ELSEVIER

Theoretical Computer Science 163 (1996) 1–54

Theoretical Computer Science

Tutorial

Basic notions of universal algebra for language theory and graph grammars☆

Bruno Courcelle*

Université Bordeaux I, Laboratoire d'Informatique (associé au CNRS), 351, Cours de la Libération, 33405 TALENCE Cedex, France

Received January 1995
Communicated by M. Nivat

Abstract

This paper reviews the basic properties of the equational and recognizable subsets of general algebras; these sets can be seen as generalizations of the context-free and regular languages, respectively. This approach, based on Universal Algebra, facilitates the development of the theory of formal languages so as to include the description of sets of finite trees, finite graphs, finite hypergraphs, tuples of words, partially commutative words (also called traces) and other similar finite objects.

Contents

☆Supported by ESPRIT Basic Research Working Group "Computing by graph transformation" and by the "Programme de Recherches Coordonnées: Mathématiques et Informatique".

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

SSDI 0304-3975(95)00145-X

0. Introduction

The context-free and the regular languages are the two main classes of formal languages. We review how their basic concepts can be used for the description of sets of finite objects like trees, graphs, hypergraphs, tuples of words, traces (equivalence classes of words with respect to partial commutation).

Context-free languages are usually defined by *grammars*, in terms of certain iterated rewritings; they can also be described as the components of the least solutions of certain *systems of recursive set equations* in languages. Grammars defining trees, graphs, hypergraphs have been introduced. But in each case one faces the problem of deciding which grammars are context-free and which are not; one has to decide among the variants in definitions which are the really important ones. One has also many basic facts to reprove in each case like the decidability of the emptiness problem. By the theorem of Ginsburg and Rice [30], the context-free languages can be characterized as the components of the least solutions of systems of equations naturally associated with context-free grammars. These systems are mutually recursive definitions of sets of words using set union and extension to sets of the concatenation of words. Similar systems of equations (i.e. of recursive definitions) can be used for sets of finite objects like trees or graphs, provided operations on them generalizing concatenation are defined. As soon as they are formulated in terms of systems of equations, many results concerning context-free grammars can be proved at the Universal Algebra level, and their applicability to context-free grammars of trees or graphs is immediate. We shall develop this view point and survey the main properties of systems of equations that hold in general. The sets defined by these systems will be called the *equational sets*. They will be our "general context-free" sets. We shall give examples dealing with trees and graphs.

The regular languages can be defined in several equivalent ways: by finite automata (deterministic or not), by rational expressions, by finite congruences. Kleene's theorem states the equivalence of these definitions. When we call these languages "rational" we

refer to their descriptions by rational expressions. Following Mezei and Wright [38] we call them "recognizable" in order to refer to their characterization by finite congruences. (Many authors use the term "recognizable" in the context of some notion of automaton; see the discussion in [12]). The notion of a rational set makes sense in arbitrary monoids (a rational set is defined by a rational expression). So does that of a recognizable set (defined in terms of finite congruences). However, Kleene's Theorem does not hold in all monoids. In general, we must distinguish the recognizable sets from the rational ones. Furthermore, the notion of a rational set is particular to monoids (see Section 5 for a discussion) whereas the notion of a recognizable one is more general because finite congruences are meaningful in arbitrary algebras. Hence, we shall take the *recognizable sets* (introduced by Mezei and Wright in their fundamental paper [38]) as our generalization of regular languages. (We use "regular" as a neutral term designating a class of languages without reference to any specific definition technique.) Let us conclude by discussing automata briefly. The notion of a finite automaton extends nicely to trees representing terms but not to graphs (some definitions have been proposed, but they work only for special types of graphs). Furthermore, it has no meaning for abstract algebraic objects. So one cannot use it at a general level.

We shall review the general properties of recognizable sets and their relationships with equational ones. The result stating that the intersection of an equational set and a recognizable one is equational is fundamental and especially useful in constructions concerning context-free graph grammars. We shall also give a general form of Parikh's Theorem, with applications to equational sets of graphs, i.e., to context-free sets of graphs.

This paper assumes a basic knowledge of Formal Language Theory; however, most proofs will be given in detail: it will be clear that they are no more difficult at an abstract level than in concrete cases. Its aim is to collect results that are easily applicable to the equational (generalized context-free) or recognizable (generalized regular) sets of finite objects like trees or graphs. Outside of the scope of this paper are the descriptions of infinite objects (for which some form of topology is needed) and those of finite and infinite ones by logical formulas.

Very few references are given in the main text. Section 7 reviews and comments the relevant references, discusses applications and extensions of the surveyed results.

1. Basic notation

We shall use $:=$ for "equal definition", i.e., for introducing a new notation, or a definition. The notation $:\Leftrightarrow$ will be used similarly for defining logical conditions.

We first review some general mathematical notation. The set of non-negative integers is denoted by $\mathbb{N}$, the set of positive ones is denoted by $\mathbb{N}_+$ and the set $\{k \in \mathbb{N} \mid i \leqslant k \leqslant j\}$ is denoted by $[i, j]$.

The cardinality of a set A is denoted by $\mathbf{card}(A)$.

If f maps $B_1 \times \cdots \times B_k$ into A and if g_i maps C into B_i for every i, then we denote by $f \circ (g_1, \ldots, g_k)$ the mapping $h: C \to A$ such that $h(x) = f(g_1(x), \ldots, g_k(x))$ for every x in C.

The powerset of a set A is denoted by $\wp(A)$. The set extension of a mapping $f: B_1 \times \cdots \times B_k \to A$ into a mapping $\wp(B_1) \times \cdots \times \wp(B_2) \to \wp(A)$ is denoted by ${}^{\wp}f$ when we need to distinguish it from f, and simply by f (as usual) otherwise; (it is defined by ${}^{\wp}f(C_1, \ldots, C_k) := \{f(d_1, \ldots, d_k) \mid d_1 \in C_1, \ldots, d_k \in C_k\}$).

The set of sequences of elements of a set A is denoted by $\mathbf{seq}(A)$ and the generic sequence is denoted by $(a_1, \ldots, a_n)$. The empty sequence is denoted by (). The concatenation of two sequences is denoted by . . The length of a sequence μ is denoted by $|\mu|$. When A is an alphabet, i.e., when its elements are letters, then a sequence $(a_1, a_2, \ldots, a_n)$ can be written unambiguously $a_1 a_2 \ldots a_n$, the empty sequence is denoted by ε, the set $\mathbf{seq}(A)$ is denoted by A^* and its elements are called *words*.

A set is *explicitely given* if it is finite and given as a list of elements. A set A is *effectively given* if it is defined by an *effective coding*, i.e., by a triple $(\| A \|, \gamma_A, \chi_A)$ consisting of a subset $\| A \|$ of $\mathbb{N}$, a bijection $\gamma_A: \| A \| \to A$ and a total recursive function $\chi_A: \mathbb{N} \to \{0, 1\}$, such that $\| A \| = \chi_A^{-1}(1)$ (which is the characteristic function of $\| A \|$). Computations on elements of A can be done by computable functions on the numbers in $\| A \|$ that code them. The mapping γ_A performs the "decoding". In many cases, like for an example when $A = B^*$ for some explicitly given alphabet B, we need no encoding, because computability on words is a well-defined concept. We introduce the notion of an effective coding in order to obtain a very general notion of computability.

An effective coding of an explicitly given set is easy to construct. Conversely, let A be effectively given. Can one compute the cardinality of A by an algorithm taking as input a finitary definition of χ_A? The answer is no, because one cannot decide the finiteness of $f^{-1}(1)$ for a total recursive mapping f. Similarly, one cannot decide the emptiness of A. Even if one knows that A is finite, one cannot compute its cardinality. However, if one knows in addition the exact number of elements, one can compute the finite set $\| A \|$. When we say: "let A be a finite set", we mean that A is explicitly given.

An alternative way of specifying effectively a set A is by an *effective presentation*, i.e., by a 4-tuple $(\| A \|, \gamma_A, \chi_A, \eta_A)$, where $\| A \|$ and χ_A are as in an effective coding, γ_A is a surjective mapping $\| A \| \to A$ and η_A is a total recursive binary mapping on $\mathbb{N}$ such that $\eta_A(x, y) = 1$ if and only if x and y belong to $\| A \|$ and $\gamma_A(x) = \gamma_A(y)$. Hence, an element of A can be represented by an integer in several ways, and, by using η_A, one can decide whether two integers represent the same element of A.

From an effective presentation $(\| A \|, \gamma_A, \chi_A, \eta_A)$ of a set A, one can construct an effective coding $(\| A \|', \gamma'_A, \chi'_A)$ of it as follows: one lets $\| A \|'$ be the set of elements x of $\| A \|$ such that for no y strictly smaller than x we have $\eta_A(x, y) = 1$; we let γ'_A be the restriction of γ_A to $\| A \|'$; we let χ'_A be the characteristic function of $\| A \|'$. Conversely, one gets easily an effective presentation from an effective coding. Hence, an effectively given set can be defined by an effective presentation as well as by an effective coding.

As an illustration, we consider effectively given quotient sets. Let A be defined by an effective coding $(\|A\|, \gamma_A, \chi_A)$. An equivalence relation $\sim$ on A is *effectively given* if for some total recursive binary mapping h on integers, we have $h(x, y) = 1$ if and only if x and y belong to $\|A\|$ and $\gamma_A(x) \sim \gamma_A(y)$. If this is the case, then the quotient set $A/\sim$ is effectively given with effective presentation $(\|A\|, \gamma'_A, \chi_A, \eta_A)$, where $\eta_A = h$, and $\gamma'_A(x)$ is, for every x in $\|A\|$, the equivalence class of $\gamma_A(x)$. Hence, the set $A/\sim$ is also effectively given.

A mapping $f: A_1 \times \cdots \times A_n \to B$ is *computable* if $A_1, \ldots, A_n$, B are effectively given and

$$f(a_1, \ldots, a_n) = \gamma_B(\|f\|(\gamma_{A_1}^{-1}(a_1), \ldots, \gamma_{A_n}^{-1}(a_n)))$$

for all $a_1 \in A_1, \ldots, a_n \in A_n$, where $\|f\|$ is a (known) total recursive mapping: $\|A_1\| \times \cdots \times \|A_n\| \to \|B\|$, and the mappings $\gamma_{A_1}, \ldots, \gamma_{A_n}$, γ_B refer to effective codings of $A_1, \ldots, A_n$, B, respectively. These notions of effectivity will be illustrated on graphs.

Example 1.1 (*Graphs*). Unless otherwise mentioned, graphs will be finite and undirected; they may have loops and multiple edges. In most of our results, we shall consider that any two isomorphic graphs are equal. However, in some constructions, for instance in the definition given below of the disjoint union of graphs, we shall need to work with "concrete" graphs. Furthermore, we want our sets of graphs to be effectively given.

Formally, we define a *concrete graph* as a pair (V, E), where V is a finite set of integers (the set of *vertices*; we use integers for having a convenient "machine" representation) and E is a finite multiset of edges where an *edge* is a set of one or two elements of V. Hence, concrete graphs form an effectively given set (we omit the formal definition of an effective representation or coding).

A *graph* is the isomorphism class of a concrete graph. Since the isomorphism of two finite concrete graphs is decidable, the set of graphs is also effectively given.

We now define the *disjoint union* $G||H$ of two graphs G and H as the isomorphism class of the concrete graph K defined as the union of two vertex disjoint (whence also edge disjoint) concrete graphs belonging, respectively, to the isomorphism classes G and H. It is quite obvious that this operation is well-defined and computable.

2. Many-sorted magmas

As in many other works, we shall use the term *magma* borrowed from Bourbaki [7] for what is usually called an *abstract algebra* or an *algebra*. The words "algebra" and "algebraic" are used in many different contexts with different meanings. We prefer to avoid them completely and use fresh words. Many-sorted notions are studied in detail by Ehrig and Mahr [26], Wirsing [46] and Wechler [45]. We mainly review the

notation. We shall use *infinite* sets of sorts and *infinite* signatures, which is not usual. For this reason, we need to pay a certain attention to effectivity questions.

2.1. Definitions

Let $\mathcal{S}$ be a set called the set of *sorts*. An *$\mathcal{S}$-signature* is a set F given with two mappings $\alpha: F \to \mathbf{seq}(\mathcal{S})$, called the *arity* mapping, and $\sigma: F \to \mathcal{S}$, called the *sort* mapping. The length of $\alpha(f)$ is called the rank of f, and is denoted by $\rho(f)$. The *type* of f in F is the pair $(\alpha(f), \sigma(f))$ that we shall rather write $s_1 \times s_2 \times \cdots \times s_n \to s$ where $\alpha(f) = (s_1, \ldots, s_n)$ and $\sigma(f) = s$. If $\mathcal{S}$ has only one sort, we say that F is a *ranked alphabet*; in this case, the arity of a symbol is completely defined by its rank.

An *F-magma* (i.e., an F-algebra in the sense of [26, 45, 46]) is an object $M = \langle (M_s)_{s \in \mathcal{S}} (f_M)_{f \in F} \rangle$, where for each s in $\mathcal{S}$, M_s is a nonempty set, called the *domain of sort s* of M, and for each $f \in F$, the object f_M is a total mapping: $M_{\alpha(f)} \to M_{\sigma(f)}$. These mappings are called the *operations* of M. (For a nonempty sequence of sorts $\mu = (s_1, \ldots, s_n)$, we let $M_\mu := M_{s_1} \times M_{s_2} \times \cdots \times M_{s_n}$). We assume that $M_s \cap M_{s'} = \emptyset$ for $s \neq s'$. We let M also denote $\bigcup \{M_s \mid s \in \mathcal{S}\}$, and for $d \in M$, we let $\sigma(d)$ denote the sort of d, i.e., the unique s such that $d \in M_s$.

We say that M is *effectively given* if $\mathcal{S}$, F, and $\bigcup \{M_s \mid s \in \mathcal{S}\}$ are effectively give, and if the mappings α, σ and the mapping associating $f_M(d_1, \ldots, d_k)$ with $(f, (d_1, \ldots, d_k))$ in $F \times \mathbf{seq}(M)$ such that $k = \rho(f)$ and $d_i \in M_{s_i}$ for all $i = 1, \ldots, k$, are computable. We shall say that M is *explicitly given* if the sets $\mathcal{S}$, F, and M_s are so and if the mappings α, σ and f_M's are given by tables.

If M and M' are two F-magmas, a homomorphism $h: M \to M'$ is a mapping h that maps M_s into M'_s for each sort s, and commutes with the operations of F in a well-known way. We shall call it an *F-homomorphism* if it is useful to specify the signature F.

Let $F' \subseteq F$ be a subsignature of F. An F'-magma M' is a *sub-F'-magma* of M (we shall denote this by $M' \subseteq M$) if $M'_s \subseteq M_s$ for each s and each $f_{M'}$ (for f in F') is the restriction of f_M to the domains of M'.

A *congruence* on M (we shall say an *F-congruence* when it will be useful to specify the relevant signature) is an equivalence relation $\approx$ on $\bigcup \{M_s \mid s \in \mathcal{S}\}$ such that:

(1) any two equivalent elements have the same sort and

(2) for every f in F and $d_1, \ldots, d_k$, $d'_1, \ldots, d'_k$ of appropriate sort, we have $f_M(d_1, \ldots, d_k) \approx f_M(d'_1, \ldots, d'_k)$ if $d_i \approx d'_i$ for every $i = 1, \ldots, k$.

The *quotient F-magma* is defined as $M/\approx := \langle (M_s/\approx)_{s \in \mathcal{S}}, (f_M/\approx)_{f \in F} \rangle$, where $(f_M/\approx)([d_1], \ldots, [d_k]) = [f_M(d_1, \ldots, d_k)]$. (We denote by $[d]$ the equivalence class of an element d of M.) If M and $\approx$ are effectively given, then $M/\approx$ is effectively given. (See Section 1.)

We denote by $\mathbf{T}(F)$ the *initial F-magma*, and by $\mathbf{T}(F)_s$ its domain of sort s. The set $\mathbf{T}(F)_s$ can be identified with the set of well-formed ground terms over F which are of sort s. (The sort of a term is that of its first symbol in prefix notation). $\mathbf{T}(F)$ is

effectively given if $\mathcal{S}$ and F are effectively given and if the mappings α and σ are computable.

If M is an F-magma, we denote by $\mathbf{h}_M$ the unique homomorphism: $\mathbf{T}(F) \to M$. If $t \in \mathbf{T}(F)_s$, then the image of t under $\mathbf{h}_M$ is an element of M_s, also denoted by t_M. One can consider t as a term *denoting* t_M, and t_M as the *value of* t in M. We say that a subset of M is *generated* by F if each of its elements is the value of some term in $\mathbf{T}(F)$.

If M is effectively given, then $\mathbf{h}_M$ is computable. If, furthermore, M is generated by F, then a computable mapping $\mathbf{k}_M : M \to \mathbf{T}(F)$ that produces, for every element of M a term denoting it, can be defined by the following algorithm: given d in M, one enumerates $\mathbf{T}(F)$ in an effective way and for every term t in $\mathbf{T}(F)$ one computes t_M. The term $\mathbf{k}_M(d)$ is the first one such that $t_M = d$.

An *$\mathcal{S}$-sorted set of variables* is a pair (X, σ) consisting of a set X, and a *sort mapping* $\sigma : X \to \mathcal{S}$. It will be more simply denoted by X, unless the sort mapping must be specified. We shall denote by $\mathbf{T}(F, X)$ the set of well-formed terms written with $F \cup X$ and by $\mathbf{T}(F, X)_s$ the subset of those of sort s. Hence, $\mathbf{T}(F, X) = \mathbf{T}(F \cup X)$ and $\mathbf{T}(F, X)_s = \mathbf{T}(F \cup X)_s$. However, the notations $\mathbf{T}(F, X)$ and $\mathbf{T}(F, X)_s$ are useful because they specify the variables among the nullary symbols of $F \cup X$. The *sequence* of variables of a term $t \in \mathbf{T}(F, X)$ is defined as follows:

$$\begin{aligned}
&\mathbf{var}(t) = (x) && \text{if } t = x \in X\\
&\mathbf{var}(t) = () && \text{if } t = f \in F \text{ (and } \rho(f) = 0),\\
&\mathbf{var}(t) = \mathbf{var}(t_1).\mathbf{var}(t_2).\ldots.\mathbf{var}(t_k) && \text{if } t = f(t_1, \ldots, t_k).
\end{aligned}$$

A term t is *linear* if each variable has at most one occurrence in $\mathbf{var}(t)$.

Let χ be a finite sequence of pairwise distinct variables from X. We shall denote by $\mathbf{T}(F, \chi)_s$ the set of terms in $\mathbf{T}(F, X)_s$, having all their variables in the sequence χ. If $t \in \mathbf{T}(F, \chi)_s$, we denote by $t_{M,\chi}$ the mapping: $M_{\sigma(\chi)} \to M_s$ associated with t in the obvious way, by letting a symbol f from F denote f_M, (where $\sigma(\chi)$ denotes the sequence of sorts of the elements of the sequence χ). We call $t_{M,\chi}$ a *derived operation of* M. If χ is known from the context, we write t_M instead of $t_{M,\chi}$. This is the case in particular if t is defined as a member of $\mathbf{T}(F, \{x_1, \ldots, x_k\})_s$: the sequence χ is implicitly $(x_1, \ldots, x_k)$.

We now review *first-order substitution*. If θ is a sort preserving mapping from X to $\mathbf{T}(F, X)$, we let θ^* denote the mapping that associates with a term t in $\mathbf{T}(F, X)$ the result of the simultaneous substitution in t of $\theta(x)$ for every x in X. It is the unique F-homomorphism of $\mathbf{T}(F, X)$ into $\mathbf{T}(F, X)$ extending θ. Such an F-homomorphism is called a *first-order substitution*. If $t \in \mathbf{T}(F, X)$, if $x_1, \ldots, x_k$ are pairwise distinct variables in X, if $t_1, \ldots, t_k \in \mathbf{T}(F, X)$ and $\sigma(t_i) = \sigma(x_i)$ for $i = 1, \ldots, k$, then we shall denote by $t[t_1/x_1, \ldots, t_k/x_k]$ the term $\theta^*(t)$ where $\theta(x_i) = t_i$ and $\theta(y) = y$ for every y in X different of the x_i's. We shall also use the notation $t[t_1, \ldots, t_k]$ if the sequence $x_1, \ldots, x_k$ is clear from the context.

For $s, r \in \mathcal{S}$, we denote by $\mathbf{ctxt}(F)_{s,r}$ the set of elements of $\mathbf{T}(F, \{u\})_r$ having one and only one occurrence of u, where u is a variable of sort s. If $c \in \mathbf{ctxt}(F)_{s,r}$ and $t \in \mathbf{T}(F, X)_s$ then $c[t] := c[t/u]$ is an element t' of $\mathbf{T}(F, X)_r$. We say that c is

a context of t in t'. The specific variable u is somewhat irrelevant, and the notations $c[t]$ and $\mathbf{ctxt}(F)_{s,r}$ avoid mentioning it explicitly.

Every term t can be written in a unique way as $\mathbf{lin}(t)[x_{i_1},\ldots,x_{i_p}]$ where $\mathbf{var}(t)=(x_{i_1},\ldots,x_{i_p})$ and $\mathbf{lin}(t)$ is a linear term in $\mathbf{T}(F,\{y_1,\ldots,y_p\})$ such that $\mathbf{var}(\mathbf{lin}(t))=(y_1,\ldots,y_p)$. For example, if $t=f(a,g(x,x),g(y,z))$, where x,y,z are the variables of t, then $t=\mathbf{lin}(t)[x,x,y,z]$, where $\mathbf{lin}(t)=f(a,g(y_1,y_2),g(y_3,y_4))$. The specific variables used in $\mathbf{lin}(t)$ are actually irrelevant: we could also take $\mathbf{lin}(t)=f(a,g(z_1,z_2),g(z_3,z_4))$. The claimed unicity of $\mathbf{lin}(t)$ holds up to renaming of the variables.

