

Regular matching problems for infinite trees

Carlos Camino 

FMI, Universität Stuttgart, Germany: cfcamino@gmail.com

Volker Diekert 


FMI, Universität Stuttgart, Germany: diekert@fmi.uni-stuttgart.de

Besik Dundua 

Kutaisi International University and VIAM, Tbilisi State University: bdundua@gmail.com

Mircea Marin 

FMI, West University of Timișoara, Romania: mircea.marin@e-uvv.ro

Géraud Sénizergues 

LaBRI, Université de Bordeaux, France: geraud.senizergues@u-bordeaux.fr

Abstract

We study the matching problem of regular tree languages, that is, “ $\exists \sigma : \sigma(L) \subseteq R$?” where L, R are regular tree languages over the union of finite ranked alphabets Σ and \mathcal{X} where \mathcal{X} is an alphabet of variables and σ is a substitution such that $\sigma(x)$ is a set of trees in $T(\Sigma \cup H) \setminus H$ for all $x \in \mathcal{X}$. Here, H denotes a set of “holes” which are used to define a “sorted” concatenation of trees. Conway studied this problem in the special case for languages of finite words in his classical textbook *Regular algebra and finite machines* published in 1971. He showed that if L and R are regular, then the problem “ $\exists \sigma \forall x \in \mathcal{X} : \sigma(x) \neq \emptyset \wedge \sigma(L) \subseteq R$?” is decidable. Moreover, there are only finitely many maximal solutions, the maximal solutions are regular substitutions, and they are effectively computable. We extend Conway’s results when L, R are regular languages of finite and infinite trees, and language substitution is applied inside-out, in the sense of Engelfriet and Schmidt (1977/78). More precisely, we show that if $L \subseteq T(\Sigma \cup \mathcal{X})$ and $R \subseteq T(\Sigma)$ are regular tree languages over finite or infinite trees, then the problem “ $\exists \sigma \forall x \in \mathcal{X} : \sigma(x) \neq \emptyset \wedge \sigma_{\text{io}}(L) \subseteq R$?” is decidable. Here, the subscript “io” in $\sigma_{\text{io}}(L)$ refers to “inside-out”. Moreover, there are only finitely many maximal solutions σ , the maximal solutions are regular substitutions and effectively computable. **The corresponding question for the outside-in extension σ_{oi} remains open, even in the restricted setting of finite trees.**

In order to establish our results we use alternating tree automata with a parity condition and games.

2012 ACM Subject Classification Theory of computation \rightarrow Formal languages and automata theory

Keywords and phrases Regular tree languages, infinite trees, Factorization theory, IO and OI

Preamble

The additional material in the appendix (Sec. 8) is not needed to understand the results in the main body of the paper.

1 Introduction

1.1 Historical background

Regular matching problems using generalized sequential machines were studied first by Ginsburg and Hibbard. Their publication [13], dating back to 1964, showed that it is decidable whether there is a generalized sequential machine which maps L onto R if L and R are regular languages of finite words. The paper also treats several variants of this problem. For example, the authors notice that the decidability cannot be lifted to context-free languages. Another paper in that area is by Prieur et al. [24]. It appeared in 1997 and studies the problem whether there exists a sequential bijection from a finitely generated free monoid to a given rational set R . Earlier, in the 1960’s Conway studied regular matching problems in the following variant of [13]: Let \mathcal{X}, Σ be finite alphabets and $L \subseteq (\Sigma \cup \mathcal{X})^*$,

$R \subseteq \Sigma^*$. A substitution $\sigma : \mathcal{X} \rightarrow 2^{\Sigma^*}$ is extended to $\sigma : \Sigma \cup \mathcal{X} \rightarrow 2^{\Sigma^*}$ by $\sigma(a) = \{a\}$ for all $a \in \Sigma \setminus \mathcal{X}$. It is called a *solution* of the problem “ $L \subseteq R$?” if $\sigma(L) \subseteq R$. In his textbook [6, Chapt. 6], Conway developed a *factorization theory* of formal languages. Thereby he found a nugget in formal language theory: Given as input regular word languages $L \subseteq (\Sigma \cup \mathcal{X})^*$ and $R \subseteq \Sigma^*$, it holds:

1. It is decidable whether there is a substitution $\sigma : \mathcal{X} \rightarrow 2^{\Sigma^*}$ such that $\sigma(L) \subseteq R$ and $\emptyset \neq \sigma(x)$ for all $x \in \mathcal{X}$.
2. Define $\sigma \leq \sigma'$ by $\sigma(x) \subseteq \sigma'(x)$ for all $x \in \mathcal{X}$. Then every solution is bounded from above by a maximal solution; and the number of maximal solutions is finite.
3. If σ is maximal, then $\sigma(x)$ is regular for all $x \in \mathcal{X}$; and all maximal solutions are effectively computable.

The original proof is rather technical and not easy to digest. On the other hand, using the algebraic concept of *recognizing morphisms*, elegant and simple proofs exist. Regarding the complexity, it turns out that the problem “ $\exists \sigma : \sigma(L) \subseteq R$?” is PSPACE-complete if L and R are given by NFAs by [18, Lemma 3.2.3]. The apparently similar problem “ $\exists \sigma : \sigma(L) = R$?” is more difficult: Bala showed that it is EXPSpace-complete [1].

Conway also asked whether the unique maximal solution of the language equation $Lx = xL$ is given by a substitution such that $\sigma(x)$ is regular¹. This question was answered by Kunc in a highly unexpected way: there is a finite set L such that the unique maximal solution $\sigma(x)$ of $Lx = xL$ is co-recursively-enumerable-complete [19]. A recent survey on language equations is in [20].

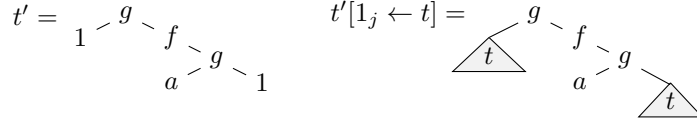
1.2 Conway’s result for trees

The present paper generalizes Conway’s result to regular tree languages. We consider finite and infinite trees simultaneously over a finite ranked alphabet Δ . We let $T(\Delta)$ be the set of all trees with labels in Δ , and by $T_{\text{fin}}(\Delta)$ we denote its subset of finite trees. More specifically, we consider finite ranked alphabets \mathcal{X} of *variables* and Σ of *function symbols*. In order to define a notion of a concatenation we also need a set of *holes* H . These are symbols of rank zero. For simplicity, throughout the set of holes is chosen as $H = \{1, \dots, |H|\}$. We require $(\Sigma \cup \mathcal{X}) \cap H = \emptyset$. Trees in $T(\Delta)$ are rooted and they can be written as terms $x(s_1, \dots, s_r)$ where $r = \text{rk}(x) \geq 0$ and s_i are trees. In particular, all symbols of rank 0 are trees. Words $a_1 \dots a_n \in \Sigma^*$ (with $a_i \in \Sigma$) are encoded as terms $a_1(\dots(a_n(\$))\dots)$ where the a_i are function symbols of rank 1 and the only symbol of rank 0 is $\$$ which signifies “*end-of-string*”. An infinite word $a_1 a_2 \dots \in \Sigma^\omega$ is encoded as $a_1(a_2(\dots))$ and no hole appears. In contrast to the case of classical term rewriting, substitutions are applied at inner positions, too. In the word case it is clear what to do. For example, let $w = xyx \in \mathcal{X}^*$ with $\sigma(x) = L_x$ and $\sigma(y) = L_y$, then we obtain $\sigma(w) = L_x L_y L_x$. Translated to term notation, we obtain $w = x(y(x(\$)))$, $\sigma(z) = \{u(1) \mid u \in L_z\}$ for $z \in \{x, y\}$ with the result $\sigma(w) = L_x L_y L_x(\$)$. On the other hand, in the tree case variables of any rank may exist. As a result, variables may appear at inner nodes as well as at leaves. Throughout, if t is any tree, then $\text{leaf}_i(t)$ denotes the set of leaves labeled by the hole $i \in H$, and by i_j we denote elements of $\text{leaf}_i(t)$.

In the following, a *substitution* means a mapping $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ such that for all $x \in \mathcal{X}$ we have $\sigma(x) \subseteq T(\Sigma \cup \{1, \dots, \text{rk}(x)\})$. A *homomorphism* (resp. *partial homomorphism*) is a substitution σ such that $|\sigma(x)| = 1$ (resp. $|\sigma(x)| \leq 1$) for all $x \in \mathcal{X}$. We write $\sigma_1 \leq \sigma_2$ if $\sigma_1(x) \subseteq \sigma_2(x)$ for all $x \in \mathcal{X}$; and we say that σ is *regular*² if $\sigma(x)$ is regular for all $x \in \mathcal{X}$.

¹ There is a unique maximal solution since the union over all solutions is a solution.

² There are several equivalent definitions for regular tree languages, e.g. see [5, 25, 22, 23, 27].



■ **Figure 1** The left tree t' has two positions labeled with hole $1 \in H$. Holes define a composition of trees. The right tree is obtained by composing t' with a tree t over the hole 1 which is denoted by $t'[1_j \leftarrow t]$.

Trees are represented graphically, too. For example, $g(1, f(g(a, 1)))$ and $g(t, f(g(a, t)))$ are represented in Fig. 1. We obtain $g(t, f(g(a, t)))$ by replacing the positions labeled by hole 1 in $g(1, f(g(a, 1)))$ by any rooted tree t . As soon as σ is not a partial homomorphism, one has to distinguish between “Inside-Out” (IO for short) and “Outside-In” (OI for short) as advocated and defined in [11, 12]. Our positive decidability results concern IO, only. The IO-definition $\sigma_{\text{io}}(s)$ for a given tree $s \in T(\Sigma \cup \mathcal{X})$ and a substitution σ has the following interpretation. First, we extend σ to a mapping from $\Sigma \cup \mathcal{X} \cup H$ to $2^{T(\Sigma \cup H)}$ by $\sigma(f) = \{f(1, \dots, \text{rk}(f))\}$ for $f \in \Sigma \setminus \mathcal{X}$. Second, we use a term notation $s = x(s_1, \dots, s_r)$ (for finite and infinite trees s) and we let $\sigma_{\text{io}}(s) \subseteq T(\Sigma)$ to be a certain fixed point of the language equation

$$\sigma_{\text{io}}(s) = \bigcup \{t[i_j \leftarrow t_i] \mid t \in \sigma(x) \wedge \forall 1 \leq i \leq r : t_i \in \sigma_{\text{io}}(s_i)\}. \quad (1)$$

The notation $t[i_j \leftarrow t_i]$ in Eq.(1) means that for all $i \in H$ and $i_j \in \text{leaf}_i(t)$ all leaves i_j are replaced by t_i where t_i depends on i , only. It is a special case of the notation $t[i_j \leftarrow t_{i_j}]$ which says that for all $i \in H$ and $i_j \in \text{leaf}_i(t)$ some tree t_{i_j} is chosen and then each leaf i_j is replaced by t_{i_j} . This more general notation appears below when defining “Outside-In” substitutions. The idea to compute the elements in $\sigma_{\text{io}}(s)$ is a recursive procedure which at each call first selects the tree $t \in \sigma(x)$ (if x is the label of the root of s) and then it makes recursive calls for each hole i which appears as a label in t to compute the elements t_i . After that, all positions i_j in t which have the label i are replaced by the same tree t_i . For finite trees the procedure terminates. For example, $g(t, f(g(a, t))) = g(1, f(g(a, 1)))[1_j \leftarrow t]$ in Fig. 1 represents an instance of an IO-substitution. For that, we let $s = g(x, f(g(a, x)))$ where x is a variable of rank zero and $\sigma(x) = \{t_1, t_2\}$. Then $\sigma_{\text{io}}(s) = \{g(t_1, f(g(a, t_1))), g(t_2, f(g(a, t_2)))\}$. For infinite trees, the recursion is not guaranteed to stop. Running it for n recursive calls defines a Cauchy sequence in an appropriate complete metric space $T_{\perp}(\Sigma \cup \mathcal{X} \cup H)$ where \perp plays the role of undefined if the procedure cannot select a tree because the corresponding set $\sigma(x)$ happens to be empty.

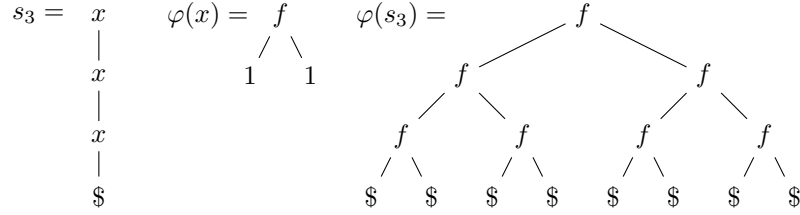
As explained above, the feature of IO is that every leaf i_j in $t \in \sigma(x)$ labeled with a hole i is substituted with the same tree $t_i \in \sigma_{\text{io}}(s_i)$. If we remove this restriction (that is: in Eq.(1) we replace $t[i_j \leftarrow t_i]$ by $t[i_j \leftarrow t_{i_j}]$), then we obtain the OI-substitution $\sigma_{\text{oi}}(s)$ which can be much larger than $\sigma_{\text{io}}(s)$. For example, for $s = g(x, f(g(a, x)))$ and $\sigma(x) = \{t_1, t_2\}$ we obtain

$$\sigma_{\text{oi}}(s) = \{g(t_i, f(g(a, t_j))) \mid i, j \in \{1, 2\}\}.$$

Clearly, $\sigma_{\text{io}}(s) = \sigma_{\text{oi}}(s)$ if σ is a partial homomorphism. Thus, for a partial homomorphism φ we may write $\varphi(s)$ without risking ambiguity. Another situation for $\sigma_{\text{io}}(s) = \sigma_{\text{oi}}(s)$ is when duplications of holes do not appear. *Duplication* means that there are some x and $t \in \sigma(x)$ where a hole $i \in H$ appears at least twice in t . Duplications cannot appear in the traditional framework of words, but “duplication” is a natural concept for trees.

Allowing duplications complicates the situation because it might happen that $\sigma_{\text{io}}(L)$ is not regular, although L and σ are regular. Actually, this may happen even if σ is defined by a homomorphism $\varphi : \mathcal{X} \rightarrow T_{\text{fin}}(\Sigma \cup H)$. Recall that the notation $T_{\text{fin}}(\Sigma \cup H)$ refers to the subset of finite trees in $T(\Sigma \cup H)$. Examples of a homomorphism φ where $\varphi(L)$ is not regular

are easy to construct. The classical example is $L = \{x^n(\$) \mid n \in \mathbb{N}\}$ and $\varphi(x) = f(1, 1)$. Then $\varphi(L)$ is not regular. The corresponding trees of height 3 are depicted in Fig. 2. This led to



■ **Figure 2** $L = x^*(\$)$ is regular, but $\varphi(L)$ is not.

the *HOM-problem*. The inputs are a homomorphism φ and a regular tree language L . The question is whether $\varphi(L)$ is regular. The problem is decidable in the setting of finite trees by [14]. It is DEXPTIME-complete by [9].

Most of our work deals with regular tree languages. However, once the results are established for regular sets, we can easily push the results further to some border of decidability. We consider a class \mathcal{C} of tree languages such that on input $L \in \mathcal{C}$ and a regular tree language K , the emptiness problem $L \cap K$ is decidable. For example, in the word case, the class \mathcal{C} can be defined by the class of context-free languages, and then Conway's result for finite words still holds if L is context-free and R is regular.

1.3 Statement of the main results

Our main results can be found in Sec. 6. We are ready to formulate them now. Thm. 29 and can be rephrased as follows. First, the following decision problem is decidable

- Input: Regular substitutions σ_1, σ_2 and tree languages $L \subseteq T(\Sigma \cup \mathcal{X})$, $R \subseteq T(\Sigma)$ such that R is regular and $L \in \mathcal{C}$. Here \mathcal{C} is as defined above such that emptiness of the intersection with regular sets is decidable.
- Question: Is there some substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying both, $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1 \leq \sigma \leq \sigma_2$?

Second, we can effectively compute the set of maximal substitutions σ satisfying $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1 \leq \sigma \leq \sigma_2$. It is a finite set of regular substitutions.

Cor. 31 shows that we can effectively compute the set of maximal substitutions σ satisfying $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma(x) \neq \emptyset$ for all $x \in \mathcal{X}$. It is a finite set of regular substitutions. The requirement $\sigma(x) \neq \emptyset$ is rather natural and it reflects the original setting of Conway.

1.4 Roadmap to prove Theorem 29

Our proof is designed to be accessible for readers who are familiar with basic results about metric spaces and regular languages over trees.³

We don't rely on (or use) the theory of monads. This is a categorical concept, which leads to a general notion of a *syntactic algebra*, see the arXiv-paper of Bojańczyk [3] for finite trees.⁴ For infinite trees notion of a *syntactic algebra* was given by Blumensath [2].

³ The simpler case of finite and infinite words can be found in Sec. 8.1 as a kind of warm-up exercise.

⁴ Conway's result for finite trees follows from the theory of monads, too. Personal communication Mikołaj Bojańczyk, 2019. For the notion of a monad see [21, Chapters III to V].

In our paper we use nondeterministic finite (top-down) parity-tree automata to define an appropriate *congruence* of finite index. This is conceptually simple but there is no free lunch: the index of the congruence defined by an automaton is not guaranteed to be the smallest one.⁵

Having a convenient notion of a congruence, the next step is to define $\sigma_{\text{io}}(s)$ for finite and infinite trees such that $\sigma_{\text{io}}(s)$ is indeed the intended fixed point for Eq.(1). For finite trees the set $\sigma_{\text{io}}(s)$ can be defined by induction on the size of s . Then $\sigma_{\text{io}}(s)$ becomes the unique least fixed point of Eq.(1) satisfying $\sigma_{\text{io}}(x) = \sigma(x)$ for all symbols of rank zero. For infinite trees the definition is more subtle. Given a tree s and a substitution σ we introduce a notion of a *choice function*. That is a function $\gamma : \text{Pos}(s) \rightarrow T(\Sigma \cup H) \cup \{\perp\}$ where $\text{Pos}(s)$ is the set of positions (that is: vertices) of s such that the following holds. If $u \in \text{Pos}(s)$ is labeled by x , then γ selects a tree $\gamma(u) \in \sigma(x)$. If $\sigma(x) = \emptyset$, then $\gamma(u)$ is not defined, which is denoted as $\gamma(u) = \perp$. To each choice function we will associate a Cauchy sequence $\gamma_n(s)$ in some complete metric space $T_\perp(\Sigma \cup \mathcal{X} \cup H)$; and we let $\gamma_\infty(s) = \lim_{n \rightarrow \infty} \gamma_n(s)$ be its limit. We can think of the space $T_\perp(\Sigma \cup \mathcal{X} \cup H)$ as a union of the usual Cantor space $T(\Sigma \cup \mathcal{X} \cup H)$ together with an isolated point \perp which has distance 1 to every other point. Then we define

$$\sigma_{\text{io}}(s) = \{\gamma_\infty(s) \mid \gamma \text{ is a choice function for } s \text{ and } \gamma_\infty(s) \neq \perp\}. \quad (2)$$

It turns out that this definition coincides with the natural definition for finite trees, and it satisfies Eq.(1), too. Another crucial step on the road to show Thm. 29 is the following result. If σ and R are regular, then the “inverse image” $\sigma_{\text{io}}^{-1}(R) = \{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq R\}$ is a regular set of trees. In order to prove this fact we use two well-known results. First, the class of regular tree languages can be characterized by alternating parity-automata, and the semantics of these automata can be defined by parity-games [22, 23, 5]. Second, parity-games are determined and have positional (= memoryless) winning strategies, [17].

2 Notation and preliminaries

We let $\mathbb{N} = \{0, 1, \dots\}$ denote the set of natural numbers, $\mathbb{P} = \mathbb{N} \setminus \{0\}$, and \mathbb{P}^* to be the monoid of finite sequences of positive integers with the operation “.” and the neutral element ϵ . For $r \in \mathbb{N}$ we let $[r] = \{1, \dots, r\}$. We write 2^S for the power set of S , and identify every element $x \in S$ with the singleton $\{x\}$.

A *rooted tree* is a nonempty, connected, and directed graph $t = (V, E)$ with vertex set V and without multiple edges such that there is exactly one vertex, the *root*, without any incoming edge and all other vertices have exactly one incoming edge. As a consequence, for every vertex $v \in V$ there is exactly one directed path from the root to v . Since there are no multiple edges we assume without restriction $E \subseteq \{(u, v) \mid u, v \in V, u \neq v\}$. If $(u, v) \in E$ is an edge, then we say that v is a *child* of u , and u is the *parent* of v . A *leaf* of t is a vertex without children. Throughout, we restrict ourselves to directed graphs where the set of edges is (at most) countable. Hence, it is possible to encode the vertex set of a rooted tree as a subset of *positions* $\text{Pos}(t) \subseteq \mathbb{P}^*$ satisfying the following conditions: $\epsilon \in \text{Pos}(t)$, and if $u.j \in \text{Pos}(t)$, then both $u \in \text{Pos}(t)$ and $u.i \in \text{Pos}(t)$ for all $1 \leq i \leq j$. Using this, we have $\text{root}(t) = \epsilon$ and edge set $\{(u, u.i) \mid u, u.i \in \text{Pos}(t)\}$. We are mainly interested in *ordered* trees: these are rooted trees where the children of a node are equipped with a “left-to-right” ordering. In such a case the “left-to-right breadth-first” ordering on positions can be represented by the length-lexicographical ordering on \mathbb{P}^* . The *size* of a tree t is the cardinality of $\text{Pos}(t)$. The *level* of a vertex u is the length of the unique directed path starting at the root and ending

⁵ With respect to worst case complexity this is more an advantage than a problem. “Generically” it is debatable whether it makes sense to spend any efforts in computing syntactic congruences. It doesn’t.

in u . Dually, the *height* of a vertex u is the length of the longest directed path starting at u . Leaves have height zero. The *height of a tree* is the height of its root.

Typically, and actually without restriction, every position in a tree has a label in some set Δ . Such a tree t can be defined therefore through a mapping $t : \text{Pos}(t) \rightarrow \Delta$ where $\text{Pos}(t)$ is the set of positions and $t(u) \in \Delta$ is the label of the position u . The set of all trees with labels in Δ is denoted by $T(\Delta)$. Its subset of finite trees is denoted by $T_{\text{fin}}(\Delta)$. More precisely, if $\Gamma \subseteq \Delta$, then $T_{\Gamma\text{-fin}}(\Delta)$ is the set of trees where the number of positions with a label in Γ is finite.

Let $t, t' \in T(\Delta)$ and $u \in \text{Pos}(t)$. We define the trees $t|_u$ and $t[u \leftarrow t']$ as usual:

$$\begin{aligned} \text{Pos}(t|_u) &= \{v \in \mathbb{P}^* \mid u.v \in \text{Pos}(t)\} \text{ with labeling } t|_u(v) = t(u.v), \\ \text{Pos}(t[u \leftarrow t']) &= \{u.u' \mid u' \in \text{Pos}(t')\} \cup \{v \in \text{Pos}(t) \mid u \text{ is not a prefix of } v\}, \\ t[u \leftarrow t'](u.u') &= t'(u') \text{ and } t[u \leftarrow t'](v) = t(v) \text{ if } u \text{ is not a prefix of } v. \end{aligned}$$

In almost all our cases (there is one exception in the proof of Prop. 13) the degree of vertices in a tree is bounded by a constant depending on Δ , and vertices have finite degree depending on their label. A *ranked alphabet* is a nonempty finite set of labels Δ with a *rank* function $\text{rk} : \Delta \rightarrow \mathbb{N}$. Trees over a ranked alphabet have to satisfy the following additional constraint:

- $\forall u \in \text{Pos}(t)$, if $t(u) = x$, then $\{1, \dots, \text{rk}(x)\} = \{i \in \mathbb{N} \mid u.i \in \text{Pos}(t)\}$.

