Wreath Products of Forest Algebras, with Applications to Tree Logics

Mikolaj Bojanczyk, Howard Straubing, Igor Walukiewicz

Wreath Products of Forest Algebras, with Applications to Tree Logics

Mikolaj Bojańczyk
University of Warsaw

Howard Straubing
Boston College

Igor Walukiewicz
CNRS, LaBRI, Bordeaux

***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.**

I. Introduction

Logics for specifying properties of labeled trees play an important role in several areas of Computer Science. Recently, attention has turned to logics for *unranked* trees, in which there is no *a priori* bound on the number of children a node may have. Barceló and Libkin [1] and Libkin [13] catalogue a number of such logics and contrast their expressive power. Many fundamental problems in this domain remain unsolved. For example, we do not as yet possess an effective criterion for determining whether a given property of trees is expressible in the temporal logics CTL, or CTL*, or in first-order logic with the ancestor relation.

For properties of *words,* analogous questions have been fruitfully studied by algebraic means: The logics in question are typically no more expressive than monadic second-order logic, and thus the property of words expressed by a formula defines a regular language L. Expressibility 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 [16] 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 [18]; temporal logics are studied by Cohen, Perrin and Pin [7] and Wilke [19], among others.

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 [8], [9]. Recently, Bojańczyk and Walukiewicz [3] introduced *forest algebras*, a generalization of 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.

In the present paper we continue this line of research by studying the *wreath product* of forest algebras. Wreath products of transformation monoids play an important role in the theory for words, and we find that the definition and important properties of this product, especially its connection to a composition operation on languages and to generalized temporal operators, extend to forest algebras with no real change. We consider some standard temporal logics, CTL, CTL* first-order logic with ancestor, PDL, as well as two less known variants: EF and graded PDL. The first, is the fragment of CTL with EF as the only operator. The second, is an extension of PDL allowing us to say, for example, that there are at least three paths satisfying some property.

We show the following: *(a)* To each of the logics mentioned above, we associate a class of 'basic' forest algebras with the property that a language of forests is definable in the logic if and only if it is recognized by an iterated wreath product of the associated basic algebras. *(b)* For EF and CTL, the base reduces to a single algebra, but for the others 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. By using our algebraic framework, we are able to give a simple and general proof of this fact. *(c)* For the logics that do not have a finite base, we are nonetheless able to establish effective necessary and sufficient conditions for a forest algebra to belong to the associated base. *(d)* We give a (new) proof of an effective necessary and sufficient condition for a forest language to be definable in EF. Our argument shows that an algebraic decomposition theory for forest algebras is both feasible and useful. *(e)* We do not, on the other hand, find analogous necessary and sufficient conditions for the other logics mentioned above, and this remains an outstanding open problem. However, we are able to use our framework to establish algebraic necessary conditions for definability in these logics, and consequently to prove that a number of specific languages are not definable in them. *(f)* We also find necessary and sufficient conditions for a given first-order definable language to be definable in CTL*, and for a given language definable in graded PDL to be definable in PDL.

We note that Ésik and Ivan [10], [11] have done work of a

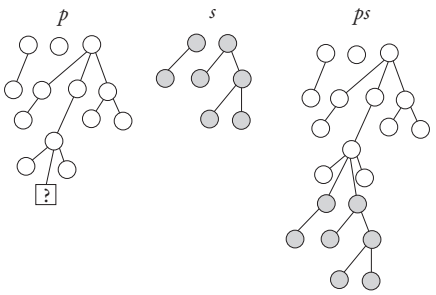
similar flavor for CTL (for trees of bounded rank). Our work here is of considerably larger scope, both in the number of different logics considered, and the concrete consequences our algebraic theory permits us to deduce. For reasons of space, many of the proofs have been reduced to a sketch, or omitted. These will appear in the full paper.

II. 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 0. This gets the recursion started. So, for example, $s = a(b0 + c(a0 + ab0)) + a(a0 + b0)$ is a forest. Normally, when we write such expressions, we delete the zeros. We depict s in the obvious fashion as an ordered forest of two ordered trees whose nodes are labeled by the letters a, b, c. In this example, the two root nodes are both labeled a, and there are five leaves altogether. The set of forests over A is a monoid with respect to forest concatenation $s + t$, with the empty forest 0 being the identity. We denote this set by H_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 deleting a leaf from a nonempty forest and replacing it 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 a similar manner, we can substitute another context q for the hole, and obtain a new context pq. The set of contexts over A forms a monoid, with respect to context composition pq, with the empty context $\Box$ being the identity. We denote this set by V_A.

