# Deciding Piecewise Testable Separability for Regular Tree Languages*

**Jean Goubault-Larrecq[1] and Sylvain Schmitz[2]**

1 **LSV, ENS Cachan & CNRS & Inria, Université Paris-Saclay, Paris, France**
goubault@lsv.fr
2 **LSV, ENS Cachan & CNRS & Inria, Université Paris-Saclay, Paris, France**
schmitz@lsv.fr

## Abstract

The piecewise testable separability problem asks, given two input languages, whether there exists a piecewise testable language that contains the first input language and is disjoint from the second. We prove a general characterisation of piecewise testable separability on languages in a well-quasi-order, in terms of ideals of the ordering. This subsumes the known characterisations in the case of finite words. In the case of finite ranked trees ordered by homeomorphic embedding, we show using effective representations for tree ideals that it entails the decidability of piecewise testable separability when the input languages are regular. A final byproduct is a new proof of the decidability of whether an input regular language of ranked trees is piecewise testable, which was first shown in the unranked case by Bojańczyk, Segoufin, and Straubing [Log. Meth. in Comput. Sci., 8(3:26), 2012].

**1998 ACM Subject Classification** F.4.3 Formal Languages, F.4.1 Mathematical Logic

**Keywords and phrases** Well-quasi-order, ideal, tree languages, first-order logic

**Digital Object Identifier** 10.4230/LIPIcs.ICALP.2016.97

## 1 Introduction

The *separability* problem for a class $\mathcal{C}$ of input languages and a class $\mathcal{S}$ of separators, asks given two input languages $L$ and $L'$ from $\mathcal{C}$ whether there exists a language $S$ from $\mathcal{S}$ such that $L \subseteq S$ and $S \cap L' = \emptyset$. This classical problem has been studied in formal language theory since the 70's [27], but has sparked renewed interest due in particular to its connection with the *definability* problem: given $L$ from $\mathcal{C}$, does $L$ belong to $\mathcal{S}$? When $\mathcal{C}$ is effectively closed under complement, this last question reduces to the separability of $L$ and of its complement. Separability thus generalises definability, and has been instrumental in the recent advances on the definability problem for the alternation hierarchy over finite words [25].

We focus in this paper on the class of *piecewise testable* languages as class $\mathcal{S}$ of separators. Over finite words, this coincides with the languages defined by $\mathcal{B}\Sigma_1(<)$, the Boolean combinations of existential first-order sentences with order, and is one of the oldest and best-known classes of languages with decidable definability: Simon [26] showed that a language is piecewise-testable if and only if its syntactic monoid is $\mathcal{J}$*-trivial*. Lately, this has been extended in two different directions:

- Over finite unranked ordered trees, Bojańczyk, Segoufin, and Straubing [6] generalise Simon's algebraic approach and characterise the syntactic forest algebræ of piecewise

* A full version of this work is available online at https://hal.inria.fr/hal-01276119/.

43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016).
Editors: Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi;
Article No. 97; pp. 97:1–97:15


Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

testable tree languages defined by the most common signatures. This yields the decidability of the piecewise testable definability problem for regular unranked tree languages.

- Over finite words, Almeida and Zeitoun [2, 3] first proved the decidability of the PTL separability problem for regular input languages, based on a topological characterisation of separability. This abstract characterisation has been turned into a much more efficient characterisation in terms of patterns found simultaneously in $L$ and $L'$ [24, 11], which culminated in the work of Czerwiński, Martens, van Rooijen, Zeitoun, and Zetzsche [12] with a proof of the decidability of piecewise separability for many classes $\mathcal{C}$ of input languages, which even includes higher-order languages [17, 8, 4]. The crux of this proof is a reduction to the computation of representations for downward-closures [30] ordered by scattered subword embedding.

**Separability for Tree Languages.** Considering these two lines of research side by side begs the question whether piecewise testable separability might be decidable over richer structures than words, and in particular over trees. In this paper, we answer positively by proving:

▶ **Theorem 1.** *Piecewise testable separability is decidable for regular languages of ranked trees ordered by homeomorphic embedding.*

Since the class of regular tree languages is effectively closed under complement, Theorem 1 entails as a sub-problem the decidability of the corresponding definability problem:

▶ **Corollary 2.** *Piecewise testable definability is decidable for regular languages of ranked trees ordered by homeomorphic embedding.*

Thus, apart from the restriction to ranked trees, Theorem 1 also yields as a byproduct a new proof of decidability for the piecewise testable definability problem studied by Bojańczyk et al. [6], in the most challenging case of homeomorphic embeddings.

**Order-Theoretic Framework.** We employ vastly different techniques from Bojańczyk et al.'s, and Theorem 1 should be thought of as a proof of concept for our main contribution, which is a very general order-theoretic framework for piecewise testable separability:

- We characterise in Section 3 separability by piecewise testable languages over any combinatorial *well-quasi-order*; this encompasses for instance words ordered by scattered subword embedding and trees ordered by homeomorphic embedding or minor ordering. Our characterisation proceeds similarly to the topological characterisation of Almeida [2] by comparing the closures of $L$ and $L'$, but is stated in purely order-theoretic terms and compares the *ideals* – i.e. the irreducible downwards-closed sets – adherent to $L$ and $L'$. The advantage of this approach is that it scales to more complex structures than words – we have a toolkit of effective ideal representations at our disposal [13, 15] – and leads to a very simple and general proof, that delegates syntactic manipulations to the ideal representation. Our techniques encompass the word case: e.g. the patterns derived by Czerwiński et al. [12] turn out to be representations for word ideals [19, 1], and we eschew the technical work on patterns that absorbed a large part of Czerwiński et al.'s proof.
- We then show in Section 4 how to turn our characterisation into a generic decision procedure, by reduction to the *adherence membership* problem for the class $\mathcal{C}$. Provided we have effective representations for ideals, this yields directly the decidability for regular tree languages. We also derive the reduction of Czerwiński et al. to the computation of downward-closures in full generality.

- In order to instantiate this framework for tree languages, we provide in Section 5 our last ingredient: an effective representation for ideals of ranked trees ordered by homeomorphic embedding. This is a contribution of independent interest, which fixes an issue with the representation proposed in [13].

We start in Section 2 by defining piecewise testable languages in the general setting of combinatorial quasi-orders, and by recalling their logical characterisation. We assume the reader is already familiar with tree languages; see [9] for a general reference.

## 2 Piecewise Testable Languages

We work in the first part of this paper with *combinatorial* classes $X$ of elements, where $X$ is a countable set equipped with a *size* function $|.|: X \to \mathbb{N}$ such that $X_{\leq n} \stackrel{\text{def}}{=} \{x \in X \mid |x| \leq n\}$ is finite for every $n$. As soon as we equip $X$ with a quasi-ordering $\leq$, we can define a notion of piecewise-testable languages (see Section 2.1). The usual setting for piecewise testable languages is however that of classes of finite structures along with the embedding relation, in which case those sets also enjoy a logical characterisation (see Section 2.2).

