Part II

CAAP

First-Order Logic on Finite Trees *

Andreas Potthoff

IRISA, Campus de Beaulieu
F-35042 Rennes Cedex
e-mail: potthoff@irisa.fr

Abstract. We present effective criteria for first-order definability of regular tree languages. It is known that over words the absence of modulo counting (the "noncounting property") characterizes the expressive power of first-order logic (McNaughton, Schützenberger), whereas noncounting regular tree languages exist which are not first-order definable. We present new conditions on regular tree languages (more precisely, on tree automata) which imply nondefinability in first-order logic. One method is based on tree homomorphisms which allow to deduce nondefinability of one tree language from nondefinability of another tree language. Additionly we introduce a structural property of tree automata (the so-called $\wedge$-$\vee$-patterns) which also causes tree languages to be undefinable in first-order logic. Finally, it is shown that this notion does not yet give a complete characterization of first-order logic over trees. The proofs rely on the method of Ehrenfeucht-Fraïssé games .

1 Introduction

Regular word languages as well as many subclasses of regular word languages can be defined using very different formalisms, e.g. finite automata, regular expressions and monadic second-order formulas. An important example is the class of star-free word languages that has been investigated by McNaughton and Schützenberger [10, 15]. In particular it was shown by McNaughton that this class coincides with the class of first-order definable word languages and from Schützenberger we know that a regular word language is star-free iff the language is aperiodic. (A word language L is called aperiodic if there exists $n \in \mathbb{N}$ such that for all $u, v, w \in \Sigma^*$ we have $uv^n w \in L \iff uv^{n+1}w \in L$.) Since aperiodicity is decidable for a regular word language by inspecting the minimal deterministic automaton accepting this language, also first-order definability turns out to be decidable.

In the introduction to [1] Büchi wrote: "The extension from unary algebras to n-ary algebras (tree automata) sometimes is obvious and sometimes requires additional ideas." The "obvious extension" applies well to the definition of regular tree languages in the above mentioned formalisms whereas the notions of

* The present work was supported by EBRA Working Group 6317 "Algebraic and Syntactic Methods in Computer Science (ASMICS 2)" and by a fellowship of the EC program "Human, Capital and Mobility".

star-freeness, aperiodicity and first-order definability introduced by Thomas [18] yield different classes of regular tree languages. In [14] it was shown that every regular tree language over an alphabet without unary symbols is star-free and in [8] that there exist aperiodic tree languages which are not first-order definable. Thus aperiodicity is only a necessary criterion for first-order definability.

Up to now, the only way to prove that an aperiodic tree language is not first-order definable is to apply the well-known Ehrenfeucht-Fraïssé game [4, 6]. In the present context this game is played by two players on two trees. Trees satisfy the same set of first-order formulas of a certain quantifier-depth n if and only if the second player has a winning strategy in the play with n rounds on these trees. Thus a tree language T is not first-order definable if there exists a sequence $(s_n, t_n)_{n \in \mathbb{N}}$ of trees such that for all $n \in \mathbb{N}$: $s_n \in T$, $t_n \notin T$, and the second player has a winning strategy in the n-round play on s_n and t_n. As can be seen in [8] it is often difficult to construct such a sequence and to verify the required properties.

The aim of this paper is to provide effective necessary conditions for first-order definability. We show that certain mappings on trees (nondeleting linear tree homomorphisms [5]) preserve winning strategies of the second player and thus allow to deduce nondefinability of one tree language from nondefinability of another tree language. We also apply these mappings to show that it suffices to deal with languages of binary trees in order to find a decision procedure for definability in first-order logic.

In [12] it was shown that a certain set of partial boolean expressions is aperiodic, but not first-order definable. From this example we extract the notion of an $\wedge$-$\vee$-pattern which is a condition on state transformations in the corresponding minimal tree automaton. We show that this notion provides a quite powerful necessary condition for first-order definability but still does not characterize the class of first-order definable tree languages. The example language proving that the absence of an $\wedge$-$\vee$-pattern does not suffice to insure first-order definability also corrects an error in [18] by showing that there exists an aperiodic tree language that is definable in chain logic but not in first-order logic.

The remainder of this paper has 5 sections. In Section 2 we introduce regular tree languages in terms of tree automata and monadic second-order formulas. In Section 3 we present an Ehrenfeucht-Fraïssé game for first-order logic on finite trees and recall some basic facts on this game. In Sections 4 we investigate tree homomorphisms and in Section 5 $\wedge$-$\vee$-patterns. An outline of future work will be given in Section 6. For lack of space some proofs have to be omitted which can be found in [13].

2 Notation

Let $\Sigma = \Sigma_0 \cup \ldots \cup \Sigma_r$ be a *finite ranked alphabet* where Σ_i denotes the set of all symbols of arity i. Let furthermore V denote a set of 0-ary variables. The set of all trees over Σ with variables in V, denoted by $T_\Sigma(V)$, is inductively defined as follows: every 0-ary symbol $a \in \Sigma_0$ and every variable $c \in V$ belongs to $T_\Sigma(V)$

and with $b \in \Sigma_i$ and $t_1, \ldots, t_i \in T_\Sigma(V)$ also $b(t_1, \ldots, t_i)$ belongs to $T_\Sigma(V)$. T_Σ denotes the set of trees over Σ without any variables. $T \subseteq T_\Sigma$ is called a tree language over Σ. If additionally $\Sigma = \Sigma_0 \cup \Sigma_2$ we call a tree language over Σ binary. In the remainder of this section we denote by a always a 0-ary symbol, by c a variable and by b a symbol of positive arity. The labelling of the leaves of a tree t from left to right is denoted $yield(t)$, e.g. $yield(a) = a$, $yield(c) = c$ and $yield(b(t_1, \ldots, t_i)) = yield(t_1) \cdot \ldots \cdot yield(t_i)$. With each tree we associate a prefix closed set $dom(t) \subset \{1, \ldots, r\}^*$ called the domain of t and defined by $dom(a) = dom(c) = \{\varepsilon\}$ and $dom(b(t_1, \ldots, t_i)) = \{\varepsilon\} \cup \bigcup_{k=1,\ldots,i} k \cdot dom(t_k)$. $<$ denotes the partial prefix ordering on the domain of a tree. We refer to the set of leaves of a tree t by $front(t) \subseteq dom(t)$. For $k \in dom(t)$ we denote by $t(k) \in \Sigma$ the label of node k in t. t^k denotes the tree obtained from t by removing all nodes that are not greater than k, i.e. $dom(t^k) = \{\, l \mid kl \in dom(t) \,\}$ and $t^k(l) = t(kl)$ for all $l \in dom(t^k)$.