Note that for all forests $s, t \in H_A$, 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 $u \in H_A$.

Strictly speaking, 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. But all the applications in the present article effectively concern unordered trees, so there is no harm in thinking of $+$ as a commutative operation on forests.

III. Logics for Forest Languages

For a general treatment of predicate and temporal logics for unranked trees, see Libkin [13]. The logics we describe below are all fragments of monadic second-order logic, and thus the languages they define are all regular forest languages. These languages can be also represented by automata that are a minor modification of the standard bottom-up tree automata. The transition function is modified to cope with unbounded branching, and the definition of acceptance needs to consider states in the roots of all the trees in the forest. See [3] for the definition.

A. 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. $Q_a x$ 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 $\prec$-incomparable occurrences of a. We denote this logic by $FO[\prec]$. 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 component ($\exists x \forall y (x \prec y)$). Thus the property that a set of trees is first-order definable does not depend on whether we choose to interpret sentences in trees or in forests.

B. 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 sometimes called *graded propositional dynamic logic* (graded PDL).

Syntax of temporal formulas. Temporal formulas are built starting with atomic *label formulas* a for $a \in A$. We combine temporal formulas with the usual boolean operations. We also define a temporal operator: if $k > 0$, Φ is a finite set of formulas, and $L \subseteq \Phi^+$ is a regular language of *words* over the alphabet Φ, then $\mathsf{E}^k L$ is a formula. The idea is

that $\mathsf{E}^k L$ says there are at least k paths that satisfy L; the precise semantics are defined below. We place an additional restriction, called *unambiguity*, on the use of this operator: We require that the formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ are formally disjoint: that is, for all $i > 1$, ϕ_i has the form $\psi \wedge \neg \bigvee_{j<i} \phi_j$ for some formula ψ. We write $\mathsf{E}L$ for $\mathsf{E}^1 L$.

Semantics of temporal formulas. Usually, satisfaction for formulas is defined with respect to trees. For forests, the conventions are less well-established. Nevertheless, semantics for forests will be important for us, especially in the context of the wreath products. We will therefore use two notions of satisfaction: a *tree-satisfaction relation* $t \models \varphi$, which coincides with the usual notion of satisfaction, and a *forest-satisfaction relation* $t \models_f \varphi$, which is slightly unusual. The definition of the two will be mutually recursive. When t is a forest, then only the forest-satisfaction $t \models_f \varphi$ is defined, but when t is a tree, then both $t \models_f \varphi$ and $t \models \varphi$ are defined, with different meanings.

A label formula a is tree-satisfied by the trees whose root label is a, but is not forest-satisfied by any forest (even if the forest contains only a single tree). Whether a formula $\mathsf{E}^k L$ is tree-satisfied by a tree as depends only on the subforest of the root and not on the root node: that is, $as \models \psi$ if and only if $s \models_f \psi$. Finally, if $s \in H_A$, then $s \models_f \mathsf{E}^k L$ if and only if there are at least k distinct paths in s that satisfy L in the following sense: A path $x_1 \cdots x_n$ is a sequence of nodes connected by the child relation, beginning in one of the roots, and ending in some node, not necessarily a leaf. The path satisfies L if there is a sequence $\phi_1, \ldots, \phi_n \in \Phi$ such that the word $\phi_1 \cdots \phi_n$ belongs to L and for each $i = 1, \ldots, n$, the subtree of x_i tree-satisfies ϕ_i. Note that tree, and not forest, satisfaction is required in the subtree of x_j. Note also that the paths need not end in leaves, and that the sequence $\phi_1, \ldots, \phi_n$ is uniquely determined by the path, due to the unambiguity condition on Φ. As the paths need not end in leaves, some paths may be prefixes of others, for instance, the tree aaa forest-satisfies $\mathsf{E}^3 a^+$. Boolean operations have their usual interpretation.

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

EF : When ψ describes a property of words, then $\mathsf{F}\psi$ describes the words where ψ holds at some position, possibly the first. We obtain an analogous temporal operator for trees and forests by defining $\mathsf{EF}\psi$ to be $\mathsf{E}L$, where $L = (\neg\psi)^*\psi$. Thus, when s is a forest, $s \models_f \mathsf{EF}\psi$ if and only if some subtree of s, possibly rooted at a root of s, tree-satisfies ψ. When t is a tree, $t \models \mathsf{EF}\psi$ if some proper subtree of t tree-satisfies ψ. Note how the tree semantics of this temporal operator resembles the "strict semantics" of EF in which one ignores the current node, while the forest semantics resemble the non-strict semantics. We denote by EF the forest languages definable by a formula built from the label formulas $a \in A$ using boolean operations and the temporal operator EF.