### 2.1 Pieces

Two elements $x$ and $y$ are *$n$-piecewise equivalent*, noted $x \equiv_n y$, if for every $z \in X_{\leq n}$, $z \leq x$ if and only if $z \leq y$. A subset $S$ of $X$ is called *$n$-piecewise testable* (an $n$-PTL) over $(X, \leq)$ if $S$ is a union of $n$-piecewise equivalence classes. A subset $S$ of $X$ is *piecewise testable* (a PTL) if it is an $n$-PTL for some $n \geq 0$. This is well-known to be equivalent to the following formulation:

▶ **Fact 3.** *$S$ is an $n$-PTL over $(X, \leq)$ if and only if $S$ is a finite Boolean combination of* principal filters $\uparrow x \stackrel{\text{def}}{=} \{x' \in X \mid x \leq x'\}$ *where $x \in X_{\leq n}$.*

### 2.2 Logical Characterisation

Although the choice of the particular quasi-order $(X, \leq)$ is irrelevant to Fact 3 and the treatment in Section 3, one normally chooses $X$ to be a class $\mathcal{K}$ of finite structures over some signature $\sigma$ and $\leq$ to be the *induced substructure* ordering, which we denote by '$\sqsubseteq$'. When $(\mathcal{K}, \sqsubseteq)$ is a *well-quasi-order*, this leads to a well-known logical characterisation in terms of the Boolean closure of existential first-order formulæ– one of the chief motivations for studying piecewise testable languages. Other orderings may lead to different logical characterisations, or have no logical characterisation.

#### 2.2.1 Finite Structures

We consider finite relational structures $\mathfrak{M} = \langle M, (R_i)_{i \in I} \rangle$ over a finite signature $\sigma = (R_i)_{i \in I}$ (we conflate the names of relations in $\sigma$ with their interpretations in $\mathfrak{M}$). The finite set $M$ is the (potentially empty) domain of $\mathfrak{M}$ and each relation $R_i \subseteq M^{r_i}$ has a prescribed arity $r_i > 0$. The size $|\mathfrak{M}|$ of $\mathfrak{M}$ is the cardinality of its domain. A class $\mathcal{K}$ of structures is a set of such finite structures sharing the same signature $\sigma$ and closed under isomorphism.

▶ **Example 4** (Finite Words)**.** Let $\Sigma$ be a finite alphabet; we write $\Sigma^*$ for the set of finite words over $\Sigma$. Such a word $w$ can be seen as a relational structure over $\sigma_s \stackrel{\text{def}}{=} ((P_a)_{a \in \Sigma}, <)$ with its set of positions $\{1, \ldots, |w|\}$ as domain (where $|w|$ denotes the length of $w$, and thus coincides with its size), and $P_a(x)$ for $x \in \{1, \ldots, |w|\}$ holds if and only if the $x$th symbol of $w$ is $a \in \Sigma$, while $x < y$ has the obvious interpretation.

▶ **Example 5** (Ranked Trees). Fix a finite ranked alphabet $\mathcal{F}$, where each symbol $f$ is mapped to a single rank $r(f) \geq 0$, and write $\mathcal{F}_r$ for the subset of $\mathcal{F}$ with rank $r$. A (finite, ranked, ordered) tree is defined inductively as a term $f(t_1, \ldots, t_r)$ where $f \in \mathcal{F}_r$ and $t_1, \ldots, t_r$ are trees (see e.g. [9]). The notions of node, child, parent, descendant, and ancestor relations between nodes of a tree are defined as usual. We denote by $T(\mathcal{F})$ the set of trees over $\mathcal{F}$.

A tree can be seen as a structure with its set of nodes as domain – note that this domain is never empty by definition –; several signatures are then pertinent (see [6] and the full paper). We shall focus on the signature $\sigma_T \stackrel{\text{def}}{=} ((P_f)_{f \in \mathcal{F}}, <, <_{\text{dfs}}, \sqcap)$ where, for all nodes $x, y$,

- $P_f(x)$ holds whenever $x$ is labelled by $f$,
- $x < y$ whenever $x$ is an ancestor of $y$,
- $x <_{\text{dfs}} y$ whenever $x$ is visited before $y$ in a depth-first, left-first traversal of the tree – this is known as the *document order* –, and
- $z = x \sqcap y$ whenever $z$ is the *least* common ancestor of $x$ and $y$, i.e. $z$ is the unique descendant of all the nodes which are ancestors of both $x$ and $y$.

**Embeddings.** Recall that $\mathfrak{A} \sqsubseteq \mathfrak{B}$ holds between two structures $\mathfrak{A}$ and $\mathfrak{B}$ if and only if there exists an *embedding* from $\mathfrak{A}$ to $\mathfrak{B}$, i.e. an injective mapping $e$ from $A$ to $B$ that preserves the relations: for all $i \in I$ and $a_1, \ldots, a_{r_i} \in A$, $(a_1, \ldots, a_{r_i}) \in R_i$ in $\mathfrak{A}$ if and only if $(e(a_1), \ldots, e(a_{r_i})) \in R_i$ in $\mathfrak{B}$. As $\sqsubseteq$ is transitive and reflexive, $(\mathcal{K}, \sqsubseteq)$ is a *quasi-order* (a qo).

**Well-Quasi-Orders.** Given a qo $(X, \leq)$ and a subset $S \subseteq X$, the *downward-closure* of $S$ is $\downarrow S \stackrel{\text{def}}{=} \{x \in X \mid \exists s \in S \,.\, x \leq s\}$. A subset $D \subseteq X$ is *downwards-closed* if $D = \downarrow D$; when $D$ is a singleton $\{x\}$ we write more succinctly $\downarrow x$. The notions of *upward-closure* and *upwards-closed* subsets are defined similarly.

A qo $(X, \leq)$ is a *well-quasi-order* (wqo) if in every infinite sequence $x_0, x_1, \ldots$ of elements of $X$, one can find an infinite sequence of indices $i_0 < i_1 < \cdots$ such that $x_{i_0} \leq x_{i_1} \leq \cdots$ [18, 20]. Equivalently, $(X, \leq)$ is well-founded (there are no infinite descending sequences $x_0 > x_1 > \cdots$) and satisfies the *finite antichain condition* (FAC), i.e. all its antichains are finite. Still equivalently, $(X, \leq)$ has the *descending chain condition*: there are no infinite descending sequences $D_0 \supsetneq D_1 \supsetneq \cdots$ of downwards-closed subsets $D_i \subseteq X$. For instance, any finite set ordered by equality is a wqo by the Pigeonhole Principle. As another example, if $(X, \leq_X)$ and $(Y, \leq_Y)$ are two wqos, then by Dickson's Lemma, their Cartesian product $X \times Y$ is well-quasi-ordered by the *product ordering* defined by $(x, y) \leq (x', y')$ if and only if $x \leq_X x'$ and $y \leq_Y y'$.

▶ **Example 6** (Scattered Subword Embedding). The signature $\sigma_s$ of Example 4 for finite words in $\Sigma^*$ gives rise to an embedding relation $\sqsubseteq_s$ known as the (scattered) *subword embedding*: $u \sqsubseteq_s v$ if and only if $u = a_1 \cdots a_m$ and $v = v_0 a_1 v_1 \cdots v_{m-1} a_m v_m$ for some $a_1, \ldots, a_m \in \Sigma$ and $v_0, \ldots, v_m \in \Sigma^*$.

By Higman's Lemma [18], the subword embedding relation $\sqsubseteq_s$ well-quasi-orders $\Sigma^*$.

