Logical Methods in Computer Science
Volume 22, Issue 1, 2026, pp. 22:1–22:30
https://lmcs.episciences.org/

Submitted Jun. 07, 2024
Published Mar. 10, 2026

CHARACTERIZATIONS OF MONADIC SECOND ORDER DEFINABLE CONTEXT-FREE SETS OF GRAPHS

RADU IOSIF [a] AND FLORIAN ZULEGER [b]

[a] Univ. Grenoble Alpes, CNRS, Grenoble INP, VERIMAG, 38000, France

[b] Institute of Logic and Computation, Technische Universität Wien, Austria

Abstract. We give a characterization of the sets of graphs that are both *definable* in Counting Monadic Second Order Logic (CMSO) and *context-free*, i.e., least solutions of Hyperedge-Replacement (HR) grammars introduced by Courcelle and Engelfriet [CE12]. We prove the equivalence of these sets with: (a) *recognizable* sets (in the algebra of graphs with HR-operations) of bounded tree-width; we refine this condition further and show equivalence with recognizability in a finitely generated subalgebra of the HR-algebra of graphs; (b) *parsable* sets, for which there is a definable transduction from graphs to a set of derivation trees labelled by HR operations, such that the set of graphs is the image of the set of derivation trees under the canonical evaluation of the HR operations; (c) images of recognizable unranked sets of trees under a definable transduction, whose inverse is also definable. We rely on a novel connection between two seminal results, a logical characterization of context-free graph languages in terms of tree-to-graph definable transductions, by Courcelle and Engelfriet [CE95] and a proof that an optimal-width tree decomposition of a graph can be built by an definable transduction, by Bojańczyk and Pilipczuk [BP16, BP22].

1. Introduction

Formal language theory studies finite representations of infinite sets of objects (e.g., words, trees, graphs). These representations can be *descriptive*, specifying logical properties of their members (e.g. planar or Hamiltonian graphs), or *constructive*, describing how the members of the set are built. In particular, constructive representations come with algebras that define sets of operations. *Context-free* sets arise from the least solutions of recursive equation systems, which use operations from the considered algebra, with unknowns ranging over sets. *Recognizable* sets are defined in terms of congruence relations over the algebra, with a finite number of equivalence classes; these equivalence classes can be used to define equivalent notions of recognizability in terms of automata or homomorphisms to finite algebras (such as monoids for words).

Monadic Second Order Logic (MSO) is the most prominent descriptive representation of graphs, and has seen decades of study, see for example [CE12]. Hyperedge Replacement (HR) algebras provide standard constructive representations, using (sorted) substitutions of a hyperedge in a graph by graph with a tuples of designated vertices, that matches the sort of the hyperedge [CE12]. The notion of context-free HR graph grammar then follows

Key words and phrases: Graph Grammars, Monadic Second Order Logic, Recognizability, Tree-width.

LOGICAL METHODS IN COMPUTER SCIENCE DOI:10.46298/LMCS-22(1:22)2026

immediately from the definition of a HR algebra of graphs. In contrast, the right notion of recognizability is somewhat less obvious. This is because words and trees have a clear beginning (root) and traversal direction (left-right, top-down or reverse), while graphs do not have either. Instead of having a congruence relation with finite index, the proposal put forward in [CE12] consists of a *locally finite* congruence relation, i.e., a congruence relation with a finite number of equivalence classes for every sort.

The comparison of the expressive powers of different representations is central to formal language theory. For words, definability in MSO coincides with recognizability [Büc90], being subsumed by context-freeness, whereas for ground terms over a finite set of function symbols definability in MSO, recognizability and context-freeness coincide[1] [Don70, EW67]. For unranked and unordered trees (i.e., trees with arbitrarily many children per node, whose order is, moreover, not important), definability in CMSO and recognizability coincide, where CMSO is the extension of MSO with modulo constraints on the cardinality of sets [Cou90]. For graphs, definability in CMSO implies recognizability but not vice versa, whereas context-freeness is incomparable to the two other notions [Cou90]. The equivalence between recognizability and definability in CMSO can be recovered for graphs of bounded *tree-width* [BP16]. Moreover, recognizability of bounded tree-width sets of graphs (by locally finite congruences) is equivalent to recognizability by congruences having finitely many classes [CL96].

Finite representations are used in system design and automated verification tools. Descriptive representations (logics) specify correctness properties, e.g., sets of safe states or behaviors (traces of states), whereas constructive representations describe the implementations of a system, with respect to the low-level details of state changes. Verification problems, such as conformance with certain safety criteria, or equivalence of two implementations, amount to checking inclusion between sets of words, trees or graphs, represented in different ways. Hence, the interest for classes of representations having a *decidable inclusion problem*.

We study the intersection between the classes of graphs that are both context-free and definable (in CMSO). The main motivation is that inclusion is decidable for the members of this intersection. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be sets defined by CMSO formulæ ϕ_1 and ϕ_2, respectively. Then, $\mathcal{L}_1 \subseteq \mathcal{L}_2$ if and only if the formula $\phi_1 \wedge \neg\phi_2$ is not satisfiable. If, moreover, $\mathcal{L}_1$ is context-free, there is an effectively computable bound on the tree-width of the models of $\phi_1 \wedge \neg\phi_2$, if any. Since the satisfiability problem for CMSO is decidable for graphs of bounded tree-width, by a seminal result of Courcelle [Cou90, Corollary 4.8 (2)], the problem $\mathcal{L}_1 \subseteq \mathcal{L}_2$ is decidable.

Our characterization of context-free and definable graph languages starts from the notion of *strongly context-free* sets, introduced by Courcelle [Cou91]. These are sets $\mathcal{L}$ generated by an HR grammar, having an additional *parsability* property: there exists a binary relation F between graphs and derivation trees, such that (i) each output tree is defined by a finite tuple of CMSO formulæ interpreted over the input graph, and (ii) for each graph $G \in \mathcal{L}$, the set $F(G)$ contains a derivation tree that evaluates to G. In this context, Courcelle stated the following conjectures:

[1]The context-free tree grammars mentioned in [Eng15] use first-order parameters in rules, being thus strictly more expressive than tree automata [CDG+08]. Here, by a grammar, we understand a finite set of recursive equations whose left-hand sides consist of a single nonterminal of arity zero. The components of the least solution of such a system are also known as *equational sets* [CE12].

Conjecture 1.1 [Cou91, Conjecture 3]**.** If a set of graphs is context-free and definable, then it is strongly context-free.

This conjecture leads to the following insight: given a graph grammar, assume that we want to prove that its language is definable. By Conjecture 1.1, now proved as a consequence of Theorem 6.10, definability is equivalent to the existence of a definable parsing transduction for the language generated by the grammar. Hence, for constructing the desired CMSO formula it is always a viable proof strategy to either explicitly or implicitly build such a parsing function as part of the overall construction. The contrapositive of this conjecture is proved as [Cou91, Theorem 4.8]. Moreover, the equivalence with the following conjecture is also proved in [Cou91]:

Conjecture 1.2 [Cou91, Conjecture 2]**.** For each $k \in \mathbb{N}$, the set of all graphs of tree-width at most k is strongly context-free.

Our Contributions. The main contribution of the paper is a detailed and self-contained proof of the above conjectures, yielding two characterizations of the intersection between the context-free and the definable classes of graphs.

The fine-grained version (Theorem 6.9), takes into account the finite set of sorts occurring in the grammar and proves the equivalence between: (1) $\mathcal{L}$ is definable in CMSO and generated by a HR grammar that uses a finite set of sorts τ, (2) $\mathcal{L}$ is recognizable in the infinitely-sorted HR algebra and is represented by a set of τ-sorted terms, (3) $\mathcal{L}$ is recognizable in the τ-sorted HR subalgebra and is represented by a set of τ-sorted terms, which (4) can be extracted from the graphs in $\mathcal{L}$ by a relation definable in CMSO.

The equivalence of (2) and (3) provides also a simpler proof of the equivalence between locally-finite and finite recognizability for graphs of bounded tree-width, initially proved by Courcelle and Lagergren [CL96]. Since we prove later that recognizability in the infinitely-sorted HR algebra is equivalent to recognizability in an infinite sequence of finitely-sorted HR subalgebras (Theorem 7.4), the equivalence of points (2) and (3) in Theorem 6.9 gives a cut-off result: a set of tree-width bounded graphs is recognizable if and only if it is recognizable in a finitely-sorted algebra.

The coarse-grained version (Theorem 6.10) quantifies existentially over the set of sorts and states the equivalence between: (1) $\mathcal{L}$ is definable in CMSO and HR context-free, (2) $\mathcal{L}$ is recognizable and of bounded tree-width, (3) $\mathcal{L}$ is represented by a set of terms that can be extracted from $\mathcal{L}$ by a relation definable in CMSO, and (4) the existence of two definable relations from graphs to trees and back, whose composition is the identity on $\mathcal{L}$. It is known that context-free sets have bounded tree-width, but not the other way around. A consequence of our result is that every definable set that has bounded tree-width is context-free.

Our results rely on two seminal ingredients. The first is a characterization of the context-free sets of graphs, as *images of recognizable ranked sets of trees under definable relations*, by Courcelle and Engelfriet [CE95]. The second is a construction of *tree decompositions of optimal width*, by means of definable relations, by Bojańczyk and Pilipczuk [BP16, BP22]. We connect the two results using (i) a generalization of [CE95] by considering *unranked* instead of ranked recognizable sets of trees (Corollary 5.10), and (ii) a definable translation of a tree decomposition into a parse tree of a HR grammar (Lemma 6.5).

Related Work. Following its initial development [Cou91, CE95], the study of definable context-free sets of graphs has seen recent interest. Our work is closely related to [Boj23], where Bojańczyk proposes the notions of *logical recognizability* and *definable tree decompositions*. The latter notion is used to formalize the condition (4) of Theorem 6.10. For his logical recognizability notion, it becomes immediate from the definitions that the recognizable subsets of a class that has definable tree decompositions are definable in CMSO. Then, [Boj23] concentrates on establishing the equivalence between congruence-based and logical recognizability on words, trees and graphs of bounded tree-width. The equivalences for words and ranked trees are based on the classical results of Büchi [Büc90] and Doner [Don70], respectively, whereas the equivalence for bounded tree-width graphs uses the same ingredients as our work [CE95, BP16, BP22], considered in more generality and on a higher level of abstraction, as the goal of [Boj23] is to avoid the introduction of sorts and graph operations as needed for the definition of context-free sets of graphs in terms of graph grammars. In summary, Bojańczyk establishes the equivalences (1) $\Leftrightarrow$ (2) $\Leftrightarrow$ (4) of our coarse-grained Theorem 6.10, but the fine-grained characterization of Theorem 6.9 cannot be immediately derived from the development in [Boj23].

We note that the problem of whether a given context-free grammar defines a recognizable (and hence definable) language is undecidable (even for words), according to a result by Greibach [Gre68]. This has motivated the search for regular grammars and regular expressions over graphs, whose languages are guaranteed to be recognizable resp. definable. In particular, already Courcelle in [Cou91, Section 5] proposed so-called *regular graph grammars*, built over hyperedge-replacement operations that need to satisfy some local connectivity requirements. In recent work, we have proposed *tree-verifiable graph grammars* [CIZ24], which strictly generalize the regular graph grammars of Courcelle. We note that these grammars do not capture all recognizable sets of graphs but only those recognizable sets of graphs of bounded *embeddable* tree-width, i.e., those sets of graphs for which there is a tree decomposition whose backbone is a spanning tree of the considered graph. There has also been recent progress on defining the recognizable sets of tree-width at most 2 (a class of graphs that is orthogonal to the class of graphs of bounded embeddable treewidth). These sets can be defined equivalently by regular expressions [Dou22] and regular graph grammars [BIZ25].

We finally mention a formal comparison between the expressivity of MSO with that of Separation Logic over graphs of bounded tree-width [IZ23].

2. Preliminaries

This section introduces the basic notions of the descriptive and constructive representations of infinite sets. We introduce (Counting) Monadic Second Order Logic as the main descriptive language for sets of structures (subsection 2.1) and relations between structures (subsection 2.2). The constructive representations we consider are the standard notions of recognizable (subsection 2.3) and context-free sets (subsection 2.4) in multi-sorted algebras.

We denote by $\mathbb{N}$ the set of natural numbers and $\mathbb{N}_+ \stackrel{\text{def}}{=} \mathbb{N} \setminus \{0\}$. Given $i, j \in \mathbb{N}$, we write $[i, j] \stackrel{\text{def}}{=} \{i, i+1, \ldots, j\}$, assumed to be empty if $i > j$. The cardinality of a finite set A is denoted by $\mathrm{card}(A)$. By writing $A \subseteq_{\mathit{fin}} B$ we mean that A is a finite subset of B. For a set A, we denote by $\mathrm{pow}(A)$ its powerset, $A^0 \stackrel{\text{def}}{=} \{\epsilon\}$, $A^{i+1} \stackrel{\text{def}}{=} A^i \times A$, for all $i \geq 0$, $A^* \stackrel{\text{def}}{=} \bigcup_{i\geq 0} A^i$ and $A^+ \stackrel{\text{def}}{=} \bigcup_{i\geq 1} A^i$, where $\times$ is the Cartesian product and ε denotes the empty sequence. Intuitively, A^* (resp. A^+) denotes the set of possibly empty (resp. nonempty) sequences of elements from A. The length of a sequence $\mathbf{a} \in A^*$ is denoted as $\mathrm{len}(\mathbf{a})$ and $\mathbf{a}_i$ denotes its i-th element, for $i \in [1, \mathrm{len}(\mathbf{a})]$.

For a relation $R \subseteq A \times B$, we denote by $\mathrm{dom}(R)$ and $\mathrm{img}(R)$ the sets consisting of the first and second components of the pairs in R, respectively. We write R^{-1} for the inverse relation and $R(S)$ for the image of a set S via R. Sometimes we write $R(a)$ instead of $R(\{a\})$, for an element $a \in A$. The *domain-restriction* $R\downharpoonright_C$ restricts the relation R to the pairs with first element in C. A bijective function f is an *A-permutation* if $\{a \in \mathrm{dom}(f) \mid f(a) \neq a\} \subseteq A \subseteq \mathrm{dom}(f)$. It is a *finite permutation* if it is an A-permutation, for some finite set A.

2.1. **Counting Monadic Second Order Logic (CMSO).** A *relational signature* $\mathbb{R}$ is a finite set of *relation symbols*, ranged over by r, of arities $\#\mathsf{r} \geq 0$. A relation symbol r is a *constant*, *unary* or *binary* if $\#\mathsf{r} = 0$, 1 or 2, respectively.

A $\mathbb{R}$-*structure* is a pair $\mathsf{S} = (\mathsf{U}, \sigma)$, where U is a *universe* and $\sigma : \mathbb{R} \to \mathrm{pow}(\mathsf{U}^*)$ is an *interpretation*, that maps each relation symbol r into a subset of $\mathsf{U}^{\#\mathsf{r}}$ of corresponding arity. Here we consider only structures with finite universe, also called *finite structures*. The set of $\mathbb{R}$-structures is denoted by $\mathcal{S}(\mathbb{R})$.

The *Counting Monadic Second Order Logic* (CMSO) is the set of formulæ written using a set $\mathbb{X}^{(1)} = \{x, y, \ldots\}$ of *first-order variables*, a set $\mathbb{X}^{(2)} = \{X, Y, \ldots\}$ of *second-order variables* and the relation symbols from *relations*, according to the following syntax:

$$\psi := x = y \mid \mathsf{r}(x_1, \ldots, x_{\#\mathsf{r}}) \mid X(x) \mid \mathsf{card}_{q,p}(X) \mid \neg\psi \mid \psi \wedge \psi \mid \exists x \;.\; \psi \mid \exists X \;.\; \psi$$

where $p, q \in \mathbb{N}$ are constants, such that $p \in [0, q-1]$. By MSO we denote the subset of CMSO consisting of formulæ that do not contain atomic propositions of the form $\mathsf{card}_{q,p}(X)$, also called *cardinality constraints*. A variable is *free* in a formula ϕ if it does not occur in the scope of a quantifier. A *sentence* is a formula with no free variables.

The semantics of CMSO is given by a satisfaction relation $(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \psi$, where the store $\mathfrak{s} : \mathbb{X}^{(1)} \cup \mathbb{X}^{(2)} \to \mathsf{U} \cup \mathrm{pow}(\mathsf{U})$ maps each variable $x \in \mathbb{X}^{(1)}$ to an element of the universe and each variable $X \in \mathbb{X}^{(2)}$ to a subset of U. This relation is defined inductively on the syntactic structure of formulæ:

$$\begin{array}{lcl}
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} x = y & \iff & \mathfrak{s}(x) = \mathfrak{s}(y) \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \mathsf{r}(x_1, \ldots, x_k) & \iff & \langle \mathfrak{s}(x_1), \ldots, \mathfrak{s}(x_k) \rangle \in \sigma(\mathsf{r}) \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} X(x) & \iff & \mathfrak{s}(x) \in \mathfrak{s}(X) \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \mathsf{card}_{q,p}(X) & \iff & \mathrm{card}(\mathfrak{s}(X)) = kq + p, \text{ for some } k \in \mathbb{N} \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \phi \wedge \psi & \iff & (\mathsf{U}, \sigma) \models^{\mathfrak{s}} \phi \text{ and } (\mathsf{U}, \sigma) \models^{\mathfrak{s}} \psi \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \neg\phi & \iff & (\mathsf{U}, \sigma) \not\models^{\mathfrak{s}} \phi \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \exists x \;.\; \psi & \iff & (\mathsf{U}, \sigma) \models^{\mathfrak{s}[x \leftarrow u]} , \text{ for some element } u \in \mathsf{U} \\
(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \exists X \;.\; \psi & \iff & (\mathsf{U}, \sigma) \models^{\mathfrak{s}[X \leftarrow V]} \psi, \text{ for some set } V \subseteq \mathsf{U}
\end{array}$$

If ϕ is a sentence, the satisfaction relation does not depend on the store and we write $(\mathsf{U}, \sigma) \models \phi$ instead of $(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \phi$. A set S of structures is *definable* iff $S = \{(\mathsf{U}, \sigma) \mid (\mathsf{U}, \sigma) \models \phi\}$, for some CMSO sentence ϕ, and MSO-*definable* in case ϕ belongs to the MSO fragment of CMSO. Two structures are *isomorphic* iff they differ only by a renaming of their elements (a formal definition is given in [EF95, Section A3]). It is known that the satisfaction relation of

CMSO does not distinguish between isomorphic structures. We note that we only consider finite structures in this paper, and hence quantification is over finite sets only[2].

