Theoretical Computer Science 80 (1991) 153-202
Elsevier

The monadic second-order logic of graphs V: on closing the gap between definability and recognizability*

Bruno Courcelle

Laboratoire d'Informatique, U.A. CNRS, Université Bordeaux-1, 351 Cours de la Libération, 33405 Talence, France

Abstract

Courcelle, B., The monadic second-order logic of graphs V: on closing the gap between definability and recognizability, Theoretical Computer Science 80 (1991) 153-202.

Context-free graph-grammars are considered such that, in every generated graph G, a derivation tree of G can be constructed by means of monadic second-order formulas that specify its nodes, its labels, the successors of a node etc. A subset of the set of graphs generated by such a grammar is recognizable iff it is definable in monadic second-order logic, whereas, in general, only the "if" direction holds.

Introduction

A fundamental theorem by Büchi [3] states that a language is recognizable iff it is definable in *monadic second-order logic* (MSOL; this logic uses quantifications over objects and sets of objects). This result has been extended to finite ranked ordered trees by Doner [11], and to sets of finite unranked unordered trees by Courcelle [7]. This latter extension uses an extension of MSOL called *counting* monadic second-order logic (CMSOL), making it possible to count the cardinalities of sets modulo positive integers.

These three results relate an algebraic aspect, namely *recognizability*, defined in terms of congruences having finitely many classes, to a logical one. Their proofs use as an intermediate tool a third notion, that of a finite-state string or tree

* This work has been supported by the "Programme de recherches coordonnées: Mathématiques et Informatique" and by the ESPRIT-BRA project GRAGRA ("Computing by graph transformations", Contract 3299).

0304-3975/91/$03.50

automaton. Our aim is to extend them to sets of graphs ("graph" means "finite hypergraph" in this paper). Since a graph can be considered as a logical structure, graph properties can be expressed by logical formulas. From this we derive the notion of a MSO-definable set of graphs, i.e., of a set of graphs characterized by a graph property expressed by a MSO formula. Graph operations (that for instance glue two graphs in a certain precise way) make it possible to equip the set of graphs with an algebraic structure, to define the notion of a congruence on the corresponding algebra, and to define the notion of a recognizable set of graphs.

One half of the analogue of Büchi's theorem holds: every MSO-definable (and even CMSO-definable) set of graphs is recognizable [7]. The other half does not: the set of square $n \times n$-grids, where n ranges over a nonrecursive set, is recognizable but is not definable. However, this counterexample uses a set of graphs of infinite tree-width, i.e., for which infinitely many graph operations are necessary to define its elements by finite graph expressions. It leaves open the case of sets of graphs of bounded tree-width. We make the following conjecture.

Conjecture 1. *If a set of graphs of bounded tree-width is recognizable, then it is CMSO-definable.*

In this paper, we propose a method that may lead to a proof of this conjecture.

Let us explain why the proofs of the three results by Büchi, Doner, and Courcelle concerning words and trees do not extend to graphs. These proofs use finite-state automata, and no such notion is known for graphs. It is not clear at all how an automaton should traverse a graph. A "general" graph has no evident structure, whereas a word or a tree is (roughly speaking) its own algebraic structure. Automata are useful because they can realize congruences on strings or trees, and because their behaviours can be simulated by MSO- or CMSO-formulas.

However, some graphs have a well-defined structure: when a graph G is generated by a context-free graph-grammar, then any derivation tree of this graph can be considered as its structure, and can be traversed by a finite-state tree-automaton. Such a tree-automaton can realize a congruence having finitely many classes. The central idea of this paper is: if a derivation tree of the considered graph G can be constructed in G by means of monadic second-order formulas, then, the behaviour of the automaton traversing the derivation tree can be described in G itself, by a MSO formula.

A context-free graph-grammar is *MSO-parsable* if, in every graph it generates, a derivation tree of this graph can be constructed by means of monadic second-order formulas, in a uniform way. We say that the set of graphs generated by such a grammar is *strongly context-free*. Our main theorem states that, for every subset of a strongly context-free set of graphs, recognizability implies MSO-definability. These notions extend to CMSOL in an easy way. Conjecture 1 is a consequence of the following.

Conjecture 2. *For every k, the set of graphs of tree-width at most k is strongly context-free.*

We prove this conjecture in the case where $k = 2$. The main step of the proof consists of establishing that the set of oriented series-parallel graphs is strongly context-free.

Let us mention that the exact definition of a CMSO-parsable grammar uses certain ***reduced derivation trees***, which we describe informally at the end of this introduction. By using derivation trees instead of reduced derivation trees, one would get a strictly weaker notion of strong context-freeness, and Conjecture 2 would be false.

If a set of graphs is strongly context-free, then it is CMSO-definable. Our main theorem (Theorem 4.8) entails that our second conjecture is equivalent to the following (see Conjectures 4.12 for a precise discussion of these conjectures and their relations).

Conjecture 3. *If a set of graphs is context-free and CMSO-definable, then it is strongly context-free.*

Note that we do not conjecture that every context-free graph grammar generating a CMSO-definable set is CMSO-parsable; this statement is actually false, and we shall give a counterexample.

We also introduce new notions, we prove results of independent interest, and we make other conjectures. We now review a few of them.

(1) We introduce *graph transductions*, and consider those that are definable in CMSOL.

The notion of a rational transduction is essential in the theory of context-free languages. Tree transductions are also important in many respects. A transduction is any nondeterministic (multivalued) mapping from words to words or from trees to trees. To be of any interest, a transduction must be specified in some finitary way, for instance by a generalized sequential machine, or a tree-transducer.

The general notion of a transduction can easily be extended to graphs, and even to relational structures. We do not specify graph transductions by machines or automata, but by monadic second-order logical formulas. We introduce and use transductions that we call definable. The transduction mapping a derivation tree to the graph it generates is definable. Its inverse is definable for CMSO-parsable grammars (rigorous definitions are given in Sections 2 and 4).

(2) A tree is usually an ordered graph representing a term, written with function symbols of a fixed arity, constants, and variables. If an operation symbol like + is associative and commutative, then a term like $+(x, +(y, z))$ can be written equally well $+(x, y, z)$ or $+(y, x, z)$. The symbol + is no longer binary (it becomes of variable arity) and the order of the arguments is irrelevant (in other words, they form a set and not a sequence). All these equivalent notations can be represented by a single tree such that the successors of a node labelled by + form a set (as opposed to a

sequence), the cardinality of which is not fixed. In order to formalize this idea, we introduce *reduced trees*, i.e., trees built with one associative and commutative operation symbol, its unit, and "arbitrary" operation symbols (denoting operations having no special property).

We conjecture that the recognizability of an equational set of reduced trees is decidable. (A set is equational if it is a component of least solution of a system of recursive set equations, written with appropriate operation symbols.) The decidability of the recognizability of a rational set in the free commutative monoid is a special known case of this result. We give easily testable sufficient conditions for this property.

(3) We define a class of context-free graph-grammars, which we call *regular* because of structural similarities with the regular tree-grammars. These grammars are CMSO-parsable.

This paper is organized as follows. We review graphs and context-free graph grammars in Section 1, and monadic second-order logic in Section 2. We also introduce definable graph transductions in Section 2. In Section 3, we introduce reduced trees. We introduce strongly context-free sets of graphs in Section 4, and we investigate their properties. We introduce the regular graph-grammars in Section 5, and we prove that they generate strongly context-free sets of graphs. In Section 6, we prove that the set of series-parallel graphs and the set of graphs of tree-width at most 2 are strongly context-free.

Notation

We denote by $\mathbb{N}$ the set of non-negative integers, and by $\mathbb{N}_+$ the set of positive integers. We denote by $[n]$ the interval $\{1, \ldots, n\}$ with, in addition, $[0] = \emptyset$.

The set of nonempty sequences of elements of a set A is denoted by A^+, and sequences are denoted by $(a_1, \ldots, a_n)$ with commas and parentheses. The empty sequence is denoted by $(\,)$, and A^* is $A^+ \cup \{(\,)\}$. The jth element of a sequence s is denoted by $s(j)$.

We use $:=$ for "equal by definition", i.e., for introducing a new notation. The notation $:\Leftrightarrow$ is used similarily for defining logical conditions.

Let $\mathcal{S}$ be a set. A (many-sorted) *$\mathcal{S}$-signature* is a set F given with a mapping $\mathbf{prof}: F \rightarrow \mathcal{S}^* \times \mathcal{S}$. We say that $\mathcal{S}$ is the *set of sorts* of F and that $\mathbf{prof}(f)$ is the *profile* of f. We also write

$$f: s_1 \times \cdots \times s_k \rightarrow s$$

in order to state that $\mathbf{prof}(f) = ((s_1, s_2, \ldots, s_k), s)$. The integer k is the *rank* $\rho(f)$ of f.

As in many other works, e.g., [4, 7, 10], we call *F-magma* what is more usually called an *F-algebra*, i.e., an object $M = \langle (M_s)_{s \in \mathcal{S}}, (f_M)_{f \in F} \rangle$ where each M_s is a set (called the *domain of sort* s of M) and each f_M is a total mapping:

$$M_{s_1} \times \cdots \times M_{s_k} \to M_s \text{ if } f \text{ is of profile: } s_1 \times \cdots \times s_1 \to s.$$

We denote by $\mathbf{M}(F)$ the *initial F-magma*, and by $\mathbf{M}(F)_s$ its domain of sort s. This set can be identified with the set of well-formed ranked trees. We denote by h_M the unique homomorphism $\mathbf{M}(F) \to M$ where M is an F-magma. If $t \in \mathbf{M}(F)_s$, the image of t under h_M is an element $h_M(t)$ of M_s, also denoted by t_M. One considers t as *an expression denoting* t_M, and t_M as *the value* of t in $\mathbf{M}$.

By a *system of (set) equations*, we mean a tuple $S = \langle u_1 = t_1, \ldots, u_n = t_n \rangle$. Its unknowns are the symbols $u_1, \ldots, u_n$, and the terms $t_1, \ldots, t_n$ defining them are formal sums (unions) of terms in $\mathbf{M}(F \cup \mathbf{Unk}(S))$, where we denote by $\mathbf{Unk}(S)$ the set of unknowns of S. One also assumes that each unknown has a sort in $\mathcal{S}$, and that all the terms in the right-hand side of its defining equation are of that sort. If M is an F-magma, then S has a least solution, where the value of an unknown is a subset of the domain of M of the corresponding sort. A set is *M-equational* if it is a component of the least solution in M of such a system. See [4] for a detailed study of these systems.

1. Graphs and context-free graph grammars

We review the basic definitions from [2] and [7]. As in these papers, we deal with a certain class of oriented hypergraphs, which we call simply graphs. The following notions are recalled or introduced: graphs, graph operations, context-free (hyperedge-replacement) graph-grammars, recognizable sets of graphs, tree-width of a graph, presentation of a set of graphs.

Definition 1.1 (*Graphs*). The *(hyper)graphs* we define have labelled (hyper)edges. The alphabet of edge labels is a *ranked alphabet* A, i.e., an alphabet that is given with a mapping $\tau : A \to \mathbb{N}$ (the integer $\tau(a)$ is called the *type* of a). A *graph over* A *of type* n is a 5-tuple $H = \langle \mathbf{V}_H, \mathbf{E}_H, \mathbf{lab}_H, \mathbf{vert}_H, \mathbf{src}_H \rangle$ where $\mathbf{V}_H$ is the set of vertices, $\mathbf{E}_H$ is the set of edges, $\mathbf{lab}_H$ is a mapping $\mathbf{E}_H \to A$ defining the *label* of an edge, $\mathbf{vert}_H$ is a mapping $\mathbf{E}_H \to \mathbf{V}_H^*$, defining the (possibly empty) *sequence of vertices* of an edge, and $\mathbf{src}_H$ is a sequence of vertices of length n. We impose the condition that the length of $\mathbf{vert}_H(e)$ is equal to $\tau(\mathbf{lab}_H(e))$, for all e in $\mathbf{E}_H$. One may also have labels of type 0, labelling edges with no vertex. An element of $\mathbf{src}_H$ is called a *source* of H. The sets $\mathbf{E}_H$ and $\mathbf{V}_H$ are assumed to be finite and disjoint.

We denote by $\mathbf{FG}(A)$ the set of all graphs over A, by $\mathbf{FG}(A)_n$ the set of those of type n. A graph of type n is also called an *n-graph*. By a *binary graph*, we mean a graph all edges of which are of type 2 (and not a 2-graph).

For every integer n in $\mathbb{N}$, we denote by $\mathbf{n}$ the n-graph with n vertices, no edge, and n pairwise distinct sources. If $a \in A$, we denote by a the $\tau(a)$-graph H with $\mathbf{V}_H = [\tau(a)]$, $\mathbf{E}_H = \{*\}$, $\mathbf{lab}_H(*) = a$, $\mathbf{vert}_H(*) = \mathbf{src}_H = (1, \ldots, \tau(a))$. Hence, A is considered as a subset of $\mathbf{FG}(A)$.

In general, we consider two isomorphic graphs as equal. However, in some proofs, we fix one graph H with its sets $\mathbf{V}_H$ and $\mathbf{E}_H$ of vertices and edges, and we consider various subgraphs of H. In such cases (made precise in the text), we consider as equal two subgraphs only if they have the same sets of vertices and the same sets of edges.

The notion of tree-decomposition of a graph, and the associated notion of tree-width are essential in the study of sets of graphs defined by forbidden minors [20], and for the construction of polynomial graph algorithms (see [1] and the references listed in [5-8]) because they provide *structurings of graphs*. For this latter reason, they also appear in the study of context-free sets of graphs. They have been originally defined for binary graphs. The extension to (hyper)graphs is straightforward.

Definition 1.2 (*Tree-width*). Let G be a graph. A *tree-decomposition* of G is a pair (T,f) consisting of an unoriented tree T, and a mapping $f:\mathbf{V}_T \to \mathcal{P}(\mathbf{V}_G)$ such that:

(1) $\mathbf{V}_G = \bigcup\{f(i) / i \in \mathbf{V}_T\}$,

(2) every edge of G has all its vertices in $f(i)$ for some i.

(3) if $i, j, k \in \mathbf{V}_T$, and if j is on the unique loop-free path in T from i to k, then $f(i) \cap f(k) \subseteq f(j)$,

(4) all sources of G are in $f(i)$ for some i in $\mathbf{V}_T$.

The *width* of such a decomposition is defined as

$$\mathbf{Max}\{\mathbf{card}(f(i)) \mid i \in \mathbf{V}_T\} - 1.$$

The *tree-width* of G is the minimum width of a tree-decomposition of G. It is denoted by $\mathbf{twd}(G)$. For a 0-graph, condition (4) is always satisfied in a trivial way. Similarily, condition (2) is always satisfied for the edges of type 0 or 1 (provided condition (1) holds). Such edges can be added to or deleted from a graph without changing its tree-width. Trees are of tree-width 1, series-parallel graphs are of tree-width 2 (or 1 in degenerated cases), a clique with n vertices is of tree-width n.

The *tree-width of a set L* of graphs (denoted by $\mathbf{twd}(L)$) is the least upper bound in $\mathbb{N} \cup \{\infty\}$ of $\{\mathbf{twd}(G) \mid G \in L\}$. The set of finite cliques and the set of finite square grids are of infinite tree-width.

We now define the substitution of a graph for an edge in a graph. From this basic notion, we shall define several important notions: graph operations, context-free graph-grammars, and recognizable sets of graphs.

Definition 1.3 (*Substitutions*). Let $G \in \mathbf{FG}(A)$, let $e \in \mathbf{E}_G$; let $H \in \mathbf{FG}(A)$ be a graph of type $\tau(e)$. We denote by $G[H/e]$ the result of the *substitution* of H for e in G. This graph can be constructed as follows:

- construct a graph G' by deleting e from G (but keep the vertices of e);
- add to G' a copy $\bar{H}$ of H, disjoint from G';

- fuse the vertex $\mathbf{vert}_G(e, i)$, i.e., the ith element of the sequence $\mathbf{vert}_G(e)$ (that is still a vertex of G'), with the ith source of $\bar{H}$; this is done for all $i = 1, \ldots, \tau(e)$;
- the sequence of sources of $G[H/e]$ is that of G'.

If $e_1, \ldots, e_k$ are pairwise distinct edges of G, if $H_1, \ldots, H_k$ are graphs of respective types $\tau(e_1), \ldots, \tau(e_k)$, then the substitutions in G of H_1 for $e_1, \ldots, H_k$ for e_k can be done in any order; the result is the same, and it is denoted by $G[H_1/e_1, \ldots, H_k/e_k]$.

Definition 1.4 (*Graph operations*). A *graph operation* is a mapping $f: \mathbf{FG}(A)_{n_1} \times \cdots \times \mathbf{FG}(A)_{n_k} :\rightarrow \mathbf{FG}(A)_n$ such that, for every k-tuple $(H_1, \ldots, H_k)$, where H_i is a graph of type n_i:

$$f(H_1, \ldots, H_k) = G[H_1/e_1, \ldots, H_k/e_k]$$

for some fixed graph G of type n, some fixed edges $e_1, \ldots, e_k$ of G of respective types $n_1, \ldots, n_k$. We say that f is of profile $n_1 \times \cdots \times n_k \rightarrow n$, and that it is defined by the tuple $(G, e_1, \ldots, e_k)$. We may have $k = 0$. Then f is a constant, the value of which is G.

A *signature of graph operations* is a pair $\pi = (P, {}^{-})$ where P is a signature with set of sorts $\mathscr{S} \subseteq \mathbb{N}$, and for every p in P, $\bar{p}$ is a tuple $(G, e_1, \ldots, e_k)$ as above, defining a graph operation, also denoted by $\bar{p}$, that has the profile of p. A P-magma $\mathbf{FG}_\pi$ is associated with π as follows: its domains are the sets $\mathbf{FG}(A)_n$ for n in $\mathscr{S}$, and the operations are the $\bar{p}$s. We denote by $\mathbf{h}_\pi$ the unique homomorphism $\mathbf{M}(P) \rightarrow \mathbf{FG}_\pi$.

A *presentation* of a set of n-graphs L is a pair (π, K) where π is a finite signature of graph operations, K is a subset of $\mathbf{M}(P)_n$ such that $L = \mathbf{h}_\pi(K)$. If $G = \mathbf{h}_\pi(t)$, then we say that t is a *syntactic tree* of G. The *parsing problem* consists of finding a syntactic tree of a given graph, in the context of a fixed signature π.

Definition 1.5 (*Context-free graph-grammars*). A *context-free* (*hyperedge replacement*) *graph-grammar* is a 4-tuple $\Gamma = \langle A, U, Q, Z \rangle$ where A is the finite *terminal* ranked alphabet, U is the finite *nonterminal* ranked alphabet, Q is the finite set of *production rules*, i.e., is a finite set of pairs of the form (u, D), usually written $u \rightarrow D$, where $D \in \mathbf{FG}(A \cup U)_{\tau(u)}$ and $u \in U$, and Z is a graph in $\mathbf{FG}(A \cup U)$ called the *axiom*. The set of graphs defined by Γ is $\mathbf{L}(\Gamma) := \mathbf{L}(\Gamma, Z)$ where for every graph $K \in \mathbf{FG}(A \cup U)_n$,

$$\mathbf{L}(\Gamma, K) := \{H \in \mathbf{FG}(A)_n \mid K \xrightarrow{*}_Q H\},$$

and $\rightarrow_Q$ is the *elementary rewriting step* defined as follows: $K \rightarrow_Q H$ iff there exists an edge e in K, the label of which is some u in U, and a production rule (u, D) in P, such that $H = K[D/e]$, i.e., such that H is the result of the *replacement* (i.e., substitution) of D for e in K.

A set of graphs is *context-free* if it is defined by a context-free graph-grammar. We denote by $\mathbf{CF}(A)_n$ the family of context-free subsets of $\mathbf{FG}(A)_n$.

The axiom Z of a context-free graph-grammar will be assumed to be a nonterminal symbol. This is not a loss of generality since, if this is not the case, one can add a new nonterminal symbol u_0 and a rule $u_0 \to Z$ in order to define a set of the form $\mathrm{L}(\Gamma, Z)$ where Z is not in U, by a grammar with the above condition.

Example 1.6 (*Oriented series-parallel graphs*). Let A consist of symbols a, b, and c, all of type 2. The set **SP** of *oriented series-parallel graphs over* A is the subset of $\mathbf{FG}(A)_2$ generated by the context-free grammar Γ, the set of production rules of which is shown in Fig. 1, with one rule of the first form for each symbol x in A. An example of a graph belonging to $\mathrm{L}(\Gamma)$ is also shown in Fig. 1.

