An Algebraic Characterization of Temporal Logics on Finite Trees, Part 1

Z. Ésik*
Dept. of Computer Science, University of Szeged
Szeged, Hungary
and
Research Group on Mathematical Linguistics, Rovira i Virgili University
Tarragona, Spain

Abstract

We associate a modal operator with each language belonging to a given class of regular tree languages and use the cascade product of tree automata to give an algebraic characterization of the expressive power of the resulting logic.

1 Introduction

The cascade product and its semigroup theoretic variants have been very useful and powerful tools in the characterization of the expressive power of several logics over finite words, including first-order logic and its extension with modular counting, [17, 24, 23], linear temporal logic and the until hierarchy, [6, 26], and modular temporal logic, [3], to mention a few references. In this paper, our aim is to show that the cascade product of tree automata has the same potential in the characterization of the expressive power of various CTL-like temporal logics on finite trees. We associate a modal operator with each language belonging to a given class of regular tree languages and use the cascade product of tree automata to give an algebraic characterization of the expressive power of the resulting logic. From our general results, we deduce algebraic characterizations of the expressive power of several specific logics on finite trees related to CTL, cf. [20]. Some of our results are extensions of the corresponding facts for finite words proved in [10].

*Supported in part by a grant from the National Foundation of Hungary for Scientific Research, grant T46686.

Some notation When n is a natural number, we denote the set $\{1, \ldots, n\}$ by $[n]$. Thus, $[0]$ is another name for the empty set. When A is a set, A^* denotes the set of all finite words over A including the empty word.

2 Algebras

A *rank type* R is a finite nonempty set of nonnegative integers. To avoid trivial situations, we assume that each rank type contains a positive integer. A *ranked alphabet* Σ of rank type R is a disjoint union of finite sets Σ_n, $n \geq 0$, such that for each n, $\Sigma_n \neq \emptyset$ iff $n \in R$. The elements of Σ_n are called *letters* or *symbols* of rank n, or when $n = 0$, *constant symbols.*

Suppose that Σ is a ranked alphabet of rank type R. A Σ-*algebra* $\mathbb{A}$ consists of a nonempty set A, called the *carrier* of $\mathbb{A}$, and an operation $\sigma_{\mathbb{A}} : A^n \to A$, for each $\sigma \in \Sigma_n$, $n \geq 0$, called the *interpretation of* Σ. Homomorphisms, subalgebras, congruences, direct products, etc. are defined as usual, see, e.g., [14]. Suppose that $\mathbb{A} = (A, (\sigma_{\mathbb{A}})_{\sigma \in \Sigma})$ is a Σ-algebra and $\mathbb{B} = (B, (\delta_{\mathbb{B}})_{\delta \in \Delta})$ is a Δ-algebra, where Σ and Δ are of the same rank type. We call $\mathbb{A}$ a *renaming* of $\mathbb{B}$ if $A = B$ and for each $\sigma \in \Sigma_n$, $n \geq 0$ there is a symbol $\delta \in \Delta_n$ with $\sigma_{\mathbb{A}} = \delta_{\mathbb{B}}$.

We will take the liberty of writing just σ for $\sigma_{\mathbb{A}}$ whenever the algebra $\mathbb{A}$ is clear from the context. We call the algebra $\mathbb{A}$ *finite* if its carrier is finite. Sometimes we do not specify the carrier of an algebra explicitly and follow the practice that if $\mathbb{A}, \mathbb{B}$ etc. denote algebras, then $A, B \ldots$ denote the corresponding carriers.

3 Trees and Tree Automata

Suppose that Σ is a ranked alphabet. Let $x_1, x_2, \ldots$ be a fixed countable sequence of variables, and for each $n \geq 0$, let X_n denote the set $\{x_1, \ldots, x_n\}$. The set $T_\Sigma(X_n)$ of n-*ary* Σ-*trees* is defined as the least set containing Σ_0 and X_n (which are assumed to be disjoint) such that whenever $t_1, \ldots, t_m$ are in $T_\Sigma(X_n)$ and $\sigma \in \Sigma_m$, then $\sigma(t_1, \ldots, t_m)$ is also in $T_\Sigma(X_n)$. When $n = 0$, we write just T_Σ. The elements of T_Σ are called *ground trees.* Note that T_Σ is nonempty iff Σ_0 is nonempty. Sometimes it is convenient to represent an n-ary tree as a directed graph which is a rooted tree equipped with a labeling function that maps vertices to letters in $\Sigma \cup X_n$ such that the outgoing edges of each vertex are linearly ordered. Moreover, a vertex is labeled in $\Sigma_0 \cup X_n$ iff it is a leaf, i.e., it has no successor, and is labeled in Σ_m for some $m > 0$ iff it has m immediate successors. The label of a vertex v in a tree will be denoted $t(v)$. The notion of *subtree of a tree* t *rooted at a vertex* v, denoted t_v, is defined as usual. The *immediate subtrees* of a tree are those rooted at the immediate successors of the root. The *depth* of a vertex v in a tree t is the length of the unique path from the root to v, where the depth of the root is 0. The depth of a tree is the length of the longest path in the tree.

Trees $c \in T_\Sigma(X_1)$ with a single leaf labeled x_1 are called *contexts*. A *primitive context* is a context of the form $\sigma(t_1, \ldots, t_{i-1}, x_1, t_{i+1}, \ldots, t_n)$, where $\sigma \in \Sigma_n$, $n > 0$, $i \in [n]$, and $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n \in T_\Sigma$. Thus, a primitive context is a context such that the leaf labeled x_1 occurs at depth 1. We let CT_Σ denote the set of all contexts in $T_\Sigma(X_1)$.

Suppose that $t \in T_\Sigma(X_n)$ and $t_1, \ldots, t_n \in T_\Sigma(X_m)$ are trees. Then the tree resulting from t by substituting, for each $i \in [n]$, a copy of t_i for each occurrence of x_i in t, is denoted $t(t_1, \ldots, t_n)$. Note that this tree is in $T_\Sigma(X_m)$. The formal definition goes by induction on the structure of t. If $t = \sigma \in \Sigma_0$, then $t(t_1, \ldots, t_n) = \sigma$, and if $t = x_i$ for some $i \in [n]$, then $t(t_1, \ldots, t_n) = t_i$. Last, if $t = \sigma(s_1, \ldots, s_k)$ with $\sigma \in \Sigma_k$, $s_1, \ldots, s_k \in T_\Sigma(X_n)$, $k > 0$, then $t(t_1, \ldots, t_n) = \sigma(s_1(t_1, \ldots, t_n), \ldots, s_k(t_1, \ldots, t_n))$.

If $\mathbb{A}$ is a Σ-algebra with carrier A and $t \in T_\Sigma(X_n)$, then t *induces* a function $A^n \to A$, denoted $t_\mathbb{A}$, or just t. The definition is standard, see, e.g., [13] or [14]. When $n = 0$, we identify $t_\mathbb{A}$ with an element of A.

To simplify the treatment, our temporal logics will be tailored so that only sets of ground trees will be definable. Accordingly, a *tree language* over Σ is a set $L \subseteq T_\Sigma$ of ground Σ-trees. In order to avoid trivial situations, when we speak of tree languages, we will always assume that the underlying rank type contains 0, so that each ranked alphabet contains constant symbols.

Suppose that Σ is a ranked alphabet of rank type R with $0 \in R$. A *Σ-tree automaton* is a Σ-algebra that contains no proper subalgebras. A tree automaton is finite if it is a finite algebra. A *homomorphism of Σ-tree automata* is a Σ-algebra homomorphism. Note that if $\mathbb{A}$ and $\mathbb{B}$ are Σ-tree automata, then there is at most one homomorphism $\mathbb{A} \to \mathbb{B}$. Moreover, any homomorphism of tree automata is a surjective function.

Let $\mathbb{A} = (A, (\sigma_\mathbb{A})_{\sigma \in \Sigma})$ be a tree automaton. The language *accepted* or *recognized by $\mathbb{A}$ with final states $F \subseteq A$* is defined by

$$L(\mathbb{A}, F) \quad = \quad \{t \in T_\Sigma : t_\mathbb{A} \in F\}.$$

A tree language $L \subseteq T_\Sigma$ is recognizable by the tree automaton $\mathbb{A}$ if $L = L(\mathbb{A}, F)$ for some $F \subseteq A$. A tree language $L \subseteq T_\Sigma$ is *regular* if it is recognizable by a finite tree automaton.

Each tree language $L \subseteq T_\Sigma$ is recognizable by a canonical tree automaton (unique up to isomorphism), the *minimal tree automaton $\mathbb{A}_L$ of L*. It has the universal property that whenever L is recognizable by a tree automaton $\mathbb{A}$ then there is a (necessarily unique) homomorphism $\mathbb{A} \to \mathbb{A}_L$. Thus, a language L is regular iff its minimal tree automaton is finite. It is known that a Σ-tree automaton $\mathbb{A} = (A, (\sigma_\mathbb{A})_{\sigma \in \Sigma})$ is isomorphic to the minimal tree automaton of $L \subseteq T_\Sigma$ iff $L = L(\mathbb{A}, F)$ for some (necessarily unique) $F \subseteq A$, and for any

$a, b \in A$ with $a \neq b$ there is a context c with $c(a) \in F$ and $c(b) \notin F$, or $c(a) \notin F$ and $c(b) \in F$. Thus, the languages recognizable by $\mathbb{A}_L$ are unions of $\sim_L$-equivalence classes, where the relation $\sim_L$ on T_Σ is defined by

$$t \sim_L t' \quad \Leftrightarrow \quad \forall c \in CT_\Sigma \ (c(t) \in L \Leftrightarrow c(t') \in L).$$

In particular, L is regular iff $\sim_L$ is of finite index. For the reader's convenience, we include a proof of the following well-known fact.

Lemma 3.1 *Suppose that $L \subseteq T_\Sigma$ is regular. Then every language recognizable by $\mathbb{A}_L$ is a boolean combination of quotients of L.*

Proof. We know that the languages recognizable by $\mathbb{A}_L$ are unions of $\sim_L$-equivalence classes. But each $\sim_L$-equivalence class $[t]$ can be written as

$$\bigcap_{t \in c^{-1}L} c^{-1}L \setminus \bigcup_{t \notin c^{-1}L} c^{-1}L,$$

where the *quotient $c^{-1}L$ of L with respect to c* is the set of all trees $t \in T_\Sigma$ with $c(t) \in L$. The intersection and the union in the above formula are finite since L is regular and each language $c^{-1}L$ is recognizable by $\mathbb{A}_L$. Thus, every $\sim_L$-equivalence class $[t]$ and every language recognizable by $\mathbb{A}_L$ is the boolean combination of quotients of L. $\square$

Suppose that a rank type R with $0 \in R$ is fixed. By a *class $\mathcal{L}$ of tree languages* we mean a collection of tree languages in T_Σ for each ranked alphabet Σ (of rank type R). A class of regular tree languages consists of regular languages.

Let Σ and Δ be two ranked alphabets of the same rank type. Given a tree $t \in T_\Sigma(X_n)$, a *relabeling* of t over Δ is obtained by changing the label of each vertex of t labeled in Σ_m to some symbol in Δ_m, for each $m \geq 0$. Labels in X_n do not change. Different occurrences of the same letter in Σ may be replaced by different letters. A related notion is that of a *literal tree homomorphism.* Suppose that h is a rank preserving function $\Sigma \to \Delta$. Then for each n, h determines a literal tree homomorphism $T_\Sigma(X_n) \to T_\Delta(X_n)$, also denoted h. The image of a tree $t \in T_\Sigma(X_n)$ is obtained from t by relabeling each vertex labeled $\sigma \in \Sigma$ by the letter $h(\sigma)$ (of the same rank). It is known that the class of regular tree languages is closed under literal homomorphisms and inverse literal homomorphisms. Thus, if $L \subseteq T_\Delta$ is regular and h is a literal homomorphism as described above, then $h^{-1}(L) = \{t \in T_\Sigma : h(t) \in L\}$ is regular.