Trees in $T(\Delta)$ are represented by the set of *terms*, too. These are the ordered trees $t \in T(\Delta)$. We adopt the standard notation of terms to denote trees: $x(s_1, \dots, s_r)$ represents the tree s with $s(\varepsilon) = x$ and $s|_i = s_i$ for all $i \in [\text{rk}(x)]$. Henceforth, if not otherwise specified, we let $\Omega = \mathcal{X} \cup \Sigma \cup H$ be a ranked alphabet consisting of three sets: \mathcal{X} is the set of *variables*, Σ is the set of *function symbols*, H is the set of *holes*. We require $(\mathcal{X} \cup \Sigma) \cap H = \emptyset$ but $\mathcal{X} \cap \Sigma \neq \emptyset$ is not forbidden. Symbols $a \in \Sigma$ with $\text{rk}(a) = 0$ are called *constants*. Holes are not constants, but they have rank 0, too. For simplicity, we assume $H = [r_{\max}]$ where $r_{\max} \in \mathbb{N}$ satisfies $\text{rk}(x) \leq r_{\max}$ for all $x \in \Sigma \cup \mathcal{X}$. To make the theory nontrivial, we assume that there is some $x \in \Sigma \cup \mathcal{X}$ with $\text{rk}(x) \geq 1$. In particular, $T(\Omega)$ contains finite trees where some leaves are labeled by a hole. For $\text{rk}(x) \geq 2$ there are also infinite trees with this property.

Given a tree $t \in T(\Omega)$ we denote by $\text{leaf}_i(t)$ the set of leaves which are labeled by the hole $i \in H$. The length-lexicographical ordering of positions induced by \mathbb{P}^* is a well-order, which has the type of either a finite ordinal or the ordinal ω . This well-order induces a linear order on $\text{leaf}_i(t)$, which can be represented by a downward-closed subset of \mathbb{P} . That is, we may write $\text{leaf}_i(t) = \{i_j \mid 1 \leq j \leq |\text{leaf}_i(t)|\}$. If $|\text{leaf}_i(t)| = \infty$, then this means $\text{leaf}_i(t) = \{i_j \mid j \in \mathbb{P}\}$. The term notation is also convenient for a concise notation of infinite trees using fixed point equations. For example, let $f \in \Sigma$ be a function symbol of rank 2, then there is exactly one tree $t \in T(\Sigma \cup \{1\})$ which satisfies the equation $t = f(t, 1)$. It is depicted in Fig. 3. The set $\text{leaf}_1(t)$ is the set of all leaves. There is no leftmost leaf, but a rightmost leaf which is in turn the first one in the length-lexicographical ordering.

$$t = f(t, 1) = \begin{array}{c} f \\ \swarrow \quad \searrow \\ f \quad 1 \\ \swarrow \quad \searrow \\ \text{---} \quad 1 \end{array} \quad \text{Pos}(t) = \begin{array}{c} \varepsilon \\ \swarrow \quad \searrow \\ 1 \quad 2 = 1_1 \\ \swarrow \quad \searrow \\ 1.1 \quad 1.2 = 1_2 \\ \swarrow \quad \searrow \\ \text{---} \quad 1.1.2 = 1_3 \end{array}$$

■ **Figure 3** The tree (representing an “infinite comb”) $t = f(t, 1)$ has infinitely many holes labeled by 1, but no leftmost hole. The set $\text{Pos}(t) \subseteq \mathbb{P}^*$ is depicted on the right. Following our convention, the subset $\text{leaf}_1(t) \subseteq \text{Pos}(t)$ is written as $\{1_1, 1_2, 1_3, \dots\}$.

Suppose that sets $T_x \subseteq T(\Sigma \cup [r])$ and $T_i \subseteq T(\Sigma)$ for $i \in [r]$ are defined. Then we define the set $T_x[i_j \leftarrow T_{i_j}] \subseteq T(\Sigma)$ as the union over all trees $t_x[i_j \leftarrow t_{i_j}]$ where $t_x \in T_x$ and $t_{i_j} \in T_{i_j}$. This is explained in more detail in Sec. 2.1.

2.1 Substitutions: outside-in and inside-out for trees in $T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$

The ranked alphabet $\Sigma \cup \mathcal{X}$ contains function symbols and variables. As mentioned in the introduction we allow $\Sigma \cap \mathcal{X} \neq \emptyset$. In the following, if $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H)}$ is a mapping which is specified on the set of variables, then we extend it to a mapping $\sigma : \Sigma \cup \mathcal{X} \rightarrow 2^{T(\Sigma \cup H)}$ by letting

$$\sigma(f) = \{f(1, \dots, \text{rk}(f))\} \text{ for all } f \in H \cup \Sigma \setminus \mathcal{X}. \quad (3)$$

We say that a mapping $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ is a *substitution* if σ satisfies the following additional property

$$\sigma(x) \subseteq T(\Sigma \cup [\text{rk}(x)]) \setminus H \text{ for all } x \in \mathcal{X}. \quad (4)$$

Note that (3) and (4) together imply that $t \in T(\Sigma \cup [\text{rk}(x)]) \setminus H$ for all $t \in \sigma(x)$ and for all $x \in \Sigma \cup \mathcal{X}$. For all elements $x \in \Sigma \cup \mathcal{X}$ of rank zero we have $\sigma(x) \subseteq T(\Sigma)$. The set of substitutions is a partial order by letting $\sigma \leq \sigma'$ if $\sigma(x) \subseteq \sigma'(x)$ for all $x \in \mathcal{X}$. A substitution σ is called a *homomorphism* (resp. *partial homomorphism*) if $|\sigma(x)| = 1$ (resp. $|\sigma(x)| \leq 1$) for all $x \in \mathcal{X}$. Since we identify elements and singletons we can also say that a homomorphism⁶ is specified by a mapping $\sigma : \mathcal{X} \rightarrow T(\Sigma \cup H) \setminus H$. Recall that $t[i_j \leftarrow t_{i_j}]$ denotes the tree produced from s by replacing every position $i_j \in \text{leaf}_i(s)$ with the tree t_{i_j} . According to [11, 12] there are two natural ways to extend a substitution σ to $T_{\text{fin}}(\Sigma \cup \mathcal{X})$: *outside-in* (OI for short) and *inside-out* (IO for short). The corresponding notations are σ_{oi} and σ_{io} respectively. Our goal is to extend σ to extensions σ_{io} and σ_{oi} from $T(\Sigma \cup H)$ to $2^{T(\Sigma \cup H)}$ such that $T(\Sigma \cup \mathcal{X})$ maps to $2^{T(\Sigma)}$. In this section we restrict ourselves to maps from $T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$ to arbitrary subsets of $T(\Sigma \cup H)$. The inside-out-extension of σ_{io} including infinite trees relies on “choice functions”. We postpone the general definition of σ_{io} to Sec. 2.3. The corresponding extension of σ_{oi} can be found in Sec. 8.3. It is not used elsewhere.

For a tree $s = x(s_1, \dots, s_r) \in T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$, the sets of trees $\sigma_{\text{io}}(s_i)$ and $\sigma_{\text{oi}}(s_i)$ are defined by induction on the maximal level of a position labeled by some variable. If $s \in T(\Sigma)$, then we let $\sigma_{\text{oi}}(s) = \sigma_{\text{io}}(s) = \{s\}$. Thus, we may assume that some variable occurs in s . For $r = 0$ we let $\sigma_{\text{oi}}(s) = \sigma_{\text{io}}(s) = \sigma(x) \subseteq T(\Sigma)$. For $r \geq 1$, the sets $\sigma_{\text{io}}(s_i) \subseteq \sigma_{\text{oi}}(s_i) \subseteq T(\Sigma)$ are defined by induction for all $i \in [r]$. Hence, we can define

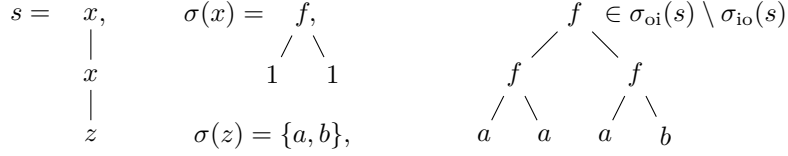
$$\sigma_{\text{io}}(s) = \{t_x[i_j \leftarrow t_i] \mid t_x \in \sigma(x) \wedge t_i \in \sigma_{\text{io}}(s_i)\} \quad (5)$$

$$\sigma_{\text{oi}}(s) = \{t_x[i_j \leftarrow t_{i_j}] \mid t_x \in \sigma(x) \wedge t_{i_j} \in \sigma_{\text{oi}}(s_i)\} \quad (6)$$

► **Remark 1.** For the interested reader we note that $\sigma_{\text{io}}(s) \neq \emptyset \iff \sigma_{\text{oi}}(s) \neq \emptyset$. Indeed, $\sigma_{\text{io}}(s) \neq \emptyset$ implies $\sigma_{\text{oi}}(s) \neq \emptyset$ because $\sigma_{\text{io}}(s) \subseteq \sigma_{\text{oi}}(s) \subseteq 2^{T(\Sigma)}$ by definition. The other direction holds for $r = 0$. For $r \geq 1$ the induction (on the maximal level) tells us that $\sigma_{\text{oi}}(s_i) \neq \emptyset$ implies $\sigma_{\text{io}}(s_i) \neq \emptyset$. ◀

There is more flexibility in OI than in IO because positions $i_j \neq i_k$ of a hole $i \in H$ may be substituted with σ_{oi} by different trees t_{i_j} and t_{i_k} , whereas with σ_{io} they are substituted by the same tree $t_{i_j} = t_{i_k}$. This means the i -th child of a position u in s is duplicated if there is some $t \in \sigma(s(u))$ where $|\text{leaf}_i(t)| \geq 2$. Hence, $\sigma_{\text{io}}(s) \subsetneq \sigma_{\text{oi}}(s)$ is possible because of “duplication”. Fig. 4 depicts an example for $\sigma_{\text{io}}(s) \neq \sigma_{\text{oi}}(s)$. Here, x is a variable of rank 1 and z is a variable of rank 0 (playing the role of “end-of-string”). Note that in this situation (with $\sigma(x) = f(1, 1)$ and $\sigma(z) = \{a, b\}$) neither $\sigma_{\text{io}}(x^*(z))$ nor $\sigma_{\text{oi}}(x^*(z))$ is regular, since every tree in $\sigma_{\text{io}}(x^n z)$ and in $\sigma_{\text{oi}}(x^n z)$ is a full binary tree with 2^n leaves. Reading the labels of the leaves from left-to-right reveals the difference. For $n \geq 1$ and $a \neq b$ the “leaf-language”

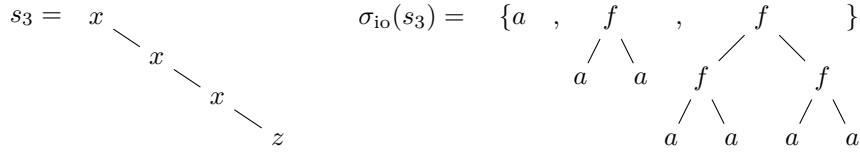
⁶ Homomorphisms σ such that $\sigma(x) \not\subseteq H$ for all $x \in \mathcal{X}$ are called *non-erasing* by Courcelle in [8, page 117]. Thus, there is some difference in notation between our paper and [8]



■ **Figure 4** Let $a \neq b$ be two constants and $n \geq 1$. Duplication of the hole 1 yields $|\sigma_{\text{io}}(x^n(z))| = 2$ and $|\sigma_{\text{oi}}(x^n(z))| = 2^{n+1}$. In particular, $\sigma_{\text{oi}}(x^n(z)) \setminus \sigma_{\text{io}}(x^n(z))$ for all $n \geq 1$.

of $\sigma_{\text{io}}(x^n z)$ is the two-element set $\{a^{2^n}, b^{2^n}\}$ whereas the “leaf-language” of $\sigma_{\text{io}}(x^n z)$ has 2^n elements. It is equal to $\{a, b\}^{2^n}$.

Consider a modification of the example in Fig. 4 by letting $\sigma(z) = \emptyset$ (or any other subset of trees in $T(\Sigma)$) and $\sigma(x) = \{f(1, 1), a\}$. This leads to a striking situation where $\sigma_{\text{io}}(x^*(z))$ is not regular but $\sigma_{\text{oi}}(x^*(z))$ is the set of all finite trees over $\{f, a\}$. Indeed, $\sigma_{\text{io}}(x^n(z))$ is the set of all full binary trees with 2^m leaves for all $0 \leq m < n$. The set $\sigma_{\text{io}}(x^3(z))$ is depicted in Fig. 5. All leaves are labeled by a and all inner nodes are labeled by f . In contrast, $\sigma_{\text{oi}}(x^n(z))$ is the set of all trees in $T(\{f, a\})$ with height less than n .



■ **Figure 5** $L = x^*(z)$ and $\sigma(x) = \{f(1, 1), a\}$. Then $\sigma_{\text{io}}(L)$ is not regular, but $\sigma_{\text{oi}}(L) = T(\{f, a\})$.

Yet another variant is given by $\Sigma = \{f, a, b\}$, $H = \{1\}$, and $\mathcal{X} = \{x, y\}$ where $\text{rk}(f) = 2$, $\text{rk}(x) = \text{rk}(y) = \text{rk}(a) = 1$, and $\text{rk}(b) = 0$. Let $\sigma(x) = t$ where $t = f(t, 1)$ as in Fig. 3 and $\sigma(y) = a^*(b)$. Then we obtain the following situation as depicted in Fig. 6.

In the sequel, we show that the domain of σ_{io} can be extended to $T(\Sigma \cup \mathcal{X})$ such that equation (5) still holds for trees in $T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$. We shall use two auxiliary concepts: complete metric spaces (Sec. 2.2) and choice functions (Sec. 2.3).

2.2 Complete metric spaces with \perp for “undefined”

Let us introduce a special constant $\perp \notin \Omega$ which plays the role of “undefined”. The idea is that an empty set $\sigma_{\text{io}}(s)$ is replaced by the singleton \perp . We turn $T(\Omega \cup \{\perp\})$ into a metric space by defining a metric d , which makes sure that both sets, $\{\perp\}$ and $T(\Omega)$, become clopen (= open and closed) subspaces in $(T(\Omega \cup \{\perp\}), d)$. We let $2^{-\infty} = 0$ and define:

$$d(s, t) = \begin{cases} 2^{-\inf\{|u| \mid u \in \text{Pos}(s) \cap \text{Pos}(t) \wedge s(u) \neq t(u)\}} & \text{if either } s, t \in T(\Omega) \text{ or both, } s, t \notin T(\Omega), \\ 1 & \text{otherwise: } s \in T(\Omega) \iff t \notin T(\Omega). \end{cases}$$

Note that every constant in $T(\Omega \cup \{\perp\})$, like the constant \perp , has distance 1 to every other element in $T(\Omega \cup \{\perp\})$. As usual, $T(\Omega)$ is closed in $T(\Omega \cup \{\perp\})$ but, thanks to the definition of d , it is also open. Hence, each of the subsets $T(\Omega)$, $\{\perp\} \cup T(\Omega)$, and their complements with respect to $T(\Omega \cup \{\perp\})$ are clopen subsets in $(T(\Omega \cup \{\perp\}), d)$. The restriction to the clopen subspace $\{\perp\} \cup T(\Omega)$ yields a compact ultra-metric space such that

$$d(s, t) = 2^{-\inf\{|u| \mid u \in \text{Pos}(s) \cap \text{Pos}(t) \wedge s(u) \neq t(u)\}} \quad (7)$$

holds for all $s, t \in \{\perp\} \cup T(\Omega)$. The ambient metric space $(T(\Omega \cup \{\perp\}), d)$ is not complete and therefore not compact. Indeed, recall that, by our convention, there is some function



■ **Figure 6** For all $n \in \mathbb{N}$ there exists $t_n \in \sigma_{\text{io}}(xy(b))$ as shown on the left. For all sequences $(n_i)_{i \in \mathbb{N}}$ there is a tree $t \in \sigma_{\text{oi}}(xy(b))$ as on the right. The set $\sigma_{\text{oi}}(xy(b))$ is regular, but $\sigma_{\text{io}}(xy(b))$ is not.

symbol of rank at least one. Hence, there exists an infinite tree s and a Cauchy sequence $(t_n)_{n \in \mathbb{N}}$ in $T(\Omega \cup \{\perp\}) \setminus T(\Omega)$ such that $(t_n)_{n \in \mathbb{N}}$ converges to s in the usual Cantor-space, but not in our metric. For example, assume $\text{rk}(a) = 1$, then the Cauchy sequence $(a^n(\perp))_{n \in \mathbb{N}}$ does not have any limit in $(T(\Omega \cup \{\perp\}), d)$. However, as \perp represents “undefined”, we wish that $\lim_{n \rightarrow \infty} a^n(\perp) = \perp$. There is a way to achieve that: we identify the clopen set $T(\Omega \cup \{\perp\}) \setminus T(\Omega)$ with the constant \perp . Thereby \perp becomes an isolated point. To be precise, let us define the equivalence relation \sim on $T(\Omega \cup \{\perp\})$ which is induced by letting $\perp \sim t$ for all $t \in T(\Omega \cup \{\perp\}) \setminus T(\Omega)$. Now, we have $a^n(\perp) \sim \perp$ for all n . Hence, the image of the sequence in the quotient space $(a^n(\perp))_{n \in \mathbb{N}}$ is equal to the constant sequence $(\perp)_{n \in \mathbb{N}}$ with the obvious limit \perp . The natural embedding of $\{\perp\} \cup T(\Omega)$ into the quotient space $T(\Omega \cup \{\perp\})/\sim$ induces an isometry between $(T(\Omega) \cup \{\perp\}, d)$ and $(T(\Omega \cup \{\perp\})/\sim, d_\sim)$ where d_\sim is the canonical quotient metric⁷ on $T(\Omega \cup \{\perp\})/\sim$.

2.3 Choice functions and the definition of σ_{io} for infinite trees

Henceforth, $T_\perp(\Sigma \cup H)$ denotes the complete metric space $(T(\Sigma \cup H) \cup \{\perp\}, d)$ which, by the previous subsection, is identified with the quotient space $(T(\Sigma \cup H \cup \{\perp\})/\sim, d_\sim)$. A *choice function* for $s \in T(\Sigma \cup \mathcal{X})$ is a mapping $\gamma : \text{Pos}(s) \rightarrow T_\perp(\Sigma \cup H)$ such that

$$\gamma(u) \in \{\perp\} \cup T(\Sigma \cup [\text{rk}(s(u))]) \setminus H \text{ with } \gamma(u) = f(1, \dots, \text{rk}(f)) \text{ if } f = s(u) \in \Sigma \setminus \mathcal{X}.$$

For $u \in \text{Pos}(s)$ we let $\gamma|_u : \text{Pos}(s|_u) \rightarrow T_\perp(\Sigma \cup H)$ be the mapping defined by $\gamma|_u(u') = \gamma(u.u')$. Note that $\gamma|_u$ is a choice function for the subtree $s|_u$ whenever $u \in \text{Pos}(s)$ and γ is a choice function for s . For every $s \in T(\Sigma \cup \mathcal{X})$ and choice function γ for s , we define the sequence of trees $(\gamma_n(s))_{n \in \mathbb{N}}$ in $T(\Sigma \cup H \cup \{\perp\})$ as follows:

$$\gamma_0(s) = \gamma(\varepsilon) \quad \text{and} \quad \gamma_n(s) = \gamma(\varepsilon)[i_j \leftarrow (\gamma|_i)_{n-1}(s|_i)] \text{ if } n > 0. \quad (8)$$

This yields a Cauchy sequence $(\gamma_n(s))_{n \in \mathbb{N}}$ in $T_\perp(\Sigma \cup H)$ and therefore, $\lim_{n \rightarrow \infty} \gamma_n(s)$ exists. Since choice functions don't map tree positions to holes, we have $\lim_{n \rightarrow \infty} \gamma_n(s) \in \{\perp\} \cup T(\Sigma)$.

► **Definition 2.** Let $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be a substitution and $s \in T(\Sigma \cup \mathcal{X})$. We define:

1. The tree $\gamma_\infty(s) = \lim_{n \rightarrow \infty} \gamma_n(s) \in \{\perp\} \cup T(\Sigma)$.
2. The set $\Gamma(\sigma, s)$ by the set of choice functions for s satisfying for all $u \in \text{Pos}(s)$ with $x = s(u)$ the following condition: if $\sigma(x) \neq \emptyset$, then $\gamma(u) \in \sigma(x)$, otherwise $\gamma(u) = \perp$.
3. The set $\sigma_{\text{io}}(s) = \{\gamma_\infty(s) \mid \gamma \in \Gamma(\sigma, s)\} \setminus \{\perp\}$.

► **Proposition 3.** Let $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be a substitution, $s = x(s_1, \dots, s_r) \in T(\Sigma \cup \mathcal{X})$ be a tree, and $\gamma \in \Gamma(\sigma, s)$ be a choice function. Then $\gamma_\infty(s) = \gamma(\varepsilon)[i_j \leftarrow (\gamma|_i)_\infty(s_i)]$.

⁷ The interested reader may consult Sec. 8.4 for the definition of a *quotient metric* in general topology.

Proof. Since $\gamma_\infty(s) = \lim_{n \rightarrow \infty} \gamma_n(s)$ and $(\gamma|_i)_\infty(s_i) = \lim_{n \rightarrow \infty} (\gamma|_i)_n(s_i)$ for all i , we obtain

$$\begin{aligned} \gamma(\varepsilon)[i_j \leftarrow (\gamma|_i)_\infty(s_i)] &= \gamma(\varepsilon)[i_j \leftarrow \lim_{n \rightarrow \infty} (\gamma|_i)_n(s_i)] \\ &= \gamma(\varepsilon)[i_j \leftarrow \lim_{n \rightarrow \infty} (\gamma|_i)_{n-1}(s_i)] \\ &= \lim_{n \rightarrow \infty} \gamma(\varepsilon)[i_j \leftarrow (\gamma|_i)_{n-1}(s_i)] \\ &= \lim_{n \rightarrow \infty} \gamma_n(s) = \gamma_\infty(s). \end{aligned}$$

◀

► **Corollary 4.** Let $s = x(s_1, \dots, s_r) \in T(\Sigma \cup \mathcal{X})$ be a tree. Then we have $\sigma_{\text{io}}(s) = \{t_x[i_j \leftarrow t_i] \mid t_x \in \sigma(x) \wedge t_i \in \sigma_{\text{io}}(s_i)\}$. In particular, for finite trees the new definition of $\sigma_{\text{io}}(s)$ in Def. 2 agrees with the earlier one given in Eq.(5).