We call context-free the graph-grammars introduced in Definition 1.5 because their derivation sequences can be described by derivation trees and because the sets they generate can be characterized as least solutions of systems of equations. Both notions can be introduced in an algebraic setting borrowed from [14] (see also [4]).

Definition 1.7 (*Systems of equations in sets of graphs*). Let P be a set of *names* given to the production rules of a context-free graph grammar Γ. We write $p: u \to D$ to

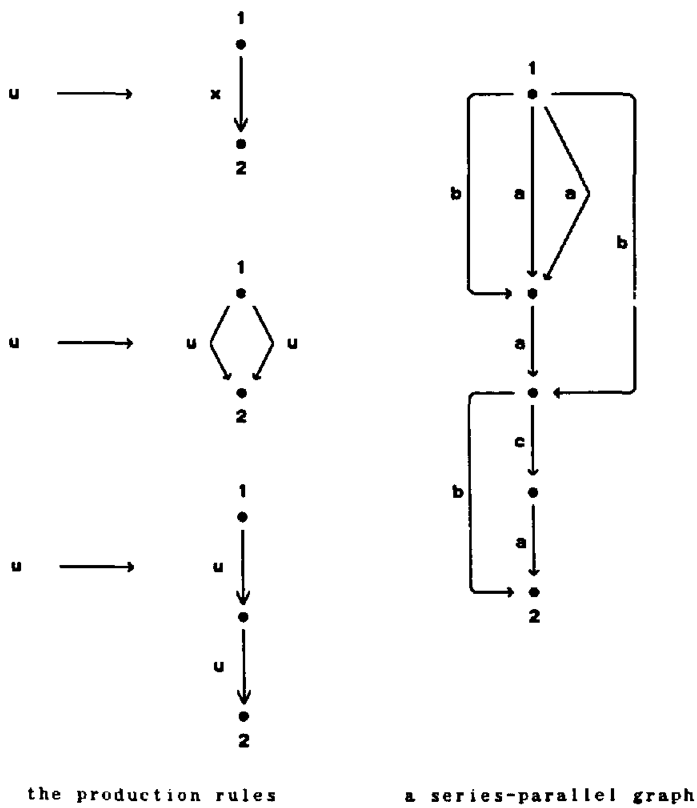


Fig. 1.

express that p names the production rule $u \to D$. We let $e_1, \ldots, e_k$ be an enumeration of the set of nonterminal edges of D. We consider $(D, e_1, \ldots, e_k)$ as a graph operation p. This defines a signature of graph operations associated with Γ. We let $\mathbf{FG}_\Gamma$ be the associated P-magma, and $\mathbf{h}_\Gamma$ be the unique homomorphism $\mathbf{M}(P) \to \mathbf{FG}_\Gamma$.

Let p be a production rule of the above form. We denote by $\hat{p}$ be the term $p(u_{i_1}, \ldots, u_{i_k})$ where u_{i_j} is the nonterminal labelling e_j for $j = 1, \ldots, k$. We let $\mathbf{S}_\Gamma$ be the system $\langle u_1 = t_1, \ldots, u_n = t_n \rangle$ where t_i is the sum of terms $\hat{p}$ such that p has left-hand side u_i. The least solution of $\mathbf{S}_\Gamma$ in the powerset magma of $\mathbf{M}(P)$ is an n-tuple of equational sets of terms (or trees; see [13]), $\langle T_1, \ldots, T_n \rangle$, where $T_i \subseteq \mathbf{M}(P)_{n_i}$. The set T_i is the set of *derivation trees*, representing the derivation sequences of Γ starting at u_i. We denote it by $\mathbf{T}(\Gamma, u_i)$. With these notations, we have the following theorem.

Theorem 1.8 (Bauderon and Courcelle [2]). (1) $\mathbf{h}_\Gamma(\mathbf{T}(\Gamma, u_i)) = \mathbf{L}(\Gamma, u_i)$, *for all* $i = 1, \ldots, n$.

(2) $\langle \mathbf{L}(\Gamma, u_1), \ldots, \mathbf{L}(\Gamma, u_n) \rangle$ *is the least solution of* $\mathbf{S}_\Gamma$ *in* $\mathcal{P}(\mathbf{FG}_\Gamma)$.

We denote by $\mathbf{T}(\Gamma)$ the set $\mathbf{T}(\Gamma, u)$, where u is the initial nonterminal of Γ, and we call it the set of *derivation trees* of Γ.

Example 1.9 (*continuation of Example* 1.6). We denote by // (read *parallel-composition*) and by • (read *series-composition*) the two binary operations on 2-graphs corresponding to the production rules of Γ of the second and third type. The system $\mathbf{S}_\Gamma$ is thus reduced to the unique equation

$$u = a + b + c + u \mathbin{/\!/} u + u \bullet u$$

(where + denotes the union of sets of graphs). A derivation tree t of the graph of $\mathbf{L}(\Gamma)$ shown in Fig. 1 is shown in Fig. 2. (Let us note that this grammar is *ambiguous*,

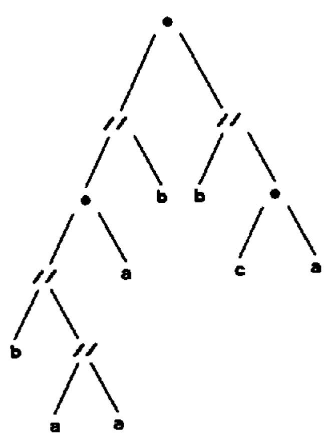


Fig. 2.

in the sense that a graph G in **SP** may be the image under $\mathbf{h}_{\Gamma'}$ of several distinct derivation trees.)

Corollary 1.10. *A set of graphs L is context-free iff it has a presentation (π, K) for some equational subset K of* $\mathbf{M}(P)$.

Proof. (*Only if*): Immediate consequence of Theorem 1.8(1).

(*If*): The image of an equational set is equational [18], and every equational set of graphs is context-free (this follows from Theorem 1.8(2). □

The construction of Definition 1.7 shows how to transform a context-free graph-grammar into a presentation of the set it generates. Conversely, from a presentation (π, K) of a set of graphs L, such that K is equational, one can construct a context-free graph-grammar as follows. We let K be given as the first component of the least solution in $\mathcal{P}(\mathbf{M}(P))$ of a system of equations $\langle u_1 = t_1, \ldots, u_n = t_n\rangle$ where each right-hand side t_i is a sum $m_1 + \cdots + m_k$ where each m_i belongs to $\mathbf{M}(P \cup U)$ and U is the set of unknowns of the system. For each of these terms m, one defines a production rule $u_i \to D$, where D is the graph in $\mathbf{FG}(A \cup U)$ defined by m. One obtains in this way a context-free graph grammar Γ with set of nonterminals U, and $L = \mathbf{h}_\pi(K) = \mathbf{L}(\Gamma, u_1)$. See [2] for the proofs.

Theorem 1.11 (Courcelle [6, 8]). *For every context-free graph-grammar Γ, one can compute an integer k such that* $\mathbf{twd}(\mathbf{L}(\Gamma)) \leq k$.

(2) *For every n and k, the set $\{G \in \mathbf{FG}(A)_n \mid \mathbf{twd}(G) \leq k\}$ is context-free. A grammar can be constructed to generate it.*

We now recall from [5–7] the fundamental notion of a *recognizable set of graphs*. If one considers the replacement of a graph for an edge in a graph as the generalization of the replacement of a word for a letter in a word, the notion of a recognizable set of graphs defined below extends that of a recognizable language.

Definition 1.12 (*Recognizable sets of graphs*). A *congruence* is an equivalence relation $\simeq$ on $\mathbf{FG}(A)$ such that, any two equivalent graphs are of the same type, and, for every graph K in $\mathbf{FG}(A)$, for every edge e of K, for every graph G of type $\tau(e)$ and every $G' \simeq G$, one has $K[G/e] \simeq K[G'/e]$. Such a congruence is *locally-finite* if it has finitely many classes of each type. A subset L of $\mathbf{FG}(A)_n$ is *recognizable* if there exists a locally finite congruence $\simeq$ such that, if $G \simeq G'$, then $G \in L$ iff $G' \in L$. We denote by $\mathbf{Rec}(\mathbf{FG}(A))_n$ the set of such subsets.

Theorem 1.13 (Courcelle [5–7]). *The intersection of a context-free and a recognizable set of graphs is context-free.*

A diagram comparing the various classes of sets of graphs we have defined in this section, together with others is given at the end of Section 2 (Fig. 3).

2. Monadic second-order logic

The use of monadic second-order logic for expressing graph properties is the subject of the series to which the present paper belongs. See in particular [5–9]. We review or introduce the following notions: relational structures, monadic second-order logic, definition in monadic second-order logic of a structure in another one, definable transductions of structures, quotient structures, structures defining graphs, definable sets of graphs, and we conclude with a diagram comparing various classes of sets of graphs.

Definition 2.1. Let R be a finite ranked set of symbols such that each element r in R has a rank $\rho(r)$ in $\mathbb{N}_+$. A symbol r in R is considered as a $\rho(r)$-ary relation symbol.

An R-*(relational) structure* is a tuple $S = \langle \mathbf{D}_S, (r_S)_{r \in R}\rangle$ where $\mathbf{D}_S$ is a possibly empty set, called the domain of S, and r_S is a subset of $\mathbf{D}_S^{\rho(r)}$ for each r in R. We denote by $\mathcal{S}(R)$ the class of finite R-structures (all structures will be finite in this paper).

We denote by $\mathcal{L}(R, W)$ the set of formulas of *counting monadic second-order logic* written with the symbols of R, and with free variables in W, where W is a set of variables $X, Y, X_1, X_2, Z, Z', \ldots$. These variables will denote subsets of $\mathbf{D}_S$, where S belongs to $\mathcal{S}(R)$.

The atomic formulas are: $X \subseteq Y$, $r(X_1, \ldots, X_n)$ where $n = \rho(r)$, and $\mathbf{card}_{p,q}(X)$ where $0 \leqslant p < q$, $q \geqslant 2$.

If $X, Y, X_1, \ldots, X_n$ denote subsets $\bar{X}, \bar{Y}, \bar{X}_1, \ldots, \bar{X}_n$ of $\mathbf{D}_S$, $S \in \mathcal{S}(R)$, then these formulas are true iff, respectively, $\bar{X} \subseteq \bar{Y}$, $r_S(x_1, \ldots, x_n)$ holds where x_i is some element of $\bar{X}_i$ for every $i = 1, \ldots, n$, and $\mathbf{card}(\bar{X}) = p + mq$ for some $m \in \mathbb{N}$.

The formulas of $\mathcal{L}(R, W)$ are formed with the Boolean connectives $\neg$ and $\vee$, and existential quantifications.

Let S be an R-structure, let $\varphi \in \mathcal{L}(R, W)$, and γ be a *W-assignment in S*, i.e., $\gamma(X)$ is a subset of $\mathbf{D}_S$ for every variable X in W (we write this $\gamma: W \to S$). We write $(S, \gamma) \models \varphi$ iff φ holds in S for γ. We write $S \models \varphi$ in the case where φ has no free variable.

A set of R-structures L is *definable* if it is the set of R-structures where some formula φ in $\mathcal{L}(R)$ holds.

In order to make formulas readable, we shall also write them with $\wedge, \Rightarrow$ and $\forall X$, and we shall use the following abbreviations:

$X = Y$	for $X \subseteq Y \wedge Y \subseteq X$
$X = \emptyset$	for $\forall Y [Y \subseteq X \Rightarrow X \subseteq Y]$
$\mathbf{sgl}(X)$	for $\forall Y [Y \subseteq X \Rightarrow Y = \emptyset \vee Y = X]$
	(to mean that X is singleton)
$\exists! X$	for there exists one and only one X.

We shall also use lowercase variables $x, y, x_1, \ldots, x_n$ to denote singletons, i.e., elements of $\mathbf{D}_S$, $S \in \mathcal{S}(R)$. This means that

- $\exists x. \varphi$ stands for $\exists x\,[\mathbf{sgl}(x) \wedge \varphi]$,
- $\forall x. \varphi$ stands for $\forall x\,[\mathbf{sgl}(x) \Rightarrow \varphi]$,
- $x \in Y$ stands for $x \subseteq Y$.

For an assignment $\gamma: W \rightarrow S$, we shall assume that $\gamma(x)$ is singleton for every lowercase variable x in W. We shall write $\gamma(x) = d$ instead of $\gamma(x) = \{d\}$.

Formal constructions and proofs will be given in terms of the restricted syntax defined at the beginning.

Definition 2.2 (*Relative definability of structures*). Let R and R' be two ranked sets of relation symbols. Let W be a finite set of uppercase variables, called here the set of *parameters*. (It is not a loss of generality to assume that the parameters are set variables; this is just convenient for some proofs.)

An (R, R')-*definition scheme* is a tuple of formulas of the form $\Delta = (\varphi, \psi_1, \ldots, \psi_k, (\theta_s)_{s \in R*k})$ where $R * k$ is the set of pairs (r, j), where r belongs to R and j is a sequence of $\rho(r)$ integers in $[k]$,

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

$$\psi_i \in \mathcal{L}(R', \{x_1\} \cup W), \qquad \text{for every } i = 1, \ldots, k,$$

$$\theta_s \in \mathcal{L}(R', \{x_1, \ldots, x_{\rho(r)}\} \cup W), \quad \text{for every } s \text{ in } R * k \text{ of the form } (r, j).$$

Let $T \in \mathcal{S}(R')$, let γ be a W-assignment in T. A structure S with domain $\subseteq \mathbf{D}_T \times [k]$ is *defined by* Δ *in* (T, γ) (this is denoted by $S = \mathbf{def}_\Delta(T, \gamma)$) if

$$(T, \gamma) \models \varphi,$$

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

(this set may be empty, and S is still well-defined)

$$r_S = \{((d_1, i_1), \ldots, (d_s, i_s)) \mid (T, \gamma, d_1, \ldots, d_s) \models \theta_{(r,j)}\},$$

where $j = (i_1, \ldots, i_s)$ and $s = \rho(r)$. (By $(T, \gamma, d_1, \ldots, d_s) \models \theta_{(r,j)}$, we mean $(T, \gamma') \models \theta_{(r,j)}$, where γ' is the assignment extending γ, such that $\gamma'(x_i) = d_i$ for all $i = 1, \ldots, s$; analogous notations will be used in the sequel).

Note that S is defined in a unique way from T, γ, and Δ.

In the special case where $k = 1$, we can replace $\mathbf{D}_T \times \{1\}$ by $\mathbf{D}_T$. Hence, $\mathbf{D}_S \subseteq \mathbf{D}_T$, and the tuple Δ can be written more simply $(\varphi, \psi, (\theta_r)_{r \in R})$.

We denote by $\mathbf{def}_\Delta(T)$ the set of structures of the form $\mathbf{def}_\Delta(T, \gamma)$ for some assignment γ. If $W = \emptyset$, then $\mathbf{def}_\Delta(T)$ is either empty or singleton. We write $S = \mathbf{def}_\Delta(T)$ iff it is the singleton reduced to S.

A relation $f \subseteq \mathcal{S}(R') \times \mathcal{S}(R)$ is called a *transduction* $\mathcal{S}(R') \rightarrow \mathcal{S}(R)$. We consider it as a total mapping $\mathcal{S}(R') \rightarrow \mathcal{P}(\mathcal{S}(R))$. Hence, we write $(T, S) \in f$ as well as $S \in f(T)$. The domain $\mathbf{Dom}(f)$ is the set of structures T such that $f(T)$ contains at least one structure. If f is functional, i.e., if $f(T)$ is empty or singleton for all T, we write $S = f(T)$ instead of $S \in f(T)$.

A transduction f is *definable* if it is equal to $\mathbf{def}_\Delta$ for some definition scheme Δ of appropriate type. We say that f is *essentially definable* if there exists a definable transduction f' such that $f' \subseteq f$ and $\mathbf{Dom}(f') = \mathbf{Dom}(f)$.

It is clear that the domain of a definable, or of an essentially definable transduction is definable. It is the set of R'-structures T such that $(T, \gamma) \models \varphi$ for some assignment γ, i.e., such that $T \models \exists W_1, \ldots, W_k\ [\varphi]$, where $W_1, \ldots, W_k$ are the parameters of Δ.

Let $\mathcal{S} \subseteq \mathcal{S}(R)$ and $\mathcal{S}' \subseteq \mathcal{S}(R')$. We say that $\mathcal{S}$ is *definable in* $\mathcal{S}'$ if there exists an (R, R')-definition scheme Δ such that $\mathcal{S} = \mathbf{def}_\Delta(\mathcal{S}') = \{\mathbf{def}_\Delta(T, \gamma) \mid T \in \mathcal{S}', \gamma : W \to T\}$. (Let us make precise that two isomorphic structures are considered as equal.)

Quotient structures form an important example.

Definition 2.3 (*Quotient structures*). Let S be an R-structure, and E be an equivalence relation on $\mathbf{D}_S$. We denote by S/E the R-structure defined as follows:

$$\mathbf{D}_{S/E} := \mathbf{D}_S / E$$

$r_{S/E}([d_1], \ldots, [d_k])$ is true iff
$r_S(d'_1, \ldots, d'_k)$ is true for some d'_1 in $[d_1], \ldots, d'_k$ in $[d_k]$ (we denote by $[d]$ the equivalence class of d with respect to the equivalence relation E).

Let us now assume that E is *generated* by the binary relation on $\mathbf{D}_S$ defined by a formula η with free variables x, y, (i.e., that E is the least equivalence relation containing this binary relation). Since the transitive closure of a binary relation that is definable in monadic second-order logic is also definable in monadic second-order logic [7, Lemma 3.7], the relation E is definable by a formula $\bar{\eta}$ with free variables x and y.

Our purpose is to construct Δ such that S/E is defined in S by Δ. Our construction uses a parameter X. We let Δ be the tuple $(\varphi, \psi, (\theta_r)_{r \in R})$ such that:

(i) φ is the formula

$$\forall x\ \exists! y\ [\bar{\eta}(x, y) \wedge y \in X],$$

saying that $\gamma(X)$ meets each equivalence class of E once and only once.

(ii) ψ is the formula $x_1 \in X$ saying that the domain of $\mathbf{def}_\Delta(S, \gamma)$ is $\gamma(X)$.

(iii) θ_r is the formula

$$\exists y_1, \ldots, y_n\ [\bar{\eta}(x_1, y_1) \wedge \cdots \wedge \bar{\eta}(x_n, y_n) \wedge r(y_1, \ldots, y_n)]$$

(where $n = \rho(r)$).

With these notations and definitions, we have the following lemma.

Lemma 2.4. *The transduction of R-structures $S \mapsto S/E$, where E is the equivalence relation on $\mathbf{D}_S$ generated by the binary relation defined by a monadic second-order formula η, is definable.*

Proof. This result follows immediately from the definitions and the construction of Δ. $\square$

The following proposition says that if $S = \mathbf{def}_\Delta(T, \gamma)$ then the monadic second-order properties of S can be expressed as monadic second-order properties of (T, γ).

Let $\Delta = (\varphi, \psi_1, \ldots, \psi_k, (\theta_s)_{s \in R*k})$ be written with a set of parameters $\mathcal{W}$. Let $\mathcal{V}$ be a set of uppercase variables disjoint from $\mathcal{W}$.

For every variable X in $\mathcal{V}$, for every $i = 1, \ldots, k$, we let X_i be a new variable. We let $\bar{\mathcal{V}} = \{X_i \mid X \in \mathcal{V}, i = 1, \ldots, k\}$. For every $\eta : \bar{\mathcal{V}} \to \mathcal{P}(D)$, we let $\gamma : \mathcal{V} \to \mathcal{P}(D \times [k])$ be defined by

$$\gamma(X) = \eta(X_1) \times \{1\} \cup \cdots \cup \eta(X_k) \times \{k\}.$$

With this notation we have the following.