▶ **Example 7** (Homeomorphic Tree Embedding). Using the signature $\sigma_T$ of Example 5 for trees in $T(\mathcal{F})$, the ordering $t \sqsubseteq_T t'$ is better known as the *homeomorphic tree embedding* relation, and holds for $t = f(t_1, \ldots, t_r)$ and $t' = g(t'_1, \ldots, t'_s)$ if and only if

- there exists $1 \leq i \leq s$ such that $t \sqsubseteq_T t'_i$, or
- $f = g$ (and thus $r = s$) and for every $1 \leq i \leq r$, $t_i \sqsubseteq_T t'_i$;

see the full paper for a proof of this folklore result. Note that $(\Sigma^*, \sqsubseteq_s)$ is isomorphic to $(T(\mathcal{F}_\Sigma), \sqsubseteq_T)$ where $\mathcal{F}_\Sigma \stackrel{\text{def}}{=} \Sigma \uplus \{\$\}$ assigns rank 1 to letters in $\Sigma$ and rank 0 to \$, so our results on ranked trees properly generalise the case of finite words.

As first shown by Higman [18], the homeomorphic tree embedding relation $\sqsubseteq_T$ similarly well-quasi-orders $T(\mathcal{F})$, and Kruskal's Tree Theorem [20] shows that this generalises to unranked trees.

**Existential First-Order Logic.** Consider first-order logic over the signature $\sigma$, and a class of structures $\mathcal{K}$ over $\sigma$. Given a sentence $\varphi$, the set of structures $\mathfrak{M} \in \mathcal{K}$ such that $\mathfrak{M} \models \varphi$ is called the (first-order) *language* of $\varphi$. It is well-known that *existential sentences* in $\Sigma_1(\sigma)$ define *upwards-closed* first-order languages $S$ with respect to embeddings, i.e. such that $S = \uparrow S \stackrel{\text{def}}{=} \{\mathfrak{M}' \in \mathcal{K} \mid \exists \mathfrak{M} \in S . \mathfrak{M} \sqsubseteq \mathfrak{M}'\}$; however the converse does not necessarily hold [28].

We are interested here in the Boolean closure $\mathcal{B}\Sigma_1(\sigma)$ of $\Sigma_1(\sigma)$:

▶ **Fact 8.** *If a set $S$ is an $n$-PTL over $(\mathcal{K}, \sqsubseteq)$, then it is definable by a sentence in $\mathcal{B}\Sigma_1(\sigma)$ with at most $n$ variables. Conversely, assuming additionally that $(\mathcal{K}, \sqsubseteq)$ is a wqo, if $S$ is definable by a sentence in $\mathcal{B}\Sigma_1(\sigma)$, then it is a PTL over $(\mathcal{K}, \sqsubseteq)$.*

## 3 Ideal Characterisation of PTL Separability

A set $S \subseteq X$ *separates* $L \subseteq X$ from $L' \subseteq X$ if $L \subseteq S$ and $S \cap L' = \emptyset$. This can be turned into a decision problem when restricting $L$ and $L'$ to a class $\mathcal{C} \subseteq 2^X$ of finitely representable sets:

▶ **Problem** (PTL separability for $\mathcal{C}$ over $(X, \leq)$).
**Input:** *Representations of $L$ and $L'$ from $\mathcal{C}$*
**Question:** *Are $L$ and $L'$* PTL separable*, i.e. does there exist $S$ a PTL over $(X, \leq)$ that separates $L$ from $L'$?*

Throughout this section, we assume that $(X, \leq)$ is a well-quasi-order. Under this assumption, we establish in Section 3.2 a characterisation of PTL separability in terms of *ideals* of $(X, \leq)$ (whose definition we recall in Section 3.1). This yields a generic framework in which the decidability of PTL separability can be tackled, which we present in Sections 4 and 5 and instantiate for the case of $(T(\mathcal{F}), \sqsubseteq_T)$ the set of ranked trees together with homeomorphic embeddings against the class $\mathcal{C} = \mathrm{Reg}(T(\mathcal{F}))$ of regular tree languages.

### 3.1 Ideals

An *ideal* of a qo $(X, \leq)$ is a downwards-closed and (up-)*directed* subset $I \subseteq X$, where this last condition ensures that $I$ is non-empty and that, given any $x \in I$ and $y \in I$, there exists $z \in I$ with $x \leq z$ and $y \leq z$. A related notion is the following. A downwards-closed subset $D$ of $(X, \leq)$ is *irreducible* if and only if it is non-empty, and for any two downwards-closed subsets $D_1$, $D_2$ such that $D \subseteq D_1 \cup D_2$, $D$ is contained in $D_1$ or in $D_2$ already; equivalently, $D$ is non-empty and cannot be written as the union of two proper, downwards-closed subsets.

▶ **Fact 9** (cf. Lemma 1 in [7]). *The following are equivalent for a downwards-closed subset $D$ of a quasi-ordered set: (a) $D$ is an ideal; (b) $D$ is directed; (c) $D$ is irreducible.*

We write $\mathrm{Idl}(X)$ for the set of ideals of $X$. Ideals are especially useful when $(X, \leq)$ is a wqo: for one thing, when $X$ is countable, $\mathrm{Idl}(X)$ is then also countable [7, Theorem 1], and furthermore any downwards-closed subset has a decomposition as a finite union of ideals:

▶ **Fact 10** (cf. Lemma 2 in [7]). *A qo is FAC if and only if every downwards-closed subset is a finite union of ideals.*

## 3.2 Characterising Separability

Over any qo $(X, \leq)$, the case where $L$ and $L'$ are not PTL separable has a well-known characterisation in terms of sequences of indistinguishable witnesses, which is a straightforward consequence of the definitions:

▶ **Lemma 11.** *$L$ and $L'$ are* not *PTL separable over a qo $(X, \leq)$ if and only if there exist two sequences of elements $(x_n)_{n\in\mathbb{N}}$ in $L$ and $(x'_n)_{n\in\mathbb{N}}$ in $L'$ such that for every $n$, $x_n \equiv_n x'_n$.*

**Proof Idea.** It suffices to observe that $L$ and $L'$ are not $n$-PTL separable if and only if there exist $x_n \in L$ and $x'_n \in L'$ such that $x_n \equiv_n x'_n$. See the full paper for a detailed proof. ◀

When $(X, \leq)$ is a wqo, we have another characterisation in terms of directed subsets of $L$ and $L'$ defining the same ideal. The fact that $(X, \leq)$ is wqo is needed in order for Lemma 12 to hold: otherwise, the full paper presents a counter-example for the *subtree ordering* over $T(\mathcal{F})$.

▶ **Lemma 12** (Key Lemma)**.** *$L$ and $L'$ are not PTL separable over a wqo $(X, \leq)$ if and only if there exist two directed sets $\Delta \subseteq L$ and $\Delta' \subseteq L'$ such that $\downarrow\Delta = \downarrow\Delta'$.*

**Proof of 'if'.** Let $\Delta \subseteq L$ and $\Delta' \subseteq L'$ be two directed sets with $\downarrow\Delta = \downarrow\Delta'$. Let us show that for every $n \in \mathbb{N}$, there exist $x_n \in L$ and $x'_n \in L'$ such that $x_n \equiv_n x'_n$, from which Lemma 11 yields the result. Write $I$ for the ideal $\downarrow\Delta = \downarrow\Delta'$. Observe that, for every $n \in \mathbb{N}$, $I \cap X_{\leq n}$ is finite since $X$ is a combinatorial class. Furthermore, for every $z \in I \cap X_{\leq n}$, by definition of $I$ there exist $x_z \in \Delta$ with $z \leq x_z$ and $x'_z \in \Delta'$ with $z \leq x'_z$. Since $\Delta$ and $\Delta'$ are directed, we can find $x_n \in \Delta$ and $x'_n \in \Delta'$ greater or equal to all those finitely many $x_z$ and $x'_z$ when $z$ ranges over $I \cap X_{\leq n}$ (if $I \cap X_{\leq n} = \emptyset$ then any $x_n \in \Delta$ and $x'_n \in \Delta'$ fit). Then $x_n \equiv_n x'_n$, since for any $z \in X_{\leq n}$, either $z \in I$ and then both $z \leq x_z \leq x_n$ and $z \leq x'_z \leq x'_n$, or $z \notin I$ and then both $z \not\leq x_n$ and $z \not\leq x'_n$ since $I$ is downwards-closed. ◀

**Proof of 'only if'.** Assume $L$ and $L'$ are not PTL separable, hence by Lemma 11 there exist two infinite sequences $(x_n)_n$ in $L$ and $(x'_n)_n$ in $L'$ with $x_n \equiv_n x'_n$ for every $n$.