Proof. We use the notation as given in Def. 2.

$$\begin{aligned} \sigma_{\text{io}}(s) &= \bigcup \{ \gamma(\varepsilon)[i_j \leftarrow (\gamma|_i)_\infty(s_i)] \mid \gamma \in \Gamma(\sigma, s) \} && \text{by Prop. 3} \\ &= \{ t_x[i_j \leftarrow (\gamma|_i)_\infty(s_i)] \mid t_x \in \sigma(x), \gamma|_i \in \Gamma(\sigma, s|_i) \} && \text{trivial} \\ &= \{ t_x[i_j \leftarrow t_i] \mid t_x \in \sigma(x) \wedge t_i \in \sigma_{\text{io}}(s_i) \} && \text{by Def. 2} \end{aligned}$$

Hence, $\sigma_{\text{io}}(s) = \{t_x[i_j \leftarrow t_i] \mid t_x \in \sigma(x) \wedge t_i \in \sigma_{\text{io}}(s_i)\}$ as desired. ◀

The following examples show that $\sigma_{\text{io}}(s)$ is not closed in $T_\perp(\Sigma)$, in general. Every (infinite) tree $s \in T(\Sigma)$ can be written as a limit $\lim_{n \rightarrow \infty} s_n$ where s_n are finite trees in $T(\Sigma)$ (assuming, as usual, that Σ contains a constant). Then there are Cauchy sequences $(t_n)_{n \in \mathbb{N}}$ with $t_n \in \sigma_{\text{io}}(s_n)$ which do not converge to any tree in $\sigma_{\text{io}}(s)$. In these examples x is a variable of rank one, $\Sigma = \{f, a, b\}$ with a constant b , $\text{rk}(a) = 1$, $\text{rk}(f) = 2$, and where only the first hole $1 \in H$ is used. In the first example we have $s = s_n$ for all n .

► **Example 5.** Consider $s = x(b)$ and $\sigma(x) = \{a^n(1) \mid n \in \mathbb{N}\}$. Then $s = \lim_{n \rightarrow \infty} s_n$ where $s_n = s$. We define $t_n = \gamma^{(n)}(s) = a^n(b)$ where $\gamma^{(n)} \in \Gamma(s, \sigma)$, $\gamma^{(n)}(\varepsilon) = a^n(1)$, and note that $(t_n)_{n \in \mathbb{N}}$ is a Cauchy sequence with every $t_n \in \sigma_{\text{io}}(s_n)$, but

$$\lim_{n \rightarrow \infty} t_n = a^\omega \notin \sigma_{\text{io}}(s) = \{a^n(b) \in \mathbb{N}\}.$$

The next example is a modification of Ex. 5 such that s becomes infinite and the sequence s_n is increasing such that $s_{n+1} = s_n[1 \leftarrow x(1)]$.

► **Example 6.** Let $s = f(a^\omega, x(b))$ and σ be defined by $\sigma(x) = \{a^n(1) \mid n \in \mathbb{N}\}$. Then $s = \lim_{n \rightarrow \infty} s_n$ where $s_n = f(a^n(1), x(b))$. We have $t_n = f(a^n(1), a^n(b)) \in \sigma_{\text{io}}(s_n)$ since for each n we may use the choice function $\gamma^{(n)}$ with $\gamma^{(n)}(u) = a^n(b)$ where $u = 2 \in \text{Pos}(s)$. The t_n 's form a Cauchy sequence with $t = f(a^\omega, a^\omega) = \lim_{n \rightarrow \infty} t_n$ but $t \notin \sigma_{\text{io}}(s) = \{f(a^\omega, a^n(b)) \mid n \in \mathbb{N}\}$.

3 Regular tree languages

There are several equivalent definitions for regular languages of finite and infinite trees, see [5, 15, 25, 27]. Here, we will consider regular languages of finite and infinite trees from $T(\Sigma \cup \mathcal{X} \cup H \cup \{\perp\})$. Note that for $\Gamma \subseteq \Delta \subseteq \Sigma \cup \mathcal{X} \cup H \cup \{\perp\}$, the sets $T_{\text{fin}}(\Delta)$ (more general: $T_{\Gamma\text{-fin}}(\Delta)$) and $T(\Delta)$ are regular subsets of $T(\Sigma \cup \mathcal{X} \cup H \cup \{\perp\})$.

3.1 Nondeterministic tree automata with parity acceptance

We use parity-NTAs (nondeterministic tree automata with a parity acceptance condition) as the basic reference to characterize regular tree languages. The parity condition is used to accept infinite trees. In our definition a parity-NTA accepts sets of finite and infinite trees. Alternating tree automata with a parity acceptance condition will be considered later.

Let Δ be any ranked alphabet (finite as usual) with a rank function $\text{rk} : \Delta \rightarrow \mathbb{N}$. A *parity-NTA* over Δ is specified by a tuple $A = (Q, \Delta, \delta, \chi)$ where Q is a nonempty finite set of states, δ is the *transition relation*, and $\chi : Q \rightarrow C$ is a coloring with $C = \{1, \dots, |C|\}$. Without restriction, we assume that $|C|$ is odd. We have

$$\delta \subseteq \bigcup_{f \in \Delta} Q \times \{f\} \times Q^{\text{rk}(f)}. \quad (9)$$

Thus, δ is a set of tuples (p, f, p_1, \dots, p_r) where $r = \text{rk}(f)$. The acceptance condition is defined as follows.

► **Definition 7.** Let A be a parity-NTA and let $t \in T(\Delta)$.

- A run ρ of t is a relabeling of the positions of t by states which is consistent with the transitions. That is, $\rho : \text{Pos}(t) \rightarrow Q$ is a mapping such that for all $u \in \text{Pos}(t)$ with $t(u) = f$ there is a transition $(p, f, p_1, \dots, p_{\text{rk}(f)}) \in \delta$ such that $\rho(u) = p$ and $\rho(u.j) = p_j$ for all $j \in [\text{rk}(f)]$. (See [15] for more details.)
If $\rho(\varepsilon) = p$, then we say that ρ is a run of t at state p .
- Let ρ be a run and (u_0, u_1, u_2, \dots) be an infinite path in $\text{Pos}(t)$ such that u_{j+1} is a child of u_j . The path is *accepting* if the number

$$\liminf_{k \rightarrow \infty} (\chi(\rho(u_k))) = \max \{ \min \{ \chi(\rho(u_i)) \in C \mid i \geq k \} \mid k \in \mathbb{N} \}$$

is even. This means that the minimal color appearing infinitely often on this path is even. The run ρ is *accepting* if all its all infinite directed paths are accepting.

- By $\text{Run}_A(t, p)$ we denote the set of **accepting** runs of t at state p . (If the context to A is clear, we might simply write $\text{Run}(t, p)$.)
- If $p \in Q$, then we let $L(A, p) = \{t \in T(\Delta) \mid \text{Run}_A(t, p) \neq \emptyset\}$. It is the accepted language of A at state p .
- For $P \subseteq Q$ we define $L(A, P)$ by

$$L(A, P) = \bigcap \{L(A, p) \mid p \in P\}. \quad (10)$$

The set $L(A, P) \subseteq T(\Delta)$ is called the *accepted language* at the set P . The convention is as usual: $L(A, \emptyset) = T(\Delta)$.

For $\text{rk}(f) = 0$ there are no children; and we accept f at state p if and only if $(p, f) \in \delta$. Therefore, no final states appear in the specification of parity-NTA. For a finite tree $t \in T(\Delta)$ we have $t \in L(A, p)$ if and only if there exists a run ρ of t with $\rho(\varepsilon) = p$ since the parity condition holds vacuously. Thus, for trees in $T_{\text{fin}}(\Delta)$ no other accepting condition is required than the existence of a run. The following fact is well-known, see for example [15]. We shall use Prop. 8 as the working definition for the present paper.

► **Proposition 8.** A language $L \subseteq T(\Delta)$ is regular if and only if there is a parity-NTA $A = (Q, \Delta, \delta, \chi)$ and a state $p \in Q$ such that $L = L(A, p)$.

It is a well-known classical fact that the class of regular tree languages forms an effective Boolean algebra [25]. This means that first, there is a parity-NTA accepting all trees in $T(\Delta)$ and second, given two parity NTAs A_1 and A_2 with states p_1 and p_2 respectively, we can effectively construct a parity NTA accepting $L(A_1, p_1) \setminus L(A_2, p_2)$. In particular, Prop. 8 implies that for each parity-NTA A and all subsets $P, P' \subseteq Q$ we can effectively construct a parity-NTA B with a single initial state q such that $L(B, q) = L(A, P) \setminus L(A, P')$.

4 Tasks and profiles

In this section we introduce the notions of a *task* and a *profile*. Throughout, we let $H = \{1, \dots, |H|\}$ be a set of holes and Σ be a ranked alphabet such that $\text{rk}(f) \leq |H|$ for all $f \in \Sigma$. We also fix a parity-NTA $B = (Q, \Sigma, \delta, \chi)$ such that $L(B, p) \subseteq T(\Sigma)$ for all $p \in Q$. Every such B is extended to a parity-NTA B_H by adding more transitions ensuring that every hole $i \in H$ is accepted at all states. Thus, we let $B_H = (Q, \Sigma \cup H, \delta_H, \chi)$ where $\delta_H = \delta \cup (Q \times H)$. Clearly, $L(B_H, p)$ is regular and therefore the complement $T(\Sigma \cup H) \setminus L(B_H, p)$ is regular, too. For the rest of this section, a run ρ of a tree t refers to a tree $t \in T(\Sigma \cup H)$ and the NTA B_H . Thus $\text{Run}(t, p) = \text{Run}_{B_H}(t, p)$ if not stated explicitly otherwise. Let ρ be any run of a tree $t : \text{Pos}(t) \rightarrow \Sigma$ (not necessarily accepting) and let $\alpha = (u_0, u_1, \dots)$, $\alpha' = (u'_0, u'_1, \dots)$ be infinite directed paths in $\text{Pos}(t)$ with $\varepsilon = u_0 = u'_0$. We concentrate on infinite paths because these paths decide whether ρ is accepting. Since directed paths in trees are directed away from the root, u_{i+1} is a child of u_i and u'_{i+1} is a child of u'_i for all $i \in \mathbb{N}$.) The paths α and α' define infinite words over Σ and the run ρ defines infinite words $\rho(\alpha)$ and $\rho(\alpha')$ over Q . Suppose each path is cut into infinitely many finite and nonempty pieces w_i and w'_i such that we can write $\rho(\alpha) = \rho(w_0)\rho(w_1)\cdots$ and $\rho(\alpha') = \rho(w'_0)\rho(w'_1)\cdots$ where all $\rho(w_i)$ and $\rho(w'_i)$ are in Q^+ . Assume that α' satisfies the parity condition. We aim for a condition based on the factors $\rho(w_i)$ and $\rho(w'_i)$ which is strong enough to imply that the other infinite word α satisfies the parity condition, too.⁸

Naturally, such a condition is related to colors which appear in $\chi\rho(w_i)$ and $\chi\rho(w'_i)$. More precisely, let c_i (resp. c'_i) denote the minimal color in $\chi\rho(w_i)$ (resp. $\chi\rho(w'_i)$) for $i \in \mathbb{N}$. We obtain two infinite words (c_0, c_1, \dots) and (c'_0, c'_1, \dots) over the alphabet C of colors. If we compare locally each color c_i with the corresponding color c'_i , then intuitively, with respect to the parity condition, an even color is better than an odd color. A small even color is better than a large even color. However, a large odd color is better than a small odd color. Based on that intuition, let us introduce the *best-ordering* \preceq_{best} on the set $\{0, \dots, |C|\}$. Note that we explicitly include 0 which is not in $\chi(Q)$ and that $|C|$ is odd by our convention. We let

$$0 \preceq_{\text{best}} 2 \preceq_{\text{best}} \cdots \preceq_{\text{best}} |C| - 1 \preceq_{\text{best}} |C| \preceq_{\text{best}} |C| - 2 \preceq_{\text{best}} \cdots \preceq_{\text{best}} 3 \preceq_{\text{best}} 1 \quad (11)$$

Indeed, in the linear order \preceq_{best} all even numbers are “better” than the odd numbers. Among even numbers smaller is better. Among odd numbers larger is better.

As a consequence, if $c_i \preceq_{\text{best}} c'_i$ for all $i \in \mathbb{N}$ and if the parity condition holds for α' , then it holds for α .

► **Definition 9.** A task is a tuple $(p, \psi_1, \dots, \psi_{|H|})$ with $p \in Q$ and $\psi_i : Q \rightarrow \{0\} \cup \chi(Q)$ for $1 \leq i \leq |H|$. A profile is a set of tasks.

The range of ψ_i in Def. 9 is a subset of the set $\{0, \dots, |C|\}$ where 0 does not belong to $\chi(Q)$, but it is the best number in the linear order \preceq_{best} . As we will see, for “satisfying” a task the value $\psi_i(q) = 0$ is better than any value in $\chi(Q)$. The best (resp. worst) value in $\chi(Q)$ is the smallest even (resp. odd) number. The best-ordering defines a partial order on the set of tasks: we write $(p, \psi_1, \dots, \psi_{|H|}) \leq (p', \psi'_1, \dots, \psi'_{|H|})$ if first, $p = p'$ and second, $\psi_i(q) \preceq_{\text{best}} \psi'_i(q)$ for all $q \in Q$ and $i \in H$.

► **Definition 10.** Let $\tau = (p, \psi_1, \dots, \psi_{|H|})$ be a task and $t, t' \in T(\Sigma \cup H \cup \{\perp\})$ be trees.

1. A run $\rho \in \text{Run}_{B_H}(t, p)$ satisfies τ if the following condition holds for all leaves $i_j \in \text{leaf}_i(t)$:

⁸ To find such a sufficient condition we just mimic the standard concept of a strongly recognizing morphism in the theory of ω -regular words as exposed for example in the appendix, Sec. 8.1.

- If $c_\rho(i_j)$ denotes the minimal color (w.r.t. to the natural order \leq for natural numbers) on the path from the root of t to position $i_j \in \text{leaf}_i(t)$ in the tree $\chi\rho : \text{Pos}(t) \rightarrow C$, then we have $c_\rho(i_j) \preceq_{\text{best}} \psi_i(\rho(i_j))$.

We also write $\rho \models \tau$ in that case. Moreover, we write $t \models \tau$ if there exists some $\rho \in \text{Run}_{B_H}(t, p)$ such that $\rho \models \tau$. Note that a tree containing the symbol \perp cannot satisfy any task because there is no run on such a tree.

2. A run $\rho \in \text{Run}_{B_H}(t, p)$ defines τ if first, $\rho \models \tau$ and second, for all $q \in Q$ and $i \in H$ we have $\psi_i(q) \geq 1 \iff \exists i_j \in \text{leaf}_i(t) : \psi_i(q) = c_\rho(i_j)$.
3. The set of all tasks τ such that there is a tree $t \in T(\Sigma \cup H \cup \{\perp\})$ with $t \models \tau$ is denoted by $\mathcal{T} = \mathcal{T}_B$.
4. By $\pi(t)$ we denote the set of tasks τ such that $t \models \tau$. In particular, $\tau \in \pi(t)$ implies $\tau \in \mathcal{T}$.
5. The set of all profiles π such that there is a tree $t \in T(\Sigma \cup H \cup \{\perp\})$ with $\pi = \pi(t)$ is denoted by $\mathcal{P} = \mathcal{P}_B$.
6. By \equiv_B we denote the equivalence relation which is defined by $t \equiv_B t' \iff \pi(t) = \pi(t')$.

Note that $\pi(t) = \emptyset$ if and only if $\text{Run}_{B_H}(t, p) = \emptyset$ for all $p \in Q$. Since $\perp \notin T(\Sigma \cup H)$, we have $\pi(\perp) = \emptyset$. The index of \equiv_B is finite since it is bounded by $|\mathcal{P}|$. More precisely,

$$|\{\{t \in T(\Sigma \cup H) \mid t \equiv_B t'\} \mid t' \in T(\Sigma \cup H)\}| \leq |\mathcal{P}_B| \leq 2^{|Q| \cdot (|C|+1)^{|H \times Q|}}. \quad (12)$$

The value 0 in the range of ψ_i plays the following role: Let $q \in Q$ and $\rho \models (p, \psi_1, \dots, \psi_{|H|})$ with $\rho \in \text{Run}(p, t)$. Then $\psi_i(q) = 0$ implies $\{i_j \in \text{leaf}_i(t) \mid \rho(i_j) = q\} = \emptyset$. The condition is vacuously true if $\text{leaf}_i(t) = \emptyset$. The following proposition says that $\{t \in T(\Sigma \cup H) \mid t \models \tau\}$ is effectively regular for every task $\tau \in \mathcal{T}_B$. The proof of Prop. 11 is based on a product construction where the first component simulates the NTA B_H and the second component remembers the minimal color appearing on paths in runs ρ of B_H at states p .

► **Proposition 11.** *Let $\tau = (p, \psi_1, \dots, \psi_{|H|}) \in \mathcal{T}_B$ be any task. Then the set of trees $\{t \in T(\Sigma \cup H) \mid t \models \tau\}$ is effectively regular. More precisely, $\{t \in T(\Sigma \cup H) \mid t \models \tau\} = L(B_\tau, (p, \chi(p)))$ where $B_\tau = (Q \times C, \Sigma \cup H, \delta_\tau, \chi_\tau)$ is the following parity-NTA.*

For a color $c \in C$, a function symbol $f \in \Sigma$ with $r = \text{rk}(f)$, a hole $i \in H$, and states $q, p_1, \dots, p_r \in Q$ we define δ_τ by the following equivalences:

$$((q, c), f, (p_1, \min\{c, \chi(p_1)\}), \dots, (p_r, \min\{c, \chi(p_r)\})) \in \delta_\tau \iff (q, f, p_1, \dots, p_r) \in \delta_B \quad (13)$$

$$((q, c), i) \in \delta_\tau \iff c \preceq_{\text{best}} \psi_i(q) \quad (14)$$

The color of a pair (q, c) is defined by $\chi_\tau(q, c) = \chi(q)$.

Proof. Let $\text{pr}_1 : Q \times C \rightarrow Q, (p, c) \mapsto p$ be the projection onto the first component. Let $t \in T(\Sigma \cup H)$. The aim is to show that pr_1 defines for t a canonical bijection between accepting runs ρ at a state p in B_H satisfying τ and the set of accepting runs ρ_τ at the state $(p, \chi(p))$ in B_τ . This implies $\{t \in T(\Sigma \cup H) \mid t \models \tau\} = L(B_\tau, (p, \chi(p)))$ and hence, $\{t \in T(\Sigma \cup H) \mid t \models \tau\}$ is effectively regular.

The definition of δ_τ says that $\text{pr}_1\rho' : \text{Pos}(t) \rightarrow Q$ is a run in $\text{Run}_{B_H}(t, p)$ for every $t \in T(\Sigma \cup H)$ and every run $\rho' \in \text{Run}_{B_\tau}(t, (p, \chi(p)))$. Here, as usual, $\text{pr}_1\rho' = \text{pr}_1 \circ \rho'$ denotes the composition. Consider any nonempty directed path $\varepsilon, u_1, \dots, u_k$ in t , then, by induction on k , we see that $\rho'(u_k) = (q_k, c_k)$ satisfies $c_k = \min\{c_1, \dots, c_k\}$. We claim that $\text{pr}_1\rho' \models \tau$. Indeed, the minimal color seen on every infinite directed path of the tree $\rho't$ with $\rho'(\varepsilon) = (p, \chi(p))$ is even. Since $\chi_\tau(q, c) = \chi(q)$, the same holds for $\text{pr}_1\rho'$. Together with the definition of $\delta_\tau((q, c), i)$ we obtain the claim that $\text{pr}_1\rho' \models \tau$. Thus, the projection onto the first component defines a mapping

$$\text{pr}_1 : \text{Run}_{B_\tau}(t, (p, \chi(p))) \rightarrow \{\rho \in \text{Run}_{B_H}(t, p) \mid \rho \models \tau\}, \rho' \mapsto \text{pr}_1(\rho') = \text{pr}_1\rho'. \quad (15)$$

We wish a bijection. Therefore, let us also define a mapping $\rho \mapsto \rho_\tau$ in the other direction from $\{\rho \in \text{Run}_{B_H}(t, p) \mid \rho \models \tau\}$ to $\text{Run}_{B_\tau}(t, (p, \chi(p)))$. Given $\rho \in \text{Run}_{B_H}(t, p)$ such that $\rho \models \tau$, we define the run ρ_τ top-down beginning at the root with $\rho_\tau(\varepsilon) = (p, \chi(p))$. Now, assume that $\rho_\tau(u) = (q, c)$ is defined for $u \in \text{Pos}(t)$ such that $\rho(u) = q$. Suppose $t(u) = f$ and $\text{rk}(f) = r$. Then, thanks to $\rho \in \text{Run}_{B_H}(t, p)$, there is some $(q, f, p_1, \dots, p_r) \in \delta_{B_H}$ such that $\rho(u.i) = p_i$ for all $1 \leq i \leq r$. However, since $\rho_\tau(u) = (q, c)$ is fixed there is exactly one corresponding transition $((q, c), f, (p_1, \min\{c, \chi(p_1)\}), \dots, (p_r, \min\{c, \chi(p_r)\})) \in \delta_\tau$ and we obtain $\rho_\tau(u.i) = \min\{c, \chi(p_i)\}$ for all $1 \leq i \leq r$. This is clear for $f \in \Sigma$ by (13) and $f = i \in H$ by (14) because $\rho \models \tau$. Moreover, since every infinite path in the tree ρ satisfies the parity condition, the same is true for the run ρ_τ because the color of a state $\chi(q, c)$ is defined by the first component: $\chi(q, c) = \chi(q)$. Thus, $\rho_\tau \in \text{Run}_{B_\tau}(t, (p, \chi(p)))$ as desired. A straightforward inspection shows $(\text{pr}_1 \rho')_\tau = \rho'$ for all $\rho' \in \text{Run}_{B_\tau}(t, (p, \chi(p)))$ and $\text{pr}_1(\rho_\tau) = \rho$ for all $\rho \in \text{Run}_{B_H}(t, p)$ where $\rho \models \tau$. This shows that the mapping pr_1 defined in (15) is bijective. We conclude $\{t \in T(\Sigma \cup H) \mid t \models \tau\} = L(B_\tau, (p, \chi(p)))$. ◀

► **Corollary 12.** *Let π be a profile, then $\{t \in T(\Sigma \cup H \cup \{\perp\}) \mid \pi(t) = \pi\}$ is effectively regular.*

Proof. Let $\pi = \emptyset$, first. Then $\{t \in T(\Sigma \cup H \cup \{\perp\}) \mid \pi(t) = \pi\}$ is the disjoint union of $T(\Sigma \cup H \cup \{\perp\}) \setminus T(\Sigma \cup H)$ (which is regular) and the set $\{t \in T(\Sigma \cup H) \mid \pi(t) = \pi\}$. For $\pi \neq \emptyset$, we have $\{t \in T(\Sigma \cup H \cup \{\perp\}) \mid \pi(t) = \pi\} \subseteq \{t \in T(\Sigma \cup H) \mid \pi(t) = \pi\}$. Thus, it is enough to show that $\{t \in T(\Sigma \cup H) \mid \pi(t) = \pi\}$ is regular. This set coincides with

$$\bigcap_{\tau \in \pi} \{t \in T(\Sigma \cup H) \mid t \models \tau\} \cap \bigcap_{\tau \notin \pi} \{t \in T(\Sigma \cup H) \mid t \not\models \tau\} \quad (16)$$

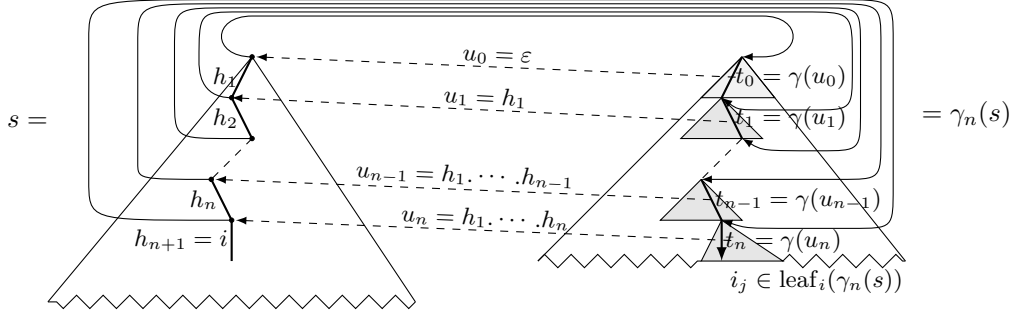
which is a finite intersection of languages, or complements of languages, from the set of languages $\mathcal{L} = \{\{t \in T(\Sigma \cup H) \mid t \models \tau\} \mid \tau \in \tau_B\}$. By Prop. 11, the family \mathcal{L} consists of regular tree languages. Since the class of regular tree languages in $T(\Sigma \cup H \cup \{\perp\})$ is an effective Boolean algebra [25], we conclude that $\{t \in T(\Sigma \cup H \cup \{\perp\}) \mid \pi(t) = \pi\}$ is effectively regular too. ◀

► **Proposition 13.** *Let B be a parity NTA with state set Q such that $L(B, p) \subseteq T(\Sigma)$ for all $p \in Q$ and $s \in T(\Sigma \cup \mathcal{X})$ be a tree. If $\gamma, \gamma' : \text{Pos}(s) \rightarrow T_\perp(\Sigma \cup H)$ are two choice functions for s such that $\gamma'(u) \equiv_B \gamma(u)$ for all $u \in \text{Pos}(s)$, then $\gamma_\infty(s) \in L(B, p_0) \iff \gamma'_\infty(s) \in L(B, p_0)$ for all states p_0 of B .*

In the following proof we encounter trees with infinite directed paths and where nodes may have an infinite degree: the β -sequences. This requires some attention.