Here is the semantical meaning of first-order substitution. If $t\in T(F,\chi)_s$, where $\chi=(x_1,\ldots,x_k)$, if $t_1,\ldots,t_k\in\mathbf{T}(F,\chi')$ and $\sigma(t_i)=\sigma(x_i)$ for $i=1,\ldots,k$, then, for every F-magma M, we have

$$t[t_1/x_1,\ldots,t_k/x_k]_{M,\chi'}=t_{1M,\chi}\circ(t_{M,\chi'},\ldots,t_{kM,\chi'}). \tag{2.1}$$

In particular, if $c\in\mathbf{ctxt}(F)_{s,r}$, then c_M is a mapping $M_s\to M_r$ and $c[t]_M=c_M\circ t_M$.

When writing terms, we shall use the prefix notation with parentheses and commas, but we shall frequently omit the parentheses surrounding the unique argument of a unary function symbol. Hence we shall use the simplified notation $fgfh(x,fx)$ for $f(g(f(h(x,f(x)))))$. For a binary associative operation, we shall use infix notation and omit parenthesis.

Example 2.1 (*Monoids of words and traces*). Let A be a finite alphabet, say $A=\{a_1,\ldots,a_n\}$. Let $F_A=A\cup\{.,\varepsilon\}$ be the ranked alphabet where $\rho(a_i)=0$ for all i, $\rho(\varepsilon)=0$, $\rho(.)=2$. We denote by $\mathbb{W}_A$ the F_A-magma $\langle A^*,.,\varepsilon,a_1,\ldots,a_n\rangle$, where A^* is the set of words over A, $.$ is the concatenation, ε is the empty word, $a_1,\ldots,a_n$ denote themselves as words. It is a monoid (with a binary associative operation having a unit) augmented with constants.

We now let R be a set of pairs of the form (a_ia_j,a_ja_j) for i,j with $1\leqslant i<j\leqslant n$. We let $\equiv$ be the least congruence on A^* containing R (also denoted by $\overset{*}{\underset{R}{\leftrightarrow}}$ if one considers R as Thue system). We denote by $\mathbb{W}_{A,R}$ the F_A-magma $\langle A^*/\equiv,.\equiv,[\varepsilon],[a_1],\ldots,[a_n]\rangle$ where $[x]$ denotes the equivalence class of x with respect to $\equiv$. It is called a monoid of *traces*.

Example 2.2 (*The unary magma of words*). We let A be as above and $F'_A=\{\varepsilon,a_1,\ldots,a_n\}$ be the ranked alphabet such that $\rho(\varepsilon)=0$ and $\rho(a_i)=1$ for $i=1,\ldots,n$. We denote by $\mathbb{U}_A$ the F'_A-magma $\langle A^*,\varepsilon,\bar{a}_1,\ldots,\bar{a}_n\rangle$, where ε is the empty word and $\bar{a}_i$ is the mapping: $A^*\to A^*$ such that $\bar{a}_i(u)=u.a_i$ for every $u\in A^*$ and $i=1,\ldots,n$. Hence, $\mathbb{U}_A$ is another algebraic structure on the set of words. The operation $\bar{a}_i$ is a derived operation of $\mathbb{W}_A$. We shall say (see Section 2.2) that $\mathbb{U}_A$ is derived from $\mathbb{W}_A$. However, the concatenation *is not* a derived operation of $\mathbb{U}_A$. The structure $\mathbb{W}_A$ is some sense *strictly richer* than $\mathbb{U}_A$.

Example 2.3 (*Trees*). A tree T is a finite connected undirected graph without multiple edges and cycles. The set of trees is denoted by $\mathbb{T}$. A *rooted tree* is a pair $R = (T, r)$ consisting of a tree T and a distinguished node r called the *root*. The set of rooted trees is denoted by $\mathbb{R}$. Any two isomorphic trees (or rooted trees) are considered as equal. (See for more details the definitions of graphs and concrete graphs in Example 1.1.) The sets $\mathbb{R}$ and $\mathbb{T}$ are effectively given.

We now define a few operations on trees and rooted trees. The types of these operations will be given in terms of two sorts, **t** and **r**, namely, the sort **t** of trees and the sort **r** of rooted trees.

The first operation is the *root-gluing* $||: \mathbf{r} \times \mathbf{r} \to \mathbf{r}$. For S and T in $\mathbb{R}$, we let $S || T$ be the rooted tree obtained by fusing the roots of S and T (or rather, of two disjoint isomorphic concrete copies of S and T). The second operation is the *extension* $\mathbf{ext}: \mathbf{r} \to \mathbf{r}$. For T in $\mathbb{R}$, we let $\mathbf{ext}(T)$ be the rooted tree obtained from T by the addition of a new node that becomes the root of $\mathbf{ext}(T)$, linked by a new edge to the root of T. We denote by **1** the rooted tree reduced to a single node (the root). Finally, we let $\mathbf{fg}: \mathbf{r} \to \mathbf{t}$ be the mapping that "forgets" the root of a rooted tree R. Formally, $\mathbf{fg}(R) = T$, where $R = (T, r) \in \mathbb{R}$.

Hence, we have an $\{\mathbf{r}, \mathbf{t}\}$-sorted signature $F := \{||, \mathbf{ext}, \mathbf{1}, \mathbf{fg}\}$ and a many-sorted F-magma $\mathbb{TREE}$ having $\mathbb{R}$ as domain of sort **r**, $\mathbb{T}$ as domain of sort **t**, and the operations defined above. Hence $\mathbb{TREE} = \langle \mathbb{R}, \mathbb{T}, ||, \mathbf{ext}, \mathbf{1}, \mathbf{fg} \rangle$.

Example 2.4 (*Graphs with sources*). Let $k \in \mathbb{N}$. A *k-graph* is a pair consisting of a graph and a sequence of k pairwise distinct distinguished vertices called its *sources*. A 0-graph has no source and is nothing but a graph. We let $\mathbb{G}_k$ denote the set of k-graphs. (We do not repeat the formal distinction between a k-graph and a concrete k-graph; the details are easy to provide, see Example 1.1.) We define some operations on k-graphs for $k \in \mathbb{N}$. We shall obtain thus a many-sorted magma $\mathbb{G}$ with infinitely many domains $\mathbb{G}_k$, for $k \in \mathbb{N}$.

We first define the *parallel composition* $G ||_k H$ of a k-graph G and a k-graph H. This operation produces a k-graph K defined as the isomorphism class of a concrete k-graph K' that is constructed as follows: one takes the union of two disjoint concrete k-graphs G' and H', respectively, isomorphic to G and to H, one fuses the ith source of G' with the ith source of H' for every $i \in [1, k]$ and one takes the sources of H (after fusion with sources of G) as the sources of K'. We shall frequently omit the subscript k in $||_k$.

Note that $||_k$ is associative and commutative and that $||_0$ is just the disjoint union of graphs without sources (see Example 1.1). Note also that the set of rooted trees $\mathbb{R}$ is a subset of $\mathbb{G}_1$ and that the root-gluing operation on trees $||$ is the restriction of $||_1$ to $\mathbb{R}$.

We let $\mathbf{fg}_k: \mathbb{G}_k \to \mathbb{G}_{k-1}$ be the *source-forgetting* operation such that

$$\mathbf{fg}_k(G, s_1, s_2, \ldots, s_k) = (G, s_1, s_2, \ldots, s_{k-1}).$$

Clearly $\mathbb{T} \subseteq \mathbb{G}_0$ and the operation **fg** is the restriction of $\mathbf{fg}_1$ to $\mathbb{R}$. We shall frequently omit the subscript k in $\mathbf{fg}_k$. We let also $\mathbf{i}_k: \mathbb{G}_k \to \mathbb{G}_{k+1}$ be the following

mapping: $\mathbf{i}_k(G, s_1, s_2, \ldots, s_k) = (G \cup \{v\}, s_1, \ldots, s_k, v)$ where v is added to G as a new isolated vertex. For every $k \geqslant 2$, for every nonidentity permutation π of $[1, k]$, we let $\mathbf{perm}_\pi : \mathbb{G}_k \to \mathbb{G}_k$ be such that

$$\mathbf{perm}_\pi(G, s_1, s_2, \ldots, s_k) = (G, s_{\pi(1)}, \ldots, s_{\pi(k)}).$$

Finally, we shall use the following nullary symbols: $\mathbf{1}$ denoting an isolated vertex that is a source, ℓ denoting a loop on a single vertex that is the unique source, e denoting an edge with two ends that are the sources. (Hence $\mathbf{1}, \ell$ are of sort 1 and e is of sort 2.) We let

$$F = \{||_k \mid k \geqslant 0\} \cup \{\mathbf{fg}_k \mid k \geqslant 1\} \cup \{\mathbf{i}_k \mid k \geqslant 0\}$$

$$\cup \{\mathbf{perm}_\pi \mid k \geqslant 2,\ \pi \text{ is a permutation of } [1, k]\} \cup \{\mathbf{1}, \ell, e\}.$$

We obtain thus an F-magma of graphs $\mathbb{G}$ with infinitely many sorts. It is not hard to see that F generates $\mathbb{G}$. Let us also note that the mapping $\mathbf{ext} : \mathbb{R} \to \mathbb{R}$ can be expressed as follows:

$$\mathbf{ext}(G) = \mathbf{fg}_2(\mathbf{perm}_\pi(\mathbf{i}_1(G)||e)),$$

where π exchanges 1 and 2. The *series-composition* of graphs of type 2 is the mapping $\bullet : \mathbb{G}_2 \times \mathbb{G}_2 \to \mathbb{G}_2$ defined as follows:

$$G \bullet H = \mathbf{fg}_3(\mathbf{perm}_\alpha(\mathbf{i}_2(G))||\mathbf{perm}_\beta(\mathbf{i}_2(H))),$$

where α exchanges 2 and 3 and β exchanges 1 and 3. Thus $G \bullet H$ is constructed from the union of disjoint concrete copies of G and H by the fusion of the second source of G with the first one of H. The first source of $G \bullet H$ is the first source of G, and its second source is the second source of H. Series composition is a quite natural notion of concatenation of graphs.

Finally, we observe that the F-magma $\mathbb{G}$ is effectively given: this follows from obvious extensions of the remarks made in Example 1.1.

2.2. Derived signatures

Let F be an $\mathcal{S}$-signature. A *derived signature of* F is a pair (G, δ) consisting of an $\mathcal{S}'$-signature G and a mapping δ satisfying the following conditions:

(1) $\mathcal{S}' \subseteq \mathcal{S}$,

(2) δ associates with every symbol g in G of type of $s_1 \times s_2 \times \cdots \times s_n \to s$ a term $\delta(g)$ in $\mathbf{T}(F, \chi)_s$ for some χ such that $\sigma(\chi) = (s_1, \ldots, s_n)$.

We shall frequently let G stand for (G, δ) and let δ_G denote the mapping δ. We shall say that G is *linearly derived of* F if each term $\delta_G(g)$ is linear, i.e., has at most one occurrence of each variable.

A G-magma P is a *derived G-magma of an F-magma M* if G is a derived signature of F and the following conditions hold:

(1) $P_s = M_s$ for every s in $\mathscr{S}$,

(2) $g_P = \delta_G(g)_M$ for every g in G.

Furthermore, we shall say that P *is linearly derived of* M if G is linearly derived of F.

The notion of *second-order substitution* that we now define makes it possible to translate every derived operation of a derived magma P of M into a derived operation of M itself. Let (G, δ) be a derived signature of F. We let $\underline{\delta}$ be the mapping from $\mathbf{T}(G, X)$ into $\mathbf{T}(F, X)$ defined as follows:

$$\underline{\delta}(x) = x \text{ for } x \text{ in } X,$$

$$\underline{\delta}(g(t_1, \ldots, t_k)) = \delta(g)[\underline{\delta}(t_1), \ldots, \underline{\delta}(t_k)].$$

Proposition 2.5. *Let P be a G-derived magma of M and $\delta = \delta_G$. For every term t in $\mathbf{T}(G, X)$, we have $t_P = (\underline{\delta}(t))_M$.*

The proof is easy by induction on the structure of t.

2.3. Equational properties and term rewriting systems

This topic is treated in detail by Wechler [45], Klop [34], and Dershowitz and Jouannaud [25]. We only review the basic definitions and facts. Let F be an $\mathscr{S}$-signature. A *term rewriting system over F* is a set R of pairs of terms (possibly with variables) of the same sort. Hence,

$$R \subseteq \bigcup \{\mathbf{T}(F, X)_s \times \mathbf{T}(F, X)_s \mid s \in \mathscr{S}\}, \text{ where } X \text{ is an } \mathscr{S}\text{-sorted set of variables.}$$

A pair (t, t') is called a (*rewriting*) *rule* and is denoted by $t \to t'$. We say that R is *ground* if $X = \emptyset$. The *one-step rewriting relation* on $\mathbf{T}(F, Y)$ associated with R is defined as follows (where Y is any set of variables): $w \underset{R}{\to} w'$ if and only if $w, w' \in \mathbf{T}(F, Y)$ and

$$w = c[t[u_1/x_1, \ldots, u_k/x_k]]$$

$$w' = c[t'[u_1/x_1, \ldots, u_k/x_k]]$$

for some $c \in \mathbf{ctxt}(F)_{s,r}$, some rule $t \to t'$ in R with $t, t' \in \mathbf{T}(F, \{x_1, \ldots, x_k\})_s$ and some terms $u_1, \ldots, u_k \in \mathbf{T}(F, Y)$ of respective sorts $\sigma(x_1), \ldots, \sigma(x_k)$.

These conditions imply that w and w' are both of sort r.

The n-fold composition of $\underset{R}{\to}$ is denoted by $\overset{n}{\underset{R}{\to}}$, its transitive closure by $\overset{+}{\underset{R}{\to}}$, its reflexive and transitive closure by $\overset{*}{\underset{R}{\to}}$. Its symmetric closure is denoted by $\underset{R}{\leftrightarrow}$ ($\underset{R}{\leftrightarrow}$ is the one step rewriting relation associated with $R \cup R^{-1}$); the definitions of $\overset{n}{\underset{R}{\leftrightarrow}}$, $\overset{+}{\underset{R}{\leftrightarrow}}$ and $\overset{*}{\underset{R}{\leftrightarrow}}$ follow immediately.

Lemma 2.6. *If* $t_i \xrightarrow[R]{*} t_i'$ *for every* $i = 0, 1, \ldots, k$ *then* $t_0[t_1/x_1, \ldots, t_k/x_k] \xrightarrow[R]{*} t_0'[t_1'/x_1, \ldots, t_k'/x_k]$.

Since $\overset{*}{\underset{R}{\leftrightarrow}} = \xrightarrow[S]{*}$, where $S = R \cup R^{-1}$, the statement of Lemma 2.6 also holds with $\overset{*}{\underset{R}{\leftrightarrow}}$ instead of $\xrightarrow[R]{*}$. The relation $\overset{*}{\underset{R}{\leftrightarrow}}$ is the least equivalence relation on $\mathbf{T}(F, Y)$ that contains R, is an F-congruence and a congruence for first-order substitution. (An equivalence relation $\sim$ on the set $\mathbf{T}(F, Y)$ is *a congruence for first-order substitution* if for every t, $t', u_1, \ldots, u_k \in \mathbf{T}(F, Y)$, $x_1, \ldots, x_k \in Y$, it holds that $t \sim t'$ implies that $t[u_1/x_1, \ldots, u_k/x_k] \sim t'[u_1/x_1, \ldots, u_k/x_k]$.)

For every rewriting system R, we let $\mathbf{E}(R)$ denote the *set of equalities associated with* R, namely $\mathbf{E}(R) := \{t = t' \mid t \to t' \in R\}$. We say that an equality $t = t'$ is *valid in* M if $t_{M,\chi} = t'_{M,\chi}$ (where χ is such that t and t' belong both to $\mathbf{T}(F, \chi)$), i.e., if t and t' define the same derived operation of M (or the same value if t and t' have no variable). A set of equalities E is *valid in* M if each equality of this set is valid; we shall also say that M is a *model* of E. Every set of equalities between terms of the same sort is of the form $\mathbf{E}(R)$ for some rewriting system R: to obtain R it suffices to make each equality $t = t'$ into a rule, either $t \to t'$ or $t' \to t$.

Proposition 2.7. *Let* R *be a rewriting system over* F. *Let* M *be a model of* $\mathbf{E}(R)$. *If* $t \xrightarrow[R]{*} t'$ *then the equality* $t = t'$ *is valid in* M.

Proof. It follows from Eq. (2.1) and the definitions that $t = t'$ is valid in M whenever $t \xrightarrow[R]{} t'$. The general case follows by transitivity. □

The *equational theory of* $\mathbf{E}(R)$ is the set of equalities that are valid in every model of $\mathbf{E}(R)$. The following Completeness Theorem [6] is a kind of converse of Proposition 2.7.

Theorem 2.8. *An equality* $t = t'$ *belongs to the equational theory of* $\mathbf{E}(R)$ *if and only if* $t \overset{*}{\underset{R}{\leftrightarrow}} t'$.

The *word problem for* R, namely the problem of deciding whether $t \overset{*}{\underset{R}{\leftrightarrow}} t'$ is undecidable in general, even for certain fixed finite systems R. However, it is decidable for certain others. The investigation of the border between decidable cases and undecidable ones is one of the main aims of the theory of term rewriting systems. We refer the reader to [25, 34, 45] on this theory.

Let us say a few words on the use of rewriting systems for characterizing the properties of an F-magma of interest, say M. We shall assume that M is generated by F which means that every element is denoted by a term.

Question 2.9. *Can one characterize in terms of rewriting systems on* $\mathbf{T}(F)$ *the equality in* M?

If we know a rewriting system R over F such that $\mathbf{E}(R)$ is valid in M, then we obtain that any two $\overset{*}{\underset{R}{\leftrightarrow}}$-equivalent terms denote the same element of M. Let us denote by $\mathbf{M}(R)$ the quotient F-magma $\mathbf{T}(F)/\overset{*}{\underset{R}{\leftrightarrow}}$. The homomorphism $\mathbf{h}_M$ factors through $\mathbf{M}(R)$ in a unique way as $h \circ \mathbf{h}_{M(R)}$, where h is an F-homomorphism: $\mathbf{M}(R) \to M$. Since $\mathbf{h}_M$ is surjective (because F generates M), so is h. A desirable situation is when h is injective, because then, M is isomorphic to $\mathbf{M}(R)$ by h, and any two terms in $\mathbf{T}(F)$ are $\overset{*}{\underset{R}{\leftrightarrow}}$-equivalent *if and only if* they denote the same element of M.

Question 2.10. *Can one characterize in terms of rewriting systems the equality of two derived operations of M?*

If M is a model of $\mathbf{E}(R)$, then any two $\overset{*}{\underset{R}{\leftrightarrow}}$-equivalent terms in $\mathbf{T}(F, X)$ define the same derived operations in M. However, even if M is isomorphic to $\mathbf{M}(R)$, it is *not* always the case that two terms are $\overset{*}{\underset{R}{\leftrightarrow}}$-equivalent if they denote the same derived operation of M.

The *inductive theory* of $\mathbf{E}(R)$ is the set of equalities $t = t'$ that are valid in $\mathbf{M}(R)$. It is in general larger than the equational theory of $\mathbf{E}(R)$ and can be strictly larger as shown by the following example.

Example 2.11. We consider the $\{\mathbf{0}, s, +\}$-magma $\langle \mathbb{N}, 0, s, + \rangle$ also denoted by $\mathbb{N}$, where s is the successor function. Let R consist of the two rewriting rules $x + \mathbf{0} \to x$, and $x + s(y) \to s(x + y)$. Then $\mathbb{N}$ is isomorphic to $\mathbf{M}(R)$. However, the equality $x + y = y + x$ is valid in $\mathbf{M}(R)$ but its two handsides are not $\overset{*}{\underset{R}{\leftrightarrow}}$-equivalent. The inductive theory of $\mathbf{E}(R)$ is strictly larger than its equational theory.

Example 2.12. We consider here the monoid of words $\mathbb{W}_A$ of Example 2.1. Let R consist of the rules $x.(y.z) \to (x.y).z$, $x.\varepsilon \to x$ and $\varepsilon.x \to x$. The corresponding equalities are valid and furthermore, $\mathbb{W}_A$ is isomorphic to $\mathbf{M}(R)$. The inductive theory of $\mathbf{E}(R)$ is in this case equal to its equational theory.

In the case of the unary F'_A-magma of words $\mathbb{U}_A$ considered in Example 2.2, we have an isomorphism of $\mathbb{U}_A$ onto $\mathbf{M}(\emptyset)$, i.e., onto $\mathbf{T}(\mathbf{F}'_A)$.

Example 2.13. We now consider the magma $\mathbb{TREE}$ of rooted and unrooted trees introduced in Example 2.3. Let $\mathscr{E}$ denote the following set of equalities:

$(\mathscr{E}_1)$ $x||y = y||x$,

$(\mathscr{E}_2)$ $(x||y)||z = x||(y||z)$,

$(\mathscr{E}_2)$ $x||\mathbf{1} = x$,

$(\mathscr{E}_4)$ $\mathbf{fg}(x||\mathbf{ext}(y)) = \mathbf{fg}(\mathbf{ext}(x)||y)$,

where x, y, z are variables of sort **r**, intended to denote rooted trees. It is clear from the definitions that these properties are valid in $\mathbb{TREE}$. Letting R be a rewriting system such that $\mathbf{E}(R) = \mathscr{E}$, we have an isomorphism of $\mathbf{M}(R)$ and $\mathbb{TREE}$ (see [18] for the proof). The inductive theory of $\mathscr{E}$ is in this case equal to its equational theory.

3. Polynomial systems and equational sets

Polynomial systems have been introduced (under the simpler name of "systems") by Mezei and Wright [38]. The qualification of "polynomial" refers to the use of set union, denoted by $+$, and distinguishes these systems from the more general "regular" systems. See Courcelle [10] for a thorough study of the systems of both kinds. Polynomial systems have least solutions, called the equational sets that can be seen as generalized context-free sets.