Let us consider the infinite sequence of *pairs* $(x_n, x'_n)_{n\in\mathbb{N}}$. By Dickson's Lemma, $X \times X$ is a wqo for the product ordering, hence there exists an infinite sequence of indices $i_0 < i_1 < \cdots$ such that $x_{i_j} \leq x_{i_{j+1}}$ and $x'_{i_j} \leq x'_{i_{j+1}}$ for every $j \in \mathbb{N}$. Define $\Delta \stackrel{\text{def}}{=} \{x_{i_j} \mid j \in \mathbb{N}\}$ and $\Delta' \stackrel{\text{def}}{=} \{x'_{i_j} \mid j \in \mathbb{N}\}$. These two sets are directed; they are actually infinite chains $x_{i_0} \leq x_{i_1} \leq \cdots$ and $x'_{i_0} \leq x'_{i_1} \leq \cdots$.

It remains to show that $\downarrow\Delta = \downarrow\Delta'$. By symmetry, it suffices to show $\Delta \subseteq \downarrow\Delta'$. Consider some $x_{i_j} \in \Delta$; then there exists some index $i_k > \max(i_j, |x_{i_j}|)$. Hence $x_{i_j} \leq x_{i_k}$, and since $x_{i_k} \equiv_{i_k} x'_{i_k}$, $x_{i_j} \leq x'_{i_k}$ and thus $x_{i_j} \in \downarrow\Delta'$. ◀

**Related Work over Finite Words.** Let $S$ be a subset of $X$. We define the *adherence* of $S$ as the set of ideals defined by the directed subsets of $S$:

$$\mathrm{Adh}(S) \stackrel{\text{def}}{=} \{\downarrow\Delta \in \mathrm{Idl}(X) \mid \Delta \subseteq S \text{ is directed } \}\,. \tag{1}$$

Lemma 12 can be restated as saying that $L$ and $L'$ are PTL separable if and only if their adherences are disjoint, i.e. $\mathrm{Adh}(L) \cap \mathrm{Adh}(L') = \emptyset$. This makes Lemma 12 quite reminiscent of several results on separability for word languages over $\Sigma^*$. Almeida [2] showed that two regular languages over $\Sigma$ are PTL separable over $(\Sigma^*, \sqsubseteq_s)$ if and only if their topological closures inside a specific profinite semigroup do not intersect. This lead to a first decision procedure [3] by explicitly constructing representations for these topological closures and

testing their intersection for emptiness – the counterpart in terms of Lemma 12 would be to compute the adherences of $L$ and $L'$.

Major improvements were brought by Place et al. [24] and Czerwiński et al. [11] and culminate in Theorem 2.1 of [12], by reducing this non-empty intersection check to identifying a common pattern 'densely' matched by both $L$ and $L'$. It turns out that these patterns are essentially finite representations for ideals of $(\Sigma^*, \sqsubseteq_s)$, so Lemma 12 subsumes Theorem 2.1 of Czerwiński et al. [12] – and has a considerably simpler proof.

# 4 Deciding PTL Separability

While Lemma 12 provides a general characterisation for PTL separability, turning it into a decision procedure requires finite representations and effectiveness assumptions on its various ingredients. We define a set of such assumptions in Section 4.1, which is sufficient to derive a generic algorithm. We describe the latter in two steps: we first show in Section 4.2 a reduction to the *adherence membership problem*. The final step in Section 4.3 is to show that this last problem is decidable for the set of regular tree languages over $(T(\mathcal{F}), \sqsubseteq_T)$.

## 4.1 Effectiveness Assumptions

In order to put Lemma 12 into practice, we need to consider in more details how we are going to represent PTLs over $(X, \leq)$, ideals in $\mathrm{Idl}(X)$, and languages in $\mathcal{C}$.

**PTLs.** Fact 3 provides a natural representation for PTLs as finite Boolean combinations of principal filters, i.e. more concretely as terms of the free Boolean algebra with elements of $X$ as atoms.

**Ideals.** Recall that over a countable wqo $(X, \leq)$, $\mathrm{Idl}(X)$ is also countable [7]. In Section 4.2, we only need to have explicit *ideal representations* as a means of enumerating ideals, while Corollary 15 further needs a means of computing representations as regular tree languages. Section 5 fulfils both requirements by providing ideal representations for $(T(\mathcal{F}), \sqsubseteq_T)$ as regular tree expressions.

**Languages.** Our last effectiveness assumptions regard the class of languages $\mathcal{C}$. We call $\mathcal{C}$ *PTL-effective* over a qo $(X, \leq)$ if $\mathcal{C}$ has *decidable emptiness*: given a representation for $L \in \mathcal{C}$, there is an algorithm that decides whether $L = \emptyset$, and if $\mathcal{C}$ is *effectively closed under intersection with PTLs*: given a representation for $L \in \mathcal{C}$ and one for $S$ a PTL over $(X, \leq)$, there is an algorithm that constructs a representation for $L \cap S$ in $\mathcal{C}$.

For instance, both over $(\Sigma^*, \sqsubseteq_s)$ and over $(T(\mathcal{F}), \sqsubseteq_T)$, a principal filter $\uparrow x$ is a regular language, and since regular languages are closed under Boolean operations by Fact 3 any PTL is regular. Thus, PTL-effective classes of word and tree languages abound, starting with regular languages themselves, but also context-free, etc.

## 4.2 Reducing to Adherence Membership

We describe our decision procedure in two steps. The first one reduces the PTL separability problem for a PTL-effective class $\mathcal{C}$ to the following problem:

▶ **Problem** (Adherence Membership for $\mathcal{C}$ over $(X, \leq)$).
**Input:** *A representation for* $L \in \mathcal{C}$ *and one for* $I \in \mathrm{Idl}(X)$.
**Question:** *Is* $I \in \mathrm{Adh}(L)$*?*

▶ **Proposition 13.** *Let $(X, \leq)$ be a wqo with ideal representations and $\mathcal{C} \subseteq 2^X$ be PTL-effective over $(X, \leq)$. Then there is a Turing reduction from the PTL separability problem to the adherence membership problem.*

**Proof.** Given an oracle for the adherence membership problem for $\mathcal{C}$ over $(X, \leq)$, our algorithm consists of two semi-decision procedures that take representations for $L \in \mathcal{C}$ and $L' \in \mathcal{C}$ as input. The first attempts to show that $L$ and $L'$ are PTL separable, and enumerates representations of PTLs $S$ until $L \cap (X \setminus S) = \emptyset$ and $L' \cap S = \emptyset$. This is possible because $\mathcal{C}$ is PTL-effective and complementing a PTL representation is trivial. The second relies on Lemma 12 and attempts to show that $L$ and $L'$ are not PTL separable by enumerating representations of ideals $I$ until $I \in \mathrm{Adh}(L)$ and $I \in \mathrm{Adh}(L')$, using the oracle for adherence membership for the tests. ◀