2.2. **Definable Transductions.** Let $\mathbb{R}$ and $\mathbb{R}'$ be relational signatures. A relation δ between $\mathbb{R}$- and $\mathbb{R}'$-structures is a *k-copying* $(\mathbb{R}, \mathbb{R}')$-*transduction* iff each output structure $\mathsf{S}' \in \delta(\mathsf{S})$ is produced from k disjoint copies of the input structure S, called *layers*. The transduction is said to be *copyless* if $k = 1$. The outcome of the transduction also depends on the valuation of zero or more *set parameters* $X_1, \ldots, X_n \in \mathbb{X}^{(2)}$, that range over the subsets of the input universe. The transduction is said to be *parameterless* if $n = 0$. Formally, we define $(\mathbb{R}, \mathbb{R}')$-transductions using *transduction schemes*, i.e., finite tuples of CMSO formulæ:

$$\Theta = \langle \varphi, \{\psi_i\}_{i \in [1,k]}, \{\theta_{(\mathsf{q}, i_1, \ldots, i_{\#\mathsf{q}})}\}_{\mathsf{q} \in \mathbb{R}',\ i_1, \ldots, i_{\#\mathsf{q}} \in [1,k]} \rangle$$

where:

- ▷ $\varphi(X_1, \ldots, X_n)$ selects the input structures (U, σ) for which the transduction has an output (U', σ'), i.e., those structures (U, σ) such that $(\mathsf{U}, \sigma) \models^{\mathfrak{s}} \varphi$, for a store $\mathfrak{s}$ that maps each X_i into a set $\mathfrak{s}(X_i) \subseteq \mathsf{U}$,
- ▷ $\psi_i(x_1, X_1, \ldots, X_n)$ defines the elements from the i-th layer copied in the output universe:

$$\mathsf{U}' \stackrel{\text{def}}{=} \{(u, i) \in \mathsf{U} \times [1, k] \mid (\mathsf{U}, \sigma) \models^{\mathfrak{s}[x_1 \leftarrow u]} \psi_i\}$$

- ▷ $\theta_{(\mathsf{q}, i_1, \ldots, i_{\#\mathsf{q}})}(x_1, \ldots, x_{\#\mathsf{q}}, X_1, \ldots, X_n)$ define the interpretation of $\mathsf{q} \in \mathbb{R}'$ in the output:

$$\sigma'(\mathsf{q}) \stackrel{\text{def}}{=} \{\langle (u_1, i_1), \ldots, (u_{\#\mathsf{q}}, i_{\#\mathsf{q}}) \rangle \mid (\mathsf{U}, \sigma) \models^{\mathfrak{s}[x_1 \leftarrow u_1, \ldots, x_{\#\mathsf{q}} \leftarrow u_{\#\mathsf{q}}]} \theta_{(\mathsf{q}, i_1, \ldots, i_{\#\mathsf{q}})}\}$$

Note that the store $\mathfrak{s}$ that defines the valuations of $X_1, \ldots, X_n$ is the same everywhere in the above definition of the output structure (U', σ'). The output of the transduction is denoted by $\mathrm{def}^{\mathfrak{s}}_{\Theta} \stackrel{\text{def}}{=} (\mathsf{U}', \sigma')$. The set $\mathrm{def}_{\Theta}(\mathsf{S})$ is the closure under isomorphism of the set $\{\mathrm{def}^{\mathfrak{s}}_{\Theta} \mid \mathsf{S} \models^{\mathfrak{s}} \varphi\}$, i.e., the output structures are the structures isomorphic to some $\mathrm{def}^{\mathfrak{s}}_{\Theta}$, whose elements are not necessarily pairs of the form $(u, i) \in \mathsf{U} \times [1, k]$. A transduction δ is *definable* iff $\delta = \mathrm{def}_{\Theta}$, for some transduction scheme Θ.

Example 2.1. Let $\mathbb{A} = \{\mathsf{left}, \mathsf{right}, \mathsf{next}\}$ and $\mathbb{B} = \{\mathsf{left}, \mathsf{right}\}$ be alphabets of edge labels, which we will use in order to encode edge-labeled binary trees (for $\mathbb{B}$) and trees with linked leaves (for $\mathbb{A}$). As usual, the left (resp. right) child of a node in the tree will be linked to its parent via a left-labeled (resp. right-labeled) edge. The next-labeled edges are used to link a leaf to its direct successor, in the lexicographic order.

We use here the *incidence encoding* of trees (more generally, graphs) as relational structures, where edges are elements of the universe and the graph is described by the incidence relation between edges and vertices (see subsection 3.3). In particular, we use relation symbols r_a for $a \in \{\mathsf{left}, \mathsf{right}, \mathsf{next}\}$ of arity three, where the first argument of r_a denotes an edge and the second and third element denote the source and destination of this edge, respectively.

Then, there is an MSO sentence φ_{tree} that defines the set of binary trees, where each non-leaf node has exactly one left- and one right-child. Moreover, we consider the following MSO formulæ:

---

[2]For infinite structures there is a difference in expressivity when only quantification over finite set is allowed and the resulting logic is known as *weak* MSO. However, quantifiers in finite structures can only capture finite sets and hence we not need distinguish between MSO and *weak* MSO.

▷ $x \prec_{lex} y$ means that x is the direct predecessor of y in the lexicographic order of the labels of the paths from the root to x and y, induced by the $\mathsf{left} < \mathsf{right}$ order,
▷ $leaf(x)$ states that x has no outgoing left- or right-labeled edges, i.e., is a leaf of the tree.

The 2-copying parameterless $(\{\mathsf{r}_b\}_{b\in\mathbb{B}}, \{\mathsf{r}_a\}_{a\in\mathbb{A}})$-transduction defined by the scheme

$$\Theta_{\mathsf{tll}} \stackrel{\text{def}}{=} \langle \varphi_{\mathsf{tree}}, \{\psi_1, \psi_2\}, \{\theta_{(\mathsf{r}_a,i,j,k)}\}_{a\in\mathbb{A},i,j,k\in[1,2]} \rangle$$

where

$$\psi_1(x_1) \stackrel{\text{def}}{=} \mathtt{true}$$

$$\psi_2(x_1) \stackrel{\text{def}}{=} leaf(x_1) \wedge \exists y \,.\, leaf(y) \wedge x_1 \prec_{lex} y$$

$$\theta_{(\mathsf{r}_a,i,j,k)}(x_1,x_2,x_3) \stackrel{\text{def}}{=} \begin{cases} \mathsf{r}_a(x_1,x_2,x_3) & \text{if } (i,j,k) = (1,1,1) \\ \mathtt{false} & \text{otherwise} \end{cases} \quad \text{for all } a \in \{\mathsf{left}, \mathsf{right}\}$$

$$\theta_{(\mathsf{r}_{\mathsf{next}},i,j,k)}(x_1,x_2,x_3) \stackrel{\text{def}}{=} \begin{cases} leaf(x_2) \wedge leaf(x_3) \wedge x_1 = x_2 \wedge x_2 \prec_{lex} x_3 & \text{if } (i,j,k) = (2,1,1) \\ \mathtt{false} & \text{otherwise} \end{cases}$$

adds a next edge between each pair of successive leaves in this lexicographic order. Note that the extra next-labeled edges are taken from the 2^{nd} layer and corresponds to the copy of its source node, i.e., the ψ_2 selects all but the right-most leaf from the tree for the 2^{nd} layer.

Dually, the copyless parameterless $(\{\mathsf{r}_a\}_{a\in\mathbb{A}}, \{\mathsf{r}_b\}_{b\in\mathbb{B}})$-transduction defined by the scheme $\Theta_{\mathsf{tll}}^{-1} \stackrel{\text{def}}{=} \langle \mathtt{true}, \{\psi_1\}, \{\theta_{(\mathsf{r}_b,1,1)}\}_{b\in\mathbb{B}} \rangle$, where

$$\psi_1(x_1) \stackrel{\text{def}}{=} \forall y \forall z \,.\, \neg \mathsf{r}_{\mathsf{next}}(x_1,y,z)$$

$$\theta_{(\mathsf{r}_b,1,1)}(x_1,x_2,x_3) \stackrel{\text{def}}{=} \mathsf{r}_b(x_1,x_2,x_3), \qquad \text{for all } b \in \{\mathsf{left}, \mathsf{right}\}$$

removes all next edges from the input structure.

The main property of definable transductions is the Backwards Translation Theorem (see e.g., [CE12, Theorem 1.40]):

Theorem 2.2 [CE12]**.** *If $\mathcal{S} \subseteq \mathcal{S}(\mathbb{R}')$ is a definable set and δ is a definable $(\mathbb{R}, \mathbb{R}')$-transduction then the set $\delta^{-1}(\mathcal{S})$ is definable.*

The following properties are direct consequences of the above theorem and the definition of definable transductions:

Proposition 2.3.

(1) *The composition of definable transductions is definable.*
(2) *The domain-restriction of a definable transduction by a definable set is definable.*
(3) *The domain of a definable transduction is definable.*

2.3. **Recognizable Sets.** Let Σ be a set of *sorts*, ranged over by σ, and let $\mathcal{F} = \{f_1, f_2, \ldots\}$ be a *functional signature.* which is a set of *function symbols* f. Each function symbol f has an associated tuple of argument sorts and a value sort, denoted $\mathrm{args}(f) = \langle \sigma_1, \ldots, \sigma_n \rangle$ and $\mathrm{sort}(f)$, respectively. The arity of f is denoted $\#f \stackrel{\text{def}}{=} n$. A variable is a sorted symbol of arity zero, not part of the signature. The sort of a variable x (resp. X) is denoted $\mathrm{sort}(x)$ (resp. $\mathrm{sort}(X)$). Terms are build from variables and function symbols of matching sorts, as usual. We write $t(x_1, \ldots, x_n, X_1, \ldots, X_m)$ if $x_1, \ldots, x_n, X_1, \ldots, X_m$ are the variables from t. A *ground term* is a term without variables. A *first-order term* does not contain second-order variables.

An $\mathcal{F}$-*algebra* $\mathbf{A} = (\{\mathcal{A}^\sigma\}_{\sigma\in\Sigma}, \{f^{\mathbf{A}}\}_{f\in\mathcal{F}})$ consists of a *universe* $\mathcal{A}^\sigma$ for each sort $\sigma \in \Sigma$ and interprets each function symbol $f \in \mathcal{F}$ by a function $f^{\mathbf{A}} : \mathcal{A}^{\sigma_1} \times \ldots \times \mathcal{A}^{\sigma_n} \to \mathcal{A}^\sigma$, where $\mathrm{args}(f) = \langle\sigma_1, \ldots, \sigma_n\rangle$ and $\mathrm{sort}(f) = \sigma$. The set $\mathcal{A} \stackrel{\text{def}}{=} \bigcup_{\sigma\in\Sigma} \mathcal{A}^\sigma$ denotes the union of all universes of $\mathbf{A}$. The sort of an element $a \in \mathcal{A}$ is denoted $\mathrm{sort}(a)$. The algebra $\mathbf{A}$ is *locally finite* iff $\mathcal{A}_\sigma$ is finite, for each $\sigma \in \Sigma$ and finite iff $\mathcal{A}$ is finite.

An $\mathcal{F}$-term $t(x_1, \ldots, x_n, X_1, \ldots, X_m)$ is viewed as a function symbol of arity $n+m$, whose interpretation is obtained by interpreting the function symbols from t in $\mathbf{A}$ and lifting the functions from elements to sets of elements of the same sort. A set $\mathcal{T}$ of first-order $\mathcal{F}$-terms defines a *derived* $\mathcal{T}$*-algebra* of $\mathbf{A}$ with the same set of sorts Σ and the same universes $\mathcal{A}^\sigma$ for each sort $\sigma \in \Sigma$, that interprets each function symbol t as the function $t^{\mathbf{A}}$. A *subalgebra* of $\mathbf{A}$ is any algebra obtained by restricting the set of sorts, signature and universes of $\mathbf{A}$.

An $\mathcal{F}$-algebra $\mathbf{A}$ is *term-generated* iff its universe is the set of interpretations of the ground $\mathcal{F}$-terms in $\mathbf{A}$ (we call these elements term-generated). The term-generated subalgebra $\mathbf{A}_{\mathsf{gen}}$ is the subalgebra defined as the restriction of $\mathbf{A}$ to its term-generated elements.

We denote by $\mathbf{M}(\mathcal{F})$ the *initial algebra* over the functional signature $\mathcal{F}$. The universes $\mathcal{M}(\mathcal{F})_\sigma$ of $\mathbf{M}(\mathcal{F})$ are the sets of ground $\mathcal{F}$-terms having the same sort σ and the interpretation of each function symbol $f \in \mathcal{F}$ is $f^{\mathbf{M}(\mathcal{F})}(t_1, \ldots, t_{\#f}) \stackrel{\text{def}}{=} f(t_1, \ldots, t_{\#f})$, for all ground terms $t_1, \ldots, t_{\#f}$ of matching sorts.

We recall below the standard notion of recognizability:

Definition 2.4. An equivalence relation $\cong$ on $\mathcal{A}$ is a *congruence* iff $a \cong b$ only if (1) $\mathrm{sort}(a) = \mathrm{sort}(b)$ and (2) for all $f \in \mathcal{F}$, if $a_i \cong b_i$ then $f^{\mathbf{A}}(a_1, \ldots, a_{\#f}) \cong f^{\mathbf{A}}(b_1, \ldots, b_{\#f})$. A congruence is *locally finite* iff it has finitely many equivalence classes of each sort. A congruence $\cong$ *saturates* a set $\mathcal{L} \subseteq \mathcal{A}$ iff $\mathcal{L}$ is a union of equivalence classes of $\cong$. A set is *recognizable* iff there exists a locally finite congruence that saturates it.

Any (not necessarily recognizable) set is saturated by a unique coarsest congruence:

Definition 2.5. The *syntactic congruence* of a set $\mathcal{L} \subseteq \mathcal{A}$ in an $\mathcal{F}$-algebra $\mathbf{A}$ is the relation $a \cong^{\mathbf{A}}_{L} b$ defined as $\mathrm{sort}(a) = \mathrm{sort}(b)$ and $t^{\mathbf{A}}(a, c_1, \ldots, c_k) \in \mathcal{L} \Leftrightarrow t^{\mathbf{A}}(b, c_1, \ldots, c_k) \in \mathcal{L}$, for all first-order $\mathcal{F}$-terms $t(x, y_1, \ldots, y_k)$ and all $c_1, \ldots, c_k \in \mathcal{A}$.

The proof that $\cong^{\mathcal{L}}_{\mathbf{A}}$ is the coarsest congruence that saturates $\mathcal{L}$ is standard, see e.g., [CE12, Proposition 3.66]. Hence, $\mathcal{L}$ is recognizable if and only if $\cong^{\mathcal{L}}_{\mathbf{A}}$ is locally finite.

For the purpose of several proofs in the paper, we introduce an equivalent definition of recognizability using homomorphisms into locally finite algebras. A *homomorphism* between $\mathcal{F}$-algebras $\mathbf{A}$ and $\mathbf{B}$ is a function $h : \mathcal{A} \to \mathcal{B}$ such that (1) $f(\mathcal{A}^\sigma) \subseteq \mathcal{B}^\sigma$, for all sorts $\sigma \in \Sigma$, and (2) $h(f^{\mathbf{A}}(a_1, \ldots, a_{\#f})) = f^{\mathbf{B}}(h(a_1), \ldots, h(a_{\#f}))$, for all function symbols $f \in \mathcal{F}$ and all elements $a_1, \ldots, a_{\#f} \in \mathcal{A}$.

Definition 2.6. A set $\mathcal{L} \subseteq \mathcal{A}$ is *recognizable* in $\mathbf{A}$ iff there exists a locally finite algebra $\mathbf{B}$ and a homomorphism h between $\mathbf{A}$ and $\mathbf{B}$ such that $\mathcal{L} = h^{-1}(\mathcal{C})$, for a set $\mathcal{C} \subseteq \mathcal{B}$.

The equivalence with the notion of recognizability introduced above is that the equivalence relation $\cong$ from Definition 2.4 is the kernel of the homomorphism h from Definition 2.6, see, e.g., [CE12, Proposition 3.64] for a proof of equivalence between the two notions of recognizability.

Recognizability in an algebra implies recognizability in each derived (sub)algebra, as shown by the following two lemmas:

Lemma 2.7. *Let* $\mathbf{D}$ *be a derived algebra of* $\mathbf{A}$*. Then,* $\mathcal{L} \subseteq \mathcal{A}$ *is recognizable in* $\mathbf{D}$ *if it is recognizable in* $\mathbf{A}$*.*

Proof. Let $\mathbf{D} = (\{\mathcal{A}^\sigma\}_{\sigma\in\Sigma}, \{t^{\mathbf{A}}\}_{t\in\mathcal{D}})$. For any locally finite $\mathcal{F}$-algebra $\mathbf{B} = (\{\mathcal{B}^\sigma\}_{\sigma\in\Sigma}, \mathcal{F}^{\mathbf{B}})$, any homomorphism h between $\mathbf{A}$ and $\mathbf{B}$ is also a homomorphism between $\mathbf{D}$ and the derived algebra $\mathbf{D'} = (\{\mathcal{B}^\sigma\}_{\sigma\in\Sigma}, \{t^{\mathbf{B}}\}_{t\in\mathcal{D}})$. The homomorphism h and the set $\mathcal{C} \subseteq \mathcal{B}$ that witness the recognizability of $\mathcal{L}$ in $\mathbf{A}$ also witness the recognizability of $\mathcal{L}$ in $\mathbf{D}$. □

Note that the converse does not hold, for instance, if we consider the algebra of words over the alphabet $\{a, b\}$ with signature consisting of the empty word ϵ and concatenation. A derived algebra is obtained by taking the empty word and the derived operation $x \mapsto axb$. Then $\{a^n b^n \mid n \in \mathbb{N}\}$ is recognizable in the derived algebra but not in the original one.