CTL : When ψ, ϕ describe properties of words, then $\psi \mathsf{U} \phi$ describes the set of words $a_1 \cdots a_n$ where for some $i = 1, \ldots, n$, the word beginning in position i satisfies ϕ, and all the words beginning in positions $1, \ldots, i-1$ satisfy ψ. The analogous operator for trees and forests is $\mathsf{E}(\psi \wedge \neg\phi)^*\phi$, which we denote $\mathsf{E}\psi\mathsf{U}\phi$. The forest semantics is that the subtree of some node x tree-satisfies the formula ϕ, and the subtree at every strict ancestor of x tree-satisfies ψ. For the tree semantics, the root of the tree is ignored. By nesting this operator we get the logic CTL. (The dual operator $\mathsf{E}\neg(\psi\mathsf{U}\phi)$ is redundant in finite trees.)

First-order logic : We can use the same formalism to characterize the languages definable in $FO[\prec]$ in terms of a temporal logic.

Theorem 1 *A forest language is definable in $FO[\prec]$ if and only if it is definable by a formula in the fragment of the language of temporal formulas using the operator $\mathsf{E}^k L$ only for word languages L that are first-order definable.*

This is a slight adaptation of Hafer and Thomas [12], and Moller and Rabinovich [14]. We will give the proof in the full paper. Note that the theorem 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. 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.)

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 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 restriction on L being first-order definable. If we place no restriction on either the multiplicity k or the regular language L, we obtain *graded PDL.* (Actually, one can show, using methods similar to Theorem 1, 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.)

C. 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 [10]. 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 cannot be finite.

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 set of nodes in $t[L_1, \ldots, L_k]$ is the same as in t, except that each node x gets a label (a, b_i), where the first coordinate a is the old label of x in t, while the second coordinate 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 $s[L_1, \ldots, L_k] \in L$.

The operation of language composition is similar to formula composition. The definition below uses this intuition, in order to define a "temporal logic" based on operators given as forest languages. 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 over 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 (although such classes will not appear in this particular 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 $TL[\mathscr{L}]$ of forest languages that contains $\mathcal{L}$, is closed under boolean operations and under language composition, i.e.

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

By restating the definitions of the temporal logics in terms of language composition, we get the following theorem.

Theorem 2 *The logics EF, CTL, $FO[\prec]$, CTL* , PDL and graded PDL have language bases as depicted in Figure 1.*

Note that the part about $FO[\prec]$ depends on Theorem 1.

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" : $k \in \mathbb{N}, L$ regular}

Figure 1. Language bases for temporal logics

IV. Forest Algebras

A. Definition of forest algebras

Forest algebras, introduced by Bojańczyk and Walukiewicz in [3], 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. Forest algebras satisfy an 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 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 $a \in A$.

B. 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. In particular, (H_L, V_L) *divides*—that is, it is a quotient of a subalgebra of—every forest algebra 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].)

C. Wreath product

We are going to give a definition of the wreath product of two forest algebras. In the case of words, the wreath product models cascade product of automata, composition of formulas, or sequential composition of morphisms. This will be also the case for forests.

It is easier to begin by describing the sequential composition of two morphisms. Assume that

$$\alpha : A^\Delta \to (H, V)$$

is a forest algebra morphism. For a forest t over A, let t^α be the forest over $A \times H$ obtained from t by changing the label of each node x from its original label a to the pair (a, h), where h is the value of α on the subforest of x. In the sequential composition, we will use a second morphism that reads the relabeling t^α, namely

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

The sequential composition of α and β is the function $\alpha \rhd \beta$

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

The wreath product $(H, V) \circ (G, W)$ of forest algebras is designed to describe sequential composition. We stated the correspondence in the following theorem, which, we hope, will aid the reader in understanding the motivation behind the wreath product definition.

Theorem 3 *For every two morphisms*

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

there is a morphism from A^Δ into the wreath product $(H, V) \circ (G, W)$ that, when restricted to forests, is equal to the sequential composition $\alpha \rhd \beta$. Conversely, for every morphism form A^Δ into the wreath product there is an equivalent sequential composition.