Proof. Let $s = x(s_1, \dots, s_r)$ and $\gamma_\infty(s) \in L(B, p_0)$. By symmetry, it is enough to show $\gamma'_\infty(s) \in L(B, p_0)$. Recall that $\gamma_\infty(s)$ is the limit of trees $\gamma_n(s)$. Fig. 7 depicts the basic relationship between paths in $\gamma_n(s)$ and paths in s . For the path in s it shows positions $u \in \text{Pos}(s)$ which can be identified with positions on the corresponding path in $\gamma_n(s)$. Let us explain the relationship of the depicted paths in more detail. We begin with a directed path p in $\gamma_n(s)$ starting at the root ε which defines $t_0 = \gamma(\varepsilon)$. The path p stays inside t_0 or it leaves t_0 using a hole labeled by some $h_1 \in H$. Assuming the second case, we let $u_0 = \varepsilon$ and $h_1 = u_1 \in \text{Pos}(s) \cap \mathbb{P}$. Hence, $t_1 = \gamma(u_1)$ exists⁹. The path p continues in the tree $\gamma_n(s)$. Again, there are two possibilities: it stays inside t_1 or it leaves t_1 using a hole labeled by some $h_2 \in H$. If it leaves, we let $u_2 = h_1.h_2 \in \text{Pos}(s) \cap \mathbb{P}^2$. If possible, we continue by defining $u_3 \in \text{Pos}(s)$, $t_3 = \gamma(u_3)$, and $h_3 \in H$, etc. From the very beginning we face two cases: either the path stops at some leaf of $\gamma_n(s)$ (which is also a leaf of t_n) labeled by some

⁹ Observe that $\text{Pos}(s) \cap H = [r]$. Thus, the choice function determines t_1 .



■ **Figure 7** A directed path in $\gamma_n(s)$ starting at the root maps to a unique sequence of positions $\varepsilon, h_1, h_1.h_2, \dots$ in s (indicated by dashed arrows). These positions of the sequence can be identified with positions in $\gamma_n(s)$: the top positions of the small triangles (indicated by plain arrows from positions in s to positions in $\gamma_n(s)$). The depicted directed path ends in a leaf i_j , but it could also stay inside t_n or end at a leaf of t_n which is a constant.

$h_{n+1} \in H$ or there is some $m \leq n$ such that the path stays in t_m . Fig. 7 shows a situation where p ends in a leaf of $\gamma_n(s)$. This implies that there exists $u_n.h_{n+1} \in \text{Pos}(s)$. Assume p stays in t_m for some $m \leq n$. This path can be finite or it can be infinite. Moreover, it may possibly stay inside t_m although, perhaps, for all $h \in H$ there exist positions $u_m.h \in \text{Pos}(s)$. However, in that case, we do not define any position u_{m+1} .

Now, consider $u \in \text{Pos}(s)$ and $p \in Q$ such that $\text{Run}_{B_H}(\gamma(u), p) \neq \emptyset$. Then each run $\rho \in \text{Run}_{B_H}(\gamma(u), p)$ defines a task $\tau_\rho = (p, \psi_1, \dots, \psi_{|H|})$ by

$$\psi_i(q) = \sup_{\preceq_{\text{best}}} \{c_\rho(i_j) \in \{0, \dots, |C|\} \mid \exists i_j : q = \rho(i_j)\}. \quad (17)$$

The existence of ρ implies that $\gamma(u) \models \tau_\rho$. The task τ_ρ is minimal¹⁰ among all tasks τ such that $\gamma(u) \models \tau$. The minimality of τ_ρ implies that there is some leaf i_j in $\gamma(u)$ where $q = \rho(i_j)$ if and only if $\psi_i(q) \geq 1$. Moreover, Since $\gamma(u) \equiv_B \gamma'(u)$, there exists some $\rho' \in \text{Run}_{B_H}(\gamma'(u), p)$ such that $\rho' \models \tau_\rho$. It follows that there is a mapping

$$\theta : \text{Run}_{B_H}(\gamma(u), p) \rightarrow \text{Run}_{B_H}(\gamma'(u), p), \quad \rho \mapsto \rho' \quad (18)$$

such that $\rho' = \theta(\rho)$ satisfies $\rho' \models \tau_\rho$.

The definition of θ is crucial to prove $\gamma'_\infty(s) \in L(B, p_0)$. Note that the shape of $t = \gamma(u)$ and $t' = \gamma'(u)$ might be very different. For example, t can be finite whereas t' is infinite, or vice versa. The trees share however $\text{leaf}_i(t) \neq \emptyset \iff \text{leaf}_i(t') \neq \emptyset$ because $\gamma(u) \equiv_B \gamma'(u)$. Still, the cardinalities of $\text{leaf}_i(t)$ and $\text{leaf}_i(t')$ might differ drastically.

The rest of the proof uses an additional concept of an *IO-prefix*. Let $t, t_\varepsilon, t_i \in T_\perp(\Sigma \cup H)$ be trees for each $i \in H$ where $\text{leaf}_i(t_\varepsilon) \neq \emptyset$. We say that t_ε is an *IO-prefix* of t if we can write $t = t_\varepsilon[i_j \leftarrow t_i]$.

In order to see $\gamma'_\infty(s) \in L(B, p_0)$ we fix a run $\rho \in \text{Run}(\gamma_\infty(s), p_0)$ witnessing $\gamma_\infty(s) \in L(B, p_0)$. Using this fixed ρ we construct a set of (finite or infinite) sequences $\alpha = (u_0, p_0, \rho_0) \cdots (u_k, p_k, \rho_k)$ and $\beta = (u_0, p_0, \rho'_0) \cdots (u_k, p_k, \rho'_k)$ such that for all i we have $u_i \in \text{Pos}(s)$, $p_i \in Q$, and $\rho_i \in \text{Run}(\gamma(u_i), p_i)$ (resp. $\rho'_i \in \text{Run}(\gamma'(u_i), p_i)$). See Fig. 7 for an intuition about the positions u_i . We require that u_{j+1} is child of u_j for all $0 \leq j < k$. The set of sequences α (resp. β) form themselves a rooted tree where the root is the empty sequence. The parent of a nonempty sequence $w_0 \cdots w_{k-1} w_k$ of length k is the prefix $w_0 \cdots w_{k-1}$. As we will see later, each node in the tree defined by an α -sequence has at most $|H \times C|$ children whereas a node in the tree defined by a β -sequence may have infinitely many children. The

¹⁰ Minimal w.r.t. the natural partial order on tasks defined by \preceq_{best}

number of children is actually equal to the size of the index sets I_α resp. I_β as defined in (19) and (20) below.

Every such sequence $\alpha = (u_0, p_0, \rho_0) \cdots (u_k, p_k, \rho_k)$ will define below a unique position $\nu(\alpha)$ in $\gamma_\infty(s)$ such that the subtree of $\gamma_\infty(s)$ at position $\nu(\alpha)$ has the tree $\gamma(u_k)$ as an IO-prefix as defined above. The same will be true for a sequence β with respect to $\gamma'_\infty(s)$. The position of the empty sequence $\nu(\varepsilon)$ is ε which is the root of $\gamma_\infty(s)$ and $\gamma'_\infty(s)$. We define a sequence $\alpha_1 = (u_0, p_0, \rho_0) = (\varepsilon, p_0, \rho_0)$ such that $\rho_0 \in \text{Run}(\gamma(\varepsilon), p_0)$ is induced by $\rho \in \text{Run}(\gamma_\infty(s), p_0)$. This yields $\beta_1 = (\varepsilon, p_0, \rho'_0)$ with $\rho'_0 = \theta(\rho_0)$ as defined by (18).

Now, let $k \in \mathbb{N}$ and assume that a sequence $\alpha = (u_0, p_0, \rho_0) \cdots (u_k, p_k, \rho_k)$ is already defined. By induction, we also know a position $\nu(\alpha') \in \text{Pos}(\gamma_\infty(s))$ where α' is defined by the equation $\alpha'(u_k, p_k, \rho_k) = \alpha$. Let

$$I_\alpha = \{(i, q) \in H \times Q \mid \exists i_j \in \text{leaf}_i(\gamma(u_k)) : \rho_k(i_j) = q\}. \quad (19)$$

For each $(i, q) \in I_\alpha$ we define a triple $(u_k.i, q, \rho(i, q))$. Here, $u_k.i$ is the i -th child of $u_k \in \text{Pos}(s)$. The children of α are defined by the sequences $\alpha \cdot (u_k.i, q, \rho(i, q))$. Thus, there are at most $|H \times Q|$ children in α -sequences. Before we define the run $\rho(i, q)$, let us define the position $\nu(\alpha) \in \text{Pos}(\gamma_\infty(s))$. By induction, the subtree at position $\nu(\alpha') \in \text{Pos}(\gamma_\infty(s))$ has $\gamma(u_k)$ as an IO-prefix; and every leaf $i_\ell \in \text{leaf}_i(\gamma(u_k))$ defines a unique position in $\gamma_\infty(s)$ in the subtree at position $\nu(\alpha')$. We choose $\nu(\alpha)$ as a leaf in $\gamma(u_k)$. For that we choose any leaf $i_j \in \text{leaf}_i(\gamma(u_k))$ such that $c_{\rho_k}(i_j) = \psi_i(q)$ where $\psi_i(q)$ refers to the task τ_{ρ_k} defined in (17). Hence, the minimal color on the path from the root in $\gamma(u_k)$ to the chosen leaf i_j induced by the run ρ_k is the greatest among all these colors among all $i_\ell \in \text{leaf}_i(\gamma(u_k))$ with respect to best ordering. Since the root of $\gamma(u_k)$ is $\nu(\alpha')$ we can define $\nu(\alpha)$ by the corresponding position of the chosen position i_j . The subtree of $\gamma_\infty(s)$ at position $\nu(\alpha)$ has the tree $\gamma(u_k.i)$ as an IO-prefix. Hence, ρ induces a run $\rho(i, q) \in \text{Run}(\gamma(u_k.i), q)$; and $(u_k.i, q, \rho(i, q))$ is defined by $(i, q) \in I_\alpha$. Note that α has a child if and only if $I_\alpha \neq \emptyset$. If $I_\alpha = \emptyset$ the sequence α is a leaf in the corresponding tree.

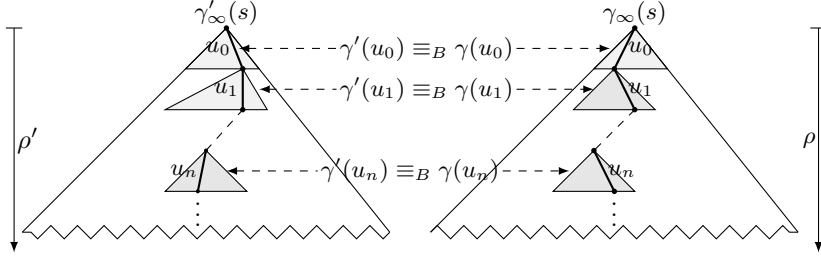
The definition of the tree of β -sequences is defined analogously. We let $k \in \mathbb{N}$ and assume that a sequence $\beta = (u_0, p_0, \rho'_0) \cdots (u_k, p_k, \rho'_k)$ is already defined such that $\rho'_i = \theta(\rho_i)$ for all $1 \leq i \leq k$. The mapping θ was defined in (18). By induction, we also know a position $\nu(\beta) \in \text{Pos}(\gamma'_\infty(s))$. It might be that β has infinitely many children because we do not choose a particular leaf (as we did for α -sequences) but have to consider all leaves i_j . More precisely, we define

$$I_\beta = \{(i, q, i_j) \in H \times Q \times \text{leaf}_i(\gamma'(u_k)) \mid i_j \in \text{leaf}_i(\gamma'(u_k)) \wedge \rho'_k(i_j) = q\}. \quad (20)$$

The set I_β is in a canonical bijection with the union $\bigcup \{\text{leaf}_i(\gamma'(u_k)) \mid i \in H\}$. For each $(i, q, i_j) \in I_\beta$ we define the run $\rho'(i, q) = \theta(\rho(i, q))$. (The run $\rho(i, q)$ was defined above for the α -sequence by choosing a particular leaf.) Thereby we obtain a triple $(u_k.i, q, \rho'(i, q))$. For $\beta' = \beta \cdot (u_k.i, q, \rho'(i, q))$ the position $\nu(\beta') \in \text{Pos}(\gamma'_\infty(s))$ is given in analogy to the position $\nu(\alpha')$. That is, we consider the position $\nu(\beta) \in \text{Pos}(\gamma'_\infty(s))$. The subtree $\nu(\beta)$ has $\gamma'(u_k)$ as an IO-prefix; and we let $\nu(\beta') = \nu(\beta(i, q, i_j))$ the position which corresponds to the leaf i_j . Since I_β is in a canonical bijection with the set of all leaves in $\gamma'(u_k)$ labeled by some hole $i \in H$, we see that we actually have defined runs $\rho'_n : \text{Pos}(\gamma'_n(s)) \rightarrow Q$ for all $n \in \mathbb{N}$, hence a run $\rho' : \text{Pos}(\gamma'_\infty(s)) \rightarrow Q$.

It remains to show that the run ρ' is accepting. Since it is a run, it is enough to consider infinite paths w' . These paths result from a finite or infinite sequence $\beta = (u_0, p_0, \rho'_0)(u_1, p_1, \rho'_1) \cdots$. If the infinite path w' is defined by some finite prefix of β , then it is accepting¹¹ because every finite prefix of β defines an accepting path of $\gamma'_\infty(s)$ at p_0 .

¹¹ Accepting paths were defined in Def. 7.



■ **Figure 8** The $u_i \in \text{Pos}(s)$ are the top positions of the small triangles according to Fig. 7. The run ρ defines a run ρ' . It satisfies the parity condition because every path in tree ρ has this property.

So, we may assume that β is infinite, too. Hence, w' can be cut into infinitely many finite paths $w' = w'_0 w'_1 \dots$ where each w'_i is a finite path inside $\gamma'(u_i)$. Each finite path belongs to a run $\rho'_{v'}$ of $\gamma'(u)$ at some position $v' \in \text{Pos}(\gamma'_\infty(s))$. It corresponds to the some run ρ_v of $\gamma(u)$ at some position $v \in \text{Pos}(\gamma_\infty(s))$. In this way we obtain an infinite path in $w = w_0 w_1 \dots$ in $\gamma_\infty(s)$ where each w_i is a path in $\gamma(u_i)$. Recall that β was defined together with a corresponding sequence $\alpha = (u_0, p_0, \rho_0)(u_1, p_1, \rho_1) \dots$ such that $\rho'_i = \theta(\rho_i)$. Every path in tree $\rho' : \text{Pos}(\gamma'_\infty(s))$ can be mapped to some accepting path in $\rho : \text{Pos}(\gamma_\infty(s))$. This situation is depicted in Fig. 8. The task τ_v defined by ρ_v is satisfied by $\rho'_{v'}$. Hence, the minimal color seen on the path from the root in $\gamma'(u)$ to a leaf i_j is never greater in the best-ordering than the minimal color seen on the path from the root in $\gamma(u)$ to a leaf i_j . As a consequence, the minimal color seen infinitely often on the run defined by the path w' is not greater in the best-ordering than the one defined by w . Since the minimal color seen infinitely often defined by w is even, the same is true for the minimal color seen infinitely often defined by w' .

Let's repeat the essential steps in the proof by looking at Fig. 8. A high-level interpretation of the picture shows us the top levels of two trees of possible infinite trees where the degree of nodes on the right is finite whereas the degree on of nodes on left can be infinite. On the other hand, $\gamma'_\infty(s)$ is just a tree where every vertex has at most r_{\max} children. In order to see whether $\gamma'_\infty(s) \in L(B, p_0)$ we have to consider all directed infinite path in $\gamma'_\infty(s)$. The left tree in the picture shows such a path ρ' . This path maps to a unique path in $\gamma_\infty(s)$ which is chosen such that acceptance is made most difficult, but still the acceptance condition holds because we $\gamma_\infty(s)$ has an accepting run ρ . We use the small triangles on the right to define local runs on the small triangles on the left. This is possible because $\gamma(u)$ and $\gamma'(u)$ satisfy the same set of tasks. The final step is to understand that the local runs on the left glue together to an accepting run of $\gamma'_\infty(s)$. This was the goal. ◀

4.1 The saturation of σ

We continue with parity-NTAs $B = (Q, \Sigma, \delta, \chi)$ and $B_H = (Q, \Sigma \cup H, \delta_H, \chi)$ where, as above, $\delta_H = \delta \cup (Q \times H)$.

► **Definition 14.** Let $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be a substitution. The saturation of σ (w.r.t. B) is the substitution $\hat{\sigma} : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ defined by

$$\hat{\sigma}(x) = \{t' \in T(\Sigma \cup \mathcal{X}) \setminus H \mid \exists t \in \sigma(x) : t' \equiv_B t\}.$$

Clearly, $\sigma \leq \hat{\sigma}$ and $\hat{\sigma}(x) \neq \emptyset$ imply $\hat{\sigma}(x) \neq \emptyset$.

► **Proposition 15.** Let $q_0 \in Q$ be a state of the NTA B and $L \subseteq T(\Sigma \cup \mathcal{X})$ be any subset. Then $\hat{\sigma}(x)$ is regular for all $x \in \mathcal{X}$. Moreover, $\sigma_{\text{io}}(L) \subseteq L(B_H, q_0) \iff \hat{\sigma}_{\text{io}}(L) \subseteq L(B, q_0)$.

Proof. We have $\widehat{\sigma}(x) = \bigcup_{\pi \in \{\pi(t) \mid t \in \sigma(x)\}} \{t' \in T(\Sigma \cup H) \mid \pi(t') = \pi\}$. This is a finite union. Thus, $\widehat{\sigma}(x)$ is regular by Cor. 12. Moreover, if $\widehat{\sigma}_{\text{io}}(L) \subseteq L(B, q_0)$, then $\sigma_{\text{io}}(L) \subseteq \widehat{\sigma}_{\text{io}}(L) \subseteq L(B, q_0) \subseteq L(B_H, q_0)$. Hence, let $\sigma_{\text{io}}(L) \subseteq L(B_H, q_0)$. Since $\sigma_{\text{io}}(L) \subseteq T(\Sigma)$, we may suppose $\sigma_{\text{io}}(L) \subseteq L(B, q_0)$. It suffices to show that, if $s \in T(\Sigma \cup \mathcal{X})$ and $\sigma_{\text{io}}(s) \subseteq L(B, q_0)$, then $\widehat{\sigma}_{\text{io}}(s) \subseteq L(B, q_0)$, that is, $\gamma'_{\infty}(s) \in L(B, q_0)$ for all $\gamma' \in \Gamma(s, \widehat{\sigma})$. Let $\gamma' \in \Gamma(s, \widehat{\sigma})$. Then, by definition of $\widehat{\sigma}$, for every $u \in \text{Pos}(s)$, where $\sigma(s(u)) \neq \emptyset$, we can choose a tree $\gamma(u) = t \in \sigma(s(u))$ such that $\gamma'(u) \equiv_B \gamma(u)$. Thus, $\gamma \in \Gamma(s, \sigma)$ and $\gamma_{\infty}(s) \in L(B, q_0)$ because $\sigma_{\text{io}}(s) \subseteq L(B, q_0)$. By Prop. 11, we have $\gamma'_{\infty}(s) \in L(B, q_0)$ too. Hence, $\widehat{\sigma}_{\text{io}}(s) \subseteq L(B, q_0)$ as desired. \blacktriangleleft

► **Corollary 16.** *Let $\sigma_1, \sigma_2 : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be regular substitutions such that $\sigma_1 \leq \sigma_2$, $R \subseteq T(\Sigma)$ be a regular tree language, and $L \subseteq T(\Sigma \cup \mathcal{X})$ be any subset. Then there is an effectively computable finite set S_2 of regular substitutions such that the following property holds. For every $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying both, $\sigma_1 \leq \sigma \leq \sigma_2$ and $\sigma_{\text{io}}(L) \subseteq R$ (resp. $\sigma_{\text{io}}(L) = R$), there is some maximal substitution $\sigma' \in S_2$ such that $\sigma'_{\text{io}}(L) \subseteq R$ (resp. $\sigma'_{\text{io}}(L) = R$) and $\sigma \leq \sigma' \leq \sigma_2$.*

Proof. We may assume that $R = L(B, q_0)$. Throughout the proof, given any substitution σ , the notation $\widehat{\sigma}$ refers to its saturation according to Def. 14. Moreover, by Prop. 15, every $\widehat{\sigma}$ is a regular substitution and if $\sigma_{\text{io}}(L) \subseteq R$, then $\sigma_{\text{io}}(L) \subseteq \widehat{\sigma}_{\text{io}}(L) \subseteq R$. Based on these facts, we define in a first step the following set of substitutions.

$$S_0 = \{\widehat{\sigma} \mid \sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H} \text{ is a substitution}\}. \quad (21)$$

The set S_0 is finite and effectively computable. Its cardinality is bounded by $2^{|\mathcal{P}_B \times \mathcal{X}|}$. Since σ_1 and $\widehat{\sigma}$ are regular for every $\widehat{\sigma} \in S_0$, we can effectively compute the following subset S_1 of S_0 , too.

$$S_1 = \{\widehat{\sigma} \in S_0 \mid \sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H} \wedge \sigma_1 \leq \sigma\} = \{\widehat{\sigma} \in S_0 \mid \sigma_1 \leq \widehat{\sigma}\}. \quad (22)$$

The second equation in (22) uses $\widehat{\widehat{\sigma}} = \widehat{\sigma}$. Now, for every substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying $\sigma_1 \leq \sigma \leq \sigma_2$ we have $\widehat{\sigma} \in S_1$. But $\widehat{\sigma} \leq \sigma_2$ might fail. However, since σ_2 is regular, we can calculate for each $\widehat{\sigma} \in S_1$ the substitution σ' such that $\sigma'(x) = \widehat{\sigma}(x) \cap \sigma_2(x)$ for all $x \in \mathcal{X}$. Hence, the following finite set is effectively computable:

$$S_2 = \{\sigma' : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H} \mid \exists \widehat{\sigma} \in S_1 \forall x \in \mathcal{X} : \sigma'(x) = \widehat{\sigma}(x) \cap \sigma_2(x)\}. \quad (23)$$

An easy reflection shows that for all σ such that $\sigma_1 \leq \sigma \leq \sigma_2$ and $\sigma_{\text{io}}(L) \subseteq R$ there is some substitution $\sigma' \in S_2$ such that $\sigma \leq \sigma' \leq \sigma_2$ and $\sigma'_{\text{io}}(L) \subseteq R$. Since S_2 is finite, there is a maximal element σ' in S_2 such that $\sigma \leq \sigma' \leq \sigma_2$ and $\sigma'_{\text{io}}(L) \subseteq R$. The assertion of Cor. 16 follows. \blacktriangleleft

► **Remark 17.** Note that Cor. 16 does not tell us how to decide whether there exists any substitution σ such that $\sigma_{\text{io}}(L) \subseteq R$ or $\sigma_{\text{io}}(L) = R$. However, it tells us that if there exists such a substitution σ , then there exist such a substitution in an effectively computable and finite set of regular substitutions. If this set is empty, then $\sigma_{\text{io}}(L) \subseteq R$ does not hold. \blacktriangleleft

4.2 The specialization of σ

The purpose of this section is to add for each profile $\pi = \pi(t)$ a specific finite tree t_π which satisfies the same profile and which depends on π , only. The roadmap for that is as follows. We enlarge the set Σ by various new function symbols in order to obtain a larger ranked alphabet $\Sigma_{\mathcal{P}}$. In particular, for each $\pi \in \mathcal{P}_B$ there is a function symbol f_π and a finite tree t_π with $t_\pi(\varepsilon) = f_\pi$. The tree t_π is either a constant or it has height 2. Simultaneously, we

define an extension $B_{\mathcal{P}}$ of B_H such that for each $t \in T(\Sigma \cup H)$ with $\pi = \pi(t)$ (w.r.t. the NTA B_H) we have $\pi = \pi(t) = \pi(t_\pi)$ where $\pi(t)$ and $\pi(t_\pi)$ are defined with respect to $B_{\mathcal{P}}$.