We will now introduce a concatenation on trees via replacement of variables. Let $T, T_1, \ldots, T_m \subseteq T_\Sigma(V)$ and $\{c_1, \ldots, c_m\} \subseteq V$, then we denote by $T[c_1 \leftarrow T_1, \ldots, c_m \leftarrow T_m]$ the set of all trees obtained from trees in T by replacing all variables c_i by possibly different trees in the corresponding tree language T_i, more formally we put $T[\bar{c} \leftarrow \bar{T}] = \bigcup_{t \in T} t[\bar{c} \leftarrow \bar{T}]$ with $a[\bar{c} \leftarrow \bar{T}] = a$ for $a \in \Sigma_0 \cup V \setminus \{c_1, \ldots, c_m\}$, $c_i[\bar{c} \leftarrow \bar{T}] = T_i$ and $b(t_1, \ldots, t_i)[\bar{c} \leftarrow \bar{T}] = \{b(t'_1, \ldots, t'_i) \mid t'_j \in t_j[\bar{c} \leftarrow \bar{T}] \text{ for } j = 1, \ldots, i\}$. For $T \subseteq T_\Sigma(V)$ and $c \in V$ we put $T^{*,c} = \bigcup_{i \geq 0} T^{c,i}$ with $T^{c,0} = \{c\}$ and $T^{c,i+1} = T^{c,i} \cup T^{c,i}[c \leftarrow T]$.

A tree $s \in T_\Sigma(\{c\})$ with exactly one leaf labelled c is called a *special tree*. The set of all special trees is denoted by S_Σ. For $s \in S_\Sigma$ and $t \in T_\Sigma \cup S_\Sigma$ we write $s \cdot t$ instead of $s[c \leftarrow t]$. $(S_\Sigma, \cdot, c)$ is a monoid that serves to extend the notions of Nerode-congruence and aperiodicity from word languages to tree languages. Let $T \subseteq T_\Sigma$ and $t, t' \in T_\Sigma$. t and t' are called (Nerode-)congruent with respect to T ($t \cong_T t'$) iff $\forall s \in S_\Sigma \; s \cdot t \in T \iff s \cdot t' \in T$. T is called aperiodic iff $\exists n \in \mathbb{N} \; \forall s', s \in S_\Sigma \forall t \in T_\Sigma \; s' \cdot s^n \cdot t \in T \iff s' \cdot s^{n+1} \cdot t \in T$.

A *deterministic bottom-up tree automaton (DBA)* $\mathcal{A} = (Q, \Sigma, \delta, F)$ consists of a finite set of states Q, a set of final states $F \subseteq Q$, a ranked alphabet Σ and a transition function $\delta : \bigcup_{k=0,\ldots,r} \Sigma_k \times Q^k \to Q$. We can extend δ in the usual way to a mapping $\delta : T_\Sigma \to Q$. The tree language accepted by $\mathcal{A}$ is $T(\mathcal{A}) = \{t \in T_\Sigma \mid \delta(t) \in F\}$. A tree language T is called *regular* if $T = T(\mathcal{A})$ for a DBA $\mathcal{A}$. Let $p \in Q$, $s \in S_\Sigma$ and $t \in T_\Sigma$ with $\delta(t) = p$. Then we put $\delta(s, p) = \delta(s \cdot t)$.

As in the case of regular word languages there exists a *minimal* DBA for every regular tree language which is unique up to state renaming and which is characterized by the following two conditions: $\forall q \in Q \; \exists t \in T_\Sigma \; \delta(t) = q$ and $\forall p, q \in Q \quad p \neq q \Rightarrow \exists s \in S_\Sigma \;\; \delta(s, p) \in F \iff \delta(s, q) \notin F$. Furthermore the minimal DBA can be computed effectively from an arbitrary DBA accepting the tree language. The states of this minimal DBA correspond to the equivalence classes of the Nerode-congruence, thus a tree language is regular iff the Nerode-congruence has finite index. Furthermore, if $\mathcal{A}$ is a minimal DBA accepting T

we have that T is aperiodic iff there exists $n \in \mathbb{N}$ such that for all $s \in S_\Sigma$ and for all $q \in Q$ we have $\delta(s^n, q) = \delta(s^{n+1}, q)$. Therefore we obtain that aperiodicity is decidable for regular tree languages. For a more detailed introduction in tree language theory see [7].

We will now describe tree languages in terms of monadic second-order logic. With every tree t we associate a relational structure $t = (dom(t), <, S_1, \ldots, S_r, (P_a)_{a \in \Sigma})$ where $<$ denotes the partial prefix ordering on $dom(t)$ and S_i denotes the i-th successor relation (with kS_ik' iff $ki = k'$). P_a consists of all nodes labelled a.

We use x, y, ... to denote variables ranging over nodes and X, Y, ... to denote variables ranging over sets of nodes. Atomic formulas are the following: $x < y$, $x = y$, xS_iy, Xx and P_ax. From the atomic formulas we build up the set of all monadic second-order formulas using the boolean connectives $\wedge$, $\vee$, $\neg$, and the quantifiers $\exists$ and $\forall$ for both kinds of variables. We denote by $\varphi(x_1, \ldots, x_n, X_1, \ldots, X_m)$ a formula with free variables among $x_1, \ldots, x_n$ and $X_1, \ldots, X_m$. The satisfaction relation $(t, k_1, \ldots, k_n, K_1, \ldots, K_m) \models \varphi(x_1, \ldots, x_n, X_1, \ldots, X_m)$ is defined as usual. By FO we denote the set of first-order formulas, i.e. formulas without any set variables. The quantifier-depth of a first-order formula is inductively defined as follows: $qd(\varphi) = 0$ for all atomic formulas φ, $qd(\varphi \vee \psi) = qd(\varphi \wedge \psi) = max\{qd(\varphi), qd(\psi)\}$, $qd(\neg\varphi) = qd(\varphi)$ and $qd(\exists x\varphi) = qd(\forall x\varphi) = qd(\varphi) + 1$. Every formula φ without free variables defines a tree language $T(\varphi) = \{t \mid t \models \varphi\}$. A tree language is called monadic second-order definable (first-order definable) if there exists a monadic second-order formula (first-order formula) φ such that $T = T(\varphi)$. Regular and monadic second-order definable tree languages are related by the following theorem:

Theorem 1. *[2, 17] A tree language T is regular iff T is monadic second-order definable.*

Chain logic and antichain logic have been introduced by Thomas [18] as fragments of monadic second-order logic. In chain logic set quantifiers range only over chains, i.e. sets of nodes that are totally ordered by the partial prefix ordering. An antichain in a tree is a set of nodes such that no two nodes in this set are comparable in the partial prefix ordering. Antichain formulas (where set quantifiers are restricted to range over antichains) define exactly the class of star-free tree languages.