Proposition 2.5. (1) *For every formula β in $\mathcal{L}(R, \mathcal{V})$, one can construct a formula $\bar{\beta}$ in $\mathcal{L}(R', \bar{\mathcal{V}} \cup \mathcal{W})$ such that, for every T in $\mathcal{S}(R')$, for every $\mu : \mathcal{W} \to T$, for every $\eta : \bar{\mathcal{V}} \to T$, we have: $\mathbf{def}_\Delta(T, \mu)$ is defined (if it is, we denote it by S), γ is a $\mathcal{V}$-assignment in S, and $(S, \gamma) \models \beta$ iff $(T, \eta \cup \mu) \models \bar{\beta}$.*

(2) *If $\mathcal{S}' \subseteq \mathcal{S}(R')$ has a decidable monadic theory, and if $\mathcal{S}$ is definable in $\mathcal{S}'$, then the monadic theory of $\mathcal{S}$ is also decidable.*

Proof. (1) We take $\bar{\beta}$ equal to

$$\hat{\beta} \wedge \varphi \wedge \bigwedge\{\forall x.\, [x \in X_i \Rightarrow \psi_i(x)] \mid 1 \leq i \leq k\},$$

where $\hat{\beta}$ is constructed by induction on the structure of β as follows:

- if β is $X \subseteq X'$, then $\hat{\beta}$ is $X_1 \subseteq X'_1 \wedge \cdots \wedge X_k \subseteq X'_k$,
- if β is $r(X^1, \ldots, X^n)$, then $\hat{\beta}$ is

$$\exists y_1, \ldots, y_n \quad [\bigvee \{\theta_{(r,j)}(y_1, \ldots, y_n) \wedge y_1 \in X^1_{j(1)} \wedge \cdots \wedge y_n \in X^n_{j(n)} \mid j \in [k]^n\}]$$

(where we denote by $j(i)$ the ith element of the sequence j),

- if β is $\mathbf{card}_{p,q}(X)$, then $\hat{\beta}$ is

$$\bigvee \{\mathbf{card}_{p_1,q}(X_1) \wedge \cdots \wedge \mathbf{card}_{p_k,q}(X_k) \mid 0 \leq p_1, \ldots, p_k < q, p_1 + \cdots + p_k = p \mathbf{\ mod\ } q\},$$

- if β is $\neg\beta_1$, then $\hat{\beta}$ is $\neg\hat{\beta}_1$,
- if β is $\beta_1 \vee \beta_2$, then $\hat{\beta}$ is $\hat{\beta}_1 \vee \hat{\beta}_2$,
- if β is $\exists X.\, \beta_1$, then $\hat{\beta}$ is $\exists X_1, \ldots, X_k.\, \hat{\beta}_1$ (we assume that all variables of β are in $\mathcal{V}$).

The verification that $\bar{\beta}$ satisfies the desired properties is easy by induction on the structure of β.

(2) Immediate consequence of (1). $\square$

Part (2) of Proposition 2.5 states that the monadic theory of $\mathcal{S}$ is *interpretable* in that of $\mathcal{S}'$. Interpretability of theories is a strong form of reducibility. See [21] on interpretations of monadic theories of graphs.

The following proposition is an easy consequence of the previous one.

Corollary 2.6. *Let $\mathcal{S} \subseteq \mathcal{S}(R)$, $\mathcal{S}' \subseteq \mathcal{S}(R')$, and $\mathcal{S}'' \subseteq \mathcal{S}(R'')$. If $\mathcal{S}$ is definable in $\mathcal{S}'$, and $\mathcal{S}'$ is definable in $\mathcal{S}''$, then $\mathcal{S}$ is definable in $\mathcal{S}''$. The composition of two definable transductions is a definable transduction.*

Let $f : \mathcal{S}(R) \to \mathcal{S}(R')$ be a transduction. Let L be a set of R-structures, and K be a set of R'-structures. The *domain-restriction* of f by L is the transduction $f \cap (L \times \mathcal{S}(R'))$ and the *codomain-restriction* of f by K is the transduction $f \cap (\mathcal{S}(R) \times K)$.

Corollary 2.7. *If f is definable (resp. essentially definable) and if L is definable, then the domain-restriction of f by L is definable (resp. essentially definable). If f is definable, and if K is definable, then the codomain-restriction of f by K is definable.*

Definition 2.8 (*Relational structures representing graphs*). Let A be a finite set of edge labels as in Definition 1.1; let $n \in \mathbb{N}$. Let $\mathbf{R}(A, n)$ be the following set of relation symbols:

- $\mathbf{v}$ of arity 1,
- $\mathbf{edg}_a$ of arity $\tau(a)+1$, for each a in A,
- $\mathbf{ps}_i$ of arity 1, for each $i = 1, \ldots, n$.

For every n-graph G over A, we let $|G|$ be the $\mathbf{R}(A, n)$-structure such that:

$$\mathbf{D}_{|G|} := \mathbf{V}_G \cup \mathbf{E}_G \quad \text{(we assume that } \mathbf{V}_G \cap \mathbf{E}_G = \emptyset\text{)},$$

$$\mathbf{v}_{|G|}(x) = \mathbf{true} \text{ iff } x \in \mathbf{V}_G,$$

$$\mathbf{edg}_{a|G|}(x, y_1, \ldots, y_k) = \mathbf{true} \text{ iff } x \in \mathbf{E}_G,\ \mathbf{lab}_G(x) = a, \text{ and } \mathbf{vert}_G(x) = (y_1, \ldots, y_k),$$

$$\mathbf{ps}_{i|G|}(x) = \mathbf{true} \text{ iff } x = \mathbf{src}_G(i).$$

It is clear that $|G|$ *represents* G, i.e., that, for any two graphs G and G', $|G|$ is isomorphic to $|G'|$ iff G is isomorphic to G'.

A subset L of $\mathbf{FG}(A)_n$ is *definable* iff there exists a formula φ in $\mathcal{L}(\mathbf{R}(A, n))$ such that $G \in L$ iff $|G| \models \varphi$, i.e., iff the set of structures representing it is definable. The notions of a *graph transduction*, of a *definable graph transduction*, and of an *essentially definable graph transduction* follow in a similar way from the corresponding notions concerning structures.

In [5-9] a slightly different version of counting monadic second-order logic is used: the structures representing graphs have two domains (the set of edges, and the set of vertices), and the formulas are written with variables of two possible sorts (the variables of sort "vertex" denote vertices or sets of vertices, and those of sort "edge" denote edges or sets of edges). It is not hard to prove that the two logical languages yield the same definable sets of graphs. The proof is similar to that of Proposition 2.5.

We now recall a basic theorem from [7].

Theorem 2.9. *Every definable set, all graphs of which are of the same type, is recognizable.*

To conclude this section, we present a diagram, comparing the various families of sets of graphs we have discussed. (On this diagram (Fig. 3), the scope of a family name is the largest rectangle, at the upper left corner of which it is written.)

The following families of sets of graphs are compared in Fig. 3:

- **REC**, the family of recognizable set of graphs;
- **DEF**, the family of definable sets of graphs;
- **CF**, the family of context-free sets of graphs;
- **B**, the family of sets of graphs of finite tree-width;
- **SCF**, the family of strongly context-free sets of graphs that we shall introduce in Section 4.

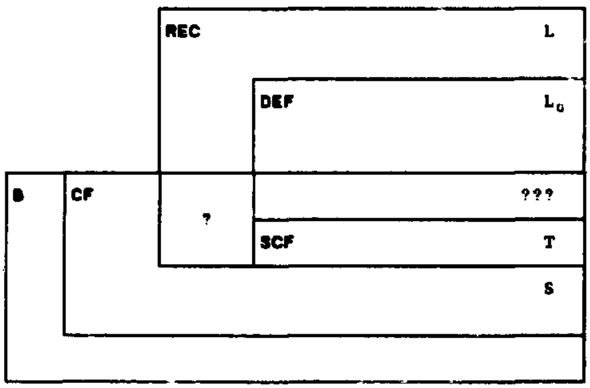


Fig. 3.

Provided the reference alphabet contains at least one symbol of type at least 2, the families **REC** and **B** are uncountable. The other ones are countable. The inclusions shown on the diagram, are strict, except possibly the two inclusions

$$\mathbf{SCF} \subseteq \mathbf{CF} \cap \mathbf{DEF}, \tag{1}$$

$$\mathbf{CF} \cap \mathbf{DEF} \subseteq \mathbf{CF} \cap \mathbf{REC}. \tag{2}$$

We make the following conjecture, saying that the equality holds in (2), i.e., that the box with ? in Fig. 3 is empty.

Conjecture 2.10. *If a set of graphs is recognizable and has a finite tree-width, then it is definable.*

Since every context-free set has a finite tree-width, and since every recognizable set of graphs of finite tree-width is context-free (by Theorems 1.11 and 1.13), one can replace "has a finite tree-width" by "is context-free", and one gets an equivalent conjecture. We shall establish it for sets of graphs of tree-width at most 2 (see Section 6).

We shall also make the related conjecture that the box with ??? is empty. See Conjectures 4.12 for a discussion of these conjectures.

The diagram also locates several sets of graphs:

- L_G, the set of square grids;
- L, the set of all $n \times n$ square grids, where n is an element of some nonrecursive subset of $\mathbb{N}$;
- S, the set of graphs corresponding to the language $\{a^n b^n \mid n > 0\}$;
- T, the set of binary graphs representing unranked unordered trees.

See [7] for the proofs.

3. Reduced trees

Finite ranked ordered trees represent terms in a well known way. If some binary operation is known to be associative and commutative, the corresponding terms can be represented by *reduced trees*, some nodes of which have a *set* and not a *sequence* of successors. This idea has been introduced by Franchi-Zannettacci in the context of attribute grammars [12]: derivation trees are reduced in this way, and this improves the efficiency of the evaluation of attributes. This reduction will be applied to the syntactic trees representing graphs as defined in Section 1.

In this section, we give definitions making it possible to deal rigorously with reduced trees. We represent these trees as graphs, we state that a set of reduced trees is definable in monadic second-order logic iff it is recognizable (extending the corresponding theorem of Doner for ranked ordered trees recalled in the introduction), and we give sufficient conditions for the recognizability of an equational set of reduced trees.

These technical results will be used in Sections 4–6.

Definition 3.1. Let P be a one-sort signature containing one binary (infixed) symbol $/\!/$ and a constant $\mathbf{e}$. By a *P-ac-magma*, we mean a P-magma M in which the operation $/\!/_M$ is associative, commutative, and has unit $\mathbf{e}_M$.

The quotient P-magma $\mathbf{RM}(P) := \mathbf{M}(P)/\overset{*}{\leftrightarrow}_R$ where R is the set of equational axioms $\{x /\!/ y = y /\!/ x, x /\!/ (y /\!/ z) = (x /\!/ y) /\!/ z, x /\!/ \mathbf{e} = x\}$ is the initial P-ac-magma. Its elements can be represented by trees, the nodes labelled by $/\!/$ of which have an unbounded, unordered set of successors, or by graphs as defined below.

Definition 3.2. Let $B := P - \{/\!/, \mathbf{e}\}$, where P is as above, with rank function $\rho : P \to \mathbb{N}$. We make B into a set of edge labels with type function $\tau : B \to \mathbb{N}_+$, defined by $\tau(b) := \rho(b) + 1$.

We make $\mathbf{FG}(B)_1$ into a P-ac-magma M by letting

$$\mathbf{e}_M := \mathbf{1},$$

$$G \mathbin{/\!/}_M H := A[G/e, H/f],$$

$$b_M(G_1, \ldots, G_k) = B_k[G_1/e_1, \ldots, G_k/e_k],$$

where the graphs A and B_k are shown in Fig. 4. The operations $/\!/_M$ and b_M can be described informally as follows:

(a) $L = G \mathbin{/\!/}_M H$ is obtained by fusing the sources of G and H into a single vertex becoming the source of L (one assumes that G and H are disjoint).

(b) $K = b_M(G_1, \ldots, G_k)$ is obtained by taking a disjoint union of $G_1, \ldots, G_k$, by adding a new edge labelled by b with sequence of vertices

$$(\mathbf{src}_{G_1}(1), \ldots, \mathbf{src}_{G_k}(1), v),$$

where v is a new vertex, becoming the source of K.

These operations are illustrated in Fig. 4 for $k = 3$. It is clear that $/\!/_M$ is associative, commutative, and has unit $\mathbf{e}_M$.

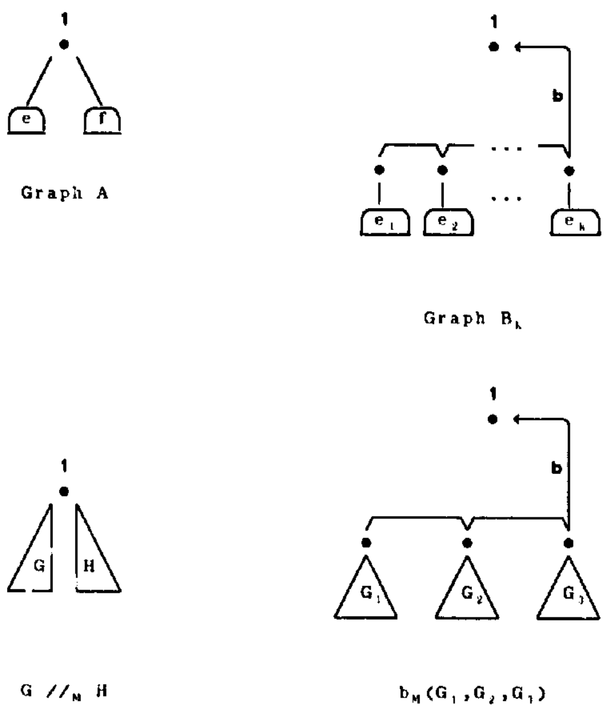


Fig. 4.

Hence, there is a unique homomorphism $\mathbf{k}:\mathbf{RM}(P)\to\mathbf{FG}(B)_1$. It is not hard to establish that $\mathbf{k}$ is one-to-one. We denote by $\mathbf{RT}(B)$ the subset $\mathbf{k}(\mathbf{RM}(P))$ of $\mathbf{FG}(B)_1$. Hence, $\mathbf{k}$ defines a bijection of $\mathbf{RM}(P)$ onto the set of graphs $\mathbf{RT}(B)$.

We introduce some terminology concerning this bijection. Let t in $\mathbf{RM}(P)$ correspond to G in $\mathbf{RT}(B)$. A node w of t having a label in B corresponds to an edge e of G having the same label. This node is the root r of t, or is a successor of r where r is labelled by $/\!/$, iff the last vertex of e is the (unique) source of G. We say in this case that e is a *0-edge* of G. Otherwise, w is the ith successor of some node w' that has a label in B, or is separated from such a node by a sequence of nodes labelled by $/\!/$. Let e' correspond to w'. We say that e is an *i-edge* of G, and that e is an *i-successor* of e' where i is such that the last vertex of e is the ith one of e'.

In Fig. 5, we show an element t of $\mathbf{M}(P)$, its value $\tilde{t}$ in $\mathbf{RM}(P)$, and the graph $\mathbf{k}(\tilde{t})$. The edges of $\mathbf{k}(\tilde{t})$ labelled by $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{f}$ are 0-edges. The one labelled by $\mathbf{c}$ is a 2-edge, and is the unique 2-successor of the one labelled by $\mathbf{f}$. The two edges labelled by $\mathbf{d}$ are 3-edges and are the 3-successors of the one labelled by $\mathbf{f}$.

It is not hard to establish that $\mathbf{RT}(B)$ is definable (as a subset of $\mathbf{FG}(B)_1$).

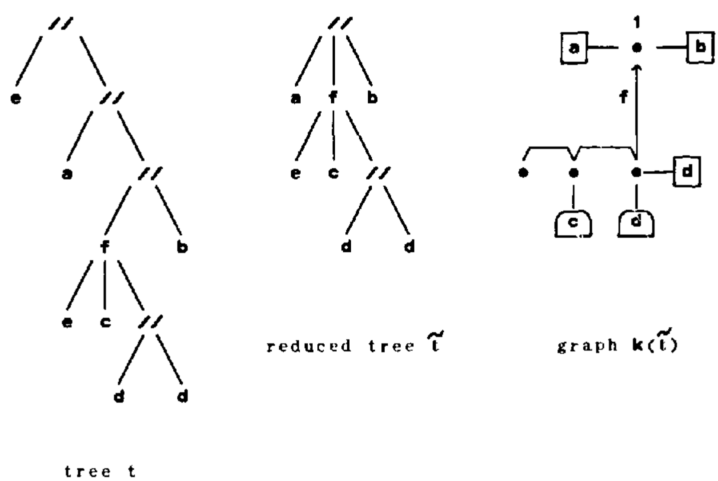


Fig. 5.

Theorem 3.3. *Let $L\subseteq\mathbf{RM}(P)$. The following conditions are equivalent:*

(1) *L is $\mathbf{RM}(P)$-recognizable,*

(2) *$\mathbf{k}(L)$ is a recognizable subset of $\mathbf{FG}(B)_1$,*

(3) *$\mathbf{k}(L)$ is a definable subset of $\mathbf{FG}(B)_1$,*

(4) *$\mathbf{k}(L)$ is a definable subset of $\mathbf{RT}(B)$, i.e.,*

$$\mathbf{k}(L)=\{G\in\mathbf{RT}(B)\mid G\models\varphi\}\ \textit{for some}\ \varphi\ \textit{in}\ \mathscr{L}(\mathbf{R}(B,1)).$$

Proof. (3)⇒(4) is trivial. (4)⇒(3) since $\mathbf{RT}(B)$ is definable in $\mathbf{FG}(B)_1$. (3)⇒(2) by Theorem (2.8).

(2)⇒(1): If $\mathbf{k}(L)$ is recognizable with respect to $\mathbf{FG}(B)$, then it is recognizable with respect to M, because M is a derived magma of $\mathbf{FG}(B)$ (see ([7]). We obtain that L is $\mathbf{RM}(P)$-recognizable because recognizability is preserved under inverse homomorphisms.

(1)⇒(3): Theorem 5.3 of [7] establishes this result in the special case where all symbols of B are of rank 0 or 1. The extension to the present case is straightforward. □

It is essential that definability be understood with respect to *counting* monadic second-order logic. Otherwise, i.e., without the atomic formulas of the form $\mathbf{card}_{p,q}(X)$, the implication (1)⇒(3) does not hold. This has been proved in [7, Corollary 6.6].

Definition 3.4 (*Equational sets of reduced trees*). Let P be a ranked set of symbols as in Definition 3.1, with two special symbols $/\!/$ and $\mathbf{e}$. Let M be a P-magma, let S be a polynomial system $\langle u_1 = t_1, \ldots, u_n = t_n \rangle$. We denote by $\mathbf{L}((S, M), u_i)$ the ith component of the least solution of S in $\mathcal{P}(M)$. If $h: M \to M'$ is a homomorphism into a P-magma M', then

$$\mathbf{L}((S, M'), u_i) = h(\mathbf{L}(S, M), u_i) := \{h(m) \mid m \in \mathbf{L}((S, M), u_i)\}$$

by a lemma in [18, Lemma 5.3] (see also [3, Proposition 13.11]).

Let M be a P-ac-magma. Then, the unique homomorphism $\mathbf{h}_M : \mathbf{M}(P) \to M$ factors uniquely into

$$\mathbf{h}_M : \mathbf{M}(P) \xrightarrow{\mathbf{h}_{ac}} \mathbf{RM}(P) \xrightarrow{\mathbf{rh}_M} M$$

where $\mathbf{h}_{ac}$ is the unique homomorphism $\mathbf{M}(P) \to \mathbf{RM}(P)$, $\mathbf{rh}_M$ is the unique homomorphism $\mathbf{RM}(P) \to M$. It follows that

$$\mathbf{L}((S, M), u_i) = \mathbf{rh}_\pi(\mathbf{L}((S, \mathbf{RM}(P)), u_i))$$

and that

$$\mathbf{L}((S, \mathbf{RM}(P)), u_i) = \mathbf{h}_{ac}(\mathbf{L}((S, \mathbf{M}(P)), u_i))$$

for all $i = 1, \ldots, n$.

The sets of ordered ranked trees $\mathbf{L}((S, \mathbf{M}(P)), u_i)$, abbreviated as $\mathbf{L}(S, u_i)$, are equational and recognizable with respect to $\mathbf{M}(P)$. Their images under $\mathbf{h}_{ac}$ are equational but not necessarily recognizable with respect to $\mathbf{RM}(P)$. (Let us recall that equational sets are preserved under homomorphisms, and that recognizable sets are not in general; they are preserved under inverse homomorphisms.)

We say that S is *ac-compatible* if the sets $\mathbf{L}((S, \mathbf{RM}(P)), u_i)$ are $\mathbf{RM}(P)$-recognizable.

We call $\mathbf{h}_{ac}(t)$ the *associative-commutative image of* t, for t in $\mathbf{M}(P)$. We let $\mathbf{h}_{ac}(L) := \{\mathbf{h}_{ac}(t) \mid t \in L\}$ be the *associative-commutative image* of L, for $L \subseteq \mathbf{M}(P)$.

Conjecture 3.5. *One can decide whether the associative-commutative image of a* $\mathbf{M}(P)$*-recognizable set is* $\mathbf{RM}(P)$*-recognizable.*

If this conjecture is correct, it follows that one can decide whether a system is ac-compatible.

