Logical Methods in Computer Science
Vol. 8(3:19)2012, pp. 1–39
www.lmcs-online.org

Submitted Oct. 2, 2010
Published Sep. 19, 2012

WREATH PRODUCTS OF FOREST ALGEBRAS, WITH APPLICATIONS TO TREE LOGICS *

MIKOLAJ BOJAŃCZYK [a], HOWARD STRAUBING [b], AND IGOR WALUKIEWICZ [c]

[a] University of Warsaw
e-mail address: bojan@mimuw.edu.pl

[b] Boston College
e-mail address: Straubing@cs.bc.edu

[c] LaBRI (Université de Bordeaux - CNRS)
e-mail address: igw@labri.fr

Abstract. We use the recently developed theory of forest algebras to find algebraic characterizations of the languages of unranked trees and forests definable in various logics. These include the temporal logics CTL and EF, and first-order logic over the ancestor relation. While the characterizations are in general non-effective, we are able to use them to formulate necessary conditions for definability and provide new proofs that a number of languages are not definable in these logics.

1. Introduction

Logics for specifying properties of labeled trees play an important role in several areas of Computer Science. We say that a class of regular languages of trees $\mathscr{L}$ has an *effective characterization* if there is an algorithm which decides if a given regular language of trees belongs to $\mathscr{L}$. Effective characterizations are known only for a few logics. In particular, we do not know if such characterizations exist for the classes of languages defined by the most common logics such as : CTL, CTL*, PDL, or first-order logic with the ancestor relation.

In this paper we consider logics for *unranked* trees, in which there is no *a priori* bound on the number of children a node may have. Many such logics, including all the logics that are considered in this paper, are no more expressive than monadic second-order logic, and thus the properties they define can be described using automata. Barcelo and Libkin [1] and Libkin [15] catalogue a number of such logics and contrast their expressive power. We use recently developed theory of forest algebras to find algebraic characterisations of the

1998 ACM Subject Classification: F.4.3.
Key words and phrases: tree language, temporal logic, forest algebra, wreath product.
* Extended abstract of the paper has appeared at LICS'09.
[b] Research supported by NSF Grant CCF-0915065.
[c] Research supported by ANR 2010 BLAN 0202 01 FREC.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.2168/LMCS-8(3:19)2012

languages of unranked trees definable in some most common logics. While the characterizations are in general non-effective, we are able to use them to formulate necessary conditions for definability and provide new proofs that a number of languages are not definable in these logics.

For properties of *words,* such questions have been fruitfully studied by algebraic means. Whether or not a regular word language L can be defined in a given logic can often be determined by verifying some property of the *syntactic monoid* of L—the transition monoid of the minimal automaton of L. The earliest work in this direction is due to McNaughton and Papert [20] who studied first-order logic with linear order, and showed that a language is definable in this logic if and only if its syntactic monoid is aperiodic—that is, contains no nontrivial groups. A comprehensive survey treating many different predicate logics is given in Straubing [23]; temporal logics are studied by Cohen, Perrin and Pin [8] and Wilke [24], among others.

Algebraic techniques provide a striking alternative to purely model-theoretic methods for studying the expressive power of logics over words. In many cases they have led to effective characterizations of certain logics, and actually to reasonably efficient algorithms. Even in the absence of effective characterizations, it is frequently possible to obtain effective necessary conditions for expressibility in a logic and use these to show the non-expressibility of certain languages. For instance, the strictness of the Σ_k-hierarchy in first-order logic on words–the *dot-depth* hierarchy– was first proved by such algebraic means (Brzozowski and Knast [7], Straubing [22]), while effective characterization of the levels of the hierarchy remains an open problem.

There have been a number of efforts to extend this algebraic theory to trees; a notable recent instance is in the work of Ésik and Weil on preclones [9, 10]. Recently, Bojańczyk and Walukiewicz [3] introduced *forest algebras*, and along with it the *syntactic forest algebra,* which generalize monoids and the syntactic monoid for languages of forests of unranked trees. This algebraic model is rather simple, and in contrast to others studied in the literature, has already yielded effective criteria for definability in a number of logics: see Bojańczyk [4], Bojańczyk-Segoufin-Straubing [6], Bojańczyk-Segoufin [5]. Forest algebras are also implicit in the work of Benedikt and Segoufin [2] on first-order logic with successor and of Place and Segoufin [18] on locally testable tree languages.

In the present paper we continue the study of forest algebras, by developing a theory of composition of forest algebras, using the *wreath product*. The wreath product of transformation monoids plays an important role in the theory for words. In particular, it is connected to a composition operation on languages and to generalized temporal operators. This paper is concerned with describing the connection between formula composition and the wreath product of forest algebras, in the case of unranked trees. Here is a brief summary of our results:

(1) To each logic $\mathscr{L}$ among EF, CTL, CTL*, first-order logic with ancestor, PDL and graded PDL, we associate a class of forest algebras, called the base of $\mathscr{L}$. We show that a language of forests is definable in the logic $\mathscr{L}$ if and only if it is recognized by an iterated wreath product of the forest algebras from the base of $\mathscr{L}$. (Theorem 5.2.)
(2) In the cases of EF and CTL, the base has a single forest algebra. For the other cases we show that there is no finite base. As a consequence, none of these logics can be generated by a finite collection of generalized temporal operators. Using our algebraic framework, we give a simple and general proof of this fact. (Theorem 5.5.)

(3) For the logics that do not have a finite base, we give an effective characterization of the base. (Theorems 5.3 and 5.4.) Note that an effective characterization of a base does not imply an effective characterization for wreath products of the base, so this result does not give an effective characterization of any of the logics mentioned in item (1).
(4) Going one step further, we provide an effective characterization for the *path languages* (Theorem 5.4): boolean combinations of languages from the base of graded PDL.
(5) We give a new proof, based on the wreath product, of an effective characterization of the logic EF. This result was proved earlier by other means. (Bojanczyk and Walukiewicz [3].) Our argument here computes a decomposition based on the ideal structure of the underlying forest algebra.
(6) Although we do not find effective characterizations for other prominent logics from our list, we are able to use our framework to establish necessary conditions for definability in these logics, and consequently to prove that a number of specific languages are not definable in them. (Theorem 8.2.)
(7) We give an effective characterization of CTL* languages within first-order definable languages. Similarly for PDL languages within languages definable in graded PDL (Theorem 9.2.)

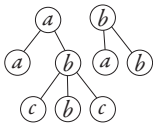
Plan of the paper. In Sections 2-4 we present the basic terminology concerning, respectively, trees, logic, and forest algebras. Our treatment of temporal logics is somewhat unorthodox, since our algebraic theory requires us to interpret formulas in forests as well as in trees, therefore the precise syntax and semantics are different in the two cases. Section 4 includes a detailed treatment of the wreath product of forest algebras. In Section 5 we establish the first of our main results, giving wreath product characterizations of all the logics under consideration. In Section 6 we give the effective characterization of EF, and in Section 7 the necessary conditions for definability in the other logics. Section 8 is devoted to applications of these conditions.

We note that Ésik and Ivan [11, 12] have done work of a similar flavor for CTL (for trees of bounded rank). Our work here is of considerably larger scope, both in the number of different logics considered, and the concrete consequences our algebraic theory permits us to deduce.

The present article is the complete version of an extended abstract presented at the 2009 IEEE Symposium on Logic in Computer Science.

2. Trees, Forests and Contexts

Let A be a finite alphabet. Formally, forests and trees over A are expressions generated by the following rules: *(i)* if s is a forest and $a \in A$ then as is a tree; *(ii)* if $(t_1, \ldots, t_k)$ is a finite sequence of trees, then $t_1 + \cdots + t_k$ is a forest. We permit this summation to take place over an empty sequence, yielding the *empty forest,* which we denote by 0, and which gets the recursion started. So, for example, the following forest with two roots



is described by the expression

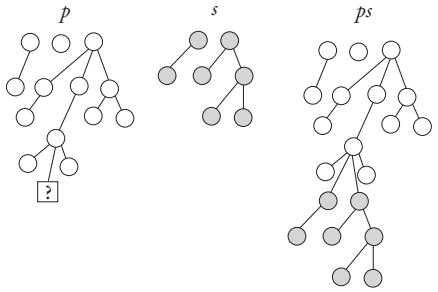
$$a(a0 + b(c0 + b0 + c0)) + b(a0 + b0).$$

Normally, when we write such expressions, we delete the zeros. We denote the set of forests over A by H_A. This set forms a monoid with respect to forest concatenation $s+t$, with the empty forest 0 being the identity. We denote the set of trees over A by T_A.

If x is a node in a forest, then the *subtree* of x is simply the tree rooted at x, and the *subforest* of x is the forest consisting of all subtrees of the children of x. In other words, if the subtree of x is as, with $a \in A$ and $s \in H_A$, then the subforest of x is s. Note that the subforest of x does not include the node x itself, and is empty if x is a leaf.

A *forest language* over A is any subset of H_A.

A *context* p over A is formed by replacing a leaf of a nonempty forest by a special symbol $\Box$. Think of $\Box$ as a kind of place-holder, or *hole*. Given a context p and a forest s, we form a forest ps upon substituting s for the hole in p. In the interpretation of forests as expressions, this really is just substitution of the expression s for the hole of p; the graphical interpretation of this operation is depicted below.



In a similar manner, we can substitute another context q for the hole, and obtain a new context pq. We obtain in this way a composition operation on contexts. We denote the set of contexts over A by V_A. This set forms a monoid, with respect to this composition operation, with the empty context $\Box$ as the identity.

Note that for any $s, t \in H_A$, V_A contains a context $s + \Box + t$, in which the hole has no parent, such that $(s + \Box + t)u = s + u + t$ for all $u \in H_A$.

Our trees, forests and contexts are ordered, so that $s + t$ is a different forest from $t + s$ unless $s = t$ or one of s, t is 0. This noncommutativity is important in a number of applications. However the present article really deals with unordered trees, so there is no harm in thinking of $+$ as a commutative operation on forests.

3. Logics for Forest Languages

We can define *regular* forest languages by means of an automaton model that is a minor modification of the standard bottom-up tree automaton. The transition function has to be altered to cope with unbounded branching, and the acceptance condition needs to take account of the sequence of states in the roots of all the trees in the forest. See [3] for a precise definition of such an automaton model. The usual equivalence between monadic second-order logic and regularity holds in this setting.

For a general treatment of predicate and temporal logics for unranked trees, we refer the reader to Libkin [15] and Barceló-Libkin [1]. We will have to give a somewhat different description of similar logics in order to express properties of forests as well as of trees. In all cases the logics that we describe are fragments of monadic second-order logic, and thus the languages they define are all regular forest languages.

3.1. **First-order logic for trees and forests.** Let A be a finite alphabet. Consider first-order logic equipped with unary predicates Q_a for each $a \in A$, and a single binary predicate $\prec$. Variables are interpreted as nodes in forests over A. Formula $Q_a x$ is interpreted to mean that node x is labeled a, and $x \prec y$ to mean that node x is a (non-strict) ancestor of node y. A *sentence* ϕ—that is, a formula without free variables—consequently defines a language $L_\phi \subseteq H_A$ consisting of forests over A that satisfy ϕ. For example, the sentence

$$\exists x \exists y (Q_a x \wedge Q_a y \wedge \neg(x \prec y) \wedge \neg(y \prec x))$$

defines the set of forests containing two incomparable occurrences of a. We denote this logic by $FO[\prec]$. Note that this logic has no predicates to access the order of siblings. In particular, any language defined by the logic will be horizontally commutative, i.e. closed under reordering sibling trees.

It is more traditional to consider logics over trees rather than over forests. For $FO[\prec]$ we need not worry too much about this distinction, since we can express in first-order logic the property that a forest has exactly one root (by the sentence $\exists x \forall y (x \prec y)$). Thus the question of whether a given set of trees is first-order definable does not depend on whether we choose to interpret sentences in trees or in forests.

3.2. **Temporal logics.** We describe here a general framework for temporal logics interpreted in trees and forests. By setting appropriate parameters in the framework we generate all sorts of temporal logics that are traditionally studied.

The general framework is called *graded propositional dynamic logic* (graded PDL).

Syntax of temporal formulas. We distinguish between two kinds of formulas: *tree formulas* and *forest formulas*. The syntax of these formulas is defined by mutual recursion, as follows:

- $\mathbf{T}$ and $\mathbf{F}$ are forest formulas.
- If $a \in A$, then a is a tree formula. (Such formulas are called *label formulas*.)
- Finite boolean combinations of tree formulas are tree formulas, and finite boolean combinations of forest formulas are forest formulas.
- Every forest formula is a tree formula.
- Before defining the key construction we need to introduce the concept of an *unambiguous* set of formulas. Such a set $\Phi = \{\phi_1, \ldots, \phi_{n+1}\}$ is constructed from a sequence of tree formulas $\psi_1, \ldots, \psi_n$ by a simple syntactic operation ensuring that every tree satisfies exactly one formula from Φ:

$$\phi_i = \psi_i \wedge \bigwedge_{j \leq i-1} \neg\psi_j \text{ for } i = 1, \ldots, n \qquad \text{and} \qquad \phi_{n+1} = \bigwedge_{j \leq n} \neg\psi_j.$$

- If Φ is a finite unambiguous set of tree formulas, $k > 0$ is an integer, and $L \subseteq \Phi^*$ is a regular language then $\mathsf{E}^k L$ is a forest formula.

Semantics of temporal formulas. We define two notions of satisfaction: *tree satisfaction* $t \models_t \phi$, where t is a tree and ϕ is a tree formula, which coincides with the usual notion of satisfaction; and *forest satisfaction* $t \models_f \varphi$, where t is a forest and ϕ is a forest formula, which is somewhat unusual. Again, these relations are defined by mutual recursion.

- If $t \in H_A$ then $t \models_f \mathbf{T}$ and $t \not\models_f \mathbf{F}$.
- If $t \in T_A$ and $a \in A$, then $t \models_t a$ if and only if the root node of t is labeled a.
- Boolean operations have their usual meaning; *e.g.,* if $t \in H_A$ and ϕ_1, ϕ_2 are forest formulas, then $t \models_f \phi_1 \wedge \phi_2$ if and only if $t \models_f \phi_1$ and $t \models_f \phi_2$.

- Let ϕ be a forest formula and $t \in T_A$, so that $t = as$ for a unique $s \in H_A$, $a \in A$. Then $t \models_t \phi$ if and only if $s \models_f \phi$.
- Let $k > 0$, and let Φ be a finite unambiguous set of tree formulas, with $L \subseteq \Phi^*$ a regular language. Let $s \in H_A$. If x is a node of s, then we label the node by ϕ_i if $t_x \models_t \phi_i$, where t_x is the subtree of x. Note that because of the unambiguity requirement, there is exactly one such label. A path $x_1, \ldots, x_n$ of consecutive nodes of s beginning at a root, but not necessarily extending to a leaf, thus yields a unique word $\phi_{i_1} \ldots \phi_{i_n} \in \Phi^*$, which we call the Φ-path of x_n. We say $s \models_f \mathsf{E}^k L$ if there are at least k nodes in the forest whose Φ-path belongs to L.

We stress that in counting paths, we do not require the paths to be disjoint, and we do not require them to extend all the way to the leaves. For example, the forest $aa + a$ contains three different nonempty paths from the root, so this forest satisfies the formula $\mathsf{E}^3 a^+$.