## 4.3 Deciding Adherence Membership

**Regular Languages.** In the case of regular tree languages over $T(\mathcal{F})$, the adherence membership problem is decidable thanks to the following lemma:

▶ **Lemma 14.** *Let $(X, \leq)$ be a qo and $L \subseteq X$. Then $I \in \mathrm{Adh}(L)$ if and only if $I \subseteq {\downarrow}(I \cap L)$.*

**Proof.** The 'only if' part is immediate or the 'if' part we show that $I \cap L$ is directed. Since $I$ is non-empty and included in ${\downarrow}(I \cap L)$, $I \cap L$ is also non-empty. Furthermore, if $x, y$ are in $I \cap L$, then since $I$ is directed there exists $z \in I$ such that $x \leq z$ and $y \leq z$, and since $I \subseteq {\downarrow}(I \cap L)$ there exists $z' \in I \cap L$ such that $z \leq z'$. ◀

Now, if $L$ is a regular tree language, $I$ is also effectively regular using the ideal representations from Section 5, and so are $I \cap L$ and ${\downarrow}(I \cap L)$, hence $I \subseteq {\downarrow}(I \cap L)$ is decidable since inclusion of regular tree languages is decidable, proving Theorem 1:

▶ **Corollary 15.** *Adherence membership is decidable for regular tree languages over $(T(\mathcal{F}), \sqsubseteq_T)$.*

**Generic Approach.** Let us finally generalise the previous idea. Our issue is that, for instance when $\mathcal{C}$ is the class of context-free languages over $\Sigma^*$, while $I$ is a regular language and ${\downarrow}(I \cap L)$ is a computable context-free language, the inclusion test between a regular language and a context-free one is in general undecidable. Thankfully, ${\downarrow}(I \cap L)$ is regular for arbitrary languages $L$ by Fact 10, since it is a finite union of ideals and ideals are regular. However, the issue is then to compute a representation of ${\downarrow}(I \cap L)$ *as a regular language*, or more generally to solve the following problem:

▶ **Problem** (Ideal Decomposition for $\mathcal{C}$ over $(X, \leq)$)**.**
**Input:** *A representation for $L \in \mathcal{C}$*
**Output:** ${\downarrow}L$ *as a finite union of representations of ideals in* $\mathrm{Idl}(X)$.

The following result requires effective operations on ideal representations. We refer the reader to [13, 15] for a systematic study of algorithms on finite ideal representations for wqos; here we use the notion of *effective* ideal representations as defined by Goubault-Larrecq et al. [15] to prove (see the full paper):

▶ **Proposition 16.** *Let $(X, \leq)$ be a wqo with effective ideal representations and $\mathcal{C} \subseteq 2^X$ be PTL-effective over $(X, \leq)$. Then the adherence membership problem and the ideal decomposition problem are Turing-equivalent.*

The seemingly innocuous hypothesis that $\mathcal{C}$ is PTL-effective is actually crucial: Theorem V.2 and Theorem VIII.1 in [22] provide an instance of undecidable adherence membership but computable ideal decompositions over a wqo with effective ideal representations; in these results, $\mathcal{C}$ is the set of run structures between two configurations of a vector addition system, and has decidable emptiness but lacks effective closure under intersection with PTLs.

**Related Work over Finite Words.** Propositions 13 and 16 together show that PTL separability reduces to the computation of ideal decompositions for downward-closures over any wqo with effective ideal representations. This includes as a special case $(\Sigma^*, \sqsubseteq_s)$ using the representations in [19, 1], and since ideal decompositions of downward-closures are computable for many classes of PTL-effective word languages [30] – including context-free languages [29, 10], reachability languages of vector addition systems [16], matrix languages [30], and higher-order languages [17, 8, 4] –, PTL separability is decidable for them.

In fact, propositions 13 and 16 subsume most of Theorem 2.5 of Czerwiński et al. [12], which is stated for *full trios* $\mathcal{C}$ over $\Sigma^*$, i.e. classes of languages closed under rational transductions – which are thus effectively closed under intersection with PTLs. There is a small price to pay for our level of generality, which is that their Theorem 2.5 also shows a converse reduction over $(\Sigma^*, \sqsubseteq_s)$ from the ideal decomposition problem back to the PTL separability problem. In our general setting, the best we know is that $L \stackrel{\text{def}}{=} I$ and $L' \stackrel{\text{def}}{=} X \setminus {\downarrow}(I \cap S)$ are PTL-separable if and only if $I \in \mathrm{Adh}(S)$ by Lemma 14, but this either uses the closure of $\mathcal{C}$ under complementation and downward-closure, or already uses computable ideal decompositions. The former however holds for regular languages:

▶ **Proposition 17.** *There is a many-one reduction from the adherence membership problem to the PTL separability problem for regular tree languages over* $(T(\mathcal{F}), \sqsubseteq_T)$.

## 5 Ideals for Ranked Trees with Homeomorphic Embedding

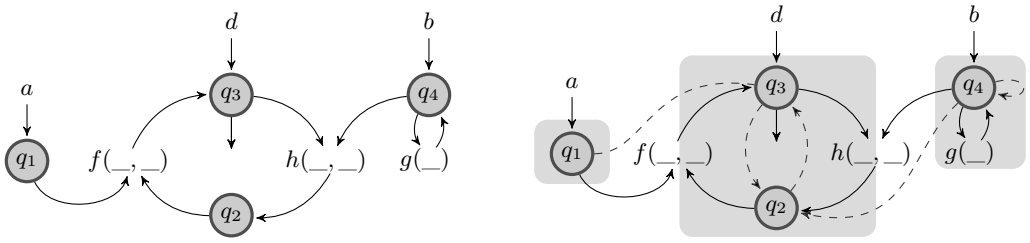
In this section, we provide finite representations for ideals of ranked trees ordered by homeomorphic embedding. These representations are expressed as tree regular expressions [9, Section 2.2]. We show in Section 5.1 that any downwards-closed subset of $T(\mathcal{F})$ can be represented as a *simple tree regular expression* (STRE), which we construct from its tree automaton. In Section 5.2, we then characterise ideals in this syntax as *tree products*, obtained as the summands of the normal form according to a rewrite system. Thanks to this particular proof strategy, the entire construction also solves the ideal decomposition problem: given a regular tree language, first build the STRE for its downward-closure, then normalise this STRE into a union of tree products representing its ideals.

Note that a concrete syntax for ranked tree ideals was proposed in [13]. However, no proof was given, and indeed the proposed tree regular expressions failed to be downwards-closed.

### 5.1 Simple Tree Regular Expressions

Let $L \subseteq T(\mathcal{F})$ be a downwards-closed language. Its complement is upwards-closed and has finitely many minimal elements, i.e. $T(\mathcal{F}) \setminus L = {\uparrow}\{t_1, \ldots, t_n\} = \bigcup_{1 \leq i \leq n} {\uparrow} t_i$, hence this is a regular tree language and $L$ is also regular. Thus $L$ is recognised by a finite (bottom-up) tree automaton $\mathcal{A}$, or equivalently by a tree regular expression [9, Section 2.2].