Lemma 2.8. *Let* $\mathbf{B}$ *be a subalgebra of* $\mathbf{A}$*. Then,* $\mathcal{L} \subseteq \mathcal{B}$ *is recognizable in* $\mathbf{B}$ *if it is recognizable in* $\mathbf{A}$*.*

Proof. Let $\mathbf{B} = (\{\mathcal{B}^\sigma\}_{\sigma\in\Sigma'}, \{f^{\mathbf{B}}\}_{f\in\Sigma'})$ and $\mathbf{D} = (\{\mathcal{D}^\sigma\}_{\sigma\in\Sigma}, \mathcal{F}^{\mathbf{D}})$ be a locally finite $\mathcal{F}$-algebra and $h : \mathcal{A} \to \mathcal{D}$ be a homomorphism, such that $\mathcal{L} = h^{-1}(\mathcal{C})$, for some $\mathcal{C} \subseteq \mathcal{D}$. Let $\mathbf{E}$ be the subalgebra of $\mathbf{D}$ obtained by restricting $\mathbf{D}$ to the sorts Σ' and the signature $\mathcal{F}'$. Then $h' \stackrel{\text{def}}{=} h\downharpoonright_{\mathcal{B}}$ is a homomorphism between the algebras $\mathbf{B}$ and $\mathbf{E}$ and $\mathcal{L} = h^{-1}(\mathcal{C} \cap \mathcal{E}) \cap \mathcal{B} = h'^{-1}(\mathcal{C} \cap \mathcal{E})$, which witnesses the recognizability of $\mathcal{L}$ in $\mathbf{B}$. □

2.4. **Context-Free Sets.** Let $\mathcal{F}$ be a functional signature and $\mathbb{U} \subseteq \mathbb{X}^{(2)}$ be a set of second-order variables, called *nonterminals*, ranged over by U, V, etc. A *grammar* is a finite set of *rules* of the form $U \to t$, where U is a nonterminal and t is a $\mathcal{F}$-term with variables from $\mathbb{U}$.

A *solution* of Γ over an $\mathcal{F}$-algebra $\mathbf{A}$ is a mapping $\mathcal{S} : \mathbb{U} \to \mathrm{pow}(\mathcal{A})$ such that $t^{\mathcal{S}} \subseteq \mathcal{S}(U)$ for each rule $U \to t \in \Gamma$, where $t^{\mathcal{S}}$ denotes the evaluation of the term with regard to the sets $\mathcal{S}(U) \subseteq \mathcal{A}$, for each variable $U \in \mathbb{U}$. Since the evaluation of terms with set variables is monotonic with regard to set containment, a least solution exists and is unique. We denote by $\mathcal{L}_U(\Gamma)$ the component corresponding to U within the least solution of Γ.

Definition 2.9. A set $\mathcal{L}$ is *context-free* iff $\mathcal{L} = \mathcal{L}_U(\Gamma)$, for a nonterminal $U \in \mathbb{U}$ and a grammar Γ.

The following theorem generalizes a classical result, namely that the intersection of a context-free and a regular set of words is context-free. We restate the result in its algebraic form, also known as the Filtering Theorem:

Theorem 2.10 [CE12, Theorem 3.88]**.** *Let* $\mathcal{L}$ *be a context-free set and* $\mathcal{K}$ *be a recognizable set. Then,* $\mathcal{L} \cap \mathcal{K}$ *is context-free. Moreover, the grammar for* $\mathcal{L} \cap \mathcal{K}$ *has the same sorts as the one for* $\mathcal{L}$*.*

Here, by the sorts of a grammar, we understand the set of sorts that occur as argument or value sorts in each function symbol occurring in that grammar.

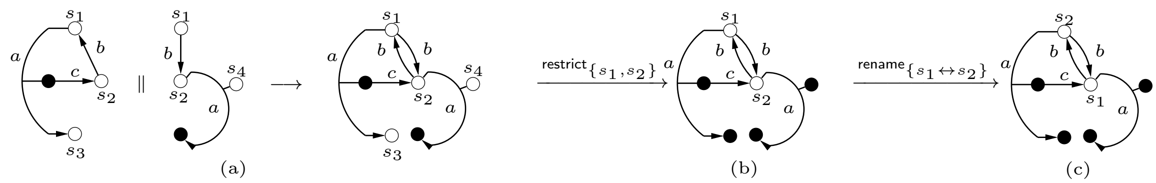

Figure 1: Composition (a), Restriction (b) and Renaming (c) of Graphs. Sources are denoted by hollow and internal vertices by solid circles. Arrows indicate the order of vertices attached to the corresponding edges.

3. Graphs

This section introduces hyper-graphs with edges labeled by symbols from a finite alphabet and distinguished source vertices, the hyperedge replacement algebra (subsection 3.1), the subalgebras that use finitely many source labels (subsection 3.2) and the notion of definable sets of graphs, via the encoding of the incidence relation of a graph by a relational structure (subsection 3.3).

Let $\mathbb{S}$ be a countably infinite set of *source labels* and $\mathbb{A}$ be an alphabet of *edge labels*, disjoint from $\mathbb{S}$. Each edge label $a \in \mathbb{A}$ has an associated *arity* $\#a \geq 1$, i.e., we do not consider edge labels of arity zero. The sets $\mathbb{S}$ and $\mathbb{A}$ are fixed in the rest of the paper.

Definition 3.1. Let $\tau \subseteq_{fin} \mathbb{S}$ be a finite set of source labels. A *concrete graph* of sort τ is a tuple $G = \langle V_G, E_G, \lambda_G, \upsilon_G, \xi_G \rangle$, where:

- ▷ V_G is a finite set of *vertices*,
- ▷ E_G is a finite set of *edges*, disjoint from V_G,
- ▷ $\lambda_G : E_G \to \mathbb{A}$ is a mapping that defines the labels of the edges,
- ▷ $\upsilon_G : E_G \to V_G^+$ is a mapping that associates each edge a nonempty sequence of vertices attached to the edge, such that $\#(\lambda_G(e)) = \mathrm{len}(\upsilon_G(e))$, for each $e \in E_G$,
- ▷ $\xi_G : \tau \to V_G$ is a *one-to-one* mapping that designates the *sources* of G. The vertex $\xi_G(s)$ is called the s-source of G. Because ξ_G is injective, a vertex cannot be both an s- and s'-source, for $s \neq s'$. Vertices that are not sources are called *internal*.

We identify concrete graphs up to isomorphism and define *graphs* as isomorphism-equivalence classes of concrete graphs. We denote by $\mathcal{G}$ the set of graphs.

Example 3.2. The leftmost graph in Figure 1 (a) has four vertices of which three sources labeled s_1, s_2 and s_3 and three edges labeled a, b and c. The a-labeled edge is attached to three vertices, whereas the b- and c-labeled edges are binary. The arrows on the edges indicate the order of the vertices attached to them. The middle graph is of sort $\{s_1, s_2, s_4\}$ and the rightmost one of sort $\{s_1, s_2, s_3, s_4\}$.

3.1. The Hyperedge Replacement Algebra.

We introduce the *hyperedge replacement* (HR) algebra of operations on graphs. There are several equivalent definitions of this algebra in the literature. For instance, the definition from [Cou91] uses operations described by graphs with sources denoted by $\{1, \ldots, k\}$, having designated edges e_i, for $i \in [1, n]$, that are deleted and replaced by graphs G_i of sorts $\{1, \ldots, \#\lambda_{G_i}\}$, respectively. These operations are strongly typed. Instead, we consider an algebra over a signature of polymorphic operations, having the same expressivity [CE12, Definition 2.32]. To ensure compliance with the general definition of multi-sorted algebras from subsection 2.3, these polymorphic operations can be understood as families of sorted operations, one for each choice of the argument sorts.

We fix the set of sorts Σ_{HR} to be the set of finite subsets of $\mathbb{S}$. The signature $\mathcal{F}_{\mathsf{HR}}$ consists of the constants $\mathbf{0}_\tau$, for all $\tau \subseteq_{fin} \mathbb{S}$ and $\mathbf{a}_{(s_1,\ldots,s_{\#a})}$, for all $a \in \mathbb{A}$ and $s_1,\ldots,s_{\#a} \in \mathbb{S}$, the unary function symbols $\mathsf{restrict}_\tau$, for all $\tau \subseteq_{fin} \mathbb{S}$, and rename_α, for all finite permutations $\alpha : \mathbb{S} \to \mathbb{S}$ and the binary function symbol $\|$. The *graph algebra* $\mathbf{G}$ interprets the symbols in $\mathcal{F}_{\mathsf{HR}}$ as follows:

(1) **sources only**: the graph $\mathbf{0}_\tau^{\mathbf{G}}$ consists of one s-source for each $s \in \tau$ and no edges.
(2) **single edge**: the graph $\mathbf{a}^{\mathbf{G}}_{(s_1,\ldots,s_{\#a})}$ consists of an s_i-source, for each $i \in [1,\#a]$, and a single edge labeled with a attached to the $s_1,\ldots,s_{\#a}$-sources, in this order.
(3) **restriction**: the unary function $\mathsf{restrict}^{\mathbf{G}}_\tau$ takes as input a graph of sort τ' and returns the graph of sort $\tau \cap \tau'$ obtained by removing the source labels in $\tau' \setminus \tau$ from G. Note that the vertices from $\mathrm{img}(\tau)$ are not removed in the output graph. Formally, each concrete graph G is mapped into the concrete graph $\langle V_G, E_G, \lambda_G, \upsilon_G, \xi_G\downarrow_\tau\rangle$ and $\mathsf{restrict}^{\mathbf{G}}_\tau$ is defined as the lifting of this operation from concrete graphs to graphs.
(4) **rename:** the unary function $\mathsf{rename}^{\mathbf{G}}_\alpha$ takes as input a graph of sort τ and returns the graph of sort $\alpha^{-1}(\tau)$ obtained by renaming its sources according to α. Formally, each concrete graph G is mapped into the concrete graph $\langle V_G, E_G, \lambda_G, \upsilon_G, \xi_G \circ \alpha\rangle$ and $\mathsf{rename}^{\mathbf{G}}_\alpha$ is defined as the lifting of this operation from concrete graphs to graphs.
(5) **composition:** the binary function $\|^{\mathbf{G}}$ takes the disjoint union of two graphs of sorts τ_1 and τ_2 and fuses the vertices labeled by the same source label in both. The result is a graph of sort $\tau_1 \cup \tau_2$. Formally, let G_i be concrete graphs of sort τ_i, for $i = 1, 2$, such that $V_{G_1} \cap V_{G_2} = \emptyset$ and $E_{G_1} \cap E_{G_2} = \emptyset$. Let $\sim \subseteq (V_{G_1} \cup V_{G_2})^2$ be the least equivalence relation such that $u_1 \sim u_2$ if $u_i = \xi_{G_i}(s)$, for $i = 1, 2$ and $s \in \tau_1 \cap \tau_2$. Then the composition G_{12} of G_1 with G_2 is defined as follows:
   - ▷ $V_{G_{12}} = \{[u]_\sim \mid u \in V_{G_1} \cup V_{G_2}\}$, $E_{G_{12}} = E_{G_1} \cup E_{G_2}$ and $\lambda_{G_{12}} \stackrel{\mathsf{def}}{=} \lambda_{G_1} \cup \lambda_{G_2}$,
   - ▷ $\upsilon_{G_{12}}(e) \stackrel{\mathsf{def}}{=} \langle [u_1]_\sim, \ldots, [u_k]_\sim\rangle$ for every edge $e \in E_{G_i}$, such that $\upsilon_{G_i}(e) = \langle u_1, \ldots, u_k\rangle$,
   - ▷ $\xi_{G_{12}}(s) \stackrel{\mathsf{def}}{=} [\xi_{G_i}(s)]_\sim$ iff $s \in \tau_i$, for $i = 1, 2$.

   where $[u]_\sim$ is the $\sim$-equivalence class of the vertex u. Then, $\|^{\mathbf{G}}$ is the lifting of this binary operation from concrete graphs to graphs.

Example 3.3. For example, Figure 1 (a) shows the result of the composition of two graphs, whereas (b) and (c) show the result of applying restriction and renaming to this composition, respectively. By $\{i \leftrightarrow j\}$ we denote the finite permutation on $\mathbb{N}$ that swaps i with j and maps every $k \in \mathbb{N} \setminus \{i, j\}$ to itself.

3.2. **Hyperedge Replacement Subalgebras.** For each $\tau \subseteq_{fin} \mathbb{S}$, let $\mathcal{G}^\tau \stackrel{\mathsf{def}}{=} \{G \in \mathcal{G} \mid \mathrm{sort}(G) \subseteq \tau\}$ be the set of graphs of sort included in τ and $\mathbf{G}^\tau$ be the subalgebra of $\mathbf{G}$ with the universe $\mathcal{G}^\tau$ and the finite signature of operations that use only source labels from τ:

$$\mathcal{F}^\tau_{\mathsf{HR}} \stackrel{\mathsf{def}}{=} \{\mathbf{0}_{\tau'}\}_{\tau' \subseteq \tau} \cup \{\mathbf{a}_{(s_1,\ldots,s_{\#a})}\}_{\substack{a \in \mathbb{A} \\ s_1,\ldots,s_{\#a} \in \tau}} \cup \{\mathsf{restrict}_{\tau'}\}_{\tau' \subseteq \tau} \cup \{\mathsf{rename}_\alpha\}_{\alpha\ \tau\text{-permutation}} \cup \{\|\}$$

We denote by $\mathcal{G}^\tau_{\mathsf{gen}}$ the set of term-generated elements of $\mathbf{G}^\tau$ and by $\mathbf{G}^\tau_{\mathsf{gen}}$ the term-generated subalgebra of $\mathbf{G}^\tau$. Note that, while all elements of the graph algebra $\mathbf{G}$ are term-generated, each term-generated algebra $\mathbf{G}^\tau_{\mathsf{gen}}$ is a strict subalgebra of $\mathbf{G}^\tau$.

3.3. **Definable Sets of Graphs.** In order to describe sets of graphs using CMSO, we encode graphs as relational structures over finite relational signatures. To this end, we consider the alphabet $\mathbb{A}$ of edge labels to be finite. Given a sort $\tau \subseteq_{fin} \mathbb{S}$, we define the relational signature $\mathbb{R}^{\tau}_{\mathsf{graph}} \stackrel{\mathsf{def}}{=} \{\mathsf{r}_a \mid a \in \mathbb{A}\} \cup \{\mathsf{r}_s \mid s \in \tau\}$ whose relation symbols have the arities $\#\mathsf{r}_a \stackrel{\mathsf{def}}{=} \#a + 1$, for all $a \in \mathbb{A}$, and $\#\mathsf{r}_s \stackrel{\mathsf{def}}{=} 1$, for all $s \in \tau$. Note that the signature $\mathbb{R}^{\tau}_{\mathsf{graph}}$ is finite because both $\mathbb{A}$ and τ are finite. The *encoding* of a concrete graph $G \in \mathcal{G}^{\tau}$ is the structure $\|G\| = (V_G \cup E_G, \sigma_G) \in \mathcal{S}(\mathbb{R}^{\tau}_{\mathsf{graph}})$, where:

$$\sigma_G(\mathsf{r}_a) \stackrel{\mathsf{def}}{=} \{(e, v_1, \ldots, v_{\#a}) \mid e \in E_G,\ \lambda_G(e) = a,\ \upsilon_G(e) = (v_1, \ldots, v_{\#a})\}, \text{ for all } a \in \mathbb{A}$$
$$\sigma_G(\mathsf{r}_s) \stackrel{\mathsf{def}}{=} \{\xi_G(s)\}, \text{ for all } s \in \tau$$

We note that the encodings of isomorphic concrete graphs are isomorphic structures. The encoding of a graph denotes the isomorphism-equivalence class obtained from the encodings of the concrete graphs in the graph.

This type of encoding, where edges are elements of the universe and the graph is described by the incidence relation between edges and vertices, is known as the *incidence encoding*. Another encoding used in the literature is the *edge encoding*, where the universe consists of vertices only and edges are tuples from the interpretation of the relations corresponding to the labels. The expressiveness of CMSO differs in the two encodings, e.g., the existence of a Hamiltonian cycle can be described in CMSO using the incidence but not the edge encoding [CE12, Proposition 5.13]. Moreover, key results used in the paper [CE95, Theorems 1.10 and 2.1] hold under the incidence but not the edge encoding.

Definition 3.4. A set of graphs $\mathcal{L} \subseteq \mathcal{G}^{\tau}$ is *definable* if there exists a CMSO formula ϕ over the relational signature $\mathbb{R}^{\tau}_{\mathsf{graph}}$ such that $\|\mathcal{L}\| = \{\mathsf{S} \mid \mathsf{S} \models \phi\}$.

Note that a set of graphs that is not included in $\mathcal{G}^{\tau}$, for any finite $\tau \subseteq_{fin} \mathbb{S}$, is not definable, because a CMSO formula can only speak of finitely many relation symbols. We recall the following result [Cou90, Theorem 4.4]:

Theorem 3.5 [Cou90]**.** *Any definable set of graphs is recognizable in* $\mathbf{G}$.

4. Trees

We deviate from the definition of trees used in the classical literature on tree automata (see, e.g., [CDG+08] for a survey) and define trees as the term-generated elements of a suitable derived subalgebra of graphs. In contrast to the standard definition, where the number and order of the children of a node in the tree is determined by the label of that node, our definition encompasses also trees whose nodes have an unbounded number of children, such that, moreover, the order of siblings is not important. As explained below, this definition of trees is strictly more general than the classical definition of terms over ranked alphabets [CDG+08]. This generality is needed, because trees are used in the definition of tree decompositions, that impose no restrictions on the number or the order of siblings.