In addition to relabelings and literal tree homomorphisms, we will also make use of quotients defined in the proof of Lemma 3.1. It is known that the class of regular tree languages is closed under quotients, i.e., if $L \subseteq T_\Sigma$ is regular and $c \in CT_\Sigma$, then $c^{-1}L$ is regular. Moreover, a tree language $L \subseteq T_\Sigma$ is regular iff it has a finite number of different quotients, i.e., when the set $\{c^{-1}L : c \in CT_\Sigma\}$ is finite. It is clear that a class $\mathcal{L}$ of tree languages is closed under quotients iff it is closed under quotients with respect to primitive contexts.

Other operations under which the class of regular languages is closed include the boolean operations.

For the above facts and more results on tree automata and tree languages, refer to any standard text such as [13].

4 Extended Temporal Logic

We now define our temporal logics on trees. We assume that a rank type R with $0 \in R$ is fixed and only consider ranked alphabets of rank type R. Further, we assume that each ranked alphabet comes with a fixed lexicographic order.

Syntax. For a ranked alphabet Σ, the set of *formulas over* Σ *is* the least set containing the letters p_σ, for all $\sigma \in \Sigma$, closed with respect to the boolean connectives $\vee$ (disjunction) and $\neg$ (negation), as well as the following construct. Suppose that $L \subseteq T_\Delta$ and that for each $\delta \in \Delta$, φ_δ is a formula over Σ. Then

$$L(\delta \mapsto \varphi_\delta)_{\delta \in \Delta} \tag{1}$$

is a formula over Σ. The notion of *subformula* of a formula is defined as usual.

Semantics. Suppose that φ is a formula over Σ and $t \in T_\Sigma$. We say that t *satisfies* φ, in notation $t \models \varphi$, if

- $\varphi = p_\sigma$, for some $\sigma \in \Sigma_n$, and the root of t is labeled σ, i.e., $t = \sigma(t_1, \ldots, t_n)$, for some $t_1, \ldots, t_n$, or
- $\varphi = \varphi' \vee \varphi''$ and $t \models \varphi'$ or $t \models \varphi''$, or
- $\varphi = \neg\varphi'$ and it is not the case that $t \models \varphi'$, or
- $\varphi = L(\delta \mapsto \varphi_\delta)_{\delta \in \Delta}$, and the *characteristic tree* $\widehat{t} \in T_\Delta$ determined by t and the family $(\varphi_\delta)_{\delta \in \Delta}$ belongs to L. Here, $\widehat{t}$ has the same underlying digraph as t, and a vertex v is labeled $\delta \in \Delta_n$ in $\widehat{t}$ iff v is labeled by some $\sigma \in \Sigma_n$ in the tree t, moreover, δ is the first in lexicographic order on Δ_n such that the subtree of t rooted at v satisfies φ_δ, i.e., $t_v \models \varphi_\delta$. If no such letter exists, then δ is the last in the lexicographic order on Δ_n.

For any formula φ of over Σ, we let L_φ denote the *language defined by* φ:

$$L_\varphi \quad = \quad \{t \in T_\Sigma : t \models \varphi\}.$$

We say that formulas φ and ψ over Σ are *equivalent* if $L_\varphi = L_\psi$. Throughout the paper we will use the boolean connectives $\wedge$ (conjunction) and $\to$ (implication) as abbreviations. Moreover, for any ranked alphabet Σ and $n \in R$, we define $\mathtt{tt}_n = \bigvee_{\sigma \in \Sigma_n} p_\sigma$ and $\mathtt{ff}_n = \neg\mathtt{tt}_n$. Thus, $t \models \mathtt{tt}_n$ iff the root of t is labeled in Σ_n. We further let $\mathtt{tt} = p_\sigma \vee \neg p_\sigma$, where σ is any letter in Σ and $\mathtt{ff} = \neg\mathtt{tt}$.

We will consider subsets of formulas associated with classes of tree languages. When $\mathcal{L}$ is a class of tree languages, we let $\mathrm{FTL}(\mathcal{L})$ denote the collection of formulas all of whose subformulas of the form (1) above are such that L belongs to $\mathcal{L}$. We define $\mathbf{FTL}(\mathcal{L})$ to be the class of all languages definable by formulas in FTL($\mathcal{L}$). It is clear that for each formula $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ in FTL($\mathcal{L}$) over an alphabet Σ there is an equivalent formula $L(\delta \mapsto \varphi'_\delta)_{\delta\in\Delta}$ in FTL($\mathcal{L}$) over Σ such that the subformulas φ'_δ satisfy the following condition: There exist no $t \in T_\Sigma$ and distinct letters $\delta, \delta' \in \Delta_n$, for some n, such that $t \models \varphi'_\delta \wedge \varphi'_{\delta'}$. Indeed, when the lexicographic order on Δ_n is $\delta_1 < \ldots < \delta_k$, then we may define

$$\varphi'_{\delta_i} \quad = \quad \varphi_{\delta_i} \wedge \bigwedge_{j<i} \neg\varphi_{\delta_j},$$

for all $i \in [k]$. Alternatively, we may define

$$\varphi'_{\delta_i} \quad = \quad \mathtt{tt}_n \wedge \varphi_{\delta_i} \wedge \bigwedge_{j<i} \neg\varphi_{\delta_j},$$

for all $i < k$ as above, and

$$\varphi'_{\delta_k} \quad = \quad \mathtt{tt}_n \wedge \bigwedge_{j<k} \neg\varphi_{\delta_j}.$$

Thus, the modal formulas in FTL($\mathcal{L}$) over Σ associated with a language $L \subseteq T_\Delta$ in $\mathcal{L}$ may equivalently be written as $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$, where the family $(\varphi_\delta)_{\delta\in\Delta}$ satisfies the following condition: For each tree $t \in T_\Sigma$ there is exactly one δ with $t \models \varphi_\delta$, and if the root of t is labeled in Σ_n, then this unique δ belongs to Δ_n. Below we will call such families $(\varphi_\delta)_{\delta\in\Delta}$ *deterministic*. Accordingly, we will sometimes write modal formulas over Σ associated with a language $L \subseteq T_\Delta$ as $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$, where $(\varphi_\delta)_{\delta\in\Delta}$ is a deterministic family of formulas over Σ. When $(\varphi_\delta)_{\delta\in\Delta}$ is a deterministic family, we have $t \models L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ iff there exists a relabeling $\widehat{t}$ of t in L such that for all vertices v, $t_v \models \varphi_{\widehat{t}(v)}$. We call a formula φ deterministic if for every subformula of φ of the form $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$, the family $(\varphi_\delta)_{\delta\in\Delta}$ is deterministic. As shown above, for each $\varphi \in$ FTL($\mathcal{L}$) there is a deterministic formula in FTL($\mathcal{L}$) which is equivalent to φ.

Remark 4.1 When $R = \{0, 1\}$, our logics FTL($\mathcal{L}$) are closely related to the extended propositional linear temporal logics of [27].

Remark 4.2 Suppose that each Δ_n with $n \in R$ contains exactly k letters, $\delta_1^{(n)}, \ldots, \delta_k^{(n)}$, ordered as indicated. For each $i \in [k]$, let φ_i denote a formula over Σ and let $L \subseteq T_\Delta$. Then let $L(\delta_i \mapsto \varphi)_{i\in[k]}$ denote the formula $L(\delta_i^{(n)} \mapsto \varphi_i)_{i\in[k],n\in R}$. Thus, a tree $t \in T_\Sigma$ satisfies this formula iff the tree $s \in T_\Delta$ is in L, where s is obtained from t by relabeling each vertex v of t labeled in Σ_n by the first letter $\delta_i^{(n)}$ with $t_v \models \varphi_i$. When there is no such formula, then the label of v in s is $\delta_k^{(n)}$.

Each formula $L(\delta_i^{(n)} \mapsto \psi_i^{(n)})_{i\in[k],n\in R}$ is equivalent to some formula of the above sort. Indeed, define

$$\varphi_i \quad = \quad \bigwedge_{i\in R} (\mathtt{tt}_n \to \varphi_i^{(n)}),$$

for all $i \in [k]$. Then $L(\delta_i^{(n)} \mapsto \psi_i^{(n)})_{i\in[k],n\in R}$ is equivalent to $L(\delta_i \mapsto \varphi_i)_{i\in[k]}$.

Example 4.3 For each alphabet Σ, $\mathbf{FTL}(\emptyset)$ consists of those languages that are unions of languages of the form $\{\sigma(t_1,\ldots,t_n) : t_1,\ldots,t_n \in T_\Sigma\}$, where $\sigma \in \Sigma_n$, $n \geq 0$.

Example 4.4 The *boolean ranked alphabet* Bool has exactly two symbols of rank n, for each $n \in R$, the symbols $\uparrow_n$ and $\downarrow_n$. Below we assume that the lexicographic order on Bool satisfies $\uparrow_n < \downarrow_n$, for each $n \in R$.

For each $i \in [\max(R)]$, let L_{X_i} denote the regular tree language of all trees in T_{Bool} of depth ≥ 1 such that the root has n immediate successors for some $n \geq i$, and the ith immediate successor of the root is labeled by $\uparrow_m$, for some m. Then the modal operator corresponding to L_{X_i} is a sort of next modality: When $(\varphi_\delta)_{\delta\in\mathrm{Bool}}$ is a family of formulas over Σ and $t \in T_\Sigma$, then $t \models L_{\mathrm{X}_i}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$ iff the root of t is labeled by some symbol in Σ_n with $i \leq n$, and the ith immediate subtree satisfies $\varphi_{\uparrow_m}$, where m denotes the rank of the symbol labeling the root of this subtree. Let $L_{\mathrm{X}} = \cup_{i\in[\max(R)]} L_{\mathrm{X}_i}$. Then $t \models L_{\mathrm{X}}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$ iff t is of depth ≥ 1 and the subtree of t rooted at some immediate successor of the root vertex satisfies $\varphi_{\uparrow_m}$ for that m for which the successor is labeled in Σ_m.

Thus, when φ is a fixed formula over Σ and $(\varphi_\delta)_{\delta\in\Delta}$ is such that $\varphi_{\uparrow_n} = \varphi$, for all $n \in R$, then $t \models L_{\mathrm{X}_i}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$ for a tree t in T_Σ and $i \in [\max(R)]$ iff the depth of t is at least 1 and the ith immediate subtree of t exists and satisfies φ. We may denote this formula by $\mathrm{X}_i\varphi$. Note also that $t \models L_{\mathrm{X}}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$ iff some immediate subtree of t satisfies φ. Conversely, if $(\varphi_\delta)_{\delta\in\mathrm{Bool}}$ is any family of formulas over Σ, then $L_{\mathrm{X}_i}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$ may be expressed as $\mathrm{X}_i(\bigwedge_{n\in R}(\mathtt{tt}_n \to \varphi_{\uparrow_n}))$.

Next, let $L_{\mathrm{EF}} \subseteq T_{\mathrm{Bool}}$ denote the regular language of those trees in T_{Bool} having at least one vertex labeled in $\{\uparrow_n : n \in R\}$. Then for any $(\varphi_\delta)_{\delta\in\mathrm{Bool}}$ and t as above, $t \models L_{\mathrm{EF}}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$ iff the subtree rooted at some vertex labeled in Σ_n, for some n, satisfies $\varphi_{\uparrow_n}$. Thus, the modal operator corresponding to this language L_{EF} is closely related to the EF modality of CTL, cf. [20]. In the same way, the CTL-modalities AG, EG, AF are closely related to the modal operators associated with the following languages, where we use the letter p to range over the maximal paths of a tree, and v ranges over vertices:

$$\begin{aligned}
L_{\mathrm{AG}} &= \{t \in T_{\mathrm{Bool}} : \forall v\ t(v) \in \{\uparrow_n : n \in R\}\} \\
L_{\mathrm{EG}} &= \{t \in T_{\mathrm{Bool}} : \exists p \forall v \in p\ t(v) \in \{\uparrow_n : n \in R\}\} \\
L_{\mathrm{AF}} &= \{t \in T_{\mathrm{Bool}} : \forall p \exists v \in p\ t(v) \in \{\uparrow_n : n \in R\}\}.
\end{aligned}$$

Example 4.5 Next we define tree languages such that the corresponding modal operators are closely related to the EU and AU modalities of CTL. For this reason, we consider the ranked alphabet Tern having three symbols for each

$n \in R$, $\uparrow_n, \vee_n, \downarrow_n$, ordered as indicated. Below we will write $u < v$ for vertices u, v in the tree t to express that v is strictly below u, i.e., $u \neq v$ and v is a vertex of the subtree t_u rooted at u. Let

$$\begin{aligned} L_{\mathrm{EU}} &= \{t \in T_{\mathrm{Tern}} : \exists p \exists v \forall u < v\ (t(v) \in \{\uparrow_n : n \in R\} \wedge t(u) \in \{\vee_n : n \in R\})\} \\ L_{\mathrm{AU}} &= \{t \in T_{\mathrm{Tern}} : \forall p \exists v \forall u < v\ (t(v) \in \{\uparrow_n : n \in R\} \wedge t(u) \in \{\vee_n : n \in R\})\}, \end{aligned}$$

where p ranges over maximal paths as above. The modal operators corresponding to these languages are closely related to the EU and AU modalities of CTL.

