_{VR}(C)$. Hypergraphs will be over a fixed finite ranked set A of hyperedge labels that we will not specify in notation. We recall that CMS_i refers to counting monadic second-order logic where a graph or a hypergraph G is represented by the structure $\mid G \mid_i$. (See Sections 2 and 3). We only consider finite graphs and hypergraphs, but we keep the notations $\mathcal{GP}(C)$ and $\mathcal{HS}(C)$.

Theorem 6.5 *: (1) Let C be a finite set of port labels. Every CMS_1-definable subset of $\mathcal{GP}(C)$ is $\mathcal{GP}(C)$-recognizable.*
(2) Let C be a finite set of source labels. Every CMS_2-definable subset of $\mathcal{HS}(C)$ is $\mathcal{HS}$-recognizable.

We shall sketch the proof of the first assertion. For the second assertion, we shall refer the reader to the proofs given in [11] and [15].

We need some notation. For every $n \in \mathcal{N}$, $q \in \mathcal{N}$-$\{0,1\}$, for every finite subset C of $\mathcal{C}$, for every finite set $\mathcal{W}$ of set variables, $L(n,q,C,\mathcal{W})$ denotes the subset of $CMS(\mathcal{R}_s(C),\mathcal{W})$ consisting of formulas of quantification depth at most n, possibly written with the atomic formulas $Card_p(X)$ for $2 \leq p \leq q$ (see Subsection 3.1).

We shall say that two formulas are *tautologically equivalent* if they can be transformed into each other by renamings of bound variables and uses of the laws of Boolean calculus. (For instance φ is tautologically equivalent to $\varphi \wedge \varphi$). Every two tautologically equivalent formulas are equivalent and we shall assume that every formula is transformed into a tautologically equivalent formula of some standard form. With this convention, each set $L(n,q,C,\mathcal{W})$ is finite. Each formula $\varphi \in L(n,q,C,\emptyset)$ defines a predicate $\hat{\varphi}$ on $\mathcal{GP}(C)$ such that for every $G \in \mathcal{GP}(C)$:

$$\hat{\varphi}(G) \text{ is true iff } \mid G \mid_1 \models \varphi.$$

We let C be fixed and we denote by $\hat{L}(n,q)$ the corresponding family of predicates. It is finite.

Lemma 6.6 *For every $n,q \in \mathcal{N}$ with $q \geq 2$, the family of predicates $\hat{L}(n,q)$ is $F_{VR}(C)$-inductive.*

Proof sketch : We first consider the inductivity condition for the operation $add_{c,d}$ where

$c, d \in C$. Let us recall that $G' = add_{c,d}(G)$ iff G' is obtained from G by the addition of an edge from x to y where x is any c-port and y is any d-port such that $(x, y) \notin edg_G$ and $x \neq y$. For every formula φ in $CMS(\mathcal{R}_s(C), \emptyset)$ one can construct a formula ψ in the same set and of no larger quantification depth such that, for every $G \in \mathcal{GP}(C)$

$$\hat{\varphi}(add_{c,d}(G)) = \hat{\psi}(G) \tag{3}$$

i.e.,

$$| \, add_{c,d}(G) \, |_1 \models \varphi \quad \textit{iff} \quad | \, G \, |_1 \models \psi \tag{4}$$

In other words the family of predicates on $\mathcal{GP}(C)$ associated with the formulas in $L(n, q, C, \emptyset)$ is $\{add_{c,d}\}$-inductive. The decomposition of $\hat{\varphi}$ relative to $add_{c,d}$ is simply $(B, \hat{\psi})$ (where B is the trivial Boolean combination such that $B[\psi]$ is ψ).

The proof of (4) is actually an immediate consequence of Lemma 1.5 because if $G' = add_{c,d}(G)$ we have the following definition of $edg_{G'}$:

$$edg_{G'}(x, y) \Longleftrightarrow edg_G(x, y) \vee (pt_{c_G}(x) \wedge pt_{d_G}(y) \wedge \neg x = y).$$

Hence, one can take ψ equal to $\varphi[\theta(x_1, x_2)/edg]$ where θ is the quantifier-free formula :

$$edg(x_1, x_2) \vee (pt_c(x_1) \wedge pt_d(x_2) \wedge \neg x_1 = x_2).$$

It is important to note that this transformation does not increase the quantification depth because the formula substituted for edg has no quantifier. The same holds (with similar argument) for the operations mdf_P. The inductivity of $\hat{L}(n, q)$ with respect to the operations mdf_P can be proved similarly.

Next we consider the case of $\oplus$, the disjoint union operation. The statement analogous to (3) is the following one : for every formula φ in $L(n, q, C, \emptyset)$ one can find $m \in \mathcal{N}$ and construct formulas $\psi_1, \psi'_1, \cdots, \psi_m, \psi'_m$ in $L(n, q, C, \emptyset)$ such that, for every G and $G' \in \mathcal{GP}(C)$:

$$\hat{\varphi}(G \oplus G') = \bigvee_{1 \leq i \leq m} \hat{\psi}_i(G) \wedge \hat{\psi'}_i(G'). \tag{5}$$

(The validity of (5) is actually known from Shelah [59], Section 2, given without proof as a easy modification of a previous result by Fefermann and Vaught [40] for first-order formulas). In this case the decomposition of $\hat{\varphi}$ with respect to $\oplus$ is :

$$((x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee \cdots \vee (x_{2m-1} \wedge x_{2m}), \hat{\psi}_1, \hat{\psi'}_1, \hat{\psi}_2, \hat{\psi'}_2, \cdots, \hat{\psi}_m, \hat{\psi'}_m).$$

The proof is more complicated than for (3) ; it can be done by induction on the structure of φ and the cases where $\varphi = \varphi_1 \wedge \varphi_2, \varphi = \varphi_1 \vee \varphi_2, \varphi = \neg\varphi_1$ are rather straightforward. However, in order to handle inductively the cases where $\varphi = \exists X \varphi_1$, one needs a formulation of (5) where φ has free variables. Let $\mathcal{W} = \{X_1, \cdots, X_r\}$. Let $\varphi \in L(n, q, C, \mathcal{W})$. Let

$G \in \mathcal{GP}(C)$ and $V_1, \cdots, V_r$ be sets of vertices of G (i.e., subsets of the domain of the structure $\mid G \mid_1$). We let $\hat{\varphi}(G, V_1, \cdots, V_r) = \mathit{true}$ if

$$(\mid G \mid_1, V_1, \cdots, V_r) \models \varphi(X_1, \cdots, X_r)$$

and $\hat{\varphi}(G, V_1, \cdots, V_r) = \mathit{false}$ otherwise. The generalized version of (5) is now : for every formula $\varphi \in L(n, q, C, \mathcal{W})$, one can find $m \in \mathcal{N}$ and construct $\psi_1, \psi'_1, \cdots, \psi_m, \psi'_m \in L(n, q, C, \mathcal{W})$ such that, for every $G, G' \in \mathcal{GP}(C)$ such that $V_G \cap V'_G = \emptyset$, for every $V_1, \cdots, V_r \subseteq V_G$, for every $V'_1, \cdots, V'_r \subseteq V'_G$:

$$\hat{\varphi}(G \oplus G', V_1 \cup V'_1, \cdots, V_r \cup V'_r) = \bigvee_{1 \leq i \leq m} \hat{\psi}_i(G, V_1, \cdots, V_r) \wedge \hat{\psi'}_i(G', V'_1, \cdots, V'_r) \qquad (6)$$

This can now be proved by induction on the structure of φ.