We shall see that $\mathcal{A}$ is equivalent to an expression of a specific shape, called a *simple tree regular expression* (STRE). We describe next a general procedure that converts any ($\varepsilon$-free) finite tree automaton $\mathcal{A}$ to an STRE $S$ whose language is the downward-closure ${\downarrow} L(\mathcal{A})$ of the




**(a)** Initial tree automaton. **(b)** $\varepsilon$-strongly connected components.

**Figure 1** Converting tree automata to STREs.

language recognised by $\mathcal{A}$. This procedure is best explained on an example: see Figure 1a, where there is one 0-ary transition $a$ (from no state) to state $q_1$, one binary transition $f$ from the pair of states $q_1$, $q_2$ to $q_3$, and so on. In textual form, we write these transitions as rewrite rules [9]: $a \to q_1$, $f(q_1, q_2) \to q_3$, $h(q_3, q_3) \to q_2$, $c \to q_3$, $g(q_4) \to q_4$, $b \to q_4$.

We first extend our automaton with $\varepsilon$-transitions. An $\varepsilon$-transition from $s$ to $s'$ will be drawn as a dashed arrow (see Figure 1b), and is just a rewrite rule of the new form $s \to s'$. This implies that every tree recognised at $s$ is also recognised at $s'$. For each transition, say $f(s_1, s_2, \cdots, s_n) \to s$, of $\mathcal{A}$, we add $n$ $\varepsilon$-transitions $s_1 \to s$, $s_2 \to s$, ..., $s_n \to s$. Call the resulting automaton $\downarrow\mathcal{A}$. It is an easy exercise to show that $L(\downarrow\mathcal{A}) = \downarrow L(\mathcal{A})$.

There is a graph underlying $\downarrow\mathcal{A}$, whose vertices are the states of $\downarrow\mathcal{A}$, and whose edges are the $\varepsilon$-transitions. Consider its strongly connected components, shown against a grey background in Figure 1b. Any two states in the same strongly connected component $C$ recognise exactly the same trees, so it makes sense to talk of the language $L_C(\downarrow\mathcal{A})$ of those trees recognised at any state of $C$. Let $C \to C'$ if and only if $s \to s'$ for some $s \in C$, $s' \in C'$. The strict ordering $\prec \stackrel{\text{def}}{=} \to^+$ is well-founded, and we shall build an expression $S_C$ whose language is $L_C(\downarrow\mathcal{A})$ by induction along $\prec$.

**Trivial Components.** If $C$ is a trivial strongly connected component (one state $s$, no self-edge), then enumerate its incoming non-$\varepsilon$ transitions $f_i(s_{i1}, s_{i2}, \cdots, s_{in_i}) \to s$, $1 \leq i \leq m$. Let $S_{ij}$ be an expression whose language is the set of trees recognised at $s_{ij}$, which is given by induction hypothesis. Then $S_C \stackrel{\text{def}}{=} P_1 + \cdots + P_m$ with $P_i \stackrel{\text{def}}{=} f_i^?(S_{i1}, S_{i2}, \cdots, S_{in_i})$ is our desired expression. For instance, the set of trees recognised at the leftmost state $q_1$ of Figure 1b is the language of $S_1 \stackrel{\text{def}}{=} a^?$.

Here, a tree $t$ is in the language of an expression $P = f^?(S_1, \ldots, S_n)$ (written '$t \in P$') for $f \in \mathcal{F}_n$ and expressions $S_1, \ldots, S_n$ if and only if either $t$ is of the form $f(t_1, \cdots, t_n)$ with $t_i \in S_i$ for every $i$, $1 \leq i \leq n$, *or* if $t \in \bigcup_{i=1}^n S_i$; as the notation suggests, for $S = P_1 + \cdots + P_m$, $t \in S$ if and only if $t \in P_j$ for some $j$, $1 \leq j \leq m$.

**Iterators.** If $C$ is a non-trivial strongly connected component, then enumerate the non-$\varepsilon$ transitions $f_i(s_{i1}, s_{i2}, \cdots, s_{in_i}) \to s_i$, $1 \leq i \leq m$, whose end state $s_i$ is in $C$. For each pair $i, j$, if $s_{ij}$ is in $C$, then $S_{\Box ij} \stackrel{\text{def}}{=} \Box$ is a placeholder; otherwise, let $S_{\Box ij}$ be an expression whose language is the set of trees recognised at $s_{ij}$, which we obtain by induction hypothesis. Then $S_C \stackrel{\text{def}}{=} (A_1 + \cdots + A_m)^*.0$ for $A_i \stackrel{\text{def}}{=} f_i(S_{\Box i1}, S_{\Box i2}, \cdots, S_{\Box in_i})$ is a suitable expression. For example, the rightmost strongly connected component $\{q_4\}$ of Figure 1b yields the expression $S_4 \stackrel{\text{def}}{=} (b + g(\Box))^*.0$. One might have expected an expression of the more intuitive form $(g(\Box))^*.b^?$, however, as we are going to see, they define exactly the same language.

Here, 0 denotes the empty sum with empty language, and intuitively the language of an *iterator* $C^*.S$ for $C = A_1 + \cdots + A_m$ should consist of all trees obtained by repeatedly applying *contexts* from $A_1, \ldots, A_m$ – i.e. trees over the extended alphabet $\mathcal{F} \cup \{\Box\}$ where $\Box$ has arity 0 –, until one reaches a tree in $S$. More precisely, the language of an *atom* $A = f(S_{\Box 1}, \cdots, S_{\Box n})$ consists of those contexts in $T(\mathcal{F} \cup \{\Box\})$ of the form $f(c_1, \cdots, c_n)$ where $c_i \in S_{\Box i}$ for every $i$. In turn, $c \in S_\Box$ if and only if either $S_\Box = \Box$ and $c$ is the trivial context $\Box$, or $S_\Box$ is an expression $S$, $c$ is a tree $t$ in $T(\mathcal{F})$, and $t \in S$. The language of $C = A_1 + \cdots + A_m$ is the union of the languages of $A_j$, $1 \leq j \leq m$. For example, $(f(\Box))^*.a^?$ will recognise all trees of the form $f^n(a)$, $n \in \mathbb{N}$. There are however two catches:

1. The first one has to do with atoms $A$ where the placeholder $\Box$ occurs more than once: as usual with tree regular expressions, when replacing $\Box$ by a tree from $S$, several occurrences of $\Box$ can be replaced by *different* trees from $S$. Hence $(f(\Box, \Box))^*.(a^? + b^?)$ consists of all binary-branching trees with inner nodes labelled $f$ and leaves labelled $a$ or $b$, including $f(f(a, a), a)$ and $f(f(b, b), b)$ but also $f(f(a, b), a))$ or $f(f(b, b), a)$ among others. For a context $c$, and a set of trees $S$, we shall write $c[S]$ for the set of trees obtained from $c$ by replacing each occurrence of $\Box$ by a (possibly different) tree in $S$.
2. The second catch has to do with downward-closure. It is tempting to define the trees of $C^*.S$ as those in $c_1[\cdots[c_k[S]]\cdots]$, for some $k \in \mathbb{N}$ and some $c_1, \cdots, c_k \in C$. However, there are cases where that language would fail to be downwards-closed, e.g., $(f(a^?, \Box))^*.b^?$ would contain $f(a, b)$ but not $a$, according to that semantics.
   We repair that as follows. For $A = f(S_{\Box 1}, \cdots, S_{\Box n})$, define $\operatorname{supp} A$ as follows: if those $S_{\Box i}$, $1 \leq i \leq n$, that are different from $\Box$ define non-empty languages, then $\operatorname{supp} A$ is the union of those languages; if some $S_{\Box i} \neq \Box$ has an empty language, then $\operatorname{supp} A = \emptyset$. Hence, for example, $\operatorname{supp} f(\Box, \Box) = \emptyset$, $\operatorname{supp} f(a^?, \Box) = a^? = \{a\}$, and $\operatorname{supp} f'(a^?, \Box, 0) = \emptyset$. For $C = A_1 + \cdots + A_m$, let $\operatorname{supp} C = \bigcup_{j=1}^m \operatorname{supp} A_j$.