Example 4.6 Last, we consider a version of modular counting. Let $d > 1$ and r with $0 \leq r < d$ be given natural numbers. Let $L_{d,r}$ denote the set of all those trees in T_{Bool} such that the number of vertices labeled in $\{\uparrow_n : n \in R\}$ is congruent to r modulo d. If t is a tree in T_Σ and $(\varphi_\delta)_{\delta \in \mathrm{Bool}}$ is a family of formulas over Σ, then $t \models L_{d,r}(\delta \mapsto \varphi_\delta)_{\delta \in \Delta}$ iff the number of vertices v labeled in Σ_n, $n \in R$ with $t_v \models \varphi_{\uparrow_n}$ is congruent to r modulo d.

5 Basic Results

In this section, we establish some elementary properties of the classes $\mathbf{FTL}(\mathcal{L})$, where $\mathcal{L}$ denotes a class of tree languages. We also study conditions on $\mathcal{L}$ and $\mathcal{L}'$ under which $\mathbf{FTL}(\mathcal{L}) = \mathbf{FTL}(\mathcal{L}')$. We again assume that a rank type R with $0 \in R$ is fixed and that all considered ranked alphabets have rank type R.

Theorem 5.1 *For each class $\mathcal{L}$ of tree languages, $\mathbf{FTL}(\mathcal{L})$ contains $\mathcal{L}$ and is closed with respect to the boolean operations and inverse literal tree homomorphisms.*

Proof. It is obvious that $\mathbf{FTL}(\mathcal{L})$ is closed under the boolean operations. Moreover, each language $L \subseteq T_\Sigma$ in $\mathcal{L}$ is definable by the formula $L(\sigma \mapsto p_\sigma)_{\sigma \in \Sigma}$ in $\mathrm{FTL}(\mathcal{L})$. Assume now that $h : T_{\Sigma'} \to T_\Sigma$ is a literal tree homomorphism. We argue by induction on the structure of the formula φ over Σ in $\mathrm{FTL}(\mathcal{L})$ to show that $h^{-1}(L_\varphi)$ is definable by some formula ψ in $\mathrm{FTL}(\mathcal{L})$. When $\varphi = p_\sigma$, for some letter σ, then we define $\psi = \bigvee_{h(\sigma')=\sigma} p_{\sigma'}$. When σ is not in the range of h then this formula is ff. It is clear that $L_\psi = h^{-1}(L_\varphi)$. Suppose now that $\varphi = \varphi_1 \vee \varphi_2$ and that $L_{\psi_i} = h^{-1}(L_{\varphi_i})$, $i = 1, 2$. Then we define $\psi = \psi_1 \vee \psi_2$. When $\varphi = \neg\varphi_1$ and $L_{\psi_1} = h^{-1}(L_{\varphi_1})$, then let $\psi = \neg\psi_1$. In either case, we have $L_\psi = h^{-1}(L_\varphi)$. Finally, assume that $\varphi = L(\delta \mapsto \varphi_\delta)_{\delta \in \Delta}$, and that for each δ there is a formula ψ_δ in $\mathrm{FTL}(\mathcal{L})$ with $L_{\psi_\delta} = h^{-1}(L_{\varphi_\delta})$. Then define $\psi = L(\delta \mapsto \psi_\delta)_{\delta \in \Delta}$. Let $t \in T_{\Sigma'}$. Since for all $\delta \in \Delta$ and vertex v,

$$t_v \models \psi_\delta \Leftrightarrow h(t_v) \models \varphi_\delta \Leftrightarrow (h(t))_v \models \varphi_\delta,$$

the characteristic tree determined by t and the formulas $(\psi_\delta)_{\delta \in \Delta}$ is the same as that determined by $h(t)$ and the formulas φ_δ. It follows that $t \models \psi$ iff $h(t) \models \varphi$. $\square$

To prove that **FTL** is a closure operator, we need:

Lemma 5.2 *Suppose that $(\varphi_\delta)_{\delta\in\Delta}$ is a deterministic family of formulas over Σ and $(\tau_\gamma)_{\gamma\in\Gamma}$ is a deterministic family of formulas over Δ. Let $t \in T_\Sigma$ and $\widehat{t}$ the characteristic tree determined by t and $(\varphi_\delta)_{\delta\in\Delta}$. Let s be the characteristic tree determined by $\widehat{t}$ and the family $(\tau_\gamma)_{\gamma\in\Gamma}$. Then s is also the characteristic tree determined by t and the deterministic family $(L_\gamma(\delta \mapsto \varphi_\delta)_{\delta\in\Delta})_{\gamma\in\Gamma}$.*

Proof. First note that $(L_{\tau_\gamma}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta})_{\gamma\in\Gamma}$ is also a deterministic family. Given a tree $t \in T_\Sigma$, for every vertex v,

$$t_v \quad \models \quad L_{\tau_{s(v)}}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta},$$

since for every vertex w in t_v, $t_w \models \varphi_{\widehat{t}(w)}$, and since $\widehat{t_v} \in L_{\tau_{s(v)}}$. □

Next we show that **FTL** is a closure operator.

Theorem 5.3 **FTL** *is a closure operator on language classes.*

Proof. We have already seen that $\mathcal{L} \subseteq \mathbf{FTL}(\mathcal{L})$ holds for all $\mathcal{L}$. It is clear that $\mathbf{FTL}(\mathcal{L}_1) \subseteq \mathbf{FTL}(\mathcal{L}_2)$ whenever $\mathcal{L}_1 \subseteq \mathcal{L}_2$. Thus, to complete the proof, it suffices to show that for any class $\mathcal{L}$ of languages, $\mathbf{FTL}(\mathbf{FTL}(\mathcal{L})) = \mathbf{FTL}(\mathcal{L})$. The inclusion from right to left follows from Theorem 5.1. To prove that $\mathbf{FTL}(\mathbf{FTL}(\mathcal{L})) \subseteq \mathbf{FTL}(\mathcal{L})$, we argue by induction on the structure of the formula φ over Δ in FTL($\mathcal{L}$) to show that for every deterministic family $(\varphi_\delta)_{\delta\in\Delta}$ of formulas in FTL($\mathcal{L}$) over an alphabet Σ, the formula $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is expressible in FTL($\mathcal{L}$), i.e., there exists a formula in FTL($\mathcal{L}$) which is equivalent to it. Assume first that $\varphi = p_{\delta_0}$, for some $\delta_0 \in \Delta_{n_0}$. Then L_φ is the set of all trees in T_Δ whose root is labeled δ_0. It is clear that a tree $t \in T_\Sigma$ satisfies $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ iff t satisfies φ_{δ_0}, so that $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to φ_{δ_0}. In the induction step, assume first that $\varphi = \varphi_1 \vee \varphi_2$. Then $L_\varphi = L_{\varphi_1} \cup L_{\varphi_2}$ and thus $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to $L_{\varphi_1}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta} \vee L_{\varphi_2}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$. By induction, there exist ψ_1 and ψ_2 in FTL($\mathcal{L}$) such that $L_{\varphi_i}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to ψ_i, $i = 1, 2$. It follows that $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to $\psi_1 \vee \psi_2$ which is in FTL($\mathcal{L}$). Suppose next that $\varphi = \neg\varphi_1$, so that $L_\varphi = \overline{L_{\varphi_1}}$, the complement of L_{φ_1}. Then $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to $\neg(L_{\varphi_1}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta})$. It follows from the induction hypothesis that $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to a formula in FTL($\mathcal{L}$). Assume finally that $\varphi = K(\gamma \mapsto \tau_\gamma)_{\gamma\in\Gamma}$, where $K \subseteq T_\Gamma$, $K \in \mathcal{L}$ and the family $(\tau_\gamma)_{\gamma\in\Gamma}$ is deterministic. Let $\widehat{t}$ denote the characteristic tree determined by t and the family $(\varphi_\delta)_{\delta\in\Delta}$, and let s denote the characteristic tree determined by $\widehat{t}$ and the family $(\tau_\gamma)_{\gamma\in\Gamma}$. By Lemma 5.2, s is also the characteristic tree determined by t and $(L_\gamma(\delta \mapsto \varphi_\delta)_{\delta\in\Delta})_{\gamma\in\Gamma}$. Thus,

$$t \models K(\gamma \mapsto L_{\tau_\gamma}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta})_{\gamma\in\Gamma} \quad \Leftrightarrow \quad s \in K.$$

We have thus shown that $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to $K(\gamma \mapsto L_{\tau_\gamma}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta})_{\gamma\in\Gamma}$. By the induction hypothesis, for each γ there is a formula ψ_γ

in $\mathrm{FTL}(\mathcal{L})$ which is equivalent to $L_{\tau_\gamma}(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$. Thus, $L_\varphi(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is equivalent to $K(\gamma \mapsto \psi_\gamma)_{\gamma\in\Gamma}$. $\square$

The language classes $\mathbf{FTL}(\mathcal{L})$ are not necessarily closed under quotients. However, we have:

Theorem 5.4 *The following conditions are equivalent for a class of tree languages $\mathcal{L}$:*

1. $\mathbf{FTL}(\mathcal{L})$ *is closed with respect to quotients.*
2. *Each quotient of any language in $\mathcal{L}$ belongs to $\mathbf{FTL}(\mathcal{L})$.*
3. *For each formula $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ in $\mathrm{FTL}(\mathcal{L})$, over any alphabet Σ, and for each context c over Δ there is a formula in $\mathrm{FTL}(\mathcal{L})$ which is equivalent to $(c^{-1}L)(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$.*
4. *For each formula $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ in $\mathrm{FTL}(\mathcal{L})$, over any alphabet Σ, and for each primitive context c over Δ there is a formula in $\mathrm{FTL}(\mathcal{L})$ which is equivalent to $(c^{-1}L)(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$.*

Proof. It is clear that the first condition implies the second and the third condition implies the fourth. Moreover, the second condition implies the third by Theorem 5.3. It remains to show that the fourth condition implies the first. Suppose that φ is a formula over Σ in $\mathrm{FTL}(\mathcal{L})$ and c is a primitive context over Σ. We show that $c^{-1}L_\varphi$ belongs to $\mathbf{FTL}(\mathcal{L})$. It follows by a straightforward induction argument that $\mathbf{FTL}(\mathcal{L})$ is closed under quotients with respect to any context.