Proof of Theorem 6.5 (1) Let C be finite, let $L \subseteq \mathcal{GP}(C)$ be defined by a formula φ of CMS relatively to the representation of the graphs G in $\mathcal{GP}(C)$ by structures of the form $\mid G \mid_1$. For some large enough n and q the formula φ belongs to $L(n, q, C, \emptyset)$. The predicate $\hat{\varphi}$ belongs to the finite family $\hat{L}(n, q)$ of predicates which is $F_{VR}(C)$-inductive (Lemma 6.6). Hence L is $\mathcal{GP}(C)$-recognizable.

(2) The proof is similar. We indicate the main ideas. The finite set A is fixed. The F_{HR}-magma of hypergraphs $\mathcal{HS}$ is many-sorted and its set of sorts is the set $\mathcal{S}$ of finite sets of source labels. For every $n, q, C, \mathcal{W}$ as above, we denote by $L'(n, q, C, \mathcal{W})$ the set of formulas in $CMS(\mathcal{R}_m(A, C), \mathcal{W})$ of quantification depth at most n and using the special atomic formulas $Card_p(X)$ for $p \leq q$. This set is finite. For each C, we consider the finite family of predicates $\{\hat{\varphi} \ / \ \varphi \in L'(n, q, C, \emptyset)\}$ where for $H \in \mathcal{HS}(C)$:

$$\hat{\varphi}(H) \text{ is true iff } \mid H \mid_2 \ \models \varphi$$

We denote it by $\hat{L}'(n, q, C)$. Hence we obtain a locally finite family of predicates on $\mathcal{HS}$ where $\hat{\varphi} \in \hat{L}'(n, q, C)$ is of sort C. For every $n, q \in \mathcal{N}$ with $q \geq 2$, the family $\cup\{\hat{L}'(n, q, C) \ / \ C \in \mathcal{S}\}$ is F_{HR}-inductive. It follows as above (see [11] or [15] for details) that every CMS_2-definable subset of $\mathcal{HS}(C)$ is $\mathcal{HS}$-recognizable. □

Corollary 6.7 *: (1) The intersection of a VR set of graphs and a CMS_1-definable subset of $\mathcal{GP}(C)$ (where C is finite) is a VR set. The CMS_1-satisfiability problem for a VR set is decidable.*
(2) The intersection of an HR subset of $\mathcal{HS}(C)$ (where C is finite) and a CMS_2-definable one is HR. The CMS_2-satisfiability problem for an HR set is decidable.

Proof : (1) Let L be a VR set of graphs. Let D be the finite set consisting of all port labels occuring in the operation of some system of equations defining L. Then, L is $\mathcal{GP}(D)$-equational and $L \subseteq \mathcal{GP}(D)$. Let K be a CMS_1-definable subset of $\mathcal{GP}(C)$ for some finite $C \subseteq \mathcal{C}$. Then L is $\mathcal{GP}(C \cup D)$-equational and K is a CMS_1-definable subset of $\mathcal{GP}(C \cup D)$. Hence $L \cap K$ is $\mathcal{GP}(C \cup D)$-equational by Theorem 6.3 hence VR since K is $\mathcal{GP}(C \cup D)$-recognizable. (Theorem 6.5).

(2) Let L be an HR-subset of $\mathcal{HS}(C)$ and K be a CMS_2-definable one. Then K is $\mathcal{HS}$-recognizable by Theorem 6.5 and $L \cap K$ is $\mathcal{HS}$-equational (Theorem 6.3) hence HR. □

Corollary 6.8 : *Let C be a finite set of port or source labels. Let A be a finite ranked alphabet. (1) For every formula $\varphi \in CMS(\mathcal{R}_s(C), \emptyset)$ the property $\mid G \mid_1 \models \varphi$ can be decided in time $O(size(t))$ where $G = val(t)$ and $t \in T(F_{VR}(C))$.*
(2) For every formula $\varphi \in CMS(\mathcal{R}_m(A, C), \emptyset)$ the property $\mid H \mid_2 \models \varphi$ can be decided in time $O(size(t))$ where $H = val(t)$ and $t \in T(F_{HR}(A))_C$.

In these two assertions, the input is a term t denoting G or H. Since not all finite graphs (hypergraphs) are the values of terms in $T(F_{VR}(C))(T(F_{HR}(A))_C)$, this corollary gives efficient algorithms only for CMS properties on (finite) graphs (hypergraphs) in special classes. Furthermore, the graphs or hypergraphs must be "parsed", i.e., given by terms.

Proof : Let F be a finite signature. The membership of a term t in a recognizable subset of $T(F)$ is decidable in time $O(size(t))$ because recognizable sets of terms are defined by finite-state deterministic frontier-to-root automata. The two assertions follow then from Proposition 6.2 and Theorem 6.5. □

6.3 Inductively computable functions and a generalization of Parikh's theorem

Let F be an $\mathcal{S}$-signature and M be an F-magma. Let N be a set and let $\mathcal{E}$ be a family of mappings called *evaluations* where each $e \in \mathcal{E}$ maps $M_s \to N$ for some $s \in S$. We call the sort s the *type* of e. We say that $\mathcal{E}$ is *F-inductive* if for every $e \in \mathcal{E}$ and $f \in F$ there exists a partial function $g_{e,f} : N^m \to N$ and k sequences $(e_1^1, \cdots, e_{n_1}^1), \cdots, e(_1^k \cdots, e_{n_k}^k)$ where $k = \rho(f)$ such that $m = n_1 + \cdots + n_k$, each function e_i^j has type $\alpha(f)_j$ and we have, for every $d_1, \cdots, d_k \in M$ with $\sigma(d_i) = \sigma(f)_i$, for all $i = 1, \cdots, k$:

$$e(f_M(d_1, \cdots, d_k)) = g_{e,f}(e_1^1(d_1), \cdots, e_{n_1}^1(d_1), e_1^2(d_2), \cdots, e_1^k(d_k), \cdots, e_{n_k}^k(d_k)). \qquad (7)$$

This means that the value of e at $f_M(d_1, \cdots, d_k)$ can be computed, by means of some fixed functions $g_{e,f}$, from the values at $d_1, \cdots, d_k$ of m mappings of $\mathcal{E}$ (the mappings e_i^j are not necessarily pairwise distinct and some of them may be equal to e). We shall call the tuple

$$(g_{e,f}, (e_1^1, \cdots, e_{n_1}^1), (e_1^2, \cdots), \cdots, (e_1^k, \cdots, e_{n_k}^k)) \qquad (8)$$

the *decomposition of e relative to f_M*. In Subsection 6.1, we have introduced inductive families of predicates : they are just the special case of inductive families of evaluations where N consists of the truth values *true* and $false$. If $\mathcal{E}$ is F-inductive, if $d \in M$ is given as t_M for some $t \in T(F)$ (i.e., this means that d has been "parsed" in terms of the operations of M), then one can evaluate $e(d)$ by the following algorithm using two traversals of t.