The following theorems summarize the relations between the notions of star-freeness, aperiodicity and first-order definability:

Theorem 2. *[14] Every regular binary tree language is star-free.*

Theorem 3. *[8] Every first-order definable tree language is aperiodic, but there exist aperiodic tree languages that are not first-order definable.*

The only difference between word languages and tree languages of unary trees is that word automata process a word starting from the minimal node in

the usual ordering whereas bottom-up tree automata start at the maximal node in the partial prefix ordering. Since regular word languages are closed against reversing we can apply the results of Schützenberger and McNaughton to sets of unary trees. In [11] Perrin presents an effective construction of a star-free expression defining all words which effect the same transformation on the states of an aperiodic word automaton. Star-free expressions on words can be rewritten syntactically to a first-order formula defining the same word language. With a technique of restricting the range of quantifiers we finally arrive at the following proposition:

Lemma 4. *Let $\mathcal{A} = (Q, \Sigma, \delta, F)$ be an aperiodic DBA and $m : Q \to Q$. Then there exists a first-order formula $\varphi_m(x)$ such that for all $t \in T_\Sigma$ we have: $(t, k) \models \varphi_m(x)$ iff the maximal unary subtree above x effects state transformation m.*

3 An Ehrenfeucht-Fraïssé Game

Ehrenfeucht-Fraïssé games are widely used in the literature to show that certain properties of relational structures are not expressible in a given logic [3, 4, 6]. In this section we recall the rules of the first-order game played on trees as introduced in [8]. Furthermore we enumerate some basic facts on this game. Most proofs can be found in [8, 12, 13] and are therefore omitted here.

The Ehrenfeucht-Fraïssé game is played on two trees by two players, here called player I and player II, with the aim of player I to show that the given trees are different. By $G_n((t, k_1, \ldots, k_m), (t', k'_1, \ldots, k'_m))$ we denote a play with n rounds on two trees t and t' with a sequence of m specified nodes in both trees. In each move player I chooses a node in t or t' and player II reacts by choosing a node in the other tree. Let k_{m+i} resp. k'_{m+i} be the nodes chosen in t resp. t' in the i-th round. Player II wins the play if the nodes $k_1, \ldots, k_{n+m}$ in t and $k'_1, \ldots, k'_{n+m}$ in t' define a partial isomorphism from t to t', i.e. if we have for all $1 \leq i, j \leq m+n$, for all $a \in \Sigma$ and for all $1 \leq l \leq r$: $k_i = (<, S_l)\, k_j \iff k'_i = (<, S_l)\, k'_j$ and $P_a k_i \iff P_a k'_i$. Since we deal with finite trees either player I or player II has a winning strategy. This fact yields to the following definition:

Definition 5. Let T_Σ^m denote the set of all trees in T_Σ with a sequence of m specified nodes and let FO^m denote the set of all first-order formulas with m free variables. For $(t, \bar{k}), (t', \bar{k}') \in T_\Sigma^m$ we denote by $(t, \bar{k}) \approx_n (t', \bar{k}')$ that player II has a winning strategy in the play $G_n((t, \bar{k}), (t', \bar{k}'))$.

$\approx_n$ and $\approx_{n+1}$-equivalence are related by the so-called back and forth property:

Lemma 6. *Let $(t, \bar{k}), (t', \bar{k}') \in T_\Sigma^m$.*
$(t, \bar{k}) \approx_{n+1} (t', \bar{k}')$ iff
$\forall k \in dom(t)\ \exists k' \in dom(t')$ $(t, \bar{k}, k) \approx_n (t', \bar{k}', k')$ and
$\forall k' \in dom(t')\ \exists k \in dom(t)$ $(t, \bar{k}, k) \approx_n (t', \bar{k}', k')$

From this property the following theorem can be deduced easily:

Theorem 7. *[8] The relations $\approx_n$ are equivalence relations of finite index on T_Σ^m and we have for all $(t,\bar{k}),(t',\bar{k}') \in T_\Sigma^m$:*
$(t,\bar{k}) \approx_n (t',\bar{k}')$ *iff*
$\forall \varphi(\bar{x}) \in FO^m \ qd(\varphi(\bar{x})) \leq n \Rightarrow [(t,\bar{k}) \models \varphi(\bar{x}) \iff (t',\bar{k}') \models \varphi(\bar{x})]$

We will use the last theorem in the following way to prove that a tree language is not first-order definable:

Corollary 8. *$T \subseteq T_\Sigma$ is not first-order definable iff there exists a sequence $(s_n,t_n)_{n\in\mathbb{N}}$ such that $s_n \in T$, $t_n \notin T$ and $s_n \approx_n t_n$ for all $n \in \mathbb{N}$.*

We will construct such sequences of pairs of trees by induction. For the inductive step we need some technical propositions that allow us to construct $\approx_{n+1}$-equivalent trees from $\approx_n$-equivalent ones. The first proposition states that winning strategies of player II on certain parts of two trees can be combined to get a winning strategy for the whole trees.

Lemma 9 (Composition Lemma).
Let $s,t \in S_\Sigma$ with $s(l) = t(k) = c$ and $s',t' \in T_\Sigma$.

$$\begin{array}{lccc} & (s,l_1,\ldots,l_m,l) & \approx_n & (t,k_1,\ldots,k_m,k) \\ \wedge & (s',\varepsilon,l_1',\ldots,l_r') & \approx_n & (t',\varepsilon,k_1',\ldots,k_r') \\ & & \iff & \end{array}$$

$$(s\cdot s',l_1,\ldots,l_m,l,ll_1',\ldots,ll_r') \approx_n (t\cdot t',k_1,\ldots,k_m,k,kk_1',\ldots,kk_r')$$

The previous lemma allows us to weaken the precondition of Corollary 8.

Remark 10. *Let T be a regular tree language. T is not first-order definable iff there exists a sequence $(s_n,t_n)_{n\in\mathbb{N}}$ such that $s_n \not\cong_T t_n$ and $s_n \approx t_n$ for all $n \in \mathbb{N}$.*

Proof. The direction from left to right follows directly from Corollary 8. For the reverse direction we use that the Nerode-congruence has finite index. Thus there exists a subsequence $(s_n',t_n')_{n\in\mathbb{N}}$ such that $s_n' \cong_T s_m'$ and $t_n' \cong_T t_m'$ for all $n,m \geq 0$. From the definition of this congruence we obtain a special tree s such that $s\cdot s_n' \in T \iff s\cdot t_n' \notin T$ for all $n \in \mathbb{N}$. The Composition Lemma 9 shows that either the sequence $(s(s_n'),s(t_n'))_{n\in\mathbb{N}}$ or the sequence $(s(t_n'),s(s_n'))_{n\in\mathbb{N}}$ proves nondefinability of T according to Corollary 8.