Let $\mathbb{B} \subseteq \mathbb{A}$ be a set of edge labels of arities at most two and $\mathfrak{r}$ be a source label. We use c and b to denote the symbols of arities one and two of $\mathbb{B}$, respectively. The $\mathfrak{r}$-source of a graph is called its *root*. The signature of trees with $\mathbb{B}$-labeled edges is the following set of

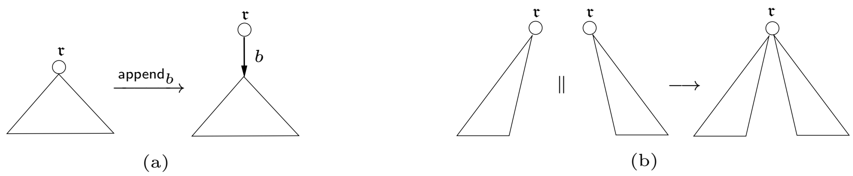

Figure 2: Append (a) and Composition (b) of Trees. The $\mathfrak{r}$-sources are denoted by hollow circles.

function symbols:

$$\mathcal{F}_{\mathsf{tree}}(\mathbb{B}) \stackrel{\mathsf{def}}{=} \{\mathbf{c} \mid c \in \mathbb{B}, \#c = 1\} \cup \{\mathsf{append}_b \mid b \in \mathbb{B}, \#b = 2\} \cup \{\|\}$$

where $\mathbf{c}$ are constants (we omit specifying the $\mathfrak{r}$ source label) and append_b are unary. The constants $\mathbf{c}$ are interpreted as trees consisting of a single root vertex, whereas append_b and $\|$ are interpreted as in Figure 2. Formally, $\mathsf{append}_b(x) \stackrel{\mathsf{def}}{=} \mathsf{rename}_{\mathfrak{r}\leftrightarrow aux}(\mathsf{restrict}_{\{aux\}}(\mathbf{b}_{(aux,\mathfrak{r})} \| x))$, where aux is an auxiliary source label used only here and $\mathfrak{r} \leftrightarrow aux$ is the permutation that switches $\mathfrak{r}$ with aux. We denote by $\mathbf{T}(\mathbb{B})$ the term-generated algebra whose universe $\mathcal{T}(\mathbb{B})$ (of a single sort $\{\mathfrak{r}\}$) is the set of interpretations of the ground $\mathcal{F}_{\mathsf{tree}}(\mathbb{B})$-terms in $\mathbf{G}$.

The standard terminology for trees is immediately retrieved from the above definition. The vertices of a tree T are called *nodes*. For a binary edge $e \in E_T$, we say that $\upsilon_T(e)_1$ is the *parent* of $\upsilon_T(e)_2$ and $\upsilon_T(e)_2$ is a *child* of $\upsilon_T(e)_1$. A node with no children is called a *leaf*. Since trees are interpretations of ground terms, each leaf corresponds to the interpretation of at least some constant $\mathbf{c}$, thus is attached to a unary edge labeled with c. The *rank* of a tree is the maximum number of children of a node. A set of trees is *ranked* if the corresponding set of ranks is finite and *unranked*, otherwise. In particular, the set $\mathcal{T}(\mathbb{B})$ is unranked.

4.1. **Definable Sets of Trees.** Because trees are graphs, the encoding of trees is no different from that of graphs. We consider a finite set $\mathbb{B}$ of edge labels and define the relational signature $\mathbb{R}_{\mathsf{tree}}(\mathbb{B}) \stackrel{\mathsf{def}}{=} \{\mathsf{r}_b \mid b \in \mathbb{B}\} \cup \{\mathsf{r}_{\mathfrak{r}}\}$, where $\mathfrak{r}$ is the singleton source label associated with the root. Interestingly, the expressiveness of CMSO using incidence and the edge encodings coincide, when trees are considered, instead of graphs (a consequence of the Sparseness Theorem [CE12, Theorem 1.44]).

A classical result of Courcelle [Cou90] is the equivalence between definability and recognizability for (possibly unranked) sets of trees:

Theorem 4.1 [Cou90]**.** *For any finite alphabet $\mathbb{B}$, a set of trees $\mathcal{K} \subseteq \mathcal{T}(\mathbb{B})$ is recognizable in $\mathbf{T}(\mathbb{B})$ iff $\mathcal{K}$ is definable.*

As a consequence of Theorem 4.1, we obtain the equivalence of the recognizability of a set of trees in the graph and tree algebras:

Corollary 4.2. *For each finite alphabet $\mathbb{B} \subseteq \mathbb{A}$, a set of trees is recognizable in $\mathbf{T}(\mathbb{B})$ iff it is recognizable in $\mathbf{G}$.*

Proof. "⇒" By Theorem 4.1, if a set of trees is recognizable in $\mathbf{T}(\mathbb{B})$ then it is definable. By Theorem 3.5, any definable set of graphs is recognizable in $\mathbf{G}$. "⇐" By Lemmas 2.7 and 2.8, because $\mathbf{T}(\mathbb{B})$ is a derived subalgebra of $\mathbf{G}$. □

As a remark, each definable ranked set of trees can be described using the MSO fragment of CMSO, by [Cou90, Proposition 6.11]. The distinction between unranked and ranked sets of trees is formally established by the fact that there exists sets of trees definable in CMSO that are not MSO-definable [Cou90, Corollary 6.6]. For instance, the set of trees of height one having an even number of leaves is CMSO-definable but not MSO-definable. In other words, CMSO is strictly more expressive that MSO.

4.2. **Tree Decompositions.** A *tree decomposition* is another way of encoding a graph as a tree whose edges are all labeled by a binary label parent. A set of nodes $C \subseteq V_T$ is *connected* in T iff between any two nodes in C there exists an undirected path of edges from E_T that traverses only nodes from C. Tree decompositions are used to formalize the notion of *tree-width*:

Definition 4.3. A *tree decomposition* of a concrete graph G of sort τ is a pair (T, β), where $T \in \mathcal{T}(\mathsf{parent})$ and $\beta : V_T \rightarrow \mathrm{pow}(V_G)$ is a mapping, such that:

(1) for each edge $e \in E_G$ there exists a node $n \in V_T$, such that $\upsilon_G(e)_i \in \beta(n)$, for all $1 \leq i \leq \#\lambda_G(e)$,
(2) for each vertex $v \in V_G$, the set $B_T(v) \stackrel{\mathsf{def}}{=} \{n \in V_T \mid v \in \beta(n)\}$ is nonempty and connected in T.
(3) the bag of the root $\mathfrak{r}$ of T contains all sources of G, i.e., we have $\xi_G(s) \in \beta(\xi_T(\mathfrak{r}))$ for all sources $s \in \tau$.

The *width* of the tree decomposition is $\mathrm{wd}(T, \beta) \stackrel{\mathsf{def}}{=} \max\{\mathrm{card}(\beta(n)) \mid n \in V_T\} - 1$ and the *tree-width* of G is $\mathrm{twd}(G) \stackrel{\mathsf{def}}{=} \min\{\mathrm{wd}(T, \beta) \mid (T, \beta) \text{ is a tree decomposition of } G\}$. The tree-width of a graph is the tree-width of any concrete graph from the isomorphism equivalence class (isomorphic concrete graphs have the same tree-width). We denote by $\mathcal{G}^{\leq k}$ the set of graphs G such that $\mathrm{twd}(G) \leq k$.

At this point, we must stress the importance of considering unranked sets of trees, when reasoning about the tree decompositions of a set of graphs. In particular, the tree decompositions extracted by the definable transduction whose existence is stated in [BP16, Theorem 2.4] have no bound on the number of children of a node. Since our results crucially depend on the existence of such a transduction, we must consider the unranked sets of trees, obtained from these tree decompositions, in their full generality.

We assume basic acquaintance with the notion of grid and the fact that an $n \times n$ square grid has tree-width n [Bod98]. To see the difference between the sets $\mathcal{G}^\tau$ and $\mathcal{G}^\tau_{\mathsf{gen}}$, note that a $n \times n$ square grid with no sources belongs to $\mathcal{G}^\tau$ but not to $\mathcal{G}^\tau_{\mathsf{gen}}$, for any $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ such that $\mathrm{card}(\tau) < n + 1$. The following result shows the fundamental difference between graphs and *term-generated* graphs with sources from a given finite set. We restate it here using our notation, for self-containment:

Theorem 4.4 [CE12, Theorem 2.83]**.** *Let $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ be a sort. For each graph $G \in \mathcal{G}^\tau_{\mathsf{gen}}$, we have* $\mathrm{twd}(G) \leq \mathrm{card}(\tau) - 1$.

The encoding of tree decompositions as relational structures uses the relational signature $\mathbb{R}^\tau_{\mathsf{decomp}} \stackrel{\mathsf{def}}{=} \mathbb{R}^\tau_{\mathsf{graph}} \cup \{\mathsf{node}, \mathsf{parent}, \mathsf{bag}\}$, where node is a unary relation symbol and parent, bag are binary relation symbols, respectively. We encode triples (G, T, β), where $G \in \mathcal{G}^\tau$ is a concrete graph and (T, β) is a tree decomposition of G, such that T is a tree having one binary edge label parent[3]. We encode G by a structure $\|G\| = (\mathsf{U}_G, \sigma_G)$ over the relational signature $\mathbb{R}^\tau_{\mathsf{graph}}$. The tree decomposition is encoded by an extended structure $\|G, T, \beta\| \stackrel{\text{def}}{=} (\mathsf{U}_G \cup V_T, \sigma'_G)$, where σ'_G agrees with σ_G over $\mathbb{R}^\tau_{\mathsf{graph}}$ and the unary and binary relation symbols node, parent and bag are interpreted as follows:

$$\begin{aligned}
\sigma'(\mathsf{node}) &\stackrel{\text{def}}{=} V_T \\
\sigma'(\mathsf{bag}) &\stackrel{\text{def}}{=} \{(v, n) \in V_G \times V_T \mid v \in \beta(n)\} \\
\sigma'(\mathsf{parent}) &\stackrel{\text{def}}{=} \{(n, m) \in V_T \times V_T \mid \exists e \in E_T \,.\, \upsilon_T(e) = (n, m)\}
\end{aligned}$$

Note that using the edge encoding to represent trees is without loss of generality, because each edge of a tree with binary edges is represented by its unique target node.

5. Context-Free Sets of Graphs

As one expects, a *context-free set of graphs* is a component of the least solution of a grammar written using the operations from the signature $\mathcal{F}_{\mathsf{HR}}$. A prominent member of this class is the set of graphs that are values of $\mathcal{F}_{\mathsf{HR}}$-terms having finitely many sorts:

Proposition 5.1. *For each sort $\tau \subseteq_{fin} \mathbb{S}$, the set of graphs $\mathcal{G}^\tau_{\mathsf{gen}}$ is context-free.*

Proof. We fix a sort $\tau \subseteq_{fin} \mathbb{S}$. Let Γ be the grammar having a single non-terminal X and the rules:

$$\begin{aligned}
&X \to \mathbf{0}_{\tau'}, \text{ for each } \tau' \subseteq \tau \\
&X \to \mathbf{a}_{(s_1,\ldots,s_{\#a})}, \text{ for each } a \in \mathbb{A} \text{ and } s_1, \ldots, s_{\#a} \in \tau \\
&X \to \mathsf{restrict}_{\tau'}(X), \text{ for each } \tau' \subseteq \tau \\
&X \to \mathsf{rename}_\alpha(X), \text{ for each } \tau\text{-permutation } \alpha \\
&X \to X \parallel X
\end{aligned}$$

As all rules of Γ only use $\mathcal{F}^\tau_{\mathsf{HR}}$-operations, we clearly have $\mathcal{L}_X(\Gamma) \subseteq \mathcal{G}^\tau_{\mathsf{gen}}$. On the other hand, every ground term of $\mathcal{G}^\tau_{\mathsf{gen}}$ can be constructed by the rules of this grammar. Hence, $\mathcal{L}_X(\Gamma) \supseteq \mathcal{G}^\tau_{\mathsf{gen}}$. □

In the rest of this section, we recall results relating a graph produced by a grammar to the trees that describe the partial order in which the grammar rules are applied in order to produce that particular graph. These notions mirror standard concepts used in word grammars, such as *parse trees* and *yields* (i.e., words obtained by reading the symbols of the leaves of a parse tree in the lexicographic order).

5.1. Parse Trees.

We define parse trees by fixing an alphabet $\mathbb{B}_{\mathsf{parse}}$ of unary and binary edge labels and the functional signature $\mathcal{F}_{\mathsf{parse}}$ to be the following sets:

$$\begin{aligned}
\mathbb{B}_{\mathsf{parse}} &\stackrel{\text{def}}{=} \{\underline{\mathbf{0}}_\tau\}_{\tau \subseteq_{fin} \mathbb{S}} \cup \{\underline{\mathbf{a}}_{(s_1,\ldots,s_{\#a})}\}_{\substack{a \in \mathbb{A} \\ s_1,\ldots,s_{\#a} \in \mathbb{S}}} \cup \{\underline{\mathsf{restrict}}_\tau\}_{\tau \subseteq_{fin} \mathbb{S}} \cup \{\underline{\mathsf{rename}}_\alpha\}_{\alpha \text{ finite permutation}} \\
\mathcal{F}_{\mathsf{parse}} &\stackrel{\text{def}}{=} \mathcal{F}_{\mathsf{tree}}(\mathbb{B}_{\mathsf{parse}})
\end{aligned}$$

[3] In addition to unary labels of leafs, that are not important here.

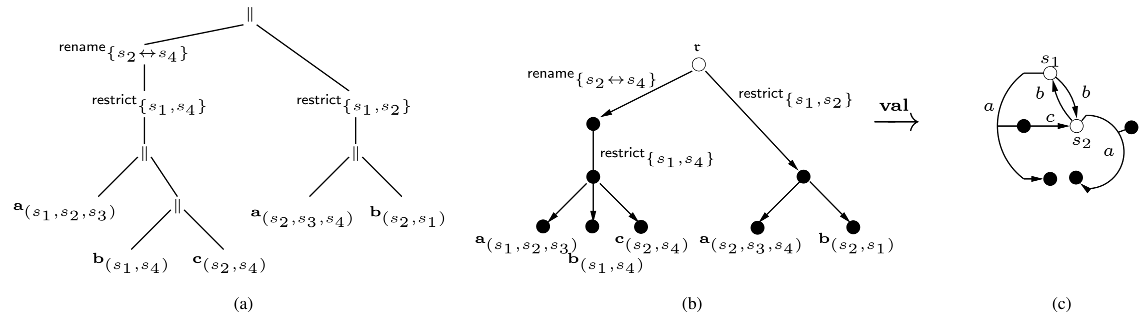

Figure 3: A $\mathcal{F}_{\mathsf{HR}}$-term (a), its corresponding parse tree (b) and the canonical evaluation (c). Sources are denoted by hollow and internal vertices by solid circles.

The edge labels $\underline{\mathbf{0}}_\tau$ and $\underline{\mathbf{a}}_{(s_1,\ldots,s_{\#a})}$ are unary, whereas $\underline{\mathsf{restrict}}_\tau$ and $\underline{\mathsf{rename}}_\alpha$ are binary. We further assume that $\mathbb{B}_{\mathsf{parse}}$ is disjoint from the alphabet $\mathbb{A}$ of edge labels for graphs[4].

Note that $\mathbb{B}_{\mathsf{parse}}$ and $\mathcal{F}_{\mathsf{parse}}$ are infinite, because $\mathbb{S}$ is infinite. In cases where finite restrictions of these sets are necessary, we denote by $\mathbb{B}^\tau_{\mathsf{parse}}$ (resp. $\mathcal{F}^\tau_{\mathsf{parse}}$) the finite restriction of $\mathbb{B}_{\mathsf{parse}}$ (resp. $\mathcal{F}_{\mathsf{parse}}$) to symbols that use only source labels from the finite set $\tau \subseteq_{\mathit{fin}} \mathbb{S}$.

The *algebra of parse trees* $\mathbf{P}$ is a single-sorted $\mathcal{F}_{\mathsf{HR}}$-algebra whose universe is the set of trees $\mathcal{T}(\mathbb{B}_{\mathsf{parse}})$, i.e., with edge labels from $\mathbb{B}_{\mathsf{parse}}$, and the function symbols from $\mathcal{F}_{\mathsf{HR}}$ are interpreted over trees instead of graphs, as follows:

$$\mathsf{restrict}^{\mathbf{P}}_\tau(T) \stackrel{\text{def}}{=} \mathsf{append}^{\mathbf{T}(\mathbb{B}_{\mathsf{parse}})}_{\underline{\mathsf{restrict}}_\tau}(T)$$

$$\mathsf{rename}^{\mathbf{P}}_\alpha(T) \stackrel{\text{def}}{=} \mathsf{append}^{\mathbf{T}(\mathbb{B}_{\mathsf{parse}})}_{\underline{\mathsf{rename}}_\alpha}(T), \text{ for all } T \in \mathcal{T}(\mathbb{B}_{\mathsf{parse}})$$

The rest of the function symbols from $\mathcal{F}_{\mathsf{HR}}$ are interpreted in $\mathbf{P}$ the same as in $\mathbf{G}$.

Example 5.2. Figure 3 shows a $\mathcal{F}_{\mathsf{HR}}$-term (a) and its parse tree (b). Intuitively, a parse tree is obtained from a $\mathcal{F}_{\mathsf{HR}}$-term by (i) moving the node labels to the incident edges and (ii) collapsing all of the adjacent $\|$-nodes in the term to one node.