Algorithm :

Input : a term t given as a tree, an evaluation e_0 in $\mathcal{E}$:
Output : the value $e_0(t_M)$.
Method :
First traversal (top-down) : One associates with every node u of the tree t a set of evaluations $\mathcal{E}(u)$, that will have to be computed at u, i.e., for the argument defined by the subtree of t issued of u. For the root r, we let $\mathcal{E}(r) := \{e_0\}$. For every node u such that $\mathcal{E}(u)$ is already known and that has successors $u_1, \cdots, u_k$, for every e in $\mathcal{E}(u)$, if f is the operation labelling u, and

$$(g, (e_1^1, \cdots, e_{n_1}^1), (e_1^2, \cdots), \cdots, (e_1^k, \cdots, e_{n_k}^k)) \tag{9}$$

is the decomposition of e relative to f_M, we add to each set $\mathcal{E}(u_r), i = 1, \cdots, k$, the evaluations $e_1^i, \cdots, e_{n_i}^i$.

Second traversal (bottom-up) : Starting from the leaves, one computes at each node u the values of each function $e \in \mathcal{E}(u)$ by using the decomposition of e relative to f_M (see (5)). One obtains at the root the desired value of e_0. Denoting by $e(u)$ such a value (i.e., we denote by $e(u)$ the value of e for $(t/u)_M$, we use the formula :

$$e(u) := g(e_1^1(u_1), \cdots, e_{n_1}^1(u_1), e_1^2(u_2), \cdots, e_1^k(u_k), \cdots, e_{n_k}^k(u_k)).$$

based of the decomposition of e relative to f_M of the form (9). One obtains at the root the desired value of e_0.

This technique is useful for certain graph algorithms, taking as input graphs or hypergraphs expressed as values of VR or HR expressions. We refer the reader to [11], [12], [2], [25] for the construction of efficient graph algorithms based on this algorithm.

We shall now be interested in understanding the structure of sets of the form $e(L) := \{e(d) \ / \ d \in L\} \subseteq N$ where e is a mapping belonging to an F-inductive family $\mathcal{E}$ and L is M-equational. We shall need the following assumptions :

(H1) $\mathcal{E}$ has finitely many mappings of each type $s \in \mathcal{S}$,

(H2) N is an H-magma for some signature H with set of sorts $\mathcal{S}'$ ($\mathcal{S}'$ is not necessarily equal to $\mathcal{S}$),

(H3) the functions $g_{e,f}$ are derived operations of N ; each of them is defined by a term $t_{e,f}$.

A term is *linear* if no variable has more than one occurrence.

Proposition 6.9 *If conditions (H1) - (H3) hold, if in Equalities (7) we have* $n_1 \leq 1, \cdots, n_k \leq 1$ *and if the terms* $t_{e,f}$ *are linear, then* $e(L) \in Equat(N)$ *for every* $L \in Equat(M)$.

Proof : Let $L = L((S, M), u_1)$ where S is a uniform polynomial system with unknowns $u_1, \cdots, u_n$. We shall assume that M and N have only one sort : this simplifies the notation and is not a loss of generality. For every $e \in \mathcal{E}$ and $i \in [1, n]$, we let $[e, u_i]$ be a new unknown. We shall build a polynomial system S' such that the component of its least solution corresponding to $[e, u_i]$ is the set $e(L((S, M), u_i))$ for each $e \in \mathcal{E}$ and $i \in [1, n]$. For every equation

$$u_i = \cdots + f(u_{i_1}, \cdots, u_{i_k}) + \cdots$$

of S, every $e \in \mathcal{E}$ we create the equation

$$[e, u_i] = \cdots + t_{e,f}([e^1, u_{i_1}], \cdots, [e^k, u_{i_k}]) + \cdots$$

where $(g, (e^1), \cdots, (e^k))$ is the decomposition of e relative to f_M and $t_{e,f}$ is a linear term defining in N the function g. We let $(A_1^j, \cdots, A_n^j)$ be the j-th iterate (where $j \geq 0$) approximating the least solution of S in $\mathcal{P}(M)$ (see Subsection 5.1). Similarly, we denote by $A_{e,i}^j$ for $e \in \mathcal{E}$ and $i \in [1, n]$ the component corresponding to $[e, u_i]$ of the j-th iterate of $S'_{\mathcal{P}(N)}$.

Claim : *For every $j \in \mathcal{N}$, every $i \in [1, n]$, every $e \in \mathcal{E}$, we have $e(A_i^j) = A_{e,i}^j$.*

Proof : By induction on j. See Courcelle [22], Theorem 6.2 for details. □

As an application, we obtain a form of Parikh's theorem. We let $q \in \mathcal{N}_+$; we let N be the commutative monoid $< \mathcal{N}^q, \mathbf{0}, \oplus, \mathbf{1}_1, \cdots, \mathbf{1}_q >$ where $\mathbf{0} = (0, \cdots, 0), \mathbf{1}_i = (0, 0, \cdots, 1, \cdots, 0)$ (and 1 is at position i), and $\oplus$ is the vector addition : $(a_1, a_2, \cdots, a_q) \oplus (b_1, \cdots, b_q) = (a_1 \oplus b_1, \cdots, a_q \oplus b_q)$. We assume that for every $s \in \mathcal{S}$ there is in $\mathcal{E}$ a unique evaluation mapping $e_s : M_s \to N$, and that the functions $g_{e,f}$ of the decompositions are of the form :

$$g_{e,f}(x_1, \cdots, x_k) = x_1 \oplus \cdots \oplus x_k \oplus b$$

where $b \in \mathcal{N}^q$. Since b can be written as a finite sum of constants $\mathbf{0}, \mathbf{1}_1, \cdots, \mathbf{1}_q$ the operation $g_{e,f}$ is defined by a linear term in $T(\{\oplus, \mathbf{0}, \mathbf{1}_1, \cdots, \mathbf{1}_q\})$. It follows that Equalities (7) have the form :

$$e(f_M(d_1, \cdots, d_k)) = e_1(d_1) \oplus \cdots \oplus e_k(d_k) \oplus b \tag{10}$$

where e_i is the evaluation in $\mathcal{E}$ of type $\sigma(d_i)$. We shall say that $\mathcal{E}$ is a *Parikh family of evaluations* on M.

We recall a few definitions. A set $A \subseteq \mathcal{N}^q$ is *linear* if it is of the form $A = \{\lambda_1 a_1 \oplus \cdots \oplus \lambda_n a_n \oplus b \ / \ \lambda_1, \cdots, \lambda_n \in \mathcal{N}\}$ for some $a_1, \cdots, a_n, b \in \mathcal{N}^q$ and where, for $\lambda \in \mathcal{N}$, $a \in \mathcal{N}^q, \lambda a = \mathbf{0}$ if $\lambda = 0$ and $\lambda a = a \oplus a \oplus \cdots \oplus a$ with λ times a if $\lambda \geq 1$. A subset of $\mathcal{N}^q$ is *semi-linear* if it is a finite union of linear sets. Since a linear set as above can be written $A = a_1^* a_2^* \cdots a_n^* b$ it follows that every linear set, hence, every semi-linear set is rational. Conversely, by using the laws : $(A + B)^* = A^* B^*$ and $(A^* B)^* = \varepsilon + A^* B^* B$ which hold for arbitrary subsets A and B of a commutative monoid, one can transform a

rational expression defining $A \in Rat(\mathcal{N}^q)$ into a sum of terms of the form $a_1^* a_2^* \cdots a_n^* b$ for $a_1, \cdots, a_n, b \in \mathcal{N}^q$. Hence, every rational subset of $\mathcal{N}^q$ is semi-linear.