Given a temporal formula ψ, we write L_ψ for the set of forests that forest satisfy ψ:

$$L_\psi = \{s \in H_a : s \models_f \psi\}$$

We specialize the above framework by restricting either the value of k or the language L in the application of the operator $E^k L$, or both. This leads, in the case of trees, to some logics that have been widely studied. We catalogue these below:

EF. As a first example, we show how to implement the operator "there exists a descendant", often denoted by EF. This example also highlights the difference between tree satisfaction and forest satisfaction. Consider the special case of $\mathsf{E}^k L$ where, for some tree formula ψ,

$$\Phi = \{\psi, \neg\psi\} \qquad k = 1 \qquad L = (\neg\psi)^*\psi. \tag{3.1}$$

In this special case, we write $\mathsf{EF}\psi$ instead of $\mathsf{E}^k L$. It is easy to see that $\mathsf{EF}\psi$ is forest satisfied by a forest s if and only if ψ is satisfied by some subtree of s. In the special case when s is a tree, this subtree may be s itself. The semantics shows how to interpret $\mathsf{EF}\psi$ as a tree formula. If t is a tree, then t tree satisfies $\mathsf{EF}\psi$ if and only if t has a proper subtree that satisfies ψ. In other words, tree satisfaction of $\mathsf{E}\psi$ corresponds to the so called strict semantics, while forest satisfaction of $\mathsf{E}\psi$ corresponds to the non strict semantics. We will use the term EF for the fragment of graded PDL where the operator $\mathsf{E}^k L$ is only used in the special case of $\mathsf{EF}\psi$.

CTL. As a second example, we show how to implement the operator $\mathsf{E}\psi\mathsf{U}\phi$ of CTL. Consider the special case of $\mathsf{E}^k L$ where, for some tree formulas ψ and ϕ,

$$\Phi = \{\psi \wedge \neg\phi, \neg\psi \wedge \neg\phi, \phi\} \qquad k = 1 \qquad L = (\psi \wedge \neg\phi)^*\phi. \tag{3.2}$$

In this special case, we write $\mathsf{E}\psi\mathsf{U}\phi$ instead of $\mathsf{E}^k L$. It is easy to see that $\mathsf{E}\psi\mathsf{U}\phi$ is forest satisfied by a forest s if and only if the subtree of some node x tree satisfies the formula ϕ, and the subtree at every proper ancestor of x tree satisfies ψ. Let us look now at the tree semantics of the formula $\mathsf{E}\psi\mathsf{U}\phi$. If t is a tree, then t tree satisfies $\mathsf{E}\psi\mathsf{U}\phi$ if and only if the subtree of some non-root node x tree satisfies ϕ, and every non-root proper ancestor of x tree satisfies ψ. As was the case with the operator EF, tree satisfaction corresponds to strict semantics and forest satisfaction corresponds to non strict semantics. We will use

the term CTL for the fragment of graded PDL where the operator $\mathsf{E}^k L$ is only used in the special case of $\mathsf{E}\psi\mathsf{U}\phi$. [1]

First-order logic. We use our temporal framework to characterize the languages definable in $FO[\prec]$.

Theorem 3.1. *A forest language is definable in $FO[\prec]$ if and only if it is definable by a forest formula in which the operator $\mathsf{E}^k L$ is restricted to word languages L that are first-order definable over an unambiguous finite alphabet Φ of tree formulas.*

Proof. The theorem is very similar to the result of Hafer and Thomas [14] who show that first-order logic coincides with CTL* on finite binary trees. The theorem is even closer to the result Moller and Rabinovich [17] who show that over infinite unranked trees Counting-CTL* is equivalent to *Monadic Path Logic (MPL)*. To deduce our theorem from their result it is enough to clarify the relations between different logics.

The logics considered by Moller and Rabinovich express properties of infinite unranked trees, which can have both infinite branches and finite branches that end in leaves. A *maximal path* is therefore defined as a path that begins in any node, is directed away from the root, and either continues infinitely or ends in a leaf. Monadic Path Logic (MPL) is the restriction of monadic second-order logic over the predicate $\prec$ in which second-order quantification is restricted to maximal paths. In other words, MPL is the extension of $FO[\prec]$ that allows quantification over maximal paths. Over infinite trees, MPL is more expressive than first-order logic, since it can define the property "some path contains infinitely many a's ", which cannot be defined in $FO[\prec]$. However, over finite unranked trees, $FO[\prec]$ has the same expressive power as MPL. This is because a maximal path in a finite tree can be described by its first node and the leaf where it ends.

The logic counting-CTL* can be interpreted as the the fragment of graded PDL where the operator $\mathsf{E}^k L$ is only allowed in the following two restricted forms:

- A next operator X^k. This formula holds in a tree if subtrees of at least k children of the root satisfy ϕ. If $\mathsf{X}^k\phi$ is a formula of counting-CTL* and $\widehat{\phi}$ is a translation of ϕ into graded-PDL then $\mathsf{X}^k\phi$ is translated into a tree formula $\mathsf{E}^k L_\phi$ where $L_\phi = \{\widehat{\phi}\}$. Indeed, such a formula requires existence of k different paths of length 1 whose labellings belong to $\widehat{\phi}$.
- An existential path operator, which we denote here by E' (the original paper uses E, but we use E' to highlight the slight change in semantics). This operator works like our $\mathsf{E}L$, but with the the difference that $\mathsf{E}'L$ is a tree formula, and the path begins in the unique root of the tree. Rabinovich and Moller require that L is definable in LTL, which is equivalent to first-order definability.

So counting-CTL* can be translated to a fragment of graded-PDL using only first-order definable word languages in quantification. Hence, by the result of Moller and Rabinovich we get a translation of $FO[\prec]$ to this fragment. The translation in the opposite direction is straightforward. □

[1] In most presentations CTL also has the "next" operator $\mathsf{EX}\phi$, as well as the dual operator $\mathsf{E}\neg(\psi\mathsf{U}\phi)$. The next operator is redundant thanks to the strict semantics, and the dual operator is redundant in finite trees.

Note that Theorem 3.1 fails without the restriction on unambiguity of the alphabet Φ. For instance, if we took $A = \{a, b, c\}$, $\Phi = \{\phi_1, \phi_2\}$, where $\phi_1 = a \vee c$, $\phi_2 = b \vee c$, then $L = (\phi_1\phi_2)^+$ is first-order definable as a word language. One can imagine what the semantics of $\mathsf{E}L$ should be in the case of such Φ: a node labelled with c can be labelled either with ϕ_1 or with ϕ_2. With this semantics however, the language defined by $\mathsf{E}L$ is not first-order definable. (If it were, we would be able to define in first-order logic the set of forests consisting of a single path with an even number of occurrences of c.)

Actually, one can show, using composition theorems similar to those used by Hafer and Thomas, or Moller and Rabinovich, that graded PDL has the same expressive power as chain logic, which is the fragment of monadic second order logic where set quantification is restricted to chains, i.e. subsets of paths.

CTL* and PDL. Finally, we define two more temporal logics by modifying the definitions above. CTL* is like the fragment of temporal logic in Theorem 3.1, except that we only allow $k = 1$ in $\mathsf{E}^k L$. In particular, CTL* is a subset of $FO[\prec]$. We also consider PDL, which is obtained by restricting the temporal formulas $\mathsf{E}^k L$ to $k = 1$, but without the requirement that L be first-order definable. If we place no restriction on either the multiplicity k or the regular language L, we obtain *graded PDL.*

3.3. **Language composition and bases.** In this section we provide a more general notion of temporal logic, where the operators are given by regular forest languages. This is similar to notions introduced by Ésik in [11]. The benefit of the general framework is twofold. First, it corresponds nicely with the algebraic notion of wreath product presented later in the paper. Second, it allows us to state and prove negative results, for instance our infinite base theorem, which says that the number of operators needed to obtain first-order logic is necessarily large.

We introduce a composition operation on forest languages. Fix an alphabet A, and let $\{L_1, \ldots, L_k\}$ be a partition of H_A. Let $B = \{b_1, \ldots, b_k\}$ be another alphabet, with one letter b_i for each block L_i of the partition. The partition and alphabet are used to define a relabeling

$$t \in H_A \qquad \mapsto \qquad t[L_1, \ldots, L_k] \in H_{A \times B}$$

in the following manner. The nodes in the forest $t[L_1, \ldots, L_k]$ are the same as in the forest t, but the labels are different. A node x that had label a in t gets label (a, b_i) in the new forest, where b_i corresponds to the unique language L_i that contains the subforest of x in t. For the partition and B as above, and L a language of forests over $A \times B$, we define $L[L_1, \ldots, L_k] \subseteq H_A$ to be the set of all forests t over A for which $t[L_1, \ldots, L_k] \in L$.

The operation of language composition is similar to formula composition. The definitions below use this intuition, in order to define a "temporal logic" based on operators given as forest languages. Formally, we will define the closure of a language class under language composition. First however, we need to comment on a technical detail concerning alphabets. In the discussion below, a forest language is given by two pieces of information: the forests it contains, and the input alphabet. For instance, we distinguish between the set L_1 of all forests over alphabet $\{a\}$, and the set L_2 of all forests the alphabet $\{a, b\}$ where b does not appear. The idea is that sometimes it is relevant to consider a language class $\mathscr{L}$ that contains L_1 but does not contain L_2, such as the class of *definite* languages that only look at a bounded prefix of the input forest (such classes will not appear in this particular

Logic	Languages in the language base for alphabet A
EF	$\{$"some node with a" : $a \in A\}$
CTL	$\{$"some path in B^*b" : $B \subseteq A, b \in A\}$
$FO[\prec]$	$\{$"at least k paths in L" : $k \in \mathbb{N}, L \in FO_A[<]\}$
CTL*	$\{$"some path in L" : $L \in FO_A[<]\}$
PDL	$\{$"some path in $L \subseteq A^+$" : L regular$\}$
graded PDL	$\{$"at least k paths in $L \subseteq A^+$" : $k \in \mathbb{N}, L$ regular$\}$

Figure 1: Language bases for temporal logics

paper). This distinction will be captured by our notion of language class: a language class is actually a mapping $\mathscr{L}$, which associates to each finite alphabet a class of languages over this alphabet.

Let $\mathscr{L}$ be a class of forest languages, which will be called the *language base*. The temporal logic with language base $\mathscr{L}$ is defined to be the smallest class $\mathsf{TL}[\mathscr{L}]$ of forest languages that contains $\mathcal{L}$ and is closed under boolean combinations and language composition, i.e.

$$L_1, \ldots, L_k, L \in \mathsf{TL}[\mathscr{L}] \quad \Rightarrow \quad L[L_1, \ldots, L_k] \in \mathsf{TL}[\mathscr{L}].$$

Formally speaking, in the above we should highlight the alphabets (the languages $L_1, \ldots, L_k$ and $L[L_1, \ldots, L_k]$ belong to the part of $\mathsf{TL}[\mathscr{L}]$ for alphabet A, while the language L belongs to the part of $\mathsf{TL}[\mathscr{L}]$ for alphabet $A \times B$, as in the definition of the composition operation).

We can translate the definitions of the temporal logics we have considered in terms of language composition. This gives the following theorem.

Theorem 3.2. *The logics EF, CTL, $FO[\prec]$, CTL* , PDL and graded PDL have language bases as depicted in Figure 1.*

Note that the assertion about $FO[\prec]$ depends on Theorem 3.1.

4. Forest Algebras

4.1. **Definition of forest algebras.** Forest algebras, introduced in [3] by Bojańczyk and Walukiewicz, extend the algebraic theory of syntactic monoid and syntactic morphism for regular languages of words to the setting of unranked trees and forests. A *forest algebra* is a pair (H, V) of monoids together with a faithful monoidal left action of V on the set H. This means that for all $h \in H$, $v \in V$, there exists $vh \in H$ such that (i) $(vw)h = v(wh)$ for all $v, w \in V$ and $h \in H$, (ii) if $1 \in V$ is the identity element, then $1h = h$ for all $h \in H$, and (iii) if $vh = v'h$ for all $h \in H$, then $v = v'$. We write the operation in H additively, and denote the identity of H by 0. We call H and V, respectively, the *horizontal* and *vertical* components of the forest algebra. The idea is that H represents forests and V represents contexts. As was the case with the addition in H_A, this is not meant to suggest that H is a commutative monoid, although in all the applications in the present paper H will indeed be commutative. We require one additional condition: For each $h \in H$ there are elements $1 + h, h + 1 \in V$ such that for all $g \in H$, $(1 + h)g = g + h$, and $(h + 1)g = h + g$. A consequence is that every element $h \in H$ can be written as $h = v0$ for some $v \in V$, namely $v = h+1$. A homomorphism of forest algebras consists of a pair of monoid homomorphisms $(\alpha_H, \alpha_V) : (H, V) \to (H', V')$ such that $\alpha_H(vh) = \alpha_V(v)\alpha_H(h)$ for all $v \in V$ and $h \in H$.

We usually drop the subscripts on the component morphisms and simply write α for both these maps.

Of course, if A is a finite alphabet, then (H_A, V_A) is a forest algebra. The empty forest 0 is the identity of H_A, and the empty context $\Box$ is the identity of V_A. This is the *free forest algebra* on A, and we denote it A^Δ. It has the property that if (H, V) is any forest algebra and $f : A \to V$ is a map, then there is a unique homomorphism α from A^Δ to (H, V) such that $\alpha(a\Box) = f(a)$ for all $a \in A$.

4.2. **Recognition and syntactic forest algebra.** Given a homomorphism $\alpha : A^\Delta \to (H, V)$, and a subset X of H, we say that α *recognizes* the language $L = \alpha^{-1}(X)$, and also that (H, V) recognizes L. A forest language is regular if and only if it is recognized in this fashion by a *finite* forest algebra. Moreover, for every forest language $L \subseteq H_A$, there is a special homomorphism $\alpha_L : (H_A, V_A) \to (H_L, V_L)$ recognizing L that is minimal in the sense that α_L is surjective, and factors through every homomorphism that recognizes L. We call α_L the *syntactic morphism* of L, and (H_L, V_L) the *syntactic forest algebra* of L. If $s, s' \in H_A$, then $\alpha_L(s) = \alpha_L(s')$ if and only if for all $v \in V_A$, $vh \in L \Leftrightarrow vh' \in L$. This equivalence is called the *syntactic congruence* of L. An important fact in applications of this theory is that one can effectively compute the syntactic morphism and algebra of a regular forest language L from any automaton that recognizes L. (See [3].)

We say that a forest algebra (H_1, V_1) *divides* (H_2, V_2), in symbols $(H_1, V_1) \prec (H_2, V_2)$ if (H_1, V_1) is a quotient of a subalgebra of (H_2, V_2). In particular, (H_L, V_L) *divides* every forest algebra that recognizes L.

There is a subtle point in the definition of division of forest algebras given above that we will need to address. We have defined this in a way that directly generalizes the standard notion of division of monoids: A divisor of a monoid M is a quotient of a submonoid of M. But a forest algebra, is, in particular, a transformation monoid, and there is a second notion of division, which comes from the theory of transformation monoids, that will be particularly useful when we deal with wreath products: We say that (H, V) *tm-divides* (H', V') if there is a submonoid K of H', and a surjective monoid homomorphism $\Psi : K \to H$ such that for each $v \in V$ there exists $\hat{v} \in V'$ with $\hat{v}K \subseteq K$, and for all $k \in K$,

$$\Psi(\hat{v}k) = v\Psi(k).$$

Fortunately, the two notions of division coincide, as shown in the following Lemma.