The application is for deciding whether a regular substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfies $\sigma_{\text{io}}(L) \subseteq L(B, q_0)$. In the decision procedure, we compute, in a preprocessing phase, for each x the set of profiles $\Pi_x = \{\pi(x) \mid t \in \sigma(x)\}$. The set Π_x is defined w.r.t. B_H , but this is the same as being defined w.r.t. $B_{\mathcal{P}}$. Hence, we can also write $\Pi_x = \{\pi(x) \mid t \in \check{\sigma}(x)\}$ where $\check{\sigma}(x)$ is a finite set of finite trees t_π of height at most 2. The substitution $\check{\sigma}$ yields the “specialization” of σ ; and it is enough to decide $\check{\sigma}_{\text{io}}(L) \subseteq L(B, q_0)$.

The formal definitions of f_π , $\Sigma_{\mathcal{P}}$ and $B_{\mathcal{P}}$ are in Def. 18 and Def. 19. Since every symbol f_π has a rank which only depends on π , but which will be defined through a tree t where $\pi = \pi(t)$, it is important to note that $\emptyset \neq \pi(t) = \pi(t')$ implies $\text{leaf}_i(t) \neq \emptyset \iff \text{leaf}_i(t') \neq \emptyset$ for all holes $i \in H$. This follows because if $\emptyset \neq \pi(t)$, then $\text{leaf}_i(t) \neq \emptyset$ if and only if for all tasks $(p, \psi_1, \dots, \psi_{|H|}) \in \pi$ there is some $q \in Q$ such that $\psi_i(q) > 0$. Indeed, let $\rho \in \text{Run}(t, p)$ such that $\rho \models \tau$. Then for all $i_j \in \text{leaf}_i(t)$ the run ρ yields some state $\rho(i_j) \in Q$. Since $\chi\rho(i_j) > 0$ we must have $\psi_i(\rho(i_j)) > 0$. In particular, $\emptyset \neq \pi(t) = \pi(t')$ implies $H(t) = H(t')$. Thus, for $\pi \in \mathcal{P}$: if $\pi = \pi(t)$, then $H(t)$ depends on π but not on the chosen t .

- **Definition 18.** 1. For each $t \in T(\Sigma \cup H)$ we denote by $H(t)$ the subset of holes $i \in H$ such that $\text{leaf}_i(t) \neq \emptyset$.
2. The alphabet $\Sigma_{\mathcal{P}}$ contains Σ and it contains, in addition, first a new function symbol $\$i$ of rank 1 for each $i \in H$, and second, for each $t \in T(\Sigma \cup H)$ and each profile $\emptyset \neq \pi = \pi(t) \in \mathcal{P}$ there is a new function symbol f_π of rank $|Q \times H(t)|$. In particular, if t is without holes, then f_π is a constant. We also include a new constant f_\emptyset in case $\emptyset \in \mathcal{P}_B$ (but we will make sure that no run at any state of $B_{\mathcal{P}}$ accepts f_\emptyset).
3. For each $\emptyset \neq \pi \in \mathcal{P}$ we choose any tree $t \in T(\Sigma \cup H)$ such that $\pi = \pi(t)$. Then we define the finite tree t_π which has the following structure. Its root is labeled by the symbol f_π . If t has no holes, then f_π is a constant. In the other case it has height 2. The children of the root are labeled by $\$i$ where $i \in H(t)$; and if a vertex is labeled by $\$i$, then its children are holes i_j (and thus labeled by $i \in H(t)$). There are $k = |Q \times H(t)|$ holes, all of them belong to $H(t)$, and each hole $i \in H(t)$ appears exactly $|Q|$ times (in some fixed order). The corresponding tree is depicted in Fig. 9. For $H(t) \neq \emptyset$ we have $k \geq 1$, for $H(t) = \emptyset$ we have $k = 0$ and the picture shows the constant f_π .

We are ready to define an extension $B_{\mathcal{P}}$ of the automaton B_H such that for all $t \in T(\Sigma \cup H)$ the finite tree t_π satisfies $t \equiv_{B_{\mathcal{P}}} t_\pi$ if and only if $\pi(t) = \pi$.

► **Definition 19.** We let $B_{\mathcal{P}} = (Q', \Sigma_{\mathcal{P}} \cup H, \delta_{\mathcal{P}}, \chi')$ denote the following parity-NTA. It is an extension of B_H . The set of states is $Q' = Q \cup (C \times Q \times H)$. The coloring χ is extended to states in $C \times Q \times H$ by $\chi'((c, q, h)) = c$.

The set of transitions $\delta_{\mathcal{P}}$ of $B_{\mathcal{P}}$ contains all transitions from B_H (that is: $\delta_H = \delta \cup Q \times H$) and, in addition, for each $\emptyset \neq \pi(t) \in \mathcal{P}_B$ the following set of tuples.

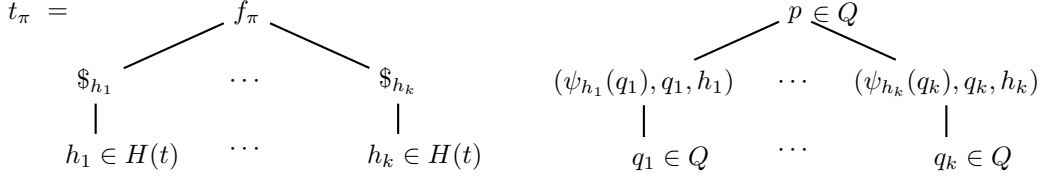
- Transitions $(p, f_\pi, (p_{s,j})_{(s,j) \in Q \times H(t)})$ for $\tau = (p, \psi_1, \dots, \psi_{|H|}) \in \pi$ such that

$$\{p_{s,j} \mid (s,j) \in Q \times H(t)\} = \{(\psi_i(q), q, i) \mid q \in Q, i \in H(t), \psi_i(q) \neq 0\}. \quad (24)$$

The sequence $(p_{s,j})_{(s,j) \in Q \times H(t)}$ is a tuple of length $\text{rk}(f_\pi)$ and each state $p_{s,j}$ is a triple such that Eq.(24) holds. The tuple allows repetitions of entries $p_{s,j}$.

- Transitions $((c, q, i), \$i, q)$ for all $(c, q, i) \in C \times Q \times H(t)$.

The cardinality of the set $\{(\psi_i(q), q, i) \mid q \in Q, i \in H(t), \psi_i(q) \neq 0\}$ in (24) varies. It can be any number between $|H(t)|$ and $\text{rk}(f_\pi)$; and it is not necessarily the same for all $\tau \in \pi$. This is the reason, why triples $(\psi_i(q), q, i)$ may occur several times. Note that for every $\tau \in \pi$ there exists some transition $(p, f_\pi, (p_{s,j})_{(s,j) \in Q \times H(t)})$ in $\delta_{\mathcal{P}}$.



■ **Figure 9** The tree t_π where $k = \text{rk}(f_\pi)$ and an accepting run ρ on t_π for $\tau = (p, \psi_1, \dots, \psi_{|H|}) \in \pi$.

► **Lemma 20.** *Let $t \in T(\Sigma \cup H)$, $\pi = \pi(t) \in \mathcal{P}_B$, and t_π as defined above. Then for all tasks $\tau \in \mathcal{T}_{B_P}$ we have $t_\pi \models \tau \iff \tau \in \pi$. In particular, $t \equiv_{B_P} t_\pi$ and $\pi(t_\pi) = \pi$.*

Proof. For each $\tau = (p, \psi_1, \dots, \psi_{|H|}) \in \pi$ the NTA B_P admits for t_π a successful run at p . This is clear for $H(t) = \emptyset$ because then t_π is a constant and $(p, t_\pi) \in \delta_P$. For $H(t) \neq \emptyset$ the accepting run is depicted on the right of Fig. 9. It is accepting because $Q \times H \subseteq \delta_H \subseteq \delta_P$. Hence, $\tau \in \pi$ implies $t_\pi \models \tau$. The converse follows from the definition of δ_P . In particular, $t_\pi \models \tau$ implies $\tau \in \mathcal{T}_B$. Since $\Sigma \subseteq \Sigma_P$ and $Q \subseteq Q'$, we have $t \in T(\Sigma_P \cup H)$ and $\mathcal{T}_B \subseteq \mathcal{T}_{B_P}$ and therefore, $t \equiv_{B_P} t_\pi$. ◀

Prop. 22 reduces the computation of $\sigma_{\text{io}}^{-1}(R)$ to the computation of $\check{\sigma}_{\text{io}}^{-1}(R)$. Here, $\check{\sigma}$ is the specialization of σ according to Def. 21 such that $\check{\sigma}(x)$ is a subset of the finite set of finite trees $\{t_\pi \mid \pi \in \mathcal{P}_B\}$.

► **Definition 21.** *Let $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H)}$ be a substitution and $\pi \in \mathcal{P}$ be a profile. The specialization $\check{\sigma} : \mathcal{X} \rightarrow 2^{T(\Sigma_P \cup H)}$ w.r.t. B_P is defined by the substitution*

$$\check{\sigma}(x) = \{t_\pi \in T(\Sigma_P \cup H) \mid \exists t \in \sigma(x) : \pi = \pi(t)\}. \quad (25)$$

Here, t_π is the tree defined according to Def. 18. That is, if $\pi = \pi(t)$ for some t without holes, then $t_\pi = f_\pi$. Otherwise $k \geq 1$ and t_π has height two according to Fig. 9 on the left-side.

Note that $\sigma(x) = \emptyset \iff \check{\sigma}(x) = \emptyset$ and $t_\emptyset \in \check{\sigma}(x) \iff \exists t \in \sigma(x) : t \models \emptyset$.

► **Proposition 22.** *Let $R = L(B, q_0) \subseteq T(\Sigma)$ and let Σ_P and $B_P = (Q', \Sigma_P \cup H, \delta_P, \chi')$ according to Def. 19. Let $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be any substitution and $\check{\sigma} : \mathcal{X} \rightarrow 2^{T(\Sigma_P \cup H)}$ its specialization according to Def. 21. Then we have*

$$\{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq L(B, q_0)\} = \{s \in T(\Sigma \cup \mathcal{X}) \mid \check{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\}. \quad (26)$$

Proof. The equality $\{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq L(B, q_0)\} = \{s \in T(\Sigma \cup \mathcal{X}) \mid \widehat{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\}$ follows directly from the definition of B_P . We apply Prop. 15 twice, once to B and once to B_P . We obtain $\{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq L(B_P, q_0)\} = \{s \in T(\Sigma \cup \mathcal{X}) \mid \widehat{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\}$ as well as $\{s \in T(\Sigma \cup \mathcal{X}) \mid \widehat{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\} = \{s \in T(\Sigma \cup \mathcal{X}) \mid \check{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\}$.

Note that $\widehat{\sigma}_{\text{io}}(s)$ refers to $\widehat{\sigma}(x) = \{t' \in T(\Sigma_P \cup H) \mid \exists t \in \sigma(x) : t' \equiv_{B_P} t\}$. That is to the saturation with respect to terms in $T(\Sigma_P \cup H)$. By Lem. 20, we have $t_\pi \equiv_{B_P} t$ for every $t \in \sigma(x)$ and profile $\pi = \pi(t)$. Hence, $\{s \in T(\Sigma \cup \mathcal{X}) \mid \widehat{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\} = \{s \in T(\Sigma \cup \mathcal{X}) \mid \check{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\}$. Putting everything together we see

$$\begin{aligned} \{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq L(B, q_0)\} &= \{s \in T(\Sigma \cup \mathcal{X}) \mid \widehat{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\} \\ &= \{s \in T(\Sigma \cup \mathcal{X}) \mid \check{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\} = \{s \in T(\Sigma \cup \mathcal{X}) \mid \check{\sigma}_{\text{io}}(s) \subseteq L(B_P, q_0)\}. \end{aligned}$$

The last line uses Prop. 13. The result follows. ◀

5 Parity games

A *directed (multi-)graph* is a tuple $G = (V, E, \iota, \mathbf{o})$ where V is the set of vertices and the sets ι and \mathbf{o} are functions from E to V . Here, $\iota(e)$ denotes the source of e and $\mathbf{o}(e)$ denotes the target of e . In our paper an *arena* is any tuple $A = (V_0, V_1, E, \iota, \mathbf{o})$ such that $(V, E, \iota, \mathbf{o})$ is a directed graph where $V = V_0 \cup V_1$ and V_0, V_1 are disjoint.

Phrased differently, every vertex is in exactly one subset V_i of V and we allow multiple edges between vertices. The maps ι and \mathbf{o} are extended to all paths of the graph A in the natural way: if $w = p_0, e_1, p_1, \dots, e_m, p_m$ is a path of A where $m \geq 0$ and $p_{i-1} = \iota(e_i)$, $p_i = \mathbf{o}(e_i)$ for all $1 \leq i \leq m$, then we let $\iota(w) = p_0$ and $\mathbf{o}(w) = p_m$.

A (*board of a*) *parity game* is defined by a pair (A, χ) where A is an arena and $\chi : V \rightarrow C$ is a mapping to a set of *colors* C .

Without restriction, we always assume that $C = \{1, \dots, |C|\}$ and that $|C|$ is odd. Let $p_0 \in V$. A *game at p_0* is a finite or infinite sequence $p_0, e_1, p_1, e_2, \dots$ such that $p_{i-1} = \iota(e_i)$, $p_i = \mathbf{o}(e_i)$ and $e_i \in E$ for $i \geq 1$. Moreover, we require that a game is infinite unless it ends in a sink. A *sink* is a vertex without outgoing edges¹². There are two players: P_0 (*Prover*) and P_1 (*Spoiler*). The rules of the game are as follows. It starts in the vertex p_0 . Let $m \geq 0$ such that a path $p_0, e_1, p_1, \dots, e_m, p_m$ is already defined with $p_m \in V_i$. If p_m is a sink, then player P_i loses. In the other case player P_i chooses an edge $e_{m+1} \in E$ such that $p_m = \iota(e_{m+1})$. Let $p_{m+1} = \mathbf{o}(e_{m+1})$, then the game continues with $p_0, \dots, p_m, e_{m+1}, p_{m+1}$.

If the game does not end in a sink, then the mutual choices define an infinite sequence. Prover P_0 wins an infinite game if the least color which appears infinitely often in $\chi(p_0), \chi(p_1), \dots$ is even. Otherwise, Spoiler P_1 wins that game.

A *positional strategy* for player P_i is a subset $E_i \subseteq E$ such that for each $u \in V_i$ there is at most one edge $e \in E_i$ with $\iota(e) = u$. Hence, every $e \in E_i$ is an edge of the arena. However, the property that E_i is *winning* depends on the pair $(\iota(e), \mathbf{o}(e))$, only. Therefore, we can assume $E_i \subseteq V_i \times V$. Each positional strategy defines a subarena $A_i = (V, E_i \cup \{e \in E \mid \iota(e) \in V_{1-i}\})$. In the arena A_i player P_{1-i} wins at a vertex p if and only if there exists some path starting at p which satisfies his winning condition defined above. Let $W_i \subseteq V$ be the set of vertices $p \in V$ of A_i where no path in A_i starting at p satisfies the winning condition of P_{1-i} . Note that W_i depends on the positional strategy E_i . The set W_i contains all vertices where player P_i wins positional (also called *memoryless*) by choosing E_i . The set W_i is a set of *winning positions* for P_i in the original arena A , because player P_i is able to win, no matter how P_{1-i} decides for $p_m \in V_{1-i}$ on the next outgoing edge $e \in E$ with $\iota(e) = p_m$.

► **Theorem 23** ([17]). *There exist positional strategies $E_i = V_i \times V \subseteq E$ for both players P_i such that their sets of winning positions $W_i \subseteq V$ (w.r.t. the subarenas A_i) satisfy $W_{1-i} = V \setminus W_i$. That is, V is the disjoint union of W_{1-i} and W_i .*

The theorem implies that for parity games there is no better strategy than a positional one. Thm. 23 is due to Gurevich and Harrington. Simplified proofs are e.g. in [15, 28].

5.1 Alternating tree automata

Let Δ be a finite ranked alphabet with the rank function $\text{rk} : \Delta \rightarrow \mathbb{N}$. Nondeterministic tree automata are special instances of alternating tree automata. ATAs were introduced in [22] and further investigated in [23]. A *parity-ATA* for Δ is a tuple $A = (Q, \Delta, \delta, \chi)$ where Q is a finite set of states, δ is the *transition function*, and $\chi : Q \rightarrow C$ is a coloring with $C = \{1, \dots, |C|\}$ and where, without restriction, $|C|$ is odd. For each $(q, f) \in Q \times \Delta$ there is exactly one transition which has the form (q, f, Φ) where Φ is a positive Boolean formula

¹²Some papers require that an arena has no sinks.

over the set $[\text{rk}(f)] \times Q$ and, moreover, Φ is written in disjunctive normal form¹³. This means that $(q, f, \Phi) \in \delta$ is written as

$$(q, f, \bigvee_{j \in J} \bigwedge_{k \in K_j} (d_k, p_k)) \quad (27)$$

where J and K_j are finite sets and $(d_k, p_k) \in [\text{rk}(x)] \times Q$. The first component $d_k \in [\text{rk}(f)]$ is also called a *direction*. With each $j \in J$ there is an associated finite set of indices K_j . We use a syntax for Boolean formulae defined by a context-free grammar, but in the notation redundant brackets are typically removed as we did in (27). The syntax still allows repetitions of pairs (d_k, p_k) . For example, we may have $K_j = \{k, \ell\}$ with a conjunction $((d_k, p_k) \wedge (d_\ell, p_\ell))$ where $(d_k, p_k) = (d_\ell, p_\ell)$. This will have no influence on the semantics which we define next, but it will introduce “multiple edges” in the corresponding game-arena, which will make the set of choices for Spoiler larger.¹⁴

By definition, let $p \in Q$, then $L(A, p)$ is the set of trees $s \in T(\Delta)$ such that Prover P_0 has a winning strategy the following parity-game in the arena $G(A, s)$ at vertex (ε, p) .

The set V_0 of vertices belonging to Prover of the arena $G(A, s)$ is defined by $V_0 = \text{Pos}(s) \times Q$. The color of (u, q) is $\chi(q)$. Next we define the set E_0 of outgoing edges at $(u, q) \in V_0$ by set of all (u, q, j) with $j \in J$ and where J appears under the disjunction in (27) of the unique transition $(q, s(u), \bigvee_{j \in J} \bigwedge_{k \in K_j} (d_k, p_k)) \in \delta$. We let $\iota(u, q, j) = (u, q)$ and we let $\mathbf{o}(u, q, j) = (u, q, j)$. We define the set of positions belonging to Spoiler by $V_1 = \mathbf{o}(E_0)$ and we let $\chi(u, q, j) = \chi(q)$. Thus, target function $\mathbf{o} : E_0 \rightarrow V_1$ is surjective by definition and every vertex in V_1 has an incoming edge. Note that for $(u, q) \in V_0$ and any $v \in V_1$ there is at most one edge $e \in E$ with $\iota(e) = (u, q)$ and $\mathbf{o}(e) = v$. No multiple edges occur here. This changes when we define the outgoing edges for vertices $(u, q, j) \in V_1$. The outgoing edges are the quadruples (u, q, j, k) where $k \in K_j$ for index sets under the conjunction in the transition $(q, s(u), \bigvee_{j \in J} \bigwedge_{k \in K_j} (d_k, p_k)) \in \delta$. We define $\iota(u, q, j, k) = (u, q, j)$. The index k defines a pair (d_k, p_k) and then we let $\mathbf{o}(u, q, j, k) = (u, d_k, p_k)$. Thus, there can be many edges with the source $(u, q, j) \in V_1$ and target (u, d_k, p_k) . Thanks to the condition $(d_k, p_k) \in [\text{rk}(x)] \times Q$ we are sure that the position $u.d_k$ exists as soon as $K_j \neq \emptyset$.

Let us phrase the definition of the arena from the perspective of the players if the game starts at some vertex $(u, q) \in V_0$. Then Prover chooses, if possible, an index $j \in J$ according to set E_0 . If J is empty, then Prover loses. (An empty disjunction is “false”.) In the other case, it is the turn of Spoiler P_1 , and the game continues at the vertex (u, q, j) belonging to P_1 with color $\chi(q)$. If K_j is empty, then Spoiler loses. (An empty conjunction is “true”.) Otherwise, Spoiler chooses an index $k \in K_j$ and the game continues at the vertex $(u, d_k, p_k) \in V_0 = \text{Pos}(s) \times Q$ (which exists). That is, the game continues at the position of the d_k -th child of u at state p_k . Prover wins an infinite game if and only if the least color occurring infinitely often is even.

A *parity-NTA* A is a special instance of an alternating automaton, where every transition as in Eq.(27) has the special form $(q, f, \bigvee_{j \in J} \bigwedge_{i \in [\text{rk}(f)]} (i, p_i))$. For convenience we use only the traditional syntax that δ is a set of tuples

$$(q, f, p_1, \dots, p_{\text{rk}(f)}). \quad (28)$$

The main results of alternating tree automata can be formulated as follows.

► **Theorem 24** ([22, 23]). *Let A be a parity-NTA as defined in Sec. 3.1 and p be a state. Then the following assertions hold.*

¹³ Let S be any (finite) set. Then the set of *positive Boolean formulae* $\mathbb{B}_+(S)$ is defined inductively. 1. The symbols \perp and \top and all $s \in S$ belongs to $\mathbb{B}_+(S)$. (The elements of S are the *atomic propositions*.) 2. If $\varphi, \psi \in \mathbb{B}_+(S)$, then $(\varphi \vee \psi) \in \mathbb{B}_+(S)$ and $(\varphi \wedge \psi) \in \mathbb{B}_+(S)$.