We are now ready to define the language of $C^*.S$, as the language of trees in $c_1[\cdots[c_k[S \cup \operatorname{supp} C]]\cdots]$, for some $k \in \mathbb{N}$ and some $c_1, \cdots, c_k \in C$.

Finally, $\downarrow L(\mathcal{A})$ is the union of the languages of the strongly connected components containing a final state of $\mathcal{A}$; in our example the strongly connected component in the middle yields the final expression $(d + f(S_1, \Box) + h(\Box, S_4))^*.0$.

**General Syntax.** Summing up, we define *simple tree regular expressions* (STREs) by the following abstract syntax:

$$S ::= P_1 + \cdots + P_m \qquad P ::= f^?(S, \cdots, S) \mid C^*.S$$
$$C ::= A + \cdots + A \qquad A ::= f(S_\Box, \cdots, S_\Box) \qquad S_\Box ::= S \mid \Box$$

where $f \in \mathcal{F}_r$ in $f^?(S_1, \cdots, S_r)$ and in $f(S_{\Box 1}, \cdots, S_{\Box r})$, the sum operation $+$ is associative and commutative (we shall sometimes write $\sum_{i=1}^m P_i$ for $P_1 + \cdots + P_m$) with 0 denoting the empty sum, and $\Box \notin \mathcal{F}$ is a placeholder. Note that $\Box$ is not meant to denote a family of placeholders, rather a single one. The STREs of the form $P$ are called *tree pre-products*. Among them, the *tree products* will be our notations for ideals, see Section 5.2.

▶ **Proposition 18.** *Every STRE defines a downwards-closed language of $(T(\mathcal{F}), \sqsubseteq_T)$. Every downwards-closed language of $(T(\mathcal{F}), \sqsubseteq_T)$ is the language of some STRE.*

**Proof.** The first part can be shown by structural induction on STREs; see the full paper for details. The second part was sketched above. ◀

$$
\begin{array}{rcll}
P + P' & \to_1 & P' & \text{if } P \subseteq P' \\
0 + P & \to_1 & P & \\
0^*.S & \to_1 & S & \\
f^?(\vec{S}_1, 0, \vec{S}_2) & \to_1 & 0 & \\
f(\vec{S}_{\Box 1}, 0, \vec{S}_2) & \to_1 & 0 & \\
C^*.0 & \to_1 & 0 & \\
C^*.(S + S') & \to_1 & C^*.S + C^*.S' & \text{if } C \text{ is } \Box\text{-linear}
\end{array}
\qquad
\begin{array}{rcll}
A + A' & \to_1 & A' & \text{if } A \subseteq A' \\
0 + A & \to_1 & A & \\
(C + f(S_1, \cdots, S_n))^*.S & \to_1 & C^*.(S + f^?(S_1, \cdots, S_n)) & \\
f^?(\vec{S}_1, S + S', \vec{S}_2) & \to_1 & f^?(\vec{S}_1, S, \vec{S}_2) + f^?(\vec{S}_1, S', \vec{S}_2) & \\
f(\vec{S}_{\Box 1}, S_\Box + S'_\Box, \vec{S}_{\Box 2}) & \to_1 & f(\vec{S}_{\Box 1}, S_\Box, \vec{S}_{\Box 2}) + f(\vec{S}_{\Box 1}, S'_\Box, \vec{S}_{\Box 2}) & \\
\multicolumn{4}{c}{\text{if } C = \sum_{i=1}^m f_i(\Box, \cdots, \Box) \text{ and no } f_i \text{ is 0-ary}}
\end{array}
$$

**Figure 2** The rewrite relation $\to_1$.

## 5.2 Tree Products

We now characterise the STREs that define ideals of $(T(\mathcal{F}), \sqsubseteq_T)$. We define a rewrite relation $\to_1$ on STREs that moves all $+$ signs to the outside: for a $\to_1$-normal STRE $S = P_1 + \cdots + P_m$, each $P_i$ will be irreducible, hence $S$ will be an ideal if and only if $m = 1$ by Fact 9.

The rewrite relation $\to_1$ is defined in Figure 2. Recall that $+$ is understood modulo associativity and commutativity. Letters matter, too: $S$, $S'$, $S_1$, …, $S_n$ are STREs, while $P$, $P'$ are those special STREs of the form $f^?(S_1, \cdots, S_n)$ or $C^*.S$, etc. In particular, the third rule of the second column applies provided the pattern $f(S_1, \cdots, S_n)$ does not contain $\Box$ at all. In the first rules, note that inclusion of STREs is decidable. For the two bottom rules, we need some auxiliary definitions. Say that a pattern $A = f(S_{\Box 1}, \cdots, S_{\Box n})$ is $\Box$-*linear* if and only if at most one $S_{\Box i}$ is the placeholder $\Box$. Writing $C$ as $A_1 + \cdots + A_m$, we say that $C$ is $\Box$-linear if and only if every non-empty $A_i$ is $\Box$-linear. The $\Box$-linearity restriction imposed on the last two rules is needed for the following to hold.

▶ **Fact 19.** *If $S \to_1^* S'$ then $S$ and $S'$ define the same language.*

▶ **Lemma 20.** *Every STRE $S$ has a normal form with respect to $\to_1$.*

**Proof.** Using Bachmair and Plaisted's associative path ordering $>_{apo}$ [5] on a precedence where $+$ is minimal, $f > f^?$ for each symbol $f$, and the $(\_)^*.\_$ operator has lexicographic status, we see that $\to_1$ is even a terminating relation. ◀

▶ **Definition 21.** A *tree product* is any $\to_1$-normal tree pre-product $P$.

▶ **Lemma 22.** *Every ideal of $T(\mathcal{F})$ is the language of some tree product.*

**Proof.** By Proposition 18, an ideal $I$ is the language of some STRE $S$, which has a $\to_1$-normal form by Lemma 20; write it $P_1 + \cdots + P_m$. Since $I$ is non-empty, $m \geq 1$, and since $I$ is irreducible (Fact 9), it is included in, hence equal to, the language of some $P_i$. ◀

Conversely, the language of any tree product is directed (see the full paper for a proof), thus:

▶ **Theorem 23.** *The ideals of $T(\mathcal{F})$ are exactly the languages of tree products.*

# 6 Concluding Remarks

We have presented a general order-theoretic characterisation of PTL separability, and shown that it could be applied to decide PTL separability of languages beyond finite words, namely of ranked regular tree languages ordered by homeomorphic embedding. Our work further adds

to the growing body of algorithmic applications of downwards-closed sets and ideals of well-quasi-orders in logic and verification, e.g. in forward analysis [13, 14], backward analysis [21], inference of inductive invariants [23], and reachability in vector addition systems [22].

We are confident our techniques apply to unranked trees, by defining suitable ideal representations. In the same vein, it would be interesting to develop ideal representations for the tree minor ordering, and to try to decide PTL separability for context-free tree languages.

An open-ended question is how to finely relate our order-theoretic characterisation with the algebraic and topological characterisations typically employed for the definability and separability problems. For instance, can one derive the characteristic equations of Bojańczyk et al. [6] for piecewise-testable tree languages from Lemma 12?