It holds in the special case where $\rho(b)=0$ for each b in $B=P\text{-}\{/\!/, \mathbf{e}\}$, because $\mathbf{RM}(P)$ is then the free commutative monoid generated by B, with $/\!/$ as multiplication and $\mathbf{e}$ as unit. Deciding whether a recognizable subset of $\mathbf{M}(P)$ has a $\mathbf{RM}(P)$-recognizable associative-commutative image reduces to the problem of deciding whether a rational subset of the free commutative monoid is recognizable. This is decidable. An algorithm has been given in [15].

Example 3.6. Here is an example of a system (actually reduced to a single equation) that is not ac-compatible:

$$u=(a /\!/ b) /\!/ u+\mathbf{e}$$

The trees forming its least solution in $\mathcal{P}(\mathbf{M}(P))$ can be linearly written $(a /\!/ b) /\!/ (a /\!/ b) /\!/ (a /\!/ b) /\!/ \cdots /\!/ (a /\!/ b) /\!/ \mathbf{e}$ (with the convention that the operation $/\!/$ associates to the right).

Their images in $\mathbf{RM}(P)=\mathbb{N}^{\{a,b\}}$ are the commutative words with as many as and bs, and they do not form a recognizable set.

We now define easily testable syntactical conditions ensuring that a system is ac-compatible.

Definition 3.7. Let S be a system with set of unknowns U. We denote by AC the following condition on S:

(AC) There is a subset W of U such that every equation $u=p$ of S satisfies one of the following two conditions:

(AC1) $u\in W$ and the monomials forming p are of the form $b(u_1,\dots,u_k)$ for some $u_1,\dots,u_k$ in U, some b in B of rank $k\geq 0$,

(AC2) $u\in U-W$ and p is of the form:

$$w_1 /\!/ u+\cdots+w_n /\!/ u+w'_1 /\!/ w'_2 /\!/ \cdots /\!/ w'_k$$

where $w_1,\dots,w_n, w'_1,\dots,w'_k$ are unknowns in W.

Let us note the following special cases of AC2: $n=0$ and p is $w'_1 /\!/ w'_2 /\!/ \cdots /\!/ w'_k$, $n=k=0$ and p is $\mathbf{e}$, and finally, $k=0$, $n\neq 0$, and p is $w_1 /\!/ u+\cdots+w_n /\!/ u+\mathbf{e}$.

There exists at most one set W for which AC holds. It can be equal to U, or empty.

Proposition 3.8. *A system which satisfies condition AC is ac-compatible.*

Before proving this proposition, we state a lemma. A term with leading (left-most) symbol in B is called a B-term. Every term t in $\mathbf{M}(P\cup U)$ can be written in a unique way as follows:

$$t=s[t_1/x_1,\dots,t_k/x_k]$$

where $t_1, \ldots, t_k$ are B-terms, s is written with $/\!/$, $\mathbf{e}$, the variables $x_1, \ldots, x_k$, such that they occur in this order, once and only once.

We call this writing the *B-decomposition of t*.

Lemma 3.9. (1) *Let t, t' in $\mathbf{M}(P \cup U)$ have the B-decompositions $s[t_1, \ldots, t_k]$ and $s'[t'_1, \ldots, t'_n]$. Then $t \overset{*}{\leftrightarrow}_R t'$ iff $k = n$ and, for some permutation σ of $[k]$, we have*

$$s \overset{*}{\leftrightarrow}_R s'[x_1/x_{\sigma(1)}, \ldots, x_n/x_{\sigma(n)}]$$

and $t_i \overset{*}{\leftrightarrow}_R t'_{\sigma(i)}$ for all i in $[k]$.

(2) *If t, t' are B-terms of the respective forms $b(t_1, \ldots, t_k)$ and $b'(t'_1, \ldots, t'_n)$, then $t \overset{*}{\leftrightarrow}_R t'$ iff $b = b'$ and $t_i \overset{*}{\leftrightarrow}_R t'_i$ for all i in $[n]$.*

Its proof can be done by inductions on the lengths of rewriting sequences in a standard way.

Proof of Proposition 3.8. For every subset L of $\mathbf{M}(P)$, we denote by $\bar{L}$ the set

$$\{t \in \mathbf{M}(P) \mid t \overset{*}{\leftrightarrow}_R t', t' \in L\}.$$

We say that L is *saturated* if $L = \bar{L}$. By Proposition 12 in [10], $\mathbf{h}_{ac}(L)$ is $\mathbf{RM}(P)$-recognizable iff $\bar{L}$ is $\mathbf{M}(P)$-recognizable.

Let S be as in Definition 3.7; we shall construct a system $\bar{S}$ such that $U \subseteq \mathbf{Unk}(\bar{S})$, and $L(\bar{S}, u) = \overline{L(S, u)}$ for every u in U. We let $\bar{e}$ be a new unknown. The system $\bar{S}$ has the unknowns of $U \cup \{\bar{e}\}$, together with other ones we shall introduce below. Its equations are of three types:

Type I: the unique equation defining $\bar{e}$

$$\bar{e} = \bar{e} /\!/ \mathbf{e} + \mathbf{e} /\!/ \bar{e} + \mathbf{e}$$

Type II: for every w in W, we let its defining equation in S be

$$w = \bar{e} /\!/ w + w /\!/ \bar{e} + p$$

where its defining equation in S is $w = p$ (of the form AC1).

Type III: we now associate several equations with an equation of the form

$$u = w_1 /\!/ u + \cdots + w_k /\!/ u + w'_1 /\!/ w'_2 /\!/ \cdots /\!/ w'_n$$

as in AC2. For every subset K of $[n]$, we define a new unknown (u, K), and we identify $(u, [n])$ with u.

We define (u, K) by the equation

$$\begin{aligned}(u, K) = {} & w_1 /\!/ (u, K) + (u, K) /\!/ w_1 + \cdots + w_k /\!/ (u, K) + (u, K) /\!/ w_k \\ & + (u, K) /\!/ \bar{e} + \bar{e} /\!/ (u, K) + p_K\end{aligned}$$

where p_K is defined as follows: if $K = \emptyset$, then p_K is $\bar{e}$; if $K = \{j\}$, then p_K is w'_j; if K has at least two elements, then p_K is

$$\sum \{(u, K') /\!/ (u, K'') \mid K' \text{ and } K'' \text{ form a partition of } K, \text{ with } K' \text{ and } K'' \text{ nonempty}\}.$$

It is interesting to consider some special cases. If $k=0$, then we have the equations

$$(u,\emptyset)=(u,\emptyset) \mathbin{/\!/} \bar{e}+\bar{e} \mathbin{/\!/} (u,\emptyset)+\bar{e}$$

that can be replaced by the simpler equation $(u,\emptyset)=\bar{e}$, and the equations

$$(u,K)=(u,K) \mathbin{/\!/} \bar{e}+\bar{e} \mathbin{/\!/} (u,K)+p_K$$

as above if K is nonempty. If $k=n=0$ then, we need only take $u=\bar{e}$. If $n=0$, $k\neq 0$, then, we need only take

$$u=w_1 \mathbin{/\!/} u+u \mathbin{/\!/} w_1+\cdots+w_k \mathbin{/\!/} u+u \mathbin{/\!/} w_k+\bar{e} \mathbin{/\!/} u+u \mathbin{/\!/} \bar{e}+\bar{e}.$$

Claim. $\mathbf{L}(\bar{S},u)=\overline{\mathbf{L}(S,u)}$ *for every* u *in* U.

Proof of claim. ($\subseteq$): For every u in U, every t in $\mathbf{L}(\bar{S},u)$, one can find t' in $\mathbf{L}(S,u)$ such that $t \leftrightarrow_R t'$ (we shall say that t and t' are *R-equivalent*). This can be proved by induction on n such that $u \to_{\bar{S}}^{n} t$ where $\bar{S}$ is considered as a regular tree grammar (see [13] or [4, Section 13]).

To prove the other direction, one first observes that $\mathbf{L}(S,u)\subseteq\mathbf{L}(\bar{S},u)$ by the way $\bar{S}$ is constructed. We now present the main steps of the proof that for every unknown u' of $\bar{S}$, the set $\mathbf{L}(\bar{S},u')$ is saturated.

Let t in $\mathbf{M}(P)$ be such that $u' \to_{\bar{S}}^{n} t$, and t' be R-equivalent to t. We shall prove by induction on $n+\mathbf{size}(t)$ that t' is derivable from u' in $\bar{S}$.

Let $s[t_1,\ldots,t_m]$ be the B-decomposition of t. We have the following cases.

Case 1: $u'=u$ and belongs to W. Then, $m=1$ and by Lemma 3.9, the B-decomposition of t' is necessarily of the form $s'[t_1']$ for some t_1' that is R-equivalent to t_1. Then t_1 is $b(r_1,\ldots,r_h)$, where each r_i is derivable from some unknown of $\bar{S}$, by a derivation sequence of length at most n, and t_1' is of the form $b(r_1',\ldots,r_h')$, with r_i' R-equivalent to r_i. By applying the induction to the terms r_i, one obtains the result. We omit the details.

Case 2: $u'=(u,K)$ with u in $U-W$. Let the defining equation of u in S be

$$u=w_1 \mathbin{/\!/} u+\cdots+w_n \mathbin{/\!/} u+w_{n+1} \mathbin{/\!/} \cdots \mathbin{/\!/} w_{n+k}.$$

Then each B-term t_i is derivable in $\bar{S}$ from some w_j with j in $[n+k]$. By Lemma 3.9, the B-decomposition of t' is of the form $s'[t'_{\sigma(1)},\ldots,t'_{\sigma(m)}]$ for some permutation σ of $[m]$ such that t_i' is R-equivalent to t_i. The term t' can be also written as $r \mathbin{/\!/} r'$ where the B-decomposition of r is of the form $c[t'_{\sigma(1)},\ldots,t'_{\sigma(p)}]$, and that of r' is of the form $c'[t'_{\sigma(p+1)},\ldots,t'_{\sigma(m)}]$.

We let K' be the set of integers i such that $n+i=\sigma(j)$ for some j in $[p]$, and K'' be $K-K'$. It is now easy to verify that r is R-equivalent to some r_1 derivable from (u,K'), and that r' is R-equivalent to some r_1' derivable from (u,K''). It follows by induction that r and r' are respectively derivable from (u,K') and (u,K''), from which we conclude that $t'=r \mathbin{/\!/} r'$ is derivable from (u,K).

Case 3: u is $\bar{e}$. This case is much simpler than the preceding ones and is left to the reader.

This concludes the proof of the claim and, consequently, of the proposition. □

Remark 3.10. If a system S does not satisfy condition AC, it may happen that it can be transformed into a system S' satisfying it, such that $\mathbf{Unk}(S) \subseteq \mathbf{Unk}(S')$ and $\mathbf{L}((S, \mathbf{RM}(P)), u) = \mathbf{L}((S', \mathbf{RM}(P)), u)$ for all u in $\mathbf{Unk}(S)$. Then the system S is ac-compatible. Such transformations of systems, that in a certain sense preserve their least solutions, have been examined in detail in [4].

Another example is the transformation that replaces in a system S every monomial t by an R-equivalent one, yielding a system S' with the same set of unknowns. The two systems S and S' are equivalent in $\mathcal{P}(\mathbf{RM}(P))$ (they have the same least solution), and S is ac-compatible if S' satisfies condition AC.

Remark 3.11. *The many-sorted case*: we now assume that P is a many-sorted signature, that $\mathcal{S}$ is its set of sorts, and that for every $s \in \mathcal{S}$ one has a binary symbol $/\!\!/_s$ of profile $s \times s \to s$, and a constant $\mathbf{e}_s$ of sort s. A many-sorted P-magma M is a *P-ac-magma* if the equational axioms expressing that $/\!\!/_s$ is associative and commutative with unit $\mathbf{e}_s$. All the definitions, results, and conjectures of this section extend easily to the many-sorted signature P. We omit the routine details.

4. Monadic second-order parsable graph-grammars

A set of graphs L is *strongly context-free* iff it has a presentation such that, in every graph G of L, a syntactic tree of G can be specified by a fixed definition scheme. Roughly speaking, this definition says that L is generated by a context-free graph-grammar Γ such that, in every graph, the associative-commutative image of a derivation tree of the graph G relative to the grammar Γ is defined by a fixed definition scheme.

In this section, we let $\pi = (P, \bar{\ })$ be a finite signature of graph operations, we let $\mathcal{S}$ be its set of sorts. The signature π may contain special symbols $/\!\!/_n$, $\mathbf{e}_n$ denoting fixed graph operations defined below, such that $/\!\!/_n$ is associative and commutative with unit $\mathbf{e}_n$. We let $B := P - \{/\!\!/_n, \mathbf{e}_n \mid n \geqslant 0\}$.

We denote by $\mathbf{h}_\pi$ the unique homomorphism $\mathbf{M}(P) \to \mathbf{FG}_\pi$, and by $\mathbf{rh}_\pi$ the unique homomorphism $\mathbf{RM}(P) \to \mathbf{FG}_\pi$. A *presentation* of a set of graphs L is a pair of the form (π, K) where either $K \subseteq \mathbf{M}(P)$ and $L = \mathbf{h}_\pi(K)$, or $K \subseteq \mathbf{RM}(P)$ and $L = \mathbf{rh}_\pi(K)$. A set L is context-free iff it has a presentation where K is an equational subset of $\mathbf{M}(P)$, or of $\mathbf{RM}(P)$ (by an immediate extension of Corollary 1.10). Actually, we shall only consider presentations where K is a subset of $\mathbf{RM}(P)$. This is not a loss of generality for the following reasons. Some of the operations defined by symbols in B may be associative and commutative, without being declared so. This means that they are treated as "ordinary" operations. Their associativity and commutativity properties are not used to reduce syntactic trees, as done in Section 3. Hence, the case where $K \subseteq \mathbf{M}(P)$ is nothing but the special case of that where $K \subseteq \mathbf{RM}(P)$ and none of the special symbols $/\!\!/_n$ and $\mathbf{e}_n$ occurs in K.

Definition 4.1 (*Parallel-composition*). We generalize the parallel-composition operation introduced in Example 1.9 to graphs of type n. For G, G' in $\mathbf{FG}(A)_n$, we let $H = G \mathbin{/\!/_n} G'$ be the n-graph formed as a disjoint union of G and G' in which, for all i, the ith source of G is fused with the ith source of G', in order to form the ith source of H. Hence, the operation $/\!/$ introduced in Example 1.9 is $/\!/_2$, and the operation $/\!/_M$ introduced in Definition 3.2 is $/\!/_1$.

We let $\mathbf{e}_n$ be a constant denoting the graph $\mathbf{n}$, with n vertices, no edge, and n pairwise distinct sources. It is clear that $/\!/_n$ is associative and commutative, and that $G \mathbin{/\!/_n} \mathbf{e}_n = \mathbf{e}_n \mathbin{/\!/_n} G = G$ for every graph of type n. If G is a graph of type n, we denote by $/\!/_n^p\, G$ the parallel composition of p disjoint copies of G. We obtain $\mathbf{n}$ if $p = 0$.

These definitions also apply if $n = 0$. Then $G \mathbin{/\!/_0} G'$ is the disjoint union of G and G' (denoted by $G \oplus G'$ in [2]; it has no source) and $\mathbf{e}_0$ is the empty graph $\mathbf{0}$.

Definition 4.2 (*Definable and parsable presentations*). Let (π, K) be a presentation of a set of graphs L, with $K \subseteq \mathbf{RM}(P)_n$. It is *definable* if the mapping $\mathbf{rh}_\pi : K \to L$ is definable as a transduction (see Definition 2.2). It is *monadic second-order parsable* (we shall simply say *parsable*) if the transduction $\mathbf{rh}_\pi^{-1} : L \to K$ is essentially definable.

A set of graphs is *strongly context-free* if it has a parsable presentation. (We shall prove that a strongly context-free set of graphs is context-free, which is not obvious from the definition.)

Let $\Gamma = \langle A, U, Q, u_1 \rangle$ be a context-free graph grammar, with nonterminal symbols $u_1, \ldots, u_n$. We say that it is constructed over π if, for every production rule $q : u \to D$, the graph D is the value of some term $\tilde{q}$ in $\mathbf{M}(P \cup U)$ (i.e., $D = \mathbf{h}_\pi(\tilde{q})$). We let $S_{\Gamma,\pi}$ be the system of equations $\langle u_1 = t_1, \ldots, u_n = t_n \rangle$ where each t_i is the sum of all terms $\tilde{q}$ such that q is a production rule with left-hand side u_i. If K is the first component of its least solution in $\mathcal{P}(\mathbf{M}(P))$, we have $\mathbf{L}(\Gamma) = \mathbf{rh}_\pi(K)$. It follows that $(\pi, \mathbf{h}_{ac}(K))$ is a presentation of $L = \mathbf{L}(\Gamma)$, with $\mathbf{h}_{ac}(K) \subseteq \mathbf{RM}(P)$.

We say that Γ is *monadic second-order parsable* if it is constructed over a signature, such that the associated presentation of $\mathbf{L}(\Gamma)$ as defined above is parsable. It follows that the set of graphs defined by such a grammar is strongly context-free.

Lemma 4.3. *For every signature of graph operations π, the transduction $\mathbf{rh}_\pi : \mathbf{RM}(P) \to \mathbf{FG}_\pi$ is definable.*

Proposition 4.4. (1) *A presentation (π, K) is definable iff K is definable, iff K is recognizable in* $\mathbf{RM}(P)$.

(2) *A set L has a definable presentation iff it is context-free.*

Lemma 4.3 says that every graph can be defined in its syntactic tree. A set of graphs is strongly context-free if, conversely, in every graph of the set, a syntactic tree of this graph can be defined.

Proof of Proposition 4.4. (1) If (π, K) is definable, then K is a definable subset of $\mathbf{RM}(P)$. Hence, it is recognizable by Theorem 3.3.

Conversely, if K is recognizable, then it is definable by Theorem 3.3. Since $\mathbf{rh}_\pi$ is definable by Lemma 4.3, its domain-restriction to K is also definable by Corollary 2.7.

(2) If L has a definable presentation (π, K), then K is recognizable (by (1)), and equational because $\mathbf{RM}(P)$ is finitely generated (so that every $\mathbf{RM}(P)$-recognizable set is equational). Hence, $\mathbf{rh}_\pi(K) = L$ is equational. It is thus context-free by Corollary 1.10.

Every context-free set has a presentation with $K \subseteq \mathbf{M}(P)$, that is definable by (1). □

The proof of Lemma 4.3 necessitates a few technical definitions.

Definition 4.5. Let $\pi = (P, \bar{\ })$ be a signature of graph operations as described in Section 1. We let $B := P - \{/\!/_n, \mathbf{e}_n \mid n \geqslant 0\}$.

We aim to define the graph $\mathbf{rh}_\pi(t)$, for t in $\mathbf{RM}(P)$, as a gluing of copies of the graphs defining the operations p for p in B. Following Definition 3.2, we consider a tree t in $\mathbf{RM}(P)$ as a graph in $\mathbf{RT}(B)$ also denoted by t. Hence, we let $t = \langle \mathbf{V}_t, \mathbf{E}_t, \mathbf{lab}_t, \mathbf{vert}_t, \mathbf{src}_t \rangle$. We let m be the sort of t, i.e., the type of the graph it defines.

For every p in B of profile $n_1 \times \cdots \times n_k \to n$, defined by a tuple $(D, e_1, \ldots, e_k)$, we denote by $\mathbf{v}(p, j, i)$ the ith vertex of e_j. We denote by $\mathbf{H}(p)$ the graph $D[\mathbf{n}_1/e_1, \ldots, \mathbf{n}_k/e_k]$. Hence, $\mathbf{H}(p)$ is the graph D from which the "nonterminal" edges $e_1, \ldots, e_k$ have been deleted.

For every e in $\mathbf{E}_t$, we let $\mathbf{H}(e)$ be a copy of $\mathbf{H}(\mathbf{lab}_t(e))$ disjoint from all other graphs under consideration. Technically, if $\mathbf{lab}_t(e) = p$, if $\mathbf{H}(p) = (V, E, \mathbf{lab}, \mathbf{vert}, \mathbf{src})$ then we can let

$$\mathbf{H}(e) := (V \times \{e\}, E \times \{e\}, \mathbf{lab}', \mathbf{vert}', \mathbf{src}') \text{ with } \mathbf{lab}'((e', e)) := \mathbf{lab}(e'),$$
$$\mathbf{vert}'((e', e), i) := (\mathbf{vert}(e', i), e),$$

and $\mathbf{src}'(j) := (\mathbf{src}(j), e)$ for all $e' \in E$, all $i \in [\tau(e')]$, all $j \in [\tau(\mathbf{H}(p))]$.