3.1. Definitions

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

$$\wp(M) := \langle (\wp(M_s))_{s \in \mathcal{S}}, (f_{\wp(M)})_{f \in F_+} \rangle,$$

where

$$\Omega_{s\,\wp(M)} = \emptyset,$$

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

and

$$f_{\wp(M)}(A_1, \ldots, A_k) := {}^{\wp}f_M(A_1, \ldots, A_k) \text{ i.e.,}$$

$$:= \{ f_M(a_1, \ldots, a_k) \mid a_1 \in A_1, \ldots, a_k \in A_k \}$$

$$\text{for } A_1 \subseteq M_{s_1}, \ldots, A_k \subseteq M_{s_k}, \text{ where } \alpha(f) = (s_1, \ldots, s_k).$$

A *polynomial system* over F is a sequence of equations $S = \langle u_1 = p_1, \ldots, u_n = p_n \rangle$, where $U = \{u_1, \ldots, u_n\}$ is an $\mathcal{S}$-sorted set of variables called the set of *unknowns* of S. Equivalently, one can define S as a set of equations, and equip U with some linear ordering. In both cases, the set of unknowns is linearly ordered. Each right-hand side of an equation p_i is a *polynomial*, i.e., a term of the form Ω_s or $t_1 +_s t_2 +_s \cdots +_s t_m$, where the terms t_j are monomials of sort $s = \sigma(u_i)$. A *monomial* is a term in $\mathrm{T}(F \cup U)$. The subscript s is usually omitted in $+_s$ and in Ω_s.

A mapping $S_{\wp(M)}$ of $\wp(M_{\sigma(u_1)}) \times \cdots \times \wp(M_{\sigma(u_n)})$ into itself is associated with S and M as follows: for $A_1 \subseteq M_{\sigma(u_1)}, \ldots, A_n \subseteq M_{\sigma(u_n)}$, we let

$$S_{\wp(M)}(A_1, \ldots, A_n) = (p_{1\,\wp(M)}(A_1, \ldots, A_n), \ldots, p_{n\wp(M)}(A_1, \ldots, A_n)).$$

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

$$A_i = p_{i\wp(M)}(A_1,\ldots,A_n) \quad \text{for every } i=1,\ldots,n. \tag{3.1}$$

A solution of S is also called a fixed-point of $S_{\wp(M)}$. By the well-known Kleene's fixed-point lemma, (see [35]), every such system S has a least solution in $\wp(M)$ denoted by $(\mathbf{L}((S,M),u_1),\ldots,\mathbf{L}((S,M),u_n))$. ("Least" is understood with respect to set inclusion.) This n-tuple can be concretely described as follows:

$$\mathbf{L}((S,M),u_i)=\bigcup\{A_i^j \mid j\geqslant 0\},$$

where $A_i^0=\emptyset$ for all $i=1,\ldots,n$, and $(A_1^{j+1},\ldots,A_n^{j+1})=S_{\wp(M)}(A_1^j,\ldots,A_n^j)$. The components of the least solution in $\wp(M)$ of a polynomial system are the *M-equational* sets. We denote by $\mathbf{Equat}(M)$ the family of M-equational sets.

A *quasi-solution* of S in $\wp(M)$ is an-tuple $(A_1,\ldots,A_n)$ such that

$$A_i \supseteq p_{i\wp(M)}(A_1,\ldots,A_n) \quad \text{for every } i=1,\ldots,n. \tag{3.2}$$

(Note that the equalities in (3.1) are replaced in (3.2) by inclusions.) The least solution of S in $\wp(M)$ is also its least quasi-solution.

Example 3.1. We consider again the monoid of words $\mathbb{W}_A$ introduced in Example 2.1. $\mathbf{Equat}(\mathbb{W}_A)$ is the set of *context-free languages* over A by the theorem of Ginsburg and Rice [30] that characterizes these languages as the components of the least solutions of systems of recursive equations written with the nullary symbols ε, $a_1,\ldots,a_n$, the concatenation and, of course, set union (denoted here by $+$). Take, for an example, the context-free grammar $G=\{u\to auuv, u\to avb, v\to avb, v\to ab\}$ with nonterminal symbols u and v and terminal symbols a and b. The corresponding system of equations is

$$S=\langle u=a.(u.(u.v))+a.(v.b),\quad v=a.(v.b)+a.b\rangle.$$

The set $\mathbf{Equat}(\mathbb{U}_A)$, where $\mathbb{U}_A$ is the unary magma of words introduced in Example 2.2 is the set of components of least solutions of systems of left-linear equations hence is the set of *regular languages*. The system corresponding to the left-linear grammar $H=\{u\to ua, u\to vb, v\to vb, v\to a, v\to\varepsilon\}$ is thus:

$$S=\langle u=\bar{a}(u)+\bar{b}(v),\ v=\bar{b}(v)+\bar{a}(\varepsilon)+\varepsilon\rangle.$$

We shall give below a general result (Proposition 3.17) that establishes (and extends) the validity of this correspondence between grammars and polynomial systems.

Example 3.2. The set of trees of odd degree (i.e., such that the degree of every node is odd) is defined as $\mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), u)$, where S is the system:

$$S\begin{cases} u = \mathbf{fg}(w), \\ v = w||w + v||w||w, \\ w = \mathbf{ext}(v) + \mathbf{ext}(\mathbf{1}). \end{cases}$$

The sorts of u, v and w are **t**, **r** and **r**. (See Example 2.3 for the definition of $\mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}$). We claim that $\mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), u)$ is the set L of (finite) trees of odd degree, and we shall indicate how this claim can be proved. We let L' denote the set of rooted trees different from $\mathbf{1}$, all nodes of which except the root have odd degree. We let L'' be the set of rooted trees such that all nodes have odd degree and the root has degree one.

Fact. *The triple* (L, L', L'') *is a quasi-solution of the system* S.

This means that we have

$$L \supseteq \mathbf{fg}(L''),$$

$$L' \supseteq L''||L'' \cup L'||L''||L'',$$

$$L'' \supseteq \mathbf{ext}(L') \cup \mathbf{ext}(\mathbf{1}),$$

which is actually easy to verify from the definitions of L, L' and L''. Since the least solution of a system is also its least quasi-solution, it follows that

$$\mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), u) \subseteq L,$$

$$\mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), v) \subseteq L',$$

$$\mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), w) \subseteq L''.$$

In order to prove the opposite inclusions, one can prove by induction on the size of an element t in $\mathbb{T} \cup \mathbb{R}$ that

$$\text{if } t \in L \quad \text{then } t \in \mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), u),$$

$$\text{if } t \in L' \quad \text{then } t \in \mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), v),$$

$$\text{if } t \in L'' \quad \text{then } t \in \mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), w),$$

which is also not difficult. This establishes that $\mathbf{L}((S, \mathbb{T}\mathbb{R}\mathbb{E}\mathbb{E}), u)$ is equal to L, i.e., is the set of trees of odd degree.

Example 3.3. *Series-parallel graphs with two sources.*
The equation

$$u = u||_2 u + u \bullet u + e, \tag{SP1}$$

where $||_2$ *is the parallel-composition and* $\bullet$ is the series-composition of graphs of type 2 (see the definitions in Example 2.4). An equivalent writing of this equation is

$$u = u||_2 u + \mathbf{fg}_3(\mathbf{perm}_\alpha(\mathbf{i}_2(u))||_3 \mathbf{perm}_\beta(\mathbf{i}_2(u))) + e, \tag{SP2}$$

where α and β are appropriate permutations (see Example 2.4). However, solving (SP2) necessitates to handle six graph operations of two sorts. The equation (SP1) is built with only two operations and one sort; it can be solved in $\wp(\langle \mathbb{G}_2, ||_2, \bullet, e\rangle)$ whereas (SP2) uses a more complicated magma. That (SP1) and (SP2) are indeed equivalent will be seen in Section 3.4.

3.2. *Unicity of solutions*

We shall consider some sufficient conditions insuring that a polynomial system has a unique solution in some given $\wp(M)$. Let $S = \langle u_i = p_i; 1 \leqslant i \leqslant n\rangle$ be a polynomial system. Let $m \in \mathbf{T}(F, U)$ be a monomial of some of the polynomials p_i. Let us write it $m = \mathbf{lin}(m)[u_{i_1}, \ldots, u_{i_k}]$ (see Section 2) and let f be the mapping $\mathbf{lin}(m)_M : M^k \to M$. We let $d \to_m d'$ if $d = f(d_1, \ldots, d_k)$ for some $d_1, \ldots, d_k \in M$ and $d' = d_i$ for some $1 \leqslant i \leqslant k$. We let $d \to_S d'$ if and only if $d \to_m d'$ for some monomial m occurring in some right-hand side of S. We shall say that M is *well-founded with respect to* S if there is no infinite sequence: $d_1 \to_S d_2 \to_S d_3 \to_S \cdots \to_S d_k \to_S \cdots$

Proposition 3.4. *Let* M *be well-founded with respect to a polynomial system* S *with unknowns* $u_1, \ldots, u_n$. *Then* $(\mathbf{L}((S, M), u_1), \ldots, \mathbf{L}((S, M), u_n))$ *is the unique solution of* S *in* $\wp(M)$.

Proof. Let $(A_1, \ldots, A_n) = (\mathbf{L}((S, M), u_1), \ldots, \mathbf{L}((S, M), u_n))$ be the least solution of S and let $(B_1, \ldots, B_n)$ be an arbitrary solution. We have $A_i \subseteq B_i$ for every $i = 1, \ldots, n$. Let us assume that $\bigcup\{(B_i - A_i) | 1 \leqslant i \leqslant n\}$ is nonempty and let d belong to $B_i - A_i$ for some i. Since $(B_1, \ldots, B_n)$ is a solution we have $d \in \mathbf{lin}(m)_M(d_1, \ldots, d_k)$ for some monomial m on the right-hand side of the equation $u_i = p_i$. Let $(u_{i_1}, \ldots, u_{i_k})$ be the sequence of unknowns of m; we have $d_j \in B_{i_j}$ for every $j = 1, \ldots, k$. If $d_j \in A_{i_j}$ for every j, then $d \in A_i$ contradicting its choice as a member of $B_i - A_i$. Hence $d_j \in B_{i_j} - A_{i_j}$ for some j and $d \to_S d_j$. We can repeat the argument with d_j instead of d and we get an infinite sequence $d \to_S d_j \to_S \cdots$ contrary to the assumption that M is well-founded. Hence $(A_1, \ldots, A_n) = (B_1, \ldots, B_n)$. □

As applications, one gets the classical results saying that the system S associated with a *strict* context-free grammar (i.e., a grammar such that every right-hand side of a rule is either the empty word or contains a terminal symbol) has a unique solution, and that the system S associated with a *proper* context-free grammar (i.e., a grammar such that the right-hand side of a rule is neither the empty word nor a single nonterminal symbol) has a unique solution in languages without the empty word.

In both cases, one observes that if $w \to_S w'$ then w' is shorter than w so that $\to_S$ is well-founded.

As another application, let us go back to Example 3.2. The system S has a unique solution in sets of trees which are not reduced to a single node. One can prove easily that the triple (L, L', L'') is *a* solution of S, hence is *the unique* solution of S and is thus equal to:

$$(\mathbf{L}((S, \mathbb{TREE}), u), \mathbf{L}((S, \mathbb{TREE}), v), \mathbf{L}((S, \mathbb{TREE}), w)).$$

This gives a slightly simpler proof than the one sketched in Example 3.2.

3.3. *Finite images of equational sets*

We first consider some cases where polynomial systems can be solved explicitly.

Proposition 3.5. *If F, σ and M are explicitly given, then for every polynomial system S and for every unknown u_i of S one can compute the finite set* $\mathbf{L}((S, M), u_i)$.

Proof. That F, σ and M are explicitly given implies that they are all finite. The sequence $S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)$ is increasing for component-wise inclusion of tuples of sets, i.e., $A^j_i \subseteq A^{j+1}_i$ for every i and every j where $(A^j_1, \ldots, A^j_n) = S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)$. Since M is finite, this sequence cannot be strictly increasing at all steps, hence we have

$$S^{j+1}_{\wp(M)}(\emptyset, \ldots, \emptyset) = S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)$$

for some j, and then,

$$S^{k+1}_{\wp(M)}(\emptyset, \ldots, \emptyset) = S^k_{\wp(M)}(\emptyset, \ldots, \emptyset) = S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)$$

for every $k \geqslant j$ (by induction on k). It follows that $S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)$ is the least solution of S in M. Since M is explicitly given, one can compute the tuples (of finite sets) $S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)$ for $j = 0, 1, \ldots$ and stop as soon as two successive tuples are equal. This algorithm terminates since M is finite, and gives the least solution of S in M. The desired set $\mathbf{L}((S, M), u_i)$ is thus the ith component of the obtained least solution. □

Easy modifications of the above proof yield the following improvement which concerns infinite magmas.

Proposition 3.6. *Let F, σ, M be effectively given. If the components of the least solution of a polynomial system are all finite, then they can be effectively computed.*

The following result is due to Mezei and Wright [38].

Proposition 3.7. *If $h: M \to M'$ is an F-homomorphism, if S is a polynomial system over F, then* $\mathbf{L}((S, M'), u) = h(\mathbf{L}((S, M), u))$ *for every unknown u.*

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

$$S^j_{\wp(M')}(\emptyset, \ldots, \emptyset) = {}^{\wp}h(S^j_{\wp(M)}(\emptyset, \ldots, \emptyset)),$$

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

Propositions (3.5)–(3.7) can be used jointly as follows to compute certain finite images of equational sets. Assume that M is an F-magma and h is a mapping : $M \to P$, where P is a finite, explicitely given set (initially without operations on it). We may want to compute the finite set ${}^{\wp}h(L) \subseteq P$ for a set $L \in \mathbf{Equat}(M)$. It suffices to define on P a structure of F-magma making $h: M \to P$ into an F-homomorphism. If $L = \mathbf{L}((S, M), u)$, then ${}^{\wp}h(L) = \mathbf{L}((S, P), u)$ by Proposition 3.7, and $\mathbf{L}((S, P), u)$ is computable by Proposition 3.5. The case of an infinite but effectively given set P can be dealt with similarily by means of Proposition 3.6 for systems S having a least solution $(L_1, \ldots, L_n)$ such that each set ${}^{\wp}h(L_i)$ is finite.

An algorithm that decides whether $\mathbf{L}((S, M), u)$ is empty can be obtained by taking for P the set $\{\underline{s} \mid s \in \mathcal{S}\}$, where $\underline{s}$ is an object of sort s. There is thus a unique (trivial) structure of F-magma on P and the mapping $h: M \to P$ that maps onto $\underline{s}$ every element of M of sort s is a homomorphism. It follows that $\mathbf{L}((S, M), u) \neq \emptyset$ if and only if $h(\mathbf{L}((S, M), u)) = \{\underline{\sigma(u)}\}$. The sets ${}^{\wp}h(\mathbf{L}((S, M), u))$ can thus be computed and give the desired information. (It follows from these facts, and also from Corollary 3.8 that whether $\mathbf{L}((S, M), u)$ is empty or not depends only on S. It is essential here that the operations be assumed total.)

Here is another consequence of Proposition 3.7. (We recall that $\mathbf{h}_M$ denotes the unique homomorphism $\mathbf{T}(F) \to M$).

Corollary 3.8. *For every polynomial system S and for every unknown u of S we have* $\mathbf{L}((S, M), u) = \mathbf{h}_M(\mathbf{L}((S, \mathbf{T}(F)), u))$.

This means that an equational set, defined as a component of the least solution of polynomial system S, is the image of the corresponding component of the least solution of S in the F-magma of terms $\mathbf{T}(F)$, under the canonical homomorphism.

3.4. Linearly derived operations in powerset magmas

Let M be an F-magma and f be a function in F. The mapping $f_{\wp(M)}$ is defined as the set extension ${}^{\wp}f_M$ of the mapping f_M. The following lemma shows that the linearly derived operations of $\wp(M)$ can be characterized similarly as the set extensions of the corresponding ones of M.

Lemma 3.9. *For every linear term w in $\mathbf{T}(F, X)$, for every F-magma M, we have $w_{\wp(M)} = {}^{\wp}w_M$.*

Proof. Let $\chi = (x_1, \ldots, x_k) = \mathbf{var}(w)$. We shall prove that for every k-tuple of sets $(A_1, \ldots, A_k)$ such that $A_i \subseteq M_{\sigma(x_i)}$, we have

$$w_{\wp(M)}(A_1, \ldots, A_k) = \{w_M(d_1, \ldots, d_k) \mid d_1 \in A_1, \ldots, d_k \in A_k\}. \tag{3.3}$$

This is clear from the definition of $f_{\wp(M)}$ if $w = f(x_1, \ldots, x_k)$. The proof is by induction on the structure of w in the general case. The base cases, where w is either the variable x_1 or a constant (in this case $k = 0$) are obviously true. We only consider the case where $w = g(w_1, w_2)$ (the case where g has rank other than 2 is essentially the same). We let $\chi' = (x_1, \ldots, x_p) = \mathbf{var}(w_1)$ and $\chi'' = (x_{p+1}, \ldots, x_k) = \mathbf{var}(w_2)$.

We let $\vec{A}$ denote $(A_1, \ldots, A_k)$. If any set A_i is empty, then both hands of (3.3) are empty. Otherwise, we have

$$\begin{aligned} w_{\wp(M)}(\vec{A}) &= g_{\wp(M)}(w_{1_{\wp(M),\chi}}(\vec{A}), w_{2_{\wp(M),\chi}}(\vec{A})) \\ &= \{g_M(e_1, e_2) \mid e_1 = w_{1_{M,\chi}}(a_1, \ldots, a_k), e_2 = w_{2_{M,\chi}}(b_1, \ldots, b_k) \\ &\qquad a_1, b_1 \in A_1, \ldots, a_k, b_k \in A_k\}. \end{aligned}$$

(The last equality follows from the induction hypothesis.) We have thus:

$$\begin{aligned} w_{\wp(M)}(\vec{A}) &= \{g_M(w_{1_{M,\chi}}(a_1, \ldots, a_p), w_{2_{M,\chi''}}(b_{p+1}, \ldots, b_k)) \mid \\ &\qquad a_1 \in A_1, \ldots, a_p \in A_p, b_{p+1} \in A_{p+1}, \ldots, b_k \in A_k\} \\ &= \{g_M(w_{1_{M,\chi}}(d_1, \ldots, d_k), w_{2_{M,\chi}}(d_1, \ldots, d_k)) / d_1 \in A_1, \ldots, d_k \in A_k\} \\ &= \{w_M(d_1, \ldots, d_k) / d_1 \in A_1, \ldots, d_k \in A_k\} \end{aligned}$$

as was to be proved. □

Proposition 3.10. *Let G be a signature that is linearly derived of a signature F. Let P be the G-derived magma of an F-magma M. For every term $t \in \mathbf{T}(F, X)$, we have $t_{\wp(P)} = \underline{\delta}(t)_{\wp(M)}$, where δ denotes δ_G.*

We recall that we denote by $\underline{\delta}(t)$ the result of the replacement in term t of every symbol of G by its "definition" $\delta(g)$ (see Section 2.2).

Proof. Let $(x_1, \ldots, x_n) = \mathbf{var}(t)$. Let $L_1, \ldots, L_n \subseteq M$ with $L_i \subseteq M_{\sigma(x_i)}$. We shall prove that

$$t_{\wp(P)}(L_1, \ldots, L_n) = \underline{\delta}(t)_{\wp(M)}(L_1, \ldots, L_n) \tag{3.4}$$

by induction on the structure of t. If $t = f \in F$, then $\underline{\delta}(t)$ has no variable and equality (3.4) holds because its both hands are equal to $\{f_P\} = \{\delta(f)_M\}$. If $t = x_i$ then equality

(3.4) holds because its both hands are equal to L_i. If $t = f(t_1, \ldots, t_k)$, then we have (letting $\vec{L}$ denote $(L_1, \ldots, L_n)$):

$$
\begin{aligned}
t_{\varphi(P)}(\vec{L}) &= f_{\varphi(P)}(t_{1\varphi(P)}(\vec{L}), \ldots, t_{k\varphi(P)}(\vec{L})) \\
&= \{ f_P(d_1, \ldots, d_k) \mid d_i \in t_{i\varphi(P)}(\vec{L}), \quad i = 1, \ldots, k \} \\
&= \{ \delta(f)_M(d_1, \ldots, d_k) / d_i \in t_{i\varphi(P)}(\vec{L}), \quad i = 1, \ldots, k \} \\
&= \{ \delta(f)_M(d_1, \ldots, d_k) / d_i \in \underline{\delta}(t_i)_{\varphi(M)}(\vec{L}), \quad i = 1, \ldots, k \} \\
&\qquad \text{(by the induction hypothesis),} \\
&= \delta(f)_{\varphi(M)}(\underline{\delta}(t_1)_{\varphi(M)}(\vec{L}), \ldots, \underline{\delta}(t_k)_{\varphi(M)}(\vec{L})) \\
&\qquad \text{(because } \delta(f) \text{ is linear and by Lemma 3.9).} \\
&= \underline{\delta}(t)_{\varphi(M)}(\vec{L}) \\
&\qquad \text{(since } \underline{\delta}(t) = \delta(t)[\underline{\delta}(t_1), \ldots, \underline{\delta}(t_k)]. \qquad \square
\end{aligned}
$$

Corollary 3.11. *If P is a linearly derived magma of a magma M, then* **Equat(P)** $\subseteq$ **Equat**(M).