Since the tree algebra $\mathbf{T}(\mathbb{B}_{\mathsf{parse}})$ is term-generated, each tree is the value of a ground $\mathcal{F}_{\mathsf{HR}}$-term obtained from a ground $\mathcal{F}_{\mathsf{parse}}$-term, by replacing each constant $\underline{\mathbf{c}}$ with $\mathbf{c}$ and each unary function symbol $\mathsf{append}_{\underline{b}}$ with b. Then, recognizability is equivalent in the tree and the parse tree algebras:

Lemma 5.3. *A set $\mathcal{K} \subseteq \mathcal{T}(\mathbb{B}_{\mathsf{parse}})$ is recognizable in $\mathbf{T}(\mathbb{B}_{\mathsf{parse}})$ iff it is recognizable in $\mathbf{P}$.*

Proof. "$\Rightarrow$" Assume that $\mathcal{K} \subseteq \mathcal{T}(\mathbb{B}_{\mathsf{parse}})$ is recognizable in $\mathbf{T}(\mathbb{B}_{\mathsf{parse}})$ and let $\mathbf{B} = (\{\mathcal{B}\}, \mathcal{F}^{\mathbf{B}}_{\mathsf{parse}})$ be a locally finite $\mathcal{F}_{\mathsf{parse}}$-algebra and $h : \mathcal{T}(\mathbb{B}_{\mathsf{parse}}) \to \mathcal{B}$ be a homomorphism between $\mathbf{T}(\mathbb{B}_{\mathsf{parse}})$ and $\mathbf{B}$, such that $\mathcal{K} = h^{-1}(\mathcal{C})$, for some set $\mathcal{C} \subseteq \mathcal{B}$. We consider the locally finite $\mathcal{F}_{\mathsf{HR}}$-algebra $\mathbf{D} = (\{\mathcal{B}\}, \mathcal{F}^{\mathbf{D}}_{\mathsf{HR}})$, where:

$$\mathbf{c}^{\mathbf{D}} \stackrel{\text{def}}{=} \mathbf{c}^{\mathbf{B}}, \text{ for all constants } \mathbf{c} \in \mathcal{F}_{\mathsf{HR}}$$

$$b^{\mathbf{D}} \stackrel{\text{def}}{=} \mathsf{append}^{\mathbf{B}}_{\underline{b}}, \text{ for all unary function symbols } b \in \mathcal{F}_{\mathsf{HR}}$$

$$\|^{\mathbf{D}} \stackrel{\text{def}}{=} \|^{\mathbf{B}}$$

[4]The initial assumption $\mathbb{B} \subseteq \mathbb{A}$ does not work for $\mathbb{B}_{\mathsf{parse}}$ because then $\mathbb{A}$ would need to contain the symbols $\underline{\mathbf{a}}_{(s_1,\ldots,s_{\#a})}$, for all $a \in \mathbb{A}$, etc. To avoid this confusion, we chose to keep $\mathbb{B}_{\mathsf{parse}}$ and $\mathbb{A}$ disjoint.

Then, h is a homomorphism between $\mathbf{P}$ and $\mathbf{D}$ and $\mathcal{K} = h^{-1}(\mathcal{C})$ witnesses the recognizability of $\mathcal{K}$ in $\mathbf{P}$. "$\Leftarrow$" This direction uses a symmetric reasoning. □

5.2. **Canonical Evaluation of Parse Trees.** Parse trees are representations of graphs. The function $\mathbf{val} : \mathcal{T}(\mathbb{B}_{\mathsf{parse}}) \to \mathcal{G}$ yields the graph represented by a parse tree. **val** is defined inductively on the structure of the parse tree, by interpreting the edge labels as operation in the graph algebra, in the obvious way. In particular, if the root of the tree has two or more children, the graph is the composition of the graphs obtained by the evaluation of the subtrees rooted in the children. Note that, because the composition is associative and commutative, the result of **val** does not depend on the order of the children in the parse tree. Henceforth, we refer to **val** as the *canonical evaluation* function. For example, Figure 3 (c) shows the result of the canonical evaluation of the parse tree (b).

Formally, the function **val** yielding the graph represented by a parse tree $t^{\mathbf{P}}$, for a ground $\mathcal{F}_{\mathsf{HR}}$-term t, is defined inductively on the structure of the underlying term as the unique function that satisfies $\mathbf{val}(t^{\mathbf{P}}) = t^{\mathbf{G}}$. To show that the function **val** exists and is unique, we note that the interpretation $\|^{\mathbf{G}}$ of the composition operation is associative and commutative, hence its value on a given parse tree does not depend on the particular order of the subterms composed via $\|$ in the $\mathcal{F}_{\mathsf{HR}}$-term whose value that tree is. The following proposition formalizes this statement, for trees in general:

Proposition 5.4. *Every tree T of height $n \geq 0$ is the value in $\mathbf{T}(\mathbb{B})$ of a $\mathcal{F}_{\mathsf{tree}}(\mathbb{B})$-term of the form $t = \big(\, \|_{i \in I}\, \mathbf{c}_i\big) \,\|\, \big(\, \|_{j \in J}\, \mathsf{append}_{b_j}(t_j)\big)$, such that $I = \emptyset$ implies $J \neq \emptyset$ and $t_j^{\mathbf{T}(\mathbb{B})}$ are trees of height strictly less than n, for all $j \in J$. Moreover, all $\mathcal{F}_{\mathsf{tree}}(\mathbb{B})$-terms that represent the same tree are equal up to the commutativity and associativity of composition.*

Proof. By induction on the height $n \geq 0$ of T. For the base case $n = 0$, the tree consists of a single root node attached to one or more edges labeled with unary symbols $c_i \in \mathbb{B}$, for $i \in I$. Then there exists a term $\|_{i \in I}\, \mathbf{c}_i$ that represents T and this term is unique modulo the commutativity and associativity of composition. For the induction step $n \geq 1$, the root of T has children T_j of height strictly less than n, for $j \in J$, where J is a nonempty set. By the inductive hypothesis, there exist terms t_j such that $t_j^{\mathbf{T}(\mathbb{B})} = T_j$, for all $j \in J$. Moreover, these terms are unique modulo commutativity and associativity of the composition. Let $c_i \in \mathbb{B}$, $i \in I$ be the unary labels of the root of T and $b_j \in \mathbb{B}$, $j \in J$ be the binary labels of the edges to which the root of T is attached. We consider the term $t \stackrel{\text{def}}{=} (\|_{i \in I}\, \mathbf{c}_i) \,\|\, \big(\, \|_{j \in J}\, \mathsf{append}_{b_j}(t_j)\big)$. Then, $t^{\mathbf{T}(\mathbb{B})} = T$ and, moreover, any other term u such that $u^{\mathbf{T}(\mathbb{B})} = T$ differs from t by a permutation of c_i and t_j. □

We prove below that recognizability of a set of graphs in a finite-sorted subalgebra $\mathbf{G}^\tau$ is preserved under inverse canonical evaluations. Note that this is not a consequence of the standard closure of recognizable sets under inverse homomorphisms, because **val** is not a homomorphism between $\mathbf{P}$ and $\mathbf{G}$. In fact, no such homomorphism exists because $\mathbf{P}$ has one sort $\{\mathfrak{r}\}$, whereas $\mathbf{G}$ has infinitely many sorts, i.e., all the finite subsets of $\mathbb{S}$.

Lemma 5.5. *Let $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ be a sort. For each set $\mathcal{L}$ of graphs, $\mathbf{val}^{-1}(\mathcal{L})$ is recognizable in $\mathbf{P}$ if $\mathcal{L}$ is recognizable in $\mathbf{G}^\tau$.*

Proof. Let $\mathbf{B} = (\{\mathcal{B}^{\tau'}\}_{\tau' \subseteq \tau}, \{f^{\mathbf{B}}\}_{f \in \mathcal{F}_{\mathsf{HR}}})$ be a locally finite algebra and $h : \mathcal{G}^\tau \to \mathcal{B}$ be a homomorphism between $\mathbf{G}^\tau$ and $\mathbf{B}$, such that $\mathcal{L} = h^{-1}(\mathcal{C})$, for a set $\mathcal{C} \subseteq \mathcal{B}$. Let

$\mathbf{B}' = (\{\mathcal{B}\}, \{f^{\mathbf{B}}\}_{f \in \mathcal{F}_{\mathsf{HR}}})$ be the algebra with a single sort $\{\mathfrak{r}\}$ and finite $\{\mathfrak{r}\}$-universe consisting of the union of all $\mathcal{B}^{\tau'}$, for $\tau' \subseteq \tau$. Then, $h \circ \mathbf{val}$ is a homomorphism between $\mathbf{P}$ and $\mathbf{B}'$ and, moreover, $\mathbf{val}^{-1}(\mathcal{L}) = (h \circ \mathbf{val})^{-1}(\mathcal{C})$. □

5.3. **Definable Transductions Define Context-Free Sets of Graphs.** A classical result is that each context-free word language is obtained from a recognizable ranked set of trees by reading the word on the frontier of each tree from left to right. This result has been generalized to context-free sets of graphs, by noticing that the *yield* that produces the word corresponding to the frontier of a tree is a definable transduction. The result of Courcelle and Engelfriet [CE95, Theorems 1.10 and 2.1] is that HR context-free sets of graphs are the images of recognizable sets of ground terms via definable transductions. We extend this result further, by removing the restriction on having an input set of ground terms. In particular, having unranked sets of trees is crucial to obtain the final characterization result in Theorem 6.10. We also restate some of the known results in our notation, in which parse trees are trees over a finite alphabet $\mathbb{B}^{\tau}_{\mathsf{parse}}$ of operations, that use only source labels from a given set $\tau \subseteq_{fin} \mathbb{S}$.

We start by restating a known result, namely that $\mathbf{val}\rfloor_{\mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})}$ is a definable transduction between parse trees and graphs, for every sort τ. Note that the restriction to the alphabet $\mathbb{B}^{\tau}_{\mathsf{parse}}$ of edge labels is necessary to ensure the finiteness of the formulae defining the transduction. Given a tree $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$, a source label s is said to be *present* in T if $\mathbf{val}(T)$ has an s-source. The following lemma shows that the presence of a source label in a tree is a definable property:

Lemma 5.6. *For each sort $\tau \subseteq_{fin} \mathbb{S}$ and each $s \in \mathbb{S}$, one can build an MSO sentence ϕ such that $\|T\| \models \phi$ iff s is present in T, for each tree $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$.*

Proof. Let $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$ be a tree. The construction of ϕ relies on the following equivalent condition, that can be easily expressed by a MSO sentence:

Fact 5.7. s is present in T iff there are $n_0, \ldots, n_m \in V_T$ and $s_0, \ldots, s_m \in \tau$, such that:

(1) n_0 is the root of T and $s_0 = s$,
(2) n_{i+1} is a child of n_i in T, for all $i \in [0, m-1]$,
(3) n_m is attached to an edge labeled by a unary label, which is either $\underline{\mathbf{0}}_{\tau'}$ and $s_m \in \tau'$, for some $\tau' \subseteq \tau$, or $\underline{\mathbf{a}}_{(s'_1,\ldots,s'_k)}$ and $s_m \in \{s'_1, \ldots, s'_k\}$,
(4) the edge between n_i and n_{i+1} is labeled by a binary label, either $\underline{\mathsf{restrict}}_{\tau'}$ and $s_i = s_{i+1} \in \tau'$, or $\underline{\mathsf{rename}}_{\alpha}$ and $\alpha(s_i) = s_{i+1}$.

Proof. "$\Rightarrow$" By Proposition 5.4, we have $T = t^{\mathbf{T}(\mathbb{B}_{\mathsf{parse}})}$, for some term $t = \big(\|_{i \in I}\, \mathbf{c}_i \big) \parallel \big(\|_{j \in J}\, \mathsf{append}_{b_j}(t_j) \big)$. Then s is present in T because either:

- ▷ $\mathbf{c}_i = \underline{\mathbf{0}}'_{\tau}$, for some $i \in I$, such that $s \in \tau'$,
- ▷ $\mathbf{c}_i = \underline{\mathbf{a}}_{(s'_1,\ldots,s'_k)}$, for some $i \in I$, such that $s \in \{s'_1, \ldots, s'_k\}$,
- ▷ $b_j = \mathsf{restrict}_{\tau'}$, for some $j \in J$, such that $s \in \tau'$ and s is present in $t_j^{\mathbf{T}}$,
- ▷ $b_j = \mathsf{rename}_{\alpha}$, for some $j \in J$, such that $s = \alpha(s')$ and s' is present in $t_j^{\mathbf{T}}$,

In the first two cases, we set $m = 0$, n_0 the root of T and $s_0 \stackrel{\mathsf{def}}{=} s$. In the last two cases, we set n_0 as the root of T, $s_0 \stackrel{\mathsf{def}}{=} s$ and continue building $n_1, \ldots, n_m$ and $s_1, \ldots, s_m$ from t_j. It is easy to check that the conditions (1-4) are satisfied by the sequences $n_0, \ldots, n_m$ and $s_0, \ldots, s_m$ built as described above.

"$\Leftarrow$" Let $T_0, \ldots, T_m$ be the subtrees of T rooted in $n_0, \ldots, n_m$, respectively. By condition (2) T_{i+1} is a subtree of T_i, for each $i \in [0, m-1]$. By induction on $m - i$, one shows that s_i is present in T_i, for all $i \in [0, m]$. The base case $i = m$ follows from condition (3). The inductive case $i < m$ follows from condition (4). Since $s_0 = s$ and n_0 is the root of T, by condition (1), s is present in T. □

Back to the construction of ϕ, the existence of a path starting in the root can be described by an MSO formula $\psi(X)$ in the relational signature $\mathbb{R}_{\mathsf{tree}}(\mathbb{B}^\tau_{\mathsf{parse}})$. Further, the local conditions (3) and (4) can be encoded by formulæ $\eta_1(x)$ and $\eta_2(x, y)$, respectively. Note that τ being a given finite set is crucial in encoding the conditions such as $s \in \tau'$ and $s = \alpha(s')$, for $\tau' \subseteq \tau$ and α a τ-permutation by finite formulæ. Finally, we define:

$$\phi \stackrel{\mathsf{def}}{=} \exists X \,.\, \psi(X) \wedge \big(\exists x \,.\, X(x) \wedge \eta_1(x)\big) \wedge \big(\forall x \forall y \,.\, X(x) \wedge X(y) \wedge \bigvee_{b \in \mathbb{B}^\tau_{\mathsf{parse}}} \exists z \,.\, \mathsf{r}_b(z, x, y) \rightarrow \eta_2(x, y)\big)$$

□

A statement similar to the next lemma is proved in [CE12, Proposition 7.48]. For reasons of self-containment, we give a proof using our notation, that makes the set of used source labels explicit:

Lemma 5.8. *For each $\tau \subseteq_{fin} \mathbb{S}$, $\mathbf{val}\rfloor_{\mathcal{T}(\mathbb{B}^\tau_{\mathsf{parse}})}$ is a definable $(\mathbb{R}_{\mathsf{tree}}(\mathbb{B}^\tau_{\mathsf{parse}}), \mathbb{R}^\tau_{\mathsf{graph}})$-transduction.*

Proof. We define the desired transduction in two steps. In the first step, we expand each node of the input tree into at most $\mathrm{card}(\tau) + 1$ many nodes, one for each source that is present in the respective subtree plus one extra node that represents an edge. In the second step, we merge the nodes that are fused by the composition operations. The first step uses an extra binary relation symbol $\equiv$ that keeps track of the nodes which are to be merged in the second step. This relation symbol is interpreted over different layers by formulæ in the relational signature $\mathbb{R}^\tau_{\mathsf{graph}}$.

We now describe the first step. We use a transduction that creates $\mathrm{card}(\tau) + 1$ copies of the input structure. We will use the sources in τ and an additional source label $\Box \in \mathbb{S} \setminus \tau$ to index the copies of the input structure. Formally, we define a parameterless transduction scheme $\Theta = \langle \varphi, \{\psi_s\}_{s \in \tau \cup \{\Box\}}, \{\theta_{(a, s_1, \ldots, s_{\#a})}\}_{a \in \mathbb{A}, s_1, \ldots, s_{\#a} \in \tau} \cup \{\theta_{(\equiv, s, t)}\}_{s, t \in \tau} \rangle$, as follows:

- ▷ φ specifies the domain of the transduction, i.e., φ expresses that the input structure is the encoding $\|T\|$ of some tree $T \in \mathcal{T}(\mathbb{B}^\tau_{\mathsf{parse}})$. It is easy to verify that such an MSO-formula can be built.
- ▷ ψ_s defines the universe of the s-th layer of the result, for each $s \in \tau$. Namely, ψ_s holds for an element of the universe of the input structure iff this element is a vertex and if s is present at the subtree rooted at this vertex (these elements will represent the vertices of the output structure). Such a formula can be built according to Lemma 5.6. Moreover, $\psi_\Box$ holds for all elements of the input structure that are edges labeled by unary symbols $\underline{\mathbf{a}}_{(s_1, \ldots, s_n)}$ (these elements will represent the edges of the output structure).
- ▷ each $\theta_{(a, s_1, \ldots, s_{\#a})}$ has free variables $x_0, x_1, \ldots, x_{\#a}$ and defines the interpretation of $(\mathsf{r}_a, s_1, \ldots, s_{\#a})$ in the result, for all $a \in \mathbb{A}$. We define $\theta_{(a, s_1, \ldots, s_{\#a})}$ to hold for tuples $((u_0, \Box), (u_1, s_1), \ldots, (u_{\#a}, s_{\#a}))$ iff u_0 represents a graph edge labeled by the unary symbol $\underline{\mathbf{a}}(s_1, \ldots, s_{\#a})$, such that $u_1 = \ldots = u_{\#a}$ is a tree node incident to u_0. It is easy to build an MSO-formula $\theta_{(a, s_1, \ldots, s_{\#a})}$ defining these properties.
- ▷ each $\theta_{(\equiv, s, t)}$, for $s, t \in \tau$, has free variables x_1 and x_2, and defines the interpretation of $(\equiv, s, t)$ in the result. For a tuple $((u_1, s), (u_2, t))$, we define $\theta_{(s, \equiv, t)}$ to hold iff u_1 and u_2

are tree nodes, such that u_2 is the child of u_1, for some tree edge labeled by one of the following symbols:
– $\underline{\mathsf{restrict}}_{\tau'}$, such that $s \in \tau'$ and $s = t$, or
– $\underline{\mathsf{rename}}_{\alpha}$, such that $\alpha(s) = t$.
It is easy to build an MSO formula $\theta_{(\equiv,s,t)}$ defining these properties.

The second step of the construction is a transduction that takes the least equivalence relation that subsumes the relation defined by the MSO formula $\theta_{\equiv} \stackrel{\text{def}}{=} \bigvee_{s,t\in\tau} \theta_{(\equiv,s,t)}$ and constructs its quotient structure. It is well known that the quotient structure with regard to an equivalence relation definable in MSO can be expressed as a definable transduction, e.g., see [Cou91, Lemma 2.4]. It is now routine to verify that the composition of the two transductions above has the desired properties. Moreover, by Proposition 2.3 (2), the composition of definable transductions is definable. □

Following a standard convention [CDG+08, CE95], we identify a ground term t over a finite functional signature $\mathcal{F}$ with a tree, such that each node $n \in V_t$ is attached to exactly one unary edge labeled with a function symbol $f \in \mathcal{F}$ and is, moreover, the source of $\#f$ binary edges labeled $1, \ldots, \#f$. The intuition is that the destination of the edge labeled i is the i-th child of n. More precisely, a ground term $t \in \mathcal{M}(\mathcal{F})$ is the tree $t \in \mathcal{T}(\mathbb{B}_{\mathcal{F}})$, where $\mathbb{B}_{\mathcal{F}} \stackrel{\text{def}}{=} \mathcal{F} \cup [1, \max_{f\in\mathcal{F}} \#f]$. Since each ground term is a graph, we write $\|t\|$ for the relational structure encoding the term t (subsection 3.3).