**Acknowledgements.** The authors thank Wojciech Czerwiński, Luc Segoufin, and Georg Zetzsche for their insightful comments, and an anonymous reviewer for correcting Fact 8.

## References

1 Parosh Aziz Abdulla, Aurore Collomb-Annichini, Ahmed Bouajjani, and Bengt Jonsson. Using forward reachability analysis for verification of lossy channel systems. *Formal Methods in System Design*, 25(1):39–65, 2004. doi:10.1023/B:FORM.0000033962.51898.1a.

2 Jorge Almeida. Some algorithmic problems for pseudovarieties. *Publicationes Mathematicae Debrecen*, 54(suppl.):531–552, 1999.

3 Jorge Almeida and Marc Zeitoun. The pseudovariety J is hyperdecidable. *RAIRO Theoretical Informatics and Applications*, 31:457–482, 1997.

4 Kazuyuki Asada and Naoki Kobayashi. On word and frontier languages of unsafe higher-order grammars. In *ICALP 2016*, Leibniz International Proceedings in Informatics, 2016. To appear. URL: https://arxiv.org/abs/1604.01595.

5 Leo Bachmair and David A. Plaisted. Termination orderings for associative-commutative rewriting systems. *Journal of Logic and Computation*, 1(4):329–349, 1985. doi:10.1016/S0747-7171(85)80019-5.

6 Mikołaj Bojańczyk, Luc Segoufin, and Howard Straubing. Piecewise testable tree languages. *Logical Methods in Computer Science*, 8(3), 2012. doi:10.2168/LMCS-8(3:26)2012.

7 Robert Bonnet. On the cardinality of the set of initial intervals of a partially ordered set. In *Infinite and finite sets: to Paul Erdős on his 60th birthday, Vol. 1*, Coll. Math. Soc. János Bolyai, pages 189–198. North-Holland, 1975.

8 Lorenzo Clemente, Paweł Parys, Sylvain Salvati, and Igor Walukiewicz. The diagonal problem for higher-order recursion schemes is decidable. In *LICS 2016*. ACM, 2016. To appear.

9 H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. *Tree Automata Techniques and Applications*. Inria, 2007. URL: http://tata.gforge.inria.fr/.

10 Bruno Courcelle. On constructing obstruction sets of words. *Bulletin of the EATCS*, 44:178–186, 1991.

11 Wojciech Czerwiński, Wim Martens, and Tomáš Masopust. Efficient separability of regular languages by subsequences and suffixes. In *ICALP 2013*, volume 7966 of *Lecture Notes in Computer Science*, pages 150–161. Springer, 2013. doi:10.1007/978-3-642-39212-2_16.

12 Wojciech Czerwiński, Wim Martens, Lorijn van Rooijen, Marc Zeitoun, and Georg Zetzsche. A characterization for decidable separability by piecewise testable languages. Preprint, 2015. An extended abstract appeared as:

W. Czerwiński, W. Martens, L. van Rooijen, and M. Zeitoun. A note on decidable separability by piecewise testable languages. In *FCT 2015*, volume 9210 of *LNCS*, pages 173–185. Springer, 2015. URL: http://arxiv.org/abs/1410.1042v2.

13 Alain Finkel and Jean Goubault-Larrecq. Forward analysis for WSTS, part I: Completions. In *STACS 2009*, volume 3 of *Leibniz International Proceedings in Informatics*, pages 433–444. LZI, 2009. doi:10.4230/LIPIcs.STACS.2009.1844.

14 Alain Finkel and Jean Goubault-Larrecq. Forward analysis for WSTS, part II: Complete WSTS. *Logical Methods in Computer Science*, 8(3), 2012. doi:10.2168/LMCS-8(3:28)2012.

15 Jean Goubault-Larrecq, Prateek Karandikar, K. Narayan Kumar, and Philippe Schnoebelen. The ideal approach to computing closed subsets in well-quasi-orderings. In preparation, 2016. See also an earlier version in: J. Goubault-Larrecq. On a generalization of a result by Valk and Jantzen. Research Report LSV-09-09, LSV, ENS Cachan, 2009. URL: http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/PDF/rr-lsv-2009-09.pdf.

16 Peter Habermehl, Roland Meyer, and Harro Wimmel. The downward-closure of Petri net languages. In *ICALP 2010*, volume 6199 of *Lecture Notes in Computer Science*, pages 466–477. Springer, 2010. doi:10.1007/978-3-642-14162-1_39.

17 Matthew Hague, Jonathan Kochems, and C.-H. Luke Ong. Unboundedness and downward closures of higher-order pushdown automata. In *POPL 2016*, pages 151–163. ACM, 2016. doi:10.1145/2837614.2837627.

18 Graham Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, 3(2):326–336, 1952. doi:10.1112/plms/s3-2.1.326.

19 Pierre Jullien. *Contribution à l'étude des types d'ordres dispersés*. Thèse de doctorat, Université de Marseille, 1969.

20 Joseph B. Kruskal. Well-quasi-ordering, the Tree Theorem, and Vazsonyi's Conjecture. *Transactions of the American Mathematical Society*, 95(2):210–225, 1960. doi:10.2307/1993287.

21 Ranko Lazić and Sylvain Schmitz. The ideal view on Rackoff's coverability technique. In *RP 2015*, volume 9328 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2015. doi:10.1007/978-3-319-24537-9_8.

22 Jérôme Leroux and Sylvain Schmitz. Demystifying reachability in vector addition systems. In *LICS 2015*, pages 56–67. IEEE Press, 2015. doi:10.1109/LICS.2015.16.

23 Oded Padon, Neil Immerman, Sharon Shoham, Aleksandr Karbyshev, and Mooly Sagiv. Decidability of inferring inductive invariants. In *POPL 2016*, pages 217–231. ACM, 2016. doi:10.1145/2837614.2837640.

24 Thomas Place, Lorijn van Rooijen, and Marc Zeitoun. Separating regular languages by piecewise testable and unambiguous languages. In *MFCS 2013*, volume 8087 of *Lecture Notes in Computer Science*, pages 729–740. Springer, 2013. doi:10.1007/978-3-642-40313-2_64.

25 Thomas Place and Marc Zeitoun. Automata column: The tale of the quantifier alternation hierarchy of first-order logic over words. *SIGLOG News*, 2(3):4–17, 2015. doi:10.1145/2815493.2815495.

26 Imre Simon. Piecewise testable events. In *Automata Theory and Formal Languages*, volume 33 of *Lecture Notes in Computer Science*, pages 214–222. Springer, 1975. doi:10.1007/3-540-07407-4_23.

27 Thomas G. Szymanski and John H. Williams. Noncanonical extensions of bottom-up parsing techniques. *SIAM Journal on Computing*, 5(2):231–250, 1976. doi:10.1137/0205019.

28 W. W. Tait. A counterexample to a conjecture of Scott and Suppes. *Journal of Symbolic Logic*, 24(1):15–16, 1959. doi:10.2307/2964569.

**29** Jan van Leeuwen. Effective constructions in well-partially-ordered free monoids. *Discrete Mathematics*, 21(3):237–252, 1978. doi:10.1016/0012-365X(78)90156-5.

**30** Georg Zetzsche. An approach to computing downward closures. In *ICALP 2015*, volume 9135 of *Lecture Notes in Computer Science*, pages 440–451. Springer, 2015. doi:10.1007/978-3-662-47666-6_35.