Proof. Let (G, δ) be a signature that is linearly derived of a signature F. Let P be the G-derived magma of an F-magma M. Let L be a P-equational set defined as a component of the least solution of a polynomial system S over G. Let S' be the polynomial system obtained from S by replacing every monomial t of S by $\underline{\delta}(t)$. It follows from Proposition 3.10 that the mappings $S'_{\varphi(M)}$ and $S_{\varphi(M)}$ are equal. Hence, they have the same least fixed point. The set L is one of its components, hence, is M-equational. $\square$

Example 3.12. We consider the system of equations $S = \langle u = a.(u.(u.v)) + a.(v.b),\ v = a.(v.b) + a.b \rangle$ associated with the context-free grammar of Example 3.1. We define a linearly derived signature G of three symbols p, q, s as follows: $\delta(p) = a.(x_1.(x_2.x_3))$, $\delta(q) = a.(x_1.b)$, $\delta(s) = a.b$. The system S interpreted in $\wp(\mathbb{W}_A)$ is "the same" as the system $S' = \langle u = p(u, u, v) + q(v),\ v = q(v) + s \rangle$ interpreted in the G-derived magma of $\mathbb{W}_A$.

It is important to assume in Corollary 3.11 that G is *linearly derived of* F, as shown by the following counterexample.

Example 3.13. Let S be the equation $\langle u = \mathbf{sq}(u) + a \rangle$. Let $M = \mathbb{W}_A$ (see Example 2.1). Let $\delta(\mathbf{sq})$ be the nonlinear term $x_1.x_1$. Let S' be the equation obtained by replacing in S the monomial $\mathbf{sq}(u)$ by the monomial $u.u$.

We obtain the equation $S' = \langle u = u.u + a \rangle$. Let P be the $\{\mathbf{sq}, a\}$-magma $\langle A^*, \mathbf{sq}, a \rangle$, where $\mathbf{sq}(u) = uu$ for every $u \in A^*$. Then $\mathbf{L}((S', \mathbb{W}_A), u) = \{a^n \mid n \geqslant 1\}$, and $\mathbf{L}((S, P), u) = \{a^{2^m} \mid m \geqslant 0\}$. These two sets are different, hence the proof of Corollary 3.11 does not work. This shows also that $\mathbf{Equat}(P)$ is not included in $\mathbf{Equat}(M)$ since $\mathbf{Equat}(M)$ is the set of context-free languages over A and the language $\mathbf{L}((S, P), u)$ is P-equational but not context-free. Hence Corollary 3.11 does not apply to M and P. Proposition 3.10 does not apply here because we have, for every language L:

$$\mathbf{sq}_{\wp(P)}(L) = \{uu \mid u \in L\},$$

which differs from:

$$(x_1 . x_1)_{\wp(P)}(L) = \{uv \mid u, v \in L\}.$$

3.5. *Regular term grammars*

We give here a characterization of the M-equational sets in terms of ground term rewriting systems. Let $S = \langle u_1 = p_1, \ldots, u_n = p_n \rangle$ be a polynomial system; let $U = \{u_1, \ldots, u_n\}$. We denote by $\mathbf{R}(S)$ the ground rewriting system over $F \cup U$ consisting of the rules of the form $u_i \to t$, where $i \in \{1, \ldots, n\}$ and t is one of the monomials forming the polynomial p_i. It is important to note that the symbols in U are constant (nullary) symbols and not variables; thus $\mathbf{R}(S)$ is *ground*. Hence, in the application of a rule, the symbols in U will not be substituted by arbitrary terms as it would be the case if they would be treated as variables. We have $s \xrightarrow[\mathbf{R}(S)]{} s'$ if and only if $s = c[u_i]$, $s' = c[t]$ for some context c in $\mathbf{ctxt}(F \cup U)$ and some rule $u_i \to t$ in $\mathbf{R}(S)$. We let $\mathbf{L}(S, u_i) := \{t \in \mathbf{T}(F) \mid u_i \xrightarrow[\mathbf{R}(S)]{*} t\}$.

We shall say that $\langle F, U, \mathbf{R}(S) \rangle$ is a *regular term grammar*.

Proposition 3.14. *For every polynomial system S over F and every unknown u_i of S, we have*: $\mathbf{L}(S, u_i) = \mathbf{L}((S, \mathbf{T}(F)), u_i)$.

This means that the tuple of sets of terms $\mathbf{L}(S, u_i)_{i=1,\ldots,n}$ generated by the regular term grammar $\langle F, U, \mathbf{R}(S) \rangle$ is also the least solution in $\wp(\mathbf{T}(F))$ of the polynomial system S. Its proof will use some lemmas.

Lemma 3.15. *Let $w, w' \in \mathbf{T}(F \cup U)$, where $w = f(w_1, \ldots, w_k)$, $f \in F$ and $w_1, \ldots, w_k \in \mathbf{T}(F \cup U)$. Let $n \in \mathbb{N}$. The following conditions are equivalent*:

(1) $w \xrightarrow[\mathbf{R}(S)]{n} w'$

(2) $w' = f(w'_1, \ldots, w'_k)$, *for some* $w'_1, \ldots, w'_k \in \mathbf{T}(F \cup U)$ *such that* $w_i \xrightarrow[\mathbf{R}(S)]{n_i} w'_i$ *for each* i, *where* $n_1, \ldots, n_k$ *are integers such that* $n = n_1 + n_2 + \cdots + n_k$.

Proof. The implication (2) ⇒ (1) is clear. The proof of the implication (1) ⇒ (2) is by induction on n. The cases $n = 0$ and 1 are easy. In the case $n > 1$, we let

$$w \xrightarrow[\mathbf{R}(S)]{m} w'' \xrightarrow[\mathbf{R}(S)]{p} w'$$

with $n = m + p$, $m, p < n$. The induction hypothesis gives

$$w'' = f(w''_1, \ldots, w''_k),$$

$$w_i \xrightarrow[\mathbf{R}(S)]{m_i} w''_i \quad \text{for every } i = 1, \ldots, k,$$

$$m = m_1 + \cdots + m_k.$$

Considering now the derivation $w'' \xrightarrow[\mathbf{R}(S)]{p} w'$, we get also by the induction hypothesis:

$$w' = f(w'_1, \ldots, w'_k),$$

$$w''_i \xrightarrow[\mathbf{R}(S)]{p_i} w'_i \quad \text{for every } i = 1, \ldots, k,$$

$$p = p_1 + \cdots + p_k,$$

and we get the desired result with $n_i = m_i + p_i$. □

Let $w \in \mathbf{T}(F \cup U)$. We can write $w = \mathbf{lin}(w)\,[u_{i_1}, \ldots, u_{i_m}]$, where $\mathbf{lin}(w)$ is linear over F (see Section 2). With this notation we have the following consequence of Lemma 3.15.

Lemma 3.16. *Let* $w \xrightarrow[\mathbf{R}(S)]{n} w'$ *and* $\mathbf{var}(w) = (u_{i_1}, \ldots, u_{i_m})$. *We have* $w' = \mathbf{lin}(w)\,[w'_1, \ldots, w'_m]$, *where*

$$u_{i_j} \xrightarrow[\mathbf{R}(S)]{n_j} w'_j \quad \textit{for every } j = 1, \ldots, m, \textit{ and } n = n_1 + n_2 + \cdots + n_m.$$

Proof. By induction on the structure of w and by using the implication (1) ⇒ (2) of Lemma 3.15. □

Proof of Proposition 3.14. We first verify that the n-tuple $\vec{L} := (\mathbf{L}(S, u_i))_{i=1,\ldots,n}$ satisfies

$$\vec{L} \supseteq S_{\varphi(\mathbf{T}(F))}(\vec{L}). \tag{3.5}$$

From the definition of $S_{\varphi(\mathbf{T}(F))}$, proving (3.5) amounts to proving

$$\mathbf{L}(S, u_i) \supseteq t_{\varphi(\mathbf{T}(F))}(\vec{L}) \tag{3.6}$$

for every $i = 1,\ldots,n$ and every monomial t that is a summand of the polynomial p_i, hence to proving that

$$t' \in \mathbf{L}(S, u_i), \tag{3.7}$$

where t' is obtained by the substitution in t of a term in $\mathbf{L}(S, u_j)$ for each occurrence of u_j, $j = 1,\ldots,n$. But this is clearly true since $u_i \to t$ is a rule of $\mathbf{R}(S)$ and by the definition of the sets $\mathbf{L}(S, u_j)$. It follows that

$$\vec{L} \supseteq S^j_{\wp(\mathbf{T}(F))}(\emptyset,\ldots,\emptyset) \tag{3.8}$$

for all j by induction on j (the case $j = 0$ is clear and (3.8) implies that $S^{j+1}_{\wp(\mathbf{T}(F))}(\emptyset,\ldots,\emptyset) \subseteq S_{\wp(\mathbf{T}(F))}(\vec{L}) \subseteq \vec{L}$ by (3.5) and the monotonicity of $S_{\wp(\mathbf{T}(F))}$ for set inclusion). Hence, $\vec{L}$ contains the least solution of S in $\wp(\mathbf{T}(F))$, and we have

$$\mathbf{L}(S, u_i) \supseteq \mathbf{L}((S, \mathbf{T}(F)), u_i) \tag{3.9}$$

for all $i = 1,\ldots,n$. In order to prove the opposite inclusion, we let, for $m \in \mathbb{N}$:

$$\mathbf{L}^m(S, u_i) := \{t \in \mathbf{T}(F) \mid u_i \xrightarrow[\mathbf{R}(S)]{m'} t,\ m' \leqslant m\}.$$

We shall prove that, for every solution $\vec{M} = (M_1,\ldots,M_n)$ of S in $\wp(\mathbf{T}(F))$, and in particular for the least one, we have

$$\mathbf{L}^m(S, u_i) \subseteq M_i. \tag{3.10}$$

The proof is done by induction on m (simultaneously for all $i = 1,\ldots,n$). The case $m = 0$ is clear because $\mathbf{L}^0(S, u_i) = \emptyset$. For the general case we let $w \in \mathbf{L}^m(S, u_i)$ and we let $u_i \xrightarrow[\mathbf{R}(S)]{} t \xrightarrow[\mathbf{R}(S)]{m-1} w$ be the corresponding rewriting sequence. Note that t is a summand of p_i. There are two cases.

Case 1: $t = u_j$. Then we have $M_j \subseteq M_i$ (since $\vec{M}$ is a solution of S) and $w \in \mathbf{L}^{m-1}(S, u_j)$. Then, $w \in M_j$ by induction hypothesis, hence $w \in M_i$.

Case 2: $t = f(t_1,\ldots,t_k)$. We can write $t = \mathbf{lin}(t)\,[u_{i_1},\ldots,u_{i_\ell}]$ and we get from Lemma 3.16 that $w = \mathbf{lin}(t)\,[w_1,\ldots,w_\ell]$ where $u_{i_j} \xrightarrow[\mathbf{R}(S)]{m_j} w_j$ for every $j = 1,\ldots,\ell$, and $m - 1 = m_1 + m_2 + \cdots + m_\ell$. Hence $w_j \in M_j$ for every j (we can use the induction hypothesis since $m_j < m$), and $w \in \mathbf{lin}(t)_{\wp(\mathbf{T}(F))}(M_{i_1},\ldots,M_{i_\ell}) = t_{\wp(\mathbf{T}(F))}(M_1,\ldots,M_n)$. Since $\vec{M}$ is a solution of S, we have $t_{\wp(\mathbf{T}(F))}(M_1,\ldots,M_n) \subseteq M_i$ hence finally, we have $w \in M_i$ as desired.

Hence we have established (3.10) for every m. It follows that

$$\mathbf{L}(S, u_i) = \bigcup\{\mathbf{L}^m(S, u_i) \mid m \geqslant 0\} \subseteq M_i. \tag{3.11}$$

Taking in (3.11) $M_i = \mathbf{L}((S, \mathbf{T}(F)), u_i)$ and by (3.9), we get that the equality holds in (3.9) as desired. □

By combining Corollary 3.8 and Proposition 3.14, we get the following characterization:

Proposition 3.17. *For every polynomial system S over F and every unknown u of S, for every F-magma M:*

$$\begin{aligned}\mathbf{L}((S, M), u) &= \mathbf{h}_M(\mathbf{L}((S, \mathbf{T}(F)), u)) \\ &= \{t_M \mid t \in \mathbf{L}(S, u)\}.\end{aligned}$$

The sets $\mathcal{S}$ and F may be infinite in all definitions and results concerning equational sets except in Proposition 3.5. However, a polynomial system is a finite object, written with finitely many function symbols (forming a subset F' of F). The set of sorts appearing in the types of the symbols in F' or as sorts of the unknowns of S is a finite subset $\mathcal{S}'$ of $\mathcal{S}$. We let M' be the many-sorted F'-magma with set of sorts $\mathcal{S}'$ with domains M_s for $s \in \mathcal{S}'$ and functions f_M for $f \in F'$. We shall say that M' is the *restriction of M to the finite subsignature F' of F.*

Corollary 3.18. *Every M-equational set is finitely generated. Every M-equational set is M'-equational for some restriction M' of M to a finite subsignature of F.*

Proof. From Proposition 3.17 we have

$$\begin{aligned}\mathbf{L}((S, M), u) &= \{t_M \mid t \in \mathbf{L}(S, u)\} \\ &= \{t_{M'} \mid t \in \mathbf{L}(S, u)\} \\ &= \mathbf{L}((S, M'), u),\end{aligned}$$

where M' is the restriction of M to the finite subsignature of F consisting of the symbols occurring in S. In particular, $\mathbf{L}((S, M), u)$ is generated by this subsignature. □

This corollary shows that the possible infiniteness of the signature does not affect very much the theory of equational sets. The situation will be different in the next section for recognizable sets.

3.6. *Uniform systems*

The notion of a uniform polynomial system is the natural generalization of that of a context-free grammar in Chomsky normal form. A system is *uniform* if every monomial has exactly one occurrence of a function symbol (it can be a constant), i.e., is of the form $f(u_{i_1}, \ldots, u_{i_k})$ (f in the case $k = \rho(f) = 0$) where $U = \{u_1, \ldots, u_n\}$ is the set of unknowns.

Proposition 3.19. *Let S be a polynomial system with set of unknowns U. One can construct a uniform polynomial system S' with set of unknowns $U' \supseteq U$ such that for every M and $u \in U$:* $\mathbf{L}((S', M), u) = \mathbf{L}((S, M), u)$.

Proof. Let S be polynomial system that is not uniform. The monomials that are not of the appropriate form can be of two types:

(a) either they have no occurrence of a function symbol, i.e., are equal to some unknown u,

(b) or they have more than one occurrence of a function symbol, i.e., are of the form $f(t_1, \ldots, t_k)$ where at least one of $t_1, \ldots, t_k$ is not in U.

We first transform S in an equivalent system S'' with same set of unknowns that has no rule of the form (a). If S has already no rule of the form (a), we let of course $S'' = S$. Otherwise we let

$$A = \{(u, u') \in U \times U \mid u \xrightarrow[\mathbf{R}(S)]{*} u',\ u' \neq u\},$$

and we describe S'' in terms of the associated regular term grammar $\mathbf{R}(S'')$. This grammar is obtained from $\mathbf{R}(S)$ as follows:

(1) one deletes all rules with a right-hand side in U,

(2) for every rule $u \to m$, where m is not in U, for every u' such that $(u', u) \in A$, one adds the rule $u' \to m$.

It is not hard to prove that the rewriting relations $\xrightarrow[\mathbf{R}(S'')]{*}$ and $\xrightarrow[\mathbf{R}(S)]{*}$ are the same. It follows then from Proposition 3.17 that $\mathbf{L}((S'', M), u) = \mathbf{L}((S, M), u)$ for every M and every $u \in U$.

It remains to eliminate the rules of $\mathbf{R}(S'')$ that are of the form (b). We let B be the set of terms $t \in \mathbf{T}(F \cup U) - U$ that are proper subterms of right-hand sides of rules of $\mathbf{R}(S'')$. For every such term t, we let $\bar{t}$ be a new unknown and we let $U' = U \cup \{\bar{t} \mid t \in B\}$. We let also $\bar{t} \to f(w_1, \ldots, w_k)$ be a new rule (that we shall call *the rule defining* $\bar{t}$), where $t = f(t_1, \ldots, t_k)$, $w_i = t_i$ if $t_i \in U$, $w_i = \bar{t}_i$ if $t_i \notin U$ (so that $t_i \in B$). We now let S' be the polynomial system with set of unknowns U' such that $\mathbf{R}(S')$ consists of the following rules:

(1) the rules defining the unknowns $\bar{t}$ for $t \in B$,

(2) the rules of $\mathbf{R}(S'')$, the right-hand side of which has exactly one occurrence of a symbol in F,

(3) the rules $u \to f(w_1, \ldots, w_k)$ for every rule $u \to f(t_1, \ldots, t_k)$ of $\mathbf{R}(S'')$ where at least one of $t_1, \ldots, t_k$ is not in U and $w_i = t_i$ if $t_i \in U$ and $w_i = \bar{t}_i$ otherwise.

Again it is easy to establish that $\mathbf{L}(S', u) = \mathbf{L}(S'', u)$ for every $u \in U$ and it follows that $\mathbf{L}((S, M), u) = \mathbf{L}((S', M), u)$ for every F-magma M and $u \in U$. □

By using this construction for a context-free grammar, one obtains a grammar in Chomsky Normal Form.

3.7. Derivation trees

A polynomial system S with set of unknowns $U = \{u_1, \ldots, u_n\}$ is *strongly uniform* if every monomial in S is of the form $f(u_{i_1}, \ldots, u_{i_k})$ for some $f \in F$ and if each symbol f in F occurs in at most one monomial of S. In this case, the set $F' \subseteq F$ of symbols of F having one occurrence in S is in bijection with the set of rules $\mathbf{R}(S)$. The set $\bigcup\{\mathbf{L}(S, u) \mid u \in U\} \subseteq \mathbf{T}(F')$ is called the set of *derivation trees* of S. More precisely, we let $\mathbf{Der}(S, u) := \mathbf{L}(S, u)$ and we call it *the set of derivation trees of S relative to u.*

Let us now consider an arbitrary polynomial system. Let P be a new alphabet in bijection with $\mathbf{R}(S)$. Let p in P correspond to a rule of the form $u_i \to t$, where $t \in \mathbf{T}(F, U)$. Note that t is a monomial of S. Let $(u_{j_1}, \ldots, u_{j_\ell}) = \mathbf{var}(t)$ be the sequence of unknowns of t; then we define the type of p as $\sigma(u_{j_1}) \times \cdots \times \sigma(u_{j_\ell}) \to \sigma(u_i)$.

We let S' be the polynomial system obtained from S be replacing in every equation $u_i = \cdots + t + \cdots$ the monomial t by $p(u_{j_1}, \ldots, u_{j_\ell})$ (where, as above p corresponds to the rule $u_i \to t$ of $\mathbf{R}(S)$ and $(u_{j_1}, \ldots, u_{j_\ell}) = \mathbf{var}(t)$). Then S' is a strongly uniform system. We define the set of derivation trees of S as that of S'; more precisely, we let $\mathbf{Der}(S, u) := \mathbf{Der}(S', u)$. If S is already strongly uniform, then we take $P := F'$ (the set of symbols of F having an occurrence in S) and then $S' = S$.

Our next aim is to relate the semantics of S' to that of S. Let S and S' be as above. We make P into a derived signature of F by defining $\delta_P(p) := \mathbf{lin}(t)$, where p corresponds to a rule $u_i \to t$ of $\mathbf{R}(S)$. We let M' be the derived P-magma of M. The following proposition is a consequence of Proposition 3.10 and its Corollary 3.11.

Proposition 3.20. *For every unknown u of S, we have* $\mathbf{L}((S, M), u) = \mathbf{L}((S', M'), u)$.

This proposition gives a semantic meaning to derivation trees: a derivation tree is not only a syntactic representation of the way an object is generated but also a term over a derived signature, the value of which is the considered object.

Example 3.21. We consider again the context-free grammar of Example 3.1, $G = \{u \to auuv, u \to avb, v \to avb, v \to ab\}$. The corresponding system of equations is

$$S = \langle u = a.(u.(u.v)) + a.(v.b), \quad v = a.(v.b) + a.b \rangle.$$

Let us name the four rules p, q, r, s. We obtain the system:

$$S' = \langle u = p(u, u, v) + q(v), v = r(v) + s \rangle.$$

Note that the monomial $a.(v.b)$ occurs in two equations of S, and is replaced by $q(v)$ in the first equation and by $r(v)$ in the second. The derived operations associated with p, q, r, s are defined by the terms $\delta(p) = a.(x_1.(x_2.x_3))$, $\delta(q) = a.(x_1.b)$, $\delta(r) = a.(x_1.b)$, and $\delta(s) = a.b$. The derivation tree corresponding to the derivation sequence generating the word *aaabbaabbaabb* is thus $p(q(s), q(s), r(s))$.

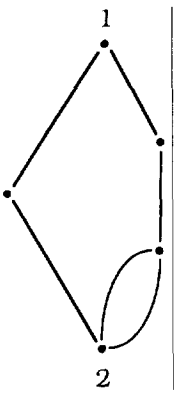

Fig. 1.

Example 3.22 (*Derivation trees of series-parallel graphs with two sources*). As in Example 2.3, we define series-parallel graphs by the equation

$$u = u||_2 u + u \bullet u + e$$

solved in $\wp(\langle \mathbb{G}_2, ||_2, \bullet, e\rangle)$. The set of derivation trees is defined by the same equation. An example of derivation tree is $(e \bullet e)||_2(e \bullet e \bullet (e||_2 e))$ (written as a term). The corresponding series-parallel graph is shown in Fig. 1.

3.8. *Closure properties of* **Equat**(M)

Proposition 3.23. *For every F-magma M, the family* **Equat**(M) *contains the empty sets and the singletons defined by terms in* $\mathbf{T}(F)$. *It is closed under union of sets of the same sort, and under the operations of F.*

Proof. The first two assertions are clear. Let $L, L' \in$ **Equat**(M), $L, L' \subseteq M_s$ for some s.

Case 1: If L and L' are defined by the same system S, say $L = \mathbf{L}((S, M), u_i)$, $L' = \mathbf{L}((S, M), u_j)$ then we have $L \cup L' = \mathbf{L}((S_1, M), u)$, where u is a new unknown and $S_1 = S \cup \{u = u_i + u_j\}$.