Lemma 4.1. *Let (H_1, V_1) and (H_2, V_2) be forest algebras. $(H_1, V_1) \prec (H_2, V_2)$ if and only if (H_1, V_1) tm-divides (H_2, V_2).*

Proof. First suppose (H_1, V_1) divides (H_2, V_2). Then there is a submonoid V' of V_2 and a forest algebra homomorphism

$$\alpha : (V' \cdot 0, V') \to (H_1, V_1).$$

(Strictly speaking, we should reduce V' to the quotient that acts faithfully on $V' \cdot 0$, but leaving this reduction out does not change the argument.) Let $v \in V_1$, and set $\hat{v}$ to be any element of V' such that $\alpha(\hat{v}) = v$. We then have for $h \in V' \cdot 0$,

$$\alpha(\hat{v}h) = \alpha(\hat{v})\alpha(h) = v\alpha(h),$$

so (H_1, V_1) tm-divides (H_2, V_2).

Conversely, suppose (H_1, V_1) tm-divides (H_2, V_2), with underlying homomorphism $\alpha : H' \to H_1$. Let A be an alphabet at least as large as V_1, and let $\gamma : A \to V_1$ be an onto map. This extends, because of the universal property of the free forest algebra, to a (surjective) forest algebra homomorphism $\gamma : A^\Delta \to (H_1, V_1)$. We define $\delta : A \to V_2$ by setting

$$\delta(a) = \widehat{\gamma(a)}$$

for all $a \in A$, and consider its extension δ to a forest algebra homomorphism. It is enough to show that for $x, y \in V_A$, $\delta(x) = \delta(y)$ implies $\gamma(x) = \gamma(y)$. This will imply that γ factors through δ and give the required division.

Observe that if $s \in H_A$, then $\delta(s)$ is in the domain H' of α, because $s = x \cdot 0$ for some $x \in V_1$, and thus

$$\begin{aligned}
\gamma(s) &= \gamma(x)\gamma(0) \\
&= \gamma(x)\alpha(\delta(0)) \\
&= \alpha(\widehat{\gamma(x)}\delta(0)) \\
&= \alpha(\delta(x)\delta(0)) \\
&= \alpha(\delta(x \cdot 0)) \\
&= \alpha(\delta(s)).
\end{aligned}$$

So by assumption, we have

$$\gamma(a)\alpha(\delta(s)) = \alpha(\delta(a)\delta(s))$$

for all $s \in H_A$, $a \in A$. A straightforward induction on the number of nodes in x implies that for any $x \in V_A$,

$$\gamma(x)\alpha(\delta(s)) = \alpha(\delta(x)\delta(s)).$$

Now suppose $h \in H_1$ and $\delta(x) = \delta(y)$. As noted above, $h = \alpha(\delta(s))$ for some $s \in H_A$, and consequently

$$\begin{aligned}
\gamma(x) \cdot h &= \gamma(x)\alpha(\delta(s)) \\
&= \alpha(\delta(x)\delta(s)) \\
&= \alpha(\delta(y)\delta(s)) \\
&= \gamma(y)\alpha(\delta(s)) \\
&= \gamma(y) \cdot h.
\end{aligned}$$

Since h was arbitrary, we get $\gamma(x) = \gamma(y)$, by faithfulness.

□

4.3. **Wreath product.** Here we introduce the wreath product of forest algebras. We first try to give some intuition behind the construction. The wreath product originally arose in the theory of permutation groups, but it was subsequently adapted to provide an algebraic model of serial composition of automata. The idea is that the first automaton reads an input word $a_1 \cdots a_n$ beginning in state q_0. The second automaton sees both the run of the first automaton on this input string, as well as the original input string—that is, it reads the sequence

$$(q_0, a_1), (q_0 a_1, a_2), \ldots, (q_0 a_1 \cdots a_{n-1}, a_n)$$

as an input word, beginning in its initial state p_0. This defines a composite action of words over the original input alphabet A on pairs of states (p, q). The wreath product is, essentially, the transition monoid of this action.

The idea behind the wreath product of two forest algebras is also to model sequential composition. The first algebra 'runs' on an input forest, and then a second automaton runs on the same forest, but also gets to see to see the run of the first automaton. We will make this composition precise by defining the sequential composition of two homomorphisms. Assume that

$$\alpha : A^\Delta \to (G, W)$$

is a forest algebra homomorphism. For a forest t over A, let t^α be the forest over $A \times G$ obtained from t by changing the label of each node x from a to the pair (a, g), where $g \in G$ is the value assigned by α to the subforest of x. In other words, t^α is the forest $t[L_1, \ldots, L_k]$, where $G = \{g_1, \ldots, g_k\}$ and $L_i = \alpha^{-1}(g_i)$. The sequential composition, will use a second homomorphism that reads the relabeling t^α and yields a value in a second forest algebra; that is,

$$\beta : (A \times G)^\Delta \to (H, V) \ .$$

The sequential composition of α and β is the function $\alpha \otimes \beta : H_A \to G \times H$ defined by

$$t \quad \mapsto \quad (\alpha(t), \beta(t^\alpha)) \ .$$

The wreath product $(G, W) \circ (H, V)$ of forest algebras is defined to capture this notion of sequential composition. While it is hardly surprising that there is an algebraic construction that models sequential composition for forests, just as there is such a construction for words, it is rather remarkable that the construction for forest algebras is identical to the one used for transformation monoids. (In fact, one could even argue that the wreath product is better suited to forest languages, since it works directly on the forest algebra, while for word languages one goes from monoids to transformation monoids.)

We now present the definition of the wreath product of two forest algebras (H_1, V_1) and (H_2, V_2). This wreath product denoted by $(H_1, V_1) \circ (H_2, V_2)$.

Note that forest algebras are transformation monoids, for which the wreath product is a classical operation. We will apply the classical definition without changes in this setting, yielding some of the ingredients of a forest algebra, namely: 1) the carriers of the horizontal and vertical monoid; 2) the action of the vertical monoid on the horizontal monoid; and 3) the composition operation in the vertical monoid. The missing ingredient, not given by the classical definition, will be 4) the monoid operation in the horizontal monoid.

We describe below the classical definition of wreath product of transformation monoids, as applied to the special case of forest algebras. The states that are transformed, which in the case of forest algebras correspond to the horizontal monoid, are the cartesian product $H_1 \times H_2$ with component-wise addition. The transforming monoid, which in the case of forest algebra corresponds to the vertical monoid, is more sophisticated, its carrier set is $V_1 \times V_2^{H_1}$. The action of the transforming monoid $V_1 \times V_2^{H_1}$ on the transformed states $H_1 \times H_2$ is defined by

$$(v_1, f)(h_1, h_2) = (v_1 h_1, f(h_1)h_2).$$

The composition operation in the transforming monoid $V_1 \times V_2^{H_1}$ is defined by

$$(v, f) \cdot (v', f') = (vv', f'') \qquad f''(h) = (f(v'h)) \cdot (f'(h)).$$

As is well known, this definition turns $V_1 \times V_2^{H_1}$ into a monoid of faithful transformations on $H_1 \times H_2$. (Observe that since we define forest algebras using a left action of V on H, rather than a right action, our definition of the wreath product is the reverse of the customary one, with the first algebra in the composition written as the left-hand factor in the wreath product, rather than as the right-hand factor.)

By applying the definition of wreath product for transformation monoids, we have obtained most of the ingredients of forest algebra. We are missing the monoid operation on the horizontal monoid; for this we use the usual direct product.

The last missing condition is that for every element h of the horizontal monoid, a forest algebra should have elements $1 + h$ and $h + 1$ of the vertical monoid that satisfy

$$(1 + h)g = g + h \qquad \text{and} \qquad (h + 1)g = h + g.$$

We show that these elements exist in the wreath product. Let then $h = (h_1, h_2) \in H_1 \times H_2$. Consider the map $f : H_1 \to V_2$ that sends every element to $(1 + h_2)$. Then for any $g = (g_1, g_2) \in H_1 \times H_2$, we have

$$\begin{aligned}(1 + h_1, f)(g_1, g_2) &= ((1 + h_1)g_1, (1 + h_2)g_2) \\ &= (g_1 + h_1, g_2 + h_2) \\ &= (g_1, g_2) + (h_1, h_2).\end{aligned}$$

Therefore, the element $(1 + h_1, f)$ plays the role of $1 + (h_1, h_2)$. Similarly, we find $V_1 \times V_2^{H_1}$ contains the transformation $(h_1, h_2) + 1$.

Thus *the wreath product of two forest algebras is a forest algebra.*

Well-known properties of the wreath product of transformation semigroups and monoids carry over unchanged to this setting. In particular, the wreath product is associative, so we can talk about the wreath product of any sequence of forest algebras, and about the *iterated wreath product* of an arbitrary number of copies of a single forest algebra. Likewise, the direct product of two forest algebras embeds in their wreath product in either direction. As a consequence, if L_1, L_2 are recognized by forest algebras $(H_1, V_1), (H_2, V_2)$ respectively, then their union and intersection are both recognized by $(H_1, V_1) \circ (H_2, V_2)$.

The connection with sequential composition is given by:

Theorem 4.2. *For every pair of forest algebra homomorphisms*

$$\alpha : A^\Delta \to (G, W) \qquad \beta : (A \times G)^\Delta \to (H, V)\ .$$

there is a homomorphism into the wreath product $(G, W) \circ (H, V)$ that, when restricted to forests, is equal to the sequential composition

$$\alpha \otimes \beta : H_A \to G \times H.$$

Conversely, every homomorphism from a free forest algebra A^Δ into the wreath product of two forest algebras is realized in this manner by the sequential composition of two homomorphisms.

Proof. Given homomorphisms α, β as above, consider the map from A into the vertical monoid of $(G, W) \circ (H, V)$ given by

$$a \mapsto (\alpha(a\Box), f_a),$$

where for all $a \in A$, $g \in G$,

$$f_a(g) = \beta((a, g)\Box).$$

By the universal property of A^Δ, this map extends to a unique homomorphism γ with domain A^Δ. A straightforward induction on the construction of a forest $t \in H_A$ shows that $\gamma(t) = (\alpha(t), \beta(t^\alpha))$: The crucial step is when $t = as$ for some $a \in A$, $s \in H_A$. We then have $t^\alpha = (a, \alpha(s)) \cdot s^\alpha$, so that

$$\begin{aligned}
\gamma(t) &= \gamma(a) \cdot \gamma(s) \\
&= (\alpha(a\Box), f_a) \cdot (\alpha(s), \beta(s^\alpha)) \\
&= (\alpha(a\Box) \cdot \alpha(s), \beta(f_a(\alpha(s))) \cdot \beta(s^\alpha)) \\
&= (\alpha(as), \beta((a, \alpha(s)) \cdot \beta(s^\alpha)) \\
&= (\alpha(t), \beta(t^\alpha)).
\end{aligned}$$

Conversely, if $\gamma : A^\Delta \to (G, W) \circ (H, V)$ is a homomorphism, then for each $a \in A$, $\gamma(a\Box))$ has the form (w_a, f_a) for some $w_a \in W$, $f_a : G \to V$. We define homomorphisms

$$\alpha : A^\Delta \to (G, W) \qquad \beta : (A \times G)^\Delta \to (H, V) \ .$$

by setting, for each $a \in A$, $g \in G$,

$$\alpha(a\Box) = w_a \qquad \beta((a, g)\Box) = f_a(g) \ .$$

As we saw above, $\alpha \otimes \beta$ is the unique homomorphism mapping $a\Box$ to (w_a, f_a), so $\gamma = \alpha \otimes \beta$. □

5. Wreath Product Characterizations of Language Classes

When $\mathscr{A}$ is a class of forest algebras, we write $\mathsf{TL}[\mathscr{A}]$ for the class of languages recognized by iterated wreath products of forest algebras from $\mathscr{A}$. The following corollary to Theorem 4.2 justifies this notation.

Corollary 5.1. *Let $\mathscr{L}$ be the class of languages recognized by a class of forest algebras $\mathscr{A}$. Then $\mathsf{TL}[\mathscr{L}] = \mathsf{TL}[\mathscr{A}]$.*

We also say that $\mathscr{A}$ is an *algebraic base* of the language class $\mathsf{TL}[\mathscr{A}]$ (note that there may be several algebraic bases, just as there may be several language bases). We will now exhibit algebraic bases for the logics discussed in Section 3. By the above corollary, all we need to do is to provide, for each logic, a class of forest algebras that captures the language base. We could, of course, simply say that an algebraic base consists of the syntactic forest algebras of the members of the language base, but we prefer more explicit algebraic descriptions. These are given in the following theorem; the algebras used in the statement are described immediately afterwards, while the detailed proofs are not given until Section 7.

Theorem 5.2. *The logics EF, CTL, FO[$\prec$], CTL*, PDL and graded PDL have algebraic bases as depicted in Figure 2.*

We now proceed to describe the algebras mentioned in Figure 2. The bases have been chosen so that each base is either finite, or in the case it is an infinite class of algebras, then it has an effective characterization, i.e. there is an algorithm that checks if the syntactic algebra of a given forest language belongs to the base. Furthermore, the infinite algebraic

Logic	Algebraic base
EF	$\mathcal{U}_1$
CTL	$\mathcal{U}_2$
$FO[\prec]$	aperiodic path algebras
CTL*	distributive aperiodic algebras
PDL	distributive algebras
graded PDL	path algebras

Figure 2: Algebraic bases for temporal logics

bases are given by identities in the forest algebra, and therefore the algorithm reduces to checking if the identities hold.

First, we recall that an *aperiodic* finite monoid S is one that contains no nontrivial groups. Equivalently, there exists $m > 0$ such that $s^m = s^{m+1}$ for all $s \in S$. When we say that a forest algebra (H, V) is aperiodic, we mean that the vertical monoid V is aperiodic (which implies that H is aperiodic).

$\mathcal{U}_1$ is the forest algebra $(\{0, \infty\}, \{1, 0\})$, with $0 \cdot \infty = 0 \cdot 0 = \infty$. Note that since we use additive notation in the horizontal monoid, the additive absorbing element is denoted ∞, while the multiplicative absorbing element is 0. The vertical monoid of $\mathcal{U}_1$ is the unique smallest nontrivial aperiodic monoid, denoted U_1 in the literature. Another description of $\mathcal{U}_1$ is that it is the syntactic forest algebra of the forest language "some node with a" over an alphabet $A \ni a$ with at least two letters. If follows that every language in the language base of EF is recognized by $\mathcal{U}_1$, and every language recognized by $\mathcal{U}_1$ is a boolean combination of members of the language base of EF, so this algebra forms an algebraic base for EF.

$\mathcal{U}_2$ is the forest algebra $(\{0, \infty\}, \{1, c_0, c_\infty\})$ with $c_h \cdot h' = h$ for all horizontal elements h, h'. If one reverses the action from left to right and ignores the additive structure, $\mathcal{U}_2$ is the aperiodic unit in the Krohn-Rhodes Theorem. The underlying monoid of this transformation semigroup is usually denoted U_2. Every language recognized by $\mathcal{U}_2$ is a boolean combination of members of the language base of CTL, and all languages recognized by $\mathcal{U}_2$ are in CTL, so $\mathcal{U}_2$ forms an algebraic base for CTL.

So much for the singleton bases. We now describe the infinite bases.