We also let $\mathbf{v}(e, j, i)$ denote the vertex $(\mathbf{v}(p, j, i), e)$ of $\mathbf{H}(e)$. The graphs $\mathbf{H}(e)$, $\mathbf{H}(e')$ are disjoint for every two edges $e, e' \neq e$, of t. We let

$$\mathbf{K}(t) := K := \langle \mathbf{V}_K, \mathbf{E}_K, \mathbf{lab}_K, \mathbf{vert}_K, \mathbf{src}_K \rangle$$

where $\mathbf{V}_K := [m] \cup \bigcup \{\mathbf{V}_{H(e)} \mid e \in \mathbf{E}_t\}$, $\mathbf{E}_K := \bigcup \{\mathbf{E}_{H(e)} \mid e \in \mathbf{E}_t\}$, $\mathbf{lab}_K$ is the mapping such that $\mathbf{lab}_K \upharpoonright \mathbf{E}_{H(e)} = \mathbf{lab}_{H(e)}$, $\mathbf{vert}_K$ is the mapping such that $\mathbf{vert}_K \upharpoonright \mathbf{E}_{H(e)} = \mathbf{vert}_{H(e)}$, and $\mathbf{src}_K := (1, 2, \ldots, m)$.

We finally let $\mathbf{glue}(t)$ be the graph $\mathbf{K}(t)/\approx$ where $\approx$ is the equivalence relation on $\mathbf{V}_{K(t)}$ generated by the set of pairs of vertices $R_1 \cup R_2$, where

$$R_1 := \{(i, \mathbf{src}_{H(e)}(i)) \mid i \in [m],\ e \text{ is a 0-edge of } t\},$$
$$R_2 := \{(\mathbf{v}(e', j, i), \mathbf{src}_{H(e)}(i)) \mid i \in [\tau(\mathbf{lab}_t(e))],\ e, e' \in \mathbf{E}_t,$$
$$e \text{ is a } j\text{-successor of } e'\}.$$

(Let us recall that if $\approx$ is an equivalence relation on the set of vertices of a graph K, then the quotient graph $H := K/\approx$ is defined as $\langle \mathbf{V}_K/\approx, \mathbf{E}_K, \mathbf{lab}_K, \mathbf{vert}_H, \mathbf{src}_H \rangle$, where $\mathbf{vert}_H(e, i)$ is the equivalence class of $\mathbf{vert}_K(e, i)$, and $\mathbf{src}_H(i)$ is that of $\mathbf{src}_K(i)$.)

Lemma 4.6. *For every t in* $\mathbf{RM}(P)$, *we have* $\mathbf{rh}_\pi(t) = \mathbf{glue}(t)$.

Proof. By induction on the structure of t. $\square$

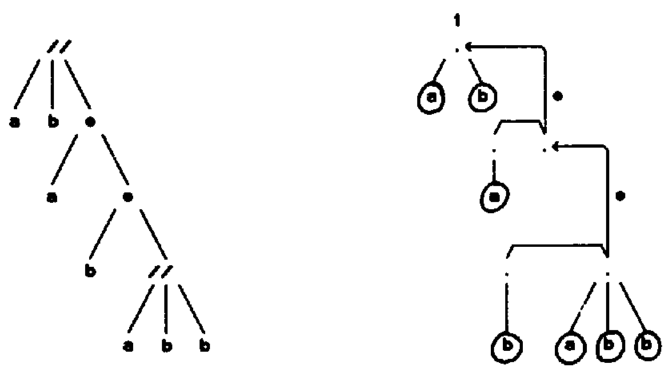
For future reference, let us define as follows a family of binary relations on $\mathbf{E}_t$: if $i, i' \in \mathbb{N}$, $p, p' \in B$, $x \in \mathbf{V}_{H(p)}$, $x' \in \mathbf{V}_{H(p')}$, then, for every e, e' in $\mathbf{E}_t$:

$$\eta_{i,p,x,i',p',x'}(e, e') \text{ iff } e \text{ is an } i\text{-edge, } e' \text{ is an } i'\text{-edge,}$$
$$\text{and } (x, e) \approx (x', e').$$

For each 6-tuple i, p, x, i', p', x' as above, the associated binary relation on t can be defined by a monadic second-order formula (to be interpreted in the logical structure representing t) also denoted by $\eta_{i,p,x,i',p',x'}$.

Example 4.7. Here is an example illustrating the construction of Definition 4.5. We use again the signature of graph operations P defining series-parallel graphs (see Example 1.9).

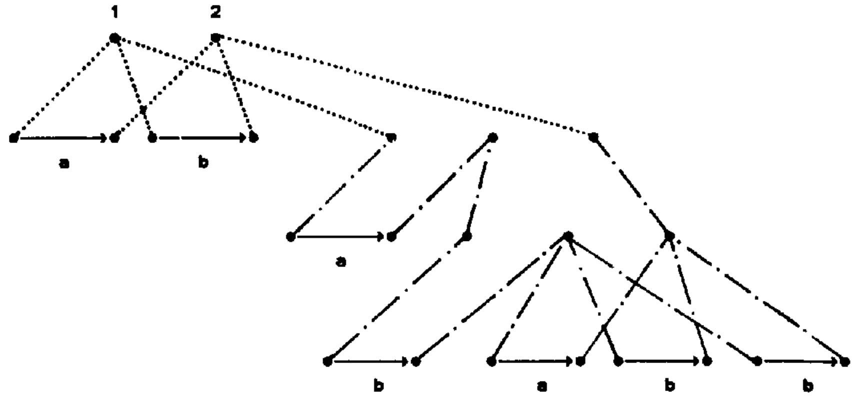
Figure 6 shows a term t in $\mathbf{RM}(P)$, and its representation as a graph in $\mathbf{RT}(B)$. Figure 7 shows the intermediate graph $K(t)$, and its quotient $\mathbf{glue}(t)$. In Fig. 7, the



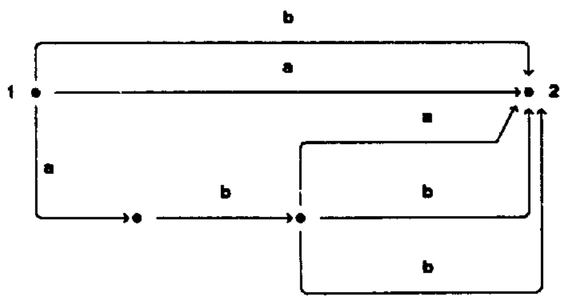
a reduced tree t in **RM**(P)

the tree t represented as a graph in **RT**(B)

Fig. 6.



the graph K(t)



the graph $rh_R(t)$ = glue(t)

Fig. 7.

dotted lines are not edges, but represent the pairs of vertices in R_1; the dash-dot lines represent pairs in R_2.

Proof of Lemma 4.3. This result is actually an immediate application of Lemma 4.6. Let us consider t in $\mathbf{RM}(P)$, of sort m. We wish to define in t the graph $\mathbf{glue}(t)$.

Let k be $\mathbf{Max}\{m, \mathbf{size}(\mathrm{H}(p)) \mid p \in B\}$ where the size of a graph H is $\mathbf{Card}(\mathbf{V}_H) + \mathbf{Card}(\mathbf{E}_H)$.

It follows from Definition 4.5 that $\mathbf{K}(t)$ can be defined in $\mathbf{D}_t \times [k]$. More precisely, the sources of $\mathbf{K}(t)$ are represented by the pairs $(\mathbf{src}_t(1), i)$, $i = 1, \ldots, m$. The vertices and edges of the graphs $\mathbf{H}(e)$, $e \in \mathbf{E}_t$ that form $\mathbf{K}(t)$ can be represented by pairs of the form (e, i), $i \in [\mathbf{size}(\mathbf{H}(e))]$. Hence, $\mathbf{K}(t)$ is definable in t. Its quotient $\mathbf{glue}(t)$ is definable in $\mathbf{K}(t)$ by Lemma 2.4, hence, $\mathbf{glue}(t)$ is definable in t. □

Theorem 4.8. *Let L be a strongly context-free set of graphs.*

(1) *L is definable, recognizable, and context-free.*

(2) *A subset of L is definable iff it is recognizable.*

Proof. (1) Let us assume that $L \subseteq \mathbf{FG}(A)_n$ is strongly context-free. We let $R := \mathbf{R}(A, n)$ and $\Delta = \langle \varphi, \psi_1, \ldots, \psi_k, (\theta_s)_{s \in R_{*k}} \rangle$ be a definition scheme, with set of parameters $\mathcal{W} = \{W_1, \ldots, W_m\}$, that defines, in every graph G in $\mathbf{FG}(A)_n$, a syntactic tree of this graph, relative to a fixed signature of graph operations. Hence, for every such graph G:

$$
\begin{aligned}
G \in L \ &\text{iff}\ \mathbf{def}_\Delta(G, \gamma) \text{ is defined for some } \gamma,\\
&\text{iff}\ (G, \gamma) \models \varphi,\\
&\text{iff}\ G \models \exists W_1, \ldots, W_m . [\varphi].
\end{aligned}
$$

It follows that L is definable. Hence, it is recognizable by Theorem 2.9. It is also context-free, because it is of finite tree-width (since its elements are the values of graph expressions constructed over a finite signature of graph operations (see Theorem 1.11), and because every recognizable set of graphs of finite tree-width is context-free.

(2) Let $M \subseteq L$ be recognizable. Since recognizability is preserved under inverse homomorphisms, the set $D := \mathbf{rh}_\pi^{-1}(M) \subseteq \mathbf{RM}(P)$ is recognizable. Hence, D is definable by Theorem 3.3. Let β be a formula defining it, i.e., be such that for every tree t in $\mathbf{RM}(P)$, $t \in D$ iff $t \models \beta$.

By Proposition 2.5, one can construct a formula $\bar{\beta}$ with free variables in $\mathcal{W}$ such that for every G in $\mathbf{FG}(A)_n$ and every $\mathcal{W}$-assignment γ in G:

$$t = \mathbf{def}_\Delta(G, \gamma) \text{ is defined and } t \models \beta \ \text{ iff } \ (G, \gamma) \models \bar{\beta}.$$

In order to complete the proof of the theorem, it suffices to prove the following claim.

Claim. *$G \in M$ iff $G \models \exists W_1, \ldots, W_m . [\bar{\beta}]$.*

Proof of the claim. If $G \models \exists W_1, \ldots, W_m . [\bar{\beta}]$, then $(G, \gamma) \models \bar{\beta}$ for some γ. Hence, $\mathbf{def}_\Delta(G, \gamma)$ is a tree t such that $t \models \beta$. Hence, $t \in D$ and $\mathbf{rh}_\pi(t) = G$ belongs to M.

Let conversely $G \in M$. For some $\gamma : \mathcal{W} \to G$, $\mathbf{def}_\Delta(G, \gamma)$ is a well-defined tree t such that $\mathbf{rh}_\pi(t) = G$. Hence, $t \in D$. It follows that $t \models \beta$ and that $(G, \gamma) \models \bar{\beta}$. Hence, $G \models \exists W_1, \ldots, W_m . [\bar{\beta}]$. □

Let us recall that every definable set of graphs is recognizable but that some recognizable sets are not definable [7]. Hence, part (2) of this theorem proves that every recognizable set that is "bounded" in some way (here, "bounded" means "included in a strong context-free set") is definable.

Special cases are known from [3] (see also [22, Theorem 3.2]) for sets of words and [11, Theorem 3.9] for sets of ranked ordered trees (see also [22, Theorem 11.1]). Our Theorem 3.3 establishes the corresponding property for the class of trees **RM**(P). All these sets (of words, of trees of various types) are strongly context-free as we shall see in Section 5.

What about languages? Let us say that a context-free (*string*) grammar Γ is *monadic second-order parsable* if the transduction from words in $L(\Gamma)$ to their derivation trees (relative to Γ) is essentially definable. These grammars generate regular languages by Büchi's theorem. Conversely, a context-free grammar generating a regular language is not necessarily monadic second-order parsable, as shown by the following example.

Example 4.9. Let Γ_1 be a context-free grammar generating $\{a^nb^n \mid n \geq 1\}$ with initial nonterminal u_1, and Γ_2 be another one generating $\{a, b\}^* - L(\Gamma_1)$, with initial nonterminal u_2. Let us assume that these grammars have disjoint sets of nonterminals. Let Γ be the union of Γ_1 and Γ_2, augmented with the rule $u_1 \to u_2$, generating $\{a, b\}^*$ from the initial nonterminal u_1.

The grammar Γ is not monadic second-order parsable. Let us assume, by contradiction, that it is. Let f be a definable transduction from words to derivation trees (of Γ) expressing that. Its codomain-restriction by the set of derivation trees of Γ_1 (that "eliminates" the derivation trees of Γ_2) would be definable, and Theorem 4.8 would show that Γ_1 is monadic second-order parsable. The language $L(\Gamma_1)$ would be definable, hence, regular, which is not the case.

We denote by $\mathbf{SCF}(A)_k$ the class of strongly context-free subsets of $\mathbf{FG}(A)_k$.

Theorem 4.10. (1) *The intersection of a strongly context-free set of graphs with a recognizable one is strongly context-free.*

(2) *The class* $\mathbf{SCF}(A)_k$ *is closed under union, intersection, and difference.*

(3) *If* L *is context-free and* L' *is strongly context-free, then the inclusion* $L \subseteq L'$ *is decidable. The equality of two strongly context-free sets of graphs is decidable.*

Proof. (1) Let L be strongly context-free. Let L' be definable. The definable transduction $f: L \to K$ expressing that L is strongly context-free can be restricted into a definable transduction $L \cap L' \to K$, establishing that $L \cap L'$ is strongly context-free. If L' is assumed to be recognizable, then $L'' := L \cap L'$ is recognizable, hence definable by Theorem 4.8. The above argument (with L'' instead of L') establishes that L'' is strongly context-free.

(2) Let L and L' be strongly context-free subsets of $\mathbf{FG}(A)_k$, given by parsable presentations (π, K) and (π, K') over a same signature π. (It is easy to make two signatures into a single one, by renaming some symbols if necessary.) It is not hard to establish that the presentation $(\pi, K \cup K')$ of $L \cup L'$ is parsable.

Since a strongly context-free set is definable, the other closure assertions follow from (1). Let us recall that $\mathbf{FG}(A)_k$ is not context-free as soon as A contains at least one symbol of type >1, by the results recalled in Definition 1.2 and Theorem 1.11. Hence, the class $\mathbf{SCF}(A)_k$ has no maximal element.

(3) Let L be a context-free and L' be a strongly context-free set of graphs over A of type k. Then $L \subseteq L'$ iff the set $M := L \cap (\mathbf{FG}(A)_k - L')$ is empty. Since $\mathbf{FG}(A)_k - L'$ is definable, the set M is context-free, and its emptiness can be tested. If L and L' are both strongly context-free, the two inequalities $L \subseteq L'$ and $L' \subseteq L$ can be tested, hence so can be the equality $L = L'$. □

Remark 4.11. Let us first recall that there exist context-free sets of graphs having an NP-complete membership problem. An example is the set of graphs of cyclic bandwidth at most 3 (see [17]).

Let now $L \subseteq \mathbf{FG}(A)_k$ be strongly context-free. The membership of a graph G in L can be decided in time $O(\mathbf{size}(G)^2)$. We sketch the proof of this fact. Let φ' be the formula that defines L (see Theorem 4.8(1)). One can find an integer m such that $\mathbf{twd}(L) \leq m$, and an algorithm that, for every graph G in $\mathbf{FG}(A)_k$, gives in time $O(\mathbf{size}(G)^2)$ the following possible answers (see [8]):

(1) $G \notin L$;

(2) $\mathbf{twd}(G) \leq m$ and $G \models \varphi'$;

(3) $\mathbf{twd}(G) \leq m$ and $G \models \neg\varphi'$.

Hence, one obtains $G \notin L$ in cases (1) and (3) and $G \in L$ in case (2).

Let us now consider the case where G is in L. The formula φ' is of the form $\exists W_1, \ldots, W_n . [\varphi]$, where $W_1, \ldots, W_n$ are parameters. From sets $W_1, \ldots, W_n$ satisfying φ, a syntactic tree of G can be obtained. We think, that by the results of [1], one can obtain in linear time a $\{W_1, \ldots, W_n\}$-assignment in G, and, from this assignment, one can construct in polynomial time, the syntactic tree it defines. □

Conjectures 4.12. We compare the various conjectures we made in the introduction (Conjectures 1-3), and in Section 2 (Conjecture (2.10)). We fix a nontrivial alphabet A. Without loss of generality, we shall only compare sets of 0-graphs. We let L_k denote the set of 0-graphs of tree-width at most k.

Let us consider the following statements.

(A) For every k, the set L_k is strongly context-free (Conjecture 2).

(B) If a set of graphs is context-free and definable, then it is strongly context-free (Conjecture 3).

(C) If a set of graphs is context-free and recognizable, then it is definable (by Theorems 1.11, 1.13, and 2.9, this satement is equivalent to Conjecture 1, also reformulated as Conjecture 2.10).

By the same three theorems, one can replace in statement (B) the condition "is context-free" by "is of finite tree-width", and statements (B) and (C) are respectively equivalent to:

(B′) for every definable set of graphs K, the set $K \cap L_k$ is strongly context-free,

(C′) for every recognizable set of graphs K, the set $K \cap L_k$ is definable.

We now observe that (A) and (B) are equivalent. Statement (B) implies (A) because L_k is context-free (Theorem 1.11) and definable [8]. And (A) implies (B) by Theorems 1.11(1) and 4.8. They imply the validity of (C), i.e., of Conjectures 1 and 2.10. It does not seem that (C) implies them.

Let us now consider again the diagram of Fig. 3. The conjecture that (A) and (B) hold is equivalent to stating that L_k belongs to **SCF**, and that the box with ??? is empty. It implies that the box with ? is empty. The apparently weaker conjecture that (C) holds is equivalent to stating that the box with ? is empty.

Example 4.13. The following set of graphs L has a parsable presentation (π, K) with $K \subseteq \mathbf{RM}(P)$, but no parsable presentation with $K \subseteq \mathbf{M}(P)$.

We let a be a symbol of type 0 and L be the set of graphs G of the form $/\!/_0^n\, a$, for $n > 0$. Assume that we have a parsable presentation (π, K) of L with $K \subseteq \mathbf{M}(P)$ for some P. The corresponding definition scheme is written with special predicates $\mathbf{card}_{p,q}$ for $p, q \in \mathbb{N}$, with q in some finite set N of integers. Hence, this definition scheme defines in each graph G of L a syntactic tree of L that is an ordered tree.

A graph in L is just an unordered set of undistinguishable edges. Let us consider the set L' of graphs in L with a number of edges equal to a multiple of some prime number M larger the least common multiple of all the elements of N. It is proved in [7] that the predicate $\mathbf{card}_{0,M}$ can be expressed by a monadic second-order formula in structures where some linear order is definable, which is the case of the syntactic trees of the graphs of L. It follows that L' can be defined by a formula using the special predicates $\mathbf{card}_{p,q}$ for p, q with q in N, hence that $\mathbf{card}_{0,M}$ can be expressed in terms of them. The proof of [7] showing that the counting monadic second-order logic is strictly more powerful than the noncounting one can be adapted and proves that this is not possible. Hence, one obtains a contradiction as desired.

5. Regular graph-grammars

We introduce a class of graph operations such that, for every signature π built with them, the transduction $\mathbf{rh}_\pi^{-1}$ is definable. It follows that every presentation of the form (π, K) where K is recognizable is parsable. The context-free graph-grammars associated with such presentations are called *regular*. The regular tree-grammars and the left-linear (word) grammars are of this form (via appropriate transformations into graph-grammars).

Definition 5.1 (*Regular graph operations*). As in the last section, we denote by P a finite signature of graph operations over the ranked alphabet A. We let $\mathcal{S} \subseteq \mathbb{N}$ be the finite set of sorts of this signature. We let $B := P - \{/\!/_n, \mathbf{e}_n \mid n \geq 0\}$. In addition, we assume that 0 is not in $\mathcal{S}$, and that all elements of A are of positive rank (i.e., type).

For every p in B, the associated graph operation is defined by a tuple $(\mathbf{D}(p), e_1, \ldots, e_k)$. The edges $e_1, \ldots, e_k$ of $\mathbf{D}(p)$ are its *nonterminal* edges. The other ones, labelled in A, are its *terminal* edges. We let $\mathbf{H}(p)$ be associated with p as in the construction of Definition 4.5.