It is not surprising that there is an algebraic construction that models sequential composition. Such constructions have been long known for monoids, and for word and tree automata. The noteworthy fact is that the algebraic construction for forest algebras happens to be the exact same wreath product as used for transformation monoids. Actually, 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 so-called transformation monoids.

We now proceed to give the definition of wreath product. A forest algebra is, in particular, a transformation monoid– so we can form the wreath product of two of these to obtain another transformation monoid: Given (H_1, V_1) and (H_2, V_2), we define $(H_1, V_1) \circ (H_2, V_2)$ to be the pair $(H_1 \times H_2, V_1 \times V_2^{H_1})$, with the action defined by

$$(v_1, f)(h_1, h_2) = (v_1 h_1, f(h_1) v_2).$$

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.) Of course, since H_1, and H_2 are themselves monoids, we can give $H_1 \times H_2$ a monoid structure through the usual direct product. Finally, let $h_1 \in H_1$, $h_2 \in H_2$ and consider the map $f : H_1 \to V_2$ that sends every element to $(1 + h_2)$. Then for any $g_1 \in H_1, g_2 \in H_2$, we have

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

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 $(H_1, V_1) \circ (H_2, V_2)$.

V. Wreath Product Characterizations

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 3 justifies this notation.

Corollary 4 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 provide algebraic bases for the logics discussed in Section III. By the above corollary, all we need to do is to provide, for each logic, a class of forest algebras that

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

captures the language base. This is stated in the following theorem; the algebras used in the statement are described immediately afterwards.

Theorem 5 *The logics EF, CTL, $FO[\prec]$, CTL*, PDL and graded PDL have algebraic bases as depicted in Figure 2.*

We proceed to describe the algebras mentioned in Figure 2. As we will see later, there are no finite bases for the last four logics in the figure. For the infinite bases we propose, we will give effective characterizations in terms of identities in forest algebras. Therefore, to check if a given forest language belongs to the base it is enough to check if the identities hold in the finite, syntactic forest algebra of the language.

First, we recall that an *aperiodic* finite semigroup 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. 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 in the language base of CTL can be recognized by U_2, and conversely, every language recognized by $\mathcal{U}_2$ is a boolean combination of members of the language base of CTL. So $\mathcal{U}_2$ forms an algebraic base for CTL.

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 $v \in V$, $h_1, h_2 \in H$.

Theorem 6 *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* if it is a boolean combination of members of the language base of $FO[\prec]$.

Theorem 7 *A finite forest algebra (H, V) recognizes only path languages if and only if H is aperiodic and commutative and*

$$vg + vh = v(g + h) + v0 \tag{1}$$

$$u(g + h) = u(g + uh) \tag{2}$$

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 1 and 2 with H aperiodic and commutative.

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 had one algebra, while our other bases contained infinitely many algebras. This turns out to be optimal, as stated below.

Theorem 8 (Infinite base theorem) *Suppose a language class $\mathscr{L}$ contains all languages L_n = "exists a path in $(a^n b)^* c$". Then $\mathscr{L}$ cannot have a finite algebraic base.*

Here is a brief idea of the proof for aperiodic language classes (like CTL* and $FO[\prec]$): If there were a finite algebraic base for these classes, there would exist an integer m such for every algebra (H, V) of the base and all $v \in V$, $v^m = v^{m+1}$. It is shown, by induction on the length of the product, that an iterated wreath product of algebras satisfying $v^m = v^{m+1}$ cannot recognize L_m.

VI. EF

The following theorem is proved in [3].

Theorem 9 *$L \subseteq H_A$ is in 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 a decision

procedure for determining whether or not a forest language given, say, by an automaton that recognizes it, is definable in EF. This procedure can also be adapted to testing whether a tree language is EF-definable with tree semantics.

In light of Theorem 5, we have the following result:

Theorem 10 *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 this statement is purely algebraic—it makes no mention of trees, forests, languages or logic. This suggests that it might be proved in quite a different fashion, reasoning solely from the structure of the forest algebra. This would provide a different proof of Theorem 9.