A *distributive algebra* is a forest algebra (H, V) such that H is commutative and such that the action of V on H is distributive: $v(h_1 + h_2) = vh_1 + vh_2$ for all $v \in V$, $h_1, h_2 \in H$. The assertion that distributive algebras form algebraic bases for the given language classes is a consequence of the following theorem:

Theorem 5.3. *A forest language is a boolean combination of languages* $\mathsf{E}L$ *(respectively, languages* $\mathsf{E}L$ *with* L *first-order definable) if and only if it is recognized by a distributive forest algebra (respectively, an aperiodic distributive forest algebra).*

Let us define a *path language* to be any boolean combination of members of the language base of graded PDL, and an *fo path language* to be a boolean combination of members of the language base of $FO[\prec]$. We have the following analogue to Theorem 5.3.

Theorem 5.4. *A finite forest algebra* (H, V) *recognizes only path languages if and only if* H *is aperiodic and commutative and*

$$vg + vh = v(g + h) + v0 \tag{5.1}$$

$$u(g+h) = u(g+uh) \tag{5.2}$$

hold for all $g, h \in H$ and $u, v \in V$ with $u^2 = u$. (H, V) recognizes only fo-path languages if and only if H is aperiodic and commutative, V is aperiodic, and (H, V) satisfies the two identities above.

We define a *path algebra* to be a forest algebra (H, V) satisfying identities 5.1 and 5.2 with H aperiodic and commutative. We will give the proofs of Theorems 5.3 and 5.4 in Section 7.

Because of the connection with logic, we will call divisors of the six kinds of iterated wreath products described above EF-algebras, CTL-algebras, CTL*-algebras, FO-algebras, PDL-algebras, and graded PDL-algebras, respectively.

Note that for EF and CTL, the algebraic base has one algebra, while our other bases contain infinitely many algebras. This turns out to be optimal, as stated below.

Theorem 5.5 (Infinite base theorem)**.** *None of the language classes CTL*, FO[$\prec$], PDL, or graded PDL has a finite algebraic base.*

Proof. If a language class has an algebraic base consisting of a finite set of forest algebras

$$(H_1, V_1), \ldots, (H_k, V_k),$$

then it has a base containing just the single algebra

$$(H, V) = (H_1, V_1) \times \cdots \times (H_k, V_k).$$

This is because each of the (H_i, V_i) divides (H, V), and (H, V) embeds into the wreath product of the (H_i, V_i), in any order. Consequently, iterated wreath products of the (H_i, V_i) and iterated wreath products of (H, V) have the same divisors, and so recognize the same languages.

By these observations, it suffices to show that none of the classes in the statement of the theorem has an algebraic base consisting of a single forest algebra (H, V). We will give two different arguments for this, one applicable to the aperiodic classes CTL* and $FO[\prec]$, and the other for the nonaperiodic classes.

Suppose the language class $FO[\prec]$ is generated by a single algebra (H, V). Since (H, V) is required to recognize only languages in this class, V is aperiodic, and thus there is an integer n such that $v^n = v^{n+1}$ for all $v \in V$. We will show that no iterated wreath product of copies of (H, V) can recognize the language L_n consisting of all forests over $A = \{a, b, c\}$ in which there is a path from the root with the label in $(a^n b)^* c$. Since L_n is in $CTL^* \subseteq FO[\prec]$, this will give the desired conclusion also for CTL^*.

We prove this by induction on the number of factors k in the wreath product, showing that there are forests $s_k \in L_n$ and $t_k \notin L_n$, such that $\phi(s_k) = \phi(t_k)$ is satisfied for every homomorphism ϕ from A^Δ into the k-fold wreath product of (H, V). For $k = 1$, we can simply take $s_1 = a^n bc$ and $t_1 = a^{n+1} bc$. For the inductive step we suppose the claim holds for some $k \geq 1$, and let (G, W) denote the k-fold wreath product of the (H, V). Consider a homomorphism ϕ from A^Δ into the $(k+1)$-fold wreath product $(G, W) \circ (H, V)$. Recalling the definition of the wreath product we have $\phi : A^\Delta \to (G \times H, W \times V^G)$. If we compose ϕ with the projection onto the left coordinate we obtain a homomorphism ψ into (G, W). Note that since aperiodicity is preserved under wreath products, there is an m such that $w^m = w^{m+1}$ for all $w \in W$.

We first claim that if p and q are contexts in V_A such that $\psi(p) = \psi(q)$, then

$$\phi(p^n q^{m+1}) = \phi(p^{n+1} q^{m+1}).$$

To see this, first take (g_0, h_0) in $G \times H$. We have

$$\psi(q^m)g_0 = \psi(q^{m+1})g_0 = \psi(p)\psi(q^m)g_0,$$

so we have

$$\phi(q^{m+1})(g_0, h_0) = (g_1, h_1),$$

where $\psi(p)g_1 = g_1$. Let us write $\phi(p)$ as $(\psi(p), f)$, where $f : G \to V$. We then have

$$\phi(p^n)(g_1, h_1) = (g_1, f(g_1)^n h_1) = (g_1, f(g_1)^{n+1} h_1) = \phi(p^{n+1})(g_1, h_1).$$

Since g_0, h_0 are arbitrary, this proves $\phi(p^n q^{m+1}) = \phi(p^{n+1} q^{m+1})$, as claimed. We now make particular choices for p and q, namely

$$p = a\Box + bt_k, \quad q = a\Box + bs_k.$$

Since $\psi(s_k) = \psi(t_k)$, we have $\psi(p) = \psi(q)$, and thus by our claim above, $\phi(p^n q^{m+1}) = \phi(p^{n+1} q^{m+1})$. Set $s_{k+1} = p^n q^{m+1} \cdot 0$, and $t_{k+1} = p^{n+1} q^{m+1} \cdot 0$. So $\phi(s_{k+1}) = \phi(t_{k+1})$. For every path w from the root in s_k, there is a path in s_{k+1} with label $a^n bw$. On the other hand, for every path with a label $a^n bv$ from the root of t_{k+1} we have $v \in t_k$. Thus $s_{k+1} \in L_n$ and $t_{k+1} \notin L_n$, as claimed.

We now turn to the nonaperiodic case. Let p be a prime that does not divide the order of any group in (H, V), and let L be the set of forests over $\{a, b\}$ in which there is a path from the root of the form $a^m b$, where p divides m. We will show that (H, V) cannot recognize L. Since L has the form EK for a regular word language K, L is in PDL, so this will complete the proof.

It is easy to see that the vertical monoid of the syntactic forest algebra of L contains a group of order p: Let $0 \leq r < p$, and let H_r be the set of forests in which every path from the root has an initial segment of the form $a^j b$, where $r = j \bmod p$. Each H_r is a class of the syntactic congruence, all p of these classes are distinct, and the context $a\Box$ cyclically permutes them. On the other hand, the set of simple groups dividing a transformation monoid is preserved under wreath product, so no iterated wreath product of copies of (H, V) can contain a group of order p, and thus cannot recognize L.

□

6. **EF**

The logic EF was one of the first logics over trees to have a decidable characterization [3]. The result has been since then reproved several times with different methods [25, 13]. Here we give a new proof based on wreath product. Our argument is purely algebraic. It computes a decomposition based on the ideal structure of the underlying forest algebra.

The following theorem is proved in [3].

Theorem 6.1. *A forest language $L \subseteq H_A$ is defined by a forest formula of EF if and only if (i) H_L is idempotent and commutative, and (ii) for every $v \in V_L$, $h \in H_L$, we have $vh + h = vh$.*

Because this property can be effectively verified from the multiplication tables of H_L and V_L, we have an effective characterization of EF. More specifically, there is a decision procedure for determining whether or not a forest language given, say, by an automaton that recognizes it, is definable by a forest formula of EF. This procedure can also be adapted to testing whether a tree language is EF-definable with tree semantics.

In light of Theorem 5.2, Theorem 6.1 can be formulated as follows.

Theorem 6.2. *A forest algebra (H, V) divides an iterated wreath product of copies of $\mathcal{U}_1$ if and only if H is idempotent and commutative, and $vh + h = vh$ for all $h \in H$, $v \in V$.*

Note that Theorem 6.2 is purely algebraic. It makes no mention of trees, forests, languages or logic. This suggests that it might be proved reasoning solely from the structure of the forest algebra.

Here we present such a proof. The easy direction is to show that every divisor of an iterated wreath product of copies of $\mathcal{U}_1$ is horizontally idempotent and commutative and satisfies the identity $vh + h = vh$. Identities are always preserved under division, and obviously $\mathcal{U}_1$ itself satisfies the properties, so we just need to show that the properties are preserved under wreath product. Let (G, W) and (H, V) be forest algebras satisfying the identity, with G, H idempotent and commutative. The horizontal monoid of the wreath product is just $G \times H$, which is idempotent and commutative. Let $h = (h_0, h_1) \in (G, W)$, $v = (v_0, f) \in W \times V^G$ be horizontal and vertical elements of the wreath product. We have

$$\begin{aligned} vh + h &= (v_0, f)(h_0, h_1) + (h_0, h_1) \\ &= (v_0 h_0 + h_0, f(h_0)h_1 + h_1) \\ &= (v_0 h_0, f(h_0)h_1) \\ &= (v_0, f)(h_0, h_1) \\ &= vh. \end{aligned}$$

For the converse, we suppose (H, V) is horizontally idempotent and commutative and satisfies the identity. We prove by induction on $|H|$ that (H, V) divides an iterated wreath product of copies of $\mathcal{U}_1$.

Since H is idempotent and commutative, it is partially ordered by the relation $\leq$ defined by $h_1 \leq h_2$ if and only if $h_1 = h_2 + h$ for some $h \in H$. Transitivity and reflexivity of this relation are obvious. Antisymmetry follows from the observation that if $h_1 = h_2 + h \leq h_2$, then $h_1 + h_2 = h_2 + h + h_2 = h_2 + h = h_1$. Thus if we have both $h_1 \leq h_2$ and $h_2 \leq h_1$, then $h_1 = h_1 + h_2 = h_2$. This is just the standard $\mathcal{J}$-ordering, one of the Green relations, on the monoid H. Thus our identity $vh + h = vh$ implies $vh \leq h$ for all $v \in V$, $h \in H$. Conversely, if $vh \leq h$, then there is some $h' \in H$ such that $vh = h + h'$, and thus $vh + h = h + h' + h = h + h' = vh$. So we can replace the identity by the inequality $vh \leq h$ for all $v \in V$, $h \in H$.

The sum of all the elements of H is the (necessarily unique) absorbing element, which, following our usual practice, we denote ∞. This is the unique $\leq$-minimal element, since obviously $\infty + h = \infty$ for all $h \in H$. If $|H| \leq 2$, then (H, V) is either trivial, or isomorphic to $\mathcal{U}_1$, so we can assume $|H| > 2$. Thus there is at least one minimal element $h \neq 0$ in $H \setminus \{\infty\}$. We call such an element a *subminimal* element. It has the property that for all $v \in V$, $vh = h$ or $vh = \infty$.

For each subminimal h, we define H_h to be the *set* $\{\infty\} \cup \{g : h \in Vg\}$. Observe that H_h is a submonoid of H, because if $v_1 h_1 = h$ and $v_2 h_2 = h$, then

$$h = h + h = v_1 h_1 + v_2 h_2 + h_1 + h_2 = u(h_1 + h_2) \qquad \text{where } u = v_1 h_1 + v_2 h_2 + 1.$$

For $v \in V$ and $g \in H_h$ we set $v * g = vg$ if $vg \in H_h$, and otherwise set $v * g = \infty$. It is straightforward to verify that for all $v_1, v_2 \in V$, $g \in H_h$,

$$v_1 * g + g = v_1 * g,$$
$$(v_1 v_2) * g = v_1 * (v_2 * g),$$

so we get a well-defined action of V on H_h. We can collapse this action to make this faithful, and thus we get a well-defined forest algebra (H_h, V_h), that satisfies the hypotheses of the theorem. If there is more than one subminimal element, then each H_h has strictly smaller cardinality than H. Further, consider the map

$$\iota : (H, V) \to \prod (H_h, V_h),$$

where the direct product is over all subminimal elements h, defined by setting the h-component of $\iota(g)$ to be g if $g \in H_h$, and ∞ otherwise. It is straightforward to verify that ι is a homomorphism embedding (H, V) into the direct product. Since the direct product in turn embeds into the wreath product, we get the result by the inductive hypothesis.

It remains to consider the case where there is just one subminimal element h. In this case (H_h, V_h) is identical to (H, V). The elements of H different from ∞ form a submonoid G of H. We get a well-defined action $**$ of V on G by setting $v{**}g = vg$ if $g \in G$, and $v{**}g = h$ otherwise. Once again, the resulting forest algebra (G, W) satisfies the necessary identities, so by the inductive hypothesis (G, W) divides a wreath product of copies of $\mathcal{U}_1$. We complete the proof by showing that (H, V) embeds in the wreath product $(G, W) \circ \mathcal{U}_1$. We map $g \in H - \{\infty\}$ to $\alpha(g) = (g, 0)$ and ∞ to $\alpha(\infty) = (h, \infty)$. We further map $v \in V$ to (v, f_v), where $f_v(g) = 0$ if $vg = \infty$, and $f_v(g) = 1$, otherwise. This is obviously an injective homomorphism on the additive structure. To show that it is a homomorphism on the multiplicative structure, it suffices to show that for all $v \in V$, $g \in H$, $\alpha(vg) = (v, f_v)\alpha(g)$. There are several cases to consider. First, if $g = \infty$, then $vg = \infty$, so we have

$$\alpha(vg) = (h, \infty) = (v{**}h, f_v(h)\infty) = (v, f_v)(h, \infty) = (v, f_v)\alpha(g).$$

If $g \neq \infty$ but $vg = \infty$, we have

$$\alpha(vg) = (h, \infty) = (v{**}g, 0 \cdot 0) = (v{**}g, f_v(g) \cdot 0) = (v, f_v)(g, 0) = (v, f_v)\alpha(g).$$

Finally, if neither g nor vg is ∞, we have

$$\alpha(vg) = (vg, 0) = (v{**}g, 1 \cdot 0) = (v, f_v)(g, 0) = (v, f_v)\alpha(g).$$

□

Theorem 6.2 is the exact analogue for forest algebras of a Theorem of Stiffler [21] showing that a finite monoid is $\mathcal{R}$-trivial if and only if it divides a wreath product of copies of $\mathcal{U}_1$. Because of our conventions on the direction of the action, all our EF-algebras have $\mathcal{L}$-trivial, rather than $\mathcal{R}$-trivial vertical monoids.

7. Path Algebras and Distributive Algebras

In this section we prove Theorems 5.3 and 5.4.

7.1. **Distributive algebras.** We begin with Theorem 5.3, whose proof is significantly simpler than the proof of Theorem 5.4. Recall that a *distributive algebra* is a forest algebra (H, V) where H is commutative and which satisfies

$$v(h_1 + h_2) = vh_1 + vh_2 \ .$$

Note that instead of the two requirements, horizontal commutativity and the above identity, we could use a single identity

$$v(h_1 + h_2) = vh_2 + vh_1 \ ,$$

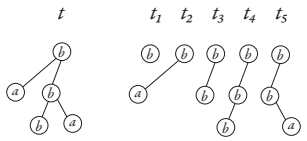
which, when $v = 1$, gives also horizontal commutativity. Nevertheless, we prefer separating the two conditions.