Corollary 6.10 *: If $L \in Equat(M)$ and e belongs to a Parikh family of evaluations then $e(L) \in Equat(\mathcal{N}^q)$ and is semi-linear.*

Proof : That $e(L) \in Equat(\mathcal{N}^q)$ follows from Proposition 6.9. Pilling has proved in [51] that every equational set of a commutative monoid is rational, hence, we get that $e(L)$ is semi-linear. □

Example 1 : We consider the magma of series-parallel graphs $< \mathcal{SP}, ., //, e >$ of Subsection 5.1. We consider the evaluation $\# : \mathcal{SP} \to \mathcal{N}^2$ such that :

$\#(G) =$ (number of edges of G, number of internal vertices of G).

Since $\#(G//G') = \#(G) \oplus \#(G')$, $\#(G.G') = \#(G) \oplus \#(G') \oplus \mathbf{1}_2$, and $\#(e) = \mathbf{1}_1$, from the equation $u = u//u + u.u.u + e$ which defines a proper subset L of $\mathcal{SP}$, we get the following equation that defines $\#(L) \subseteq \mathcal{N}^2$:

$$u = u \oplus u + u \oplus u \oplus u \oplus \mathbf{1}_2 \oplus \mathbf{1}_2 + \mathbf{1}_1.$$

We now define a more powerful extension of Parikh's theorem which has also applications in graph grammars. We let M be an F-magma, we let N be a G-magma, we let $\mathcal{E}$ be a finite family of mappings : $\mathcal{P}(M) \to \mathcal{P}(N)$ satisfying the following conditions, for all sets $A_1, \cdots, A_i, \cdots \subseteq M$ of appropriate sorts, and for all $f \in F$:

1. $e(A_1 \cup A_2 \cup \cdots) = e(A_1) \cup e(A_2) \cup \cdots$
2. $e(f_{P,M}(A_1, \cdots, A_k)) = \bigcup_{1 \leq i \leq m} t_{i_{\mathcal{P}(N)}}(e_{i,1}(A_1), e_{i,2}(A_2), \cdots, e_{i,k}(A_k))$

where $t_1, \cdots, t_m$ are linear terms in $T(G, \{x_1, \cdots, x_k\})$ and $e_{i,j}, 1 \leq i \leq m, 1 \leq j \leq k$ are elements of $\mathcal{E}$.

Proposition 6.11 *: With these conditions, for every $L \in Equat(M)$, we have $e(L) \in Equat(N)$.*

Proof : Essentially the same as the proof of Proposition 6.9. □

Example 2 : As in Example 1, we use series-parallel graphs, but directed ones : this means that the basic graph e is a single edge directed from the first source to the second one. Series-parallel graphs have no circuit. We denote their set by $\mathcal{SP}'$. For every graph G in $\mathcal{SP}'$, we let $\pi(G)$ be the set of lengths of directed paths in G that link the first source to the second one. Hence π maps $\mathcal{SP}'$ into $\mathcal{P}(\mathcal{N})$. For $L \subseteq \mathcal{SP}'$ we let $\Pi(L) = \bigcup\{\pi(G) \ / \ G \in L\}$. This mapping satisfies the conditions of Proposition 6.11 because it a homomorphism for union and :

$$\begin{array}{lcl} \pi(G//G') & = & \pi(G) \cup \pi(G') = t1_{\mathcal{P}(\mathcal{N})}(\pi(G)) \cup t2_{\mathcal{P}(\mathcal{N})}(\pi(G')) \\ \pi(G.G') & = & \pi(G) \oplus \pi(G') \\ & = & \{n \oplus n' / n \in \pi(G), n' \in \pi(G')\} \\ & = & t3_{\mathcal{P}(\mathcal{N})}(\pi(G), \pi(G')) \end{array}$$

where $t_1 = x_1, t_2 = x_2$ and $t_3 = x_1 \oplus x_2$. Hence, for every $\{//, .\}$-equational set of series-parallel graphs L, the set $\Pi(L)$ is a semi-linear subset of $\mathcal{N}$. For the equation of Example 2 we obtain :

$$u = u \oplus u \oplus u + \mathbf{1}_1$$

Its least solution is the set of odd numbers {1, 3, 5, 7...}. □

This example is actually a special case of a general result of [19] that we recall and that we shall derive from Proposition 6.11 and some results of [25].

If G is a hypergraph represented by the structure $\mid G \mid_2$, if φ is an MS_2-formula with free set variables $X_1, \cdots, X_k$, we let

$$sat(G, \varphi) = \{(D_1, \cdots, D_k) \ / \ D_i \subseteq V_G \cup E_G, (\mid G \mid_2, D_1, \cdots, D_k) \models \varphi\}$$

$$\#sat(G, \varphi) = \{(Card(D_1), \cdots, Card(D_k)) \ / \ (D_1, \cdots, D_k) \in sat(G, \varphi)\} \subseteq \mathcal{N}^k.$$

If L is a set of hypergraphs, we let

$$\#sat(L, \varphi) = \bigcup \{sat(G, \varphi) \ / \ G \in L\}.$$

Proposition 6.12 *If L is a HR set of hypergraphs and φ is an MS_2-formula with free variables $X_1, \cdots, X_k$, then the subset $\#sat(L, \varphi) \subseteq \mathcal{N}^k$ is semi-linear. A similar statement holds for VR sets of graphs and MS_1-formulas.*

This result is proved in [19] as Corollary 3.3. It yields immediately the result of Example 2 if one takes the formula $\varphi(X_1)$ saying that "X_1 is the set of edges of a path from the first source to the second one". We now derive the first assertion of Proposition 6.12 from Proposition 6.11 and some results of [25].