When φ is p_σ, for a letter $\sigma \in \Sigma$, and the root of c is labeled by a letter other than σ, then $c^{-1}L_\varphi$ is $\emptyset$, which is definable by the formula $\mathtt{ff}$. When the root of c is labeled σ then $c^{-1}L_\varphi = T_\Sigma$ which is defined by the formula $\mathtt{tt}$. We continue by induction on the structure of φ. Suppose that $\varphi = \varphi_1 \vee \varphi_2$ or $\varphi = \neg\varphi_1$, and assume that $c^{-1}L_{\varphi_i}$ is defined by $\widetilde{\varphi}_i$ in $\mathrm{FTL}(\mathcal{L})$, $i = 1, 2$. Then $c^{-1}L_\varphi$ is defined by $\widetilde{\varphi}_1 \vee \widetilde{\varphi}_2$ or $\neg\widetilde{\varphi}_1$, respectively. Assume finally that φ is $L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$, where $L \subseteq T_\Delta$ and $(\varphi_\delta)_{\delta\in\Delta}$ is a deterministic family. Suppose that the root of c is labeled in Σ_{n_0}. Then for each $\delta_0 \in \Delta_{n_0}$, let c_{δ_0} denote the context over Δ obtained from c by relabeling its root by δ_0 and any other vertex u of c labeled in Σ_m, $m \geq 0$ by that letter $\delta \in \Delta_m$ such that the subtree of c rooted at u satisfies φ_δ. By the induction assumption, for any $\delta \in \Delta$ there exists a formula $\widetilde{\varphi}_\delta$ in $\mathrm{FTL}(\mathcal{L})$ defining $c^{-1}L_{\varphi_\delta}$. Moreover, by assumption, for each $\delta_0 \in \Delta_{n_0}$ there is a formula τ_{δ_0} in $\mathrm{FTL}(\mathcal{L})$ such that for all trees $t \in T_\Sigma$,

$$t \models \tau_{\delta_0} \quad \Leftrightarrow \quad t \models (c_{\delta_0}^{-1}L)(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}.$$

Then let

$$\widetilde{\varphi} \quad = \quad \bigvee_{\delta_0\in\Delta_{n_0}} (\widetilde{\varphi}_{\delta_0} \wedge \tau_{\delta_0}).$$

We have, for all $t \in T_\Sigma$,

$$\begin{aligned}
t \models \widetilde{\varphi} \quad &\Leftrightarrow \quad \exists \delta_0 \in \Delta_{n_0}\ t \models \widetilde{\varphi}_{\delta_0} \wedge t \models \tau_{\delta_0} \\
&\Leftrightarrow \quad \exists \delta_0 \in \Delta_{n_0}\ c(t) \models \varphi_{\delta_0} \wedge t \models (c_{\delta_0}^{-1} L)(\delta \mapsto \varphi_\delta)_{\delta \in \Delta} \\
&\Leftrightarrow \quad \exists \delta_0 \in \Delta_{n_0}\ c(t) \models \varphi_{\delta_0} \wedge \exists s \in c_{\delta_0}^{-1} L\ \forall v\ t_v \models \varphi_{s(v)} \\
&\Leftrightarrow \quad \exists \delta_0 \in \Delta_{n_0}, s \in T_\Delta\ c_{\delta_0}(s) \in L \wedge c(t) \models \varphi_{\delta_0} \wedge \forall v\ t_v \models \varphi_{s(v)} \\
&\Leftrightarrow \quad c(t) \models L(\delta \mapsto \varphi_\delta)_{\delta \in \Delta} \\
&\Leftrightarrow \quad c(t) \models \varphi.
\end{aligned}$$

This concludes the proof of Theorem 5.4. □

Corollary 5.5 *1. For any class $\mathcal{L}$ of tree languages, $\mathbf{FTL}(\mathcal{L}) = \mathbf{FTL}(\mathcal{L}')$, where $\mathcal{L}'$ is the least class containing $\mathcal{L}$ closed with respect to the boolean operations and inverse literal morphisms.*

2. For any class $\mathcal{L}$ of tree languages closed with respect to quotients, or such that the modal operators associated with the quotients of the languages in $\mathcal{L}$ are expressible in $\mathrm{FTL}(\mathcal{L})$ *as in Theorem 5.4, $\mathbf{FTL}(\mathcal{L}) = \mathbf{FTL}(\mathcal{L}')$, where $\mathcal{L}'$ is the least class containing $\mathcal{L}$ closed with respect to the boolean operations, quotients, and inverse literal morphisms.*

Suppose that $\mathbf{K}$ is a class of tree automata. We let $\mathcal{L}_\mathbf{K}$ denote the class of all tree languages recognizable by the tree automata in $\mathbf{K}$. Conversely, when $\mathcal{L}$ is a class of tree languages, let $\mathbf{K}_\mathcal{L}$ denote the class of all minimal tree automata of the languages in $\mathcal{L}$. For each class $\mathbf{K}$ of tree automata, we define $\mathrm{FTL}(\mathbf{K}) = \mathrm{FTL}(\mathcal{L}_\mathbf{K})$ and $\mathbf{FTL}(\mathbf{K}) = \mathbf{FTL}(\mathcal{L}_\mathbf{K})$.

Corollary 5.6 *Let $\mathcal{L}$ denote a class of regular tree languages. The following conditions are equivalent.*

1. $\mathbf{FTL}(\mathcal{L}) = \mathbf{FTL}(\mathbf{K}_\mathcal{L})$.
2. *There exists some class $\mathbf{K}$ of finite tree automata with $\mathbf{FTL}(\mathcal{L}) = \mathbf{FTL}(\mathbf{K})$.*
3. *There exists some class $\mathcal{L}'$ of regular tree languages closed with respect to quotients with $\mathbf{FTL}(\mathcal{L}) = \mathbf{FTL}(\mathcal{L}')$.*
4. *Each quotient of any language in $\mathcal{L}$ belongs to $\mathbf{FTL}(\mathcal{L})$.*
5. $\mathbf{FTL}(\mathcal{L})$ *is closed with respect to quotients.*
6. *For each L in $\mathcal{L}$, the modalities associated with the quotients of L are expressible in* $\mathrm{FTL}(\mathcal{L})$ *as in Theorem 5.4.*

Proof. The last three conditions are equivalent by Theorem 5.4. The first condition clearly implies the second and the second the third which in turn implies the fourth, since for a class $\mathbf{K}$ of tree automata, $\mathcal{L}_\mathbf{K}$ is closed under quotients. It remains to show that the fourth condition implies the first. But by

Lemma 3.1, when $\mathcal{L}$ consists of regular languages, every language recognizable by a tree automaton in $\mathbf{K}_{\mathcal{L}}$ is a boolean combination of quotients of some language in $\mathcal{L}$. It follows using Theorem 5.3 that $\mathbf{FTL}(\mathbf{K}_{\mathcal{L}}) \subseteq \mathbf{FTL}(\mathcal{L})$, while the reverse inclusion is obvious. $\square$

Example 5.7 Let L be any of the languages L_{X_i}, $i \in [\max(R)]$, L_{EF}, L_{EG}, L_{EU}, L_{AF}, L_{AG}, L_{AU}. Then each quotient of L is definable in $\mathrm{FTL}(\{L\})$. Thus, if $\mathcal{L}$ is any subcollection of these languages, then the equivalent conditions of Corollary 5.6 hold for $\mathcal{L}$.

6 A Variety Theorem

Several different concepts of varieties of regular tree languages with corresponding variety theorems have been proposed in the literature, cf. [1, 2, 21, 22, 9, 11]. The abundance of variety theorems is due to the fact that there exist several different reasonable notions of homomorphisms and quotients for trees, and the notion of syntactic algebra can be defined in several different frameworks: ordinary algebras, [1, 2], [21, 22], clones or Lawvere theories, [9], or preclones, [11]. Here we present yet another variety theorem that bears close connection to that given in [22].

In this section, all ranked alphabets are assumed to have a fixed common rank type R containing 0. Suppose that $\mathbb{A}$ and $\mathbb{B}$ are Σ-tree automata. Since tree automata are Σ-algebras, the direct product of $\mathbb{A}$ and $\mathbb{B}$ as an algebra is defined. However, the direct product may not be a tree automaton since it is not always generated by the constants. Therefore we define the *tree automaton direct product*, or *ta-direct product* of $\mathbb{A}$ and $\mathbb{B}$ as the smallest subalgebra of the usual direct product. The direct product of any finite number of tree automata is defined in the same way. We have already defined renamings of algebras. This notion gives rise to *tree automaton renamings*, or *ta-renamings*. Suppose that $\mathbb{A}$ is a Σ-tree automaton and the Δ-algebra $\mathbb{B}$ is a renaming of $\mathbb{A}$. (Δ is also of rank type R.) Then $\mathbb{B}$ has a least subalgebra which is called a ta-renaming of $\mathbb{A}$.

Recall that if $\mathbb{A}$ and $\mathbb{B}$ are Σ-tree automata and h is a homomorphism $\mathbb{A} \to \mathbb{B}$, then h is necessarily surjective. Thus we call $\mathbb{B}$ a *quotient* of $\mathbb{A}$.

For the purposes of this paper, we define a *(pseudo)variety of finite tree automata* to be any nonempty class of finite tree automata closed under the ta-direct product, ta-renaming and quotients. A *literal variety of tree languages* is any nonempty class of regular tree languages closed under the boolean operations, quotients and inverse literal tree homomorphisms. It is clear that both varieties of tree automata and literal varieties form (algebraic) lattices.

The relevance of literal varieties to the logics $\mathrm{FTL}(\mathcal{L})$ is justified by the following fact:

Corollary 6.1 *When $\mathcal{L}$ is a class of regular languages such that each quotient of any language in $\mathcal{L}$ belongs to* $\mathbf{FTL}(\mathcal{L})$ *, then* $\mathbf{FTL}(\mathcal{L})$ *is a literal variety. Thus, when* $\mathbf{K}$ *is a class of finite tree automata, then* $\mathbf{FTL}(\mathbf{K})$ *is a literal variety.*

Proof. By Theorem 5.1 and Corollary 5.6. The fact that when $\mathcal{L}$ consists of regular languages then $\mathbf{FTL}(\mathcal{L})$ is a class of regular languages will be established independently in Corollary 9.9. Alternatively, one can embed $\mathrm{FTL}(\mathcal{L})$ into monadic second-order logic and use one direction of the main result of [25] to the effect that every language definable in this logic is regular. $\square$

We now state and prove a Variety Theorem that provides a basis of the results of the paper.

Theorem 6.2 *The lattice of varieties of finite tree automata is isomorphic to the lattice of literal varieties of tree languages. An isomorphism is given by the assignment that maps each variety* $\mathbf{V}$ *of finite tree automata to the class* $\mathcal{L}_{\mathbf{V}}$ *of those tree languages recognizable by the members of* $\mathbf{V}$*.*

Proof. If $\mathbf{V}$ is a variety of finite tree automata then the class $\mathcal{L}_{\mathbf{V}}$ of regular tree languages is clearly nonempty and closed under the boolean operations (since $\mathbf{V}$ is closed under the direct product), quotients and inverse literal tree homomorphisms (since $\mathbf{V}$ is closed under renamings). We show that every literal variety $\mathcal{V}$ of tree languages corresponds to some variety of finite tree automata. Given $\mathcal{V}$, let $\mathbf{V}$ consist of those finite tree automata that only accept languages in $\mathcal{V}$. It is clear that $\mathbf{V}$ is nonempty. Since any language recognized by the ta-direct product of two finite tree automata is a boolean combination of languages recognized by the two tree automata, it follows that $\mathbf{V}$ is closed under the ta-direct product. Since $\mathcal{V}$ is closed under inverse literal homomorphisms, we have that $\mathbf{V}$ is closed under ta-renamings. Finally, since any language recognizable by a quotient of a tree automaton $\mathbb{A}$ is recognizable by $\mathbb{A}$, $\mathbf{V}$ is closed under quotients. Thus, $\mathbf{V}$ is a variety of finite tree automata. Let $\mathcal{W}$ denote the literal variety $\mathcal{L}_{\mathbf{V}}$ of all tree languages recognizable by the members of $\mathbf{V}$. We want to show that $\mathcal{V} = \mathcal{W}$. The inclusion $\mathcal{W} \subseteq \mathcal{V}$ is clear. Suppose now that $L \subseteq T_\Sigma$ is in $\mathcal{V}$ and consider the minimal tree automaton $\mathbb{A}_L$ of L. We know from Lemma 3.1 that every language recognizable by $\mathbb{A}_L$ is a boolean combination of quotients of L. It follows that every language recognizable by $\mathbb{A}_L$ is in $\mathcal{V}$, so that $\mathbb{A}_L \in \mathbf{V}$. Thus, since L is recognizable by $\mathbb{A}_L$, we have $L \in \mathcal{W}$. This proves that $\mathcal{V} \subseteq \mathcal{W}$.