Theorem 5.3 says that a forest language is a boolean combination of languages $\mathsf{E}L$ (respectively, languages $\mathsf{E}L$ with L first-order definable) if and only if it is recognized by a distributive forest algebra (respectively, an aperiodic distributive forest algebra).

The "only if"part is fairly straightforward, applying any of the identities required from a distributive algebra does not change the set of paths in a tree. For the "if" part, only a little bit of effort is needed. The idea is that by applying the conditions on distributivity, one can show that if α is a homomorphism into a distributive algebra, then a forest is equal to the sum of its paths. More precisely, if t is a forest with nodes $x_1, \ldots, x_n$ then

$$\alpha(t) = \alpha(t_1 + \cdots + t_n) \tag{7.1}$$

where each tree t_i is obtained by taking the node x_i and removing all nodes from t that are not ancestors of x_i. This is depicted in the picture below.



Note that H, apart from being a commutative monoid, is also idempotent, by

$$h = (h + 1)(0 + 0) = (h + 1)0 + (h + 1)0 = h + h \ .$$

In particular, the value of

$$\alpha(t) = \alpha(t_1 + \cdots + t_n) = \alpha(t_1) + \cdots + \alpha(t_n)$$

does not depend on the order or multiplicity of types in the sequence $\alpha(t_1), \ldots, \alpha(t_n)$, and only on the set of values $\{\alpha(t_1), \ldots, \alpha(t_n)\}$. For each $g \in H$, we define the word language

$$L_g = \{a_1 \cdots a_i \in A^* : \alpha(a_1 \cdots a_i 0) = g\}.$$

It is not difficult to see that a forest t from (7.1) satisfies the formula $\mathsf{E}L_h$ if and only if one of the types $\alpha(t_1), \ldots, \alpha(t_n)$ is g. Combining the observations above, we conclude that for every $h \in H$

$$\alpha(t) = g \qquad \text{iff} \qquad \bigvee_{\substack{G \subseteq H \\ h = \sum_{g \in G} g}} \Big(\bigwedge_{g \in G} \mathsf{E}L_g \quad \wedge \quad \bigwedge_{g \in H - G} \neg \mathsf{E}L_g \Big).$$

Furthermore, if the monoid V is aperiodic, each word language L_h, as a word language recognized by V, is first-order definable by the McNaughton-Papert theorem.

7.2. **Path algebras.** We now proceed to prove Theorem 5.4. We use the term *path algebra* for a forest algebra that satisfies the conditions in the theorem, namely that the horizontal monoid is aperiodic and commutative, along with identities identities (5.1) and (5.2), which we recall here

$$vg + vh = v(g + h) + v0 \tag{5.1}$$

$$u(g + h) = u(g + uh) \qquad \text{for } u \text{ s.t. } u^2 = u \tag{5.2}$$

Recall that a path language is a boolean combination of the languages from the base of graded PDL: languages of the form "at least k paths in L, for some regular L. A first-order definable path language is defined similarly but L is required to be definable in $FO_A[<]$.

Theorem 5.4 says that a forest language is a path language (respectively, a first-order definable path language) if and only if it is recognized by a path algebra (respectively, a vertically aperiodic path algebra).

The "only if" part is simple; the identities are designed to hold in any syntactic algebra of a path language (respectively, a first-order definable path language). The rest of this section is devoted to showing the "if" implication of the theorem.

For the moment, we concentrate on path algebras, as opposed to aperiodic path algebras. After doing the proof, we show how it can be modified to obtain the case for aperiodic path algebras.

We begin with the following lemma, which illustrates the significance of identity (5.1). When speaking of paths, we refer to paths that begin in one of the roots of a forest, but that end in any node, not necessarily a leaf.

Lemma 7.1. *Forests with the same multisets of paths have the same image under any homomorphism into an algebra satisfying* (5.1) *and horizontal commutativity.*

Proof. We will show that two forests with the same multisets of paths are equal in the quotient of the free forest algebra under the identities (5.1) and $h + g = g + h$. In other words, we show that if (5.1) and $h + g = g + h$ are treated as rewriting rules on real forests (and not elements of the forest algebra), then each two forests with the same multisets of paths can be rewritten into each other. The idea is to transform each forest into a normal form, such that the normal form is uniquely determined by the multiset of paths. The transformation into normal form works as follows. Let t be a forest. Let $a_1, \ldots, a_n$ be the labels that appear in the roots of t. By applying horizontal commutativity, the forest t is rewritten into a forest

$$\sum_i a_i t_{i,1} + a_i t_{i,2} + \cdots + a_i t_{i,n_i}.$$

By applying the identity (5.1) and horizontal commutativity, the above is rewritten into

$$\sum_i a_i(t_{i,1} + t_{i,2} + \cdots + t_{i,n_i}) + \overbrace{a_i 0 + \cdots + a_i 0}^{(n_i - 1) \text{ times}}$$

Finally, for each i, we rewrite the forest $t_{i,1} + t_{i,2} + \cdots + t_{i,n_i}$ into normal form. The result of this rewriting is a forest where every two different non-leaf nodes have a different sequence of labels on their paths. Such a forest, modulo commutativity, is uniquely determined by the multiset of paths. □

The above lemma shows membership in a language L recognized by an algebra satisfying (5.1) is uniquely determined by the multiset of paths in a forest. However, this on its own does not mean that L is a path language (otherwise, we would not need identity (5.2)), as witnessed by the following example.

Example. Consider the language $a^*(a+a)$. A forest belongs to this language if and only if for some $n \in \mathbb{N}$, the multiset of paths is

$$\epsilon, a, a^2, \ldots, a^n, a^{n+1}, a^{n+1}$$

This language is not a path language. It does not even belong to a quite general class defined below. Let $\alpha : A^* \to M$ be a morphism from words into a finite monoid. The α-profile of a forest s is a vector in $\mathbb{N}^M$ that says, for each $m \in M$, how many times a path with value m appears in the forest s. A language is called *path-profile testable* if for some morphism α, membership in the language is uniquely determined by the α-profile of a forest. It is not difficult to see that the language $a^*(a+a)$ is not even path-profile testable, since a path-profile testable language will confuse $a^n(a+a^na)$ with $a^na^n(a+a)$ for certain large values of n (more precisely for $n = \omega$, the notion of ω will be defined below).

We now return to proving the "if" implication in Theorem 5.4. The theorem follows immediately from the proposition below, by taking v to be the empty context.

Proposition 7.2. *Let (H, V) be a path algebra. For any $v \in V$ and $h \in H$, the forest language $\{t : v\alpha(t) = h\}$ is a path language.*

For the rest of this section we fix a path algebra (H, V) and a homomorphism $\alpha : A^\Delta \to (H, V)$. For a tree t we will often refer to $\alpha(t)$ as *type of* t. Similarly for contexts.

We will prove the proposition by induction on the size of the set $vV \subseteq V$. We write $v \sim w$ if $vV = wV$ (this is Green's $\mathcal{R}$-equivalence in the context monoid).

Apart from Green's relations, we will also use the ω power from monoid theory. For a finite monoid – in this case, the monoid is V – we define ω to be a number such that v^ω is idempotent for any $v \in V$. Such a number always exists in a finite monoid, it suffices to take ω to be the factorial of the size of V.

The induction base is when the set vV is minimal.

Lemma 7.3. *If vV is minimal, then the context v is constant, which means that $vg = vh$ holds for every $g, h \in H$*

Proof. Let $h_1, \ldots, h_n$ be all elements of H. Consider the context

$$w = v(h_1 + \ldots + h_n + 1).$$

We show that the context w is constant. It suffices to show that $wh = w0$ for every $h \in H$. Because $h_1, \ldots, h_n$ contains all elements of H, then there must be some i such that h_i is $\omega \cdot h$, which is defined by

$$\omega \cdot h = \overbrace{h + \cdots + h}^{\omega \text{ times}}.$$

By aperiodicity of H, we know that $h_i + h = h_i$. By commutativity of H, we see that

$$h_1 + \cdots + h_n = h_1 + \cdots + h_n + h$$

and therefore $wh = w0$. We have thus established that w is constant. Since w is constant, one can easily see that $wu = w$ holds for every $u \in V$, and therefore $wV = \{w\}$. Since

$w \in vV$, it follows that $wV \subseteq vV$. By minimality of vV, we infer that $vV = \{w\}$. Because V contains an identity context, it follows that $v \in vV$ and therefore $w = v$, and therefore v is constant. □

When the context v is constant, the language in the proposition is either empty, or all forests, in either case it is a path language.

We now proceed to the induction step. We fix v and h as in the statement of the proposition.

A *path context* is a context of the form $a_1 \cdots a_n \Box$. A *preserving context* is a context p whose type satisfies $v\alpha(p) = v$, for our fixed v. A forest is called *negligible* if it is a concatenation of trees of the form $p0$, where p is a preserving context.

Lemma 7.4. *If $vu \sim v$ then $v = v(1 + u0)$. In particular, if g is the type of a negligible forest, then $v = v(g + 1)$.*

Proof. In the proof, we will use the identity

$$v^\omega = v^\omega(1 + v^\omega 0) \ . \tag{7.2}$$

Note that by iterating the above ω times, we get

$$v^\omega = v^\omega(1 + \omega \cdot v^\omega 0) \ . \tag{7.3}$$

In the above, $\omega \cdot v^\omega 0$, or more generally $n \cdot h$ for any number n and forest type h, denotes the n-fold sum $h + \cdots + h$. The proof of (7.2) is by applying the identity (5.2) from the definition of path algebras:

$$v^\omega g = v^\omega(g + 0) = v^\omega(g + v^\omega 0) \ .$$

We now proceed to prove the lemma. If $vu \sim v$ then $vuw = v$ for some w. We can assume that uw is idempotent, by replacing uw with $(uw)^\omega$. By the identity (7.3), we get

$$v = vuw = v(uw)(uw + \omega uw0) = v(uw + \omega uw0) \ .$$

Let $x = (uw + \omega uw0)$. We know that $x = x + \omega uw0$, and therefore also $x^\omega = x^\omega + \omega uw0$.

$$v = vx^\omega = vx^\omega(x^\omega + \omega uw0) \ .$$

By applying (5.2) the above becomes

$$vx^\omega(1 + \omega uw0) = v(1 + \omega uw0).$$

If we can show that $\omega uw0 = \omega uw0 + u0$, then we would be done, by

$$v = v(1 + \omega uw0) = v(1 + \omega uw0 + u0) = v(1 + u0).$$

It remains to show $\omega uw0 = \omega uw0 + u0$:

$$\omega uw0 = \omega uw0 + \omega uw0 \overset{(5.1)}{=} \omega u(w0 + w0) + \omega u0$$

and the last expression is clearly invariant under adding $u0$. □

A *guarded context* is a path context $pa\Box$ where the prefix $p\Box$ is preserving, but the whole context $pa\Box$ is not. A forest is in *guarded form* if it is a concatenation of trees of the form pat, where $pa\Box$ is a guarded context. The following lemma shows that, up to negligible forests, each forest has the same multiset of types as some guarded context.

We say two forests t, t' are *negligibly equivalent* if for some negligible forests s and s', the forests $t + s$ and $t' + s'$ have the same multiset of paths. This is indeed an equivalence relation (it is transitive since a concatenation of negligible forests is also negligible).

Lemma 7.5. *Each forest is negligibly equivalent to a guarded forest.*

Proof. Let t be a forest. For each node x in t, let q_x be the path context $a_1 \cdots a_m\Box$ obtained by reading the path that leads to x inside t, including the node x (which has the last label a_m). Let X be the set of nodes x for which the context q_x is a guarded context, in particular, $q_x = p_x a_x \Box$, with p_x a preserving path context and $a_x \in A$. Note that the set X is an antichain: a node $x \in X$ is chosen as the first time when the path leading to x stops preserving v. For each $x \in X$, let $t|_x$ be the subtree of the node x, the subtree includes x. Let t' be the forest obtained from t by removing all subtrees $t|_x$, for $x \in X$. The forests

$$t + \sum_{x \in X} p_x 0 \qquad \text{and} \qquad t' + \sum_{x \in X} p_x t|_x$$

clearly have the same multisets of paths. Since $\sum_{x \in X} p_x 0$ is a negligible forest, and $\sum_{x \in X} p_x t|_x$ is a guarded forest, it remains to prove that t' is negligibly equivalent to the empty forest. But this follows since all paths inside t' correspond to preserving contexts, by construction of t' □

A path language L is called *guarded* if it is invariant under concatenation with negligible forests, i.e.

$$t + s \in L \qquad \text{iff} \qquad t \in L$$

holds for any negligible forest s.

Lemma 7.6. *For any $h \in H$ there is a guarded path language L_h such that for any guarded forest t,*

$$t \in L_h \qquad \textit{iff} \qquad v\alpha(t) = vh \ . \tag{7.4}$$

Before showing the lemma above, we show in the lemma below that it concludes the proof of Proposition 7.2.

Lemma 7.7. *Let h and L_h be as in Lemma 7.6. Then the equivalence (7.4) holds for all forests t, and not only guarded forests.*

Proof. Let t be a forest. By applying Lemma 7.5, we can find negligible forests s, s' and a guarded forest t' such that $t + s$ and $t' + s'$ have the same multiset of paths.

We begin with the left to right implication in (7.4). Assume that $t \in L_h$. Since s is negligible and the language L_h is guarded, L_h also contains $t + s$. Since L_h is a path language, and the forests $t + s$ and $t' + s'$ have the same multiset of paths, then L_h also contains $t' + s'$. We now apply Lemma 7.6 to conclude that $v\alpha(t' + s') = vh$. Since $t' + s'$ and $t + s$ have the same multiset of paths, they have the same value under α by Lemma 7.1. This gives us

$$vh = v\alpha(t' + s') = v\alpha(t + s) = v\alpha(t) \ ,$$

where the last equality is by Lemma 7.4.

The right to left implication is by reversing the above reasoning. □

7.3. **The path language L_h.** We are only left with proving Lemma 7.6.

We say that two types g, h are $v+$-equivalent if $vug = vuh$ holds for any context $vu \not\sim v$.

Lemma 7.8. *For every $h \in H$, there is a path language M_h that contains all forests whose type is $v+$-equivalent to h.*

Proof. Define

$$W = \{w \in vV : w \not\sim v\}$$

By definition, a forest type $g \in H$ is $v+$-equivalent to h if and only if $wg = wh$ holds for all $w \in W$. By the induction assumption in Proposition 7.2, we know that for every $w \in W$ and $g \in H$, the forest language

$$L_{w,g} = \{t : w \cdot \alpha(t) = g\}$$

is a path language. The type of a forest t is $v+$-equivalent to h if for every context $w \in W$, the result of placing t in a context of type w is the same as the result of placing h in w. In other words, the set of forests whose type is $v+$-equivalent to h is

$$\bigcap_{w \in W} L_{w,wh},$$

which is a path language, as an intersection of path languages. □

Below, we write *guard* for the set of pairs $(w, a) \in V \times A$ such that $vw \sim v$ but $vw\alpha(a) \not\sim v$. In other words, a pair $(w, a) \in guard$ describes a guarded context pa. Consider a forest in guarded form

$$t = p_1 a_1 t_1 + \cdots + p_n a_n t_n \ .$$

For each $(w, a) \in guard$, let $I_{w,a}$ be the set of indexes i such that $\alpha(p_i) = w$ and $a_i = a$. The *guarded profile* of this forest is the function

$$\tau_t \quad : \quad guard \quad \to \quad \{0, 1, \ldots, \omega\} \times [H]_{\equiv_{v+}}$$

which maps a pair (w, a) to the pair (n, h), where n is the size of $I_{w,a}$ (up to threshold ω) and h is the equivalence class

$$[\alpha(\sum_{i \in I_{w,a}} t_i)]_{\equiv_{v+}} \ .$$

Lemma 7.6, and thus also Proposition 7.2 and Theorem 5.4, will follow from the two lemmas below.