With these conventions in mind, we recall a characterization of context-free sets of graph as images of recognizable set of terms via definable transductions, found by Courcelle and Engelfriet [CE95, Theorems 1.10 and 2.1]. The statement below is given according to our definitions:

Theorem 5.9 [CE95]. *A set of graphs $\mathcal{L} \subseteq \mathcal{G}^{\emptyset}$ is context-free iff there exists:*
(1) *a finite functional signature $\mathcal{F}$,*
(2) *a set $\mathcal{K} \subseteq \mathcal{M}(\mathcal{F})$ of ground terms recognizable in $\mathbf{M}(\mathcal{F})$, and*
(3) *a definable $(\mathcal{F},\ \mathbb{R}^{\emptyset}_{\mathsf{graph}})$-transduction F,*
such that $\|\mathcal{L}\| = F(\|\mathcal{K}\|)$.

We extend below the result of Theorem 5.9 from terms to unranked trees. The main difficulty here is the difference between the initial algebra $\mathbf{M}(\mathcal{F})$ over a finite functional signature $\mathcal{F}$ and the tree algebra $\mathbf{T}(\mathbb{B})$ over a finite edge label alphabet $\mathbb{B}$. Note that a tree is built in $\mathbf{M}(\mathcal{F})$ by taking all the subtrees starting at the children of the root at once, whereas in $\mathbf{T}(\mathbb{B})$, building the same tree requires a series of append_b and $\|$ operations.

Corollary 5.10. *A set of graphs $\mathcal{L} \subseteq \mathcal{G}^{\emptyset}$ is context-free iff there exists:*
(1) *a finite alphabet $\mathbb{B}$ of edge labels,*
(2) *a (possibly unranked) set $\mathcal{K} \subseteq \mathcal{T}(\mathbb{B})$ of trees recognizable in $\mathbf{T}(\mathbb{B})$, and*
(3) *a definable $(\mathbb{R}_{\mathsf{tree}}(\mathbb{B}),\ \mathbb{R}^{\emptyset}_{\mathsf{graph}})$-transduction F,*
such that $\|\mathcal{L}\| = F(\|\mathcal{K}\|)$.

Proof. "⇒" By the left to right direction of Theorem 5.9, there exists a finite functional signature $\mathcal{F}$, a recognizable set $\mathcal{K}$ of ground terms over $\mathcal{F}$ and a definable $(\mathcal{F},\ \mathbb{R}^{\emptyset}_{\mathsf{graph}})$-transduction F, such that $\|\mathcal{L}\| = F(\|\mathcal{K}\|)$. Note that the alphabet $\mathbb{B}_{\mathcal{F}}$ of unary and binary edge labels is finite, because $\mathcal{F}$ is finite. We take $\mathbb{B}$ to be $\mathbb{B}_{\mathcal{F}}$, in the following. Since $\mathcal{K}$ is recognizable in $\mathbf{M}(\mathcal{F})$, the set $\|\mathcal{K}\|$ is MSO-definable, by the classical equivalence between recognizability in $\mathbf{M}(\mathcal{F})$ and MSO-definability of sets of ground terms over a finite functional signature $\mathcal{F}$ [Don70, EW67]. Since the set of trees $\mathcal{K} \in \mathcal{T}(\mathbb{B})$ is MSO-definable, it is recognizable in $\mathbf{T}(\mathbb{B})$, by Theorem 4.1.

"$\Leftarrow$" Let $\mathcal{K}$ be a set of trees recognizable in $\mathbf{T}(\mathbb{B})$ and let $\mathcal{T} \stackrel{\text{def}}{=} \{t \in \mathcal{M}(\mathcal{F}_{\mathsf{tree}}(\mathbb{B})) \mid t^{\mathbf{T}(\mathbb{B})} \in \mathcal{K}\}$ be the set of ground $\mathcal{F}_{\mathsf{tree}}(\mathbb{B})$-terms that evaluate to a tree in $\mathcal{K}$. By Theorem 4.1, the set $\|\mathcal{K}\|$ is definable. We denote by $\overline{\mathbf{val}}$ the function that maps $\|t\|$ into $\|t^{\mathbf{T}(\mathbb{B})}\|$, for each $t \in \mathcal{M}(\mathcal{F}_{\mathsf{tree}}(\mathbb{B}))$. Assume that $\overline{\mathbf{val}}$ is a definable transduction. Then, $\|\mathcal{T}\| = \overline{\mathbf{val}}^{-1}(\|\mathcal{K}\|)$ is definable, by Theorem 2.2. By the classical result of [Don70, TW68], the set $\mathcal{T}$ of ground $\mathcal{F}_{\mathsf{tree}}(\mathbb{B})$-terms is recognizable in $\mathcal{M}(\mathcal{F}_{\mathsf{tree}}(\mathbb{B}))$. Since $\overline{\mathbf{val}}$ was assumed to be definable, the transduction $\overline{\mathbf{val}} \circ F$ is definable, by Proposition 2.3 (1), thus $\mathcal{L}$ is context-free, by Theorem 5.9. It remains to show that $\overline{\mathbf{val}}$ is a definable transduction. Let $\mathfrak{r}$ be the (only) source label necessary to build any parse tree (over the alphabet $\mathbb{B}^{\{\mathfrak{r}\}}_{\mathsf{parse}}$). By Lemma 5.8, $\mathbf{val}\downharpoonright_{\mathcal{T}(\mathbb{B}^{\{\mathfrak{r}\}}_{\mathsf{parse}})}$ is a definable transduction. Then, $\overline{\mathbf{val}} = \mathbf{val}\downharpoonright_{\mathcal{T}(\mathbb{B}^{\{\mathfrak{r}\}}_{\mathsf{parse}})} \circ \delta$, where δ is the transduction that turns ground terms into parse trees. It is easy to see that δ is definable, hence $\overline{\mathbf{val}}$ is definable, by Proposition 2.3 (1). □

6. Parsable Sets of Graphs

This section gives the main result of the paper, i.e., a characterization of the class of definable context-free sets of graphs. Essentially, we show that these are exactly the *parsable* sets, for which the parse trees of the grammar can be extracted from each graph, by means of a definable transduction.

Parsable sets are closely related to the notion of *strongly context-free* sets of graphs introduced by Courcelle [Cou91, Definition 4.2]. According to the original definition, a strongly context-free set $\mathcal{L}$ is the image of a set $\mathcal{K}$ of parse trees over a finite signature of hyperedge-replacement operations (i.e., graphs G having special nonterminal edges $e \in E_G$, that can be substituted by any graph H of sort τ with $\mathrm{card}(\tau) = \#\lambda_G(e)$ via the canonical evaluation function $\mathbf{val}$), such that, moreover, there exists a definable transduction $\pi \subseteq (\mathbf{val}\downharpoonright_{\mathcal{K}})^{-1}$ such that $\mathrm{dom}(\pi) = \mathcal{L}$. Here, by *parsable* set, we denote the existence of the inverse transduction between graphs and parse trees over $\mathcal{F}_{\mathsf{HR}}$:

Definition 6.1. Let $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ be a sort. A set of graphs $\mathcal{L} \subseteq \mathcal{G}$ is *τ-parsable* iff there exists a definable $(\mathbb{R}^{\tau}_{\mathsf{graph}}, \mathbb{R}_{\mathsf{tree}}(\mathbb{B}^{\tau}_{\mathsf{parse}}))$-transduction π such that:
(1) $\|\mathcal{L}\| = \mathrm{dom}(\pi)$, and
(2) if $(\|G\|, \|T\|) \in \pi$ then $\mathbf{val}(T) = G$.
We call a set of graphs $\mathcal{L} \subseteq \mathcal{G}$ *parsable*, if $\mathcal{L}$ is *τ-parsable* for some $\tau \subseteq_{\mathit{fin}} \mathbb{S}$.

Choosing $\mathcal{K}$ such that $\|\mathcal{K}\| = \pi(\|\mathcal{L}\|)$ for a parsable set of graphs $\mathcal{L}$, we obtain that $\mathcal{L} = \mathbf{val}(\mathcal{K})$, i.e., $\mathcal{L}$ is strongly context-free in the sense of [Cou91, Definition 4.2] (noting that the $\mathcal{F}_{\mathsf{HR}}$ operations can be expressed as hyperedge-replacement operations). On the other hand, the original notion of strongly context-free set may appear, at first sight, to be more general, because it refers to any signature of hyperedge-replacement operations, whereas our notion of parsable set fixes the signature to $\mathcal{F}_{\mathsf{HR}}$. However, every hyperedge-replacement operation can be expressed by a $\mathcal{F}_{\mathsf{HR}}$ term and it is straightforward to define transductions that replace parse trees of hyperedge-replacement operations by $\mathcal{F}_{\mathsf{HR}}$ parse trees that encode these operations.

A first result of this section (subsection 6.1) is that the set of graphs whose tree-widths are bounded by a constant is parsable (Theorem 6.8), thus proving Conjecture 1.2. As stated in the introduction, this also establishes Conjecture 1.1, which states that the definable and context-free sets of graphs are exactly the parsable ones. The first step in order to prove Theorem 6.8 is the ability of extracting an optimal-width tree decomposition from a graph by means of a definable transduction (for this we use [BP16, Theorem 2.4] and [BP22, Theorem 2.1]). For the second step, we then prove that each tree decomposition can be further translated into a parse tree by a definable transduction.

Finally, we give two theorems that characterize the definable context-free sets of graphs using four equivalent conditions (subsection 6.2). The first theorem (Theorem 6.9) is explicit about the finite set of sources used in the grammar. The second theorem (Theorem 6.10) quantifies this set existentially in each condition and provides, in addition to Theorem 6.9, a purely logical condition, in terms of a pair of definable transductions that act as encoding (graphs to trees) and decoding (trees to graphs).

We start by proving that the recognizable subsets of a parsable set are also parsable:

Lemma 6.2. *Let $\tau \subseteq \mathbb{S}$ be a sort, $\mathcal{L}$ be a τ-parsable set of graphs and $\mathcal{L}' \subseteq \mathcal{L}$ be a set recognizable in $\mathbf{G}^\tau$. Then, $\mathcal{L}'$ is τ-parsable.*

Proof. Let π be the definable $(\mathbb{R}^\tau_{\mathsf{graph}}, \mathbb{R}_{\mathsf{tree}}(\mathbb{B}^\tau_{\mathsf{parse}}))$-transduction that witnesses the parsability of $\mathcal{L}$ as in Definition 6.1. Then $\mathcal{K} \stackrel{\text{def}}{=} \mathbf{val}^{-1}(\mathcal{L}')$ is a set of parse trees for $\mathcal{L}'$. By Lemma 5.5, $\mathcal{K}$ is recognizable in $\mathbf{P}$. By Lemma 5.3, $\mathcal{K}$ is also recognizable in $\mathbf{T}(\mathbb{B}_{\mathsf{parse}})$, hence $\mathcal{K}$ is definable, by Theorem 4.1. By Theorem 2.2, we obtain that $\mathcal{L}' = \pi^{-1}(\mathcal{K})$ is definable, hence the domain-restriction of π to $\mathcal{L}'$ is definable, by Proposition 2.3 (1). $\square$

The "only if" direction of Conjecture 1.1 is proved next. The "if" direction will be proved as part of Theorem 6.9.

Proposition 6.3. *Any τ-parsable set of graphs is both definable and context-free, defined by a grammar over the signature $\mathcal{F}^\tau_{\mathsf{HR}}$.*

Proof. Let $\mathcal{L}$ be a τ-parsable set of graphs, for a sort $\tau \subseteq_{\mathit{fin}} \mathbb{S}$. Then, $\mathcal{L}$ is definable because π is definable, thus $\|\mathcal{L}\| = \mathrm{dom}(\pi)$ is definable, where π is the $(\mathbb{R}^\tau_{\mathsf{graph}}, \mathbb{R}_{\mathsf{tree}}(\mathbb{B}^\tau_{\mathsf{parse}}))$-transduction from Definition 6.1. By Theorem 3.5, $\mathcal{L}$ is recognizable in $\mathbf{G}$. Since $\mathcal{L}$ is τ-parsable, we obtain that $\mathcal{L}$ is also recognizable in the subalgebra $\mathbf{G}^\tau_{\mathsf{gen}}$. This is because every graph in $\mathcal{L}$ can be built using only operations that are the interpretations of the function symbols from $\mathcal{F}^\tau_{\mathsf{HR}}$, which occur on the edge labels of some parse tree from $\mathcal{T}(\mathbb{B}^\tau_{\mathsf{parse}})$. By Proposition 5.1, we have that set of graphs $\mathcal{G}^\tau_{\mathsf{gen}}$ is context-free. Hence, $\mathcal{L} = \mathcal{L} \cap \mathcal{G}^\tau_{\mathsf{gen}}$ is context-free by Theorem 2.10. $\square$

6.1. **Parsing with Tree Decompositions.** The definition of parsable sets of graphs requires a definable transduction from graphs to trees that produces, for each input graph, a parse tree of that graph relative to some grammar, that does not depend on the input graph. A candidate for such a parse tree is any tree decomposition that witnesses the tree-width of a graph. We recover such an optimal tree decomposition from a seminal result of Bojańczyk and Pilipczuk that states the existence of a definable transduction which computes some optimal tree decomposition of a given graph. The following theorem combines the results of [BP16, Theorem 2.4] and [BP22, Theorem 2.1]:

Theorem 6.4 [BP16, BP22]. *For every $k \in \mathbb{N}$, there exists a definable $(\mathbb{R}^{\emptyset}_{\mathsf{graph}}, \mathbb{R}^{\emptyset}_{\mathsf{decomp}})$-transduction $\mathcal{I}$, such that the following holds:*

(1) $\mathsf{S} \in \mathrm{dom}(\mathcal{I})$ *iff* $\mathsf{S} = \|G\|$ *for some graph G, such that* $\mathrm{twd}(G) \leq k$,
(2) *if* $(\|G\|, \mathsf{S}) \in \mathcal{I}$ *for some graph G, then* $\mathsf{S} = \|G, T, \beta\|$ *for some tree decomposition* (T, β) *of G of width at most k.*

We show next that each encoding of some tree decomposition of a graph G can be mapped to a parse tree that evaluates to G via the canonical evaluation:

Lemma 6.5. *For all sorts $\tau \subseteq_{fin} \mathbb{S}$ and $\tau' \subseteq \tau$, there is a definable $(\mathbb{R}^{\tau'}_{\mathsf{decomp}}, \ \mathbb{R}_{\mathsf{tree}}(\mathbb{B}^{\tau}_{\mathsf{parse}}))$-transduction $\mathcal{J}$, such that:*

(1) $\mathsf{S} \in \mathrm{dom}(\mathcal{J})$ *iff* $\mathsf{S} = \|G, D, \beta\|$ *for some graph G with* $\mathrm{twd}(G) \leq \mathrm{card}(\tau) - 1$, *witnessed by a tree decomposition (D, β), and*
(2) *if* $(\|G, D, \beta\|, \|T\|) \in \mathcal{J}$ *then* $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$ *and* $\mathbf{val}(T) = G$.

Proof. Let $k = \mathrm{card}(\tau) - 1$. The idea of the transduction $\mathcal{J}$ is to use the tree D, encoded by the interpretation of the node and parent relation symbols from $\mathbb{R}^{\tau}_{\mathsf{decomp}}$, as the skeleton for the output tree T. In order to label the edges of T with unary and binary edge labels, we guess a coloring of the vertices in the input graph, using the parameters $\{X_s\}_{s\in\tau}$, such that every vertex is labeled by exactly one color X_s. Given a node $n \in V_D$, let $\mathtt{colors}(n) \stackrel{\text{def}}{=} \{s \mid$ there is a vertex v colored by X_s and $\mathsf{bag}(v, n)$ holds$\}$ be the colors of the vertices in the bag $\beta(n)$. Moreover, for every edge $e \in E_G$, we let $\mathtt{node}(e)$ be the closest node n to the root with $\upsilon_G(e)_i \in \beta(n)$, for all $1 \leq i \leq \#\lambda_G(e)$. Note that $\mathtt{node}(e)$ exists by Definition 4.3 (1) and that $\mathtt{node}(e)$ is unique, by Definition 4.3 (2) (i.e., if there exist two distinct nodes containing all vertices attached to e, then all these vertices must also belong to the bag of their unique common ancestor).

We are going to use a transduction that creates three layers (i.e., copies of the input structure) indexed by the names vertex, source and edge, respectively. Then, $\mathcal{J}$ is the transduction defined by the scheme:

$$\Theta \stackrel{\text{def}}{=} \langle \varphi, \psi_{\mathsf{vertex}}, \psi_{\mathsf{source}}, \psi_{\mathsf{edge}}, \{\theta_{\underline{\mathsf{restrict}}_{\tau'}}\}_{\tau' \subseteq \tau}, \{\theta_{\underline{\mathsf{rename}}_{\alpha}}\}_{\alpha \text{ is } \tau\text{-permutation}}, \{\theta_{\underline{\mathbf{a}}(s_1,\ldots,s_{\#a})}\}_{a\in\mathbb{A}, s_1,\ldots,s_{\#a}\in\tau}, \{\theta_{\underline{\mathbf{0}}_{\tau'}}\}_{\tau'\subseteq\tau}\rangle$$

where:

▷ $\varphi(\{X_s\}_{s\in\tau})$ defines the domain of the transduction, by checking that the following hold:
1. the sets $\{X_s\}_{s\in\tau}$ forms a partition of the vertices of V_G, i.e., every vertex is labeled of V_G is labelled by exactly one color X_s,
2. the bags of the tree decomposition are all of size at most $k+1$,
3. for each edge $e \in E_G$, there is a node $n \in V_D$ whose bag $\beta(n)$ contains all vertices from V_G attached to e,
4. the set of nodes $\{n \in V_D \mid v \in \beta(n)\}$ is non-empty and connected in D,
5. the sets $\{X_s\}_{s\in\tau}$ form a partition of V_G that is consistent with the tree decomposition, i.e., that in each bag there is at most one vertex labelled by X_s, for all $s \in \tau$, and
6. for each s-source $v \in V_G$, the color of v is indeed X_s and s belongs to the bag associated with the root of D.