We need some terminology concerning paths in (hyper)graphs. With a graph G, we associate the set

$$\mathbf{P}(G) := \{(v, e, i, j, v') \mid v, v' \in \mathbf{V}_G,\ e \in \mathbf{E}_G,\ i, j \in [\tau(e)],\ i \neq j,\ v = \mathbf{vert}_G(e, i),\\ v' = \mathbf{vert}_G(e, j)\}.$$

A *path from v to v'* in G (or *linking v to v'*), is a nonempty sequence π of elements of $\mathbf{P}(G)$ of the form

$$\pi = (v, e_1, i_1, j_1, v_1)(v_1, e_2, i_2, j_2, v_2) \ldots (v_{k-1}, e_k, i_k, j_k, v').$$

Its length is k, and its *sequence of vertices* is defined as

$$\mathbf{vert}(\pi) := (v, v_1, v_2, \ldots, v_{k-1}, v').$$

If $v_1, \ldots, v_{k-1}$ are *internal* vertices, i.e., are not sources of G, then π is an *internal path*. Note that v and v' may be internal or not. We say that π is a *terminal* path if all its edges are terminal.

Let us consider the following conditions concerning a graph $\mathbf{D}(p)$ for p in B.

(R1) $\mathbf{D}(p)$ has pairwise distinct sources.

(R2) $\mathbf{D}(p)$ has at least one edge. Either it is reduced to a single terminal edge, all vertices of which are sources, or each of its edges has at least one internal vertex.

(R3) Any two vertices of $\mathbf{D}(p)$ are linked by a terminal and internal path.

We say that P is *regular* if conditions (R1)-(R3) hold for each p in B, if 0 is not a sort of P, and if no element of A is of type 0.

The main theorem of this section is the following.

Theorem 5.2. *Let π be a signature of regular graph operations over A. The transduction $\mathbf{rh}_\pi^{-1} : \mathbf{FG}(A) \to \mathbf{RM}(P)$ is definable.*

The basic technique of this proof has already been used in [9] to establish that an infinite graph defined as the initial solution of a system of graph equations can be characterized by a monadic second-order formula. Before starting the proof, we give a few examples showing that the theorem fails without some of conditions (R1)-(R3).

Examples 5.3. We let A consist of a, b, c, d, f. We shall introduce graph operations p, r, s, t forming with $/\!/_1$ and $/\!/_2$ a signature π. We shall consider sets of graphs L that are not definable, but that are of the form $\mathbf{rh}_\pi(K)$ for recognizable sets K. If $\mathbf{rh}_\pi^{-1} : \mathbf{FG}(A) \to \mathbf{RM}(P)$ would be definable, its codomain restriction to the sets K would be definable, and the sets L would be definable by Theorem 4.8.

Let p be the graph operation defined by the graph:

$$1 \bullet \xrightarrow{\ a\ } \bullet \xrightarrow{\ u\ } \bullet \xrightarrow{\ b\ } \bullet\, 2$$

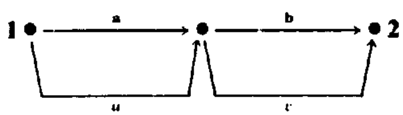
the nonterminal edge of which is labelled by u. We let c be an edge label of type 2, hence also a graph. Then $L_1 := \mathbf{rh}_\pi(\{p^n(c) \mid n \geq 0\})$ is the set of graphs of the following form:

$$1 \bullet \xrightarrow{a} \bullet \cdots \bullet \xrightarrow{a} \bullet \xrightarrow{c} \bullet \xrightarrow{b} \bullet \cdots \bullet \xrightarrow{b} \bullet\, 2$$

with as many b's as a's. Since one cannot express the equality of the cardinalities of two sets in monadic second-order logic, this set is not definable (see [7] for more details). Hence, $\mathbf{rh}_\pi^{-1}$ is not definable. Note that $\mathbf{D}(p)$ satisfies conditions (R1) and (R2), but not condition (R3).

Let now $r = d \mathbin{/\!/}_1 f$, where d and f are both of type 1. Let $L_2 := \mathbf{rh}_\pi(\{/\!/_1^n\, r \mid n \geq 1\})$. This set consists of graphs with one vertex, one source, and an equal number of unary edges labelled by d and by f. As above, it is not definable. Here, $\mathbf{D}(r)$ contradicts condition (R2) but satisfies the two others.

Now let s be the binary graph operation defined by the following graph:



with nonterminal edges labelled by u and v. Let also t be the graph

$$1,2 \bullet \!-\!\!-\!\!-\, ⓓ.$$

Then the set $L_3 := \mathbf{rh}_\pi(\{/\!/_2^n\, s(t,t) \mid n \geq 1\})$ is not definable by the same counting argument as above. Note that $\mathbf{D}(s)$ satisfies (R1)-(R3), and that t satisfies (R2) and (R3) only. The fusion of vertices due to the substitution of t for u and v in $\mathbf{D}(s)$ destroys condition (R2) (for $s(t,t)$).

Let us note in passing that every graph operation built as a finite combination of regular graph operations is regular.

We now start the proof of Theorem 5.2.

Remarks 5.4. Some preliminary remarks and notations are necessary. Let $t \in \mathbf{RM}(P)_n$. Then $G = \mathbf{rh}_\pi(t) := \mathbf{K}(t)/\approx\ \in \mathbf{FG}(A)_n$ where $\mathbf{K}(t)$ and $\approx$ are as in Definition 4.5. We shall use the notation of this definition.

We assume that t is not reduced to an isolated root, i.e., that G is not reduced to $\mathbf{n}$. We also assume that each graph $\mathbf{H}(p)$ has at least one internal vertex.

We let h be the canonical surjective homomorphism $\mathbf{K}(t) \to G = \mathbf{K}(t)/\approx$.

For every p in B, we let $\{f_{p,1}, \ldots, f_{p,n_p}\}$ be an enumeration of the set of edges of $\mathbf{H}(p)$. We let $E_{i,p,j}$ be the set of edges of G of the form $h((f_{p,j}, e))$ where $e \in \mathbf{E}_t$, $\mathbf{lab}_t(e) = p$, $1 \leq j \leq n_p$ and e is an i-edge. (Hence, $0 \leq i \leq M := \mathbf{Max}\{\rho(p) \mid p \in B\}$.)

Every edge e' of G is of the form $h((f, e))$ for a unique edge e of t, a unique edge f of $\mathbf{H}(p)$. We let $f_{p,j}$ be this edge and $k(e') := (i, p, j)$ where i is such that e is an i-edge of t.

It is clear that the family $\mathscr{E} = \{E_{i,p,j}\}$ forms a finite partition of $\mathbf{E}_G$.

For each p in B, let us also choose an internal vertex c_p of $\mathbf{H}(p)$ (we have assumed that each $\mathbf{H}(p)$ has one internal vertex hence is not reduced to a single terminal edge).

We let $C = \{h((c_p, e)) \mid e \in \mathbf{E}_t,\ p = \mathbf{lab}_t(e)\}$. Note that C is in bijection with $\mathbf{E}_t$. Our purpose is to establish that t can be defined in G in terms of $\mathscr{E}$ and C.

For every path π in G of the form

$$(v, e_1, i_1, j_1, v_1)(v_1, e_2, i_2, j_2, v_2) \dots (v_{k-1}, e_k, i_k, j_k, v')$$

we define its *trace* as the sequence

$$\mathbf{tr}(\pi) := (k(e_1), i_1, j_1)(k(e_2), i_2, j_2) \dots (k(e_k), i_k, j_k).$$

Now let μ be a path

$$(v, e_1, i_1, j_1, v_1) \dots (v_{k-1}, e_k, i_k, j_k, v')$$

in $\mathbf{H}(p)$. Let $e \in \mathbf{E}_t$, $p = \mathbf{lab}_t(e)$. We denote by $h(\mu, e)$ the following path in G:

$$(h(v, e), h(e_1, e), i_1, j_1, h(v_1, e)) \dots (h(v_{k-1}, e), h(e_k, e), i_k, j_k, h(v', e)).$$

If e is an i-edge then $\mathbf{tr}(h(\mu, e))$ is the sequence

$$\tau = ((i, p, m_1), i_1, j_1)((i, p, m_2), i_2, j_2) \dots ((i, p, m_k), i_k, j_k)$$

where $e_j = f_{p,m_j}$ for each $j = 1, \dots, k$.

Lemma 5.5. *Let G, t be as in Remarks 5.4. Let μ be a path in G of the form $h(\pi, e)$ for some e in $\mathbf{E}_t$, some path π in $\mathbf{H}(\mathbf{lab}_t(e))$, all vertices of which, except perhaps the last one are internal. If μ' is another path in G having the same trace and the same initial vertex, then $\mu' = \mu$.*

Proof. Let $\mu = (v, e_1, i_1, j_1, v_1)(v_1, e_2, i_2, j_2, v_2) \dots (v_{k-1}, e_k, i_k, j_k, v_k)$ and $\mu' = (v, e'_1, i'_1, j'_1, v'_1)(v'_1, e'_2, i'_2, j'_2, v'_2) \dots$ be the two paths. Since they have the same trace, they have the same length k and $i_s = i'_s$, $j_s = j'_s$ for all $s = 1, \dots, k$.

We have $v = h(\bar{v}, e)$, $e_1 = h(\bar{e}_1, e)$, and $e'_1 = h(\bar{e}'_1, e')$ for some $\bar{v}, \bar{e}_1, \bar{e}'_1$, and e'. We shall prove that $\bar{e}_1 = \bar{e}'_1$ and $e' = e$.

Let us assume that $e' \neq e$. We have $v = h(\bar{v}, e) = h(\bar{v}', e')$. This is possible only if $\bar{v}$ is a source of $\mathbf{H}(p)$ or $\bar{v}'$ is a source of $\mathbf{H}(p')$ (or both), where $p = \mathbf{lab}_t(e)$, $p' = \mathbf{lab}_t(e')$. Since $\bar{v}$ is not a source of $\mathbf{H}(p)$, $\bar{v}'$ must be a source of $\mathbf{H}(p')$. Since μ and μ' have the same trace, $k(e'_1) = k(e_1)$. It follows that $p = p'$ and $\bar{e}_1 = \bar{e}'_1$. Hence, $\bar{v}'$ is a source of $\mathbf{H}(p')$ iff $\bar{v}$ is a source of $\mathbf{H}(p)$. We obtain that $\bar{v}$ is a source of $\mathbf{H}(p)$, a contradiction.

Hence, $e' = e$. It follows that $v'_1 = v_1$. We can repeat this argument for the right factors $(v_1, e_2, i_2, j_2, v_2) \dots$ and $(v'_1, e'_2, i'_2, j'_2, v'_2) \dots$ of μ and μ'. We finally obtain $\mu = \mu'$. $\square$

We shall now construct logical formulas with parameters denoting the sets in $\mathscr{E}$ and the set C. In these formulas, we shall denote in the same way a variable and the object or the set of objects it defines. Hence, we take as set of parameters $\mathscr{W} := \{C\} \cup \{E_{i,p,j} \mid i \leq M,\ p \in B,\ j \leq m_p\}$.

Lemma 5.6. *Let G, t, $\mathscr{E}$, C be as in Remarks 5.4. For every p in B, every i in $[0, M]$, every vertex x of $\mathbf{H}(p)$, one can construct a formula $\chi_{p,i,x}(u, w, \mathscr{W})$ such that, for every two vertices u and w of G:*

$$(G, u, w, \mathscr{W}) \models \chi_{p,i,x} \text{ iff } u = h((c_p, e)) \textit{ for some i-edge e in } \mathbf{E}_t \textit{ such that } p = \mathbf{lab}_t(e) \textit{ and } w = h((x, e)).$$

Roughly speaking, this lemma says that every vertex w of G is definable from the corresponding vertex u in G. By "corresponding", we mean that u is of the form $h((c_p, e))$ and that $w = h((x, e))$ for some e.

Proof. We first assume that $x \neq c_p$. By condition (R3) there is an internal path π in $\mathbf{H}(p)$ linking c_p to x. Let

$$(c_p, f_{p,m_1}, i_1, j_1, v_1)(v_1, f_{p,m_2}, i_2, j_2, v_2) \ldots (\ldots i_k, j_k, x)$$

be this path. We let τ be the sequence

$$((i, p, m_1), i_1, j_1)((i, p, m_2), i_2, j_2) \ldots ((i, p, m_k), i_k, j_k).$$

Let $f_{p,m}$ be an edge of $\mathbf{H}(p)$ such that $\mathbf{vert}(f_{p,m}, l) = c_p$ for some m. Let $\chi_{p,i,x}(u, w, W)$ be the formula expressing that

(i) $u \in C$ and u is the lth vertex of an edge in $E_{i,p,m}$,

(ii) there is a path with trace τ from u to w.

In order to express that, for some edge e in a path, one has $k(e) = (i, p, j)$, it suffices to write $e \in E_{i,p,j}$. It follows that a monadic second-order formula $\chi_{p,i,x}$ can be written to express (i) and (ii).

If $u = h((c_p, e))$ for some i-edge e in $\mathbf{E}_t$ with $p = \mathbf{lab}_t(e)$ and $w = h((x, e))$, then (i) and (ii) hold with path $h(\pi, e)$ satisfying (ii).

Let conversely u, w satisfy $\chi_{p,i,x}$. Let μ be a path satisfying (ii). Then $u = h((c_p, e))$ and e is an i-edge by (i). The path $h(\pi, e)$ links u to $h((x, e))$; its trace is τ. Lemma 5.5 yields that $\mu = h(\pi, e)$, hence $w = h((x, e))$.

We still have to consider the case where $x = c_p$. We take $\chi_{p,i,x}$ expressing that $u \in C$, that $u = w$, and that u is the lth vertex of some edge in $E_{i,p,m}$, as in condition (i). This case is actually simpler than the previous one. $\square$

Lemma 5.7. *One can construct a formula $\mu_{p,i,p',i'}(u, w, \mathscr{W})$ such that, for every two vertices u, w of G:*

$$(G, u, w, \mathscr{E}, C) \models \mu_{p,i,p',i'} \text{ iff } u = h((c_p, e)),\ w = h((c'_p, e')) \textit{ for some } e, e' \textit{ in } \mathbf{E}_t \textit{ where } p = \mathbf{lab}_t(e),\ p' = \mathbf{lab}_t(e'),\ e \textit{ is an i-edge, and } e' \textit{ is an } i'\textit{-successor of } e \textit{ in } t.$$

Proof. For every p, p' in B, every i in $[0, M]$, every i' in $[\rho(p)]$, we construct a formula $\mu_{p,i,p',i'}(u, v, \mathcal{W})$ as follows. We let $x_1, \ldots, x_k$ be the sequence of vertices of the i'th nonterminal edge of p (these vertices belong to $\mathbf{H}(p)$). We let $(y_1, \ldots, y_k)$ be the sequence of sources of $\mathbf{H}(p)$.

We let $\mu_{p,i,p',i'}$ be the formula

$$\exists v_1, \ldots, v_k . [\chi_{p,i,x_1}(u, v_1, \mathcal{W}) \wedge \chi_{p',i',y_1}(w, v_1, \mathcal{W}) \ldots \wedge \chi_{p,i,x_k}(u, v_k, \mathcal{W}) \wedge \chi_{p',i',y_k}(w, v_k, \mathcal{W})].$$

If $u = h((c_p, e))$, $w = h((c_{p'}, e'))$ for some e, e' as in the statement, then $v_j = h((x_j, e)) = h((y_j, e'))$. It follows from Lemma 5.6 that $\chi_{p,i,x_j}(u, v_j, \mathcal{W})$ and $\chi_{p',i',y_j}(w, v_j, \mathcal{W})$ hold for all $j = 1, \ldots, k$. Hence, $\mu_{p,i,p',i'}(u, w, \mathcal{W})$ holds.

Let us conversely assume that $\mu_{p,i,p',i'}(u, w, \mathcal{W})$ holds. Let $v_1, \ldots, v_k$ be vertices such that $\chi_{p,i,x_j}(u, v_j, \mathcal{W})$ and $\chi_{p',i',y_j}(w, v_j, \mathcal{W})$ hold for all j. We have $u = h((c_p, e))$, $w = h((c_{p'}, e'))$ for some i-edge e labelled by p, some i'-edge e' labelled by p'. We need only prove that e' is an i'-successor of e. We make the following observation concerning h.

Fact. *If x, x' are vertices such that $h((x, e)) = h((x', e'))$, with $e' \neq e$ and, if x is internal in $\mathbf{H}(p)$, then x' is a source of $\mathbf{H}(p')$ and e is an ancestor of e'.*

We now complete the proof. We have $h((x_j, e)) = h((y_j, e'))$ for all j. By condition (R2), some vertex x_j is internal in $\mathbf{H}(p)$. It follows from the fact above that e is an ancestor of e'. If e' is not a successor of e, then e' is a successor of some edge $e'' \neq e$ such that e is an ancestor of e''. Every vertex $h((y_j, e'))$ is equal to $h((x_j, e))$, it is also equal to $h((z_j, e''))$, where $(z_1, \ldots, z_k)$ is the sequence of vertices of a nonterminal edge in $\mathbf{D}(\mathbf{lab}_t(e''))$. Because of (R2), some vertex z_i is internal in $\mathbf{D}(\mathbf{lab}_t(e''))$, but by the fact, $h(z_i, e''))$ cannot be equal to $h((x_i, e))$. Hence, e' is a successor of e, and actually an i'-successor since e' is an i'-edge.

This completes the proof of Lemma 5.7. $\square$

Proof of Theorem 5.2. If $G = \mathbf{n}$, then the tree t is $\mathbf{e}_n$. This special case can be easily recognized and treated separately.

We assume that each graph $\mathbf{H}(p)$ has internal vertices, hence, is not reduced to a single terminal edge.

Let us consider G and t such that $G = \mathbf{rh}_\pi(t)$. It follows from Lemma 5.7 that t can be defined in G in terms of $\mathscr{E}$ and C, where $\mathscr{E}$ and C are associated with G and t as in Remarks 5.4.

In particular it suffices to define $\mathbf{E}_t$ as equal to C with $\mathbf{lab}_t(c) = p$ iff $c \in E_{i,p,j}$ for some i, j. It is easy to define $\mathbf{V}_t$ in terms of C. The formulas defined in Lemma 5.7 are then useful to express the incidence relations in t. We omit the other details.

In order to complete the proof, we need only construct a formula φ with free variables in $\mathcal{W}$, such that, if G is an arbitrary graph in $\mathbf{FG}(A)_n$, if ν is a $\mathcal{W}$-assignment in G, then $(G, \nu) \models \varphi$ iff ν defines a tree t in $\mathbf{RM}(P)$ in the above sense, $G = \mathbf{rh}_\pi(t)$, and ν defines the sets $\mathscr{E}, C$ as in Remarks 5.4.

Construction of φ

We shall not construct it explicitly; we only indicate that φ should express the following facts:

(C1) $\mathscr{E}$ forms a partition of $\mathbf{E}_G$ (some sets of $\mathscr{E}$ may be empty), and the label of an edge in $E_{i,p,j}$ is that of $f_{p,j}$ for all i, p, j; if $E_{i,p,j}$ is nonempty for some j, then it is nonempty for all j, $1 \leq j \leq n_p$ (see Remarks 5.4 for the notation).

(C2) $C \subseteq \mathbf{V}_G$.

(C3) C and $\mathscr{E}$ define a tree t in $\mathbf{RM}(P)_n$ with $\mathbf{E}_t = C$, the structure of which is described by means of the formulas of Lemma 5.7.

To formulate the subsequent conditions, we introduce some notation.

$$C_{i,p} = \{v \in C \mid v \text{ belongs to an edge in } E_{i,p,j} \text{ for some } j\}$$

(if $C, \mathscr{E}$ are as in Remarks 5.4, then, $C_{i,p} = \{h((c_p, e)) \mid e \in \mathbf{E}_t,\ \mathbf{lab}_t(e) = p,\ e$ is an i-edge of $t\}$. Let x be a vertex of $\mathbf{D}(p)$. We say that (i, p, x, u) *defines* a vertex w of G if $u \in C_{i,p}$, $w \in \mathbf{V}_G$ and w is the unique vertex of G such that $(G, u, w, \mathscr{E}, C) \models \chi_{p,i,x}$.