Case 2: L and L' are defined by different systems. We have $L = \mathbf{L}((S, M), u_i)$ and $L' = \mathbf{L}((S', M), u'_j)$; we first rename if necessary the unknowns of S' in such a way that S and S' have disjoint sets of unknowns and one has $L \cup L' = \mathbf{L}((S_2, M), u)$, where u is a new unknown and $S_2 = S \cup S' \cup \{u = u_i + u'_j\}$. The correctness of these constructions is easily proved by Proposition 3.17.

Let $f \in F$ have type $s \times s' \to s''$. Let L and L' be M-equational such that $L \subseteq M_s$ and $L' \subseteq M_{s'}$. We want to prove that $f_{\wp(M)}(L, L') := \{f_M(d, d') | d \in L,\ d' \in L'\}$ is M-equational. We have two cases as above and we use the same notation.

Case 1: We have

$$f_{\wp(M)}(L, L') = \mathbf{L}((S_3, M), u),$$

where

$$S_3 = S \cup \{u = f(u_i, u_j)\}.$$

Case 2: We have

$$f_{\varphi(M)}(L, L') = \mathbf{L}((S_4, M), u),$$

where

$$S_4 = S \cup S' \cup \{u = f(u_i, u'_j)\}.$$

Again the correctness of these constructions can be established by using Proposition 3.17. The closure of $\mathbf{Equat}(M)$ under operations of F of rank other than 2 can be proved similarly. □

Proposition 3.7 establishes that if h is an F-homomorphism: $M \to M'$ and if $L \in \mathbf{Equat}(M)$, then $h(L) \in \mathbf{Equat}(M')$. (This is a closure property of the family $\bigcup\{\mathbf{Equat}(M) \mid M \text{ is an } F\text{-magma}\}$.)

These results generalize the "easy" closure properties of the family of context-free languages, under union, concatenation and homomorphism. The closure under inverse homomorphism is a particular property of equational sets of words, actually more complicated to prove. It does not hold in general, as shown by the following example.

Example 3.24 (*Nonclosure under inverse homomorphisms of equational sets*). We let $M = \langle \mathbb{Z}, 0, s, m \rangle$, where $s(n) = n + 1$ and $m(n) = n - 1$ for all n in $\mathbb{Z}$. The singleton $L = \{0\}$ is equational but the set of terms $\mathbf{h}_M^{-1}(L) \subseteq \mathbf{T}(\{0, s, m\})$ is not. (If it would be equational, it would be also recognizable by Proposition 5.3; hence, it would be defined by a finite tree-automaton (see Section 4.3) ; from such an automaton, one would obtain a finite automaton defining the set of words in $\{s, m\}^*$ with an equal number of occurrences of s and m; this impossible because this language is not regular.)

4. Recognizable sets

The notion of a recognizable set is due to Mezei and Wright [38]; it generalizes the notion of a regular language like the notion of an equational set generalizes that of a context-free one. It was originally defined for one-sort magmas, and we adapt it to many-sorted ones, possibly with infinitely many sorts.

4.1. Definitions

Let F be an $\mathcal{S}$-signature. An F-magma A is *locally finite* if each domain A_s, $s \in \mathcal{S}$, is finite.

Let M be an F-magma and $t \in \mathcal{S}$. A subset B of M_t is *M-recognizable* if there exists a locally finite F-magma A, a homomorphism $h: M \to A$, and a (finite) subset C of A_t such that $B = h^{-1}(C)$. The pair (h, A) is called a *semi-automaton*, and the triple (h, A, C) is called an *automaton*. Intuitively, C is the set of "accepting states" of a deterministic automaton, the set of states of which is $\bigcup\{A_s \mid s \in \mathcal{S}\}$. (The relationships with the classical notion of a finite automaton will be discussed in Section 4.3.) We shall denote by $\mathbf{Rec}(M)_t$ the family of M-recognizable subsets of M_t.

We say that B as above is *effectively M-recognizable* if M is effectively given, if B is recognized by an automaton (h, A, C), where A is effectively given (and defined by an effective coding $(\|A\|, \gamma_A, \chi_A)$, h and the mapping associating with every sort s the finite set of integers $\gamma_A^{-1}(A_s)$ are computable and $C = \gamma_A(C')$ for some explicitly given subset C' of $\|A\|$. In this case, one can decide whether an element m of M_t belongs to B: it suffices to compute $h(m)$ (where m is given by a number coding it) and to test whether $h(m)$ belongs to C, which is possible by the computability assumptions.

A language included in A^* is regular if and only if it is $\mathbb{W}_A$-recognizable if and only if it is $\mathbb{U}_A$-recognizable. The recognizable sets of terms, i.e, the $\mathbf{T}(F)$-recognizable sets, where F is a finite signature, can be characterized by finite *tree*-automata (see Section 4.3)). The classical identification of terms with finite ordered ranked trees explains the now classical although improper qualification of "tree"-automaton.

Recognizable sets can also be characterized in terms of congruences (reviewed in Section 2). A congruence $\sim$ on an F-magma M is *finite* if it has finitely many classes: this is possible only if M has finitely many sorts. It is *locally finite* if it has finitely many classes of each sort. A subset L of M_s is *saturated for* $\sim$ (or *$\sim$-saturated*) if, for every $d, d' \in M_s$, if d belongs to L and $d \sim d'$, then d' also belongs to L. We prove below (Proposition 4.1) that a subset L of M_s is M-recognizable if and only if it is saturated for a locally finite congruence on M. This generalizes a well-known characterization of regular languages. The notion of syntactic congruence can also be generalized to arbitrary subsets of M (all elements of which have the same sort) and yields another characterization of the M-recognizable sets. Let $L \subseteq M_u$. We associate with L a congruence $\sim_L$ on M called the *syntactic congruence* of L and defined as follows. For d, $d' \in M$: $d \sim_L d'$ if and only if $\sigma(d) = \sigma(d')$ and for every integer n, for every linear term t in $\mathbf{T}(F, \{x_1, \ldots, x_n\})_u$ such that $\sigma(x_1) = \sigma(d)$, for every $d_2, \ldots, d_n$ in $M_{\sigma(x_2)}, \ldots, M_{\sigma(x_n)}$:

$$t_M(d, d_2, \ldots, d_n) \in L \Leftrightarrow t_M(d', d_2, \ldots, d_n) \in L.$$

In the special case where F generates M, the elements $d_2, \ldots, d_n$ are defined by terms, hence, they can be "merged with t". In other words:
$d \sim_L d'$ if and only if $\sigma(d) = \sigma(d')$ and for every $t \in \mathbf{ctxt}(F)_{\sigma(d),u}$ we have

$$t_M(d) \in L \Leftrightarrow t_M(d') \in L.$$

By a *predicate* on a set E, we mean a total mapping: $E \to \{\mathbf{true}, \mathbf{false}\}$. If M is a many-sorted F-magma with set of sorts $\mathcal{S}$, a *family of predicates* on M is an indexed set $\{\hat{p} \mid p \in P\}$, such that each p in P has an arity $\alpha(p)$ in $\mathcal{S}$ (which means that $\hat{p}$ is a unary

function: $M_{\alpha(p)} \to \{\textbf{true}, \textbf{false}\}$). Such a family will also be denoted by P. For $p \in P$, we let $L_p = \{d \in M_{\alpha(p)} \mid \hat{p}(d) = \textbf{true}\}$.

The family P is *locally finite* if, for each $s \in \mathcal{S}$, the set $\{p \in P \mid \alpha(p) = s\}$ is finite. We say that P is *f-inductive* where f is an operation in F, if for every $p \in P$ of arity $s = \sigma(f)$, there exist $m_1, \ldots, m_n$ in $\mathbb{N}$, (where n is the rank of f), an $(m_1 + \cdots + m_n)$-place Boolean expression B, and a sequence of $(m_1 + \cdots + m_n)$ elements of P, $(p_{1,1}, \ldots, p_{1,m_1}, p_{2,1}, \ldots, p_{2,m_2}, \ldots, p_{1,1}, \ldots, p_{n,m_n})$, such that, if the type of f is $s_1 \times s_2 \cdots \times s_n \to s$ we have:

(1) $\alpha(p_{i,j}) = s_i$ for all $j = 1, \ldots, m_i$,

(2) for all $d_1 \in M_{s_1}, \ldots, d_n \in M_{s_n}$:

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

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

Proposition 4.1. *Let M be an F-magma. For every $s \in \mathcal{S}$, for every subset L of M_s, the following conditions are equivalent:*

(i) *L is M-recognizable,*

(ii) *L is saturated for a locally finite congruence on M,*

(iii) *the syntactic congruence of L is locally finite,*

(iv) *$L = L_p$ for some predicate p belonging to a locally finite F-inductive family of predicates on M.*

Proof. (i) $\Rightarrow$ (iv): Let $h^{-1}(C) \subseteq M_s$ for some automaton (h, A, C). The domains of A are pairwise disjoint (see Section 2). We let $P = \bigcup\{A_t \mid t \in \mathcal{S}\} \cup \{p\}$. Each element a of A_t has arity t (considered as a member of P), and p has arity s. For $d \in M_t$ and $a \in A_t$, we let:

$$\hat{a}(d) = \textbf{true} \quad \text{if } h(d) = a,$$

$$= \textbf{false} \quad \text{otherwise.}$$

For $d \in M_s$, we let

$$\hat{p}(d) = \textbf{true} \quad \text{if } h(d) \in C,$$

$$= \textbf{false} \quad \text{otherwise.}$$

It is clear that P is locally finite. It is not hard to prove that it is F-inductive, and, clearly, $L = L_p$.

(iv) $\Rightarrow$ (ii): Let P be a locally finite F-inductive family of predicates. The relation such that $d \sim d' :\Leftrightarrow \sigma(d) = \sigma(d')$ and $\hat{p}(d) = \hat{p}(d')$ for every $p \in P$ of arity $\sigma(d)$, is an

equivalence relation on M. It has finitely many classes of each sort since P is locally finite. (Let t be a sort, for each d of sort t, let $\pi(d)$ be the set of predicates p of arity t such that $\hat{p}(d)$ holds; then $d \sim d'$ if and only if $\pi(d) = \pi(d')$; since π takes at most 2^q values, where q is the number of predicates of arity t, the equivalence $\sim$ has at most 2^q classes of sort t.) It is a congruence since P is F-inductive (the verification is straightforward), and, for every p in P, the set L_p is $\sim$-saturated.

(ii) $\Rightarrow$ (i) and (iii): If L is saturated for a locally finite congruence $\sim$ on M, then one takes $\mathcal{A}(L, \sim) := (h, M/\sim, h(L))$ as an automaton defining L, where h is the canonical surjective homomorphism: $M \to M/\sim$. We have also $\sim \subseteq \sim_L$. Hence $\sim_L$ is locally finite since $\sim$ is.

(iii) $\Rightarrow$ (ii) Holds trivially. □

If $L \in \mathbf{Rec}(M)$, then $\mathcal{A}(L, \sim_L)$ is called the *minimal automaton* of L. If $M = \mathbb{U}_A$ and $L \in \mathbf{Rec}(\mathbb{U}_A)$, then $\mathcal{A}(L, \sim_L)$ is the usual minimal (deterministic) automaton of L.

Remark. Going back to the general case, let $L \subseteq M_t$ be such that for every f in F of rank at least 1, there is no $d_1, \ldots, d_n$ such that $f_M(d_1, \ldots, d_n)$ belongs to L. Intuitively speaking, this means that the operations in F are not powerful enough to "break" the elements of L. The set L is M-recognizable: it is not hard to verify that $d \sim_L d'$ if and only if d and d' are of the same sort and belong both, either to L or to its complement. Hence, there are at most two classes of each sort and L is recognizable.

A family of predicates P on an F-magma M is *effectively* locally finite if the following conditions hold:

(1) M and P are effectively given, the mapping α (defining the arities of the elements of P) is computable, and the partial function: $P \times M \to \{\mathbf{true}, \mathbf{false}\}$ associating $\hat{p}(d)$ with $p \in P$ and $d \in M_{\alpha(p)}$ is computable;

(2) P is locally finite and the mapping α^{-1} is computable (where $\alpha^{-1}(t)$ is the finite set $\{p \in P \mid \alpha(p) = t\}$ for every sort t).

It is *effectively* F-inductive if condition (1) holds together with:

(3) there exists an algorithm producing a decomposition of p relative to f, for every f in F and p in P.

Proposition 4.2. *Let M be an effectively given F-magma. An M-recognizable subset L of M_s is effectively M-recognizable if and only if $L = L_p$ for some predicate p of arity s belonging to an effectively locally finite and effectively F-inductive family of predicates on M.*

Proof. "*Only if*". By (i) $\Rightarrow$ (iv) of the proof of Proposition 4.1.

"*If*": Let P be an effectively locally finite and effectively F-inductive family of predicates on M. For every $t \in \mathcal{S}$, we let P_t be the finite set of predicates of arity t; we let Θ_t be the set of all functions: $P_t \to \{\mathbf{true}, \mathbf{false}\}$, and we let **tv** (where **tv** stands for "truth value") be the mapping $M_t \to \Theta_t$ such that, for every $m \in M_t$, $\mathbf{tv}(m)$ is the mapping

$p \mapsto \hat{p}(m)$ for $p \in P_t$. From the hypothesis that P is effectively F-inductive, one can determine for every $f \in F$, a mapping f_Θ such that

$$\mathbf{tv}(f_M(m_1,\ldots,m_k)) = f_\Theta(\mathbf{tv}(m_1),\ldots,\mathbf{tv}(m_k))$$

for all $(m_1,\ldots,m_k) \in M_{\alpha(f)}$. Hence $\Theta = \langle (\Theta_t)_{t\in s},\ (f_\Theta)_{f\in F}\rangle$ is an effectively given F-magma and $\mathbf{tv}$ is a computable homomorphism $M \to \Theta$. Hence $(\mathbf{tv}, \Theta)$ is a semi-automaton, since Θ is locally finite. We have $L_p = \mathbf{tv}^{-1}(\Theta')$, where $\Theta' = \{\theta \in \Theta \mid \theta(p) = \mathbf{true}\}$. Hence L_p is effectively M-recognizable. □

Example 4.3. Let L be the set of rooted trees with a number of nodes that is at least 7 and is not a multiple of 3. Let p be the corresponding predicate on $\mathbb{R}$ (we use the notation and definitions of Example 2.3). Let us consider the following predicates: for $i = 0, 1$ we let $q_i(t)$ hold if and only if the number of nodes of t is of the form $3k + i$ for some k; for $i = 1,\ldots,6$, we let $r_i(t)$ hold if and only if the number of nodes of t is equal to i. It is easy to check that $P = \{q_0, q_1, r_1,\ldots,r_6\}$ is inductive with respect to the operations **ext** and $||$ on rooted trees; this verification uses in particular the following facts which hold for all rooted trees t and t':

$$q_1(t||t') = (q_1(t) \wedge q_1(t'))$$
$$\vee\ (q_0(t) \wedge \neg q_0(t') \wedge \neg q_1(t')) \vee \{\neg q_0(t) \wedge \neg q_1(t) \wedge q_0(t')\},$$
$$q_0(\mathbf{ext}(t)) = \neg q_0(t) \wedge \neg q_1(t),$$
$$q_1(\mathbf{ext}(t)) = q_0(t),$$
$$r_4(t||t') = (r_1(t) \wedge r_4(t')) \vee (r_2(t) \wedge r_3(t')) \vee (r_3(t) \wedge r_2(t')) \vee (r_4(t) \wedge r_1(t')).$$

Since p is equivalent to $\neg q_0 \wedge \neg r_1 \wedge \neg r_2 \wedge \cdots \wedge \neg r_6$, we get the automaton witnessing that L is recognizable by taking in the construction of the preceding proof: $\Theta' = \{\theta \in \Theta / \theta(m) = \mathbf{false}$ if $m \in \{q_0, r_1,\ldots, r_6\}\}$.

Proposition 4.4. *Let M be an F-magma generated by F and u be a sort. A subset L of M_u is M-recognizable if and only if $\mathbf{h}_M^{-1}(L)$ is $\mathbf{T}(F)$-recognizable.*

Proof. "*Only if*": *Let* $L = h^{-1}(C)$ for some homomorphism $h: M \to A$, where A is locally finite, then $\mathbf{h}_M^{-1}(L) = (h \circ \mathbf{h}_M)^{-1}(C)$, and, since $h \circ \mathbf{h}_M$ is a homomorphism: $\mathbf{T}(F) \to A$, the set $\mathbf{h}_M^{-1}(L)$ is $\mathbf{T}(F)$-recognizable.

"*If*": Let $L \subseteq M_u$ be such that $T = \mathbf{h}_M^{-1}(L)$ is $\mathbf{T}(F)$-recognizable. If s is a sort and $m, m' \in M_s$ then $m \sim_L m'$ if and only if for all $c \in \mathbf{ctxt}(F)_{s,u}$:

$$c_M(m) \in L \Leftrightarrow c_M(m') \in L.$$

But, for every $t \in \mathbf{T}(F)_s$ and c as above:

$$c_M(t_M) \in L \Leftrightarrow c[t] \in \mathbf{h}_M^{-1}(L) = T.$$

Hence for any two terms t and t' of sort s:

$$t_M \sim_L t'_M \quad \text{if and only if } t \sim_T t'.$$

This proves that $\sim_L$ and $\sim_T$ have the same number of classes of each sort. Hence L is recognizable since T is assumed to be so, and furthermore $\mathbf{T}(F)/\sim_T$ is isomorphic to $M/\sim_L$. □

This proposition shows that, in order to decide whether an element m of M belongs to L, it suffices to take *any* term t in $\mathbf{T}(F)$ denoting m and to decide whether it belongs to the recognizable set $\mathbf{h}_M^{-1}(L)$ (for instance by running an automaton on this term). The key point is that the answer is the same for any term t denoting m. This should be contrasted with the characterization of equational sets given in Proposition 3.17 (see also Corollary 5.4), which says that, if L is equational, then it is of the form $\mathbf{h}_M(K)$ for some recognizable set of terms K: in this case, in order to verify that m belongs to L, one must find a term t denoting m and belonging to K. This verification cannot be made from an arbitrary term in $\mathbf{h}_M^{-1}(m)$.

Proposition 4.5. *The emptiness of an effectively given M-recognizable set is not decidable in general. It is decidable under the additional conditions that the signature F is finite, explicitly given and generates M.*

Proof. We first establish the decidability result. Let M be effectively given and generated by a finite explicitly given signature F. If $L \in \mathbf{Rec}(M)_s$, then $\mathbf{h}_M^{-1}(L)$ is an effectively given recognizable subset of $\mathbf{T}(F)$. Its emptiness can be decided by the algorithm of Section 3.3 that decides the emptiness of an equational set since every recognizable set of terms over a finite signature is equational (see Proposition 5.3).

We now consider the undecidability. We give two examples showing that none of the two hypotheses can be omitted.

We first consider the infinite one-sort signature F consisting of a constant a, and of unary functions f_n for all $n \in \mathbb{N}$. Let g be a total recursive mapping $\mathbb{N} \to \{0, 1\}$. Let A be the finite F-magma associated with g as follows:

$$A = \{0, 1\}, \qquad a_A = 0, \qquad f_{nA}(1) = 1, \qquad f_{nA}(0) = g(n).$$

Let $B = \mathbf{h}_A^{-1}(\{1\}) \subseteq \mathbf{T}(F)$. It is effectively $\mathbf{T}(F)$-recognizable. It is clear that $B \neq \emptyset$ if and only if $g(n) = 1$ for some $n \in \mathbb{N}$, and this not decidable. In this example the infinite signature F generates $\mathbf{T}(F)$.

Here is a second similar example where the signature F' is finite but does not generate the relevant magma M. We let F' be the signature reduced to the constant a. Let $M = \langle \mathbb{N}, a_M \rangle$ with $a_M = 0$. Let A and g be as above. The mapping h such that $h(0) = 0$, $h(i) = g(i)$ if $i \geqslant 1$, is a homomorphism: $M \to A$. Hence $h^{-1}(\{1\})$ is an effectively given M-recognizable set. It is nonempty if $g(i) = 1$ for some $i \geqslant 1$. And this is not decidable. □

4.2. Closure properties of **Rec**(M)

In the following propositions M and M' are arbitrary F-magmas, and s is one of their sorts.

Proposition 4.6. *The family of sets* $\mathbf{Rec}(M)_s$ *contains* $\emptyset$, M_s, *and is closed under union, intersection, and difference.*

Proof (*Sketch*). If L_i is recognized by (h_i, A_i, C_i), $i = 1, 2$ then L_1 and L_2 are both recognized by the semi-automaton $(h_1 \times h_2, A_1 \times A_2)$, with respective sets of "final states" $C_1 \times A_2$ and $A_1 \times C_2$. The closure under union, intersection, and difference follows immediately. The other assertions are easy to verify. □

Proposition 4.7. *If* h *is an* F*-homomorphism:* $M \to M'$, *and* L *is* M'*-recognizable, then* $h^{-1}(L)$ *is* M*-recognizable.*

Proof. Follows from Proposition 4.1. □

Proposition 4.8. *If* f *is a mapping:* $M_s \to M_u$ *defined by* $f(d) = t_M(d, d_2, \ldots, d_k)$, *where* t *belongs to* $\mathbf{T}(F, \{x_1, \ldots, x_k\})_u$, $\sigma(x_1) = s$ *and* $d_2, \ldots, d_k$ *are elements of* M *of respective sorts* $\sigma(x_2), \ldots, \sigma(x_k)$, *if* L *is an* M*-recognizable subset of* M_u, *then* $f^{-1}(L)$ *is* M*-recognizable.*

Proof. Consequence of Proposition 4.1 because, if a congruence saturates L then it saturates $f^{-1}(L)$. □

If F and F' are two signatures over a same set of sorts $\mathcal{S}$ and $F' \subseteq F$, if M is an F-magma, then the notation $M' \subseteq M$ means that M' is a sub-F'-magma of M.

Proposition 4.9. (1) *Let* $F' \subseteq F$ *and* $M' \subseteq M$. *For every* $s \in \mathcal{S}$, *if* $L \in \mathbf{Rec}(M)_s$ *then* $L \cap M'_s \in \mathbf{Rec}(M')_s$.