¹⁴ This larger set of choices is explicitly used in the definition of w_g in the proof of Lemma 26

1. Viewing A as a special instance of a parity-ATA using (27) or using the traditional syntax (28) and its semantics according to Def. 7 yields the same set $L(A, p)$.
2. Parity-ATAs characterize the class of regular tree languages: if $L(A, p)$ is defined by a parity-ATA A at a state p , then we can construct effectively a parity-NTA B and a state q such that $L(A, p) = L(B, q)$.

5.2 Alternating tree automata for languages of type $\sigma_{\text{io}}^{-1}(R)$

Thanks to Prop. 22 we know that $\sigma_{\text{io}}(L) \subseteq L(B, q_0) \iff \check{\sigma}_{\text{io}}(L) \subseteq L(B, q_0)$ where $\check{\sigma}_{\text{io}}$ is the specialization of σ . Since $\check{\sigma}_{\text{io}}(x)$ is, by construction, a finite set of finite trees, is enough to consider the concept of alternating tree automata in the setting where there is a finite set $T \subseteq T_{\text{fin}}(\Sigma \cup H) \setminus H$ of finite trees such that $\bigcup \{\sigma(x) \mid x \in \mathcal{X}\} \subseteq T$. This is not essential but allows to use the traditional definition in (27) for transitions where the index sets J and K_j are assumed to be finite.

► **Remark 25.** Recall that a choice function is defined for a substitution and yields an element in the cartesian product $\prod_{s \in T(\Sigma \cup \mathcal{X})} T_{\perp}(\Sigma \cup H)^{\text{Pos}(s)}$. Let us show that the set of mappings $\mathcal{X} \rightarrow T(\Sigma \cup H)$ can be viewed as a subset of $\prod_{s \in T(\Sigma \cup \mathcal{X})} T_{\perp}(\Sigma \cup H)^{\text{Pos}(s)}$. Indeed there is a natural embedding

$$\iota : T(\Sigma \cup H)^{\mathcal{X}} \rightarrow \prod_{s \in T(\Sigma \cup \mathcal{X})} T(\Sigma \cup H)^{\text{Pos}(s)} \quad (29)$$

Indeed, if $\varphi \in T(\Sigma \cup H)^{\mathcal{X}}$ and $x = s(u)$, then $\iota(\varphi)(x)$ is defined canonically by $\iota(\varphi)(s(u)) = \varphi(x)$ for $x \in \mathcal{X}$ and by $\iota(\varphi)(s(u)) = f(1, \dots, \text{rk}(f))$ for $x = f \in \Sigma \setminus \mathcal{X}$. It is also clear that ι is an embedding since $x(1, \dots, \text{rk}(x)) \in T(\Sigma \cup \mathcal{X})$. Moreover, if $\varphi \in T(\Sigma \cup H)^{\mathcal{X}}$ is a homomorphism and $\gamma = \iota(\varphi)$, then $\varphi_{\text{io}}(s) = \{\gamma_{\infty}(s)\}$ is a singleton. ◀

► **Lemma 26.** Let $R \subseteq T(\Sigma)$ be a regular tree language and $\varphi : \mathcal{X} \rightarrow T_{\text{fin}}(\Sigma \cup H) \setminus H$ be a homomorphism to the set of finite trees. Let $\gamma = \iota(\varphi)$ be its canonically associated choice function as defined in (29). Then the set of trees

$$\varphi_{\text{io}}^{-1}(R) = \{s \in T(\Sigma \cup \mathcal{X}) \mid \gamma_{\infty}(s) \in R\}$$

is effectively regular.

Proof. We have $R = L(B, q_0)$ for some parity-NTA $B = (Q, \Sigma, \delta, \chi)$ and $q_0 \in Q$. Thus, it is enough to show that the set $\{s \in T(\Sigma \cup \mathcal{X}) \mid \gamma_{\infty}(s) \in L(B, p)\}$ is effectively regular for all $p \in Q$. As usual, we assume that $\chi : Q \rightarrow \{1, \dots, |C|\}$ where $|C|$ is odd. Let us define another parity-ATA $A = (Q \times C, \Sigma \cup \mathcal{X}, \delta_A, \chi_A)$ (where $\chi_A(q, c) = \text{pr}_2(q, c) = c$) such that for all $p \in Q$ we have

$$\{s \in T(\Sigma \cup \mathcal{X}) \mid \gamma_{\infty}(s) \in L(B, p)\} = L(A, (p, \chi(p))). \quad (30)$$

The set δ_A is defined by the following transitions

$$((p, c), x, \bigvee_{\rho \in \text{Run}_{B_H}(\gamma(x), p)} \bigwedge_{i_j \in K_{\rho}} (i, (\rho(i_j), c_{\rho}(i_j)))) \quad (31)$$

The notation in (31) is as follows: $x \in \Sigma \cup \mathcal{X}$ with $\gamma(x) = x(1, \dots, \text{rk}(x))$ and

$$K_{\rho} = \{i_j \in \text{Pos}(\gamma(x)) \mid \exists i \in H : i_j \in \text{leaf}_i(\gamma(x))\} \quad (32)$$

Recall that for a tree $t \in T(\Sigma \cup H)$ and a run $\rho \in \text{Run}_{B_H}(t, p)$, the color $c_{\rho}(i_j)$ denotes the minimal color in the tree $\rho(t)$ which appears on the unique path from the root ε to the leaf i_j . If $\text{Run}_{B_H}(\gamma(x), p) = \emptyset$, then the disjunction is over the empty set, hence the transition in

(31) becomes (p, x, false) . If $\text{Run}_{B_H}(\gamma(x), p) \neq \emptyset$ but $\gamma(x)$ is a tree without holes, then the transition in (31) becomes (p, x, true) .

Using Thm. 24 it is enough to show that for all $p \in Q$ we have $\gamma_\infty(s) \in L(B, p)$ if and only if Prover P_0 has a winning strategy in the associated parity game for the arena $G(A, s)$ at vertex $(\varepsilon, (p, \chi(p)))$ where A is the parity-ATA above. Note that the index set K_ρ allows Spoiler to pick any hole which appears in $\gamma(x)$.

One direction is easy. If $\gamma_\infty(s) \in L(B, p)$, then there exists an accepting run $\rho_0 \in \text{Run}_B(\gamma_\infty(s), p)$ and Prover can react to all choices of Spoiler by choosing $\rho = \rho_{x,p}$ in the disjunction in (31) as parts of the global run ρ . Note that due to the second component $c_\rho(i_j)$ in the states, every infinite game α corresponds to a unique infinite directed path in $\rho_0(\gamma_\infty(s))$ such that the minimal color occurring infinitely often on that infinite path is the same color as the minimal color occurring infinitely often in game α . Prover's winning strategy can also be explained by looking at the right tree depicted in Fig. 8. Prover begins by choosing the partial run of ρ_0 on the top triangle. Spoiler has no other option than to choose some leaf i_j . This leads to the second triangle, and Prover chooses again the partial run according to ρ_0 etc. Since $\gamma_\infty(s) \in L(B, p)$, Spoiler cannot win.

For the other direction, let us assume that Prover has a winning strategy in the arena $G(A, s)$ at vertex $(\varepsilon, (p, \chi(p)))$. Then Prover P_0 has a positional winning strategy by [17].

We will show that this winning strategy of P_0 defines an accepting run $\rho_0 \in \text{Run}_B(\gamma_\infty(s), p)$. The positional strategy chosen by Prover P_0 defines for each vertex belonging to P_0 at most one outgoing edge, all other outgoing edges at that vertex are deleted from the arena. After the deletion of edges, Prover is released and without any further interaction in the game. The game becomes a solitaire game where the outcome depends only on the choices of Spoiler.

Henceforth, all games are defined in a subarena $G'(A, s)$ where all games at vertex $(\varepsilon, (p, \chi(p)))$ are won by Prover. Moreover, we may assume that all vertices in $G'(A, s)$ are reachable from the vertex $(\varepsilon, (p, \chi(p)))$. Thus, without restriction, every vertex in $G'(A, s)$ belonging to Prover has exactly one outgoing edge. In particular, since all nodes are reachable from $(\varepsilon, (p, \chi(p)))$, all finite games starting at any vertex in $G'(A, s)$ are won by Prover, and if α is any directed infinite path starting at any vertex in $G'(A, s)$, then the minimal color occurring infinitely often at vertices of α is even.

Whenever a game reaches a vertex $(u, (q, c))$ via a finite directed path in $G'(A, s)$ starting at $(\varepsilon, (p, \chi(p)))$, then $\text{Run}_{B_H}(\gamma(s(u)), q) \neq \emptyset$, and the unique outgoing edge defines a run $\rho \in \text{Run}_{B_H}(\gamma(s(u)), q) \neq \emptyset$ which depends on (u, q, c) , only. For better reading we denote the run ρ as $\rho(u, q, c)$, too¹⁵. Note that, if $(q, c) = (\rho(i_j), c_\rho(i_j))$, then the run $\rho(u, q, c)$ does not reveal the position of the leaf i_j , in general. That implies that P_0 chose $\rho(u, q, c)$ without knowing i or i_j , in general. The outgoing edge (which defines ρ) ends at the vertex $(u, (q, c), \rho)$ belonging to spoiler.

In the following we assume without restriction that all games α start at vertices belonging to P_0 . Moreover, if α is finite, then α ends in a vertex belonging to Spoiler.

Our aim is construct an accepting run of $\gamma_\infty(s)$. The run is constructed top-down by induction. We use the following notation. Let $u \in \text{Pos}(s)$. We say that $\rho \in \text{Run}_{B_H}(\gamma(s(u)), q)$ is a *partial run* of the subtree $\gamma_\infty(s|_u)$ if the run labels a prefix closed subset of positions in $\gamma_\infty(s|_u)$ with states. The partial run labels the root with q . (Recall that $(s|_u)$ denotes the subtree of s rooted at u .) A *decision sequence* for Spoiler is a finite or infinite sequence $g(\alpha) = (g_1, g_2, \dots)$ where each g_ℓ is of the form $g_\ell = (h_\ell)_{j_\ell}$ with $h_\ell \in H$ for all $\ell \geq 1$ such that $g(\alpha)$ records all choices of Spoiler in the game α . The full information about the solitaire game α in $G'(A, s)$ starting at $(\varepsilon, (p, \chi(p)))$ is encoded in the sequence $g(\alpha)$.

¹⁵ Thus, the same letter ρ denotes a function or the specific value of that function. The context makes clear what we mean.

Let $\mathcal{D}(s)$ be set of all decision sequences of Spoiler. In the next step, we define four items for every finite prefix $g = (g_1, \dots, g_k)$ of a decision sequence $g(\alpha)$ in $\mathcal{D}(s)$.

- A path w_g in $\gamma_\infty(s)$. The other items are defined through the path w_g .
- A position $v(w_g)$ in $\gamma_\infty(s)$.
- A “terminal” position $\mathbf{o}(w_g) = (u, (q, c))$ in $V_0 = \text{Pos}(s) \times (Q \times C)$ such that the tree $\gamma(s(u))$ appears as an IO-prefix of the subtree $\gamma_\infty(s|_u)$.
- For $\mathbf{o}(w_g) = (u, (q, c))$ a partial run $\rho_g \in \text{Run}_{B_H}(\gamma(s(u)), q)$ of the subtree $\gamma_\infty(s|_u)$.

If g is the empty sequence, then no choice of Spoiler has been recorded. We let w_g be the empty path, $v(w_g) = \varepsilon \in \text{Pos}(\gamma_\infty(s))$, and $\mathbf{o}(w_g) = (\varepsilon, (p, \chi(p))) \in V_0$. The game α starts at that vertex $(\varepsilon, (p, \chi(p)))$ belonging to P_0 . The game defines a unique run $\rho_1 \in \text{Run}_{B_H}(\gamma(s(\varepsilon)), p)$. If no hole appears in $\gamma(s(\varepsilon))$, then $\gamma_\infty(s)$ is equal to $\gamma(s(\varepsilon))$. We are done in this case: knowing $\rho_\varepsilon = \rho_1$, all four items are defined if g is the empty sequence.

In the other case there exist some $i \in H$ and $i_j \in \text{leaf}_i(\gamma(s(u)))$. Thus, the game α is not finished; and Spoiler decides on some leaf labeled by a hole. Let $g' = (g_1, \dots, g_{k+1})$ be corresponding prefix. By induction, w_g is defined for $g = (g_1, \dots, g_k)$. Let $\mathbf{o}(w_g) = (u, (q, c))$, and $x = s(u)$. Hence, there is a tree $t_u = \gamma(x)$, and, by induction, t_u is an IO-prefix of the subtree $\gamma_\infty(s)|_v$ of $\gamma_\infty(s)$ rooted at the position $v = v(w_g)$ in $\gamma_\infty(s)$. Note that $u \in \text{Pos}(s)$ and $v \in \text{Pos}(\gamma_\infty(s))$. In particular, since t_u is an IO-prefix, every position v in t_u can be identified with a unique position $\text{pos}(v) \in \text{Pos}(\gamma_\infty(s))$ by the endpoint of the path $w_g.w_u(v)$. Here, $w_u(v)$ denotes the unique directed path in t_u from the root to the position v in t_u . The choice g_{k+1} of spoiler defines a position $i_j \in \text{leaf}_i(t_u)$ for some $i \in [\text{rk}(x)]$. We let $w_{g'}$ be the unique path which starts at the the root has w_g as prefix and stops in $\text{pos}(i_j) \in \text{Pos}(\gamma_\infty(s))$. Then we define $v(w_{g'}) = \text{pos}(i_j)$. It is the endpoint of the path $w_{g'}$. The unique outgoing edge at $\mathbf{o}(w_g) = (u, (q, c))$ defines a run $\rho_{g'} \in \text{Run}_{B_H}(\gamma(s(u)), q)$. Since g_{k+1} is defined, the position $u.i \in \text{Pos}(s)$ exists. and we let $\mathbf{o}(w_{g'}) = (u.i, (\rho'(i_j), c_{\rho'}(i_j)))$ where $\rho' = \rho_{g'}$. Note that the tree $\gamma(s(u.i))$ appears as an IO-prefix of the subtree $\gamma_\infty(s|_{u.i})$. Thus, all desired items are defined for the sequence g' .

See Fig. 7 how a finite directed path w in $\gamma_\infty^s(s)$ starting at the root defines a corresponding path in s . The figure also shows that w defines a unique position in $v(w) \in \text{Pos}(\gamma_\infty(s))$ via the arrows which start in $\gamma_\infty(s)$, leave the tree on the left, and make a clockwise turn upwards to enter the tree from the right. This visualizes the formal definition of a path w_g and its position $v(w_g)$ in $\gamma_\infty(s)$.

Suppose that $\mathbf{o}(w) = (u, (q, c))$ and $\mathbf{o}(w') = (u, (q', c'))$ where $w = w_g$ and $w' = w_{g'}$ are finite prefixes of decision sequences for Spoiler. We claim that if $(q, c) \neq (q', c')$, then the positions of $v(w)$ and $v(w')$ are incomparable. That is, there is no directed path between them. Indeed, if $(q, c) = (\rho(i_j), c_\rho(i_j))$ and $(q', c') = (\rho(i'_{j'}), c_\rho(i'_{j'}))$, then we must have $i'_{j'} \neq i_j$. The claim follows by induction on $|w|$. Details are left to the reader. Thus, the runs $\rho(u, q, c) \in \text{Run}_{B_H}(\gamma(s(u)), q)$ and $\rho(u, q', c') \in \text{Run}_{B_H}(\gamma(s(u)), q)$ (chosen by P_0 in his positional strategy) might be different, for example because $c \neq c'$. Since the positions $v(w)$ and $v(w')$ are incomparable, we can use $\rho(u, q, c)$ to define a partial run of the subtree rooted at $v(w) \in \text{Pos}(\gamma_\infty(s))$ and we can use $\rho(u, q', c')$ to define a partial run of the subtree rooted at $v(w')$.

Let $n \in \mathbb{N}$. Consider all finite directed paths w in $\gamma_\infty(s)$ beginning at the root such that $|w| \leq n$, and among them those which are induced by finite prefixes of decision sequences g of Spoiler. Let call them w_g . Then the the runs ρ_g constructed above for w_g yield partial runs $\tilde{\rho}_n$ of $\gamma_\infty(s)$. Note that these partial runs are compatible: for every common positions of two such partial runs, the labels (being states) are the same. This follows because $w_g = w_{g'}$ implies $\rho_g(v(w_g)) = \rho_{g'}(v(w_{g'}))$. Moreover, the sequence $(\tilde{\rho}_n)_{n \in \mathbb{N}}$ has a well-defined limit $\rho_0 = \lim_{n \rightarrow \infty} \tilde{\rho}_n$. Indeed, for every $k \geq 0$ there is some n_k such that $\tilde{\rho}_n(v) = \tilde{\rho}_{n_k}(v)$ for all $n \geq n_k$ and all positions $v \in \text{Pos}(\gamma_\infty(s))$ having a distance less than k to the root.

Therefore, the set of all directed paths in the arena $G'(A, s)$ starting at $(\varepsilon, (p, \chi(p)))$ yields ρ_0 as a (totally defined) run $\rho_0 : \text{Pos}(\gamma_\infty) \rightarrow Q$ such that $\rho_0(\varepsilon) = p$. (We write ρ_0 because its definition depends on the positional winning strategy of P_0 .)

We have to show that the run ρ_0 is accepting. For that we have to consider all directed paths in the tree defined by ρ_0 . If a path ends at a leaf, then it is accepting because ρ_0 is a run. Every infinite path in ρ_0 is due to some infinite game α starting at $(\varepsilon, (p, \chi(p)))$. The game defines a decision sequence $g(\alpha)$ and the run ρ_0 labels all vertices on the unique infinite direct path β which visits all positions $v(w_{g_k})$ where g_k runs over all finite prefixes of g . By construction, the minimal color $c(\alpha)$ appearing infinitely often in the game α is the same color as occurring infinitely often on β . Since Prover wins all games, $c(\alpha)$ is even. Hence, the run ρ_0 is accepting. Thus, the assertion we were looking for. \blacktriangleleft

6 Main results: The inclusion problem into regular sets

Henceforth, if $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ is a substitution and $R \subseteq T(\Sigma)$ is a subset, then we let

$$\sigma_{\text{io}}^{-1}(R) = \{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq R\}. \quad (33)$$

Eq.(33) extends the earlier notation used for homomorphisms; and following lemma generalizes Lem. 26 by switching from a homomorphism γ to a regular substitution σ .

► **Lemma 27.** *Let $R \subseteq T(\Sigma)$ be regular and $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be a regular substitution. Then the set $\sigma_{\text{io}}^{-1}(R) = \{s \in T(\Sigma \cup \mathcal{X}) \mid \sigma_{\text{io}}(s) \subseteq R\}$ is effectively regular.*

Proof. Let us begin to prove the lemma with the notation Σ' , R' , and σ' . Thus, formally, we begin with $R' \subseteq T(\Sigma')$ and $\sigma' : \mathcal{X} \rightarrow 2^{T(\Sigma' \cup H) \setminus H}$. The aim is to show that $\sigma'_{\text{io}}^{-1}(R') = \{s \in T(\Sigma' \cup \mathcal{X}) \mid \sigma'_{\text{io}}(s) \subseteq R'\}$ is effectively regular. For that we introduce a new constant a and we define a larger alphabet than Σ' by letting Σ to be the disjoint union of Σ' and $\{a\}$. We have $R' \cup \sigma'(x) \subseteq T(\Sigma' \cup H) \setminus H$ for all $x \in \mathcal{X}$. Define $\sigma(x) = \sigma'(x) \cup (T(\Sigma) \setminus T(\Sigma'))$ for all $x \in \mathcal{X}$ and $R = R' \cup (T(\Sigma) \setminus T(\Sigma')) \subseteq T(\Sigma)$. Then σ and R are regular. Moreover, R contains all trees which have a leaf labeled by a . For all $s \in T(\Sigma')$ we have

$$\sigma'_{\text{io}}(s) \subseteq R' \iff \sigma_{\text{io}}(s) \subseteq R \iff s \in \sigma_{\text{io}}^{-1}(R).$$

Moreover, if $\sigma_{\text{io}}^{-1}(R)$ is regular, then $\sigma_{\text{io}}^{-1}(R) \cap T(\Sigma')$ is regular, too. Thus, it is enough to prove the lemma under the assumptions that first, $\sigma(x) \neq \emptyset$ for all $x \in \mathcal{X}$ and second, there is a constant $a \in \Sigma$ such that $T(\Sigma) \setminus T(\Sigma \setminus \{a\}) \subseteq R$. Let $R = L(B, q_0)$ for some parity-NTA B . According to Def. 19 in Sec. 4.2 we embed the NTA B first into B_H and then B_H into the parity-NTA $B_{\mathcal{P}}$. We also embed $T(\Sigma)$ into $T(\Sigma_{\mathcal{P}})$ such that both, $R \subseteq L(B_{\mathcal{P}}, q_0)$ and for every profile $\pi = \pi(t) \in \mathcal{P}_B$ there is some finite tree $t_\pi \in T_{\text{fin}}(\Sigma_{\mathcal{P}})$ satisfying $\pi(t_\pi) = \pi$. Since every tree with a leaf labeled by a belongs to R , we may assume that $a \in L(B, p)$ for all states p . Hence, if $t \models \tau \in \mathcal{T}_B$, then $t[i_j \leftarrow a] \models \tau \in \mathcal{T}_B$, too.

In the following let $\tilde{R} = L(B_{\mathcal{P}}, q_0)$. Since $\sigma(x)$ is regular for all $x \in \mathcal{X}$ we can compute the specialization $\tilde{\sigma}$ of σ as defined in Def. 21. Using Eq.(26) in Prop. 22 we can state

$$\forall s \in T(\Sigma \cup \mathcal{X}) : \sigma_{\text{io}}(s) \subseteq R \iff \sigma_{\text{io}}(s) \subseteq \tilde{R} \iff \tilde{\sigma}_{\text{io}}(s) \subseteq \tilde{R} \quad (34)$$

Define a new set of variables by

$$\mathcal{X}' = \{(x, \pi) \in (\Sigma \cup \mathcal{X}) \times \mathcal{P}_B \mid \exists t \in \sigma(x) : \pi = \pi(t)\} \text{ with } \text{rk}(x, \pi) = \text{rk}(x). \quad (35)$$

Note that for all $f \in \Sigma$ there exists a variable $(f, \pi(f(1, \dots, \text{rk}(f)))) \in \mathcal{X}'$. That is why, below, it is enough to work with $T(\mathcal{X}')$ rather than with $T(\Sigma \cup \mathcal{X}')$. We use the second component of $(x, \pi) \in \mathcal{X}'$ to define a homomorphism $\gamma : \mathcal{X}' \rightarrow T_{\text{fin}}(\Sigma_{\mathcal{P}} \cup H) \setminus H$ by $\gamma(x, \pi) = t_\pi$ where

t_π was defined in Fig. 9. (Recall that t_π satisfies $\pi = \pi(t_\pi)$.) Then, by Eq.(25), for every $x \in X$ the following equation holds.

$$\check{\sigma}(x) = \{t_\pi \in T(\Sigma_{\mathcal{P}} \cup H) \mid \exists t \in \sigma(x) : \pi = \pi(t)\} = \{\gamma(x, \pi) \mid \exists t \in \sigma(x) : \pi = \pi(t)\}. \quad (36)$$

Since $\sigma(x) \neq \emptyset$, the projection onto the first component $(x, \pi) \mapsto x$ defines a surjective mapping $\psi_\infty : T(\mathcal{X}') \rightarrow T(\Sigma)$ by the homomorphism

$$\psi : \mathcal{X}' \rightarrow T(\Sigma \cup \mathcal{X} \cup H) \setminus H, \quad (x, \pi) \mapsto x(1, \dots, \text{rk}(x)). \quad (37)$$

A top-down induction shows $\text{Pos}(s') = \text{Pos}(\psi_\infty(s'))$ for all $s' \in T(\mathcal{X}')$. Also note that $\check{\sigma}_{\text{io}}(s) = \gamma_\infty(\psi_\infty^{-1}(s))$ for all $s \in T(\Sigma \cup \mathcal{X})$. Indeed, consider any $u \in \text{Pos}(s)$ with $x = s(u)$. Then $h^{-1}(x) = \{(x, \pi) \mid \exists t \in \sigma(x) : t \models \pi\}$. Hence, $\gamma(h^{-1}(x)) = \{t_\pi \mid \exists t \in \sigma(x) : t \models \pi\} = \check{\sigma}(x)$ where the last equation follows by Eq.(36).