A more powerful method to combine winning strategies is formulated in the following proposition.

Lemma 11 (Substitution Lemma). *[13]*
Let $(u,l_1,\ldots,l_m),(v,k_1,\ldots,k_m) \in T_\Sigma^m(\{c_1,\ldots,c_r\})$ and $T_1,\ldots,T_r \subseteq T_\Sigma$ such that
$(u,l_1,\ldots,l_m) \approx_n (v,k_1,\ldots,k_m)$ *and*
$(s,\varepsilon) \approx_n (s',\varepsilon)$ *for all $s,s' \in T_i$ $(i=1,\ldots,r)$.*
Then $(t,l_1,\ldots,l_m) \approx_n (t',k_1,\ldots,k_m)$ for all $t \in u[c_1 \leftarrow T_1,\ldots,c_r \leftarrow T_r]$ and $t' \in v[c_1 \leftarrow T_1,\ldots,c_r \leftarrow T_r]$.

The following remark describes exactly the way how sequences of pairs of trees as required in Corollary 8 are constructed in the sequel.

Remark 12. *Let* $u, v \in T_\Sigma(\{c_1, \ldots, c_r\})$, $t_1, t_1', \ldots, t_r, t_r' \in T_\Sigma$, $s \in u[c_i \leftarrow \{t_i, t_i'\} \mid 1 \leq i \leq r]$ *and* $t \in v[c_i \leftarrow \{t_i, t_i'\} \mid 1 \leq i \leq r]$ *such that*

(1) $(u, \varepsilon) \approx_{n+1} (v, \varepsilon)$

(2) $(t_i, \varepsilon) \approx_n (t_i', \varepsilon)$ *for* $1 \leq i \leq r$

(3) $\forall k \in front(u)\ \exists k' \in front(v) \quad (u, \varepsilon, k) \approx_n (v, \varepsilon, k') \wedge s^k = t^{k'}$
$\forall k' \in front(v)\ \exists k \in front(u) \quad (u, \varepsilon, k) \approx_n (v, \varepsilon, k') \wedge s^k = t^{k'}$

Then we have $(s, \varepsilon) \approx_{n+1} (t, \varepsilon)$.

4 Tree Homomorphisms

In this section we introduce tree homomorphisms (see also [5]) and show that these mappings preserve winning strategies of player II. Then we use tree homomorphisms to show that it suffices to deal with binary tree languages in order to obtain a decidability result for first-order logic.

Definition 13. In order to define tree homomorphisms we extend the notion of special trees. The set of *n-special trees*, denoted by S_Σ^n, consists of all trees $t \in T_\Sigma(\{c_1, \ldots, c_n\})$ such that $yield(t) \in \Sigma_0^* c_1 \Sigma_0^* \ldots \Sigma_0^* c_n \Sigma_0^*$. For $t \in S_\Sigma^n$ and $t_1, \ldots, t_n \in T_\Sigma$ we abbreviate $t[c_1 \leftarrow t_1, \ldots, c_n \leftarrow t_n]$ by $t(t_1, \ldots, t_n)$.

A *tree homomorphism* from T_Σ to T_Ω assigns to a symbol $b \in \Sigma_k$ a k-special tree $h(b) \in S_\Omega^k$, i.e. $h : \bigcup_{k=0,\ldots,r} \Sigma_k \to S_\Omega^k$. h induces a mapping $h^* : T_\Sigma \to T_\Omega$ by $h^*(a) = h(a)$ for $a \in \Sigma_0$ and $h^*(b(t_1, \ldots, t_k)) = h(b)(h^*(t_1), \ldots, h^*(t_k))$ for $b \in \Sigma_i$. In the sequel we will identify h and h^*.

Since copying (multiple occurences of a variable c_i) is not allowed for tree homomorphisms we can transfer a winning strategy of player II on two trees to the images of these trees. At first we show that the restriction to noncopying mappings is indeed necessary.

Example 1. Let $\Sigma = \Sigma_0 \cup \Sigma_1$ with $\Sigma_0 = \{a\}$ and $\Sigma_1 = \{b\}$, $T \subset T_\Sigma$ the set of all trees (words) of even height, $\Omega = \Omega_0 \cup \Omega_2$ with $\Omega_0 = \{a\}$ and $\Omega_2 = \{b\}$ and finally $T_{even} \subset T_\Omega$ the set of all trees that contain a path of even length.

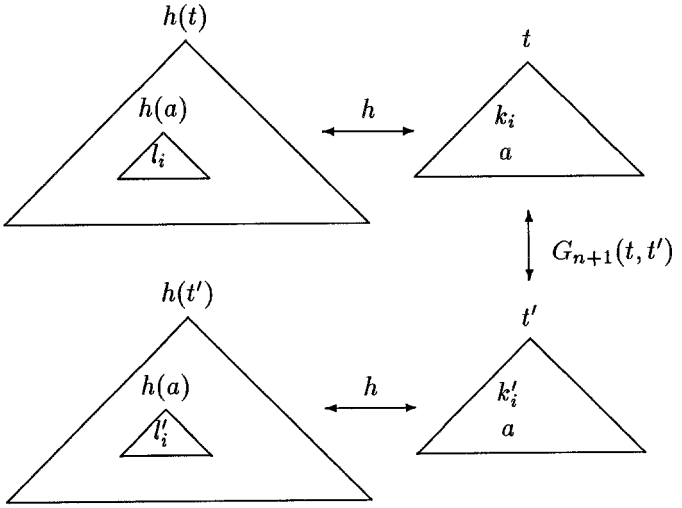
T is periodic and therefore not first-order definable. Let $h : T_\Sigma \to T_\Omega$ be defined by $h(a) = a$ and $h(b) = b(c_1, c_1)$. Then h is syntactic for T and T_{even}, but T_{even} is first-order definable.

We call the path that starts at node k and contains all nodes in the set $k(12)^*(1 \cup \varepsilon)$ the "zig-zag" path starting at k. This path has even length if the final node belongs to the set $k(12)^*$ and has odd length otherwise. It is easy to construct a first-order formula $\varphi_{even}(x)$ which is satisfied by all nodes k with a zig-zag path of even length starting at k. A tree contains a least one path of even length and one path of odd path of length iff there exists a node k such that all path below the left successor of k have even length and all path below the right successor have odd length or vice versa. In particular $\varphi_{even}(x)$ applies

to $k1$ iff $\varphi_{even}(x)$ does not apply to $k2$ in this case. Hence we are able to define the set of all trees that contain at least one path of even length and one path of odd length. If a tree has only paths of even length or only paths of odd length, then also the zig-zag path starting at the root has even length or odd length respectively. So a disjunction of two formulas defines T_{even}, one defining the set of trees with paths of even and odd length and the other one defining the trees with a zig-zag path of even length starting at the root.