Suppose that $\mathbf{V}$ is a variety of finite tree automata with corresponding literal variety $\mathcal{V}$. Let $\mathbf{W}$ denote the class of all finite tree automata that only accept languages in $\mathcal{V}$. By the above argument, we know that $\mathbf{W}$ is also a variety and is mapped to $\mathcal{V}$ under the correspondence given in the Theorem. It is clear that $\mathbf{V} \subseteq \mathbf{W}$. We want to show the reverse inclusion. So let $\mathbb{A}$ be a Σ-tree automaton in $\mathbf{W}$. For each $a \in A$, let $L_a \subseteq T_\Sigma$ denote the tree language accepted by $\mathbb{A}$ with unique final state a. Now each L_a is in $\mathcal{V}$ and thus recognizable by some tree automaton $\mathbb{B}_a$ in $\mathbf{V}$. Let $\mathbb{B}$ denote the direct product of the $\mathbb{B}_a$. Note that

$\mathbb{B} \in \mathbf{V}$. We claim that $\mathbb{A}$ is a quotient of $\mathbb{B}$. For each element $b \in B$ there is a tree $t \in T_\Sigma$ with $b = t_\mathbb{B} = (t_{\mathbb{B}_a})_{a\in A}$. We map b to $h(b) = t_\mathbb{A}$. This map is well-defined, for if $t_\mathbb{B} = s_\mathbb{B}$, for $t, s \in T_\Sigma$, then for each a, $t_{\mathbb{B}_a} = s_{\mathbb{B}_a}$, so that $t \in L_a$ iff $s \in L_a$. This means that $t_\mathbb{A} = s_\mathbb{A}$. Since it is clear that h is a homomorphism, $\mathbb{A}$ is a quotient of $\mathbb{B}$, proving that $\mathbb{A} \in \mathbf{V}$.

To complete the proof, assume now that $\mathbf{V}_1$ and $\mathbf{V}_2$ are varieties of finite tree automata with corresponding literal varieties $\mathcal{V}_1$ and $\mathcal{V}_2$. If $\mathbf{V}_1 \subseteq \mathbf{V}_2$, then clearly $\mathcal{V}_1 \subseteq \mathcal{V}_2$. Assume that $\mathcal{V}_1 \subseteq \mathcal{V}_2$. Then since $\mathbf{V}_i$ consists of all finite tree automata that only accept languages in $\mathcal{V}_i$, $i = 1, 2$, it follows that $\mathbf{V}_1 \subseteq \mathbf{V}_2$. □

Remark 6.3 Suppose that $\mathbf{V}$ is a variety of finite tree automata and $\mathcal{V}$ is the corresponding literal variety. Then a tree language belongs to $\mathcal{V}$ iff its minimal tree automaton is in $\mathbf{V}$. Moreover, $\mathbf{V}$ is the least variety of finite automata containing the minimal automata of the languages in $\mathcal{V}$, and a tree automaton $\mathbb{A}$ is in $\mathbf{V}$ iff every language recognizable by $\mathbb{A}$ is in $\mathcal{V}$.

Example 6.4 The least literal variety of regular languages contains for each ranked alphabet Σ (of rank type R) just the languages $\emptyset$ and T_Σ. The corresponding variety of finite tree automata is the class of all *trivial*, i.e., singleton tree automata. The greatest literal variety is the class of all regular languages. The corresponding variety of finite tree automata is the class of all finite tree automata.

Example 6.5 For a nonnegative integer k, a tree language $L \subseteq T_\Sigma$ is called *k-definite* if the membership of a tree $t \in T_\Sigma$ in L only depends on the cut-off of t at depth k, i.e., on that part of the tree determined by the vertices of depth *strictly less* than k. By extension, a tree language is *definite* if it is k-definite for some k. For each k, let $\mathcal{D}_k$ denote the class of k-definite tree languages, and let $\mathcal{D}$ denote the class of definite tree languages, so that $\mathcal{D} = \bigcup_{k\geq 0} \mathcal{D}_k$. Note that for every ranked alphabet Σ, the only languages over Σ contained in $\mathcal{D}_0$ are $\emptyset$ and T_Σ. Any 1-definite language over Σ is a union of languages of the form $T_\sigma = \{\sigma(t_1, \ldots, t_n) : t_1, \ldots, t_n \in T_\Sigma\}$, where $\sigma \in \Sigma_n$, $n \geq 0$. In general, any k-definite tree language is a union of languages of the form $T_t = \{t(t_1, \ldots, t_n) : t_1, \ldots, t_n \in T_\Sigma\}$, where $t \in T_\Sigma(X_n), n \geq 0$ is of depth $< k$, moreover, each x_i occurs at most (or exactly) once in t and each leaf labeled in X_n is of depth k.

Definite tree languages were introduced in [15] and subsequently studied in [18] and [8]. It is shown in these papers (though stated in different form) that $\mathcal{D}$ and each $\mathcal{D}_k$ is a literal variety of tree languages. The variety of finite tree automata corresponding to $\mathcal{D}_k$ can be described as follows. Call a Σ-algebra *k-definite* if it satisfies all equations (in the sense of Universal Algebra, cf. [14]) $t = s$ such that the trees $t, s \in T_\Sigma(X_n)$, $n \geq 0$ agree up to depth k, i.e., s and t have equal cut-offs at depth k. (Actually, it suffices to require this condition for trees of depth $\leq k$.) A *definite algebra* is an algebra which is k-definite for some k. A *definite tree automaton* (k*-definite tree automaton*) is a tree automaton which is a definite algebra (k-definite algebra, resp.). We let $\mathbf{D}$ ($\mathbf{D}_k$, resp.) denote the class of all finite definite (k-definite, resp.) tree automata. For each k, $\mathbf{D}_k$ is the variety of tree automata corresponding to $\mathcal{D}_k$, moreover, $\mathbf{D}$ is the variety corresponding to $\mathcal{D}$. See also [9].

It is clear that there exists an algorithm to decide whether a finite algebra is k-definite. It follows that each $\mathcal{D}_k$ is decidable: Given a regular tree language L (by a finite tree automaton equipped with a set of final states), there is an effective procedure to test whether or not L is k-definite. In [15], it is shown that $\mathcal{D}$ is also decidable, see also [18] and [8].

7 The Cascade Product

Let R be a rank type kept fixed in this section. All ranked sets will be assumed to be of rank type R.

Let $\mathbb{A}$ be a Σ-algebra, $\mathbb{B}$ a Δ-algebra, and α a family of functions $\alpha_n : A^n \times \Sigma_n \to \Delta_n$, $n \in R$. The *cascade product* $\mathbb{A} \times_\alpha \mathbb{B}$ determined by α is the Σ-algebra with carrier $A \times B$ and operations

$$\sigma((a_1, b_1), \ldots, (a_n, b_n)) \quad = \quad (\sigma(a_1, \ldots, a_n), \delta(b_1, \ldots, b_n)),$$

where $\delta = \alpha_n(a_1, \ldots, a_n, \sigma)$, for all $((a_1, b_1), \ldots, (a_n, b_n)) \in A \times B$, $\sigma \in \Sigma_n$, $n \in R$. When $0 \in R$ and $\mathbb{A}$ and $\mathbb{B}$ are tree automata, the *ta-cascade product* of $\mathbb{A}$ and $\mathbb{B}$ determined by α is the least subalgebra of the above cascade product. We use the same notation $\mathbb{A} \times_\alpha \mathbb{B}$ to denote a ta-cascade product of tree automata $\mathbb{A}$ and $\mathbb{B}$. The direct product is clearly a special case of the cascade product. Below we will sometimes write just cascade product for the ta-cascade product, direct product for the ta-direct product, etc. Note that if $\mathbb{C}$ is a cascade product or ta-cascade product of $\mathbb{A}$ and $\mathbb{B}$, then the projection $C \to A$ defined by $(a, b) \mapsto a$ for all $(a, b) \in C$ is a surjective homomorphism.

The cascade product of algebras (or tree automata) can be extended to several factors: $\mathbb{A}_1 \times_{\alpha_1} \mathbb{A}_2 \times_{\alpha_2} \ldots \times_{\alpha_{n-1}} \mathbb{A}_n$. Here, when $\mathbb{A}_i$ is a Σ_i-algebra, for each $i \in [n]$, then α_i is a family of functions

$$(A_1 \times \ldots \times A_{i-1})^m \times (\Sigma_1)_m \quad \to \quad (\Sigma_i)_m, \quad m \in R.$$

Note that $\mathbb{A}_1 \times_{\alpha_1} \mathbb{A}_2 \times_{\alpha_2} \ldots \times_{\alpha_{n-1}} \mathbb{A}_n$ is a Σ_1-algebra (or a Σ_1-tree automaton).

Suppose that $\mathbb{A}$ is a Σ-algebra and α is a family of functions $A^n \times \Sigma_n \to \Delta_n$. Then we call the pair $(\mathbb{A}, \alpha)$ a *tree transducer*. For each $n \geq 0$, the tree transducer $(\mathbb{A}, \alpha)$ induces a mapping $f : T_\Sigma \to T_\Delta$, called the *relabeling induced by* $(\mathbb{A}, \alpha)$. Given a tree $t \in T_\Sigma(X_n)$, $f(t)$ is defined as follows. When $t = \sigma \in \Sigma_0$, then $f(t) = \alpha_0(\sigma)$. Suppose now that $t = \sigma(t_1, \ldots, t_m)$, where

$m > 0$, $\sigma \in \Sigma_m$ and $t_1, \ldots, t_m \in T_\Sigma$. Then $f(t) = \delta(f(t_1), \ldots, f(t_m))$, where $\delta = \alpha_m((t_1)_\mathbb{A}, \ldots, (t_m)_\mathbb{A}, \sigma)$. More generally, when $t \in T_\Sigma(X_n)$ and $a_1, \ldots, a_n \in A$, $n \geq 0$, we define $f_{(a_1,\ldots,a_n)}(t)$ as follows: When $t = x_i$ with $i \in [n]$, then $f_{(a_1,\ldots,a_n)}(t) = x_i$, and when $t = \sigma \in \Sigma_0$, then $f_{(a_1,\ldots,a_n)}(t) = \alpha_0(\sigma)$. Finally, when $t = \sigma(t_1, \ldots, t_m)$, where $m > 0$, $\sigma \in \Sigma_m$ and $t_1, \ldots, t_m \in T_\Sigma(X_n)$, then $f_{(a_1,\ldots,a_n)}(t) = \delta(f_{(a_1,\ldots,a_n)}(t_1), \ldots, f_{(a_1,\ldots,a_n)}(t_m))$, where

$$\delta = \alpha_m((t_1)_\mathbb{A}(a_1, \ldots, a_n), \ldots, (t_m)_\mathbb{A}(a_1, \ldots, a_n), \sigma).$$

Below we will write $\alpha(t)$ for $f(t)$ and $\alpha_{(a_1,\ldots,a_n)}(t)$ for $f_{(a_1,\ldots,a_n)}(t)$.