▷ $\psi_{\mathsf{vertex}}(x_1) \stackrel{\text{def}}{=} \mathsf{node}(x_1)$ represents the nodes of the output tree T.

▷ $\psi_{\mathsf{source}}(x_1) \stackrel{\text{def}}{=} \mathsf{node}(x_1)$ represents the unary edges with labels $\underline{\mathbf{0}}_{\tau'}$ of T.

▷ $\psi_{\mathsf{edge}}(x_1)$ holds for those elements where $\mathsf{node}(x_1)$ holds, except for the root of the tree; these elements represent the binary $\underline{\mathsf{restrict}}_{\tau'}$-labeled edges of T. Moreover, $\psi_{\mathsf{edge}}(x_1)$ holds also for the elements that encode the edges of G; these elements represent the unary $\underline{\mathbf{a}}_{(s_1,\ldots,s_{\#a})}$-labeled edges of T.

▷ $\theta_{\underline{\mathsf{restrict}}_{\tau'}}(x_1, x_2, x_3, \{X_s\}_{s\in\tau})$ defines the interpretation of the ternary relation symbol $\mathsf{r}_{\underline{\mathsf{restrict}}_{\tau'}}$ in $\|T\|$, i.e., all triples $\langle(n_1, \mathsf{edge}), (n_2, \mathsf{vertex}), (n_3, \mathsf{vertex})\rangle \in V_T^3$, such that n_3 is the parent of n_2, $n_1 = n_2$ and $\tau' = \mathtt{colors}(n_2) \cap \mathtt{colors}(n_3)$.

▷ $\theta_{\underline{\mathsf{rename}}_\alpha}(x_1, x_2, x_3, \{X_s\}_{s\in\tau})$ is set to false for all τ-permutations α as the rename operation is not needed for the construction of graphs from tree decompositions,

▷ $\theta_{\underline{\mathbf{a}}_{(s_1,\ldots,s_{\#a})}}(x_1, x_2, \{X_s\}_{s\in\tau})$ defines the interpretation of the binary relation symbol $\mathsf{r}_{\underline{\mathbf{a}}_{(s_1,\ldots,s_{\#a})}}$ in $\|T\|$, i.e., all pairs $\langle(n_1, \mathsf{edge}), (n_2, \mathsf{vertex})\rangle \in V_T^2$, such that $n_1 \in E_G$ is an edge with label $\lambda_G(n_1) = a$ and incident vertices $\upsilon_G(n_1) = \langle v_1, \ldots, v_{\#a}\rangle$ colored by $X_{s_1}, \ldots, X_{s_{\#a}}$, respectively, and $n_2 = \mathtt{node}(n_1)$.

▷ $\theta_{\underline{\mathbf{0}}_{\tau'}}(x_1, x_2, \{X_s\}_{s\in\tau})$ defines the interpretation of the binary relation symbol $\mathsf{r}_{\underline{\mathbf{0}}_{\tau'}}$ in $\|T\|$, i.e., all pairs $\langle(n_1, \mathsf{source}), (n_2, \mathsf{vertex})\rangle \in V_T^2$, such that $n_1 = n_2$ and $\tau' = \mathtt{colors}(n_1)$.

Note that the formula φ ensures that $\mathsf{S} \in \mathrm{dom}(\mathcal{J})$ iff $\mathsf{S} = \|(G, D, \beta)\|$ for some graph G with $\mathrm{twd}(G) \leq k$, witnessed by a tree decomposition (D, β).

Let $(\|(G, D, \beta)\|, \|T\|) \in \mathcal{J}$ for some G with $\mathrm{twd}(G) \leq k$, witnessed by a tree decomposition (D, β). It is easy to verify that $T \in \mathcal{T}(\mathbb{B}^\tau_{\mathsf{parse}})$. We note that every node of n of the tree T is also a node of D and vice versa, by the definition of the transduction scheme Θ. For each $n \in V_T$, let T_n denote the subtree of T rooted in n. Let $\{X_s\}_{s\in\tau}$ be the coloring guessed by the transduction. For a node $n \in V_D$, we denote by $\mathtt{graph}(n)$ the subgraph of G consisting of all vertices that appear in bags of descendants of n and edges e, such that $\mathtt{node}(e)$ is a descendant of n. We mark a vertex v of $\mathtt{graph}(n)$ as an s-source iff v appears in the bag associated with n and is colored by X_s. We show the following:

Fact 6.6. $\mathbf{val}(T_n) = \mathtt{graph}(n)$, for all $n \in V_T$.

Proof. By induction on the structure of T, let $n \in V_T$ be a node with children $n_1, \ldots, n_l$. By the inductive hypothesis, we have that $\mathbf{val}(T_{n_i}) = \mathtt{graph}(n_i)$, for all $i \in [1, l]$. We denote by $\tau_i \stackrel{\mathsf{def}}{=} \{s \in \tau \mid \exists v \in V_G \;.\; v \text{ is colored by } X_s,\ \mathsf{bag}(v, n) \text{ and } \mathsf{bag}(v, n_i) \text{ hold}\}$ the set of labels that occur simultaneously in the bags of n and n_i, for all $i \in [1, l]$. We consider G_0 to be the subgraph of G consisting of the vertices of G from the bag of n and the edges $e \in E_G$ such that $n = \mathtt{node}(e)$. Let $\tau_0 \stackrel{\mathsf{def}}{=} \{s \in \tau \mid \exists v \in V_{G_0} \;.\; v \text{ is colored by } X_s\}$. Then, we have:

$$G_0 = \left(\mathbf{0}_{\tau_0} \parallel (\|_i\ \mathbf{a_i}(s^i_1, \ldots, s^i_{n_i}))\right)^{\mathbf{G}} = \mathbf{0}^{\mathbf{G}}_{\tau_0} \parallel^{\mathbf{G}} (\|^{\mathbf{G}}_i\ \mathbf{a_i}(s^i_1, \ldots, s^i_{n_i})^{\mathbf{G}})$$

for suitably chosen edge labels $a_i \in \mathbb{A}$ and source labels $s^i_j \in \tau$. We observe that:

$$\mathtt{graph}(n) = G_0 \parallel^{\mathbf{G}} (\|^{\mathbf{G}}_{i=1..l}\ \mathsf{restrict}^{\mathbf{G}}_{\tau_i}(\mathtt{graph}(n_i)))\ (*)$$

By the definition of the canonical evaluation, we have:

$$\mathbf{val}(T_n) = \mathbf{0}^{\mathbf{G}}_{\tau_0} \parallel^{\mathbf{G}} (\|^{\mathbf{G}}_i\ \mathbf{a_i}(s^i_1, \ldots, s^i_n)^{\mathbf{G}}) \parallel^{\mathbf{G}} (\|^{\mathbf{G}}_{i=1..l}\ \mathsf{restrict}^{\mathbf{G}}_{\tau_i}(\mathbf{val}(T_{n_i})))$$

The claim follows by the above equation, (*) and the inductive hypothesis. □

The proof is concluded by choosing n as the root of T in the above fact, which leads to $\mathbf{val}(T) = G$, as required. □

The next corollary of Theorem 6.4 provides a powerful result. By instantiating its statement for $\tau = \emptyset$, we obtain that every graph $G \in \mathcal{G}^\emptyset$ of tree-width k is the image of a parse tree under an MSO-transduction and, moreover, this parse tree uses exactly $k + 1$ source labels:

Corollary 6.7. *For all sorts $\tau \subseteq_{fin} \mathbb{S}$ and $\tau' \subseteq \tau$, there exists a definable $(\mathbb{R}^{\tau'}_{\mathsf{graph}}, \mathbb{R}_{\mathsf{tree}}(\mathbb{B}^{\tau}_{\mathsf{parse}}))$-transduction $\mathcal{K}$, such that:*

(1) $\mathsf{S} \in \mathrm{dom}(\mathcal{K})$ *iff* $\mathsf{S} = \|G\|$*, for some graph G that admits a tree decomposition (D, β) of width at most* $\mathrm{card}(\tau) - 1$*, and*
(2) *if* $(\|G\|, \|T\|) \in \mathcal{K}$ *then* $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$ *and* $\mathbf{val}(T) = G$.

Proof. The transduction $\mathcal{K}$ is the result of composing the transductions $\mathcal{I}$ (Theorem 6.4) and $\mathcal{J}$ (Lemma 6.5). However, as $\mathcal{I}$ only inputs graphs with no sources, we need to add some pre-processing in order to encode and then remove the sources. We will define two further definable transductions A and B such that the desired transduction $\mathcal{K}$ is the result of composing A, $\mathcal{I}$, B and $\mathcal{J}$, in this order, starting with A. The pre-processing requires the use of some fresh (temporary) edge label a of arity $\mathrm{card}(\tau')$. We now define A as the transduction that outputs the encoding of an input graph $G \in \mathcal{G}^{\tau'}_{\mathsf{gen}}$, removes the encoding of the sources r_s, for $s \in \tau$, and adds an a-labelled edge between all s-sources, with $s \in \tau'$. This has the effect that for every tree-decomposition of G (in particular for the output of the composition of transductions A and $\mathcal{I}$) there is a node such that the bag associated with this node contains all s-sources, for $s \in \tau'$, of G. Next, we would like to apply transduction $\mathcal{J}$. However, in order to do so, we need to ensure that the s-sources in fact appear in the root of the tree decomposition and that we reintroduce the sources $s \in \tau'$ at the appropriate places. We do so by defining the transduction B that inputs the encoding a graph G and a tree decomposition (D, β) and outputs an encoding of G and a tree decomposition (D', β'), which is obtained from (D, β) by rotating the tree-decomposition, such that the node of D that contains the s-sources becomes the root of the tree decomposition. The rotation operation is implemented by guessing the root of the rotated tree and reversing the order of the pairs from the interpretation of the of the parent relation along the unique path between the root of the input tree and the new root. Further, B deletes the a-labelled edge that has been added by A and reintroduces the predicates r_s, for $s \in \tau$. It is now easy to verify that the composition of A, $\mathcal{I}$, B and $\mathcal{J}$ has the desired properties. □

We are now ready to prove Conjecture 1.2:

Theorem 6.8. *For each sort $\tau \subseteq_{fin} \mathbb{S}$, the set $\mathcal{G}^{\tau}_{\mathsf{gen}}$ is τ-parsable.*

Proof. By Theorem 4.4, we have $\mathrm{twd}(G) \le \mathrm{card}(\tau) - 1$ for every graph $G \in \mathcal{G}^{\tau}_{\mathsf{gen}}$, witnessed by a tree decomposition (D, β) such that every s-source of G, with $s \in \tau$, appears in the bag associated with the root of D. Hence, $\|\mathcal{G}^{\tau}_{\mathsf{gen}}\| \subseteq \mathrm{dom}(\mathcal{K})$, where $\mathcal{K}$ is the definable transduction given by Corollary 6.7. Moreover, we have that $(\|G\|, \|T\|) \in K$ implies $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$ and $\mathbf{val}(T) = G$, hence $\|\mathcal{G}^{\tau}_{\mathsf{gen}}\| \supseteq \mathrm{dom}(\mathcal{K})$. To see that $\mathcal{G}^{\tau}_{\mathsf{gen}}$ is τ-parsable, we can take $\pi \stackrel{\mathsf{def}}{=} \mathcal{K}$ as required by Definition 6.1. □

6.2. **Two Characterizations of Definable Context-Free Sets of Graphs.** We combine the previously obtained results in a characterization of the intersection between the classes of context-free and definable graph languages. We state these characterization results in two versions. In the first version, we explicitly keep track of the set of sources $\tau \subseteq_{fin} \mathbb{S}$ witnessing that a set of graphs is context-free:

Theorem 6.9. *For every set $\mathcal{L} \subseteq \mathcal{G}$ of graphs and sort $\tau \subseteq_{fin} \mathbb{S}$, the following are equivalent:*

(1) *$\mathcal{L}$ is definable and context-free, for a grammar over $\mathcal{F}^{\tau}_{\mathsf{HR}}$-operations,*

(2) *$\mathcal{L}$ is recognizable in* $\mathbf{G}$ *and* $\mathcal{L} \subseteq \mathcal{G}^{\tau}_{\mathsf{gen}}$,
(3) *$\mathcal{L}$ is recognizable in* $\mathbf{G}^{\tau}$ *and* $\mathcal{L} \subseteq \mathcal{G}^{\tau}_{\mathsf{gen}}$,
(4) *$\mathcal{L}$ is τ-parsable,*

Proof. (1) $\Rightarrow$ (2) By Theorem 3.5 every definable set of graphs is recognizable in the algebra $\mathbf{G}$. Let Γ be a grammar such that $\mathcal{L} = \mathcal{L}_X(\Gamma)$, for some nonterminal X of Γ. By assumption, Γ uses only $\mathcal{F}^{\tau}_{\mathsf{HR}}$-operations, hence $\mathcal{L} \subseteq \mathcal{G}^{\tau}$. Moreover, since $\mathcal{L}_X(\Gamma)$ is the least solution of Γ, every graph $G \in \mathcal{L}_X(\Gamma)$ is term-generated, thus $\mathcal{L} \subseteq \mathcal{G}^{\tau}_{\mathsf{gen}}$.

(2) $\Rightarrow$ (3) By Lemma 2.8, since $\mathbf{G}^{\tau}$ is a subalgebra of $\mathbf{G}$.

(3) $\Rightarrow$ (4) By Theorem 6.8, the set of graphs $\mathcal{G}^{\tau}_{\mathsf{gen}}$ is τ-parsable, and by Lemma 6.2, the restriction of a τ-parsable set to a recognizable set in $\mathbf{G}^{\tau}$ is τ-parsable.

(4) $\Rightarrow$ (1) By Proposition 6.3. $\square$

The second theorem is more coarse, in that we quantify out existentially the sort and the bound on the tree-width, in each item:

Theorem 6.10. *For every set $\mathcal{L} \subseteq \mathcal{G}^{\emptyset}$ of graphs with no sources, the following are equivalent:*
(1) *$\mathcal{L}$ is definable and context-free,*
(2) *$\mathcal{L}$ is recognizable and has bounded tree-width,*
(3) *$\mathcal{L}$ is parsable,*
(4) *There exists a finite set $\mathbb{B}$ of edge labels, a definable $(\mathbb{R}_{\mathsf{tree}}(\mathbb{B}),\ \mathbb{R}^{\emptyset}_{\mathsf{graph}})$-transduction F and a definable $(\mathbb{R}^{\emptyset}_{\mathsf{graph}},\ \mathbb{R}_{\mathsf{tree}}(\mathbb{B}))$-transduction H, such that (a)* $\mathrm{dom}(F \circ H) = \|\mathcal{L}\|$, *and (b) $F \circ H$ is the identity on $\|\mathcal{L}\|$.*

Proof. (1) $\Rightarrow$ (2) As every grammar uses only $\mathcal{F}^{\tau}_{\mathsf{HR}}$-operations for the set of sources $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ that appear in the grammar, we obtain that $\mathcal{L}$ is recognizable in $\mathbf{G}$ and $\mathcal{L} \subseteq \mathcal{G}^{\tau}_{\mathsf{gen}}$ from Theorem 6.9 ($1 \Rightarrow 2$). Then, $\mathrm{twd}(G) \leq \mathrm{card}(\tau) - 1$, for every graph $G \in \mathcal{G}^{\tau}_{\mathsf{gen}}$, by Lemma 4.4.

(2) $\Rightarrow$ (3) Let $k \geq 1$ be the least integer such that $\mathrm{twd}(G) \leq k$, for all graphs $G \in \mathcal{L}$, and let G be such a graph. Since $\mathcal{L} \subseteq \mathcal{G}^{\emptyset}$, the graph G does not have sources. Hence, condition (1) of Corollary 6.7 is satisfied, for any sort $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ with $k \leq \mathrm{card}(\tau) - 1$, hence $\|G\| \in \mathrm{dom}(\mathcal{K})$, where $\mathcal{K}$ is the transduction whose existence is stated by Corollary 6.7. Then, there exists a tree $T \in \mathcal{T}(\mathbb{B}^{\tau}_{\mathsf{parse}})$ such that $(\|G\|, \|T\|) \in \mathcal{K}$ such that $\mathbf{val}(T) = G$, by condition (2) of Corollary 6.7. Hence, $\mathcal{L} \subseteq \mathcal{G}^{\tau}_{\mathsf{gen}}$ and, by Theorem 6.9, $\mathcal{L}$ is τ-parsable.

(3) $\Rightarrow$ (4) We have that $\mathcal{L}$ is τ-parsable for some sort $\tau \subseteq_{\mathit{fin}} \mathbb{S}$. Let the alphabet be $\mathbb{B} \stackrel{\mathsf{def}}{=} \mathbb{B}^{\tau}_{\mathsf{parse}}$. Then, there exists a definable $(\mathbb{R}^{\tau}_{\mathsf{graph}},\ \mathbb{R}_{\mathsf{tree}}(\mathbb{B}))$-transduction H that witnesses the τ-parsability of $\mathcal{L}$. Since $\mathcal{L} \subseteq \mathcal{G}^{\emptyset}$, we can assume w.l.o.g. that H is a $(\mathbb{R}^{\emptyset}_{\mathsf{graph}},\ \mathbb{R}_{\mathsf{tree}}(\mathbb{B}))$-transduction (using a transduction that simply removes the predicates that encode the sources). Moreover, $F \stackrel{\mathsf{def}}{=} \mathbf{val}\downharpoonright_{\mathcal{T}(\mathbb{B})}$ is definable, by Lemma 5.8 and $F \circ H$ is the identity on $\|\mathcal{L}\|$, as required.