Lemma 14. *Let $h : T_\Sigma \to T_\Omega$ be a tree homomorphism and $t, t' \in T_\Sigma$. Then we have $t \approx_{n+1} t' \Rightarrow h(t) \approx_n h(t')$.*

Proof. Let us call a subtree in $h(t)$ or $h(t')$ resulting from a node labelled a in t or t' an a-segment. Suppose player I chooses a node l in $h(t)$. Let k be the preimage of this node in t. Then player II chooses a node k' in t' according to his winning strategy in the $n+1$-round game on t and t'. If the node k in t is labelled with a then also k' in t' is labelled with a. In a last step player II computes the a-segment which is the image of k' in $h(t')$ and chooses in this segment the same node as player I. The strategy of player II is shown in the following picture.



After n moves the game ends with n chosen nodes in $h(t)$, $h(t')$, t and t'. Let us call these nodes l_i, l'_i, k_i and k'_i for $i = 1, \ldots, n$. Now we have to verify the winning conditions for player II . We only show the most difficult condition to verify, i.e. $l_i < l_j \iff l'_i < l'_j$.

So let $l_i < l_j$. If l_i and l_j belong to the same segment, i.e. $k_i = k_j$, then also $k'_i = k'_j$ and since player II has chosen the same nodes in corresponding segments, we also have $l'_i < l'_j$.

It remains to investigate the case that l_i and l_j do not belong to the same segment. Thus we have $k_i < k_j$ and $k'_i < k'_j$. From this fact we can not deduce directly that $l'_i < l'_j$ because not all nodes in the segment belonging to k'_i are predecessors of all nodes in the segment belonging to k'_j. Let $k_i r$ be the direct successor of k_i above k_j, i.e. $k_i r \leq k_j$. Thus l_i is a predecessor of the segment

belonging to the r-th successor of k_i and so is l'_i for the segment belonging to $k'_i r$. Since player II has played in t and t' according to his winning strategy in the game with $n+1$ moves and up to know only n moves a played, we can deduce that also $k'_i r \leq k'_j$. Thus l'_i is a predecessor of all nodes in the segment of k'_j, in particular of l'_j.

In order to be able to transfer nondefinability from a tree language T to another tree language T' using a tree homomorphism we need an additional condition stating that the tree homomorphism respects the Nerode-congruences of T and T'.

Definition 15. A tree homomorphism $h : T_\Sigma \to T_\Omega$ is called syntactic for $T \subseteq T_\Sigma$ and $T' \subseteq T_\Omega$ iff for all $t, t' \in T_\Sigma \quad t \not\equiv_T t' \Rightarrow h(t) \not\equiv_{T'} h(t')$.

Lemma 14 and Remark 10 yield immediately:

Theorem 16. *Let $T \subseteq T_\Sigma$ and $T' \subseteq T_\Omega$ be regular tree languages. If T is not first-order definable and if $h : T_\Sigma \to T_\Omega$ is a syntactic tree homomorphism for T and T', then T' is not first-order definable.*

One application of tree homomorphisms is the proof of the following theorem, stating that the decidability problem for first-order logic can be reduced to binary tree languages over a one letter alphabet.

Theorem 17. *[13] For every regular tree language T, we are able to construct effectively a regular binary tree language T' over a one letter alphabet such that T is first-order definable iff T' is first-order definable.*

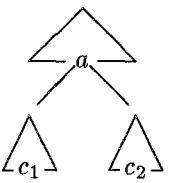
Proof. The proof can be divided into two parts. In the first part we eliminate unary symbols. Therefore state transformations obtained from unary subtrees are shifted to minimal nonunary nodes below. Then we code non-unary symbols in a binary one letter alphabet. The correctness of these transformations is shown using tree homomorphisms and manipulations of first-order formulas.

The next lemma states that it is decidable whether a syntactic tree homomorphism between two regular binary tree languages exists. The restriction to binary tree languages is motivated by Theorem 17 and also simplifies the proof. Thus in order to obtain decidability of first-order logic it would suffice to find a recursively enumerable set of regular binary tree languages such that every other nondefinable regular tree language is related via a syntactic tree homomorphism to one of these tree languages. It can be shown that at least no finite set of tree languages has this property.

Lemma 18. *Let T and T' be regular binary tree languages. Then it is decidable if there exists a syntactic tree homomorphism for T and T'.*

Proof of Lemma 18. Let $\mathcal{A} = (Q, \Sigma, \delta, F)$ and $\mathcal{A}' = (Q', \Omega, \delta', F')$ be minimal DBA accepting T and T' respectively. Every tree homomorphism h induces a mapping $H : Q \to 2^{Q'}$ defined by $H(q) = \{\delta(h(t)) | \delta(t) = q\}$ with the following properties: (i) $H(q) \neq \emptyset$ for $q \in Q$, (ii) $\delta'(h(a)) \in H(\delta(a))$ for all $a \in \Sigma_0$, (iii) $\delta'(h(b), H(p_1), H(p_2)) \subseteq H(\delta(b, p_1, p_2))$ for all $b \in \Sigma_2$ and all $p_1, p_2 \in Q$. h is syntactic if additionly (iv) $H(p) \cap H(q) = \emptyset$ for all $p \neq q$.

Condition (ii) can be satisfied for all H that already satisfy condition (i). Thus it suffices to test all mappings H which satisfy conditions (i) and (iv) whether for all $b \in \Sigma_2$ there exists $t_b \in S^2_\Omega$ such that $\delta'(t_b, H(p_1), H(p_2)) \subseteq H(\delta(b, p_1, p_2))$ for all $p_1, p_2 \in Q$. Therefore we only have to show that we can compute all binary mappings $m : Q \times Q \to Q$ effected by any tree in S^2_Ω. Every 2-special tree can be decomposed in 3 special trees and a binary symbol as shown in the following picture.



Thus it suffices to compute all state transformations obtained from special trees and then to combine these transformations with all binary symbols in all possible ways as indicated in the previous picture.

An application of Theorem 16 will be given in the next chapter.

5 $\wedge$-$\vee$-Patterns

In [12] it was shown that a certain set of partial boolean expressions is not first-order definable. The simplicity of this particular tree language allows us to transfer the proof of nondefinability in first-order logic of this tree language to all tree languages that satisfy a certain condition (called $\wedge$-$\vee$-pattern) on state transformations in the corresponding minimal DBA. We first recall the definition of this tree language.