We will present precisely such a proof of Theorem 10 in the full paper. Here we give a rough outline. It is straightforward to show that iterated wreath products of copies of $\mathcal{U}_1$ satisfy the given identities: We simply verify them for $\mathcal{U}_1$ and show that they are preserved under wreath product and division. For the hard direction, we suppose (H, V) satisfies the identities. The sum of all elements of H is necessarily the unique absorbing element, which, following our usual practice, we denote ∞. A *subminimal* element h of H has the property that for all $v \in V$, $vh = h$ or $vh = \infty$. For each such h, we define H_h to be the *set* $\{\infty\} \cup \{h' : h \in Vh'\}$. It is possible to define an additive structure and an action V_h on each H_h so that (H, V) embeds into the direct product of the (H_h, V_h) over all subminimal elements h, and such that each of the algebras (H_h, V_h) satisfies the identities. If there is more than one subminimal element, then each H_h has cardinality strictly smaller than H, and the result follows by induction on $|H|$. If H has a unique subminimal element, then it is possible to define an additive structure on $H - \{0\}$ and an action V' such that $(H - \{0\}, V')$ satisfies the identities, and such that (H, V) embeds in the wreath product $(H - \{0\}, V') \circ \mathcal{U}_1$. Again the result follows by induction on $|H|$.

Theorem 10 is the exact analogue for forest algebras of a Theorem of Stiffler [17] that a monoid is $\mathcal{R}$-trivial if and only if it divides a wreath product of copies of U_1.

VII. 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 will apply these results in the next subsection to prove nonexpressibility results for these logics. 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 [7] and Wilke [19].

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 set of holes of a multicontext p is denoted $\mathrm{holes}(p)$. A *valuation* on p is a map $\mu : \mathrm{holes}(p) \to X$, 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 $\mu : \mathrm{holes}(p) \to G$. When $G = \{g\}$ is a singleton, we just write $p[g]$. For $g \in G$ and $x \in \mathrm{holes}(p)$ 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.)

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 $x \in \mathrm{holes}(p)$:

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

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.

We say that (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.

Theorem 11

- *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.*

The proofs of all three assertions follow the same general outline: We show that the base algebras for each class satisfy the relevant non-confusing condition, and that the condition is preserved under wreath product and homomorphic images. (Preservation by subalgebras is trivial.) It is somewhat difficult to prove this directly for the base algebras for $FO[\prec]$ and graded PDL, but in these instances we can

decompose further, and obtain a base with two different kinds of algebras: $\mathsf{E}^k L$ is the composition of languages of the form $\mathsf{E}^k a$ and languages of the form $\mathsf{E}L'$, where L' is first-order definable if L is. Since languages of the form $\mathsf{E}L$ have distributive syntactic algebras, this is an easier base to work with.

Theorem 12 *It is decidable if a given algebra has a horizontal or vertical confusion.*

The first observation needed for the proof is that it is enough to consider multicontexts over alphabets consisting of vertical elements of the algebra. Then one can just enumerate all possible candidates, as what matters is not really the shape of a multicontext, but the behavior of a multicontext as function. This is rather straightforward for vertical confusion. It is bit more complicated for the horizontal variant.

VIII. Applications

Example: *(Binary trees with every leaf at even depth.)* A *labeled binary tree* is a tree where every node is either a leaf, or has exactly two children: one with label a and the other with label b. (The root label is irrelevant.) Let L_1 be the set of labeled binary trees t over $A = \{a, b\}$ where every leaf is at even depth.

A trick due to Potthoff [15] can be used to show, somewhat surprisingly, that L_1 is definable in $FO[\prec]$. Here we use Theorem 11 to show that L_1 is not definable in CTL. We will use two classes of the syntactic congruence: the class h_0 (respectively, h_1), which contains all concatenations $s+t$ of two labeled binary trees with different root labels, such that all leaves in $s+t$ are at even (respectively, odd) depth. Now let p be the multicontext $a\Box + b\Box$. Observe that $p[h_0] = h_1$ and $p[h_1] = h_0$. So (H_{L_1}, V_{L_1}) has vertical confusion with respect to p. By Theorem 11, (H_{L_1}, V_{L_1}) is not a CTL-algebra, and thus L_1 is not definable in CTL.

Example: *(Binary trees with every leaf at even depth, continued)* Now we consider the set L_2 of *unlabeled binary trees* (that is, trees over a one-letter alphabet $\{a\}$ where every node is either a leaf, or has exactly two children) such that every path from the root to a leaf has even length. Pothoff's method just cited can be used to show that this is definable in first-order logic with the ancestor and next-sibling relations (the next-sibling relation is used to recover the a and b labels). We will show, however, that the ancestor relation alone is not sufficient to recognize L_2. In other words, L_2 is not definable in $FO[\prec]$. Let h_0 (respectively, h_1) be the set of binary trees where every leaf is at even (respectively, odd) depth. (Note that h_0 is the language L_2 itself.) Let $p = a(\Box + \Box)$. This is a uniform multicontext, and we have $p[h_0] = h_1$, $p[h_1] = h_0$, so by Theorem 11, L_2 is not $FO[\prec]$-definable.