Proof of the first assertion of 6.12 : Let C, C' be finite sets of source labels ; let φ be an MS_2-formula of quantification depth at most h and free variables $X_1, \cdots, X_k$. One can find MS_2-formulas $\psi_1, \cdots, \psi_m, \psi'_1, \cdots, \psi'_m$ of quantification depth at most h and vectors $n_1, \cdots, n_m \in \mathcal{N}^k$ such that, for all hypergraphs G and G' of respective types C and C' we have :

$$\#sat(G//_{C,C'}G', \varphi) = \bigcup_{1 \leq j \leq m} \#sat(G, \psi_j) \oplus \#sat(G', \psi'_j) \oplus n_j.$$

Similarly, for every finite set C of source labels, for each unary operation f in F_{HR}, one can find MS_2 formulas $\psi_1, \cdots, \psi_m$ of quantification depth at most h, and vectors $n_1, \cdots, n_m$ in $\mathcal{N}^k$ such that, for every hypergraph G of type C :

$$sat(f(G),\varphi) = \bigcup_{1\leq j\leq m} \#sat(G,\psi_j) \oplus n_j.$$

These facts follow from Lemmas 2.4, 2.5 and 2.6 of [25]. Letting $\mathcal{E}$ be the family of evaluations e of the form $e(G) = \#sat(G,\varphi)$ for G of type C where C is a subset of a fixed finite set of source labels, we obtain that $\mathcal{E}$ satisfies the conditions of Proposition 6.11. It follows from this proposition that if L is an HR set of hypergraphs, and φ is an MS_2-formula with free variables $X_1,\cdots,X_k$ then $\#sat(L,\varphi)$ is in $Equat(\mathcal{N}^k)$ hence, by [51], is semi-linear. □

The proof of Proposition 6.12 is effective. This means that from φ and a system of equations defining L one can construct an expression of $\#sat(L,\varphi)$ as a finite union of linear sets where the linear sets are given by coefficients $a_1,\cdots,a_m,b$ (see the definition). It follows that a number of "boundness" questions can be solved effectively. It can be decided whether $Sup\{x_i \ / \ (x_1,\cdots,x_k) \in sat(L,\varphi)\}$ is finite or not, and if it is, its value can be computed. Much more generally, for every sequence integers $c_1,\cdots,c_k$, one can decide whether $Sup\{x_1c_1+\cdots+x_kc_k \ / \ (x_1,\cdots,x_k) \in sat(L,\varphi)\}$ is finite or not, and if it is, its value can be computed.

6.4 Logical characterizations of recognizability

Theorem 6.5 has established that every set of finite graphs or hypergraphs defined by a formula of an appropriate monadic second-order language is recognizable with respect to an appropriate set of operations. It is thus natural to ask for the converse, which holds in several known cases summarized in the following theorem.

Theorem 6.13 *: (1) Let A be a finite alphabet. A language $L \subseteq A^*$ is regular iff it is MS_1-definable.*
(2) Let F be a finite ranked alphabet. A set of terms $L \subseteq T(F)$ is $T(F)$-recognizable iff it is MS_1-definable.
(3) A set of finite trees is $\mathcal{TREE}$-recognizable iff it is CMS_1-definable.

Proof : Assertion (1) has been proved by Büchi and Elgot ([8], [31]), assertion (2) by Thatcher, Wright and Doner ([30], [61]) ; see [62], Thm 11.1), assertion (3) by Courcelle ([11]). □

However, no logical characterization of recognizability can exist by the following result.

Proposition 6.14 *: Every set of finite square grids is $\mathcal{HS}$-recognizable and $\mathcal{GP}(C)$-recognizable for every finite set of port labels C.*

Square grids have been defined in Section 1.9. This result is proved in Courcelle [11] for $\mathcal{HS}$-recognizability. The proof is easily adapted to yield the case of $\mathcal{GP}(C)$-recognizability. It follows that the family of $\mathcal{HS}$-recognizable (or of $\mathcal{GP}(C)$-recognizable) sets of graphs is

uncountable. Hence it cannot be characterized by the countably many formulas of any logical language like those we have considered here (including second-order logic).

Courcelle conjectured in [13] that for every k, every recognizable set of graphs of tree-width at most k is CMS_2 definable. This was proved in [13] for $k = 2$ and by Kaller [47] for $k = 3$.

7 Forbidden configurations

Many classes of graphs can be characterized in terms of *forbidden configurations*, i.e., of certain graphs that cannot appear as subgraphs. The basic example is that of planar graphs: they can be characterized as the graphs that do not contain K_5 or $K_{3,3}$ as a minor. Such characterizations yield logical descriptions that could not be obtained otherwise : for example, the definition of the planarity of a graph G in terms of an embedding in the plane cannot be expressed logically in a structure like $\mid G \mid_1$ or $\mid G \mid_2$ which describes the graph G (its vertices and edges) but not the plane. One could also try to express the existence of an embedding of G (assumed finite) in a large enough rectangular grid but this is not possible (at least immediately) because this requires to express the existence of vertices and edges forming a grid lying *outside of* G. The logical formalism allows to express properties of a graph G in terms of its vertices and edges, not in terms of objects outside of $\mid G \mid_1$ or $\mid G \mid_2$.

All graphs in this section will be finite. Hence, "graph" will mean "finite graph". We shall review the links between finitary descriptions of sets of graphs in terms of grammars, MS formulas and forbidden minors. We also show how the notion of a minor and the results of Robertson and Seymour help in understanding the structure of the sets of graphs having a decidable MS_2-theory.

7.1 Minors

Let G and H be undirected graphs. We say that G *is a minor* of H and we write this $G \trianglelefteq H$ iff there exist mappings $f : V_G \to \mathcal{P}(V_H), f' : V_G \to \mathcal{P}(E_H)$ and $g : E_G \to E_H$ satisfying the following conditions :

(1) for every $x \in V_G, (f(x), f'(x))$ is a connected subgraph of H,

(2) for every $x, x' \in V_G$ with $x \neq x'$, we have $f(x) \cap f(x') = \emptyset$,

(3) g is injective and $g(e)$ does not belong to any set $f'(x)$ for any $x \in V_G, e \in E_G$,

(4) if $e \in E_G$ links x and x', then $g(e)$ links a vertex of $f(x)$ and a vertex of $f(x')$.

It is clear that this definition depends only on the isomorphism classes of G and H. Minor inclusion is transitive and reflexive. Because graphs are finite, if $G \trianglelefteq H \trianglelefteq G$ then it is not hard to see that G and H are isomorphic. Hence minor inclusion is a partial order on graphs. (The corresponding strict order is denoted by $\triangleleft$).

7.1.1 Fact : $G \trianglelefteq H$ iff G is obtained from a subgraph H' of H by a sequence of edge contractions.

Contracting an edge linking x and y results in the fusion of x and y and deletion of the resulting loop. Edge contraction may create loops and multiple edges.

Proof : Consider $G \trianglelefteq H$ with functions f, f', g as in the definition. We let H' be the subgraph of H such that

$$V'_H = \bigcup\{f(x) \ / \ x \in V_G\}, E_{H'} = \bigcup\{f'(x) \ / \ x \in V_G\} \cup \{g(e) \ / \ e \in E_G\}.$$

Then one obtains G (up to isomorphism) by contracting (in any order) the edges in the sets $f'(x), x \in V_G$. The converse is proved similarly. □

Let M be a set of graphs. We define $FORB(M)$ as the set of undirected graphs H such that $G \trianglelefteq H$ for no $G \in M$. For example the set of undirected planar graphs can be characterized as the class $FORB(K_{3,3}, K_5)$. This is a variant of Kuratowski's theorem (see Bollobas [5]). We recall that K_n is the n-clique consisting of n vertices and an edge linking any two distinct vertices, and that $K_{n,m}$ is the complete bipartite graph with $n+m$ vertices. (Formally its vertices are $-n, -(n-1), ..., -1, 1, 2, \cdots, m$ and there is an edge linking any "negative" vertex and any "positive" one). It is clear that every class of graphs of the form $FORB(M)$ is *minor-closed* which means that every minor of every graph in the class is also in the class. In order to get a converse we let, for every class $\mathcal{C}$

$OBST(\mathcal{C}) = \{G \ / \ G \notin \mathcal{C}$, every graph H such that $H \triangleleft G$ belongs to $\mathcal{C}\}$.