Definition 19. Let $\Sigma^B = \Sigma^B_0 \cup \Sigma^B_2$ with $\Sigma^B_0 = \{0, 1\}$ and $\Sigma^B_2 = \{\wedge, \vee\}$. Let $\mathcal{A}_{\wedge,\vee}$ be the DBA over Σ^B with set of states $\{0, 1, \bot\}$, set of final states $\{1\}$ and the transition function γ with $\gamma(0) = 0$ and $\gamma(1) = 1$. For $\wedge$ and $\vee$ we define γ in the following tables:

$\wedge$	0	1	$\bot$
0	$\bot$	0	$\bot$
1	0	1	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$

$\vee$	0	1	$\bot$
0	0	1	$\bot$
1	1	$\bot$	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$

We denote $T(\mathcal{A}_{\wedge,\vee})$ by $T_{\wedge,\vee}$.

It is easy to see that $T_{\wedge,\vee}$ is aperiodic, because $\wedge$ and $\vee$ are monoton on 0 and 1. Now we introduce the notion of an $\wedge$-$\vee$-pattern.

Definition 20. Let $\mathcal{A} = (Q, \Omega, \delta, F)$ be a DBA and let t_q denote an arbitrary tree with $\delta(t_q) = q$ for all $q \in Q$. We say that $\mathcal{A}$ contains an $\wedge$-$\vee$-pattern if there exists a nonempty, irreflexive and symmetrical relation $R \subseteq Q \times Q$ such that there exists for all $(p, p') \in R$ a tree $t_{p,p'} \in T_\Omega(\{x_{r,r'}, y_{r,r'} \mid (r, r') \in R\})$ such that

(i) $\delta(t_{p,p'}[x_{r,r'} \leftarrow t_r, y_{r,r'} \leftarrow t_r]) = p$,
(ii) $\delta(t_{p,p'}[x_{r,r'} \leftarrow t'_r, y_{r,r'} \leftarrow t_r]) = p'$ and
(iii) $\delta(t_{p,p'}[x_{r,r'} \leftarrow t_r, y_{r,r'} \leftarrow t'_r]) = p'$.

The following remark justifies the notion $\wedge$-$\vee$-pattern by showing that $t_{p,p'}$ acts as a generalized $\vee$ for $p = 0$ and $p' = 1$.

Remark 21. *Let $\mathcal{A} = (Q, \Omega, \delta, F)$ be a DBA and let t_q be a tree with $\delta(t_q) = q$ for all $q \in Q$. Let furthermore $R \subseteq Q \times Q$ and $t_{p,p'} \in T_\Omega(\{x_{r,r'}, y_{r,r'} \mid (r, r') \in R\})$ for all $(p, p') \in R$ as in the previous definition. We define simultaneously for all $(p, p') \in R$ a mapping $g_{p,p'} : T_{\Sigma^B}(\{c\}) \to T_\Omega(\{c_{r,r'} \mid (r, r') \in R\})$ by*

$g_{p,p'}(0) = t_p$, $g_{p,p'}(1) = t_{p'}$, $g_{p,p'}(c) = c_{p,p'}$,

$g_{p,p'}(\vee(t_1, t_2)) = t_{p,p'}[x_{r,r'} \leftarrow g_{r,r'}(t_1), y_{r,r'} \leftarrow g_{r,r'}(t_2)]$ *and*

$g_{p,p'}(\wedge(t_1, t_2)) = t_{p',p}[x_{r,r'} \leftarrow g_{r',r}(t_1), y_{r,r'} \leftarrow g_{r',r}(t_2)]$.

Then we have for all $t \in T_{\Sigma^B}$ and all $(p, p') \in R$:

$\gamma(t) = 0 \Rightarrow \delta(g_{p,p'}(t)) = p$ *and* $\gamma(t) = 1 \Rightarrow \delta(g_{p,p'}(t)) = p'$.

The conclusion in the previous remark does not depend on a particular choice of the trees $g_{p,p'}(0)$ and $g_{p,p'}(1)$ whereas the definition of these mappings does. Therefore we will denote mappings obtained from different choices with the same name and specify $g_{p,p'}(0)$ and $g_{p,p'}(1)$ more precisely when necessary.

Now we turn to the main result of this section stating that the existence of an $\wedge$-$\vee$-pattern implies nondefinability in first-order logic. In the proof we will use the trees $u_i \in T_{\Sigma^B}(\{c\})$ for $i \in \mathrm{IN}$ defined as follows: $u_0 = c$ and $u_m = \wedge(u_{m-1}, u_{m-1})$ if m is even and $u_m = \vee(u_{m-1}, u_{m-1})$ if m is odd.

Lemma 22. *[12] Let l, l', l'' be leaves in u_m ($m \geq 2$) which are not neighbours. Then we can label all leaves of u_m with 0 and 1 in two different ways such that the resulting trees v_m and w_m have the following properties:*

- *$v_m(k) = w_m(k)$ for all leaves k except l*
- *$v_m(l) = v_m(l') = w_m(l') = 0$*
- *$w_m(l) = v_m(l'') = w_m(l'') = 1$*
- *$\gamma(v_m) = 0$ and $\gamma(w_m) = 1$*

Theorem 23. *Let $\mathcal{A} = (Q, \Omega, \delta, F)$ be a minimal DBA that contains an $\wedge$-$\vee$-pattern. Then $T(\mathcal{A})$ is not first-order definable.*

Proof. Let $R \subseteq Q \times Q$ and $t_{p,p'} \in T_\Omega(\{x_{r,r'}, y_{r,r'} \mid (r, r') \in R\})$ for all $(p, p') \in R$ as in Definition 20. We will construct simultaneously for all $(p, p') \in R$ a sequence

$(s_n^{p,p'}, t_n^{p,p'})_{n\in\mathbb{N}}$ such that $\delta(s_n^{p,p'}) = p$, $\delta(t_n^{p,p'}) = p'$ and $(s_n^{p,p'}, \varepsilon) \approx_n (t_n^{p,p'}, \varepsilon)$. Then we obtain nondefinability of $T(\mathcal{A})$ directly from Remark 10.

For $n = 0$ we put $s_0^{p,p'} = g_{p,p'}(\vee(0,0))$ and and $t_0^{p,p'} = g_{p,p'}(\vee(0,0))$ where $g_{p,p'}(0)$ and $g_{p,p'}(1)$ are arbitrary with $\delta(g_{p,p'}(0)) = p$ and $\delta(g_{p,p'}(1)) = p'$.