(C4) For every triple i, p, j such that $1 \leq j \leq n_p$, the following holds. We let $(x_1, \ldots, x_k)$ be the sequence of vertices of $f_{p,j}$ in $\mathbf{H}(p)$. We require that, for every u in $C_{i,p}$, there is a unique edge in $E_{i,p,j}$ with sequence of vertices $(w_1, \ldots, w_k)$ such that (i, p, x_l, u) defines w_l, for each $l = 1, \ldots, k$. Conversely, we also require that for every edge in $E_{i,p,j}$, there is a unique u in $C_{i,p}$ such that (i, p, x_l, u) defines the lth vertex of this edge, for each $l = 1, \ldots, k$.

(C5) Every vertex of G belongs to some edge. A vertex v of G is defined by both (i, p, x, u) and (i', p', x', u') iff $\eta_{i,p,x,i',p',x'}(u, u', \mathscr{W})$ holds, where this formula is the "translation in G" of the formula $\eta_{i,p,x,i',p',x'}(u, u')$ introduced in Definition 4.5, that defines a binary relation on $\mathbf{E}_t$. Since t can be defined in G (by condition (C3)), Proposition 2.5 entails that one can express "in G" the properties of t.

Conditions (C1)-(C5) hold for $\mathscr{E}, C$ as defined in Remarks 5.4.

Let us now assume that ν is a $\mathscr{W}$-assignment satisfying them. Conditions (C1)-C3) express that $\mathscr{E}$ and C define a tree t in $\mathbf{RM}(P)$. Condition (C4) shows that G is a certain quotient of the graph $\mathbf{K}(t)$ defined in Definition 4.5. Condition (C5) expresses that $G = \mathbf{K}(t)/\approx$ where $\approx$ is the equivalence relation of Definition 4.5.

This concludes the proof of Theorem 5.2 in the case where all graphs $\mathbf{D}(p)$ for p in B have internal vertices. If some graph $\mathbf{D}(p)$ does not satisfy this property, then it is reduced to a single terminal edge. We let c_p be this edge. The above construction must be modified accordingly. We omit the technical details. $\square$

Theorem 5.8. *A presentation (π, K) where π is a signature of regular graph operations and K is recognizable, is parsable.*

Proof. Let $L = \mathbf{rh}_\pi(K)$. Since K recognizable it is definable (by Theorem 3.3). Hence, the transduction $\mathbf{rh}_\pi^{-1} : L \to K$ is definable since it is a codomain-restriction of the definable transduction $\mathbf{rh}_\pi^{-1} : \mathbf{FG}(A) \to \mathbf{RM}(P)$ by a definable set (Corollary 2.8). $\square$

Definition 5.9 (*Regular graph-grammars*). A context-free graph-grammar Γ is *regular* (we say also that it is a *regular* graph-grammar), if it is constructed over a signature of regular graph operations, and if the associated system of equations is ac-compatible.

Theorem 5.10. *Regular graph-grammar generate strongly context-free set of graphs.*

Proof. If L is generated by a regular graph-grammar, then it is of the form $\mathbf{rh}_\pi(K)$ for some recognizable set K, and some signature of regular operations π. The result follows from Theorem 5.8. □

Proposition 5.11. *Every left-linear (word) grammar, every regular-tree grammar is (can be translated into) a regular graph grammar.*

Proof. Let Γ be a left-linear (word) grammar. Its rules are of three possible forms

$$u \longrightarrow av,$$

$$u \longrightarrow a,$$

$$u \longrightarrow \varepsilon,$$

where u, v are nonterminals, a is a terminal symbol, and ε denotes the empty word. They translate into the following rules, forming the context-free graph-grammar $\hat{\Gamma}$:

$$u \longrightarrow 1\bullet \xleftarrow{\;a\;} \bullet \text{———} v$$

$$u \longrightarrow 1\bullet \xleftarrow{\;a\;} \bullet$$

$$u \longrightarrow 1\bullet.$$

For example, if the word *abcd* is generated by Γ, then the graph

$$1\bullet \xleftarrow{\;a\;} \bullet \xleftarrow{\;b\;} \bullet \xleftarrow{\;c\;} \bullet \xleftarrow{\;d\;} \bullet$$

is generated by $\hat{\Gamma}$. The grammar $\hat{\Gamma}$ is regular; the graph operations it uses are regular, and the associated system is ac-compatible (since the operation $/\!/$ does not occur in it). (The rules of third form are represented with $\bar{\mathbf{e}}_1$.)

Let us now consider the rules of a regular tree-grammar Γ. They are of the form

$$u \longrightarrow f(u_1, u_2, \ldots, u_k)$$

$$u \longrightarrow g$$

where $u, u_1, \ldots, u_k$ are nonterminal symbols, f and g are terminal ones of respective ranks k with $k \geq 1$ and $k = 0$. The rules of the corresponding graph-grammar $\hat{\Gamma}$ are:

$$u \longrightarrow f_M(u_1, \ldots, u_k)$$

$$u \longrightarrow g_M$$

where the operations f_M and g_M are as in Definition 3.2. Again, these operations are regular, and $\hat{\Gamma}$ is a regular graph-grammar. It generates the graphs corresponding to trees in the sense of Definition 3.2. □

Example 5.12. The following graph-grammar generates the set of trees $\mathbf{RT}(B)$ defined in Section 3. The initial nonterminal is u. It is regular because the graph operations of the forms f_M and g_M are regular (as above), and the corresponding system of equations satisfies conditions AC of Definition 3.7. Here are the rules of the grammar:

$$w \to f_M(u, \ldots, u)$$

$$w \to g_M$$

$$u \to u /\!/_1 w$$

$$u \to \mathbf{e}_1 .$$

Counterexample 5.13. The signature $\{a, b, \mathbf{e}_2, /\!/_2\}$ where a and c are as in Examples 5.3, is regular but the equation

$$u = a /\!/_2 b /\!/_2 u + \mathbf{e}_2$$

is not ac-compatible and the set of graphs it defines is not strongly context-free (because it is not definable). This example is essentially identical to Example 3.6.

6. Series-parallel graphs and graphs of tree-width at most 2

In this section, we let A be a finite alphabet of symbols all of rank 2, we let $\mathbf{SP} \subseteq \mathbf{FG}(A)_2$ be the set of oriented series-parallel graphs defined in Example 1.6. We shall prove that SP is strongly context-free. From this result, we shall derive the strong context-freeness of the set of graphs of tree-width at most 2.

We need a few technical lemmas on series-parallel graphs.

Let G be a graph in $\mathbf{FG}(A)$. By a *path* in G from x to y, where $x, y \in \mathbf{V}_G$, we mean in this section, a sequence of edges $(e_1, \ldots, e_n)$ such that $x = \mathbf{vert}_G(e_1, 1)$, $y = \mathbf{vert}_G(e_n, 2)$, $\mathbf{vert}_G(e_i, 2) = \mathbf{vert}_G(e_{i+1}, 1)$ for $i = 1, \ldots, n-1$. We have an *empty path* if $n = 0$, $x = y$, and a circuit if $x = y$ and $n \neq 0$.

Let z be a vertex. A path *goes through* z if z is a vertex of one of its edges. Otherwise, it *avoids* z. If G belongs to $\mathbf{FG}(A)_2$, a *long path* in G is a path from $\mathbf{src}_G(1)$ to $\mathbf{src}_G(2)$.

The following characterization of oriented series-parallel graphs is classical (see [24]).

Lemma 6.1. *A graph G in $\mathbf{FG}(A)_2$ is in* SP *iff it satisfies the following conditions*:

(1) *every vertex belongs to a long nonempty path,*

(2) *G has no circuit,*

(3) *there is no 4-tuple (x, y_1, y_2, z) of pairwise distinct vertices with pairwise nonintersecting paths from x to y_i, from y_i to z, for $i = 1, 2$, and from y_1 to y_2.*

Definitions 6.2. A graph G in SP is concretely given by means of a set of vertices $\mathbf{V}_G$ and of a set of edges $\mathbf{E}_G$. Its sources need not be specified because they can be determined from the orientations of edges in a unique way.

A *sub-SP-graph* H of G is a graph in **SP** with set of vertices $\mathbf{V}_H \subseteq \mathbf{V}_G$, with set of edges $\mathbf{E}_H \subseteq \mathbf{E}_G$, and such that $\mathbf{lab}_H = \mathbf{lab}_G \upharpoonright \mathbf{E}_H$ and $\mathbf{vert}_H = \mathbf{vert}_G \upharpoonright \mathbf{E}_H$. We denote this by $H \subseteq G$. (Two isomorphic sub-SP-graphs of G are *not* considered as equal.)

If e is an edge of G, we denote by $G[e]$ the sub-SP-graph of G with e as unique edge. Let H and H' be sub-SP-graphs of G such that $\mathbf{E}_H \cap \mathbf{E}_{H'} = \emptyset$. If $\mathbf{src}_H = \mathbf{src}_{H'}$, we denote by $H /\!/ H'$ the sub-SP-graph of G with set of edges $\mathbf{E}_H \cup \mathbf{E}_{H'}$. If $\mathbf{src}_{H'}(1) = \mathbf{src}_H(2)$, we denote by $H \bullet H'$ the sub-SP-graph of G with set of edges $\mathbf{E}_H \cup \mathbf{E}_{H'}$.

We say that G is $\bullet$*-atomic* (resp. $/\!/$*-atomic*) if G is not equal to $H \bullet H'$ (resp. to $H /\!/ H'$) for any two sub-SP-graphs H and H'. It is clear that a graph in **SP** is $\bullet$-atomic iff it is 2-connected or is reduced to a unique edge.

Lemma 6.3. *Let* $G \in \mathbf{SP}$.

(1) *If G is not $\bullet$-atomic, there exists a unique sequence $G_1, \ldots, G_k$, $k \geqslant 2$ of $\bullet$-atomic subgraphs of G such that $G = G_1 \bullet G_2 \bullet \cdots \bullet G_k$.*

(2) *If G is $\bullet$-atomic, then we have either G is reduced to a single edge, or there exists a unique set $\{G_1, \ldots, G_k\}$ of $/\!/$-atomic subgraphs of G such that $G = G_1 /\!/ G_2 /\!/ \cdots /\!/ G_k$.*

Proof. Easy induction on the number of edges of G. □

Definition 6.4 (*Constituents*). By induction on the number of edges of G, we define a set of subgraphs of G, denoted by $\mathbf{CONST}(G)$. If $\mathbf{E}_G$ is singleton, then we let $\mathbf{CONST}(G) := \{G\}$. Otherwise, G can be decomposed in a unique way as stated in Lemma 6.3. In both cases of Lemma 6.3 we let

$$\mathbf{CONST}(G) = \{G\} \cup \mathbf{CONST}(G_1) \cup \cdots \cup \mathbf{CONST}(G_k).$$

The elements of $\mathbf{CONST}(G)$ are called *the constituents of* G. Note that for every e in $\mathbf{E}_G$, $G[e]$ is a constituent of G.

For every graph G in **SP**, Lemma 6.3 yields, by an induction on the size of G, an expression $\mathbf{t}_G$ in $\mathbf{RM}(P)$ denoting G. Here, we let $P = \{/\!/, \bullet\} \cup A$ (the binary operation $/\!/$ has no unit). The expression $\mathbf{t}_G$ is associated with G in a unique way if, in Lemma 6.3, Case 1, we choose to write $G = G_1 \bullet (G_2 \bullet (\ldots (G_{k-1} \bullet G_k) \ldots))$.

Hence, we obtain in this way a bijection $G \mapsto \mathbf{t}_G$ of **SP** onto a definable subset K of $\mathbf{RM}(P)$. In order to establish that **SP** is strongly context-free, we need only prove that $\mathbf{t}_G$ (represented by a relational structure as explained in Section 3) is definable in G. For this purpose, we introduce some new technical definitions.

Definitions 6.5. Let $G \in \mathbf{SP}$ and $x, y \in \mathbf{V}_G$.

(1) We write $x \leqslant y$ iff there exists a path in G from x to y. Since G has no cycle, the relation $\leqslant$ is a partial order on G. We denote by $<$ the associated strict order.

(2) If $x, y \in \mathbf{V}_G$, $x < y$, then we denote by $G[x, y]$ the subgraph of G consisting of all vertices z, $x \leq z \leq y$, and all edges of G linking these vertices. Its two sources are x and y.

(3) For every edge e of G, we denote by $G[e]$ the graph $G[\mathbf{vert}_G(e, 1), \mathbf{vert}_G(e, 2)]$. This graph is clearly $\bullet$-atomic.

(4) We let $\mathbf{V}'_G$ be the set of vertices in $\mathbf{V}_G$ that are avoided by some long path. Then, for x in $\mathbf{V}'_G$, we denote by $\mathbf{left}(x)$ (resp. by $\mathbf{right}(x)$) the unique vertex y such that:

(4.1) $y < x$ (resp. $x < y$),

(4.2) every long path that goes through x also goes through y,

(4.3) there is a long path that goes through y and avoids x,

(4.4) if y' is any vertex satisfying (4.1)-(4.3) then $y' \leq y$ (resp. $y \leq y'$). (Hence, y is the vertex satisfying (4.1)-(4.3) that is as close to x as possible.)

The existence and unicity of such vertices y will be proved below.

(5) If x is as above, we let $G[x] := G[\mathbf{left}(x), \mathbf{right}(x)]$.

(6) In order to have uniform nota'ion, we also let $\mathbf{left}(e) := \mathbf{vert}_G(e, 1)$ and $\mathbf{right}(e) := \mathbf{vert}_G(e, 2)$ for $e \in \mathbf{E}_G$, so that $G[e]$ can also be written $G[\mathbf{left}(e), \mathbf{right}(e)]$.

Figure 8 illustrates these definitions. We have $y = \mathbf{left}(x)$, $z = \mathbf{right}(x)$; the vertices y' and z' satisfy conditions (4.1)-(4.3) but not condition (4.4). The graphs $G[x]$ and $G[e]$ are equal (i.e., are the same concrete subgraph).

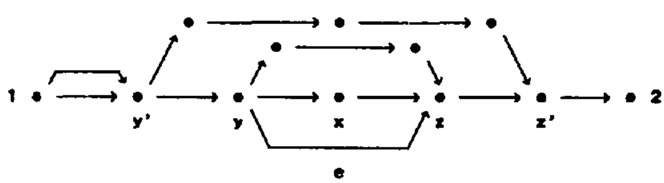


Fig. 8.

Lemma 6.6. *If $x \in \mathbf{V}'_G$, then* $\mathbf{left}(x)$ *and* $\mathbf{right}(x)$ *are well defined. The graph $G[x]$ is $\bullet$-atomic.*

Proof. The sources $\mathbf{src}_G(1)$ and $\mathbf{src}_G(2)$ satisfy conditions (4.1)-(4.3) of Definition 6.5.

Let us consider a long path containing x. On this path, there are two vertices y_1 and y_2 such that $y_1 < x < y_2$, that satisfy Conditions (4.2) and (4.3), and that are as close as possible to x. It follows that they also satisfy (4.4). Hence, conditions (4.1)-(4.4) define a unique pair of vertices that we denote functionally by $(\mathbf{left}(x), \mathbf{right}(x))$.

There exists a long path that avoids x and goes through $\mathbf{left}(x)$ and $\mathbf{right}(x)$. Otherwise, one would have two long paths avoiding x and going through y and

right(x) on one hand, and **left**(x) and z on the other, for some $y<$ **left**(x) and $z>$ **right**(x). The 4-tuple $(y, \mathbf{left}(x), \mathbf{right}(x), z)$ would contradict condition 3 of Lemma 6.1 (see Fig. 9). It follows that we cannot have $G[x]=H \bullet H'$ with x common to H and H'. If we had $G[x]=H \bullet H'$ with x in H and not in H', then condition (4.4) in the definition of **right**(x) would not be satisfied. Similarily, we cannot have x in H' and not in H. It follows that $G[x]$ is $\bullet$-atomic. $\square$

Lemma 6.7. *The set of $\bullet$-atomic constituents of G is equal to* $\{G[x] \mid x \in \mathbf{E}_G \cup \mathbf{V}'_G\}$.

Proof. Let $G=G_1 \bullet G_2 \bullet \cdots \bullet G_k$ with $G_1, \ldots, G_k$ $\bullet$-atomic, $k \geqslant 1$. For every i, there exists x in $\mathbf{E}_G \cup \mathbf{V}'_G$ such that $G_i=G[x]$. (This follows easily from Lemma 6.6.) Hence, every $\bullet$-atomic constituent of G is $G[x]$ for some x.

Conversely, let us consider $G[x]$. We prove that $G[x]$ belongs to **CONST**(G) by induction on the structure of G in the sense of Lemma 6.3.

(1) If $G=G_1 \bullet G_2 \bullet \cdots \bullet G_k$, $k \geqslant 2$ with $G_1, \ldots, G_k$ $\bullet$-atomic, then we have two subcases:

(i) x belongs to G_i and to G_{i+1} for some i; then $G[x]=G$, hence, $G[x] \in$ **CONST**(G).

(ii) x belongs to one and only one of the subgraphs G_i; then **left**(x) and **right**(x) belong both to that G_i and $G[x]=G_i[x]$. Hence, $G[x] \in$ **CONST**(G_i) (by induction) and $G[x] \in$ **CONST**(G).

(2) If $G=G_1 /\!/ G_2 /\!/ \cdots /\!/ G_k$ with $G_1, \ldots, G_k$ $/\!/$-atomic and $k \geqslant 2$; then, the argument is similar. Either $G[x]=G$ and then $G[x] \in$ **CONST**(G), or $G[x]=H[x]$ where H is a $\bullet$-atom equal to one of the G_i. By induction, we obtain $G[x] \in$ **CONST**(H) hence, $G[x] \in$ **CONST**(G). $\square$

Definition 6.8 (*Chains*). *Let H be a $\bullet$-atomic constituent of G. A H-chain in G is a sequence $C=(G_1, G_2, \ldots, G_k)$ satisfying the following conditions:*

(1) $G_1, \ldots, G_k$ are $\bullet$-atomic constituents of G that are sub-SP-graphs of H,

(2) $\mathbf{src}_{G_1}(1)=\mathbf{src}_H(1)$, and G_1 is maximal for inclusion with this condition, among the sub-SP-graphs of H,

(3) for every $i=2,3,\ldots,k-1$, G_i is maximal for inclusion among the sub-SP-graphs of H such that $\mathbf{src}_{G_{i-1}}(1)=\mathbf{src}_{G_i}(2)$.

It follows from these conditions that the graph $\bar{C} := G_1 \bullet G_2 \bullet \cdots \bullet G_k$ is a sub-SP-graph of H. A H-chain as above is *complete* if $\mathbf{src}_{G_k}(2)=\mathbf{src}_H(2)$.

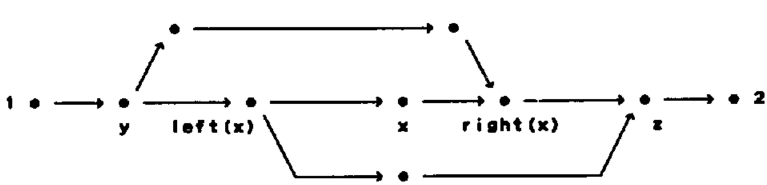


Fig. 9.

Lemma 6.9. *For every $\bullet$-atomic constituent H of G, we have $H = \bar{C}_1 \mathbin{/\!/} \bar{C}_2 \mathbin{/\!/} \cdots \mathbin{/\!/} \bar{C}_k$ where $\{C_1, \ldots, C_k\}$ is the set of complete H-chains of G.*

Proof. It is clear that each graph $\bar{C}_i$ is $/\!/$-atomic. By Lemma 6.3(2), we need only prove that every element of the set of $/\!/$-atomic constituents $\{H_1, \ldots, H_e\}$ such that $H = H_1 \mathbin{/\!/} H_2 \mathbin{/\!/} \cdots \mathbin{/\!/} H_e$, is of the form $\bar{C}_i$ for some complete H-chain C_i.

Let H_j be such a $/\!/$-atomic constituent. Then, $H_j = K_1 \bullet K_2 \bullet \cdots \bullet K_m$ where $K_1, \ldots, K_m$ are $\bullet$-atomic. It is clear that K_1 is the unique maximal $\bullet$-atomic sub-SP-graph of H such that $\mathbf{src}_H(1) = \mathbf{src}_{K_1}(1)$. Each K_i is the unique maximal sub-SP-graph of H such that $\mathbf{src}_{K_i}(1) = \mathbf{src}_{K_{i-1}}(2)$, where $i = 2, \ldots, m$. Hence, $H_j = \bar{C}$ where C is the complete H-chain $(K_1, \ldots, K_m)$. □

Theorem 6.10. *The set* **SP** *of oriented series-parallel graphs is strongly context-free.*

Proof. It follows from Lemma 6.1 that **SP** is definable, because the three conditions that characterize it as a subset of $\mathbf{FG}(A)_2$ can be written in monadic second-order logic. Hence, we need only consider a graph G in **SP**, and explain how $\mathbf{t}_G$ can be defined in G, in the sense of Section 2.