$OBST(\mathcal{C})$ is defined as a set of abstract graphs : its elements are by definition pairwise nonisomorphic. This set is called the set of *obstructions of* $\mathcal{C}$.

7.1.2 Fact : *For every minor-closed class* $\mathcal{C}$ *of graphs* : $\mathcal{C} = FORB(OBST(\mathcal{C}))$.

The major result in this field is the following result by Robertson and Seymour [56].

Theorem 7.1 *: (Graph Minor Theorem) Every infinite set of undirected graphs contains two graphs comparable by* $\trianglelefteq$. *Hence for every class* $\mathcal{C}$ *of graphs, the set* $OBST(\mathcal{C})$ *is finite.*

The obstructions of minor-closed classes are known in relatively few cases. We have already mentioned planar graphs. The set of obstructions of forests is the singleton consisting of the graph reduced to a loop. That of graphs of tree-width at most 2 is $\{K_4\}$. That of graphs of tree-width at most 3 consists of 4 graphs one of which is K_5. The obstructions of the classes of graphs of tree-width at most k are not known for $k \geq 4$. Intractable computation methods are known to compute them ([49]). In the following case, one does not even know such a method. Let $\mathcal{C}$ be the class of *almost planar* graphs, i.e. of graphs G such that $G[V_G - \{v\}]$ is planar for some vertex v. (Every planar graph is almost planar and so are

K_5 and $K_{3,3}$; K_6 is not almost planar because one must delete at least two vertices in order to obtain from K_6 a planar graph). R. Thomas knows already 60 graphs in $OBST(\mathcal{C})$ one of which is K_6, but does not know whether this list is complete.

We now discuss the logical aspects of minor inclusion.

Proposition 7.2 *: Let H be a finite graph. The property that a graph G contains H as a minor is MS_2-definable. If H is simple and loop-free, this property is MS_1-definable.*

It follows in particular that planarity is MS_1-definable.

Proof : Let H be given with vertices $1, 2, \cdots, k$ and edges $e_1, \cdots, e_m$. Let G be an arbitrary graph. In order to verify that $H \trianglelefteq G$, we have to find appropriate mappings f, f', g. Hence we need only find k subsets of V_G namely $f(1), \cdots, f(k)$, k subsets of E_G namely $f'(1), \cdots, f'(k)$, m edges of G : namely $g(e_1), \cdots, g(e_m)$ and to check the conditions of the definition. This can be done easily by an MS_2 formula.

Assuming H simple and loop-free we now check that one can avoid quantifications on sets of edges. We claim that $H \trianglelefteq G$ iff there exist $X_1, \cdots, X_k \subseteq V_G$ satisfying the following conditions :

(1') the induced subgraph $G[X_i]$ of G with set of vertices X_i is connected for each $i = 1, \cdots, k$.

(2') $X_i \cap X_j = \emptyset$ for $i \neq j$,

(3') for every edge of H linking i and j there is an edge of G linking a vertex of X_i and one of X_j.

If $H \trianglelefteq G$ with corresponding mappings f, f', g then these conditions hold with $X_i = f(i)$. Conversely, if $X_1, \cdots, X_k$ satisfy them, then we take :

$f(i) = X_i$,

$f'(i) = E_{G[X_i]}$, for every $i = 1, \cdots, k$,

$g(e_j)$ to be an edge satisfying condition (3').

These mappings satisfy the conditions of minor inclusion : (1) follows from (1'), (2) follows from (2'), (3) holds from (3') and, because H is simple and loop-free, (4) follows from (3'). The existence of $X_1, \cdots, X_k$ satisfying (1') - (3') is easily expressed by an MS_1-formula. □

We leave as an exercise the verification that the property of a graph G that it contains a fixed graph H as a minor is MS_1 where H may have loops and multiple edges, and G is assumed to be simple. With Theorem 7.1 we obtain :

Corollary 7.3 : *Every minor-closed class of graphs is MS_2-definable. Its subclass of loopfree simple graphs is MS_1-definable.*

Proof : The first assertion is immediate from the Graph Minor Theorem and Proposition 7.2. For the second one, we let :

$OBST'(\mathcal{C}) := \{G \ / \ G$ is simple and loop-free, $G \notin \mathcal{C}$, every simple loop-free graph H with $H \lhd G$ belongs to $\mathcal{C}\}$.

If $\mathcal{C}$ is a minor-closed class of simple loop-free graphs, then $\mathcal{C} = FORB'(OBST'(\mathcal{C}))$ where $FORB'(M)$ is the class of simple loop-free graphs in $FORB(M)$. The set $OBST'(\mathcal{C})$ is finite by the Graph Minor Theorem and the result follows by Proposition 7.2. □

We shall denote by $\mathbf{Minor}(G)$ the set of minors of a graph G. Thus **Minor** is a transduction from graphs to graphs. We show it is (2,2)-definable. Let G be a graph. Let $X \subseteq V_G$, let $Y, Z \subseteq E_G$. We say that (X, Y, Z) *defines a minor of* G if the following conditions hold :

(α) if $x, x' \in X, x \neq x'$, then there is no Z-path in G linking them (a *Z-path* is a path all edges of which are in Z) ;

(β) each end y of an edge in Y is linked to some vertex x of X by some Z-path ; (one may have $x = y$);

(γ) $Y \cap Z = \emptyset$.

Notice that by condition (α), the vertex x in condition (β) is uniquely defined. We shall denote it by $\widehat{y}$. The *minor defined by* the triple (X, Y, Z) is the graph H such that $V_H = X, E_H = Y$ and $vert_H(e) = \{\widehat{y}, \widehat{y'}\}$ where y and y' are such that $vert_G(e) = \{y, y'\}$. We shall denote it by $\mathbf{Minor}(X, Y, Z)$. The corresponding mappings f, f', and g are as follows : $f(v)$ is the set of vertices of G linked to v by some Z-path (including v) ; $f'(v)$ is the set of edges of Z having their ends in $f(v)$; and g is the identity : $V_H \to V_{G'}$. Hence, $\mathbf{Minor}(X, Y, Z)$ is indeed a minor of G. It is not hard to see that every minor H of G is of this form for appropriate sets X, Y, Z. Moreover, $\mathbf{Minor}(X, Y, Z)$ is a strict minor iff $Z \neq \emptyset$ or $Y \neq E_G$ or $X \neq V_G$. We have thus proved the following

7.1.3 Fact : Minor *is a* (2,2)*-definable transduction.*

Proposition 7.4 *For every MS_2-definable class M of graphs, each of the classes $FORB(M)$ and $OBST(M)$ is MS_2-definable. MS_2-formulas defining them can be effectively constructed from an MS_2-formula defining M.*

Proof : Let φ be an MS_2-formula such that $M = \{G \ / \ \mid G \mid_2 \ \models \varphi\}$. Then $H \in FORB(M)$ iff :

$| H |_2 \models \forall X, Y, Z[$"$X, Y, Z$ define a minor of H" $\Longrightarrow$
"**Minor** (X, Y, Z) does not satisfy φ"].