Proposition 7.1 *Suppose that $\mathbb{C} = \mathbb{A} \times_\alpha \mathbb{B}$ is a cascade product of the Σ-algebra $\mathbb{A}$ and the Δ-algebra $\mathbb{B}$. Then for any tree $t \in T_\Sigma$, $t_\mathbb{C} = (t_\mathbb{A}, s_\mathbb{B})$, where $s = \alpha(t)$ is the image of t with respect to the relabeling induced by $(\mathbb{A}, \alpha)$. More generally, for every $t \in T_\Sigma(X_n)$ and $(a_i, b_i) \in A \times B$, $i \in [n]$, $t_\mathbb{C}((a_1, b_1), \ldots, (a_n, b_n)) = (t_\mathbb{A}(a_1, \ldots, a_n), s_\mathbb{B}(b_1, \ldots, b_n))$, where $s = \alpha_{(a_1,\ldots,a_n)}(t)$. A similar fact holds for the ta-cascade product.*

Proof. By a straightforward induction on the structure of t. □

By a *closed variety of finite algebras* we mean a nonempty class of finite algebras (of the same rank type R) closed with respect to the cascade product, subalgebras, and homomorphic images. Similarly, a *closed variety of finite tree automata* is any nonempty class of finite tree automata closed with respect to the ta-cascade product and quotients. Note that any closed variety of finite algebras is closed under renamings, and any closed variety of finite tree automata is closed under ta-renamings. Thus any closed variety of finite tree automata is a variety. For any class $\mathbf{K}$ of finite tree automata, we let $\widehat{\mathbf{K}}$ denote the least closed variety of finite tree automata containing $\mathbf{K}$. Moreover, when $\mathbf{V}$ and $\mathbf{W}$ are closed varieties of finite tree automata, then we define $\mathbf{V} \vee \mathbf{W}$ as the least closed variety of finite tree automata containing both $\mathbf{V}$ and $\mathbf{W}$, i.e., $\mathbf{V} \vee \mathbf{W} = \widehat{\mathbf{V} \cup \mathbf{W}}$.

Remark 7.2 Suppose that $\mathbf{K}$ is a nonempty class of finite algebras. It is known (cf., e.g., [8]) that the least closed variety containing $\mathbf{K}$ consists of all homomorphic images of subalgebras of cascade products $\mathbb{A}_1 \times_{\alpha_1} \mathbb{A}_2 \times_{\alpha_2} \ldots \times_{\alpha_{n-1}} \mathbb{A}_n$, where each $\mathbb{A}_i$ is in $\mathbf{K}$. A similar fact holds for finite tree automata: The least closed variety of finite tree automata containing a class $\mathbf{K}$ of finite tree automata consists of all quotients of ta-cascade products $\mathbb{A}_1 \times_{\alpha_1} \mathbb{A}_2 \times_{\alpha_2} \ldots \times_{\alpha_{n-1}} \mathbb{A}_n$ with $\mathbb{A}_i \in \mathbf{K}$, for each $i \in [n]$.

An example of a closed variety of finite algebras is the class of all finite definite algebras. Let $\mathbb{D}_0(R)$, or just $\mathbb{D}_0$ denote the Bool-algebra (i.e., Σ-algebra with $\Sigma = \text{Bool}$) with carrier $\{0, 1\}$ and constant valued operations

$$\begin{aligned} \downarrow_n (a_1, \ldots, a_n) &= 0 \\ \uparrow_n (a_1, \ldots, a_n) &= 1, \quad n \in R. \end{aligned}$$

Note that when $0 \in R$, then $\mathbb{D}_0$ is a tree automaton. The following result was proved in [8].

Theorem 7.3 *The class of all finite definite algebras is the least closed variety containing the algebra $\mathbb{D}_0$.*

It follows that when $0 \in R$, then the class **D** of finite definite tree automata is a closed variety of finite tree automata and is generated by $\mathbb{D}_0$.

8 Definite Tree Languages, Revisited

In this section, all ranked alphabets are assumed to have a fixed rank type R with $0 \in R$.

We say that *the* next *modalities are expressible in the logic* FTL($\mathcal{L}$) if for all alphabets Σ and integers i with $i \in [\max(R)]$, and for every formula φ in FTL($\mathcal{L}$) over Σ, there exists a formula $\mathrm{X}_i\varphi$ in FTL($\mathcal{L}$) such that for all trees $t \in T_\Sigma$, $t \models \mathrm{X}_i\varphi$ iff t is of the form $\sigma(t_1, \ldots, t_n)$, where $n \geq i$, and $t_i \models \varphi$. More generally, there is a canonical way to assign a word w in $[\max(R)]^*$ to every vertex of a tree $t \in T_\Sigma$, the "address" of v (see, e.g., [18]). Given a word $w \in [\max(R)]^*$, we say that the modality X_w is expressible in FTL($\mathcal{L}$) if for every formula φ in FTL($\mathcal{L}$) over any alphabet Σ there exists a formula $\mathrm{X}_w\varphi$ in FTL($\mathcal{L}$) such that for all trees $t \in T_\Sigma$, $t \models \mathrm{X}_w\varphi$ iff t has a vertex v at the address w and the subtree t_v rooted at this vertex satisfies φ. It is clear that for all words w, w' and for all formulas φ, $\mathrm{X}_w\mathrm{X}_{w'}\varphi$ is equivalent to the formula $\mathrm{X}_{ww'}\varphi$.

The following fact is clear.

Proposition 8.1 *The following conditions are equivalent for a logic* FTL($\mathcal{L}$)*:*

1. *The next modalities are expressible in* FTL($\mathcal{L}$)*.*
2. *For every w, the modality X_w is expressible in* FTL($\mathcal{L}$)*.*
3. *For each $i \in [\max(R)]$,* **FTL**($\mathcal{L}$) *contains the language L_{X_i}.*

Proof. Since for each i, the formula $\mathrm{X}_i(\bigvee_{n\in R} p_{\uparrow_n})$ defines L_{X_i} over the ranked alphabet Bool, the first condition implies the third. Moreover, since $\mathrm{X}_i\varphi$ is expressible as $L_{\mathrm{X}_i}(\delta \mapsto \psi_\delta)_{\delta\in\mathrm{Bool}}$, where $\psi_{\uparrow_n} = \varphi$ and $\psi_{\downarrow_n} = \neg\varphi$ for all $n \in R$, the third condition implies the first. $\square$

The languages L_{X_i} were defined in Example 4.4. Let $\mathcal{L}_\mathrm{X} = \{L_{\mathrm{X}_i} : i \in [\max(R)]\}$. Below we denote the logic FTL($\mathcal{L}_\mathrm{X}$) by CTL(X), and the tree language class **FTL**($\mathcal{L}_\mathrm{X}$) by **CTL**(X).

Proposition 8.2 **CTL**(X) $= \mathcal{D}$*.*

Proof. We know that each definite language over Σ is a finite union of languages of the form T_t defined in Example 6.5. The property that a tree belongs to T_t is clearly expressible using the X_ws. For the reverse inclusion, one argues by induction on the structure of the formula φ over Σ in CTL(X) to show that $L_\varphi \in \mathcal{D}$. The base of the induction is clear. In the induction step, the case of boolean connectives is covered by the fact that $\mathcal{D}$ is a literal variety and is thus closed under the boolean operations. Finally, one proves that if $(\varphi_\delta)_{\delta\in\mathrm{Bool}}$ define definite languages, then so does the formula $\varphi = L_{\mathrm{X}_i}(\delta \mapsto \varphi_\delta)_{\delta\in\mathrm{Bool}}$, for each i. Indeed, in this case L_φ is the collection of all trees whose root is labeled by a symbol of rank $\geq i$ such that the ith immediate subtree satisfies $\varphi_{\uparrow_n}$, where the root of this subtree is labeled in Σ_n. Now if $L_{\varphi_{\uparrow_n}}$ is k-definite, then L_φ is $(k+1)$-definite. $\square$

Corollary 8.3 *The following conditions are equivalent for a logic* $\mathrm{FTL}(\mathcal{L})$*:*

1. *The next modalities are expressible in* $\mathrm{FTL}(\mathcal{L})$.
2. *For every w, the modality X_w is expressible in* $\mathrm{FTL}(\mathcal{L})$.
3. $\mathcal{L}_\mathrm{X} \subseteq \mathbf{FTL}(\mathcal{L})$.
4. $\mathcal{D}_2 \subseteq \mathbf{FTL}(\mathcal{L})$.
5. $\mathcal{D} \subseteq \mathbf{FTL}(\mathcal{L})$.

Corollary 8.4 $\mathbf{FTL}(\mathcal{L}) = \mathcal{D}$ *iff* $\mathcal{L} \subseteq \mathcal{D}$ *and* $\mathcal{L}_\mathrm{X} \subseteq \mathbf{FTL}(\mathcal{L})$.

9 Main Results

In this section, all ranked sets are assumed to have a fixed rank type R with $0 \in R$. In the next two propositions, let $\mathcal{L}$ denote a class of tree languages.

Proposition 9.1 *Suppose that $\mathbb{A}$ and $\mathbb{B}$ are finite tree automata and $\mathbb{C} = \mathbb{A} \times_\alpha \mathbb{B}$ is a ta-cascade product of $\mathbb{A}$ and $\mathbb{B}$. If every language recognizable by $\mathbb{A}$ or $\mathbb{B}$ belongs to $\mathbf{FTL}(\mathcal{L})$, and if the next modality is expressible in $\mathrm{FTL}(\mathcal{L})$, then every language recognizable by $\mathbb{C}$ also belongs to $\mathbf{FTL}(\mathcal{L})$.*

Proof. Suppose that $\mathbb{A}$ is a Σ-tree automaton and $\mathbb{B}$ is a Δ-tree automaton, so that $\mathbb{C}$ is a Σ-tree automaton. Let h denote the unique homomorphism $T_\Sigma \to \mathbb{C}$. It suffices to show that for each $(a,b) \in C$, the language $h^{-1}((a,b))$ belongs to $\mathbf{FTL}(\mathcal{L})$.

For every $t \in T_\Sigma$, $t_\mathbb{C} = (t_\mathbb{A}, s_\mathbb{B})$, where $s = \alpha(t)$ is the image of t under the relabeling induced by the tree transducer $(\mathbb{A}, \alpha)$. (See Proposition 7.1.) Thus,

$$h^{-1}((a,b)) \quad = \quad \{t : t_\mathbb{A} = a \wedge s_\mathbb{B} = b\}.$$

By assumption, and since $\mathbf{FTL}(\mathcal{L})$ is closed under inverse literal homomorphisms, for each $a \in A$ there exists a formula τ_a over Σ in $\mathrm{FTL}(\mathcal{L})$ defining the set of trees $h^{-1}(\pi^{-1}(a))$, where π denotes the projection $C \to A$, $(a, b) \mapsto a$. For each $b \in B$, let $T_b = \{s \in T_\Delta : s_{\mathbb{B}} = b\}$. We construct a (deterministic) family of formulas $(\varphi_\delta)_{\delta\in\Delta}$ over Σ such that for each tree $t \in T_\Sigma$, the characteristic tree determined by t and this family is exactly $\alpha(t)$. For each $\delta \in \Delta_n$, we define:

$$\varphi_\delta \quad = \bigvee_{\alpha_n(a_1,\ldots,a_n,\sigma)=\delta} p_\sigma \wedge X_1\tau_{a_1} \wedge \ldots \wedge X_n\tau_{a_n}.$$

Given $(a, b) \in C$, let

$$\varphi \quad = \quad \tau_a \wedge T_b(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}.$$

Then we have

$$\begin{aligned} L_\varphi &= \{t \in T_\Sigma : t_{\mathbb{A}} = a \wedge \alpha(t) \in T_b\} \\ &= \{t \in T_\Sigma : t_{\mathbb{C}} = (a, b)\}. \end{aligned}$$

Since by assumption T_b is definable in $\mathrm{FTL}(\mathcal{L})$ and the next modalities are expressible, it follows from Theorem 5.3 that there is a formula in $\mathrm{FTL}(\mathcal{L})$ which is equivalent to φ. □

Proposition 9.2 *Suppose that $\varphi = K(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ is a formula over Σ in* $\mathrm{FTL}(\mathcal{L})$*, where $(\varphi_\delta)_{\delta\in\Delta}$ is a deterministic family. Suppose that K is recognizable by $\mathbb{B}$ and that each L_{φ_δ} is recognizable by $\mathbb{A}$, where $\mathbb{A}$ and $\mathbb{B}$ are possibly infinite tree automata. Then $L_\varphi \subseteq T_\Sigma$ is recognizable by a ta-cascade product of $\mathbb{A}$ and $\mathbb{B}$.*