All notions introduced in Definition 6.5 are expressible in monadic second-order logic. Let us introduce a parameter X, denoting a subset of $\mathbf{D}_G := \mathbf{V}_G \cup \mathbf{E}_G$ (see Definition 2.8), and let us require about it the following conditions:

(C1) if $x \in X$, then $G[x]$ is defined,

(C2) if $x, y \in X$, $x \neq y$, then $G[x] \neq G[y]$,

(C3) for every x in $\mathbf{D}_G$ such that $G[x]$ is defined, there exists y in X such that $G[y] = G[x]$.

In this way, we have a bijection of X onto the set of $\bullet$-atomic constituents of G (by Lemma 6.7). One can write a formula $\varphi(x, y, X)$ saying that $G[x] \subseteq G[y]$.

It follows that the expression $\mathbf{t}_G$ in $\mathbf{RM}(P)$ (denoting G) derived from Lemma 6.3 can be defined in G. We do not give a formal construction, but we make a few observations from which the construction of a definition scheme defining it can be done.

(1) If $x \in X$, then $G[x]$ is equal to $\bar{C}_1 \mathbin{/\!/} \cdots \mathbin{/\!/} \bar{C}_e$ where $\{C_1, \ldots, C_e\}$ is the set of complete $G[x]$-chains in G. A $G[x]$-chain is completely defined by its first element $G[y]$ and the subgraph $G[x]$. Hence, the set $\{y_1, \ldots, y_e\} \subseteq C$ such that C_i is the $G[x]$-chain with first element $G[y_i]$, can be defined from x and X.

(2) If $x \in X$ and $C = (G[y_1], \ldots, G[y_m])$ is a complete $G[x]$-chain, then $\bar{C}$ is equal to

$$G[y_1] \bullet G[y_2] \bullet \cdots \bullet G[y_m].$$

It follows from the definition of a chain that each term y_{i+1} is definable from x, X, and y_i.

(3) The graph G is either $G[x]$ for some x in X (the one such that $G[y] \subseteq G[x]$ for every y in X) or is $\bar{C}$ for some complete G-chain $(G[y_1], \ldots, G[y_m])$ that one can also define.

From these remarks, one can define $\mathbf{t}_G$ in G by an appropriate definition scheme.

Note that the role of X is just to select a unique x in $\mathbf{V}'_G \cup \mathbf{E}_G$ such that $G[x] = H$ for each $\bullet$-atomic constituent H of G. The terms $\mathbf{t}_G$ associated with different sets X satisfying conditions (C1)-(C3) are thus isomorphic. $\square$

We now aim to extend Theorem 6.10 to other sets of graphs related to series-parallel graphs, and in particular to the set of graphs of tree-width at most 2.

Definitions 6.11 (*Disoriented series-parallel graphs*). Let $G \in \mathbf{FG}(A)_k$. We say that H in $\mathbf{FG}(A)_k$ is obtained from G by *reorientation* if $\mathbf{V}_H = \mathbf{V}_G$, $\mathbf{E}_H = \mathbf{E}_G$, $\mathbf{lab}_H = \mathbf{lab}_G$, $\mathbf{src}_H = \mathbf{src}_G$, and, for some subset W of $\mathbf{E}_G$,

$$\mathbf{vert}_H(e) = (\mathbf{vert}_G(e, 2), \mathbf{vert}_G(e, 1)) \quad \text{if } e \in W$$

$$= \mathbf{vert}_G(e) \quad \text{if } e \notin W.$$

We write this $H = G(W)$. It is clear that if $H = G(W)$, we also have $G = H(W)$.

For every graph G, we denote by $\sigma_0(G)$ the 0-graph equal to G except that its sources are turned into internal vertices.

The set $\mathbf{DSP}$ of *disoriented series-parallel graphs* is defined as $\{\sigma_0(G(W)) \mid G \in \mathbf{SP}, W \subseteq \mathbf{E}_G\}$.

Theorem 6.12. *The set* $\mathbf{DSP}$ *is strongly context-free.*

Proof. For every G in $\mathbf{FG}(A)_0$, every $W \subseteq \mathbf{E}_G$, every x, y in $\mathbf{V}_G$, we denote by $G(W, x, y)$ the graph in $\mathbf{FG}(A)_2$ consisting of $G(W)$ equipped with (x, y) as sequence of sources.

Let $\mathcal{W} = \{W, Y_1, Y_2, X\}$. One can modify the definition scheme Δ of Theorem 6.10 into a definition scheme Δ' with set of parameters $\mathcal{W}$ such that, for every graph G in $\mathbf{FG}(A)_0$, for every assignment $\gamma: \mathcal{W} \to G$, we have:

(1) $\mathbf{def}_{\Delta'}(G, W, Y_1, Y_2, X)$ is defined iff W is a set of edges of G, Y_1, and Y_2 are singletons $\{y_1\}$, and $\{y_2\}$, $\mathbf{def}_\Delta(G(W, y_1, y_2), X)$ is defined, and

(2) $\mathbf{def}_{\Delta'}(G, \gamma) = \mathbf{def}_\Delta(G(W, y_1, y_2), X)$ if they are both defined.

It follows that $\mathbf{def}_{\Delta'}(G, \gamma)$ is defined for some γ iff $G \in \mathbf{DSP}$. If this is the case, then $\mathbf{def}_{\Delta'}(G, \gamma)$ is the tree $\mathbf{t}_H$ denoting the oriented series-parallel graph $H = G(W, y_1, y_2)$.

By using the information given by W, i.e., the reoriented edges, it is easy to modify Δ' so that it defines an expression tree for G defining it by means of the operations $\bullet$, $/\!/$, σ_0, the constants a for all a in A, and the operation $\sigma_{2,1}$ that reverses the sequence of sources of a 2-graph. $\square$

Definition 6.13. A *basic* graph is a graph G in $\mathbf{FG}(A)_0$ of the following two possible forms:

(1) either G is reduced to one vertex and one edge (forming a loop),

(2) or G is a two-connected graph in **DSP**, equivalently, a graph of the form $\sigma_0(G(W))$ for some •-atomic (oriented) series-parallel graph G in **SP**, and some $W \subseteq \mathbf{E}_G$.

As in [23], we say that a graph is two-connected if it is nonempty, connected, and has no cut-vertex. Hence, a graph reduced to a single edge is two-connected.

Definition 6.14 (*Tree-gluings*). Let $L \subseteq \mathbf{FG}(A)_0$. Let T be an unoriented tree with set of nodes V. Let f be a mapping associating with every node v of V a graph $f(v)$ isomorphic to a graph in L. We assume that if $v \neq v'$, then $f(v)$ and $f(v')$ are disjoint. For every edge (x, y) of T, we let $f(x, y)$ be a pair (u, v) where u is a vertex of $f(x)$ and v is a vertex of $f(y)$. We assume that $f(y, x) = (v, u)$ if $f(x, y) = (u, v)$.

With (T, f) as above, we associate a graph $\mathbf{glue}(T, f)$ in $\mathbf{FG}(A)_0$ as follows. We let first K be the (disjoint) union of the graphs $f(v)$, $v \in V$. We let $\approx$ be the equivalence relation on $\mathbf{V}_K$ generated by the set of all pairs $f(x, y)$ for all edges (x, y) of T. Finally, we define $\mathbf{glue}(T, f) := K/\approx$. We say that this graph is a *tree-gluing* of graphs in L.

A maximal two-connected subgraph of a graph G is called a *block* of G [23]. Hence, every connected graph is isomorphic to a tree-gluing of its blocks. Conversely, if G is a tree-gluing of two-connected graphs, the components $f(v)$ are the blocks of G.

Lemma 6.15. *A binary 0-graph is of tree-width at most 2 iff its blocks are basic graphs.*

It follows from Theorem 6.12 that the set of basic graphs is strongly context-free. Hence, the following two lemmas entail immediately Theorem 6.18.

Lemma 6.16. *Let $L \subseteq \mathbf{FG}(A)_0$ be a strongly context-free set of nonempty connected graphs. The set L' of nonempty graphs, all connected components of which are in L, is strongly context-free.*

Lemma 6.17. *Let $L \subseteq \mathbf{FG}(A)_0$ be a strongly context-free set of two-connected graphs. The set L' of tree-gluings of graphs of L is strongly context-free.*

Theorem 6.18. *Let A consist of symbols of rank 2. The set of graphs in $\mathbf{FG}(A)_0$ of tree-width at most 2 is strongly context-free.*

Proof of Lemma 6.16. Let (π, K) be a parsable presentation of L, with signature P. We can assume that P contains $/\!/_0$ (we add this symbol to P otherwise). We let K' be the set of terms in $\mathbf{RM}(P)$ of the form $t_1 /\!/_0 t_2 /\!/_0 \cdots /\!/_0 t_k$, for $k > 0$, $t_1, \ldots, t_k$ in K. We claim that the presentation (π, K') of L' is parsable.

Let Δ be a definition scheme for L, with set of parameters $\mathcal{W}$. We aim to construct a definition scheme Δ' for L'. Its set of parameters will be some $\mathcal{W}'$. In particular, we wish to have, for all $\gamma' : \mathcal{W}' \to G$, where $G \in \mathbf{FG}(A)_0$, $t = \mathbf{def}_{\Delta'}(G, \gamma')$ iff $t = t_1 /\!\!/_0 \cdots /\!\!/_0 t_k$, $G = G_1 /\!\!/_0 \cdots /\!\!/_0 G_k$, and $t_i = \mathbf{def}_\Delta(G_i, \gamma_i)$ for some $\gamma_i : \mathcal{W} \to G_i$.

Let $G \in \mathbf{FG}(A)_0$ and x be an item of G (i.e., an edge or a vertex of G). We denote by G_x the connected component of G containing x. If γ is an assignment $\mathcal{W} \to G$, then, we denote by γ_x the $\mathcal{W}$-assignment $\mathcal{W} \to G_x$ such that $\gamma_x(W) = \gamma(W) \cap \mathbf{D}_{G_x}$ for all W in $\mathcal{W}$. Since G_x is definable in (G, x), it is not hard to construct from Δ, a definition scheme Δ_1 using an extra parameter Y such that, for every G in $\mathbf{FG}(A)_0$, for every x in $\mathbf{D}_G$, for every $\gamma : \mathcal{W} \to G$,

$$\mathbf{def}_{\Delta_1}(G, \gamma, \{x\}) = \mathbf{def}_\Delta(G_x, \gamma_x)$$

and such that one side of this equality is defined iff the other is. Let X be a nonempty subset of $\mathbf{D}_G$ such that

(i) $G_x \in L$ for each x in X,

(ii) $G_x \neq G_y$ for x, y in X, $x \neq y$,

(iii) each connected component of G is G_x for some x in X.

Such a set exists iff G belongs to L'. Let γ be a $\mathcal{W}$-assignment in G such that $t_{\gamma,x} := \mathbf{def}_\Delta(G_x, \gamma_x)$ is defined for each x in X. It is clear that the graph

$$t_\gamma := t_{\gamma,x_1} /\!\!/_1 \cdots /\!\!/_1 t_{\gamma,x_m}$$

in $\mathbf{FG}(\mathbf{R}(P, 1))_1$, where $X = \{x_1, \ldots, x_m\}$, is the desired tree, representing the graph $G = G_{x_1} /\!\!/_0 \cdots /\!\!/_0 G_{x_m}$. (Note here the use of $/\!\!/_1$ in the definition of t_γ: this is because we consider graphs representing trees in the sense of Definition 3.2 and not terms.)

One can construct from Δ_1 a definition scheme Δ'' with set of parameters $\mathcal{W}'' := \mathcal{W} \cup \{X\}$ such that for every graph G in $\mathbf{FG}(A)_0$, for every assignment $\gamma'' : \mathcal{W}'' \to G$, we have the following conditions:

(1) $\mathbf{def}''_\Delta(G, \gamma'')$ is defined iff the subset $\gamma''(X)$ of $\mathbf{D}_G$ satisfies conditions (i)–(iii). (It follows that $G \in L'$ iff $\mathbf{def}_{\Delta''}(G, \gamma'')$ is defined for some γ''.)

(2) If $S = \mathbf{def}_{\Delta''}(G, \gamma'')$ is defined, then S is the disjoint union of the graphs representing the trees t_{γ'',x_i}, $i = 1, \ldots, m$, where $\gamma''(X) = \{x_1, \ldots, x_m\}$. Hence, S is a finite disjoint union of trees in $\mathbf{RM}(P)$. The tree t we wish to construct is the result of the fusion of the roots of the m trees forming S into a single node (the root of t).

By Lemma 2.4, t can be defined in S (whence in G, by Corollary 2.6). We need only choose which of the roots of the trees $t_1, \ldots, t_m$ will be taken as the root of t. This choice can be made by means of an extra parameter Z.

It follows that one can construct Δ' with set of parameters $\mathcal{W}' := \mathcal{W} \cup \{X, Z\}$ such that, for every assignment $\gamma' : \mathcal{W}' \to G$ we have

(1) $\mathbf{def}'_\Delta(G, \gamma')$ is defined iff the subset $\gamma'(X)$ of $\mathbf{D}_G$ satisfies conditions (i)–(iii),

(2) $\gamma'(Z)$ is a singleton and $\gamma'(Z) \subseteq \gamma'(X)$,

(3) if $t = \mathbf{def}_{\Delta'}(G, \gamma')$ is defined then $t = t_1 /\!\!/_1 \cdots /\!\!/_1 t_m$ and t defines $G = G_1 /\!\!/_0 \cdots /\!\!/_0 G_m$.

The role of Z is to tell that the root of t_i, where i is such that $\gamma'(Z) = \{x_i\} \subseteq \gamma'(X)$ is taken as root of t. This concludes the proof. $\square$

Sketch of the proof of Lemma 6.17. A completely formal proof would be quite long. We only give a sketch.

Let G be a graph in $\mathbf{FG}(A)_0$. Every edge e of G belongs to a unique block G_e of G (see [23]). The set of items of G_e can be defined in G. (To be precise, one can construct a formula $\varphi(X, Y)$ such that $(G, X, Y) \models \varphi$ iff $X = \{e\}$ and $Y = \mathbf{D}_{G_e}$ for some e in $\mathbf{E}_G$.) The graph G belongs to L' iff each of these subgraphs G_e belongs to L. Since L is definable, this can be expressed in monadic second-order logic. A syntactic tree of G_e (with respect to the given parsable presentation of L) can be defined in G_e, hence in G, whenever G_e belongs to L.

Let us now consider G belonging to L'. It is a tree-gluing of a family $\{G_e \mid e \in E\}$ of blocks of G, for some $E \subseteq \mathbf{E}_G$, and the associated tree t (as in Definition 6.14), can be defined in G. By combining the tree t with the syntactic trees of the various graphs G_e, $e \in E$, one can define (by an appropriate definition scheme), a syntactic tree of G (with respect to an appropriate presentation over some extension of the given signature of graph operations). We omit the details. □

In the following extension of Theorem 6.18, we do not limit A to symbols of rank 2.

Theorem 6.19. *The set of graphs in* $\mathbf{FG}(A)_0$ *of tree-width at most* 2 *is strongly context-free.*

Proof. In Theorem 6.10, we have shown how a syntactic tree $\mathbf{t}_G$ of an oriented series-parallel graph G can be defined in G, in monadic second-order logic. From this tree, it is not hard to obtain a tree-decomposition $(\mathbf{t}_G, f)$ of G, of width at most 2. This tree-decomposition can be defined in G. One can construct a formula $\chi(x, y)$ such that $G \models \chi(x, y)$ iff x represent a node u of $\mathbf{t}_G$ and y is a vertex in $f(u)$.

This construction extends to disoriented series-parallel graphs (by Theorem 6.12) and to graphs of tree-width at most 2 (by Theorem 6.18). To summarize, in every (binary) graph of tree-width at most 2, one can define a tree-decomposition of width at most 2 of this graph, by a monadic second-order formula.

Let us now consider the case where A has symbols of all ranks. It is shown in [8] that the graphs over A can be encoded as binary graphs over a new alphabet of binary symbols. To be more specific, a graph G is encoded into a graph $K(G)$ with the same vertices. The (hyper)edges of G are replaced by cliques. It follows that every tree-decomposition of $K(G)$ is also a tree-decomposition of G.

Given G in $\mathbf{FG}(A)_0$, one can define $K(G)$ in G (in the sense of Definition 2.2), one can define in G a tree-decomposition of $K(G)$, of tree-width at most 2 if such a tree-decomposition does exist. It is not difficult to convert a tree-decomposition into a syntactic tree over an appropriately defined signature of graph operations. □

We hope that these techniques extend to tree-width k for every k, which would give a proof of Conjecture 2 (see the introduction and the discussion of Conjectures

4.12), and, finally, a better understanding of the relations between definability, recognizability, and context-freeness for sets of graphs.

Acknowledgment

I thank A. Arnold for his helpful remarks on a first version of this work.

References

[1] S. Arnborg, J. Lagergren and D. Seese, Problems easy for tree-decomposable graphs, in: *ICALP'88*, Lecture Notes in Computer Science **317** (Springer, Berlin, 1988) 38-51.

[2] M. Bauderon and B. Courcelle, Graph expressions and graph rewritings, *Math. Systems Theory* **20** (1987) 83-127.

[3] J. Büchi, Weak second order logic and finite automata, *S. Math. Logik Grundlagen Math.* **5** (1960) 66-92.

[4] B. Courcelle, Equivalences and transformations of regular systems. Applications to recursive program schemes and grammars, *Theoret. Comput. Sci.* **42** (1986) 1-122.

[5] B. Courcelle, Graph rewriting: an algebraic and logic approach, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, Amsterdam, 1990) 193-242.

[6] B. Courcelle, The monadic second-order logic of graphs: definable sets of finite graphs, in: *Workshop on Graph Theoretical Concepts in Computer Science*, Lecture Notes in Computer Science **344** (Springer, Berlin, 1989) 30-53.

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

[8] B. Courcelle, The monadic second-order logic of graphs III: tree-decompositions minors, and complexity issues, 1990, submitted.

[9] B. Courcelle, The monadic second-order logic of graphs IV: definability properties of equational graphs, *Ann. Pure Appl. Logic* **49** (1990) 193-255.

[10] B. Courcelle, On recognizable sets and tree-automata, in: H. Ait-Kaci and M. Nivat, eds., *Resolution of Equations in Algebraic Structures, Vol. 1, Algebraic Techniques* (Academic Press, New York, 1989) 93-126.

[11] J. Doner, Tree acceptors and some of their applications, *J. Comput. System Sci.* **4** (1970) 406-451.

[12] P. Franchi-Zannettacci, Attributs sémantiques et schémas de programmes, Thèse d'Etat, Bordeaux-1 University, 1982.

[13] F. Gecseg and M. Steinby, *Tree-Automata* (Akademia Kaido, Budapest, 1984).

[14] J. Goguen, J. Thatcher, E. Wagner and J. Wright, Initial algebra semantics and continuous algebras, *J. ACM* **24** (1977) 68-95.

[15] P. Gohon, An algorithm to decide whether a rational subset of $\mathbb{N}^k$ is recognizable, *Theoret. Comput. Sci.* **41** (1985) 51-59.

[16] A. Habel and H.J. Kreowski, *May We Introduce to You: Hyperedge Replacement*, Lecture Notes in Computer Science **291** (Springer, Berlin, 1987) 15-26.

[17] J. Leung, J. Witthof and O. Vornberger, On some variations on the bandwidth minimization problem, *SIAM J. Comput.* **13** (1984) 650-667.

[18] J. Mezei and J. Wright, Algebraic automata and context-free sets, *Inform. and Control* **11** (1967) 3-29.

[19] U. Montanari and F. Rossi, *An Efficient Algorithm for the Solution of Hierarchical Networks of Constraints*, Lecture Notes in Computer Science **291** (Springer, Berlin, 1987) 440-457.

[20] N. Robertson and P. Seymour, Some new results on the well-quasi-ordering of graphs, *Ann. Discrete Math.* **23** (1984) 343-354.

[21] D. Seese, The structure of the models of decidable monadic theories of graphs, Preprint 1987, *Ann. Pure Appl. Logic*, in press.
[22] W. Thomas, Automata on infinite objects, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, Amsterdam, 1990) 133–191.
[23] W. Tutte, *Graph Theory* (Addison-Wesley, Reading, MA, 1984).
[24] J. Valdes, E. Lawler and R. Tarjan, The recognition of series-parallel digraphs, *SIAM J. Comput.* 11 (1981) 298–313.