Therefore,

$$\begin{aligned} \sigma_{\text{io}}^{-1}(R) &= \check{\sigma}_{\text{io}}^{-1}(\tilde{R}) && \text{by (34)} \\ &= \left\{ s \in T(\Sigma \cup \mathcal{X}) \mid \gamma_\infty(\psi_\infty^{-1}(s)) \subseteq \tilde{R} \right\} \\ &= \left\{ s \in T(\Sigma \cup \mathcal{X}) \mid \psi_\infty^{-1}(s) \subseteq \gamma_\infty^{-1}(\tilde{R}) \right\}. \end{aligned}$$

It enough to show that $T(\Sigma \cup \mathcal{X}) \setminus \sigma_{\text{io}}^{-1}(R) = \left\{ s \in T(\Sigma \cup \mathcal{X}) \mid \psi_\infty^{-1}(s) \cap \gamma_\infty^{-1}(\tilde{R}) \neq \emptyset \right\} = \psi_\infty(T(\mathcal{X}') \setminus \gamma_\infty^{-1}(\tilde{R}))$ is regular. The set $\gamma_\infty^{-1}(\tilde{R})$ is regular by Lem. 26. We can write $T(\Sigma \cup \mathcal{X}') \setminus \gamma_\infty^{-1}(\tilde{R}) = L(A', p_0)$ for a parity-NTA $A' = (Q, \Sigma \cup \mathcal{X}', \delta', \chi)$. It is therefore enough to construct a parity-NTA $A = (Q, \Sigma \cup \mathcal{X}, \delta, \chi)$ such that $L(A, p_0) = \psi_\infty(L(A', p_0))$. The construction of A is straightforward by using another transition relation. We define δ by the following equivalence where $x \in \Sigma \cup \mathcal{X}$:

$$(p, x, q_1, \dots, q_r) \in \delta \iff \exists \pi \in \mathcal{P}_B : (p, (x, \pi), q_1, \dots, q_r) \in \delta'. \quad (38)$$

The assertion follows. ◀

By a *class of tree languages* we mean a family of sets $\mathcal{C}(\Delta)$, indexed by all finite ranked alphabets Δ , such that first, we have $\mathcal{C}(\Delta) \subseteq T(\Delta)$ and second, for every rank-preserving inclusion $\varphi : \Delta \rightarrow \Gamma$, the tree-homomorphism defined by φ induces an inclusion $\mathcal{C}(\Delta) \rightarrow \mathcal{C}(\Gamma)$.

► **Definition 28.** We say that a class of tree languages \mathcal{C} satisfies the property **INREG**, if the following inclusion problem into regular sets is decidable: For every finite ranked alphabet Δ , on input $L \in \mathcal{C}(\Delta)$ (given in some effective way) and a regular tree language $K \subseteq T(\Delta)$ (given, say, by some parity-NTA) the problem “ $L \subseteq K$?” is decidable.

► **Theorem 29.** Let \mathcal{C} be a class of tree languages fulfilling the property **INREG** of Def. 28. Then the following decision problem is decidable.

- *Input:* Regular substitutions σ_1, σ_2 and tree languages $L \subseteq T(\Sigma \cup \mathcal{X})$, $R \subseteq T(\Sigma)$ such that R is regular and $L \in \mathcal{C}(\Sigma \cup \mathcal{X})$.
- *Question:* Is there some substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying both, $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1 \leq \sigma \leq \sigma_2$?

Moreover, we can effectively compute the set of maximal substitutions σ satisfying $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1 \leq \sigma(x) \leq \sigma_2$. It is a finite set of regular substitutions.

Proof. We let $R = L(B, q_0)$ for some parity NTA B . First, we check that $\sigma_1(x) \subseteq \sigma_2(x)$ for all $x \in \mathcal{X}$ because otherwise there is nothing to do. By definition, we have $\sigma_1(f) = \sigma_2(f) =$

$f(1, \dots, [\text{rk}(f)])$ for all $f \in \Sigma \setminus \mathcal{X}$. Thus, without restriction, $\sigma_1(x)$ and $\sigma_2(x)$ are defined as regular sets for all $x \in \Sigma \cup \mathcal{X}$ with $\sigma_1 \leq \sigma_2$. If there is any substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying both, $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1 \leq \sigma \leq \sigma_2$, then σ_1 is the unique **minimal** substitution satisfying that property. The substitution σ' defined by $\widehat{\sigma}_1(x) \cap \sigma_2(x)$ satisfies that property, too. It belongs to the following effectively computable finite set S_2 of regular substitutions

$$S_2 = \{\sigma' : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H} \mid \exists \sigma \forall x \in \mathcal{X} : \sigma_1(x) \subseteq \sigma'(x) = \widehat{\sigma}(x) \cap \sigma_2(x)\}. \quad (39)$$

The set S_2 is the same as defined above in (23) in the proof of Cor. 16. The notation in Eq.(39) refers substitutions $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ and their saturations $\widehat{\sigma}$ w.r.t. the NTA B . Thus, if the set S_2 is empty, then we can stop. The answer to the question in Thm. 29 is negative; and the set of maximal substitutions σ satisfying $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1 \leq \sigma(x) \leq \sigma_2$ is empty.

Thus, we may assume $S_2 \neq \emptyset$, which is computable by Cor. 16. Next, in a second phase we consider each element $\sigma \in S_2$, one after another. By Lem. 27 we know that for each $\sigma \in S_2$, the set $\sigma_{\text{io}}^{-1}(R)$ is an effectively regular set. The assertions $\sigma_{\text{io}}(L) \subseteq R$ and $L \subseteq \sigma_{\text{io}}^{-1}(R)$ are equivalent. Since $L \in \mathcal{C}$ we can check $L \subseteq \sigma_{\text{io}}^{-1}(R)$. Thus, we end up with an effectively computable subset S'_2 of S_2 such that there is some substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying both, $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1(x) \subseteq \sigma(x) \subseteq \sigma_2(x)$ for all $x \in \mathcal{X}'$ if and only if $S'_2 \neq \emptyset$. In case $S'_2 = \emptyset$ the answer to the question in Thm. 29 is negative, again. Hence, without $S'_2 \neq \emptyset$. The maximal substitutions σ satisfying both, $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma_1(x) \subseteq \sigma(x) \subseteq \sigma_2(x)$ for all $x \in \mathcal{X}'$ are in S'_2 . Since S'_2 is a nonempty, finite, and computable set of regular substitutions, we can compute all maximal element(s) in S'_2 . ◀

Note that, in the above theorem, \mathcal{C} can be chosen strictly larger than the class of all regular languages and still fulfill the hypothesis INREG. For example we can define $\mathcal{C}(\Delta)$ as consisting of all the regular languages in $T(\Delta)$ augmented with all context-free languages over $T_{\text{fin}}(\Delta)$ as defined in [7, 11, 16, 26].

► **Remark 30.** The hypothesis INREG cannot be removed in Thm. 29. For a counter-example we can choose the family \mathcal{C} such that $\mathcal{C}(\Delta)$ consists of all regular subsets of $T(\Delta)$ together with all the subsets $T(\Delta) \setminus K$ where K is a context-free subset of $T_{\text{fin}}(\Delta)$.

The inclusion problem: “ $R \subseteq L$?” for R regular and L context-free, reduces to the problem: $\exists \sigma : \mathcal{X} \rightarrow 2^{T(\Delta \cup H) \setminus H} : \sigma(T(\Delta) \setminus L) \subseteq (T(\Delta) \setminus R)$ where $\sigma_1 = \sigma_2 = H = \emptyset$. But the inclusion of a regular language into a context-free language is undecidable, showing that the substitution-inclusion problem for the class \mathcal{C} is undecidable. ◀

The following corollary considers the special case where we demand that substitutions map variables to nonempty subsets of trees. This setting is rather natural and it is also the setting when Conway studied the problem for finite words. Cor. 31 is a straightforward consequence of Thm. 29 and the fact that saturated solutions are regular.

► **Corollary 31.** *Let \mathcal{C} be a class of tree languages fulfilling the property INREG as in Thm. 29. Then the following decision problem is decidable.*

- *Input: Tree languages $L \subseteq T(\Sigma \cup \mathcal{X})$, $R \subseteq T(\Sigma)$ such that R is regular and $L \in \mathcal{C}(\Sigma \cup \mathcal{X})$.*
- *Question: Is there some substitution $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ satisfying both, $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma(x) \neq \emptyset$ for all $x \in \mathcal{X}$?*

Moreover, we can effectively compute the set of maximal substitutions σ satisfying $\sigma_{\text{io}}(L) \subseteq R$ and $\sigma(x) \neq \emptyset$ for all $x \in \mathcal{X}$. It is a finite set of regular substitutions.

Proof. By definition of regularity, we have $R = L(B, q_0)$ for some parity-NTA B . We run the following nondeterministic decision procedure. For each $x \in \mathcal{X}$ we guess a profile $\pi_x \in \mathcal{P}_B$. Then we check that there is some $t \in T(\Sigma \cup [\text{rk}(x)]) \setminus H$ such that $\pi_x = \pi(t)$. If there is no such t , then the assertion in the corollary has a negative answer for that guess. If there exists

such t , then we let $\sigma_1(x) = \{t \in T(\Sigma \cup [\text{rk}(x)]) \setminus H \mid \pi = \pi(t)\}$ and $\sigma_2(x) = T(\Sigma \cup [\text{rk}(x)]) \setminus H$. We have $\widehat{\sigma}_1 = \sigma_1$ and therefore σ_1 is regular by Prop. 15. We now apply Thm. 29. The assertion in the corollary is positive if and only if for some guess the application of Thm. 29 yields a positive answer with a set of maximal solutions. From these data we can compute the set of maximal solutions where $\sigma(x) \neq \emptyset$ for all $x \in \mathcal{X}$. Indeed, let $\sigma(x) \neq \emptyset$ for all $x \in \mathcal{X}$. Then there is some $t_x \in \sigma(x)$ and we can define $\pi_x = \pi(t_x)$. The non-deterministic procedure can guess the mapping $x \mapsto \pi_x$ and we obtain the maximal solutions for that guess. Simulating all guesses and collecting the maximal solutions for each guess yields the result. \blacktriangleleft

7 Conclusion and open problems

The main result of the paper is Thm. 29. We included the special case Cor. 31 because the assertion in the corollary corresponds exactly to the results for regular languages over finite words due to Conway as stated in the introduction, Sec. 1. The assertions of Thm. 29 and Cor. 31 are positive decidability results, but we don't know any (matching) lower and upper complexity bounds except for a few special cases.

Various other natural problems are open, too. For example, a remaining question is whether it is possible to derive positive results for the outside-in extension σ_{oi} . Another puzzling problem is that we don't know how to decide for regular tree languages L and R whether there exists a substitution σ such that $\sigma_{\text{io}}(L) = R$. We have seen that if such a substitution σ exists, then σ is in a finite and effectively computable set of regular substitutions. Thus, the underlying problem has no existential quantifier: decide “ $\sigma_{\text{io}}(L) = R$?” on input L, R , and σ where L, R , and σ are regular. If the answer is *yes*: $\sigma_{\text{io}}(L) = R$, then $\sigma_{\text{io}}(L)$ is regular. So, one could try to solve that problem. Recall that the problem is decidable in the setting of finite trees and homomorphisms by [9, 14].

Actually, we don't know how to decide the problem “ $\sigma_{\text{io}}(L) = T(\Sigma)$?” where $L \subseteq T(\mathcal{X})$ and $\sigma(x) = T(\Sigma \cup [\text{rk}(x)])$ for all $x \in \mathcal{X}$. Both problems “ $\exists \sigma : \sigma_{\text{io}}(L) = R$?” and “ $\exists \sigma : \sigma_{\text{oi}}(L) \subseteq R$?” remain open, even if we restrict ourselves to deal with finite trees, only.

It is also open whether better results are possible if we restrict R (or L and R) to smaller classes of regular tree languages like the class of languages with Büchi acceptance.

References

- 1 Sebastian Bala. Complexity of regular language matching and other decidable cases of the satisfiability problem for constraints between regular open terms. *Theory of Computing Systems*, 39(1):137–163, 2006.
- 2 Achim Blumensath. Regular tree algebras. *Logical Methods in Computer Science*, 16, February 2020. URL: <https://lmcs.episciences.org/6101>.
- 3 Mikołaj Bojańczyk. Recognisable languages over monads. *ArXiv e-prints*, abs/1502.04898, 2015. URL: <http://arxiv.org/abs/1502.04898>, arXiv:1502.04898.
- 4 Julius Richard Büchi. On a decision method in restricted second-order arithmetic. In *Proc. Int. Congr. for Logic, Methodology, and Philosophy of Science*, pages 1–11. Stanford Univ. Press, 1962.
- 5 Hubert Comon, Max Dauchet, Rémy Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. Tree automata techniques and applications, 2007. <http://www.grappa.univ-lille3.fr/tata>.
- 6 John Horton Conway. *Regular algebra and finite machines*. Chapman and Hall, London, 1971.
- 7 Bruno Courcelle. A representation of trees by languages. II. *Theoretical Computer Science*, 7(1):25–55, 1978. URL: [https://doi-org.docelec.u-bordeaux.fr/10.1016/0304-3975\(78\)90039-7](https://doi-org.docelec.u-bordeaux.fr/10.1016/0304-3975(78)90039-7), doi:10.1016/0304-3975(78)90039-7.

- 8 Bruno Courcelle. Fundamental properties of infinite trees. *Theoretical Computer Science*, 25:95–169, 1983. URL: [https://doi-org.docelec.u-bordeaux.fr/10.1016/0304-3975\(83\)90059-2](https://doi-org.docelec.u-bordeaux.fr/10.1016/0304-3975(83)90059-2), doi:10.1016/0304-3975(83)90059-2.
- 9 Carles Creus, Adrià Gascón, Guillem Godoy, and Lander Ramos. The HOM problem is EXPTIME-complete. *SIAM J. Comput.*, 45:1230–1260, 2016. URL: <https://doi.org/10.1137/140999104>, doi:10.1137/140999104.
- 10 Volker Diekert, Manfred Kufleitner, Gerhard Rosenberger, and Ulrich Hertrampf. *Discrete Algebraic Methods. Arithmetic, Cryptography, Automata and Groups*. Walter de Gruyter, 2016.
- 11 Joost Engelfriet and Erik Meineche Schmidt. IO and OI. I. *J. Comput. Syst. Sci.*, 15:328–353, 1977. URL: [https://doi.org/10.1016/S0022-0000\(77\)80034-2](https://doi.org/10.1016/S0022-0000(77)80034-2), doi:10.1016/S0022-0000(77)80034-2.
- 12 Joost Engelfriet and Erik Meineche Schmidt. IO and OI. II. *J. Comput. Syst. Sci.*, 16:67–99, 1978. URL: [https://doi.org/10.1016/0022-0000\(78\)90051-X](https://doi.org/10.1016/0022-0000(78)90051-X), doi:10.1016/0022-0000(78)90051-X.
- 13 Seymour Ginsburg and Thomas N. Hibbard. Solvability of machine mappings of regular sets to regular sets. *J. ACM*, 11:302–312, 1964. URL: <http://doi.acm.org/10.1145/321229.321234>, doi:10.1145/321229.321234.
- 14 Guillem Godoy and Omer Giménez. The HOM problem is decidable. *J. ACM*, 60:23:1–23:44, 2013. URL: <http://doi.acm.org/10.1145/2501600>, doi:10.1145/2501600.
- 15 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002. URL: <https://doi.org/10.1007/3-540-36387-4>, doi:10.1007/3-540-36387-4.
- 16 Irène Guessarian. Pushdown tree automata. *Mathematical Systems Theory*, 16(4):237–263, 1983. URL: <https://doi-org.docelec.u-bordeaux.fr/10.1007/BF01744582>, doi:10.1007/BF01744582.
- 17 Yuri Gurevich and Leo Harrington. Trees, automata, and games. In Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard, and Lawrence H. Landweber, editors, *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 60–65. ACM, 1982. URL: <https://doi.org/10.1145/800070.802177>, doi:10.1145/800070.802177.
- 18 Dexter Kozen. Lower bounds for natural proof systems. In *Proc. of the 18th Ann. Symp. on Foundations of Computer Science, FOCS'77*, pages 254–266, Providence, Rhode Island, 1977. IEEE Computer Society Press.
- 19 Michal Kunc. The power of commuting with finite sets of words. *Theory of Computing Systems*, 40:521–551, 2007.
- 20 Michal Kunc and Alexander Okhotin. Language equations. In Jean-Éric Pin, editor, *Handbook of Automata, Vol. I*, pages 765–800. EMS Publishing House, Berlin, 2021. Final version (2018) available at the homepage of Okhotin.
- 21 Saunders MacLane. *Categories for the Working Mathematician*. Springer-Verlag, New York, 1971. Graduate Texts in Mathematics, Vol. 5.
- 22 David E. Muller and Paul E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54:267–276, 1987. URL: [https://doi.org/10.1016/0304-3975\(87\)90133-2](https://doi.org/10.1016/0304-3975(87)90133-2), doi:10.1016/0304-3975(87)90133-2.
- 23 David E. Muller and Paul E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141:69–107, 1995. URL: [https://doi.org/10.1016/0304-3975\(94\)00214-4](https://doi.org/10.1016/0304-3975(94)00214-4), doi:10.1016/0304-3975(94)00214-4.
- 24 Christophe Prieur, Christian Choffrut, and Michel Latteux. Constructing sequential bijections. In *Structures in logic and computer science*, volume 1261 of *Lecture Notes in Computer Science*, pages 308–321. Springer, Berlin, 1997. URL: https://doi-org.docelec.u-bordeaux.fr/10.1007/3-540-63246-8_19, doi:10.1007/3-540-63246-8_19.
- 25 Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

- 26 Karl M. Schimpf and Jean H. Gallier. Tree pushdown automata. *Journal of Computer and System Sciences*, 30(1):25–40, 1985. URL: [https://doi-org.docelec.u-bordeaux.fr/10.1016/0022-0000\(85\)90002-9](https://doi-org.docelec.u-bordeaux.fr/10.1016/0022-0000(85)90002-9), doi:10.1016/0022-0000(85)90002-9.
- 27 Wolfgang Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, chapter 4, pages 133–191. Elsevier Science Publishers B. V., 1990.
- 28 Wiesław Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998. URL: [https://doi.org/10.1016/S0304-3975\(98\)00009-7](https://doi.org/10.1016/S0304-3975(98)00009-7), doi:10.1016/S0304-3975(98)00009-7.

8 Appendix: some additional material

As noticed in the preamble to the present paper: it is not necessary to read anything in the appendix to understand the main results. L’annexe, c’est l’art pour l’art: ars gratia artis.

8.1 Conway’s result for finite and infinite words

This section is meant for readers who are interested to see proofs of our main results in the special (and simpler) case of finite and infinite words.

Let Σ and \mathcal{X} be finite alphabets and let Σ^∞ denote the set of finite and infinite words. That is $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ where Σ^ω is the set of infinite words. The aim is to give an essentially self-contained proof for (a generalization of) Conway’s result with respect to $\# \in \{\subseteq, =\}$ for subsets of Σ^∞ . For that we consider “regular constraints” in the spirit of Thm. 29. The proof for Σ^* is given with all details. Next, we explain that the same approach works for infinite words, too. However, we are more sketchy and our proof uses the well-known result that regular ω -languages form an effective Boolean algebra [4]. This means that on input ω -languages L_1, L_2 (for example, specified by Büchi automata) we can construct Büchi automata for Σ^ω , $L_1 \cup L_2$, and $\Sigma^\omega \setminus L_1$. There is a way to avoid an explicit complementation, for example by using ultimately period words as a witness to show that $L_1 \setminus L_2 \neq \emptyset$.

► **Proposition 32.** *Let σ_1, σ_2 be mappings from $\Sigma \cup \mathcal{X}$ to 2^{Σ^+} such that for $i = 1, 2$ first, $\sigma_i(a) = \{a\}$ for every $a \in \Sigma \setminus \mathcal{X}$ and second, $\sigma_i(x)$ is a regular language for every $x \in \mathcal{X}$. Then for regular languages $L \subseteq (\Sigma \cup \mathcal{X})^\infty$ and $R \subseteq \Sigma^\infty$ the following assertions hold for $\# \in \{\subseteq, =\}$.*

1. *The following decision problem is decidable.*

$$\text{“}\exists \sigma : \sigma(L) \# R \wedge \forall x : \sigma_1(x) \subseteq \sigma(x) \subseteq \sigma_2(x)\text{?”} \quad (40)$$

2. *Define $\sigma \leq \sigma'$ by $\sigma(x) \subseteq \sigma'(x)$ for all $x \in \mathcal{X}$. Then every solution σ of (40) is bounded from above by a maximal solution; and the number of maximal solutions is finite. Moreover, if σ is maximal, then $\sigma(x)$ is regular for all $x \in \mathcal{X}$; and all maximal solutions are effectively computable.*

Recall that a solution $\tilde{\sigma}$ of (40) is maximal and above a solution σ if and only if first, $\sigma \leq \tilde{\sigma}$ and second, if σ' is any solution of (40) with $\tilde{\sigma} \leq \sigma'$, then $\tilde{\sigma} = \sigma'$.

Proof. Clearly, we may assume without restriction that $\sigma_1(x) \subseteq \sigma_2(x)$ for all x because otherwise there are no solutions.

Finite words: The easiest situation is the original setting of Conway: $L \subseteq (\Sigma \cup \mathcal{X})^*$ is arbitrary and $R \subseteq \Sigma^*$ is regular. In that case let $B = (Q, \Sigma, \delta, I, F)$ be an NFA such that $\delta \subseteq Q \times \Sigma \times Q$ and $R = L(B)$. Without restriction, $Q = \{1, \dots, n\}$ with $n \geq 1$. For $p, q \in Q$ we denote by $L[p, q]$ the set of words accepted by the NFA $B_{p,q} = (Q, \Sigma, \delta, \{p\}, \{q\})$. Next, we consider the semiring of Boolean $n \times n$ matrices $\mathbb{B}^{n \times n}$ where, as usual, $\mathbb{B} = (\{0, 1\}, \max, \min)$. We use the multiplication of matrices to define $\mathbb{B}^{n \times n}$ as a monoid where the unit matrix is the

neutral element. For a letter $a \in \Sigma$ we denote by M_a the matrix such that for all p, q we have $M_a(p, q) = 1 \iff a \in L[p, q]$. Since Σ^* is a free monoid, the M_a 's define a homomorphism $\mu : \Sigma^* \rightarrow \mathbb{B}^{n \times n}$ such that for all $w \in \Sigma^*$ we have $M_w(p, q) = 1 \iff w \in L[p, q]$ where $M_w = \mu(w)$. We have $\mu^{-1}(\mu(R)) = R$. Indeed, for $u \in \Sigma^*$ let $[u]$ be the set of words v such that $\mu(u) = \mu(v)$. Then we have

$$R = \bigcup \{u \in L[p, q] \mid p \in I, q \in F\} = \bigcup_{p \in I, q \in F} \bigcup \{[u] \subseteq \Sigma^* \mid u \in L[p, q]\}. \quad (41)$$

The verification of (41) is straightforward from the definition of μ . The crucial observation is that the union in (41) is finite because the set $\{[u] \subseteq \Sigma^* \mid u \in \Sigma^*\}$ is finite. Its cardinality is less than $2^{2^{n^2}}$.