(2) *If* P *is a derived magma of* M, *then for every* $s \in \mathcal{S}$ *we have* $\mathbf{Rec}(M)_s \subseteq \mathbf{Rec}(P)_s$.

We omit the proof which is a straightforward verification from the definitions. The inclusions are strict in general, and M'_s is not necessarily in $\mathbf{Rec}(M)_s$. Note also that, if $M'_s = M_s$ in (1), then $\mathbf{Rec}(M)_s \subseteq \mathbf{Rec}(M')_s$.

Example 4.10. We consider the magma of finite graphs with sources $\mathbb{G}$, defined in Example 2.4. It has infinitely many domains, $\mathbb{G}_i$ for $i = 0, 1, \ldots$. We have also introduced series-composition $\bullet$ as a derived operation on $\mathbb{G}_2$, the set of graphs with two sources. We have thus a derived magma $\mathbb{SP} = \langle SP, \bullet, ||_2, e \rangle$, where SP is the set of series-parallel graphs with two sources defined in Example 3.3, since this set contains e and is stable under the two operations $\bullet$ and $||_2$.

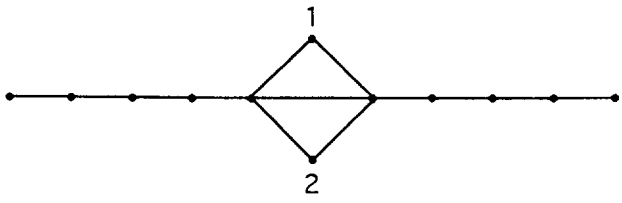

Fig. 2.

It follows then from Proposition 4.9 that if a subset of SP is $\mathbb{G}$- recognizable, then it is $\mathbb{SP}$-recognizable. It *does not* follow from Proposition 4.9 that **Rec**($\mathbb{SP}$) is included in **Rec**($\mathbb{G}$). In particular, it is not immediate that SP is in **Rec**($\mathbb{G}$). These two facts are true however, but the proof given in Courcelle [16] uses other tools than the ones we are discussing. It is *not true* that a subset of $\mathbb{G}_2$ is $\mathbb{G}$-recognizable if it is $\langle \mathbb{G}_2, \bullet, ||_2 \rangle$-recognizable. One can construct a set of graphs L that is $\langle \mathbb{G}_2, \bullet, ||_2 \rangle$-recognizable but not $\mathbb{G}$-recognizable. We let L be the set of graphs shown in Fig. 2, having their two "tails" of the same length. It is $\langle \mathbb{G}_2, \bullet, ||_2 \rangle$-recognizable because its elements cannot be obtained from smaller ones by the operations $\bullet, ||_2$ (see the remark made after Proposition 4.1). It is not $\mathbb{G}$-recognizable because otherwise it would be definable in monadic second-order logic, and so would be the language $\{a^n b^n \mid n > 0\}$, which is not the case. The reader will find in [16] the necessary machinery to complete the proof.

4.3. Concrete automata

The purpose of this subsection is to relate the algebraic notion of an automaton used above to define recognizability with the familiar notions of automata on words and on finite trees (i.e., terms). We shall also introduce a new notion of automaton for dealing with unordered unranked rooted trees. We aim at presenting here a concrete view of the notion of a recognizable set, comparable to the one of an equational set presented in Section 3.5 by means of regular term grammars.

In a certain sense, the notion of a (finite) tree-automaton could suffice since a set is recognizable if and only if the set of terms denoting its members is recognizable, and since recognizable sets of terms are defined by tree-automata. One could thus claim that tree-automata capture completely the notion of recognizability. There is even a *canonical tree-automaton*, namely the unique minimal deterministic tree-automaton for this set of terms. However, this is not fully satisfactory. Intuitively, an automaton is, or should be, a formalization of a recognition algorithm, working on the object itself (a word, a tree, either ordered or unordered, a graph, a pair of words to take a few typical examples) and not on a term denoting the considered object. Letting automata work on terms assumes that the "parsing" has been done beforehand, which is not necessarily easy, in particular in the case of graphs. Hence, doing this would hide an important issue. We shall mainly discuss

examples concerning words and trees. Clearly, there cannot exist any notion of automaton on abstract algebraic objects, but even in the case of series-parallel graphs (Example 3.3), we do not know any notion of automaton that is equivalent to recognizability.

4.3.1. Finite automata on words

Let us consider the unary magma $\mathbb{U}_A = \langle A^*, \varepsilon, \bar{a}_1, \ldots, \bar{a}_n \rangle$ of Example 2.2, where $A = \{a_1, \ldots, a_n\}$, and $\bar{a}_i(u) = ua_i$ for every $u \in A^*$. An automaton relative to $\mathbb{U}_A$ is a triple $(h, \mathbb{Q}, Q^{\text{acc}})$ where $\mathbb{Q} = \langle Q, \varepsilon_{\mathbb{Q}}, a_{1\mathbb{Q}}, \ldots, a_{n\mathbb{Q}} \rangle$ is a $\mathbb{U}_A$-magma, h is a homomorphism: $\mathbb{U}_A \to \mathbb{Q}$, Q is finite and $Q^{\text{acc}} \subseteq Q$.

One can construct from these objects a (usual) *deterministic finite automaton* $\mathscr{B} = \langle A, Q, \delta, \varepsilon_{\mathbb{Q}}, Q^{\text{acc}} \rangle$ with set of states, Q, set of accepting states Q^{acc}, initial state $\varepsilon_{\mathbb{Q}}$ and transition function $\delta : A \times Q \to Q$ defined by $\delta(a_i, q) = a_{i\mathbb{Q}}(q)$. It is not hard to verify that $h^{-1}(Q^{\text{acc}})$ is the language accepted by $\mathscr{B}$. Hence every language in the family $\mathbf{Rec}(\mathbb{U}_A)$ is regular. Conversely, every regular language $L \subseteq A^*$ is accepted by a finite deterministic automaton $\mathscr{B} = \langle A, Q, \delta, q_0, Q^{\text{acc}} \rangle$ having a total transition function $\delta : A \times Q \to Q$. One can make it into a triple $(h, \mathbb{Q}, Q^{\text{acc}})$ by letting:

$$\mathbb{Q} := \langle Q, \varepsilon_{\mathbb{Q}}, a_{1\mathbb{Q}}, \ldots, a_{n\mathbb{Q}} \rangle,$$

$$a_{i\mathbb{Q}}(q) := \delta(a_i, q) \quad \text{for all } q \in Q \text{ and } i = 1, \ldots, n,$$

$$\varepsilon_{\mathbb{Q}} := q_0$$

and h is the mapping: $A^* \to Q$ defined by

$$h(\varepsilon) := q_0,$$

$$h(ua_i) := \delta(a_i, h(u)).$$

Then $(h, \mathbb{Q}, Q^{\text{acc}})$ is an automaton in the sense of Section 4.1 and $h^{-1}(Q^{\text{acc}}) = L$. It follows that a language $L \subseteq A^*$ is regular if and only if it is $\mathbb{U}_A$-recognizable.

In Section 4.1 we have defined the syntactic congruence $\sim_L$ of a subset L of an F-magma M. In the case of $M = \mathbb{U}_A$ and $L \subseteq A^*$ we get the following definition of $\sim_L$:

$u \sim_L v$ if and only if for every integer m, for every $b_1, \ldots, b_m \in A$ we have

$$\bar{b}_1(\bar{b}_2(\ldots(\bar{b}_m(u))\ldots)) \in L \Leftrightarrow \bar{b}_1(\bar{b}_2(\ldots(\bar{b}_m(v))\ldots)) \in L.$$

Hence, from the definition of the mappings $\bar{a}$, for $a \in A$, we get

$u \sim_L v$ if and only if for every $w \in A^*$:

$$uw \in L \Leftrightarrow vw \in L.$$

Hence Proposition 4.1 gives the well-known fact that a language is regular if and only if the $\mathbb{U}_A$-congruence $\sim_L$ is finite [27]. This congruence is called the *least right semi-congruence* because the term "congruences" is reserved for congruences with respect to $\mathbb{W}_A$, and the term "syntactic congruence" is reserved to the congruence $\sim_L$ relative to $\mathbb{W}_A$ (and not to $\mathbb{U}_A$).

Since $\mathbb{U}_A$ is a derived magma of $\mathbb{W}_A$, we get that $\mathbf{Rec}(\mathbb{W}_A) \subseteq \mathbf{Rec}(\mathbb{U}_A)$ by Proposition 4.9. The converse actually holds because, if $\sim$ is a $\mathbb{U}_A$-congruence saturating $L \subseteq A^*$ then the equivalence relation $\approx$ on A^* defined by $u \approx v$ if and only if $wu \sim wv$ for every $w \in A^*$ is a $\mathbb{W}_A$-congruence saturating L, and if $\sim$ has k equivalence classes then $\approx$ has at most k^k classes. It follows that $\mathbf{Rec}(\mathbb{W}_A)$ is equal to $\mathbf{Rec}(\mathbb{U}_A)$.

4.3.2. Finite tree-automata

Tree-automata recognize sets of terms. "Tree" refers to the representation of terms by finite ordered rooted trees. (This terminology is misleading because many trees do not correspond to terms.)

The construction of a deterministic finite automaton from an "algebraic" automaton $(h, \mathbb{Q}, Q^{\mathrm{acc}})$ on $\mathbb{U}_A$ extends to $\mathbf{T}(F)$. Here, F is a finite signature, with possibly several sorts. Let $(h, \mathbb{Q}, Q^{\mathrm{acc}})$ be an ("algebraic") automaton relative to $\mathbf{T}(F)$. This means that $\mathbb{Q}$ is an F-magma $\langle (Q_s)_{s \in \mathcal{S}}, (f_{\mathbb{Q}})_{f \in F} \rangle$, that h is a homomorphism: $\mathbf{T}(F) \to \mathbb{Q}$ (actually the unique one, denoted by $\mathbf{h}_{\mathbb{Q}}$) and that $Q^{\mathrm{acc}} \subseteq Q_u$ for some u. Let $L = h^{-1}(Q^{\mathrm{acc}}) \subseteq \mathbf{T}(F)_u$ be defined by this automaton. One can build the finite tree-automaton $\mathscr{B} = \langle F, Q, \delta, Q^{\mathrm{acc}} \rangle$, where δ is the mapping such that $\delta(f, q_1, \ldots, q_k) = q$ if and only if $f \in F$, $\rho(f) = k$, $q_1, \ldots, q_k \in Q$, and $q = f_{\mathbb{Q}}(q_1, \ldots, q_k)$. If $\rho(f) = 0$ then $\delta(f) = f_{\mathbb{Q}} \in Q$. Given $t \in \mathbf{T}(F)_u$, this tree-automaton can be used as follows to decide whether $t \in L$. A *run of* $\mathscr{B}$ *on* t is mapping $r: N \to Q$, where N is the set of nodes of t such that:

(1) if $v \in N$ is a leaf with label f then $r(v) = \delta(f)$ (we have $\rho(f) = 0$);

(2) if $v \in N$ has k successors $v_1, \ldots, v_k$ and label f (with $\rho(f) = k$), then $r(v) = \delta(f, r(v_1), \ldots, r(v_k))$.

It is clear that since δ is a function, there is one and only one run of $\mathscr{B}$ on t. This run can be constructed by means of a bottom-up traversal of t and such an automaton is usually called *bottom-up* (or sometimes *frontier-to-root*) *deterministic*. We let $\mathbf{L}(\mathscr{B})$ be the set of terms t such that $r(t) \in Q^{\mathrm{acc}}$, where r is the unique run of $\mathscr{B}$ on t.

Fact. *If v is the root of t then $r(v) = h(t)$.*

The proof is easy by induction on the structure of t. It follows that $\mathbf{L}(\mathscr{B}) = h^{-1}(Q^{\mathrm{acc}}) = L$. Conversely, for every finite tree-automaton $\mathscr{B} = \langle F, Q, \delta, Q^{\mathrm{acc}} \rangle$, where δ is a mapping associating a state with every tuple $(f, q_1, \ldots, q_k)$ such that $f \in F$, $\rho(f) = k$, $q_1, \ldots, q_k \in Q$, one constructs an automaton $(h, \mathbb{Q}, Q^{\mathrm{acc}})$ such that $L = h^{-1}(Q^{\mathrm{acc}})$. It follows that $\mathbf{Rec}(\mathbf{T}(F))$ is the class of sets of terms defined by finite bottom-up deterministic tree-automata.

4.3.3. Finite automaton on rooted unordered trees

We now present a class of automata defining the recognizable sets of rooted unordered trees. The background structure is $\langle \mathbb{R}, |\,, \mathbf{ext}, \mathbf{1}\rangle$, introduced in Example 2.3 that we shall more simply denote by $\mathbb{R}$. We need a few preliminary notions.

If Q is a finite set, we denote by $\mathcal{M}(Q)$ the set of finite multisets of elements of Q; we denote by $\emptyset$ the empty multiset and by $\oplus$ the union of multisets. Letting $Q = \{q_1, \ldots, q_n\}$, we have an isomorphism j: $\langle \mathcal{M}(Q), \oplus, \emptyset\rangle \to \langle \mathbb{N}^n, \oplus, \mathbf{0}\rangle$ where $j(M) = (x_1, \ldots, x_n)$ if and only if x_i is the number of occurrences of q_i in M. (Since we use $+$ to denote set union in polynomial systems, we take the symbol $\oplus$ to denote addition of integers and of vectors of integers; we denote by $\mathbf{0}$ the vector $(0, \ldots, 0)$.) The recognizable subsets of $\mathcal{M}(Q)$ can thus be identified, via j, with those of $\mathbb{N}^n$. A subset of $\mathbb{N}$ is *ultimately periodic* if it is of the form $A \cup \{p \oplus \lambda q \mid p \in B, \lambda \in \mathbb{N}\}$, where $q \in \mathbb{N}$ and A and B are finite subsets of $\mathbb{N}$.

Lemma 4.11. *A subset of $\mathbb{N}^n$ is recognizable if and only if it is a finite union of products of the form $K_1 \times \cdots \times K_n$ where each K_i is ultimately periodic.*

Proof. We use two results of [27]: Proposition 12.2 of Chap. III states that a subset of a product monoid $M_1 \times \cdots \times M_n$ is recognizable if and only if it is a finite union of products $K_1 \times \cdots \times K_n$ where each K_i is a recognizable subset of M_i. Proposition 1.1 of Chap. V states that a set of nonnegative integers is recognizable if and only if it is ultimately periodic. The result follows from these two facts. □

An $\mathbb{R}$-*automaton* is a tuple $\mathcal{B} = \langle Q, \delta, Q^{\mathrm{acc}}\rangle$, where Q is the finite set of *states*, $Q^{\mathrm{acc}} \subseteq Q$ is the set of *accepting states* and δ is the *transition relation*, a mapping associating a set $\delta(q) \in \mathbf{Rec}(\mathcal{M}(Q))$ with each q, such that $\mathcal{M}(Q)$ is the union of all the sets $\delta(q)$. We say that $\mathcal{B}$ is *deterministic* if

(1) the sets $\delta(q)$, for $q \in Q$ form a partition of $\mathcal{M}(Q)$ and:

(2) for all $q, q' \in Q$: $\delta(q) \oplus \delta(q') := \{x \oplus x' \mid x \in \delta(q), x' \in \delta(q')\}$ is a subset of $\delta(q'')$ for some $q'' \in Q$.

A *run* of $\mathcal{B}$ on T is a mapping r of N into Q (where N is the set of nodes of T) such that if $r(x) = q$, if $x_1, \ldots, x_m$ are the successors of x, then the multiset $\{r(x_1), \ldots, r(x_m)\}$ belongs to $\delta(q)$. (This condition also applies when x is a leaf: in this case, $m = 0$ and $\{r(x_1), \ldots, r(x_m)\}$ is the empty multiset.) We call $r(x)$ the *root-state* of the run r where x is the root of the tree. Then $\mathbf{L}(\mathcal{B})$ is defined as the set of trees T such that there is a run of $\mathcal{B}$ on T, the root-state of which is accepting. On each tree T in $\mathbb{R}$ there is at least one run (one can construct one by traversing T from the leaves to the root). If $\mathcal{B}$ is deterministic there is a unique run on each tree.

Proposition 4.12. *For every subset L of $\mathbb{R}$ the following conditions are equivalent:*

(1) $L \in \mathbf{Rec}(\mathbb{R})$,

(2) $L = \mathbf{L}(\mathcal{B})$ *for some deterministic $\mathbb{R}$-automaton $\mathcal{B}$,*

(3) $L = \mathbf{L}(\mathcal{B})$ *for some $\mathbb{R}$-automaton $\mathcal{B}$.*

Proof. (1) $\Rightarrow$ (2): Let $L \in \mathbf{Rec}(\mathbb{R})$. Then $L = h^{-1}(Q^{\mathrm{acc}})$, where h is a homomorphism: $\mathbb{R} \to \mathbb{Q} = \langle Q, ||_{\mathbb{Q}}, \mathbf{ext}_{\mathbb{Q}}, \mathbf{1}_{\mathbb{Q}} \rangle$, $\mathbf{Q}$ is finite and $Q^{\mathrm{acc}} \subseteq Q$. We can assume that h is surjective (otherwise one replaces $\mathbb{Q}$ by the submagma $h(\mathbb{R})$). Hence, $||_{\mathbb{Q}}$ is associative, commutative and has unit $\mathbf{1}_{\mathbb{Q}}$: the verification is easy by using the corresponding properties of $||_{\mathbb{R}}$ and $\mathbf{1}_{\mathbb{R}}$ and the fact that h is a surjective homomorphism. We shall use the infix notation without parentheses for $||_{\mathbb{Q}}$.

We now build an $\mathbb{R}$-automaton. We let $\mathscr{B} = \langle Q, \delta, Q^{\mathrm{acc}} \rangle$ with $\delta(q) := \theta^{-1}(q)$ for every $q \in Q$, where θ is the mapping: $\mathscr{M}(Q) \to Q$ defined as follows: $\theta(\emptyset) := \mathbf{1}_{\mathbb{Q}}$ and for every nonempty multiset $M = \{q_{i_1}, \ldots, q_{i_m}\}$:

$$\theta(M) = \mathbf{ext}_{\mathbb{Q}}(q_{i_1}) ||_{\mathbb{Q}} \mathbf{ext}_{\mathbb{Q}}(q_{i_2}) ||_{\mathbb{Q}} \cdots ||_{\mathbb{Q}} \mathbf{ext}_{\mathbb{Q}}(q_{i_m})$$

(this is well-defined since $||_{\mathbb{Q}}$ is associative and commutative). Each set $\delta(q)$ is $\mathscr{M}(Q)$-recognizable: this follows from the observation that θ is a homomorphism: $\langle \mathscr{M}(Q), \oplus, \emptyset \rangle \to \langle Q, ||_{\mathbb{Q}}, \mathbf{1}_{\mathbb{Q}} \rangle$ and from the finiteness of Q. Hence the sets $\theta^{-1}(q)$ are recognizable and pairwise disjoint. They form a partition of $\mathscr{M}(Q)$. It is now easy to check (by induction on the size of T) that for every $T \in \mathbb{R}$, $h(T)$ is the root-state of the unique run of $\mathscr{B}$ on T. Hence $L = \mathbf{L}(\mathscr{B})$ as desired.

In order to prove that $\mathscr{B}$ is deterministic, we verify that $\delta(q) \oplus \delta(q') \subseteq \delta(q ||_{\mathbb{Q}} q')$. We let M belong to $\delta(q)$ and M' to $\delta(q')$. Then $\theta(M \oplus M') = \theta(M) ||_{\mathbb{Q}} \theta(M') = q ||_{\mathbb{Q}} q'$. Hence hence $M \oplus M' \in \delta(q ||_{\mathbb{Q}} q')$ as desired.

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

(3) $\Rightarrow$ (1) Let $L = \mathbf{L}(\mathscr{B})$ for some $\mathbb{R}$-automaton $\mathscr{B} = \langle Q, \delta, Q^{\mathrm{acc}} \rangle$.
There exists (by the proof of Proposition 4.6), a homomorphism $k : \mathscr{M}(Q) \to \langle P, \oplus_P, \mathbf{1}_P \rangle$ with P finite, and $P_q \subseteq P$ for every q in Q such that $\delta(q) = k^{-1}(P_q)$; so we have a single semi-automaton $(k, \langle P, \oplus_P, \mathbf{1}_P \rangle)$ defining simultaneously all the sets $\delta(q)$.

For every $T \in \mathbb{R}$ we let $\mathbf{Sub}(T)$ denote the set of concrete subtrees of T issued from the successors of the roots. We have $\mathbf{Sub}(\mathbf{1}) = \emptyset$ and $\mathbf{Sub}(\mathbf{ext}(T)) = \{T\}$ for every $T \in \mathbb{R}$. For every run r of $\mathscr{B}$ on T we let $r(\mathbf{Sub}(T))$ denote the multiset of root-states of this run on the trees in $\mathbf{Sub}(T)$. We let $h(T) = \{k(r(\mathbf{Sub}(T))) \mid r \text{ is a run of } \mathscr{B} \text{ on } T\}$. Hence h maps $\mathbb{R}$ into $\wp(P)$. We claim that h is a homomorphism of $\mathbb{R}$ into $\mathbb{P} = \langle \wp(P), \oplus_{\mathbb{P}}, \mathbf{ext}_{\mathbb{P}}, \{\mathbf{1}_P\} \rangle$, where

$$A \oplus_{\mathbb{P}} B := \{a \oplus_P b \mid a \in A, b \in B\},$$

$$\mathbf{ext}_{\mathbb{P}}(A) := \{k(\{q\}) \mid A \cap P_q \text{ is not empty}\}.$$

We now verify this claim. If $T = \mathbf{1}$ then $\mathbf{Sub}(T) = \emptyset$ and $h(T) = \{k(\emptyset)\} = \{\mathbf{1}_P\}$. Let $T = T' \parallel T''$; let $p \in h(T)$; then $p = k(M)$, where $M = r(\mathbf{Sub}(T))$ for some run r of $\mathscr{B}$ on T. We have $\mathbf{Sub}(T) = \mathbf{Sub}(T') \cup \mathbf{Sub}(T'')$ and $M = r(\mathbf{Sub}(T')) \oplus r(\mathbf{Sub}(T''))$. Hence $k(M) = k(r(\mathbf{Sub}(\mathbf{T}'))) \oplus_P k(r(\mathbf{Sub}(T')))$ and belongs to $h(T') \oplus_{\mathbb{P}} h(T'')$. Hence, we have $h(T' \parallel T'') \subseteq h(T') \oplus_{\mathbb{P}} h(T'')$. The proof of the other inclusion is similar. Let now $T = \mathbf{ext}(T')$. We need to prove that $h(T) = \mathbf{ext}_{\mathbb{P}}(h(T'))$. Let $p \in h(T)$; then $p = k(\{q\})$, where $q = r(v)$, v is the root of T' and r is some run of $\mathscr{B}$ on T. We let M be the multiset $r'(\mathbf{Sub}(T'))$. Then M belongs to $\delta(q)$ hence $k(M)$ belongs to P_q. But $k(M)$ belongs to

$h(T')$. Hence $h(T') \cap P_q$ is nonempty, and p belongs to $\mathbf{exp}_{\mathbb{P}}(h(T'))$ as was to be proved. The proof of the opposite inclusion is similar.