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).

Example: *(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 uniform multicontext $p = \vee(\wedge(\Box + \Box) + \wedge(\Box + \Box))$. If we put h_0 in one of the holes then by putting h_i in all other holes we will , obtain h_i (for $i = 0, 1$), and analogously if we put h_1 in one of the holes. So the syntactic algebra of L_3 has horizontal confusion with respect to p. Thus L_3 is not definable 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.

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. In the full paper, we will prove the following fact.

Theorem 13 *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 $(H'_1, V'_1) \circ \cdots \circ (H'_k, V'_k)$.*

Theorem 5 immediately yields the following corollary:

Theorem 14 *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 of these facts is already known in a somewhat different form: properties expressible in CTL* are exactly

the bisimulation-invariant properties expressible in monadic path logic (see Moller and Rabinovich [14].)

IX. Conclusion and further research

Results like those in Section VIII 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 11 are sufficient. The approach outlined in Section VI 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 transformation monoid divides a wreath product of transformation 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 transformation monoid $\mathcal{U}_2$, and a notion of a difficult transformation monoid (a group). 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, just as in the case of words. Finding (at least one) Krohn-Rhodes theorem for trees is probably the most ambitious goal of this research.

References

[1] P. Barceló and L. Libkin, "Temporal logics over unranked trees," in *LICS*, 2005, pp. 31–40.

[2] M. Benedikt and L. Segoufin, "Regular tree languages definable in FO," in *STACS*, ser. LNCS, vol. 3404, 2005, pp. 327–339.

[3] M. Bojanczyk and I. Walukiewicz, "Forest algebras," in *Logic and Automata: History and Perspectives*, J. Flum, E. Graedel, and T. Wilke, Eds. Amsterdam University Press, 2008.

[4] M. Bojanczyk, "Two-way unary temporal logic over trees," in *LICS*, 2007, pp. 121–130.

[5] M. Bojanczyk and L. Segoufin, "Tree languages defined in first-order logic with one quantifier alternation," in *ICALP*, ser. LNCS, vol. 5126, 2008, pp. 233–245.

[6] M. Bojanczyk, L. Segoufin, and H. Straubing, "Piecewise testable tree languages," in *LICS*, 2008, pp. 442–451.

[7] J. Cohen, D. Perrin, and J.-E. Pin, "On the expressive power of temporal logic," *J. Comput. Syst. Sci.*, vol. 46, no. 3, pp. 271–294, 1993.

[8] Z. Ésik and P. Weil, "On logically defined recognizable tree languages," in *FSTTCS*, ser. LNCS, vol. 2914, 2003, pp. 195–207.

[9] ——, "Algebraic recognizability of regular tree languages," *Theor. Comput. Sci*, vol. 340, no. 1, pp. 291–321, 2005.

[10] Z. Ésik, "Characterizing CTL-like logics on finite trees," *Theor. Comput. Sci.*, vol. 356, no. 1-2, pp. 136–152, 2006.

[11] Z. Ésik and S. Iván, "Aperiodicity in tree automata," in *CAI*, ser. LNCS, vol. 4728, 2007, pp. 189–207.

[12] T. Hafer and W. Thomas, "Computation tree logic CTL and path quantifiers in the monadic theory of the binary tree," in *ICALP*, ser. LNCS, vol. 267, 1987, pp. 260–279.

[13] L. Libkin, "Logics for unranked trees: an overview," in *ICALP*, ser. LNCS, vol. 3580, 2005, pp. 35–50.

[14] F. Moller and A. M. Rabinovich, "Counting on CTL*: on the expressive power of monadic path logic," *Inf. Comput.*, vol. 184, no. 1, pp. 147–159, 2003.

[15] A. Potthoff, "First-order logic on finite trees," in *Theory and Practice of Software Development*, ser. LNCS, vol. 915, 1995, pp. 125–139.

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

[17] P. Stiffler, "Extensions of the fundamental theory of finite semigroups," *Advances in Mathematics*, vol. 11, pp. 159–209, 1973.

[18] H. Straubing, *Finite automata, formal logic, and circuit complexity*, ser. Progress in Theoretical Computer Science. Boston, MA: Birkhäuser Boston Inc., 1994.

[19] T. Wilke, "Classifying discrete temporal properties," in *STACS*, ser. LNCS, vol. 1563, 1999, pp. 32–46.