Proof. Let h denote the unique homomorphism $T_\Sigma \to \mathbb{A}$ and h_K the unique homomorphism $T_\Delta \to \mathbb{B}$. For each $\delta \in \Delta$, let F_δ denote the set $h(L_{\varphi_\delta})$. Since $(\varphi_\delta)_{\delta\in\Delta}$ is a deterministic family, the sets F_δ are pairwise disjoint. For each $\sigma \in \Sigma_n$ and $a_1, \ldots, a_n \in A$, $n \in R$, define $\alpha_n(a_1, \ldots, a_n, \sigma) = \delta \in \Delta_n$ iff $\sigma_A(a_1, \ldots, a_n) \in F_\delta$. By the above remark, there is at most one such δ. To see that there is at least one, take $t_i \in T_\Sigma$ with $(t_i)_{\mathbb{A}} = a_i$, $i \in [n]$. Then let $t = \sigma(t_1, \ldots, t_n)$. There exists some $\delta \in \Delta_n$ with $t \models \varphi_\delta$. Therefore $\sigma_{\mathbb{A}}(a_1, \ldots, a_n) = t_{\mathbb{A}} \in F_\delta$. Now it follows that for every $t \in T_\Sigma$, the characteristic tree determined by t and the family $(\varphi_\delta)_{\delta\in\Delta}$ is exactly $\alpha(t)$, the image of t under the relabeling induced by $(\mathbb{A}, \alpha)$. It follows that L_φ is recognized by $\mathbb{C}$ with set of final states $\{(a, b) : b \in h_K(K)\}$. □

Theorem 9.3 *For any class $\mathbf{K}$ of finite tree automata, every language in the class $\mathbf{FTL}(\mathbf{K})$ is recognizable by some tree automaton in $\widehat{\mathbf{K}} \vee \mathbf{D}$.*

Proof. Let φ denote a deterministic formula over Σ in $\mathrm{FTL}(\mathbf{K})$. We show that L_φ is recognizable by some automaton in $\widehat{\mathbf{K}} \vee \mathbf{D}$. When φ is p_σ, for some $\sigma \in \Sigma$, then L_φ is 1-definite and thus recognizable by some automaton in $\mathbf{D}_1 \subseteq \mathbf{D}$. We proceed by induction on the structure of φ. Assume that $\varphi = \varphi_1 \vee \varphi_2$ such that L_{φ_i} is recognizable by $\mathbb{A}_i$ in $\widehat{\mathbf{K}} \vee \mathbf{D}$, $i = 1, 2$. Then L_φ is recognizable by the ta-direct product $\mathbb{A}_1 \times \mathbb{A}_2$ which is also in $\widehat{\mathbf{K}} \vee \mathbf{D}$. When $\varphi = \neg\varphi_1$, where L_{φ_1} is recognizable by $\mathbb{A}_1$ above, then L_φ is also recognizable by $\mathbb{A}_1$. Finally, when $\varphi = L(\delta \mapsto \varphi_\delta)_{\delta\in\Delta}$ and each L_{φ_δ} is recognizable by some tree automaton in $\widehat{\mathbf{K}} \vee \mathbf{D}$, then it follows by Proposition 9.2 that L_φ is recognizable by some tree automaton in $\widehat{\mathbf{K}} \vee \mathbf{D}$. (Note that since $\widehat{\mathbf{K}} \vee \mathbf{D}$ is closed with respect to the direct product, we may assume without loss of generality that each L_{φ_δ} is recognizable by the same tree automaton $\mathbb{A}$ in $\widehat{\mathbf{K}} \vee \mathbf{D}$). □

Remark 9.4 In the above proof, we did not use the assumption that $\mathbf{K}$ consists of finite automata. Our argument gives that for any class $\mathbf{K}$ of possibly infinite tree automata, every language in $\mathbf{FTL}(\mathbf{K})$ is recognizable by some tree automaton in the least class of tree automata containing $\mathbf{K}$ and $\mathbf{D}$, closed with respect to the ta-cascade product.

Theorem 9.5 *Suppose that the next modalities are expressible in* FTL($\mathbf{K}$), *where* $\mathbf{K}$ *is a class of finite tree automata. Then a language* L *belongs to* $\mathbf{FTL}(\mathbf{K})$ *iff its minimal tree automaton* $\mathbb{A}_L$ *belongs to* $\widehat{\mathbf{K}} \vee \mathbf{D}$ *iff* L *is recognizable by an automaton in* $\widehat{\mathbf{K}} \vee \mathbf{D}$.

Proof. We know from Corollary 8.3 that $\mathbf{FTL}(\mathbf{K})$ contains the definite tree languages and thus $\mathbf{FTL}(\mathbf{K}) = \mathbf{FTL}(\mathbf{K} \cup \mathbf{D})$, by Theorem 5.3. Let us define the *rank* of $\mathbb{A} \in \widehat{\mathbf{K}} \vee \mathbf{D}$ to be the smallest number of ta-cascade product and quotient operations needed to generate $\mathbb{A}$ from $\mathbf{K} \cup \mathbf{D}$. We prove by induction on the rank of $\mathbb{A}$ that every language recognizable by $\mathbb{A}$ is in $\mathbf{FTL}(\mathbf{K} \cup \mathbf{D})$. When the rank is 0 we have $\mathbb{A} \in \mathbf{K} \cup \mathbf{D}$ and the result is immediate. When the rank of $\mathbb{A}$ is positive, then $\mathbb{A}$ is either a quotient of a tree automaton $\mathbb{B}$ in $\widehat{\mathbf{K}} \vee \mathbf{D}$ of smaller rank, or $\mathbb{A}$ is a ta-cascade product of some tree automata in $\widehat{\mathbf{K}} \vee \mathbf{D}$ of smaller rank. In the first case, every language recognizable by $\mathbb{A}$ is recognizable by $\mathbb{B}$. In the second case, the result follows from Proposition 9.1. Conversely, by Theorem 9.3, every language in $\mathbf{FTL}(\mathbf{K})$ is recognizable by some tree automaton in $\widehat{\mathbf{K}} \vee \mathbf{D}$. □

By combining Theorem 7.3 with the above result, we have:

Corollary 9.6 *Suppose that the next modalities are expressible in* FTL($\mathbf{K}$), *where* $\mathbf{K}$ *is a class of finite tree automata. Then a language* L *belongs to* $\mathbf{FTL}(\mathbf{K})$ *iff its minimal tree automaton* $\mathbb{A}_L$ *belongs to* $\widehat{\mathbf{K} \cup \{\mathbb{D}_0\}}$ *iff* L *is recognizable by an automaton in* $\widehat{\mathbf{K} \cup \{\mathbb{D}_0\}}$.

Example 9.7 The assumption in the above result that the next modalities be expressible in FTL($\mathbf{K}$) is important. Indeed, let $\mathbf{K} = \emptyset$. Then $\mathbf{FTL}(\mathbf{K})$ is the class $\mathcal{D}_1$ of all 1-definite tree languages, while $\widehat{\mathbf{K}} \vee \mathbf{D} = \mathbf{D}$ which corresponds to the literal variety $\mathcal{D}$ of all definite tree languages that properly contains $\mathcal{D}_1$.

Corollary 9.8 *Suppose that $\mathcal{L}$ is a class of regular languages such that each quotient of any language in $\mathcal{L}$ belongs to $\mathbf{FTL}(\mathcal{L})$ and the next modalities are expressible in* FTL($\mathcal{L}$). *Then a language L belongs to $\mathbf{FTL}(\mathbf{K})$ iff its minimal tree automaton $\mathbb{A}_L$ belongs to $\widehat{\mathbf{K}_{\mathcal{L}} \cup \{\mathbb{D}_0\}}$ iff L is recognizable by some automaton in $\widehat{\mathbf{K}_{\mathcal{L}} \cup \{\mathbb{D}_0\}}$.*

Corollary 9.9 *For each class $\mathcal{L}$ of regular tree languages,* $\mathbf{FTL}(\mathcal{L})$ *consists of regular languages.*

Call a nonempty class of regular tree languages $\mathcal{L}$ *closed* if $\mathbf{FTL}(\mathcal{L}) \subseteq \mathcal{L}$ and if $\mathcal{L}$ is closed with respect to quotients. By Theorems 5.1 and 5.4, every closed class is a literal variety of tree languages. Moreover, by Corollary 5.6, $\mathcal{L}$ is closed iff $\mathcal{L} = \mathbf{FTL}(\mathcal{L}')$ for a class $\mathcal{L}'$ of regular tree languages closed with respect to quotients iff $\mathcal{L} = \mathbf{FTL}(\mathbf{K})$ for a class $\mathbf{K}$ of finite tree automata.

Recall that by Theorem 6.2, the assignment

$$\begin{aligned} \mathbf{V} \quad \mapsto \quad \mathcal{L}_{\mathbf{V}} &= \{L : L \textit{ is recognizable by some } \mathbb{A} \in \mathbf{V}\} \\ &= \{L : \mathbb{A}_L \in \mathbf{V}\} \end{aligned}$$

defines an order isomorphism between varieties $\mathbf{V}$ of finite tree automata and literal varieties $\mathcal{V}$ of tree languages. The inverse assignment maps a literal variety $\mathcal{V}$ to the class of those finite tree automata $\mathbb{A}$ such that every language recognizable by $\mathbb{A}$ belongs to $\mathcal{V}$.

Theorem 9.10 *If $\mathbf{V}$ is a closed variety of finite tree automata containing $\mathbf{D}$, then $\mathcal{L}_{\mathbf{V}} = \mathbf{FTL}(\mathbf{V})$. Moreover, the assignment $\mathbf{V} \mapsto \mathbf{FTL}(\mathbf{V})$ defines an order isomorphism between closed varieties $\mathbf{V}$ of finite tree automata containing $\mathbf{D}$ and closed classes $\mathcal{L}$ of regular tree languages containing $\mathcal{D}$.*

Proof. If $\mathbf{V}$ is a closed variety containing $\mathbf{D}$, then by Theorem 9.5 and Corollary 8.3, $\mathcal{L}_{\mathbf{V}} = \mathbf{FTL}(\mathbf{V})$. Moreover, $\mathbf{FTL}(\mathbf{V})$ contains $\mathcal{D}$. By the Variety Theorem, we have $\mathbf{V}_1 \subseteq \mathbf{V}_2$ iff $\mathcal{L}_{\mathbf{V}_1} \subseteq \mathcal{L}_{\mathbf{V}_2}$. Finally, the map is surjective, for if $\mathcal{L}$ is a closed class of regular tree languages containing the definite tree languages, then $\mathcal{L} = \mathcal{L}_{\mathbf{V}}$ for some variety $\mathbf{V}$ of finite tree automata containing $\mathbf{D}$. By Proposition 9.1, $\mathbf{V}$ is closed with respect to the ta-cascade product, and $\mathcal{L} = \mathbf{FTL}(\mathbf{V})$. □