Finally, we let ACC be the set of subsets of P that have a nonempty intersection with the union of the sets P_q for q in Q^{acc}. We claim that $\mathbf{L}(\mathscr{B}) = h^{-1}(ACC)$, which will complete the proof that $\mathbf{L}(\mathscr{B})$ is $\mathbb{R}$-recognizable. Let T be a tree in $\mathbf{L}(\mathscr{B})$, and r be a run of $\mathscr{B}$ on T with root-state q in Q^{acc}. Let $M = r(\mathbf{Sub}(T))$. Then $M \in \delta(q)$ hence $k(M) \in P_q$. By $k(M) \in h(T)$. Hence $h(T) \in ACC$. The proof in the other direction is similar. □

Proposition 4.13. *Let $L \in \mathbf{Rec}(\mathbb{R})$. The $\mathbb{R}$-automaton associated with the minimal automaton of L is the unique deterministic $\mathbb{R}$-automaton defining L having a minimum number of states.*

Proof. In the construction of (1) $\Rightarrow$ (2) in the proof of Proposition 4.12, one can use the minimal automaton of L, i.e., the quotient $\mathbb{Q} = \mathbb{R}/\sim_L$, where $\sim_L$ is the syntactical congruence of L (see Section 4.1). One obtains a deterministic $\mathbb{R}$-automaton with set of states in bijection with $\mathbb{R}/\sim_L$. Conversely, for every deterministic $\mathbb{R}$-automaton $\mathscr{B} = \langle Q, \delta, Q^{acc}\rangle$ such that $\mathbf{L}(\mathscr{B}) = L$, the equivalence relation on $\mathbb{R}$ defined by

$$T \sim T' \text{ if and only if } \mathbf{q}(T) = \mathbf{q}(T'),$$

where $q(T)$ is the root-state of the unique run of $\mathscr{B}$ on T, is a congruence that saturates L. Let us check this point. Let $\mathbf{q}(T) = \mathbf{q}(T') = q$. Then $\mathbf{q}(\mathbf{ext}(T)) = \mathbf{q}(\mathbf{ext}(T'))$ because $\{q\}$ belongs to a unique set $\delta(q')$ and $\mathbf{q}(\mathbf{ext}(T)) = q' = \mathbf{q}(\mathbf{ext}(T'))$. Let now $\mathbf{q}(T_1) = \mathbf{q}(T_1') = q_1$ and $q(T_2) = q(\mathbf{T}_2') = q_2$. Let $q = \mathbf{q}(T_1 \,||\, T_2)$ and $q' = \mathbf{q}(T_1' \,||\, T_2')$. From the condition that $\delta(q_1) \oplus \delta(q_2) \subseteq \delta(q'')$ for a unique q'' (since the automaton is deterministic), we get $q = q'' = q'$. We have thus a bijection between deterministic $\mathbb{R}$-automata and finite congruences saturating L. Since $\sim_L$ is the unique one with a minimal number of classes, we get the desired unicity result. □

5. Relationships between equational sets and recognizable sets

The main theorem of this section is Theorem 5.1. It generalizes the result that the intersection of a context-free language and a regular one is context-free. It is fundamental for the study of context-free graph grammars.

Theorem 5.1. *Let M be an F-magma and s be a sort. If $K \in \mathbf{Rec}(M)_s$ and $L \in \mathbf{Equat}(M)_s$ then $L \cap K \in \mathbf{Equat}(M)_s$.*

Proof. By Proposition 3.19 we can assume that $L = \cup\{\mathbf{L}((S, M), u) \mid u \in U'\}$, where S is a uniform polynomial system over F with set of unknown U, and $U' \subseteq U$. (We recall from Proposition 3.6 that a polynomial system is *uniform* if its equations are of

the form $u = t_1 + t_2 + \cdots + t_k$, where each t_i is of the form $f(w_1, w_2, \ldots, w_n)$ for some $f \in F$, some unknowns $w_1, \ldots, w_n \in U$.)

Let $F' \subseteq F$ be the finite set of symbols occurring in S, and let $\mathscr{S}' \subseteq \mathscr{S}$ be the finite set of sorts of these symbols together with those of the unknowns of S. Hence F' is an $\mathscr{S}'$-signature. Let $h: M \to A$ be an F'-homomorphism (with A locally finite), such that $K = h^{-1}(C)$ for some $C \subseteq A_s$. For every $u \in U$, we let $L_u := \mathbf{L}((S, M), u)$. We let W be the new set of unknowns $\{[u, a] \mid u \in U, a \in A_{\sigma(u)}\}$. It is finite. We shall define a system S', with set of unknowns W, such that $\mathbf{L}((S', M), [u, a]) = L_u \cap h^{-1}(a)$ for every $[u, a] \in W$.

Let $u \in U$ and $a \in A_{\sigma(u)}$. Let us assume that the defining equation of u in S is of the form $u = t_1 + \cdots + t_k$. Consider one of the monomials, say t_i. Let us assume that it is of the form $f(w_1, \ldots, w_n)$ for some unknowns $w_1, \ldots, w_n$. For every $a_1 \in A_{\sigma(w_1)}, \ldots, a_n \in A_{\sigma(w_n)}$ such that $f_A(a_1, \ldots, a_n) = a$, we form the monomial $f([w_1, a_1], \ldots, [w_n, a_n])$, and we let $\hat{t}_i$ denote the sum of these monomials. If no such n-tuple $(a_1, \ldots, a_n)$ exists, then $\hat{t}_i$ is defined as Ω. The defining equation of $[u, a]$ in S' is taken as

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

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

In order to establish the opposite inclusion, we define from $L_{u,a}$ the sets $L'_u = \bigcup \{L_{u,a} \mid a \in A_{\sigma(u)}\}$ for $u \in U$. Then $(L'_u)_{u \in U}$ is a solution of S in M (this is easy to verify). Hence $L_u \subseteq L'_u$ for all u. For all $a \in A_{\sigma(u)}$, we have

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

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

$$\begin{aligned} L \cap K &= (\bigcup \{L_u \mid u \in U'\}) \cap h^{-1}(C) \\ &= \bigcup \{\mathbf{L}((S', M), [u, a]) \mid u \in U', a \in C\}. \end{aligned}$$

Hence $L \cap K \in \mathbf{Equat}(M)_s$. □

The above construction is effective if K is effectively given, and L is defined by a given system. Hence, since the emptiness of an equational set (defined by a system of equations) is decidable, we have the following corollary that can be contrasted with the undecidability result of Proposition 4.5.

Corollary 5.2. *If K is an effectively given M-recognizable set, and if L is an M-equational set, one can test whether $L \cap K = \emptyset$, or whether $L \subseteq K$.*

The following two results are also due to Mezei and Wright [38].

Proposition 5.3. *If F is finite then* $\mathbf{Rec}(\mathbf{T}(F)) = \mathbf{Equat}(\mathbf{T}(F))$.

Proof (*Sketch*). Every recognizable subset K of $\mathbf{T}(F)$ is equational: it is not hard to transform a finite tree-automaton defining K into a regular term grammar; hence K is equational.

Let us consider conversely an equational subset L of $\mathbf{T}(F)$ given by a uniform polynomial system S; let us convert S into a regular term grammar; it is not hard to transform this grammar into a finite tree-automaton defining L. This automaton is not frontier-to-root deterministic in general. It can be transformed into an equivalent deterministic one; hence, L is recognizable. We refer the reader to [29] for details. □

Corollary 5.4. *A subset L of M_s is M-equational if and only if $L = \mathbf{h}_M(T)$ for some recognizable subset T of $\mathbf{T}(F')_s$, where F' is a finite subset of F.*

Proof. The "if" direction follows from Propositions 3.7 and 5.3. The "only if" direction follows from Proposition 5.3 and Corollary 3.8. □

In the following corollary, $\mathbf{Rec}(M) \subseteq \mathbf{Equat}(M)$ means: $\mathbf{Rec}(M)_s \subseteq \mathbf{Equat}(M)_s$ for all s in $\mathcal{S}$.

Corollary 5.5. *Let M be an F-magma generated by F. Then* $\mathbf{Rec}(M) \subseteq \mathbf{Equat}(M)$ *if and only if for every $s \in \mathcal{S}$, there exists a finite subset F' of F such that* $\mathbf{h}_M(\mathbf{T}(F')_s) = M_s$

Proof. "*If*": Let $L \in \mathbf{Rec}(M)$, let F' be a finite subsignature of F such that $\mathbf{h}_M(\mathbf{T}(F')_s) = M_s$. Then $T = \mathbf{h}_M^{-1}(L) \cap \mathbf{T}(F')_s \in \mathbf{Rec}(\mathbf{T}(F'))_s$ (since $\mathbf{h}_M^{-1}(L) \in \mathbf{Rec}(\mathbf{T}(F'))_s$). Hence $L = \mathbf{h}_M(T)$ and is M-equational by Corollary (5.4).

"*Only if*": *Let* $\mathbf{Rec}(M) \subseteq \mathbf{Equat}(M)$. Then $M_s \in \mathbf{Equat}(M)$ and $M_s = \mathbf{h}_M(T')$ for some $T' \in \mathbf{Rec}(\mathbf{T}(F'))_s$ with F' finite, $F' \subseteq F$. Hence $M_s = \mathbf{h}_M(\mathbf{T}(F'))_s$. □

We conclude this section with a few remarks on the difficulties of generalizing the notion of a rational subset of a monoid to arbitrary magmas. Let $F = \{\bullet, e, a_1, \ldots, a_k, \ldots\}$, where $\bullet$ is binary and $e, a_1, \ldots, a_k, \ldots$ are nullary. Let M be an F-magma that is a monoid for $\bullet_M$ with unit element e_M. For every $L \subseteq M$, one lets

$$L^* = \{e_M\} \cup L \cup L^2 \cup \cdots \cup L^n \cup \cdots, \tag{5.1}$$

where $L^n := L \bullet L \bullet \cdots \bullet L$ (with n times L), so that L^* is the least solution of the equation

$$X = \{e_M\} \cup L \bullet X \tag{5.2}$$

or equivalently of the equations $X = \{e_M\} \cup X \bullet L$ and $X = \{e_M\} \cup L \cup X \bullet X$.

The mapping $L \mapsto L^*$ is called the *star* operation. One defines $\mathbf{Rat}(M)$, the class of *rational subsets of* M as the least family of subsets of M that contains the singletons defined by $e, a_1, \ldots, a_k, \ldots$ and that is closed under union, $\bullet$ and star. These sets are the values of the *rational expressions* which are defined inductively as follows:

(1) $e, a_1, \ldots, a_k, \ldots$ are rational expressions denoting $\{e_M\}, \{a_{1_M}\}, \ldots, \{a_{k_M}\}, \ldots$ respectively,

(2) if E_1, E_2 are rational expressions denoting, respectively, L_1 and $L_2 \subseteq M$, then $(E_1)\bullet(E_2)$, $(E_1) + (E_2)$ and $(E_1)^*$ are rational expressions denoting, respectively, $L_1 \bullet L_2$, $L_1 \cup L_2$ and L_1^*.

Proposition 5.6. *For every monoid M, we have the inclusion:*

$$\mathbf{Rat}(M) \subseteq \mathbf{Equat}(M).$$

Proof. If $L = \mathbf{L}((S, M), u)$ then $L^* = \mathbf{L}((S', M), u')$, where $S' = S \cup \{u' = e + u \bullet u'\}$ and u' is an unknown that has no occurrence in S. This can be easily verified from Corollary 3.8. The result follows then from Proposition 3.23 by means of an induction on the structure of a rational expression denoting a set that we want to prove to be in $\mathbf{Equat}(M)$. □

If $M = \mathbb{W}_A$ and A has at least 2 elements then $\mathbf{Rat}(M)$ is equal to $\mathbf{Rec}(M)$, i.e., is the family of regular languages, and is properly included in $\mathbf{Equat}(M)$.

A natural question is the following: what could be the notion of a rational subset of an arbitrary F-magma? There is actually no unique natural generalization of the star operation. For any polynomial $p(x, y)$ in two variables, one can define a mapping $\wp(M) \to \wp(M)$, denoted by $\lambda X \mu Y.p(X, Y)$ such that for every subset L of M, $\mu Y.p(L, Y)$ is the least solution of the equation $Y = p(L, Y)$. A system of notations for least solutions of polynomial systems does exist and is called the *μ-calculus* [39]. However, it is too powerful for the purpose of obtaining an extension of Kleene's Theorem by means of "generalized rational" expressions defining the recognizable sets. It is too powerful because its terms, written with set union, set extensions of the base operations and *minimization* (the above construction $\mu Y.t$) define the whole class $\mathbf{Equat}(M)$. Furthermore, the full μ-calculus contains also terms written with intersection and maximization operators (defining greatest fixed points) which define sets that are not even equational. We refer the reader to [39] for more details about this language.

6. Inductively computable functions and Parikh's theorem

Inductively computable functions constitute an extension of recognizability. They "fit well" with equational sets, as shown by Theorems 6.3 and 6.7 that are somewhat similar to Theorem 5.1. As a corollary, we obtain a generalized form of Parikh's Theorem on the commutative images of context-free languages. This generalization is

useful for graph grammars. Inductively computable functions are fundamental for certain constructions of linear graph algorithms.

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

$$e(f_M(d_1,\ldots,d_k)) = g_{e,f}(e_1^1(d_1),\ldots,e_{n_1}^1(d_1), e_1^2(d_2),\ldots,e_1^k(d_k),\ldots,e_{n_k}^k(d_k)). \tag{6.1}$$

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

$$(g_{e,f}, (e_1^1,\ldots,e_{n_1}^1), (e_1^2,\ldots,e_{n_2}^2),\ldots,(e_1^k,\ldots,e_{n_k}^k)), \tag{6.2}$$

the *decomposition of e relative to f_M*. In Section 4.1, we have introduced inductive family of predicates: they are just the special case of inductive families of evaluations where $N = \{\textbf{true}, \textbf{false}\}$. If $\mathscr{E}$ is F-inductive, if $d \in M$ is given as t_M for some $t \in \mathbf{T}(F)$ (i.e., this means that d has been "parsed" in terms of the operations of M), then one can evaluate $e(d)$ by the following algorithm using two traversals of t. (We assume that $\mathscr{E}$ is finite.) We denote by t/u the subtree of t issued of u; it is also a term.

Algorithm 6.1.

Input: a term t given as a tree, an evaluation e_0 in $\mathscr{E}$;

Output: the value $e_0(t_M)$.

Method

First traversal (Root-to-frontier): One associates with every node u of the tree t a set of evaluations $\mathscr{E}(u)$, that will have to be "computed at u", i.e., for the argument $(t/u)_M$.

For the root r, we let $\mathscr{E}(r) := \{e_0\}$.

For a node u such that $\mathscr{E}(u)$ is already known and that has successors $u1,\ldots,uk$, then, for every e in $\mathscr{E}(u)$, if f is the operation labelling u, and

$$(g, (e_1^1,\ldots,e_{n_1}^1), (e_1^2,\ldots,e_{n_2}^2),\ldots,(e_1^k,\ldots,e_{n_k}^k)), \tag{6.3}$$

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

Second traversal (Frontier-to-root): Starting from the leaves, one computes at each node u the value $e((t/u)_M)$ of the functions $e \in \mathscr{E}(u)$. We use the formula:

$$e((t/u)_M) := g(e_1^1((t/u1)_M),\ldots,e_{n_1}^1((t/u1)_M),\ldots,e_1^k((t/uk)_M),\ldots,e_{n_k}^k((t/uk)_M))$$

based on the decomposition of e relative to f_M (see (6.3)). One obtains at the root the desired value $e_0(t_M)$.

This technique is useful for certain algorithms, taking as input "structured graphs" i.e., graphs expressed as values of algebraic expressions written with appropriate graph operations, typically those introduced in Example 2.4.

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

(H1) $\mathscr{E}$ has finitely many mappings of each arity $s \in \mathscr{s}$,

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

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

Theorem 6.2. *If conditions* (H1)–(H3) *hold, if in Equalities* (6.1) *we have* $n_1 \leqslant 1, \ldots, n_k \leqslant 1$ *and if the terms* $t_{e',f}$ *are all linear, then* $e(L) \in \mathbf{Equat}(N)$ *for every* $L \in \mathbf{Equat}(M)$ *and every* e *in* $\mathscr{E}$.

Proof. Let $L = \mathbf{L}((S, M), u_1)$, where S is a uniform polynomial system with unknowns $u_1, \ldots, u_n$. We shall assume that M and N have only one sort: this simplifies the notation and is not a loss of generality.

For every $e \in \mathscr{E}$ and $i \in [1, n]$, we let $[e, u_i]$ be a new unknown. We shall build a polynomial system S' with these new unknowns such that the component of its least solution corresponding to $[e, u_i]$ is the set $e(\mathbf{L}((S, M), u_i))$. For every equation

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

of S, every $e \in \mathscr{E}$, we create the equation of S':

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

where $(g, (e^1), \ldots, (e^k))$ is the decomposition of e relative to f_M and $t_{e,f}$ is a linear term over H defining in N the function g.

We let $(A_1^j, \ldots, A_n^j)$ be the jth iterate (where $j \geqslant 0$) approximating the least solution of S in $\wp(M)$ (see the proof of Proposition 3.5). Similarly, we denote by $A_{e,i}^j$ for $e \in \mathscr{E}$ and $i \in [1, n]$ the component corresponding to $[e, u_i]$ of the jth iterate of $S'_{\wp(N)}$.

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

Proof. By induction on j. The case $j = 0$ is clear: both sets are empty. Let us establish the case $j + 1$ assuming the case j. Let $a \in e(A_i^{j+1})$. We have $a = e(f_M(d_1, \ldots, d_k))$ for some $d_1 \in A_{i_1}^j, \ldots, d_k \in A_{i_k}^j$ where the equation defining u_i is $u_i = \cdots + f(u_{i_1}, \ldots, u_{i_k}) + \cdots$. Hence $a = t_{e,f_N}(e^1(d_1), \ldots, e^k(d_k))$ where $t_{e,f}, e^1, \ldots, e^k$ are as above; we have

$$e^1(d_1) \in e^1(A_{i_1}^j) = A_{e^1,i_1}^j \quad , \ldots, \quad e^k(d_k) \in e^k(A_{i_k}^j) = A_{e^k,i_k}^j$$

using the induction hypothesis, hence $a \in A_{e,i}^{j+1}$ by the definition of S'. Hence we have proved

$$e(A_i^{j+1}) \subseteq A_{e,i}^{j+1}.$$

The proof of the other inclusion is similar. If $a' = t_{e,f_N}(a'_1,\ldots,a'_k)$ with $a'_1 \in A^j_{e^1,i_1},\ldots,a'_k \in A^j_{e^k,i_k}$, then we have $a'_1 = e^1(d_1),\ldots,a'_k = e^k(d_k)$, $d_1 \in A^j_{i_1},\ldots,d_k \in A^j_{i_k}$, we can take $d = f_M(d_1,\ldots,d_k) \in A_i^{j+1}$ and $a' = e(d) \in e(A_i^{j+1})$. □

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

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

where $b \in \mathbb{N}^q$. Since b can be written as a finite sum of constants $\mathbf{0}, \mathbf{1}_1,\ldots,\mathbf{1}_q$ the operation $g_{e,f}$ is linearly derived. It follows that Equalities (6.1) have the form:

$$e(f_M(d_1,\ldots,d_k)) = e_1(d_1) \oplus \cdots \oplus e_k(d_k) \oplus b, \tag{6.4}$$

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

We recall a few definitions. A set $A \subseteq \mathbb{N}^q$ is *linear* if it is of the form $A = \{\lambda_1 a_1 \oplus \cdots \oplus \lambda_n a_n \oplus b / \lambda_1,\ldots,\lambda_n \in \mathbb{N}\}$ for some $a_1,\ldots,a_n$, $b \in \mathbb{N}^q$ and where for $\lambda \in \mathbb{N}$, $a \in \mathbb{N}^q$, $\lambda a = \mathbf{0}$ if $\lambda = 0$ and $\lambda a = a \oplus a \oplus \cdots \oplus a$ with λ times a if $\lambda \geqslant 1$. A subset of $\mathbb{N}^q$ is *semi-linear* if it is a finite union of linear sets. Since a linear set as above can be written $A = a_1^* a_2^* \ldots a_n^* b$ it follows that every linear set, hence, every semi-linear set is rational. Conversely, by using the laws:

$$(A + B)^* = A^* B^*,$$

$$(A^* B)^* = \varepsilon + A^* B^* B,$$

which hold for arbitrary subsets A and B of a commutative monoid, one can transform a rational expression defining $A \in \mathbf{Rat}(\mathbb{N}^q)$ into a sum of terms of the form $a_1^* a_2^* \ldots a_n^* b$ for $a_1,\ldots,a_n$, $b \in \mathbb{N}^q$. Hence, every rational subset of $\mathbb{N}^q$ is semi-linear.