Lemma 7.9. *For a guarded forest t, the guarded profile determines the value $v\alpha(t)$. In other words, if s, t are guarded forests with the same guarded profile, then $v\alpha(s) = v\alpha(t)$.*

Lemma 7.10. *For a guarded forest, the guarded profile can be determined by a path language. In other words, for each guarded profile τ, there is a path language L_τ such that $t \in L_\tau \Leftrightarrow \tau_t = \tau$ holds for all guarded forests t.*

The above two lemmas give us Lemma 7.6, by taking L_h to be the union of all L_τ, for profiles τ of the form $\tau = \tau_t$ where t is a guarded forest with $v\alpha(t) = h$. We begin with the proof of Lemma 7.9.

Proof. (of Lemma 7.9) Let s, t be guarded forests with the same guarded profile. Our goal is to show that $v\alpha(s) = v\alpha(t)$.

Let τ be the guarded profile of s and t. Let $(w_1, a_1), \ldots, (w_m, a_m)$ be all elements of *guard*, and let (k_i, x_i) be the value $\tau(w_i, a_i)$. Recall that k_i is the number of times a path of the form pa_i appears in the forest with $\alpha(p) = w_i$. By repeatedly applying (5.1), horizontal commutativity and aperiodicity, we know that the type of s is

$$\alpha(s) = \quad \sum_{i:k_i \geq 1} w_i \alpha(a_i) h_i \quad + \quad \sum_i (k_i - 1) \cdot w_i \alpha(a_i) 0 \ ,$$

for some $h_1, \ldots, h_n \in H$ such that each h_i belongs to the $v+$-equivalence class x_i. Likewise, we can decompose

$$\alpha(t) = \quad \sum_{i:k_i \geq 1} w_i \alpha(a_i) g_i \quad + \quad \sum_i (k_i - 1) \cdot w_i \alpha(a_i) 0 \ .$$

So the only difference between the types of s and t is that the first type uses $h_1, \ldots, h_n$ and the second type uses $g_1, \ldots, g_n$. However, we know that the types h_i and g_i are $v+$-equivalent, for any i. We will conclude the proof by showing that

$$v(w_i \alpha(a_i) h_i + h) = v(w_i \alpha(a_i) g_i + h)$$

holds for any $h \in H$. By applying the equality above for all $i = 1, \ldots, n$, we get the desired $v\alpha(s) = v\alpha(t)$. By definition of $v+$-equivalence, the above equality would follow if we showed that $v(v_i \alpha(a_i) \Box + h) \not\sim v$. This will be shown in Lemma 7.11. □

Lemma 7.11. *If $vu \not\sim v$ then $v(u + h) \not\sim v$.*

Proof. Toward a contradiction, assume that w is such that

$$v(uw + h) = v$$

We assume that $(uw + h)$ is idempotent. By (7.3), we get

$$v = v(uw + h)(uw + h + \omega(uw0 + h))$$

$$v(uw + h + \omega(uw0 + h)) = v(uw + \omega(uw0 + h))$$

Let $x = uw + \omega(uw0 + h)$. By the above we know that $vx = v$. By definition of x we know that $x = x + h$, and in particular $x^\omega = x^\omega + h$. By identity (5.2), we get

$$x^\omega = x^\omega(x^\omega + h) = x^\omega(1 + h)$$

Therefore,

$$v = vx = vx^\omega = vx^\omega(1 + h) = v(1 + h)$$

□

Now we proceed to prove Lemma 7.10.

Proof. We will show that for each

$$(w,a) \in guard \quad \text{and} \quad (i,x) \in \{0,1,\ldots,\omega\} \times [H]_{\equiv_{v+}}$$

there is a path language $L_{(w,a),(i,x)}$ such that

$$t \in L_{(w,a),(i,x)} \qquad \Leftrightarrow \qquad \tau_t(w,a) = (i,x)$$

holds for any guarded forest t. This gives Lemma 7.10 by setting

$$L_\tau = \bigcap_{(w,a) \in guard} L_{(w,a),\tau(w,a)} \ .$$

Fix (w,a) and (i,x). The easier part is to enforce that the first coordinate of $\tau_t(w,a)$ is i: we just have to say that the forest t has i paths in the word language

$$K_{w,a} = \{a_1 \cdots a_n a \in A^+ : \alpha(a_1 \cdots a_n \Box) = w\} \ .$$

Only slightly more effort is required in enforcing that the second coordinate of $\tau_t(w,a)$ is x. By Lemma 7.8, we know that the set M_x of forests whose type is in the $v+$-equivalence class x is defined by a boolean combination of path formulas. To enforce that the second coordinate of $\tau_t(w,a)$ is x, we use the same boolean combination, except that every word language is prefixed by $K_{w,a}$. □

As we promised before, we now prove that if the path algebra (H,V) in the statement of Theorem 5.4 is vertically aperiodic, then the path language only needs to use first-order definable word languages. It suffices to look at the only place where we actually wrote word languages: in the lemma above. The word language $K_{w,a}$ is a word language obtained by concatenating a to a word language that is recognized by the vertical monoid V, via the morphism $a \mapsto \alpha(a\Box)$. Since V is aperiodic, we can use the Schützenberger and McNaughton-Papert theorem to conclude that $K_{w,a}$ is first-order definable.

Actually, the argument above can be further generalized to any variety of word languages given by monoids such that the corresponding language class is closed under concatenation and contains the one letter languages $\{a\}$. Note that any such language class necessarily contains all first-order logic, since it captures all star-free expressions.

8. Multicontexts and Confusion

Here we find necessary conditions for a forest algebra to be a CTL-algebra, an FO-algebra or a graded PDL-algebra. We use these conditions to show that certain languages cannot be expressed in CTL, FO, or PDL. The conditions we find are essentially the absence of certain kinds of configurations in the forest algebra, analogous to the 'forbidden patterns' of Cohen-Perrin-Pin [8] and Wilke [24].

Let A be a finite alphabet. A *multicontext* p over A is a forest in which some of the leaves have been replaced by a special symbol $\Box$, each occurrence of which is called a *hole* of the multicontext. A special kind of multicontext, called a *uniform* multicontext, is one in which every leaf node is a hole, and all subtrees at the same level are identical. For example

$$a(b(c\Box + c\Box)) + a(b(c\Box + c\Box))$$

is a uniform multicontext.

The holes are used for substitution. The holes are independent in the sense that different forests can be substituted into different holes. The set of holes of a multicontext p is denoted holes(p). A *valuation* on p is a map $\mu : \text{holes}(p) \to X$, where X can be a set of forests, or of multicontexts, or elements of H, where (H, V) is a forest algebra. The resulting value, $p[\mu]$, found by substituting $\mu(x)$ for each hole x, is consequently either a multicontext, a forest, or an element of H. In the last case, we are assuming the existence of a homomorphism $\alpha : A^\Delta \to (H, V)$, evaluated at the nodes of p.

Given a set $G \subseteq H$ we write $p[G]$ for the set of all possible values of $p[\mu]$ where $\mu : \text{holes}(p) \to G$. When $G = \{g\}$ is a singleton, we just write $p[g]$. For $g \in G$ and $x \in \text{holes}(p)$ we define $p[g/x]$ to be the multicontext that results from p by putting a tree that evaluates to g in the hole x. (In particular, $p[g/x]$ has one less hole than p.)

We now define the various type of forbidden patterns for forest algebra.

8.1. **Horizontal confusion.** Let (H, V) be a forest algebra. As above, we assume the existence of a homomorphism from A^Δ into (H, V) in order to define the valuations on p with values in H. We say that (H, V) has *horizontal confusion* with respect to a multicontext p and a set $G \subseteq H$ with $|G| > 1$ if for every $g \in G$ and $x \in \text{holes}(p)$:

$$G \subseteq p[g/x][G].$$

Intuitively, this means that fixing the value of one of the holes of p still allows us to obtain any element of G by putting suitable elements of G into the remaining holes.

8.2. **k-ary horizontal confusion.** We can define a stronger version of confusion, which seems to be satisfied by fewer forest algebras. In the stronger version, we are allowed to fix the value in not just one, but in $k \geq 1$ holes: We say that the forest algebra (H, V) has *k-ary horizontal confusion* with respect to a multicontext p and a set $G \subseteq H$, with $|G| > 1$, if for all $g_1, \ldots, g_k \in G$ and $x_1, \ldots, x_k \in \text{holes}(p)$,

$$G = p[g_1/x_1, \cdots, g_k/x_k][G].$$

The following lemma shows that the stronger notion is in fact equivalent to horizontal confusion, because we can always amplify horizontal confusion to k-ary horizontal confusion for arbitrary k.

Lemma 8.1. *Suppose (H, V) has horizontal confusion with respect to a multicontext p and a subset G of H, with underlying homomorphism $\phi : A^\Delta \to (H, V)$. Let $k > 0$. Then there is a multicontext p_k such that (H, V) has k-ary horizontal confusion with respect to p_k, G and ϕ.*

Proof. We prove this by induction on k. We have $p_1 = p$, by hypothesis. If $k > 1$, we define p_k by placing a copy of p_{k-1} in each of the holes of p. To see that this works, fix the values in G of k of the holes holes of p_k. If the k holes do not all belong to the same copy of p_{k-1}, then each copy has fewer than $k - 1$ holes fixed, and thus we can set the values in the remaining holes to get any elements of G we want in the holes of p, and consequently any element of G as a value of p_k. If the k holes all belong to the same copy of p_{k-1}, then the resulting value $g \in G$ produced by this copy might be determined, but this will only constrain the value in one of the holes of p. Since p has horizontal confusion, we can set the remaining holes of p to values $g_1, \ldots, g_r$ to obtain any desired value as output, and we can in turn set the values of the other copies of p_{k-1} to obtain these values $g_1, \ldots, g_r$. □

8.3. **Vertical confusion.** We say that the forest algebra (H, V) has *vertical confusion* with respect to a multicontext p and a set $\{g_0, \ldots, g_{k-1}\} \subseteq H$ with $k > 1$ if for every $i = 0, \ldots, k-1$:

$$p[g_i] = g_j \text{ where } j = (i+1) \pmod{k}.$$

This condition is weaker than periodicity of vertical monoid, because p is a multicontext, and not just a context. For instance, consider the syntactic forest algebra of the tree language L, which consists of trees where every node has two or zero children, and where every leaf is at even depth.

8.4. **Confusion Theorem.** The next theorem shows how the various types of confusion are forbidden in CTL-, FO- and PDL-algebras.

Theorem 8.2 (Confusion Theorem)**.**

- *If (H, V) is a CTL-algebra, it does not have vertical confusion with respect to any multicontext.*
- *If (H, V) is an FO-algebra, it does not have vertical confusion with respect to any uniform multicontext.*
- *If (H, V) is a graded PDL-algebra, it does not have horizontal confusion with respect to any multicontext.*

Proof. For each of the three kinds of confusion and each of the corresponding language classes, we will show that the nonconfusing property *(a)* holds for the elements of the algebraic base of the class, *(b)* is preserved by wreath products, and *(c)* is preserved by quotients and subalgebras.

We begin with vertical confusion and the class CTL which has $\mathcal{U}_2$ as an algebraic base. Let $\phi : A^\Delta \to \mathcal{U}_2$ be a homomorphism, and suppose p is a multicontext over A such that U_2 has vertical confusion with respect to p and ϕ. Since $\mathcal{U}_2$ is distributive, we have

$$p[g] = \sum_{u \in \pi} \phi(u) g + \sum_{v \in \rho} \phi(v) \cdot 0,$$

where the first sum ranges over the set π of paths in p from a root to the parent of a hole, and the second over the set ρ of paths from the root to a leaf. We claim that for $g \in \{0, \infty\}$, $p[p[g]] = p[g]$. This follows easily from an enumeration of the possible cases: If $g = p[g]$, then the claim is trivial, so we can assume that either $g = \infty$ and $p[g] = 0$, or $g = 0$ and $p[g] = \infty$. In the first case, every path in π has a prefix wa with $a \in A$, $\phi(a) = c_0$, and $\phi(b) = 1$ for every letter b of w, and every path in ρ has either this form or has $\phi(b) = 1$ for every letter b. It follows that $p[0] = 0$. In the second case, some path in p has a prefix wa with $\phi(a) = c_\infty$ and $\phi(b) = 1$ for every letter b of w, and thus $p[\infty] = \infty$. Since $p[p[g]] = p[g]$ for all g in the horizontal monoid of $\mathcal{U}_2$, we cannot have vertical confusion.

We now consider the base algebras for $FO[\prec]$. Suppose that $\phi : A^\Delta \to (H, V)$ is a homomorphism into an aperiodic path algebra. Then, by Theorem 5.4, every language recognized by ϕ is an fo path language—that is, a boolean combination of languages of the form $E^k L$, where $L \subseteq A^*$ is a first-order definable word language. Let p be a uniform multicontext over A. Since p is uniform, every maximal path in p has the same label $u \in A^*$. We can dispense with the case where p has a single hole, because then $p[g]$ reduces to $\phi(u) \cdot g$, and by aperiodicity of the vertical monoid we have, for some $n \geq 0$, $p^{n+1}[g] = \phi(u^{n+1})g = \phi(u^n)g = p^n[g]$, so there is no vertical confusion. We thus suppose that p has at least two holes, so that p^n is a multicontext with at least 2^n holes. Since every language recognized

by ϕ is an fo path language, there exists a congruence $\sim$ of finite index on A^* and an integer $k > 0$ such that $A^*/\sim$ is aperiodic, with the following property: If $s, t \in H_A$ are such that for every $\sim$-class κ, the number of paths from the root of s in κ is equal, up to threshold k, to the number of paths from the root of t in κ, then $\phi(s) = \phi(t)$. ('Equal up to threshold k' means either equal, or both at least k.) Since $A^*/\sim$ is aperiodic, there is an integer r such that $u^r \sim u^{r+1}$. Let $g \in H$, and let s be any forest such that $\phi(s) = g$. Choose q such that both $q > r$ and $2^q > k$. Now consider the forests $p^{q+1}[s]$ and $p^{q+2}[s]$. Suppose that a word occurs as the label of a path from the root in $p^{q+2}[s]$ more times than it does in $p^{q+1}[s]$. Then, since p is uniform, the word must have the form $u^{q+1}v$, and since $u^{q+1}v \sim u^q v$, a word in the same $\sim$-class occurs at least $2^q > k$ times in $p^{q+1}[s]$. It follows that $p^{q+1}[g] = \phi(p^{q+1}[s]) = \phi(p^{q+2}[s]) = p^{q+2}[g]$, so there is no vertical confusion.