(4) $\Rightarrow$ (1) By Proposition 2.3 (1), $F \circ H$ is a definable transduction, hence $\mathrm{dom}(F \circ H) = \|\mathcal{L}\|$ is definable, by Proposition 2.3 (3). Then, $\mathcal{K} \stackrel{\mathsf{def}}{=} F^{-1}(\mathcal{L})$ is definable, by Theorem 2.2. By Theorem 4.1, $\mathcal{K}$ is recognizable in $\mathbf{T}(\mathbb{B})$ and $\mathcal{L}$ is context-free, by Corollary 5.10. $\square$

We note that item (4) is missing from Theorem 6.9 because there is no easy way of computing an upper bound on the tree-width of $\mathcal{L}$. That is, we identify the following problem for future work: given MSO-definable transductions F and H as stated in item 4 of Theorem 6.10, compute a bound on the tree-width of $\mathcal{L}$ based on F and H. We remark that the construction of [CE95] can be used to derive an upper bound, but that this bound is likely not the optimal one. On the other hand, item (3) of Theorem 6.9 could be added to Theorem 6.10 (we omit it for conciseness reasons).

We further note that the problem of whether one of the conditions from Theorem 6.9 (resp. Theorem 6.10) holds for the set of graphs generated by a given graph grammar is undecidable. In fact, even the problem of whether a given context-free word grammar defines a recognizable (and hence definable) word language is undecidable, according to a result by Greibach [Gre68].

The impossibility of having an algorithm that decides whether a given grammar produces a definable set of graphs motivated a number of definitions that provide sufficient conditions. Among them proposals for regular graph grammars [Cou91, CIZ24, BIZ25] over and regular expressions for graphs of tree-width at most 2 [Dou22] (for more details see the related work section in the beginning of this paper). We highlight that our characterization provides new tools for the definition of such sets. For instance, the definable transductions from Example 2.1 fit point (4) of Theorem 6.10, meaning that adding edges between the lexicographical successors of the leaves in a binary tree taken from a recognizable set produces a context-free and definable set of graphs (i.e., trees with linked leaves).

7. Finite versus Locally Finite Recognizability of Graph Sets

Theorem 6.9 (2-3) proves the equivalence between the recognizability via locally finite algebras and recognizability via finite algebras, for tree-width bounded sets of graphs. This is the case because a set $\mathcal{L} \subseteq \mathcal{G}^{\tau}_{\mathsf{gen}}$ of graphs is recognizable in $\mathbf{G}$ iff it is the inverse image of a homomorphism into a locally finite recognizer algebra having infinitely many (possibly non-empty) sorts (Definition 2.6), whereas recognizability of $\mathcal{L}$ in $\mathbf{G}^{\tau}$ means that the locally finite recognizer algebra has finitely many non-empty sorts, being thus finite. Consequently, bounded tree-width sets of graphs can be recognized using finite algebras, just like terms[5] [Don70]. This has been initially proved by Courcelle and Lagergren [CL96], using a different argument.

In this section, we prove that locally finite recognizability for graphs is the limit of recognizability in an infinite increasing sequence of finite underapproximations (Theorem 7.4). This means that the equivalence between locally finite and finite recognizability for bounded tree-width sets of graphs (points (2) and (3) of Theorem 6.9) is actually a cut-off in this infinite increasing sequence.

The following lemma is an equivalent characterization of the syntactic congruence (Definition 2.5), that uses only terms of a restricted form:

Lemma 7.1. *Let $\mathcal{L} \subseteq \mathcal{G}$ be a set of graphs. Then, $G_1 \cong^{\mathbf{G}}_{\mathcal{L}} G_2$ iff* $\mathrm{sort}(G_1) = \mathrm{sort}(G_2)$ *and* $\mathsf{rename}^{\mathbf{G}}_{\alpha} \circ \mathsf{restrict}^{\mathbf{G}}_{\tau}(G_1 \parallel G) \in \mathcal{L} \Leftrightarrow \mathsf{rename}^{\mathbf{G}}_{\alpha} \circ \mathsf{restrict}^{\mathbf{G}}_{\tau}(G_2 \parallel G) \in \mathcal{L}$, *for all graphs G, G_1, G_2, finite permutations $\alpha : \mathbb{S} \rightarrow \mathbb{S}$ and sorts $\tau \subseteq_{fin} \mathbb{S}$.*

[5] Recognizability by finite automata, as in the case of words and terms, is one step further, because automata typically require a canonical order in which the input is traversed.

Proof. We define the equivalence relation $\cong$ by setting $G_1 \cong G_2$ iff $\mathrm{sort}(G_1) = \mathrm{sort}(G_2)$ and $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(G_1 \parallel G) \in \mathcal{L} \Leftrightarrow \mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(G_2 \parallel G) \in \mathcal{L}$, for all graphs G, G_1, G_2, finite permuations $\alpha : \mathbb{S} \to \mathbb{S}$ and sets of sources $\tau \subseteq_{\mathit{fin}} \mathbb{S}$. We now consider some congruence $\equiv$ that saturates $\mathcal{L}$. Then, $G_1 \equiv G_2$ implies that $G_1 \cong G_2$, i.e., $\equiv \subseteq \cong$ (*). This is because, if $G_1 \equiv G_2$ then $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(G_1 \parallel G) \equiv \mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(G_2 \parallel G)$ (as $\equiv$ is some congruence), and hence $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(G_1 \parallel G) \in \mathcal{L} \Leftrightarrow \mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(G_2 \parallel G) \in \mathcal{L}$ (as $\equiv$ saturates $\mathcal{L}$).

We will establish below that $\cong$ saturates $\mathcal{L}$. By definition of $\cong$, we have that $G_1 \cong G_2$ implies that $G_1 \in \mathcal{L} \iff G_2 \in \mathcal{L}$, since we can choose α as the identity, $\tau = \mathrm{sort}(G_1)$ and $G = \mathbf{0}_{\mathrm{sort}(G_1)}$. By (*) we obtain that $\cong$ is the coarsest relation that saturates $\mathcal{L}$. It remains to establish that $\cong$ is a congruence. Let us consider some graphs $G_1 \cong G_2$. We show the closure under the operations of the graph algebra, by a case distinction:

- ▷ $G_1 \parallel^{\mathbf{G}} G \cong G_2 \parallel^{\mathbf{G}} G$ and $G \parallel^{\mathbf{G}} G_1 \cong G \parallel^{\mathbf{G}} G_2$, for all graphs G: By the commutativity of $\parallel^{\mathbf{G}}$, it is sufficient to show one of the implications. Let us assume $G_1 \cong G_2$ and let G be some graph. To establish that $G_1 \parallel^{\mathbf{G}} G \cong G_2 \parallel^{\mathbf{G}} G$, consider some graph G', bijective function α and $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ such that $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau((G_1 \parallel^{\mathbf{G}} G) \parallel^{\mathbf{G}} G') \in \mathcal{L}$. Then, $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau((G_2 \parallel^{\mathbf{G}} G) \parallel^{\mathbf{G}} G') \in \mathcal{L}$ follows from $G_1 \cong G_2$, by the definition of $\cong$, using the associativity of $\parallel^{\mathbf{G}}$ and chosing the graph $G'' = G \parallel G'$.
- ▷ $\mathsf{restrict}_{\tau^\circ}(G_1) \cong \mathsf{restrict}_{\tau^\circ}(G_2)$, for all $\tau^\circ \subseteq_{\mathit{fin}} \mathbb{S}$: We consider some graph G, finite permutation α and $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ such that $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(\mathsf{restrict}_{\tau^\circ}(G_1) \parallel^{\mathbf{G}} G) \in \mathcal{L}$. We need to show that $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(\mathsf{restrict}_{\tau^\circ}(G_2) \parallel^{\mathbf{G}} G) \in \mathcal{L}$. We now verify that we can choose G', α' and τ' such that $\mathsf{rename}_{\alpha'} \circ \mathsf{restrict}_{\tau'}(G_i \parallel^{\mathbf{G}} G') = \mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(\mathsf{restrict}_{\tau^\circ}(G_i) \parallel^{\mathbf{G}} G)$ for $i = 1, 2$. Indeed, we can choose $G' = \mathsf{rename}_\beta(G)$, for some permutation β that renames the sources $\mathrm{sort}(G) \setminus \tau^\circ$ to some fresh sources τ'', choosing $\tau' = \tau \cup \beta(\tau)$ and setting α' as the permutation that does all renamings of α and β^{-1}. The claim then follows from $G_1 \cong G_2$.
- ▷ $\mathsf{rename}_\beta(G_1) \cong \mathsf{rename}_\beta(G_2)$, for all finite permutations β: We consider a graph G, finite permutation α and $\tau \subseteq_{\mathit{fin}} \mathbb{S}$, such that $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(\mathsf{rename}_\beta(G_1) \parallel^{\mathbf{G}} G) \in \mathcal{L}$. We need to show that $\mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(\mathsf{rename}_\beta(G_2) \parallel^{\mathbf{G}} G) \in \mathcal{L}$. We now observe that $\mathsf{rename}_{\alpha\circ\beta} \circ \mathsf{restrict}_{\beta^{-1}(\tau)}(G_i \parallel^{\mathbf{G}} \mathsf{rename}_{\beta^{-1}}(G)) = \mathsf{rename}^{\mathbf{G}}_\alpha \circ \mathsf{restrict}^{\mathbf{G}}_\tau(\mathsf{rename}_\beta(G_i) \parallel^{\mathbf{G}} G)$ for $i = 1, 2$. The claim follows from $G_1 \cong G_2$. □

Specializing Lemma 7.1 to graphs of empty sort, we obtain the following result that has appeared in previous work:

Corollary 7.2 [CE12, Theorem 4.34]**.** *Let $\mathcal{L} \subseteq \mathcal{G}^\emptyset$ be a set of graphs. Then, $G_1 \cong^{\mathbf{G}}_{\mathcal{L}} G_2$ iff $\mathrm{sort}(G_1) = \mathrm{sort}(G_2)$ and $\mathsf{restrict}^{\mathbf{G}}_\tau(G_1 \parallel G) \in \mathcal{L} \Leftrightarrow \mathsf{restrict}^{\mathbf{G}}_\tau(G_2 \parallel G) \in \mathcal{L}$, for all graphs G, G_1, G_2 and sorts $\tau \subseteq_{\mathit{fin}} \mathbb{S}$.*

Proof. Follows from Lemma 7.1 for some $\mathcal{L} \subseteq \mathcal{G}^\emptyset$ by the observation that for any graph G we have $G \in \mathcal{L}$ iff $\mathsf{rename}^{\mathbf{G}}_\alpha(G) \in \mathcal{L}$ for any finite permutation α. □

The next step is proving that the syntactic congruences of a language of graphs of empty sort agree over the algebras $\mathbf{G}$ and $\mathbf{G}^\tau$, for any sort $\tau \subseteq_{\mathit{fin}} \mathbb{S}$:

Lemma 7.3. *Let $\mathcal{L} \subseteq \mathcal{G}^\emptyset$ be a language, $\tau \subseteq_{\mathit{fin}} \mathbb{S}$ be a sort, and $G_1, \mathcal{G}_2 \in \mathcal{G}^\tau$ be some graphs. Then, $G_1 \cong^{\mathbf{G}}_{\mathcal{L}} G_2$ iff $G_1 \cong^{\mathbf{G}^\tau}_{\mathcal{L}} G_2$.*

Proof. "⇒" Because $\cong_{\mathcal{L}}^{\mathbf{G}} \cap (\mathcal{G}^\tau \times \mathcal{G}^\tau)$ is a congruence that saturates $\mathcal{L}$ w.r.t the algebra $\mathbf{G}^\tau$ and $\cong_{\mathcal{L}}^{\mathbf{G}^\tau}$ is the greatest such congruence. "⇐" By Corollary 7.1, we need to show that $\mathsf{restrict}_{\tau'}^{\mathbf{G}}(G_1 \parallel^{\mathbf{G}} G) \in \mathcal{L}$ implies $\mathsf{restrict}_{\tau'}^{\mathbf{G}}(G_2 \parallel^{\mathbf{G}} G) \in \mathcal{L}$. Because of $G_1, G_2 \in \mathcal{G}^\tau$, it suffices to prove that $\mathsf{restrict}_{\tau'\cap\tau}^{\mathbf{G}^\tau}(G_1 \parallel^{\mathbf{G}} \mathsf{restrict}_{\tau'\setminus\tau}(G))^{\mathbf{G}^\tau} \in \mathcal{L}$ implies $\mathsf{restrict}_{\tau'\cap\tau}^{\mathbf{G}^\tau}(G_2 \parallel^{\mathbf{G}} \mathsf{restrict}_{\tau'\setminus\tau}^{\mathbf{G}^\tau}(G)) \in \mathcal{L}$. However, this follows from $G_1 \cong_{\mathcal{L}}^{\mathbf{G}^\tau} G_2$. □

Finally, we relate recognizability of a set of graphs of empty sort in the graph algebra $\mathbf{G}$ and any of its subalgebras $\mathbf{G}^\tau$:

Theorem 7.4. *Let $\mathcal{L}$ be a set of graphs with no sources. Then, $\mathcal{L}$ is recognizable in the graph algebra $\mathbf{G}$ iff for each sort $\tau \subseteq_{fin} \mathbb{S}$, the set $\mathcal{L}$ is recognizable in the algebra $\mathbf{G}^\tau$.*

Proof. By Lemma 7.3, we have that $\cong_{\mathcal{L}}^{\mathbf{G}} \cap (\mathcal{G}^\tau \times \mathcal{G}^\tau) = \cong_{\mathcal{L}}^{\mathbf{G}^\tau}$ for every $\tau \subseteq_{fin} \mathbb{S}$. In particular, $\cong_{\mathcal{L}}^{\mathbf{G}}$ is locally finite iff $\cong_{\mathcal{L}}^{\mathbf{G}^\tau}$ is locally finite for every $\tau \subseteq_{fin} \mathbb{S}$. □

Note that the condition of Theorem 7.4 is different from the *weak recognizability* notion introduced by Bojańczyk [Boj23], that considers an infinite sequence of finitely-generated *term-generated* algebras. Weak recognizability and recognizability are in fact not the same, as [Boj23, Example 11] shows. This points to the crucial difference between term-generated and non term-generated algebras (Lemma 4.4).

8. Conclusions

We have given a characterization of definable context-free sets of graphs, by showing their equivalence with bounded tree-width and recognizable sets (where recognizability is understood either in locally finite or finite algebras), parsable sets (where the parse trees can be recovered from the graph by a definable transduction) and images of recognizable unranked sets of trees under definable transductions whose inverses are definable as well. We finalize our study with a discussion on recognizability and a proof that locally finite recognizer algebras are limits of infinite sequences of finite recognizer algebras.

References

[BIZ25] Marius Bozga, Radu Iosif, and Florian Zuleger. Regular grammars for sets of graphs of tree-width 2. In *40th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2025, Singapore, June 23-26, 2025*, pages 704–717. IEEE, 2025. doi:10.1109/LICS65433.2025.00059.

[Bod98] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

[Boj23] Mikołaj Bojańczyk. The category of MSO transductions. *CoRR*, abs/2305.18039, 2023. arXiv:2305.18039, doi:10.48550/arXiv.2305.18039.

[BP16] Mikołaj Bojańczyk and Michał Pilipczuk. Definability equals recognizability for graphs of bounded treewidth. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '16, page 407–416, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2933575.2934508.

[BP22] Mikołaj Bojańczyk and Michał Pilipczuk. Optimizing tree decompositions in MSO. *Log. Methods Comput. Sci.*, 18(1), 2022. doi:10.46298/lmcs-18(1:26)2022.

[Büc90] Richard Büchi. *Weak Second-Order Arithmetic and Finite Automata*, pages 398–424. Springer New York, New York, NY, 1990. doi:10.1007/978-1-4613-8928-6_22.

[CDG+08] Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. *Tree Automata Techniques and Applications*. 2008. URL: https://inria.hal.science/hal-03367725.

[CE95] Bruno Courcelle and Joost Engelfriet. A logical characterization of the sets of hypergraphs defined by hyperedge replacement grammars. *Mathematical Systems Theory*, 28:515–552, 11 1995. doi:10.1007/BF01204169.

[CE12] Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2012. doi:10.1017/CBO9780511977619.

[CIZ24] Mark Chimes, Radu Iosif, and Florian Zuleger. Tree-verifiable graph grammars. In Nikolaj Björner, Marijn Heule, and Andrei Voronkov, editors, *Proceedings of 25th Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 100 of *EPiC Series in Computing*, pages 165–180. EasyChair, 2024. doi:10.29007/8113.

[CL96] Bruno Courcelle and Jens Lagergren. Equivalent definitions of recognizability for sets of graphs of bounded tree-width. *Mathematical Structures in Computer Science*, 6(2):141–165, 1996. doi:10.1017/S096012950000092X.

[Cou90] Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

[Cou91] Bruno Courcelle. The monadic second-order logic of graphs v: on closing the gap between definability and recognizability. *Theoretical Computer Science*, 80(2):153–202, 1991. doi:10.1016/0304-3975(91)90387-H.

[Don70] John Doner. Tree acceptors and some of their applications. *J. Comput. Syst. Sci.*, 4(5):406–451, oct 1970. doi:10.1016/S0022-0000(70)80041-1.

[Dou22] Amina Doumane. Regular expressions for tree-width 2 graphs. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 121:1–121:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.ICALP.2022.121.

[EF95] Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite model theory*. Perspectives in Mathematical Logic. Springer, 1995.

[Eng15] Joost Engelfriet. Tree automata and tree grammars. *CoRR*, abs/1510.02036, 2015. URL: http://arxiv.org/abs/1510.02036, arXiv:1510.02036.

[EW67] Mezei J. E. and Jesse B. Wright. Algebraic automata and context-free sets. *Information and Control*, 11(1):3–29, 1967. doi:10.1016/S0019-9958(67)90353-1.

[Gre68] Sheila Greibach. A note on undecidable properties of formal languages. *Math. Systems Theory*, 2:1–6, 1968. doi:10.1007/BF01691341.

[IZ23] Radu Iosif and Florian Zuleger. Expressiveness results for an inductive logic of separated relations. In Guillermo A. Pérez and Jean-François Raskin, editors, *34th International Conference on Concurrency Theory, CONCUR 2023, September 18-23, 2023, Antwerp, Belgium*, volume 279 of *LIPIcs*, pages 20:1–20:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.CONCUR.2023.20.

[TW68] James W. Thatcher and Jesse B. Wright. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical systems theory*, 2(1):57–81, 1968. doi:10.1007/BF01691346.