Corollary 6.3. *If $L \in \mathbf{Equat}(M)$ and e belongs to a Parikh family of evaluations, then $e(L) \in \mathbf{Equat}(\mathbb{N}^q)$ and is semi-linear.*

Proof. That $e(L) \in \mathbf{Equat}(\mathbb{N}^q)$ follows from Proposition 6.2. Pilling has proved in [41] that every equational set of a commutative monoid is rational, hence, we get that $e(L) \in \mathbf{Rat}(\mathbb{N}^q)$, hence is semi-linear. □

Example 6.4. We let $A = \{a, b, c\}$. We consider a context-free language $L \subseteq A^*$. For every $u \in A^*$, we let $e(u) = (|u|_a, |u|_b, |u|_c) \in \mathbb{N}^3$ where $|u|_x$ is the number of occurrences of x in u. Then $\{e\}$ is a Parikh family of evaluations since

$$e(u \bullet u') = e(u) \oplus e(u')$$

(in (6.4) we have $k = 2$ and $b = \mathbf{0}$ where f is the concatenation of words). The system of equations associated by the proof of Theorem 6.2 and Corollary 6.3 to the system of Example 3.1 is then

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

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

Example 6.5. We consider the magma of series–parallel graphs $\langle \mathbb{SP}, \bullet, ||_2, e \rangle$ of Examples 2.4, 3.3, 4.10. We consider the evaluation $\# : \mathbb{SP} \to \mathbb{N}^2$ such that

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

Then

$$\#(G ||_2 G') = \#(G) \oplus \#(G'),$$

$$\#(G \bullet G') = \#(G) \oplus \#(G') \oplus \mathbf{1}_2.$$

Hence from the equation

$$u = u||u + u.u.u + e, \tag{6.5}$$

which defines a subset L of $\mathbb{SP}$, we get the following equation that defines $\#(L) \subseteq \mathbb{N}^2$:

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

Example 6.6. One cannot omit the linearity assumptions of Theorem 6.2. Here is a counterexample. Let S be the equation over some $\{f, h\}$-magma M:

$$u = f(u) + h$$

and e be an evaluation: $M \to \mathbb{N}$ such that, for every $u \in M$:

$$e(h) = 1,$$

$$e(f(u)) = e(u) \oplus e(u) = 2e(u).$$

(We recall that $\oplus$ denotes the addition of integers and $+$ denotes set union.) The corresponding equation that defines $e(\mathbf{L}((S, M), u))$ is

$$u = 2.u + \mathbf{1}_1,$$

with solution $\{2^n | n \geqslant 0\}$ which is not a semi-linear set of integers.

We now present another extension of Parikh's theorem which has applications in graph grammars. We let M be an F-magma, we let N be a G-magma, we let $\mathscr{E}$ be a finite family of mappings: $\wp(M) \to \wp(N)$ satisfying the following conditions, for all $f \in F$, for all subsets $A_1, \ldots, A_i, \ldots$ of M of appropriate sort:

(P1) $e(A_1 \cup A_2 \cup \cdots) = e(A_1) \cup e(A_2) \cup \cdots$

(P2) $e(f_{\wp(M)}(A_1, \ldots, A_k)) = \bigcup_{1 \leqslant i \leqslant m} t_{i\wp(N)}(e_{i,1}(A_1), e_{i,2}(A_2), \ldots, e_{i,k}(A_k))$,

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

Theorem 6.7. *If $\mathscr{E}$ satisfies conditions* (P1) *and* (P2), *if e is an evaluation of $\mathscr{E}$, for every $L \in \mathbf{Equat}(M)$ we have $e(L) \in \mathbf{Equat}(N)$.*

Proof. Essentially the same as the proof of Theorem 6.2 □

Example 6.8. As in Example 6.5 we use series–parallel graphs, but directed ones: this means that the basic graph e is a single edge directed from the first source to the second one. Directed series–parallel graphs are thus acyclic. We shall denote their set by $\mathbb{SP}'$.

For every graph G in $\mathbb{SP}'$, we let $\pi(G)$ be the set of lengths of directed paths in G that link the first source to the second one. Hence π maps $\mathbb{SP}'$ into $\wp(\mathbb{N})$. For $L \subseteq \mathbb{SP}'$, we let $\Pi(L) = \bigcup\{\pi(G) \mid G \in L\}$. This mapping satisfies Condition (P1) because it is a homomorphism for union and Condition (P2) because

$$\pi(G \parallel G') = \pi(G) \cup \pi(G') = t_{1\wp(\mathbb{N})}(\pi(G)) \cup t_{2\wp(\mathbb{N})}(\pi(G')),$$

$$\pi(G \bullet G') = \pi(G) \oplus \pi(G'),$$

$$= \{n \oplus n' \mid n \in \pi(G), n' \in \pi(G')\}$$

$$= t_{3\wp(\mathbb{N})}(\pi(G), \pi(G')),$$

where $t_1 = x_1$, $t_2 = x_2$ and $t_3 = x_1 \oplus x_2$. For Eq. (6.5) of Example 6.5 we obtain

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

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

7. Guide to the literature

We review in Table 1 the main facts concerning the equational, recognizable and rational sets (when they are defined) of some basic types of finite objects.

The sets of pairs of words defined by finite deterministic two-tape automata are properly inbetween $\mathbf{Equat}(\mathbb{U}_A \times \mathbb{U}_B)$ and $\mathbf{Rec}(\mathbb{U}_A \times \mathbb{U}_B)$; see [12].

We now review the various sections of the paper, we comment on the results, list some applications and indicate what we consider to be the main references.

Table 1

Structure	Main results
$\mathbb{N} = \langle \mathbb{N}, \oplus, \mathbf{0}, \mathbf{1} \rangle = \mathbb{W}_{\{a\}}$ (One-letter words)	**Equat**($\mathbb{N}$) = **Rat**($\mathbb{N}$) = **Rec**($\mathbb{N}$) = the ultimately periodic subsets of $\mathbb{N}$ = the regular languages on $\{a\}$
$\mathbb{W}_A$ (Monoid of words on A)	**Equat**($\mathbb{W}_A$) = the context-free languages; **Rat**($\mathbb{W}_A$) = **Rec**($\mathbb{W}_A$) = the regular languages on A
$\mathbb{W}_{A,R}$ (Monoid of partially commutative words, also called *traces*)	**Rat**($\mathbb{W}_{A,R}$) $\supset$ **Rec**($\mathbb{W}_{A,R}$) = the sets defined by the cellular asynchronous automata of Zielonka [9, 47]. There is no unique minimal such automaton defining a recognizable set of traces [8].
$\mathbb{N}^n = \langle \mathbb{N}^n, \oplus, \mathbf{0}, \mathbf{1}_1, \ldots, \mathbf{1}_n \rangle$ = the finite multisets of elements of $\{a_1, \ldots, a_n\}$ = the free commutative monoid generated by $\{a_1, \ldots, a_n\}$	**Equat**($\mathbb{N}^n$) = **Rat**($\mathbb{N}^n$) by [41]; **Rec**($\mathbb{N}^n$) is the set of finite unions of products of sets of **Rec**($\mathbb{N}$) (Section 4.3.3).
$\mathbb{U}_A$ (Words with right concatenation by letters)	**Equat**($\mathbb{U}_A$) = **Rec**($\mathbb{U}_A$) = the regular languages on A. The minimal deterministic automaton corresponds to the syntactical congruence w.r.t. $\mathbb{U}_A$, i.e., to the least right semi-congruence.
$\mathbb{U}_A \times \mathbb{U}_B$ (Pairs of words with right concatenation by letters)	**Equat**($\mathbb{U}_A \times \mathbb{U}_B$) = the sets of pairs of words $(u, v) \in A^* \times B^*$ defined by finite (nondet.) two-tape automata [10]; it contains properly the class **Rec**($\mathbb{U}_A \times \mathbb{U}_B$) = the finite union of Cartesian products of regular languages.
$\mathbf{T}(F)$ (Terms over F; $\mathbb{U}_A$ is a special case of $\mathbf{T}(F)$)	**Equat**($\mathbf{T}(F)$) = **Rec**($\mathbf{T}(F)$) = the regular sets of terms = the sets of terms defined by finite tree-automata. The minimal deterministic automaton corresponds to the syntactic congruence w.r.t. $\mathbf{T}(F)$.
$\mathbb{R} = \langle \mathbb{R}, \|, \mathbf{ext}, \mathbf{1} \rangle$ (Rooted unordered trees)	**Equat**($\mathbb{R}$) $\supset$ **Rec**($\mathbb{R}$) by [12]. The recognizable sets of rooted trees are defined by certain finite automata (Proposition 4.12). There exists a unique minimal deterministic $\mathbb{R}$-automaton recognizing any member of **Rec**($\mathbb{R}$) (Proposition 4.13).
$\mathbb{G}$ (Undirected finite graphs with sources) ($\mathbb{G}$ is many sorted with infinitely many sorts and operations)	**Equat**($\mathbb{G}$) = the sets generated by context-free hyperedge replacement grammars; the set of all finite graphs is not equational. **Rec**($\mathbb{G}$) = the recognizable sets of finite graphs; it is incomparable with **Equat**($\mathbb{G}$); it is uncountable and cannot be described by finite automata.

7.1. Remarks on Section 2

There are two families of operations on graphs and hypergraphs. The operations presented in Example 2.4 yield the *hyperedge replacement sets of graphs and hypergraphs* as the corresponding equational sets. See Habel [33] on hyperedge replacement grammars; see Arnborg et al. [1], Bauderon and Courcelle [5], Courcelle [13, 19] for variants in the definitions that yield the same equational sets. Other operations on graphs and hypergraphs, yielding the *vertex replacement sets of graphs and hypergraphs* are considered in Courcelle et al. [22] and [11, 17, 21].

The identity of the equational and the inductive theory of a set of equational axioms, is called the *ω-completeness*. See Lazrek et al. [36] or Tajine [44] for tools to establish ω-completeness.

7.2. Remarks on Section 3

A thorough study of polynomial systems can be found in Courcelle [10]. This paper reviews transformations of polynomial systems and distinguishes the transformations which preserve the set of solutions from those which preserve only the least solutions.

Proposition 3.7 due to Mezei and Wright [38] is a simple but fundamental result, as pointed out by Engelfriet and Schmidt [28]. It is also the basis of *initial algebra semantics* [31] and of *algebraic semantics* [32] where it applies to recursive definitions of various types and to program schemes. It can be seen as the fundamental result of *strictness* analysis: see in particular Damm [24, Theorem 7.8] which develops the ideas of [28]. The noncircularity test for attribute grammars is based on Propositions 3.5 and 3.7 (See [10, Section 16.8]).

Overlooking the linearity requirement in results (3.9)–(3.11) has been the source of false statements. See Engelfriet and Schmidt [28] and Arnold and Dauchet [3] for discussions and correct versions.

There are two ways of defining derivation trees for context-free grammars. The "concrete one" where nodes of the derivation tree are labelled by terminal and nonterminal symbols is appropriate for establishing pumping lemmas and for syntax analysis. It has not been used here because it does not extend to arbitrary systems of equations. The "algebraic notion" of derivation tree presented here is applicable to arbitrary systems; it is useful for semantic purposes, and in particular for the translation step in compilation because the corresponding trees are smaller than the "concrete ones".

Proposition 3.17 gives the theorem of Ginsburg and Rice [30] saying that the context-free languages generated by the nonterminals of a grammar form the least solution of the corresponding system of equations. This result is similar to many others relating a "fixed-point semantics" with an "operational one": the characterization of equational sets in terms of regular term grammars is the "operational semantics" of our systems of equations. We refer the reader to Guessarian [32] or Courcelle [14] for examples of such results in semantics.

We used but did not reprove Kleene's fixed-point lemma: this "folklore" result is well-known enough. See Lassesz et al. [35].

The set of all finite graphs is not equational. The reason is that all finite graphs cannot be generated by a finite subsignature of $\mathbb{G}$. (See [5, 13, 19]). This shows a major difference with the case of words.

Not much is known about systems of equations using intersection and difference of sets. We can only cite Leiss [37].

7.3. Remarks on Section 4

Automata are particular to certain structures like words and trees. On the contrary, the purely algebraic notion of *recognizability* works in any magma. (This view point is developed in [12]).

Proposition 4.1 is more or less a "folklore" theorem: the case of monoids is stated without proof in Eilenberg [27, Proposition 12.1, Chap. III]). The general statement can be found in Steinby [43, Proposition 3.1]. Monoids that satisfy Kleene's Theorem are studied in [40, 42].

Proposition 4.5 has an important illustration with graphs. It is proved in Courcelle [15] that the set of finite graphs satisfying a property expressed in monadic second-order logic is effectively recognizable. This gives in particular an effective way of defining recognizable sets of graphs, by logical formulas. However the emptiness of such a set is undecidable: one cannot decide whether there exists a finite graph satisfying a given formula. By Corollary 5.2 one can decide whether there exists a graph *in a given equational set* that satisfies a given monadic second-order formula. (As already observed, the set of all finite graphs is not equational.) In Arnborg et al. [1], the notion of recognizability of a set of graphs is used for the construction of graph reduction systems, able to verify certain graph properties in linear time.

The book by Gecseg and Steinby [29] develops in detail the different aspects of tree-automata. See also the volume edited by Nivat and Podelski where [4] and [18] appear. The correspondence between algebraic and concrete automata is also discussed in [12]. Magmas such that the equational sets are recognizable are considered in [12].

The recognizable sets of rooted unranked unordered trees are those definable by formulas of counting monadic second-order logic. See [15] for the proof (and the necessary definitions).

7.4. Remarks on Section 5

Theorem 5.1 generalizes the fact that the intersection of a context-free language and a regular one is context-free. The application of this result to graph grammars (where recognizable sets of graphs are defined by formulas of monadic second-order logic) is an essential tool.

7.5. Remarks on Section 6

The use of Algorithm 6.1 for input graphs given with a derivation tree relative to a hyperedge-replacement grammar is developed by Courcelle and Mosbah [23], and in a different setting by Arnborg et al. [2]. See these papers for references to many previous papers using the same methodology.

Theorem 6.7 applies to certain evaluations defined in terms of monadic second-order formulas and yields another proof of the extension of Parikh's Theorem presented in [21].

Acknowledgements

I thank A. Arnold and I. Walukiewicz for their comments on first drafts of this article.

References

[1] S. Arnborg, B. Courcelle, A. Proskurowski and D. Seese, An algebraic theory of graph reduction, *J. ACM* **40** (1993) 1134–1164.
[2] S. Arnbor, J. Lagergren and D. Seese, Easy problems for tree-decomposable graphs, *J. Algorithms* **12** (1991) 308–340.
[3] A. Arnold and M. Dauchet Un théorème de duplication pour les forêts algébriques, *J. Comput. System Sci.* **13** (1976) 223–244.
[4] A. Arnold and D. Niwinski, Fixed-point characterization of weak monadic logic definable sets of trees, in: M. Nivat and A. Podelski, eds., *Tree Automata and Languages* (Elsevier, Amsterdam, 1992) 159–188.
[5] M. Bauderon and B. Courcelle, Graph rewriting and graph expressions, *Math. Systems Theory* **20** (1987) 83–127.
[6] G. Birkhoff, On the structure of abstract algebras, *Proc. Cambridge Philos. Soc.* **29** (1935) 433–454.
[7] N. Bourbaki, Algèbre, Chapitre 1, Hermann, Paris, 1970.
[8] D. Bruschi, G. Pighizzini and N. Sabadini, On the existence of minimum asynchronous automata and on the equivalence problem for unambigous regular trace languages. *Inform. Comput.* **108** (1994) 262–285.
[9] R. Cori, Y. Métivier and W. Zielonka, Asynchronous mappings and asynchronous cellular automata, *Inform. Comput.* **106** (1993) 159–203
[10] B. Courcelle, Equivalences and transformations of regular systems. Applications to recursive program schemes and grammars, *Theoret. Comput Sci.* **42** (1986) 1–122.
[11] B. Courcelle, An axiomatic definition of context-free rewriting and its application to NLC graph grammars, *Theoret. Comput. Sci.* **55** (1987) 141–181.
[12] B. Courcelle, On recognizable sets and tree-automata, in: M. Nivat and H. Aït-Kaci, eds. *Resolution of Equations in Algebraic Structures*, Vol. 1 (Academic Press, New York, 1989) 93–126
[13] 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.
[14] B. Courcelle, Recursive applicative program schemes, in J.Van Leeuwen, ed., *Handbook of Theoretical Computer Science*, Vol. B (Elsevier, Amsterdam, 1990) 460–492.
[15] B. Courcelle, The monadic second-order logic of graphs I, Recognizable sets of finite graphs, *Inform. Comput.* **85** (1990) 12–75.
[16] B. Courcelle, The monadic second-order logic of graphs V, On closing the gap between definability and recognizability, *Theoret. Comput. Sci.* **80** (1991) 153–202.
[17] B. Courcelle, The monadic second-order logic of graphs VII, Graphs as relational structures, *Theoret. Comput. Sci.* **101** (1992) 3–33.
[18] B. Courcelle, Recognizable sets of unrooted trees, in: M. Nivat and A. Podelski eds., *Tree Automata and Languages* (Elsevier, Amsterdam, 1992) 141–157.
[19] B. Courcelle, Graph grammars, monadic second-order logic, and the theory of graphs minors, in: N. Robertson and P. Seymour, eds., *Graph structure theory*, Contemporary Mathematics **147** (1993) 565–590.
[20] B. Courcelle, Recognizable sets of graphs: equivalent definitions and closure properties, *Math. Struct. Comput. Sci.* **4** (1994) 1–32.
[21] B. Courcelle, Structural properties of sets of hypergraphs generated by vertex replacement, *Inform. Comput.* **116** (1995) 275–293.
[22] B. Courcelle, J. Engelfriet and G. Rozenberg, Handle-rewriting hypergraph grammars, *J. Comput. System Sci.* **46** (1993) 218–270.
[23] B. Courcelle and M. Mosbah, Monadic second-order evaluation of tree-decomposable graphs, *Theoret. Comput. Sci.* **109** (1993) 49–82.

[24] W. Damm, The IO and OI hierarchies, *Theoret. Comput. Sci.* **20** (1982) 95–207.
[25] N. Dershowitz and J.-P. Jouannaud, Rewrite systems, in: J. Van Leeuwen, ed., *Handbook of Theoretical Computer Science*, Vol. B (Elsevier, Amsterdam, 1990) 243–320.
[26] H. Ehrig and B. Mahr, *Fundamentals of Algebraic Specifications* 1: *Equations and Initial Semantics* (Springer, Berlin 1985).
[27] S. Eilenberg, *Automata, Languages and Machines*, Vol. A (Academic Press, New York, 1974).
[28] J. Engelfriet and E. Schmidt, IO and OI, *J. Comput. System Sci.* **15** (1977) 328–353 and **16** (1978) 67–99.
[29] F. Gecseg and M. Steinby, *Tree-automata* (Akademiai Kiado, Budapest, 1984).
[30] S. Ginsburg and H. Rice, Two families of languages related to ALGOL, *J. ACM* **9** (1962) 350–371.
[31] J. Goguen, J. Thatcher, E. Wagner and J. Wright, Initial algebra semantics and continuous algebras, *J. ACM* **24** (1977) 68–95.
[32] I. Guessarian, Algebraic semantics, Lecture Notes in Computer Science, Vol. 99 (Springer, Berlin, 1981).
[33] A. Habel, Hyperedge replacement: grammars and languages, Lecture Notes in Computer Science, Vol. 643 (Springer, Berlin, 1992).
[34] J. Klop, Term rewriting systems, in: S. Abramsky, D. Gabbay, T. Maibaum, eds., *Handbook of Logic in Computer Science*, Vol. 2 (Oxford University Press, Oxford, 1992) 1–116.
[35] J.-L. Lassez, Y. Nguyen and E. Sonenberg, Fixed-point theorems and semantics: a folk tale, *Inform. Proc. Lett.* **14** (1982) 112–116.
[36] A. Lazrek, P. Lescanne and J.-J. Thiel, Tools for proving inductive equalities, relative completeness and ω-completeness, *Inform. Comput.* **84** (1990) 47–70.
[37] E. Leiss, On generalized language equations, *Theoret. Comput. Sci.* **14** (1981) 63–77.
[38] J. Mezei and J. Wright, Algebraic automata and context-free sets, *Inform. Control* **11** (1967) 3–29.
[39] D. Niwinski, Fixed points vs. infinite generation, *Proc. 3rd Symp. on Logic In Computer Science* (1988) 402–409.
[40] M. Pelletier and J. Sakarovich, Easy multiplications II: extensions of rational monoids, *Inform. Comput.* **74** (1987) 173–197.
[41] D. Pilling, Commutative regular equations and Parikh's theorem, *J. London Math. Soc.* **6** (1973) 663–666.
[42] J. Sakarovich, Easy multiplications I: the realm of Kleene's Theorem, *Inform. Comput.* **74** (1987) 173–197.
[43] M. Steinby, Some algebraic aspects of recognizability and rationality, Lecture Notes in Computer Science, Vol. 117 (Springer, Berlin, (1981) 360–372.
[44] M. Tajine, Décidabilité de l'ω-complétude des spécifications algébriques par des axiomes égalitaires clos, Université L. Pasteur, Strasbourg, France, Report 92/19, 1992.
[45] W. Wechler, *Universal Algebra for Computer Scientists* (Springer, Berlin,1992).
[46] M. Wirsing, Algebraic specifications, in: J. Van Leeuwen, ed., *Handbook of Theoretical Computer Science*, Vol. B (Elsevier, Amsterdam, 1990) 675–779.
[47] W. Zielonka, Notes on finite asynchronous automata, *RAIRO Inform. Théortique Appl.* **21** (1987) 99–135.