We now consider the base algebras for graded PDL, so we suppose $\phi : A^\Delta \to (H, V)$ is a homomorphism onto a path algebra (H, V) which has horizontal confusion with respect to a multicontext p and a set $G \subseteq H$, with $|G| > 1$. Let $G = \{g_1, \ldots, g_n\}$, and let $s_1, \ldots, s_n$ be forests such that $\phi(s_i) = g_i$ for all $1 \leq i \leq n$. As above, there is a congruence $\sim$ of finite index on A^* and an integer k, such that if two forests agree on the number of paths threshold k and modulo $\sim$, then they have the same image under ϕ. Let m be the index of $\sim$. (The only difference from the previous case is that we no longer have $A^*/\sim$ aperiodic.) By Lemma 8.1, there is a context q such that (H, V) has km-ary horizontal confusion with respect to q. We order the classes of $\sim$ arbitrarily as $\kappa_1, \ldots, \kappa_m$. We proceed to insert forests from $s_1, \ldots, s_n$ into the holes of q according to the following algorithm: For each κ_i in turn, we ask if there is a way to substitute copies of the s_j into the holes we have not yet filled in order to obtain at least k paths in κ_i. If so, we perform the necessary insertions; if not we insert enough copies of the s_j to obtain the maximum possible number of paths in κ_i. At the end of the process, we will have filled no more than km holes. However, no further substitution of forests s_j for the remaining holes can increase the number, threshold k of paths in any class of $\sim$, and thus no matter how we fill the remaining holes, the value under ϕ will be the same. But because of the km-ary confusion, we should be able to obtain any value in G by appropriately filling the remaining holes. Thus $|G| = 1$, so there is no horizontal confusion.

We now show closure under wreath product. Suppose first that neither (H_1, V_1) nor (H_2, V_2) has vertical confusion with respect to any multicontext. Let γ be a homomorphism from A^Δ into the wreath product $(H, V) = (H_1, V_1) \circ (H_2, V_2)$. Suppose (H, V) has vertical confusion with respect to some multicontext p with underlying homomorphism γ. There thus exist $g_i = (g_i^{(1)}, g_i^{(2)}) \in H = H_1 \times H_2$, with $i = 0, \ldots, n-1$, such that $p_\gamma[g_i] = p_\gamma[g_{(i+1) \bmod n}]$ for $0 \leq i < n$. (Note that here we explicitly indicate the homomorphism γ, since we will be shortly be applying the multicontext p with respect to other homomorphisms.) By Theorem 4.2, $\gamma = \alpha \otimes \beta$, where $\alpha : A^\Delta \to (H_1, V_1)$ and $\beta : (A \times H_1)^\Delta \to (H_2, V_2)$ are homomorphisms. When we project onto the left co-ordinate, we obtain

$$p_\alpha[g_i^{(1)}] = g^{(1)}_{(i+1) \bmod n}.$$

Since (H_1, V_1) does not have vertical confusion, all the $g_i^{(1)}$ must be equal. We will denote their common value by $g^{(1)}$. We now form a new multicontext $p^{(\alpha, g^{(1)})}$ by first substituting any forest evaluating to $g^{(1)}$ for the holes in p, which gives a forest t, then forming the forest t^α, and finally restoring the original holes. The resulting multicontext has the same shape as p, but its nodes are now labeled by elements of $A \times H_1$. Because the value $g^{(1)}$ is

stable after each application of p_α, we find that $p_\beta^{(\alpha,g^{(1)})}[g_i^{(2)}]$ is identical to the right-hand coordinate of $p_\gamma(g_i)$, and thus we have

$$p_\beta^{(\alpha,g^{(1)})}[g_i^{(2)}] = g^{(2)}_{(i+1) \bmod n}$$

for all $0 \leq i < n$. Since (H_2, V_2) does not have vertical confusion, we find that all the $g_i^{(2)}$, and consequently all the g_i, are identical. So (H, V) does not have vertical confusion.

In the case of vertical confusion with respect to uniform multicontexts, the proof is the same; we simply note that the multicontext $p^{(\alpha,g^{(1)})}$ defined above is uniform whenever p is. In the case of horizontal confusion with respect to some $G \subseteq H_1 \times H_2$, we use essentially the same argument: absence of confusion in the left coordinate permits us to reduce G to a set of the form $\{g^{(1)}\} \times G_2$, and we find that H_2 has horizontal confusion with respect to $p^{(\alpha,g^{(1)})}$ and G_2, so that $|G_2| = 1$, and hence $|G| = 1$.

We now show that in each case the non-confusing property is preserved under division. For subalgebras, this is trivial, but for quotients, there is something to prove. Accordingly, suppose that $\psi : (H_1, V_1) \to (H_2, V_2)$ is a surjective homomorphism of forest algebras. Let $\phi : A^\Delta \to (H_2, V_2)$ be a homomorphism. We can lift this to a homomorphism $\pi : A^\Delta \to (H_1, V_1)$ such that $\psi\pi = \phi$. First suppose (H_2, V_2) has vertical confusion with respect to some multicontext p and ϕ. We will show (H_1, V_1) has vertical confusion with respect to p and π. Vertical confusion in (H_2, V_2) gives us a sequence $g_0, \ldots, g_{n-1}$ of elements of H_2 with $n > 1$ such that $p_\phi[g_i] = g_{(i+1) \bmod n}$ for all $0 \leq i < n$. Choose an element $h_0 \in H_1$ such that $\psi(h_0) = g_0$, and define $h_1, h_2, \ldots$ by $h_{i+1} = p_\pi[h_i]$. By finiteness, there exist $j < k$ such that $h_j = h_k$. Since $\psi(h_j) = g_{j \bmod n}$ and $\psi(h_k) = g_{k \bmod n}$, we have $k - j$ is a multiple of n, and in particular, $k - j > 1$. We thus have

$$p_\pi[h_{j+i}] = h_{j+(i+1) \bmod (k-j)},$$

which gives vertical confusion in (H_1, V_1).

Now suppose that we have horizontal confusion in (H_2, V_2) with respect to ϕ. We will show how to obtain horizontal confusion in (H_1, V_1). Let $m = |H_1|$. By Lemma 8.1, there is a multicontext p such that (H_2, V_2) has m-ary horizontal confusion with respect to ϕ and some set $G' \subseteq H_2$. Let $G = \psi^{-1}(G')$. For $k > 0$, set $G_1^k = p^k[G]$. Since $p[G'] = G'$, we have $\psi(G_1^k) = G'$ for all k. In particular, $G_1^1 \subseteq G$, and by repeatedly applying p to both sides of this inclusion we obtain $G_1^{k+1} \subseteq G_1^k$ for all k. Thus this sequence eventually stabilizes, so we have some n for which $p[G_1^n] = G_1^n$. Let us set $G_1 = G_1^n$. Now it may be that (H_1, V_1) has horizontal confusion with respect to p, π, and G_1. If not, there is some hole x of p and $g_1 \in G_1$ such that $p[g_1/x][G_1] \subsetneq G_1$. So we let p' be the multicontext that results from substituting a forest that evaluates under π to g_1 for x, and set $G_2^1 = p'[G_1]$. Note that (H_2, V_2) has $(m-1)$-ary horizontal confusion with respect to p' and ϕ, so we still have $\psi(G_2^1) = G'$, as well as $G_2^1 = p'[G_1] =\subsetneq G_1$, so that $|G_2^1| < |G|$. We now repeat the procedure above, applying p' to G_2^1 until the sequence stabilizes at a set G_2, then checking if the result is a horizontal confusion for (H_1, V_1), and filling a hole of p' if it is not. We have

$$|G'| \leq \cdots |G_k| < |G_{k-1} < \cdots |G_1| \leq |G|,$$

so the process will terminate after no more than $|G| - |G'|$ generations, giving a horizontal confusion in (H_1, V_1). □

Theorem 8.3. *It is decidable if a given forest algebra has horizontal confusion, vertical confusion, or vertical confusion with respect to a uniform context.*

Proof. Confusion in a forest algebra (H, V) appears to depend on the choice of alphabet A, a multicontext p over A, and a morphism from A^Δ into (H, V). Observe, however, that we can restrict attention to a single alphabet and morphism: Consider V as a finite alphabet, and the morphism $\beta : V^\Delta \to (H, V)$ induced by the identity map on V. If (H, V) has a confusion with respect to a multicontext p over A and morphism $\alpha : A^\Delta \to (H, V)$, then we can transform it into a confusion of the same type with respect to V and β in the obvious fashion, replacing each node label $a \in A$ of p labeled by $\alpha(a) \in V$. Thus in the argument below, we suppress explicit mention of an alphabet and morphism and work simply with the elements of V.

Vertical confusion. Testing whether (H, V) has vertical confusion with respect to some multicontext reduces to verifying whether a certain monoid containing V is aperiodic. If $v, w \in V$, we define $v + w$ to be the transformation on H given by

$$(v + w)h = vh + wh$$

for all $h \in H$. Let $\hat{V}$ be the collection of all maps on H containing V and closed under composition and addition. $\hat{V}$ then consists of all multicontexts over (H, V). Furthermore, $\hat{V}$ is effectively computable from V, since whenever we have a set U of transformations on H, we can check for each $v, w \in U$ whether $v + w$ and vw belong to U, and if not, adjoin them to U. Since there are only finitely many transformations on H, we eventually reach a stage at which we can add no new elements to U, at which point the algorithm terminates.

$\hat{V}$ is a monoid under composition, and (H, V) is free of vertical confusion if and only if this monoid is aperiodic; *i.e.,* if and only if $p^k = p^{k+1}$ for all $p \in \hat{V}$ and sufficiently large k, which we can determine effectively.

Vertical confusion with respect to a uniform multicontext. The argument is the same as above, however now we must build a monoid containing V that consists of exactly all the uniform multicontexts. We accordingly close V under composition and the operations

$$v \mapsto v + v + \cdots + v.$$

Observe that the number of summands in this expression can be bounded above by the size of H, so we can compute this closure effectively as well. Let us denote the resulting monoid $\tilde{V}$. (H, V) does not have vertical confusion with respect to any uniform multicontext if and only if $\tilde{V}$ is aperiodic.

Horizontal confusion. We now test if (H, V) has horizontal confusion. The algorithm first guesses the set G. For a multicontext p, we define its profile to be the set

$$\pi(p) = \{p[g/x][G] : g \in G, x \in \text{holes}(p)\} \times p[G] \qquad \in P(P(H)) \times P(H).$$

The forest algebra has horizontal confusion with respect to a multicontext p and G if and only if the profile $\pi(p)$ only has supersets of G on the first coordinate. Therefore, to determine if the forest algebra has horizontal confusion, it suffices to compute the set

$$Y = \{\pi(p) : p \text{ is a multicontext}\}.$$

This set is computed using a fix-point algorithm, since it is the least set that satisfies the properties listed below. (In the implications, we lift the forest algebra operations to sets $F \subseteq H$ and families of sets $\mathcal{F} \subseteq P(H)$ in the natural way.)

$$(\{\{g\} : g \in G\}, G) \in Y$$
$$(\mathcal{F}, F) \in Y \Rightarrow (v\mathcal{F}, vF) \in Y \qquad \text{for every } v \in V$$
$$(\mathcal{F}_1, F_1), (\mathcal{F}_2, F_2) \in Y \Rightarrow (\mathcal{F}_1 + F_2 \cup F_1 + \mathcal{F}_2, F_1 + F_2)$$

□

9. Applications

Here we apply the results of the preceding section to exhibit a forest language in CTL* that is not in CTL, a language in PDL that is not in $FO[\prec]$, and a language that is not in graded PDL. All of our examples have syntactic forest algebras with aperiodic vertical monoids, and all the classes in question contain languages with arbitrarily complicated aperiodic vertical monoids, so we really do need machinery of forest algebras to give algebraic proofs of these separations.

9.1. **Forests with a maximal path in $(ab)^*$.** Consider the set L_1 of forests over $A = \{a, b\}$ in which there is a maximal path—that is, a path from a root to a leaf— in $(ab)^*$. This language is in CTL*. To see this, note that $\phi = \mathsf{E}(A^+)$ is a forest formula in CTL* defining the set of nonempty forests. Consider the formally disjoint formulas

$$\phi_1 = b \wedge \phi \qquad \phi_2 = b \wedge \neg\phi_1 \qquad \phi_3 = \neg\phi_1 \wedge \neg\phi_2.$$

The formula ϕ_1 holds in non-leaf nodes with label b, the formula ϕ_2 holds in leaves with label b, and the formula ϕ_3 holds in nodes with label a. Then L_1 is defined by the CTL* forest formula $\mathsf{E}((\phi_3\phi_1)^*(\phi_3\phi_2))$. We claim that L_1 is not in CTL. To do this, by Theorem 8.2, we need only exhibit a multicontext p with respect to which the syntactic forest algebra of L_1 has vertical confusion. Let $p = a\Box + b\Box$. Let h_0 be the class of the tree b in the syntactic congruence of L_1, and let h_1 be the class of the tree ab. Observe that h_0 and h_1 are distinct horizontal elements of the syntactic algebra, since h_1 contains elements of L_1 and h_0 does not. We have vertical confusion, because $p[h_0]$ is then the class of $ab + bb$, which is h_1, and $p[h_1]$ is the class of $aab + bab$, which is h_0.

9.2. **Binary trees with even path length.** This example uses *unlabeled binary trees*, which are trees over a one-letter alphabet $\{a\}$ where every node has zero or two children. Let L_2 be the set of unlabeled binary trees where every path from the root to a leaf has even length. Let p be the uniform multicontext $a(\Box + \Box)$. Let h_0 denote the set of binary trees in which every maximal path has even length, and h_1 the set of binary trees in which every maximal path has odd length. These are distinct classes in the syntactic congruence of L_2. Obviously $p[h_0] = h_1$ and $p[h_1] = h_0$, so we have vertical confusion with respect to a uniform multicontext, and thus by Theorem 8.2, L_2 is not in $FO[\prec]$.

An argument due to Potthoff [19] can be used to show that L_2 is definable in first-order logic in which there is both the ancestor and the next-sibling relations. Languages definable in $FO[\prec]$ are obviously in the intersection of the class of languages definable in FO with $\prec$ and the next-sibling relationship, and the class of languages L with commutative H_L.

This example shows that the containment is strict. Note that L_2 is expressible in graded PDL so we have also established that the languages in graded PDL with aperiodic forest algebras need not be definable in $FO[\prec]$ (there is even an example, also due to Potthoff, which shows that languages definable in graded PDL with aperiodic forest algebras need not be definable in FO with $\prec$ and the next-sibling relationship).

9.3. *(Boolean expressions).* Consider the set L_3 of trees over the alphabet $\{0, 1, \vee, \wedge\}$ that are well-formed boolean expressions (*i.e.*, all the leaf nodes are labeled 0 or 1, and all the interior nodes are labeled $\vee$ or $\wedge$) that evaluate to 1. L_3 is contained in a single equivalence class of the syntactic congruence, as is the set of well-formed trees that evaluate to 0. We denote the corresponding elements of H_{L_3} by h_1 and h_0.

Now consider the multicontext $p = \vee(\wedge(\Box + \Box) + \wedge(\Box + \Box))$. We can fix a value 1 or 0 in any single hole, and then set the remaining holes to obtain either a tree evaluating to 1 or a tree evaluating to 0. Thus the syntactic algebra of L_3 has horizontal confusion with respect to the multicontext p and the set $\{h_0, h_1\}$, and so is not in graded PDL. Observe that the vertical component of the syntactic algebra of L_3 is aperiodic: In contrast to the word case, languages recognized by aperiodic algebras are not necessarily expressible in first-order logic, or even in graded PDL.