For the induction step let us assume that $s_n^{r,r'}$ and $t_n^{r,r'}$ have the required properties for all $(r,r') \in R$ and let $(p,p') \in R$ be fixed in the sequel. Now we apply $g_{p,p'}$ to the trees u_m. For $l \in front(u_m)$ let us denote by M_l the set of images of l in $g_{p,p'}(u_m)$. From finiteness of the $\approx_n$-equivalence relation on $T_\Omega(\{c_{r,r'} \mid (r,r') \in R\})$ we obtain that for sufficiently large m there exist $l, l', l'' \in dom(u_m)$ such that:

$(*)$ $\forall k \in M_l\ \exists k' \in M_{l'}\ \exists k'' \in M_{l''}$

$$(g_{p,p'}(u_m), \varepsilon, k) \approx_n (g_{p,p'}(u_m), \varepsilon, k') \approx_n (g_{p,p'}(u_m), \varepsilon, k'').$$

Let v_m and w_m as in Lemma 22. We put $s_{p,p'}^{n+1} = g_{p,p'}(v_m)$ and $t_{p,p'}^{n+1} = g_{p,p'}(w_m)$ with $g_{r,r'}(0) = s_{r,r'}^n$ and $g_{r,r'}(1) = t_{r,r'}^n$ for all $(r,r') \in R$. From Remark 21 we obtain immediately that $\delta(s_{p,p'}^{n+1}) = p$ and $\delta(t_{p,p'}^{n+1}) = p'$.

In order to prove $\approx_{n+1}$-equivalence of $s_{p,p'}^{n+1}$ and $t_{p,p'}^{n+1}$ using Remark 12 let us first mention that $s_{p,p'}^{n+1}, t_{p,p'}^{n+1} \in g_{p,p'}(u_m)[c_{r,r'} \leftarrow \{s_{r,r'}^n, (t_{r,r'}^n\} \mid (r,r') \in R]$ and that $s_{p,p'}^{n+1\,k} = t_{p,p'}^{n+1\,k}$ for all $k \in front(g_{p,p'}(u_m)) \setminus M_l$. Thus it remains to verify condition (c) of Remark 12 for $k \in M_l$. But this condition is guaranteed by the choice of l, l', l'' satisfying $(*)$ and the construction of v_m and w_m.

It remains to state effectiveness of this condition.

Lemma 24. *It is decidable whether an $\wedge$-$\vee$-pattern exists in a DBA $\mathcal{A}$.*

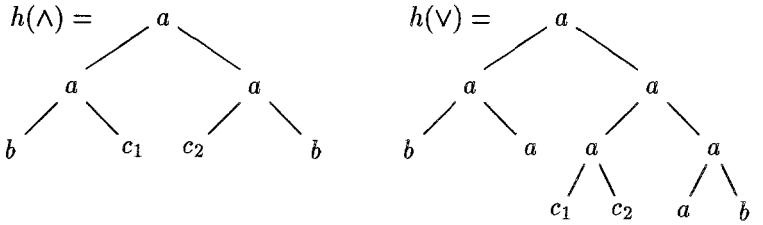
Proof. There is only a finite number of relations $R \subseteq Q \times Q$, hence it suffices to decide whether an $\wedge$-$\vee$-pattern for a fixed R exists. We can assign to every tree $t \in T_\Omega(\{x_{r,r'}, y_{r,r'} \mid (r,r') \in R\})$ three states according to the equations (i)-(iii) in Definition 20. Now let us enumerate all trees in $T_\Omega(\{x_{r,r'}, y_{r,r'} \mid (r,r') \in R\})$ by increasing height and collect the corresponding tupels of states until all trees of a certain height $m \leq |Q|^3$ do not add any new tupel to the collection. Then it remains to test whether for each pair $(p,p') \in R$ there exists a tupel (p,p',p') in the computed set. This is the case if and only if there exists an $\wedge$-$\vee$-pattern for R.

The following examples show applications of Theorem 16 and Theorem 23.

Example 2. The first tree language known to be aperiodic and not first-order definable was introduced by Heuter [8]. The proof of the latter property was quite difficult. In this example we obtain the same result using a tree homomorphism.

Let $\Sigma = \Sigma_0 = \Sigma_2 = \{a,b\}$. A set of nodes in a tree is called a cut if this set is an antichain that is maximal with respect to set inclusion. The labelling of a cut C is the labelling of the nodes of C in the lexicographical ordering. Let $T_{\Sigma^* aa \Sigma^*} = \{t \in T_\Sigma \mid$ every cut of size greater than 1 has a labelling in $\Sigma^* aa \Sigma^*\}$. Let $T \subset T_{\Sigma^B}$ be the set of boolean expressions with value 1 where $\wedge$

and $\vee$ are totally defined, i.e. $\wedge(0,0) = 0$ and $\vee(1,1) = 1$. Finally let h be the tree homomorphism defined as follows: $h(0) = a(b,b)$, $h(1) = a$,



By induction on the height of t, we are able to show:

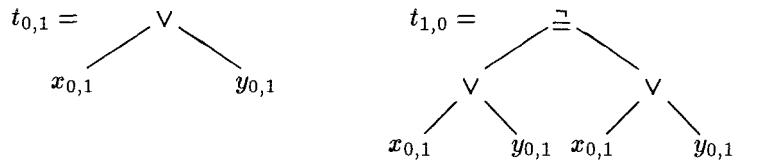
If $t \in T$ then $h(t) \in T_{\Sigma^* aa \Sigma^*}$ and if $t \in T$ then $h(t)$ contains a cut with labelling in $b\Sigma^* b \setminus \Sigma^* aa \Sigma^*$. Thus h is syntactic for T and $T_{\Sigma^* aa \Sigma^*}$ and therefore we obtain that $T_{\Sigma^* aa \Sigma^*}$ is not first-order definable.

Example 3. Let $\Sigma = \Sigma_0 \cup \Sigma_2$ with $\Sigma_2 = \{\vee, \widetilde{=}\}$ and $\Sigma_0 = \{0,1\}$ where $\vee$ and $\widetilde{=}$ are defined in the following tables:

$\widetilde{=}$	0	1	$\perp$
0	1	$\perp$	$\perp$
1	$\perp$	0	$\perp$
$\perp$	$\perp$	$\perp$	$\perp$

$\vee$	0	1	$\perp$
0	0	1	$\perp$
1	1	$\perp$	$\perp$
$\perp$	$\perp$	$\perp$	$\perp$

The following trees show that the minimal automaton (with set of states $\{0,1,\perp\}$) accepting the set of expressions with value 1 contains an $\wedge$-$\vee$-pattern. Let $R = \{(0,1),(1,0)\}$ and



In order to verify that these trees match the required properties, one only has to replace $x_{0,1}$ and $y_{0,1}$ by 0 and 1 respectively and to compute the value of the tree. So we obtain easily that this tree language is not first-order definable.