From the proof of Fact 7.1.3 this condition can be expressed by an MS_2-formula. From the definition of $OBST$ we get that for every graph H, $H \in OBST(M)$ iff :

$| H |_2 \models \neg\varphi \wedge \forall X, Y, Z[$"$X, Y, Z$ define a strict minor of H" $\Longrightarrow$
"**Minor**(X, Y, Z) satisfies φ"]

Again this latter condition is MS_2-expressible. □

One might hope to be able to construct the sets of obstructions of MS_2-definable classes of graphs. Consider for instance the class of almost planar graphs. It is not hard to prove that it is minor-closed, and to construct an MS_2-formula defining it. One obtains thus an MS_2-formula characterizing its finite set of obstructions. However this is not enough to make it possible to construct (even by an intractable algorithm) this set itself. (See the remark following Proposition 1.2.). The best one can do presently is the following

Proposition 7.5 *: Let M be an MS_2-definable class of finite graphs. For every HR set of graphs L one can compute the set $L \cap OBST(M)$.*

In particular, taking L to be the set of trees (or of graphs of tree-width at most k), one can obtain effectively the obstructions of M that are trees or that have tree-width at most k. If one knows some upper-bound on the tree-width of the graphs in $OBST(M)$ one can thus (at least in principle) compute $OBST(M)$ explicitely.

Proof : This follows immediately from Propositon 7.4, Corollary 6.7 and the fact that if a finite set of graphs is HR (given by a known grammar or system of equations), then this set can be explicitely enumerated. (See Habel [45]). □

This result was proved first in a different way by Fellows and Langston [41].

Sets of finite graphs can be specified effectively in various ways : by grammars of several types, by forbidden minors, by monadic second-order formulas. The following theorem summarizes the situation.

Theorem 7.6 *: Let L be a set of loop-free simple graphs containing the simple and loop-free minors of its members. The following properties are equivalent :*
(1) L is HR,
(2) L is VR,
(3) L has bounded tree-width,
(4) $OBST(L)$ contains a planar graph.

Assuming these conditions satisfied, from any of the following devices defining L one can construct all others, namely :

(5) an HR grammar (or F_{HR}-system of equations),
(6) a VR grammar (or F_{VR}-system of equations),
(7) an MS_1-formula,
(8) the set $OBST(L)$.

Proof : The set L is MS_1-definable by Corollary 7.3 (based on the Graph Minor Theorem). We have the following implications :

(3) $\Longleftrightarrow$ (4) by Robertson and Seymour [55]
(3) $\Longleftrightarrow$ (1) by Corollary 6.7 (2) since the set of graphs of tree-width at most k is HR (see [20])
(1) $\Longrightarrow$ (2) by Theorem 5.15
(2) $\Longrightarrow$ (1) by a result of [23].

We now consider effective constructions of such devices.

(5) $\Longleftrightarrow$ (6) because the proofs of Theorem 5.15 and [23] are effective.
(8) $\Longleftrightarrow$ (7) by Proposition 7.2
(7) $\Longleftrightarrow$ (5) because Corollary 6.7 is effective and one can find the smallest square grid not in L ; from this grid, one gets an upper bound on the tree-width of L by [55].
(5) $\Longleftrightarrow$ (8) by a result of Courcelle and Sénizergues [27]. □

7.2 The structure of sets of graphs having decidable monadic theories

By combining techniques from logic (definable graph transductions) and from graph theory (minor inclusion), one obtains the following result of Seese [58].

Theorem 7.7 *: If a set of graphs has a decidable MS_2-theory, then it is a subset of some HR set of graphs.*

Proof sketch : The mapping from a graph to the set of its minors is (2,2)-definable. Hence, if a set L of graphs has a decidable MS_2-theory, then so has the set of its minors. But this set cannot contain all square grids by Proposition 1.2. Hence L has bounded tree-width (by [55]), hence L is a subset of the set of all graphs of tree-width at most k for some k which is HR. □

This result is a kind of converse of the one (Corollary 6.7) saying that the MS_2-theory of a HR set is decidable. We make the following conjecture.

Conjecture 7.8 : *If a set of graphs has a decidable MS_1-theory then it is a subset of some VR set of graphs.*

Special cases of this conjecture are proved in Courcelle [16]. Seese [58] has made a conjecture which is equivalent to Conjecture 7.8 by the result of [34] in the generalized form given in [23].

Acknowledgements

Many thanks to Mrs. A. Dupont and to A. Pariès for the Latex typing, and to E. Grandjean and the referees for comments.

References

1. : ABITEBOUL S., HULL R., VIANU V., Foundations of Data bases, Addison Wesley, 1994.

2. : ARNBORG S., LAGERGREN J., SEESE D., Problems easy for tree-decomposable graphs, J. of Algorithms **12** (1991) 308-340.

3. : BAUDERON M., COURCELLE B., Graph expressions and graph rewritings, Mathematical Systems Theory **20** (1987) 83-127.

4. : BERSTEL J., Transductions and context-free languages, Teubner Verlag, Stuttgart, 1979.

5. : BOLLOBAS B., Extremal Graph Theory, Academic press, New York, 1978.

6. : BOSSUT F., DAUCHET M., WARIN B., A Kleene theorem for a class of planar acyclic graphs, Information and Computation **117** (1995) 251-265.

7. : BOURBAKI N., Algèbre, Chap. 1, Hermann, Paris, 1970.

8. : BUCHI J., Weak second-order logic and finite automata, Z. Math. Logik Grundlagen Math. **5** (1960) 66-92.

9. : COURCELLE B., An axiomatic definition of context-free rewriting and its application to NLC graph grammars, Theoret. Comput. Sci. **55** (1987) 141-181.

10. : COURCELLE B., Graph rewriting : An algebraic and logic approach, in "Handbook of Theoretical Computer Science, Volume B", J. Van Leeuwen ed., Elsevier, 1990, pp. 193-242.

11. : COURCELLE B., The monadic second-order logic of graphs I : Recognizable sets of finite graphs. Information and Computation **85** (1990) 12-75.

12. : COURCELLE B., The monadic second-order logic of graphs III : Tree-decompositions, minors and complexity issues, Informatique Théorique et Applications **26** (1992) 257-286.

13. : COURCELLE B., The monadic second-order logic of graphs V : On closing the gap between definability and recognizability, Theoret. Comput. Sci. **80** (1991) 153-202.

14. : COURCELLE B., The monadic second-order logic of graphs VI : On several representations of graphs by relational structures, Discrete Applied Mathematics, **54** (1994) 117-149.

15. : COURCELLE B., The monadic second-order logic of graphs VII : Graphs as relational structures, Theoret. Comput. Sci. **101** (1992) 3-33.

16. : COURCELLE B., The monadic second-order logic of graphs VIII : Orientations, Annals Pure Applied Logic **72** (1995) 103-143.

17. : COURCELLE B., The monadic second-order logic of graphs IX : Machines and their behaviours, Theoret. Comput. Sci. **151** (1995) 125-162.