9.4. **Horizontally idempotent and commutative algebras.** Obviously, we can separate CTL* and PDL from $FO[\prec]$ and graded PDL, respectively, because the syntactic algebras for the former classes have idempotent and commutative horizontal parts, while for the latter the horizontal components need only be aperiodic and commutative. Thus, for example, any language in $FO[\prec]$ that fails to satisfy the idempotency condition is not in CTL*. We can use our algebraic methods to show that this is in fact the *only* distinction:

Theorem 9.1. *Let (H, V), (H_j, V_j), $j = 1, \ldots, k$ be forest algebras such that H is idempotent and commutative, each (H_i, V_i) is a path algebra, and such that (H, V) divides $(H_1, V_1) \circ \cdots \circ (H_k, V_k)$. Then each (H_i, V_i) has a distributive homomorphic image (H'_i, V'_i) such that (H, V) divides $(H'_1, V'_1) \circ \cdots \circ (H'_k, V'_k)$.*

Proof. Let (H, V) be a path algebra. We define $e(H)$ to be the set of idempotents of H. By the commutativity of H, the sum of two idempotents is idempotent. Thus $e(H)$ is an idempotent and commutative submonoid of H. If $h \in H$ and k is a nonnegative integer, we denote by $k \cdot h$ the sum of k copies of h. We also denote by ωh the unique idempotent in $\{k \cdot h : k \in \mathbb{N}\}$. Since H is aperiodic and commutative, there exists k such that $\omega \cdot h = k \cdot h = (k + 1) \cdot h$ for all $h \in H$.

For every $v \in V$, we define a function $\overline{v} : e(H) \to e(H)$ by

$$\overline{v} \cdot e = \omega(ve).$$

We define a forest algebra $(e(H), \overline{V})$ as follows. The horizontal monoid is $e(H)$. The vertical monoid is $\overline{V} = \{\overline{v} : v \in V\}$, with function composition. The action is by applying the function $\overline{v}$ to an argument $e \in e(H)$. To prove that this is a forest algebra, we need to show that for any element $e \in e(H)$, there is an element $\overline{v} \in \overline{V}$ such that $\overline{v}f = e + f$ holds for any $f \in e(H)$. This element is simply $\overline{e + \Box}$. Indeed,

$$(\overline{e + \Box}) \cdot f = \omega((e + \Box)f) = \omega(e + f) = e + f.$$

This concludes the proof that $(e(H), \overline{V})$ is a forest algebra.

We now show that $(e(H), \overline{V})$ is distributive. In other words, the following identity holds for any $v \in V$ and $e_1, e_2 \in e(H)$.

$$\overline{v}(e_1 + e_2) = \overline{v}e_1 + \overline{v}e_2 \tag{9.1}$$

Using the first identity in the definition of path algebras (5.1), we obtain

$$\begin{aligned}
\overline{v}(e_1 + e_2) &= \omega v(e_1 + e_2) \\
&= \omega(v(e_1 + e_2) + v(e_1 + e_2)) \\
&= \omega(v(e_1 + e_2 + e_1 + e_2) + v \cdot 0) \\
&= \omega(v(e_1 + e_2) + v \cdot 0) \\
&= \omega(ve_1 + ve_2) \\
&= \omega ve_1 + \omega \cdot ve_2 \\
&= \overline{v}e_1 + \overline{v}e_2.
\end{aligned}$$

Finally, we prove that the function

$$\alpha(h) = \omega \cdot h \qquad \alpha(v) = \overline{v}$$

is a forest algebra homomorphism

$$\alpha : (H, V) \to (e(H), \overline{V}).$$

Clearly α preserves $+$. It remains to show that it preserves the remaining two operations of forest algebra, namely inserting a forest into a context and composition of two contexts. For inserting a forest into a context, we have

$$\alpha(vh) = \omega \cdot vh = \overline{v}h = \alpha(v)\alpha(h).$$

For composition of two contexts we need to show $\alpha(v_1)\alpha(v_2) = \alpha(v_1 v_2)$. Since $\overline{V}$ is defined as a set of functions on $e(H)$, we need to show that both sides of the equality describe the same function on $e(H)$. In other words, we have to prove that for every $e \in e(H)$,

$$\overline{v_1}(\overline{v_2}e) = \overline{v_1 v_2}e \tag{9.2}$$

First note that the path algebras property (5.1) implies that for all $h \in H$, $v \in V$, $m \in \{1, 2, \ldots\}$, we have

$$m \cdot (vh) = v(m \cdot h) + (m - 1) \cdot (v0).$$

Thus by aperiodicity of H,

$$\omega(vh) = v(\omega h) + \omega(v0).$$

If $e \in H$ is idempotent, this becomes

$$\omega(ve) = ve + \omega(v0).$$

Consequently we have

$$\begin{aligned}
\overline{v_1 v_2} e &= \omega(v_1 v_2 e) \\
&= v_1 \omega(v_2 e) + \omega(v_1 \cdot 0) \\
&= v_1(v_2 e + \omega(v_2 \cdot 0)) + \omega(v_1 \cdot 0) \\
&= v_1(v_2 e + \omega(v_2 \cdot 0) + \omega(v_2 \cdot 0)) + \omega(v_1 \cdot 0) \\
&= v_1(\omega(v_2 e) + \omega(v_2 \cdot 0)) + \omega(v_1 \cdot 0) \\
&= v_1 \omega(v_2 e + \omega(v_2 \cdot 0)) + \omega(v_1 \cdot 0) \\
&= \omega v_1(v_2 e + \omega(v_2 \cdot 0)) \\
&= \omega v_1(\omega v_2 e) \\
&= \overline{v}_1(\overline{v}_2 e).
\end{aligned}$$

Summing up: We have defined a forest algebra homomorphism

$$\alpha : (H, V) \to (e(H), \overline{V})$$

where the target forest algebra is distributive and horizontally commutative and idempotent.

Suppose now that (H, V) is a forest algebra with idempotent and commutative H that divides a wreath product

$$(H_1, V_1) \circ \cdots \circ (H_k, V_k),$$

where each (H_i, V_i) is a path algebra. To complete the proof of the theorem, we will show that (H, V) divides

$$(e(H_1), \overline{V_1}) \circ \cdots \circ (e(H_k), \overline{V_k}).$$

We now apply Lemma 4.1 on the equivalence of the two definitions of division. The hypothesis is then that there is a submonoid H' of $H_1 \times \cdots \times H_k$, and a homomorphism f from H' onto H with the following property: For each $v \in V$ there is $\hat{v}$ in the vertical monoid of $(H_1, V_1) \circ \cdots \circ (H_k, V_k)$ such that for all $h \in H'$,

$$f(\hat{v}h) = v f(h).$$

Note that h has the form $(h_1, \ldots, h_k)$, where $h_i \in H_i$ for $i = 1, \ldots, k$, and

$$\hat{v} = (u, g_2, \ldots, g_k),$$

where $u \in V_1$ and each

$$g_j : H_1 \times \cdots \times H_{j-1} \to V_j$$

is a map. Since H is idempotent, $f(\omega h) = f(h)$ for all $h \in H'$. We consider the restriction of f to $e(H')$, which is a subset of $e(H_1) \times \cdots \times e(H_k)$. We will show that for each $v \in V$ there is an element $\tilde{v}$ of the vertical monoid of $(e(H_1), \overline{V_1}) \circ \cdots \circ (e(H_k), \overline{V_k})$ such that for all $e \in e(H)$,

$$f(\tilde{v}e) = v f(e).$$

To do this, we simply alter $\hat{v} = (u, g_2, \ldots, g_k)$ in the obvious fashion:

$$\tilde{v} = (\overline{u}, \overline{g_2}, \ldots, \overline{g_k}),$$

where by definition

$$\overline{g}_j(x) = \overline{g_j(x)}.$$

Set $e = (e_1, \ldots, e_k) \in e(H')$. We have

$$\begin{aligned} f(\tilde{v}e) &= f(\overline{u}e_1, \overline{g_2(e_1)}e_2, \ldots, \overline{g_k(e_1, \ldots, e_{k-1})}e_k) \\ &= f(\overline{u}e_1, \overline{g_2(e_1)}e_2, \ldots, \overline{g_k(e_1, \ldots, e_{k-1})}e_k) \\ &= f(\omega(ue_1), \omega g_2(e_1)e_2, \ldots, \omega g_k(e_1, \ldots, e_{k-1})e_k) \\ &= f(ue_1, g_2(e_1)e_2, \ldots, g_k(e_1, \ldots, e_{k-1})e_k) \\ &= f(\hat{v}e) \\ &= vf(e), \end{aligned}$$

which completes the proof. □

Theorem 5.2 immediately yields the following corollary:

Theorem 9.2. *A forest language is definable in CTL* (respectively PDL) if and only if it is definable in $FO[\prec]$ (respectively graded PDL) and its syntactic algebra is horizontally idempotent.*

The first of these facts follows from a result of Moller and Rabinovich [16] who show that over infinite trees properties expressible in CTL* are exactly the bisimulation-invariant properties expressible in monadic path logic.

10. Conclusion and further research

Results like those in Section 9 are typically proved by model-theoretic methods. Here we have demonstrated a fruitful and fundamentally new way, based on algebra, to study the expressive power of these logics.

Of course, the big question left unanswered is whether we can establish effective necessary and sufficient conditions for membership in any of these classes. We do not expect that the conditions established in Theorem 8.2 are sufficient. The approach outlined in Section 6 may constitute a model for how to proceed: a deeper understanding of the ideal structure of forest algebras can lead to new wreath product decomposition theorems.

In a sense, we are searching for the right generalization of aperiodicity. For regular languages of words, aperiodicity of the syntactic monoid, expressibility in first-order logic with linear ordering, expressibility in linear temporal logic, and recognizability by an iterated wreath product of copies of the aperiodic unit U_2 are all equivalent. For forest algebras, the obvious analogues are, respectively, aperiodicity of the vertical component of the syntactic algebra, expressibility in $FO[\prec]$, expressibility in CTL, and recognizability by an iterated wreath product of copies of $\mathcal{U}_2$. As we have seen, only the last two coincide. Understanding the precise relationship among these different formulations of aperiodicity for forest algebras is an important goal of this research.

Another way of looking at this research is that it sets the scene for a Krohn-Rhodes theorem for trees. The Krohn-Rhodes theorem states that every transition monoid divides an iterated wreath product of transition monoids which are either U_2 or groups that divide the original monoid. The ingredients of the theorem are therefore: a notion of wreath product, a notion of an easy transition monoid U_2, and a notion of a difficult transition monoid (a group). For our purposes here, we are particularly interested in the (already quite difficult) version of the theorem which states that every aperiodic transition monoid divides a wreath product of copies of U_2. In this paper, we have provided some of the

ingredients: the wreath product and the easy objects. (There are several candidates for the easy objects, e.g. simply $\mathcal{U}_2$ or maybe path algebras. There are probably several Krohn-Rhodes theorems). We have provided examples of properties one expects from the difficult objects (the various types of confusion), but we still have no clear idea what they are (in other words, what is a tree group?). We have also shown that the wreath product is strongly related to logics and composition. Finding (at least one) Krohn-Rhodes theorem for trees is probably the most ambitious goal of this research.

References

[1] Pablo Barceló and Leonid Libkin. Temporal logics over unranked trees. In *LICS*, pages 31–40. IEEE Computer Society, 2005.

[2] M. Benedikt and L. Segoufin. Regular tree languages definable in FO. In Volker Diekert and Bruno Durand, editors, *STACS*, volume 3404 of *Lecture Notes in Computer Science*, pages 327–339. Springer, 2005.

[3] M. Bojanczyk and I. Walukiewicz. Forest algebras. In Erich Graedel Joerg Flum and Thomas Wilke, editors, *Logic and Automata: History and Perspectives*. Amsterdam University Press, 2008.

[4] Mikolaj Bojanczyk. Two-way unary temporal logic over trees. In *LICS*, pages 121–130, 2007.

[5] Mikolaj Bojanczyk and Luc Segoufin. Tree languages defined in first-order logic with one quantifier alternation. In *ICALP*, pages 233–245, 2008.

[6] Mikolaj Bojanczyk, Luc Segoufin, and Howard Straubing. Piecewise testable tree languages. In *LICS*, pages 442–451. IEEE Computer Society, 2008.

[7] Janusz A. Brzozowski and Robert Knast. The dot-depth hierarchy of star-free languages is infinite. *J. Comput. Syst. Sci.*, 16(1):37–55, 1978.

[8] Joëlle Cohen, Dominique Perrin, and Jean-Eric Pin. On the expressive power of temporal logic. *J. Comput. Syst. Sci.*, 46(3):271–294, 1993.

[9] Z. Ésik and P. Weil. On logically defined recognizable tree languages. In Paritosh K. Pandya and Jaikumar Radhakrishnan, editors, *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings*, volume 2914 of *Lecture Notes in Computer Science*, pages 195–207. Springer, 2003.

[10] Z. Ésik and P. Weil. Algebraic recognizability of regular tree languages. *Theor. Comput. Sci*, 340(1):291–321, 2005.

[11] Zoltán Ésik. Characterizing CTL-like logics on finite trees. *Theor. Comput. Sci.*, 356(1-2):136–152, 2006.

[12] Zoltán Ésik and Szabolcs Iván. Aperiodicity in tree automata. In Symeon Bozapalidis and George Rahonis, editors, *CAI*, volume 4728 of *Lecture Notes in Computer Science*, pages 189–207. Springer Springer, 2007.

[13] Zoltan Esik and Ivan Szabolcs. Some varieties of finite tree automata related to restricted temporal logics. *Fundamenta Informaticae*, 82:79–103, 2008.

[14] T. Hafer and W. Thomas. Computation tree logic CTL and path quantifiers in the monadic theory of the binary tree. In *International Colloquium on Automata, Languages and Programming*, volume 267 of *Lecture Notes in Computer Science*, pages 260–279, 1987.

[15] L. Libkin. Logics for unranked trees: an overview. In *Automata, languages and programming*, volume 3580 of *Lecture Notes in Comput. Sci.*, pages 35–50. Springer, Berlin, 2005.

[16] Faron Moller and Alexander Moshe Rabinovich. On the expressive power of CTL. In *LICS*, pages 360–369, 1999.

[17] Faron Moller and Alexander Moshe Rabinovich. Counting on CTL^*: on the expressive power of monadic path logic. *Inf. Comput.*, 184(1):147–159, 2003.

[18] T. Place and L. Segoufin. A decidable characterization of locally testable tree languages. In *International Colloquium on Automata, Languages and Programming*, pages 285–296, 2009.

[19] A. Potthoff. First-order logic on finite trees. *Lecture Notes in Computer Science*, 915:125–139, 1995.

[20] R. McNaughton and S. Papert. *Counter-free Automata*. MIT Press, Cambridge, USA, 1971.

[21] P. Stiffler. Extensions of the fundamental theory of finite semigroups. *Advances in Mathematics*, 11:159–209, 1973.

[22] H. Straubing. A generalization of the Schützenberger product of finite monoids. *Theor. Comput. Sci.*, 13:137–150, 1981.
[23] H. Straubing. *Finite Automata, Formal Logic, and Circuit Complexity*. Progress in Theoretical Computer Science. Birkhäuser Boston Inc., Boston, MA, 1994.
[24] Thomas Wilke. Classifying discrete temporal properties. In Christoph Meinel and Sophie Tison, editors, *STACS*, volume 1563 of *Lecture Notes in Computer Science*, pages 32–46. Springer, 1999.
[25] Zhilin Wu. A note on the characterization of TL[EF]. *Information Processing Letters*, 102((2-3)):28–54, 2007.