Many aperiodic tree languages that are not first-order definable can be treated this way. But the tree language introduced in the next definition shows that the absence of an $\wedge$-$\vee$-pattern does not guarantee first-order definability. This tree language is aperiodic, contains no $\wedge$-$\vee$-pattern and is chain definable, but not first-order definable. Thus we also correct a proposition in [18] stating that every aperiodic and chain definable tree language is already first-order definable.

Definition 25. Let $\Sigma = \Sigma_2 \cup \Sigma_0$ with $\Sigma_2 = \{\widetilde{=}, \leftrightarrow\}$ and $\Sigma_0 = \{0,1\}$. The binary symbols $\widetilde{=}$ and $\leftrightarrow$ denote partial boolean functions defined in the following tables:

$\overline{=}$	0	1	$\bot$
0	1	$\bot$	$\bot$
1	$\bot$	0	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$

$\leftrightarrow$	0	1	$\bot$
0	1	0	$\bot$
1	0	1	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$

Let $val(t) \in \{0, 1, \bot\}$ denote the value of a tree t. The set of all trees with value 1 is periodic, because the tree $\leftrightarrow(c, 0)$ is counting modulo two. Thus we have to forbid unlimited direct nestings of $\leftrightarrow$ in order to obtain an aperiodic tree language.

Let $T' = \left(\begin{array}{c} \overline{=} \\ / \quad \backslash \\ c \quad \leftrightarrow \\ \quad / \quad \backslash \\ \quad c \quad c \end{array} \right)^{*,c} \cdot^c \{0,1\}$ and let $T = T' \cap \{t \mid val(t) = 1\}$.

Theorem 26. *T is aperiodic, contains no $\wedge$-$\vee$-pattern and is chain definable.*

Proof. T is regular, because it is an intersection of a first-order definable tree language and a regular tree language. It is also easy to see that T is aperiodic and contains no $\wedge$-$\vee$-pattern because the leftmost path of a tree in T' consists of inner nodes only labelled $\overline{=}$ and already determines the value of the tree in the following sense: if this path is of even length then the value of the tree is $\bot$ or the value of the leftmost leaf, if the path is of odd length then the value of the tree is $\bot$ or the complement of the value of the leftmost leaf. The property of a path to be of even length can be easily expressed in chain logic. Thus we can construct a formula $\varphi_1(x)$ expressing that the leftmost path below node x determines the value of the subtree at node x to be 1 or $\bot$. Then we can define T by the following formula, where $\varphi_{T'}$ describes T':

$$\varphi_{T'} \wedge \exists x \forall x' \, (x \leq x' \wedge \varphi_1(x)) \wedge$$
$$\forall x \forall x_2 \forall x_{11} \forall x_{12} [P_{\overline{=}} x \wedge x S_2 x_2 \wedge x_2 S_1 x_{21} \wedge x_2 S_2 x_{22}] \Rightarrow$$
$$[(\varphi_1(x) \iff (\neg\varphi_1(x_{21}) \iff \varphi_1(x_{22}))].$$

Theorem 27. *[13] T is not first-order definable*

The proof of this theorem is rather technical and is therefore omitted.

6 Conclusion

The notions of tree homomorphisms and $\wedge$-$\vee$-patterns introduced here provide effective and powerful criteria for first-order definability of regular tree languages. Nevertheless the problem of deciding first-order definability remains unsolved.

We want to mention some other possibilities to investigate this problem. Whereas we decrease the alphabet of tree automata and increase the number of states in the proof of Theorem 17, one can also try the opposite direction, i.e. to define operations on tree automata that decrease the number of states

but preserve nondefinability of the accepted tree language. A more algebraic approach may result from the notion of tree language variety defined in [16] since the class of first-order definable tree languges build such a variety.

References

1. J.R.Büchi, *Finite automata, their algebras and grammars*, D.Siefkes, ed. (Springer Verlag, New York, 1989).
2. J.Doner, Tree acceptors and some of their applications, Journal of Computer and System Sciences **4** (1970) 406-451.
3. H.-D.Ebbinghaus, J.Flum, W.Thomas, *Mathematical Logic* (Springer Verlag, New York, Berlin, Heidelberg, Tokyo, 1984).
4. A.Ehrenfeucht, An application of games to the completeness problem for formalized theories, Fundamenta Mathematicae **49** (1961) 129-141.
5. J.Engelfriet, Bottom-up and top-down tree transformations - a comparison, Mathematical Systems Theory **9** (1975) 198-231.
6. R.Fraïssé, Sur quelques classifications des systèmes de relations, basés sur des isomorphismes restreints, Publications Scientifique de l'Université d'Alger, Série A **1** (1954) 35-182.
7. F.Gecség, M.Steinby, *Tree Automata* (Akadémiai Kiadó, Budapest, 1984).
8. U.Heuter, First-order properties of trees, star-free expressions and aperiodicity, RAIRO Theoretical Informatics and Applications **25** (1991) 125-145.
9. R.Ladner, Applications of model theoretic games to discrete linear orders and finite automata, Information and Control **33** (1977), 281-303.
10. R.McNaughton, S.Papert, *Counter-free Automata* (The M.I.T. Press, Cambridge, Massachusetts, 1971).
11. D.Perrin, Finite Automata, in: *Handbook of Theoretical Computer Science, Vol. B*, J.v.Leeuwen, ed. (Elsevier , Amsterdam, 1990) 3-57.
12. A.Potthoff, Modulo counting quantifiers over finite trees, TCS **126** (1994) 97-112.
13. A.Potthoff, Logische Klassifizierung regulärer Baumsprachen, (Dissertation, Kiel, 1994).
14. A.Potthoff, W.Thomas, Regular tree languages without unary symbols are star-free, in: *9th International Conference on Fundamentals of Computation Theory*, Zoltán Ésik, ed., Lecture Notes in Computer Science **710** (1993) 396-405.
15. M.P.Schützenberger, On monoids having only trivial subgroups, Information and Control **8** (1965) 190-194.
16. M.Steinby, A theory of tree language varieties, in *Tree automata and languages*, M.Nivat, A.Podelski, eds., Studies in Computer Science and Artifical Intelligence Vol. **10** (Elsevier, Amsterdam, 1992) 57-81.
17. J.W.Thatcher, J.B.Wright, Generalized finite automata theory with an application to a decision problem of second-order logic, Mathematical Systems Theory **2** (1968) 57-81.
18. W.Thomas, Logical aspects in the study of tree languages, in: *9th Colloquium on Trees in Algebra and Programming*, B.Courcelle, ed. (Cambridge University Press, Cambridge, 1984) 31-51.
19. W.Thomas, On the Ehrenfeucht-Fraïssé game in theoretical computer science, in: *18th Colloquium on Trees in Algebra and Programming*, M.C.Gaudel, J.P.Jouannaud, eds., Lecture Notes in Computer Science **668** (1993) 559-568.