10 Some Applications

Recall that $\mathcal{L}_{\mathrm{X}}$ denotes the set of languages $\{L_{\mathrm{X}_i},\ i \in [\max(R)]\}$. Recall the definitions of the languages L_{EF}, L_{EG}, L_{EU}. The minimal tree automata of the latter three languages can be described as follows. The minimal automaton of L_{EF} is $\mathbb{E}_F(R)$, which has two elements, $0, 1$, and operations

$$\begin{aligned}\uparrow_n (b_1, \ldots, b_n) &= 1\\ \downarrow_n (b_1, \ldots, b_n) &= b_1 \vee \ldots \vee b_n,\end{aligned}$$

for all $b_1, \ldots, b_n \in \{0, 1\}$, $n \in R$. The minimal automaton $\mathbb{E}_G(R)$ of L_{EG} also has two elements, $0, 1$. The operations are:

$$\begin{aligned}\uparrow_n (b_1, \ldots, b_n) &= \begin{cases} 1 & \text{if } n = 0\\ b_1 \vee \ldots \vee b_n & \text{if } n > 0\end{cases}\\ \downarrow_n (b_1, \ldots, b_n) &= 0,\end{aligned}$$

for all $b_1, \ldots, b_n \in \{0, 1\}$, $n \in R$. Finally, the minimal automaton $\mathbb{E}_U(R)$ of L_{EU} is defined on the set $\{0, 1\}$ by

$$\begin{aligned}\uparrow_n (b_1, \ldots, b_n) &= 1\\ \vee_n(b_1, \ldots, b_n) &= b_1 \vee \ldots \vee b_n\\ \downarrow_n (b_1, \ldots, b_n) &= 0,\end{aligned}$$

for all $b_1, \ldots, b_n \in \{0, 1\}$, $n \in R$. Below we will just write $\mathbb{E}_F, \mathbb{E}_G, \mathbb{E}_U$ for the automata $\mathbb{E}_F(R), \mathbb{E}_G(R), \mathbb{E}_U(R)$ whenever R is understood. We define

$$\begin{aligned}\mathbf{CTL}(\mathrm{X}, \mathrm{EF}) &= \mathbf{FTL}(\mathcal{L}_X \cup \{L_{\mathrm{EF}}\})\\ \mathbf{CTL}(\mathrm{X}, \mathrm{EG}) &= \mathbf{FTL}(\mathcal{L}_X \cup \{L_{\mathrm{EG}}\})\\ \mathbf{CTL}(\mathrm{X}, \mathrm{EF}, \mathrm{EG}) &= \mathbf{FTL}(\mathcal{L}_X \cup \{L_{\mathrm{EF}}, L_{\mathrm{EG}}\})\\ \mathbf{CTL} &= \mathbf{FTL}(\mathcal{L}_X \cup \{L_{\mathrm{EU}}\}).\end{aligned}$$

By Example 5.7 and Corollary 8.3, all assumptions of Corollary 9.8 apply to the sets of languages used in the above definitions. Note that the minimal automata of L_{EF} and L_{EG} renamings of some reducts of the minimal automaton of L_{EU}. Thus, $\mathbf{CTL} = \mathbf{FTL}(\mathcal{L}_X \cup \{L_{\mathrm{EF}}, L_{\mathrm{EG}}, L_{\mathrm{EU}}\})$.

Theorem 10.1 *1. For $Y \in \{F, G\}$, a tree language belongs to* $\mathbf{CTL}(\mathrm{X}, \mathrm{EY})$ *iff its minimal tree automaton is in* $\widehat{\{\mathbb{E}_Y, \mathbb{D}_0\}}$.

2. A tree language belongs to $\mathbf{CTL}(\mathrm{X}, \mathrm{EF}, \mathrm{EG})$ *iff its minimal tree automaton is in* $\widehat{\{\mathbb{E}_F, \mathbb{E}_G, \mathbb{D}_0\}}$.

3. A tree language belongs to $\mathbf{CTL}$ *iff its minimal automaton belongs to* $\widehat{\{\mathbb{E}_U\}}$.

Proof. The first two statements follow from Corollary 9.6. The third statement follows from Corollary 9.6, Theorem 7.3, and the fact that $\mathbb{D}_0$ is isomorphic to the reduct of $\mathbb{E}_U$ obtained by forgetting about the $\vee$-operation. □

Remark 10.2 The logic defining the class $\mathbf{CTL}(\mathrm{X}, \mathrm{EF}, \mathrm{EG})$ is closely related to the logic introduced in [4].

Remark 10.3 Up to renaming of the operations, $\mathbb{E}_G$ is isomorphic to the algebra on the set $\{0,1\}$ with operations

$$\begin{array}{rcl} \uparrow_n (b_1,\ldots,b_n) & = & 1 \\ \downarrow_n (b_1,\ldots,b_n) & = & b_1 \wedge \ldots \wedge b_n, \end{array}$$

for all $n \in R$.

The minimal automata of L_{AG}, L_{AF} and L_{AU} are respectively isomorphic to the minimal automata of L_{EF}, L_{EG} and L_{EU}. Thus, $\mathbf{CTL}(\mathrm{X},\mathrm{EF}) = \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{AG}}\})$, $\mathbf{CTL}(\mathrm{X},\mathrm{EG}) = \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{AF}}\})$, $\mathbf{CTL}(\mathrm{X},\mathrm{EF},\mathrm{EG}) = \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{AG}}, L_{\mathrm{AF}}\})$. Moreover, $\mathbf{CTL} = \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{EF}}, L_{\mathrm{EG}}, L_{\mathrm{EU}}, L_{\mathrm{AF}}, L_{\mathrm{AG}}, L_{\mathrm{AU}}\}) = \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{AF}}, L_{\mathrm{AG}}, L_{\mathrm{AU}}\}) = \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{AU}}\})$.

Remark 10.4 Suppose that $R = \{0,1\}$. To each finite alphabet A let us associate the ranked alphabet Σ_A of rank type R with $(\Sigma_A)_1 = A$ and $(\Sigma_A)_0 = \{\#\}$. We may identify each word $u \in A^*$ with a term in T_{Σ_A}. Using this identification, the logic CTL essentially becomes LTL, propositional linear temporal logic, cf. [20]. It follows from the above algebraic characterization of the class **CTL** that a language $L \subseteq A^*$ is definable in LTL iff its minimal automaton belongs to the least class of ordinary automata containing the "binary identity-reset automaton", closed under the cascade composition, subautomata, and homomorphic images. This fact was proven in [6] (using the wreath product instead of the cascade composition). In fact, our methods and results generalize those of [6]. The binary identity-reset automaton has two states, $0, 1$, and three input letters inducing the two constant functions and the identity function on $\{0,1\}$, respectively. Using the Krohn-Rhodes Decomposition Theorem [7], it then follows that a language $L \subseteq A^*$ is definable in LTL iff its syntactic monoid is aperiodic. Thus, by the characterization the expressive power of first-order logic on finite words in [17], one derives Kamp's theorem [16] to the effect that first-order logic on finite words is equivalent to propositional linear temporal logic, see also [12].

Recall from Example 4.6 the definition of the languages $L_{d,r}$, where $d > 1$ and $0 \le r < d$. The minimal tree automaton $\mathbb{M}_d$ of $L_{d,r}$ has d elements, $0, \ldots, d-1$, and operations

$$\begin{array}{rcl} \uparrow_n (r_1,\ldots,r_n) & = & (r_1 + \ldots + r_n + 1) \mod d \\ \downarrow_n (r_1,\ldots,r_n) & = & (r_1 + \ldots + r_n) \mod d, \end{array}$$

for all $r_1,\ldots,r_n \in \{0,\ldots,d-1\}$ and $n \in R$. For each d, let $\mathcal{L}_d = \{L_{d,r} : 0 \le r < d\}$, and let $\mathcal{L}_{\mathrm{mod}} = \bigcup_{d>1} \mathcal{L}_d$. Define

$$\begin{array}{rcl} \mathbf{CTL} + \mathbf{MOD}(d) & = & \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{EU}}\} \cup \mathcal{L}_d) \\ \mathbf{CTL} + \mathbf{MOD} & = & \mathbf{FTL}(\mathcal{L}_{\mathrm{X}} \cup \{L_{\mathrm{EU}}\} \cup \mathcal{L}_{\mathrm{mod}}). \end{array}$$

Using Theorem 9.5, we obtain:

Theorem 10.5 *1. For every $d > 1$, a tree language belongs to* **CTL+MOD**(d) *iff its minimal tree automaton is in $\widehat{\{\mathbb{E}_U, \mathbb{M}_d\}}$.*

2. A tree language belongs to **CTL + MOD** *iff its minimal tree automaton is in the least closed variety containing $\mathbb{E}_U$ and the tree automata $\mathbb{M}_d$, $d > 1$.*

11 Conclusion

We have associated a modal operator with each language L of finite trees, and a logic FTL($\mathcal{L}$) with each class $\mathcal{L}$ of languages of finite trees. We have shown that several natural modal operators can be captured by suitably chosen languages. Then, for certain classes $\mathcal{L}$ of regular tree languages, we reduced the problem of the characterization of the expressive power of the logic FTL($\mathcal{L}$) to an algebraic problem thus making it possible to study the expressive power of the logics involved by the methods of algebra. This approach has been very fruitful for logics on finite and ω-words. Our general results have many immediate applications and we have presented some.

In order to transform the obtained concrete algebraic characterizations (e.g., that in Theorem 10.1) into decision procedures, one has to develop a structure theory of finite algebras. In Part 3, we will use Theorem 10.1 to derive an effective characterization of the language class **CTL**(X, EF). This characterization complements the results recently obtained in [5]. In Part 2, we will extend our general results to finite trees such that the outgoing edges of a vertex are not ordered.

References

[1] J. Almeida, On pseudovarieties, varieties of languages, filters of congruences, pseudoidentities and related topics, *Algebra Universalis*, 27(1990), 333–350.

[2] J. Almeida, *Finite Semigroups and Universal Algebra*, World Scientific, 1994.

[3] A. Baziramwabo, P. McKenzie and D. Therien, Modular temporal logic, in: *Proc. 1999 IEEE Conference LICS, Trento, Italy.*

[4] M. Ben-Ari, Z. Manna and A. Pnueli, The temporal logic of branching time, in: *Proc. Symp. Principles of Programming Languages*, 1981, 164–176.

[5] M. Bojanczyk and I. Walukiewicz, Characterising EF and EX tree logics, *proc. CONCUR 2004*, to appear.

[6] J. Cohen, D. Perrin and J.-E. Pin, On the expressive power of temporal logic, *J. Computer and System Sciences*, 46(1993), 271–294.

[7] S. Eilenberg, *Automata, Languages, and Machines*, vol. A and B, Academic Press, 1974 and 1976.

[8] Z. Ésik, Definite tree automata and their cascade compositions, *Publ. Math.*, 48(1996), 243–262.

[9] Z. Ésik, A variety theorem for trees and theories, *Publ. Math.*, 54(1999), 711–762.

[10] Z. Ésik, Extended temporal logic on finite words and wreath products of monoids with distinguished generators, in: *Proc. DLT 02, Kyoto*, LNCS 2450, Springer, 2003, 43–58.

[11] Z. Ésik and P. Weil, On logically defined recognizable tree languages, in: *Proc. FST&TCS 03, Mumbai*, LNCS 2914, Springer, 2003, 195–206.

[12] D. Gabbay, A. Pnueli, S. Shelah and J. Stavi, On the temporal analysis of fairness, in: *Proc. 7th ACM Symp. Principles of Programming Langauges*, ACM, 1980, 163–173.

[13] F. Gécseg and M. Steinby, *Tree Automata*, Akadémiai Kiadó, 1984.

[14] G. Grätzer, *Universal Algebra*, 2nd ed., Springer, 1979.

[15] U. Heuter, Definite tree languages, *Bulletin of the EATCS*, 35(1988), 137–142.

[16] J. A. Kamp, Tense logic and the theory of linear order, Ph. D. Thesis, UCLA, 1968.

[17] R. McNaughton and S. Papert, *Counter-Free Automata*, MIT Press, 1971.

[18] M. Nivat and A. Podelski, Definite tree languages (cont'd), *Bulletin of the EATCS*, 38(1989), 186–190.

[19] G. Ricci, Cascades of tree-automata and computations in universal algebras. *Math. Systems Theory*, 7(1973), 201–218.

[20] K. Schneider, *Verification of Reactive Systems*, Springer, 2004.

[21] M. Steinby, A theory of tree language varieties, in: *Tree Automata and Languages*, North-Holland, Amsterdam, 1992, 57–81.

[22] M. Steinby, General varieties of tree languages, *Theoret. Comput. Sci.*, 205(1998), 1–43.

[23] H. Straubing, *Finite Automata, Formal Logic, and Circuit Complexity*, Birkhauser, 1994.

[24] H. Straubing, D. Therien and W. Thomas, Regular languages defined with generalized quantifiers, *Information and Computation*, 118(1995), 289–301.

[25] J. W. Thatcher and J. B. Wright, Generalized finite automata theory with an application to a decision problem of second-order logic. *Math. Systems Theory*, 2(1968), 57–81.

[26] D. Therien and Th. Wilke, Temporal logic and semidirect products: An effective characterization of the until hierarchy, *DIMACS TR-96-28*, 1996.

[27] P. Wolper, Temporal logic can be more expressive, *Information and Control*, 56(1983), 72–99.