Final Steps. We are almost done with the proof for finite words. We need a few more steps. Let $\sigma : \mathcal{X} \rightarrow 2^{\Sigma^*}$ be any substitution. Define its *saturation* $\hat{\sigma} : \mathcal{X} \rightarrow 2^{\Sigma^*}$ by $\hat{\sigma}(x) = \bigcup \{[u] \subseteq \Sigma^* \mid u \in \sigma(x)\}$. Then $\sigma \leq \hat{\sigma}$, the set $\hat{\sigma}(x)$ is regular (being a finite union of regular sets), and if $\sigma(L) \# R$, then it holds $\hat{\sigma}(L) \# R$, too. Moreover, if σ is regular, then we can calculate $\hat{\sigma}$ because we can decide for all $m \in \mathbb{B}^{n \times n}$ and $x \in \mathcal{X}$ whether $\mu^{-1}(m) \subseteq \sigma(x)$. Let us call a substitution σ' be a *candidate* as soon as first, $\sigma_1(x) \subseteq \sigma'(x)$ for all $x \in \mathcal{X}$ and second, $u \in \sigma'(x)$ implies $[u] \subseteq \sigma'(x)$. Since $\{[u] \subseteq \Sigma^* \mid u \in \Sigma^*\}$ is an effective list of finitely many regular sets, the finite list \mathcal{C} of candidates is computable: there are at most $2^{2^{n^2}|\mathcal{X}|}$ candidates. Since we assume $\sigma_1 \leq \sigma_2$, we compute for each candidate $\sigma' \in \mathcal{C}$ another regular substitution $\tilde{\sigma}$ by $\tilde{\sigma}(x) = \sigma'(x) \cap \sigma_2(x)$. The point is that whenever the problem in (40) has any solution $\sigma : \mathcal{X} \rightarrow 2^{\Sigma^*}$, then there is a maximal solution of the form $\tilde{\sigma}$ which satisfies $\tilde{\sigma}(L) \# R$. The class of regular languages is closed under substitution of letters by regular sets. This is a standard exercise in formal language theory. Hence, $\tilde{\sigma}(L)$ is regular; and we can decide $\tilde{\sigma}(L) \# R$ for every $\tilde{\sigma}$. This is the same as to decide $\tilde{\sigma}(L) \subseteq R$. Thus, we have to decide $\tilde{\sigma}(L) \cap (\Sigma^* \setminus R) = \emptyset$. The test involves the complementation¹⁶ of R .

So, we end up with a nonempty list \mathcal{L} of substitutions $\tilde{\sigma}$ satisfying $\tilde{\sigma}(L) \# R$. The list \mathcal{L} is finite and effectively given. Hence, we can compute all maximal elements. We are done.

Infinite words: Let us show that (using [4]) the case of infinite words can be explained in a similar fashion. For simplicity we restrict ourselves to substitutions where $\sigma(x)$ is a non-empty subset of Σ^+ . (The other cases are not harder, but need more case distinctions.) The starting point are two ω -regular languages $L \subseteq (\Sigma \cup \mathcal{X})^\omega$, and $R \subseteq \Sigma^\omega$. We use the fact that every ω -regular language can be accepted by a nondeterministic Büchi automaton. The syntax of a Büchi automaton is as for an NFA: $B = (Q, \Sigma, \delta, I, F)$ where $\delta \subseteq Q \times \Sigma \times Q$.

A word $w \in \Sigma^\omega$ is accepted if there are $p \in I$ and $q \in F$ such that B allows an infinite path labeled by w which begins in p and visits the state q infinitely often.

Instead of working over the Booleans \mathbb{B} , we consider the three-element commutative idempotent semiring $S = (\{0, 1, 2\}, +, \cdot)$ where $+$ = max and $x \cdot y = 0$ if $x = 0$ or $y = 0$ and otherwise $x \cdot y = \max\{x, y\}$. Note that 0 is a zero and 1 is neutral in (S, \cdot) . Let us define for $Q = \{1, \dots, n\}$ and $a \in \Sigma$ the matrix $M_a \in S^{n \times n}$ by

$$M_a(p, q) = \begin{cases} 0 & \text{if } (p, a, q) \notin \delta, \\ 1 & \text{if } (p, a, q) \in \delta \text{ but } \{p, q\} \cap F = \emptyset, \\ 2 & \text{otherwise: if } (p, a, q) \in \delta \text{ and } \{p, q\} \cap F \neq \emptyset. \end{cases}$$

The multiplicative structure $(S^{n \times n}, \cdot)$ yields a finite monoid with 3^{n^2} elements. The matrices $M_a \in S^{n \times n}$ define a homomorphism $\mu : \Sigma^* \rightarrow S^{n \times n}$. For all $p, q \in Q$ and $w \in \Sigma^*$ the

¹⁶ It is here where our exposition uses for ω -regular languages Büchi's help.

interpretation for $M_w = \mu(w)$ is as follows. We have $M_w(p, q) \neq 0$ if and only if there is a path labeled by w from state p to q . Moreover, $M_w(p, q) = 2$ if and only if there is a path labeled by w from state p to q which visits a final state. Let $T \subseteq S^{n \times n}$ be any subset, then $\mu^{-1}(T)$ is a regular language of finite words: $S^{n \times n}$ is the state set of a (deterministic!) NFA accepting $\mu^{-1}(T)$. In the terminology of ω -languages: μ *strongly recognizes* every $L = L(Q, \Sigma, \delta, \{p\}, \{q\}) \subseteq \Sigma^\omega$ where $p, q \in Q$. Strong recognition means that if $u = x_0x_1 \cdots$ and $v = y_0y_1 \cdots$ are infinite sequences of finite words such that $\mu(x_i) = \mu(y_i)$ for all $i \in \mathbb{N}$, then $u \in L \iff v \in L$. This property is crucial in the following. The verification of this property is straightforward and left to the reader.

We proceed as in the case of finite words. We let $[u] = \{v \in \Sigma^+ \mid \mu(u) = \mu(v)\}$. Now, consider any substitution $\sigma : \mathcal{X} \rightarrow 2^{\Sigma^+} \setminus \emptyset$. Let us define its saturation $\hat{\sigma}$ by $\hat{\sigma}(x) = \bigcup \{[u] \in \Sigma^+ \mid u \in \sigma(x)\}$. The saturation $\hat{\sigma}$ is regular. Moreover, if σ itself is regular, then we can compute $\hat{\sigma}$ effectively. (Note that this involves sets of finite words, only.) The number of saturated substitutions is less than $2^{3^{n^2}|\mathcal{X}|}$. We can calculate a list all saturated substitutions.

We also use the fact that if $L \subseteq (\Sigma \cup \mathcal{X})$ is ω -regular and if $\sigma : \mathcal{X} \rightarrow 2^{\Sigma^+}$ is a regular substitution, then $\sigma(L)$ is effectively ω -regular. Again, this fact is rather easy to see and it does not rely on the results of [4]. It is well-known that a language L of infinite words is ω -regular if and only if L is a finite union of languages UV^ω where U and V are regular languages of nonempty finite words. This fact can be derived in essentially the same way as the corresponding statement for finite words showing that NFAs have the same expressive power as regular expressions. Now, we can use our knowledge on finite words: if $L = \bigcup UV^\omega$, then $\sigma(L) = \bigcup \sigma(U)\sigma(V)^\omega$. If σ is a regular substitution such that the empty word is not in any $\sigma(x)$, then $\sigma(L) \subseteq \Sigma^\omega$ is ω -regular because the class of regular subsets in Σ^* is closed under regular substitutions. More precisely, let $w = x_0x_1 \cdots \in L \subseteq (\Sigma \cup \mathcal{X})^\omega$ such that $\sigma(w) \in L(B)$. Then $\sigma(w)$ is the set of all ω -words which have a factorization $u = u_0u_1 \cdots$ where for each $i \in \mathbb{N}$ we have $u_i \in \sigma(x_i)$. Consider any ω -word $v = v_0v_1 \cdots \in \Sigma^\omega$ such that $v_i \in [u_i]$ for all $i \in \mathbb{N}$. Since μ it is strongly recognizable, as explained above, $\sigma(w) \subseteq R$ implies $v \in R$. As a consequence, if $\sigma(L) \# R$, then we have $\hat{\sigma}(L) \# R$, too.

We now have all ingredients together to argue as in paragraph above “**Final Steps**” for finite words. However, to make the statements effective we use the fact that the family of ω -regular languages is an effective Boolean algebra. It is in this part where (according to our approach) Büchi’s result in [4] enters the scene: the complement of an ω -regular language is effectively ω -regular. See also the corresponding footnote (16) above where we speak about finite words and where we point to complementation. ◀

For more details about regular languages over infinite words (in the spirit of this section) we refer to the textbook [10]. In particular, the book presents the equivalence between ω -regular expressions and Büchi automata. It also shows how Büchi used a simple argument from Ramsey theory to derive his results in [4]. Of course, other textbooks or survey papers than [10] do the same.

8.2 About the existence of maximal solutions

The aim of this subsection is to show that the *regularity* hypothesis on $R \subseteq T(\Sigma)$ cannot be removed from our main results. This is done in Ex. 34. The example uses the special case of infinite words, only.

Remember that $T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$ (resp. $T_{H\text{-fin}}(\Sigma \cup H)$) denote the set of trees with only a finite number of occurrences of variables (resp. holes). Moreover, in Sec. 2.1 we used induction on the maximal level of a variable and Eq.(5) (resp. Eq.(6)) to define $\sigma_{\text{io}}(s)$ (resp. $\sigma_{\text{oi}}(s)$) for $s \in T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$ (with the additional exceptional case of a term without variable). Such

inductions can be viewed, as inductions on the following notion of *norm* of a term

$$\|s\| = \sup\{|u| + 1 \mid u \in \text{pos}(s) \wedge s(u) \in \mathcal{X}\}. \quad (42)$$

► **Proposition 33.** *Let $L \subseteq T_{\mathcal{X}\text{-fin}}(\Sigma \cup \mathcal{X})$ and $R \subseteq T(\Sigma)$ be arbitrary subsets, $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ be a substitution, and $\# \in \{\subseteq, =\}$. Then the following holds.*

- *There exists a maximal substitution σ' such that $\sigma \leq \sigma'$ and $\sigma'_{\text{io}}(L) \# R$.*
- *If, in addition, $\sigma(x) \subseteq T_{H\text{-fin}}(\Sigma \cup H)$, then exists a maximal substitution σ' such that $\sigma \leq \sigma'$ and $\sigma'_{\text{oi}}(L) \# R$, too.*

Proof. For the proof let $e \in \{\text{io}, \text{oi}\}$ one of the two possible extensions. It is enough to see that \leq is an *inductive ordering* on the set of substitutions $\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ such that $\sigma_e(L) \# R$ because then, every solution is bounded from above by some maximal solution thanks to Zorn's lemma. For that we have to consider nonempty totally ordered subsets

$$\{\sigma^{(k)} : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H} \mid k \in K\} \subseteq \{\sigma : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H} \mid \mathbb{P}(L) \# R\}.$$

Here, the index set K is totally ordered satisfying $\sigma^{(k)} \leq \sigma^{(\ell)} \iff k \leq \ell$. For $x \in \mathcal{X}$ we define $\sigma'(x) = \bigcup \{\sigma^{(k)} \mid k \in K\}$. This yields a substitution $\sigma' : \mathcal{X} \rightarrow 2^{T(\Sigma \cup H) \setminus H}$ such that $\sigma^{(k)} \leq \sigma'$. We show by induction on $\|s\|$ that if $s \in L$ and $t \in \sigma'(s)$, then $t \in \sigma^{(k)}(s)$ for some $k \in K$. For that we fix any index $k_0 \in K$. Now, let $s = x(s_1, \dots, s_r) \in L$ and $t = t_x[i_j \leftarrow t_i] \in \sigma'_{\text{io}}(s)$ (resp. $t = t_x[i_j \leftarrow t_{i_j}] \in \sigma'_{\text{oi}}(s)$) such that $t_x \in \sigma'(x)$ and $t_i \in \sigma'_{\text{io}}(s_i)$ (resp. $t_{i_j} \in \sigma'_{\text{oi}}(s_i)$). For $\|s\| = 0$ and both possibilities $e \in \{\text{io}, \text{oi}\}$, we have $\sigma_e^{(k_0)}(s) = \sigma'_e(s) = s$. We are done for $\|s\| = 0$. In the other case, there are indices k_i (resp. k_{ij}) by induction such that $t_i \in \sigma_{\text{io}}^{(k_i)}(s_i)$ (resp. $t_{i_j} \in \sigma_{\text{oi}}^{(k_{ij})}(s_i)$) for all $1 \leq i \leq \text{rk}(x)$. (This is true for $\text{rk}(x) = 0$.) Since $t_x \in \sigma'(x)$, there is some index k_x such that $t_x \in \sigma^{(k_x)}(x)$. Let $k_t = \max\{k_x, k_i \mid 0 \leq i \leq \text{rk}(x)\}$ (resp. $k_t = \max\{k_x, k_{ij} \mid 0 \leq i \leq \text{rk}(x) \wedge i_j \in \text{leaf } t_x\}$). The maximum exists for both $e = \text{io}$ and $e = \text{oi}$. We have $k_t \geq k_0$ and $t \in \sigma_e^{(k_t)}(s)$: the induction step is achieved. Therefore, $\sigma'_e(s) \subseteq R$.

Finally, if $\sigma_e^{(k)}(L) = R$ for all $k \in K$, then for each $t \in R$ there is some $s \in L$ such that $t \in \sigma_e^{(k_t)}(s)$. This implies $R \subseteq \bigcup \{\sigma_e^{(k)}(s) \mid s \in L \wedge k \in K\} = \bigcup \{\sigma'_e(s) \mid s \in L\} \subseteq R$. Hence, $\bigcup \{\sigma'_e(s) \mid s \in L\} = R$: every solution is bounded from above by some maximal solution. ◀

Ex. 34 shows that the statement of Prop. 33 might fail, when L contains some tree with infinitely many occurrences of some variable. We give such an example in the case where R is not regular and L is a set of ω -words. It fails for both extensions $\{\text{io}, \text{oi}\}$ and for both comparison relations $\# \in \{=, \subseteq\}$.

► **Example 34.** Consider $\Sigma = \{a, b\}$ and $\mathcal{X} = \{x\}$ where all three symbols have rank one. Hence, $H = \{1\}$ and an infinite tree over $\Sigma \cup \mathcal{X}$ encodes an infinite word in $(\Sigma \cup \mathcal{X})^\omega$ and vice versa. We let $L = L' \cup R$ where $L' = \{(ax)^\omega\}$ and $R = \{u \in \Sigma^\omega \mid \exists k \geq 1, b^k \text{ is no factor of } u\}$. The singleton L' is regular, but neither L nor R is not regular. Note that we have $\sigma_{\text{io}}(L) = \sigma_{\text{oi}}(L)$ for every substitution $\sigma(x) \subseteq T(\Sigma \cup H) \setminus H$. Clearly, $\sigma_{\text{io}}(L) = R \iff \sigma_{\text{io}}(L) \subseteq R \iff \sigma_{\text{io}}(L') \subseteq R$. There are many substitutions σ such that $\sigma_{\text{io}}(L') \subseteq R$. For example, let $\sigma(x) = a(1)$, then $\sigma_{\text{io}}(L') = a^\omega$. But there is no maximal substitution σ such that $\sigma_{\text{io}}(L) \subseteq R$. Indeed, given any substitution $\sigma(x) \subseteq T_{\text{fin}}(\{a, b, 1\}) \setminus H$ such that $\sigma_{\text{io}}(L') \subseteq R$, we can define $n = \max\{k \geq 0 \mid b^k \text{ is a factor of } \sigma(x)\}$. Then $\sigma'(x) = \sigma(x) \cup \{b^{n+1}(1)\}$ is strictly larger than σ , and it satisfies $\sigma'_{\text{io}}(L) \subseteq R$, too. For example, if $\sigma_{\text{io}}(L) = a^\omega$, then we obtain $\sigma'(x) = \{a(1), b(1)\}$ and $\sigma'_{\text{io}}(L) = (ab)^\omega$.

Comparing the two items in Prop. 33 we see that the “outside-in”-version for oi requires an additional assumption: every term $\sigma(x)$ has finitely many holes, only. The next example shows that we cannot drop that hypothesis.

► **Example 35.** Consider $\Sigma = \{f, a, b\}$ and $\mathcal{X} = \{x, y\}$ where $\text{rk}(f) = 2$, $\text{rk}(x) = \text{rk}(y) = \text{rk}(a) = 1$, and $\text{rk}(b) = 0$. We are interested in the finite term $x(y(b))$. Let $\sigma(x) = t$ where $t = f(t, 1)$ be infinite comb as in Fig. 3 and $\sigma(y) = a(b)$. The infinite tree $\sigma(x)$ is regular and the accepted language of a deterministic top-down tree automaton with a Büchi-acceptance condition as defined for example in [27]. The singleton $\sigma_{\text{oi}}(x(y(b)))$ is shown in Fig. 6 on the left with $n = 1$ and $\sigma(y) = a(b)$. Similar as in Ex. 34 we let

$$R = \{t \in T(\{f, a, b\}) \mid \exists k \geq 1, a^k \text{ is no factor of any path in } t\}.$$

Then $\sigma_{\text{oi}}(x(y(b))) \subseteq R$. However, there is no maximal substitution σ' such that $\sigma \leq \sigma'$ and $\sigma_{\text{oi}}(x(y(b))) \subseteq R$. Indeed, given any such σ' there is some $n \in \mathbb{N}$ such that $\sigma''(y) = \sigma'(y) \cup \{a^n(b)\}$ is strictly larger than σ' and σ'' still satisfies $\sigma''_{\text{oi}}(x(y(b))) \subseteq R$. If we define $\sigma''(y) = a^*(b)$, then $\sigma''_{\text{oi}}(x(y(b)))$ is regular, but it is not a subset of R .

8.3 The outside-in extension σ_{oi} for infinite trees

Although the definition of $\sigma_{\text{oi}}(s)$ for infinite trees is not used in the main body of the paper, we include a definition to have a reference for possible future research. Actually, the definition of $\sigma_{\text{oi}}(s)$ is technically simpler than the one of $\sigma_{\text{io}}(s)$ because no choice functions are used. Let us define $\sigma_{\text{oi}}(L)$ for a set of trees which can be finite or infinite. Without restriction, we content ourselves to define $\sigma_{\text{oi}}(L)$ for $S \subseteq T(\mathcal{X})$ where $\Sigma \cap \mathcal{X} = \emptyset$ and $\sigma(x) \subseteq T(\Sigma \cup [\text{rk}(x)]) \setminus H$ for all $x \in \mathcal{X}$. The first step reduces the problem to define $\sigma_{\text{oi}}(L)$ for a set L to the case where L is a singleton simply by letting, as expected, $\sigma_{\text{oi}}(L) = \bigcup \{\sigma_{\text{oi}}(s) \mid s \in L\}$. Thus, it is enough to define $\sigma_{\text{oi}}(s)$ for an infinite tree $s \in T(\mathcal{X})$ because for a finite tree s we employ the definition in Eq.(6). The idea is to use a nondeterministic program which transforms $s \in T(\mathcal{X})$ into a “hybrid” tree of $T(\mathcal{X} \cup \Sigma)$. During the process top-down more and more symbols appear in Σ and the subtrees (which are subtrees of s) appear at greater levels. Each run of the program defines at most one output. The set of all outputs over all runs defines the set $\sigma_{\text{oi}}(s)$. The program does not need to terminate, but if it runs forever, then, in the limit, it defines (nondeterministically) a tree $t \in T(\Sigma)$. The nondeterministic program is denoted as “ $\sigma_{\text{oi}}^{\text{nd}}$ ”. The input for the program is any tree $s \in T(\mathcal{X})$.

begin procedure

Initialize a tree variable $t := s \in T(\Sigma \cup \mathcal{X})$.

Perform the following while-loop as long as $t \notin T(\Sigma)$.

1. Choose in the breadth-first order on $\text{Pos}(t)$ the first position v such that $x = s(v) \in \mathcal{X}$.
2. Denote the subtree $t|_v$ rooted at v as $t|_v = x(t_1, \dots, t_r)$. This implies $\text{rk}(x) = r$.
3. Choose nondeterministically some $t_x \in \sigma(x)$.
If this is not possible because $\sigma(x) = \emptyset$, then EXIT without any output.
4. Replace in t the subtree $t|_v$ by $t_x[i_j \leftarrow t_i]$. Recall that this means to replace in t_x every $i_j \in \text{leaf}_i(t_x)$ by the same tree t_i where t_i is the i -th child of the root in $t|_v$. Moreover, if $\text{leaf}_i(t_x) \neq \emptyset$, then $1 \leq i \leq \text{rk}(x)$.

end procedure

Do not confuse the procedure with an inside-out extension. We visit positions in t labeled by a variable one after another and later choices of trees in $\sigma(t(v))$ are fully independent of each other. If the program terminates without using the EXIT branch, then we return the final tree t as the output $t = \sigma(\rho, s)$ of the specific nondeterministic run ρ . If the program runs forever, then let t_n be the value of t after performing the n -th loop of this run ρ . Since we always have $t_x \in T(\Sigma \cup H) \setminus H$, an easy reflection shows that there exists a unique infinite tree $\sigma(\rho, s) = \lim_{n \rightarrow \infty} t_n$. Moreover, $\sigma(\rho, s) \in T(\Sigma)$. We then define $\sigma_{\text{oi}}(s) = \{\sigma(\rho, s) \mid \rho \text{ run over } s\}$ and Eq.(6) holds.

8.4 Quotient metrics

This short subsection explains Footnote 7: our definition of a *quotient metric* agrees with the standard one in topology. Given a metric space (M, d) and any equivalence relation \sim on M , there is a canonical definition of a quotient (pseudo-)metric d_\sim on the quotient space M/\sim . For $x \in M$ let $[x] = \{x' \in M \mid x \sim x'\}$ denote its equivalence class. We associate to (M, d) a complete weighted graph with vertex set M/\sim and weight $g([x], [y]) = \inf \{d(x', y') \mid x \sim x' \wedge y \sim y'\}$. (So the weight might be 0 for $[x] \neq [y]$.) Then we define $d_\sim([x], [y])$ by the infimum over all weights of paths in the undirected graph connecting $[x]$ and $[y]$. The path can be arbitrarily long and still have weight 0. Clearly, d_\sim is a pseudo-metric (but not a metric, in general) satisfying

$$0 \leq d_\sim([x], [y]) \leq g([x], [y]) \leq d(x, y).$$

If for each $[x]$ there exists $x_0 \in [x]$ such that for all y we have

$$0 < g([x], [y]) = \inf \{d(x_0, y') \mid y \sim y'\}$$

then $g([x], [y]) = d_\sim([x], [y])$ is a metric. In our situation it is a metric: we have $(M, d) = (T(\Omega_\perp), d)$ and M/\sim results by identifying all trees $s \in (T(\Omega_\perp), d)$ where some position in $\text{Pos}(s)$ is labeled by \perp . To see this, just recall our definition of $d(s, s')$ for trees in $T(\Omega_\perp)$:

$$d(s, s') = \begin{cases} 1 & \text{if either } s \text{ or } s' \text{ uses the symbol } \perp \text{ but not both,} \\ 2^{-\inf\{|u| \in \mathbb{N} \mid u \in \mathbb{N}^* : s(u) \neq s'(u)\}} & \text{otherwise.} \end{cases}$$