18. : COURCELLE B.,The monadic second-order logic of graphs X : Linear orderings, Theoret. Comput. Sci., to appear (1996).

19. : COURCELLE B., Structural properties of context-free sets of graphs generated by vertex replacement, Information and Computation, **116** (1995) 275-293.

20. : COURCELLE B., Graph grammars, monadic second-order logic and the theory of graph minors in "Graph Structure Theory", Contemporary Mathematics **147** American Mathematical Society (1993) 565-590.

21. : COURCELLE B., Monadic second-order definable transductions : a survey, Theoret. Comput. Sci., **126** (1994) 53-75.

22. : COURCELLE B., Basic notions of universal algebra for language theory and graph grammars, to appear in Theoret. Comput. Sci., 1996.

23. : COURCELLE B., ENGELFRIET J., A logical characterization of the sets of hypergraphs defined by hyperedge replacement grammars, Mathematical Systems Theory **28** (1995) 515-522.

24. : COURCELLE B., ENGELFRIET J., ROZENBERG G., Handle-rewriting hypergraph grammars **46** (1993) 218-246.

25. : COURCELLE B., MOSBAH M., Monadic second-order evaluations on tree-decomposable graphs, Theoret. Comput. Sci. **109** (1993) 49-82.

26. : COURCELLE B., OLARIU S., Clique-width, a new complexity measure on graphs, preprint, 1995.

27. : COURCELLE B., SENIZERGUES G., The obstructions of minor-closed sets of graphs defined by a context-free grammars, Proceedings of the international workshop on Graph Grammars, Williamsburg, 1994, to appear L.N.C.S.

28. : DAUCHET M., HEUILLARD T., LESCANNE P., TISON S., Decidability of the confluence of finite ground term rewrite systems and of other related term rewrite systems, Information and Computation **88** (1990) 187-201.

29. : DREWES F., HABEL A., KREOWSKI H.-J., Hyperedge replacement graph grammars, this volume.

30. : DONER J., Tree acceptors and some of their applications, J. Comput. Syst. Sci. **4** (1970) 406-451.

31. : ELGOT C., Decision problems of finite automata design and related arithmetics, Trans. A.M.S. **98** (1961) 21-52.

32. : EHRIG H., MAHR B., Fundamentals of algebraic specifications 1 : equations and initial semantics, Springer, 1985.

33. : ENGELFRIET J., Context-free NCE graph grammars, Proc. FCT'89, Lec. Notes Comp.Sci. **380** (1989) 148-161.

34. : ENGELFRIET J., A characterization of context-free NCE graph languages by monadic second-order logic on trees, Lec. Notes Comp.Sci. **532** (1991) 311-327.

35. : ENGELFRIET J., HEYKER L., The string generating power of context-free hypergraph grammars, J. Comp. Syst. Sci. **43** (1991) 328-360.

36. : ENGELFRIET J., HEYKER L., Hypergraph languages of bounded degree, J. Comp. Syst. Sci. **48** (1994) 58-89.

37. : ENGELFRIET J., ROZENBERG G., Graph grammars based on node rewriting : an introduction to NLC graph grammars, Lec. Notes Comp.Sci. **532** (1991) 12-23.

38. : ENGELFRIET J., ROZENBERG G., Node replacement graph grammars, this volume.

39. : FAGIN R., Generalized first-order spectra and polynomial-time recognizable sets, in "Complexity of Computation", SIAM-AMS Proceedings (1974) 43-73.

40. : FEFERMAN S., VAUGHT R., The first-order properties of products of algebraic systems, Fund. Math. **47** (1959) 57-103.

41. : FELLOWS M., LANGSTON M., An analogue of the Myhill-Nerode theorem and its use in computing finite basis characterizations, 30th symp. FOCS (1989) 520-525.

42. : GECSEG F., STEINBY M., Tree Automata, Akademiai Kiado, Budapest, 1984.

43. : GINSBURG S., RICE , Two families of languages related to ALGOL, J. ACM **9** (1962) 350-371.

44. : GUREVICH Y., Monadic second-order theories, in J. Barwise and S. Feferman eds., "Model theoretic logic", Springer, Berlin, 1985, pp. 479-506.

45. : HABEL A., Hyperedge replacement : grammars and languages, Lect. Notes Comput. Sci. **643**, 1992.

46. : IMMERMAN N., Languages which capture complexity classes, SIAM J. Comput. **16** (1987) 760-778.

47. : KALLER D., Definability equals recognizability of partial 3-trees, preprint, 1995.

48. : KANNELLAKIS P., Elements of relational data base theory, "Handbook of Theoretical Computer Science, Vol. B", J. Van Leeuwen ed., Elsevier 1990, pp. 1073-1156.

49. : LAGERGREN J., ARNBORG S., Finding minimal forbidden minors using a finite congruence, LNCS **510** (1991) 532-543.

50. : MEZEI J., WRIGHT J., Algebraic automata and context-free sets, Information and Control **11** (1967) 3-29.

51. : PILLING D., Commutative regular equations and Parikh's theorem, J. London Math. Soc. **6** (1973) 663-666.

52. : PIN J.-E., Logic, semigroups and automata on words, to appear in Annals of Mathematics and Artificial Intelligence.

53. : RABIN M., A simple method for undecidability proofs and some applications, in "Logic, Methodology and Philosophy of Science II", Y. Bar-Hillel ed., North-Holland, Amsterdam, 1965, pp.58-68.

54. : RAOULT J.-C., A survey of tree transductions, in "Tree automata and languages", M. Nivat and A. Podelski eds., Elsevier, 1992, pp. 311-326.

55. : ROBERTSON N., SEYMOUR P., Graph minors V : Excluding a planar graph, J.

Comb. Theory Ser. B **52** (1986) 92-114.

56. : ROBERTSON N., SEYMOUR P., Graph minors XX : Wagner's conjecture, September 1988.

57. : ROZENBERG G., WELZL E., Boundary NLC grammars, Basic definitions, normal forms and complexity, Information and Control **69** (1986) 136-167.

58. : SEESE D., The structure of the models of decidable monadic theories of graphs, Annals Pure Applied Logic 53 (1991) 169-195.

59. : SHELAH S., The monadic theory of order, Annals of Maths **102** (1975) 379-419.

60. : STOCKMEYER L., The polynomial-time hierarchy, TCS **3** (1977) 1-22.

61. : THATCHER J., WRIGHT J., Generalized finite automata theory with an application to a decision problem in second-order logic, Math. Systems Theory **3** (1968) 57-81.

62. : THOMAS W., Automata on infinite objects, "Handbook of Theoretical Computer Science, Volume B" Elsevier 1990, pp. 133-192.

63. : THOMAS W., Classifying regular events in symbolic logic, J. Comput. System Sci. **25** (1982) 360-376.

64. : TRAKHTENBROT B., Impossibility of an algorithm for the decision problem on finite classes, Dokl. Akad. Nauk.SSSR **70** (1950) 569-572.

65. : WECHLER W., Universal Algebra for Computer Scientists, Springer, 1992.

66. : WIRSING M., Algebraic Specifications, "Handbook of Theoretical Computer Science, Volume B" J. Van Leeuwen ed., Elsevier 1990, 675-779.