Learning Cover Context-Free Grammars from Structural Data

Mircea Marin[1,⋆] and Gabriel Istrate[1,2]

[1] Department of Computer Science, West University of Timişoara,
Timişoara RO-300223, Romania
mmarin@info.uvt.ro
[2] e-Austria Research Institute,
Timişoara RO-300223, Romania

Abstract. We consider the problem of learning an unknown context-free grammar when the only knowledge available and of interest to the learner is about its structural descriptions with depth at most ℓ. The goal is to learn a *cover context-free grammar* (CCFG) with respect to ℓ, that is, a CFG whose structural descriptions with depth at most ℓ agree with those of the unknown CFG. We propose an algorithm, called LA^ℓ, that efficiently learns a CCFG using two types of queries: structural equivalence and structural membership. We show that LA^ℓ runs in time polynomial in the number of states of a minimal deterministic finite cover tree automaton (DCTA) with respect to ℓ. This number is often much smaller than the number of states of a minimum deterministic finite tree automaton for the structural descriptions of the unknown grammar.

Keywords: automata theory and formal languages, structural descriptions, grammatical inference.

1 Introduction

Angluin's approach to grammatical inference [1] is an important contribution to computational learning, with extensions to problems, such as compositional verification and synthesis [4,11], that go beyond the usual applications to natural language processing and computational biology [5].

Practical concerns, e.g. [9], seem to require going beyond regular languages to classes of languages with regular tree nature. However, Angluin and Kharitonov have shown that learning CFGs from membership and equivalence queries is intractable under plausible cryptographic assumptions [2]. A way out is to learn structural descriptions of context free languages. Sakakibara has shown that Angluin's algorithm extends to this setting [12]. His approach has applications in learning the structural descriptions of natural languages, which describe the shape of the parse trees of well chosen CFGs. Often, these structural descriptions are subject to additional restrictions arising from modelling considerations. For

[⋆] Corresponding author.

G. Ciobanu and D. Méry (Eds.): ICTAC 2014, LNCS 8687, pp. 241–258, 2014.

instance, in natural language understanding, the bounded memory restriction on human comprehension seems to limit the recursion depth of such a parse tree to a constant. A natural example with a similar flavour is the limitation imposed by the LaTeX system, that limits the number of nestings of itemised environments to a small constant.

Imposing such a restriction leads to the idea of learning cover languages, that is, languages that are accurate up to an equivalence. For regular languages modulo a finite prefix such an approach has been pursued by Ipate [8] (see also [6]).

In this paper, we extend this approach to context-free languages with structural descriptions. We propose an algorithm called LA^ℓ which asks two types of queries: structural equivalence and structural membership queries, both restricted to structural descriptions with depth at most ℓ, where ℓ is a constant. LA^ℓ stores the answers retrieved from the teacher in an *observation table* which is used to guide the learning protocol and to construct a minimal DCTA of the unknown context-free grammar with respect to ℓ. Our main result shows that LA^ℓ runs in time polynomial in n and m, where n is the number of states of a minimal DCTA of the unknown CFG with respect to ℓ, and m is the maximum size of a counterexample returned by a failed structural membership query.

The paper is structured as follows. Section 2 introduces the basic notions and results to be used later in the paper. It also describes algorithm LA. In Sect. 4 we introduce the main concepts related to the specification and analysis of our learning algorithm LA^ℓ. They are natural generalisations to languages of structural descriptions of the concepts proposed by Ipate [8] in the design and study of his algorithm L^ℓ. In Sect. 5 we analyse the space and time complexity of LA^ℓ and show that its time complexity is a polynomial in n and m, where n is the number of states of a minimal deterministic finite cover automaton w.r.t. ℓ of the language of structural descriptions of interest, and m is an upper bound to the size of counterexamples returned by failed structural equivalence queries.

2 Preliminaries

We write $\mathbb{N}$ for the set of nonnegative integers, A^* for the set of finite strings over a set A, and ϵ for the empty string. If $v, w \in A^*$, we write $v \leq w'$ if there exists $w' \in A^*$ such that $vw' = w$; $v < v'$ if $v \leq v'$ and $v \neq v'$; and $v \perp w$ if neither $v \leq w$ nor $w \leq v$.

Trees, Terms, Contexts, and Context-Free Grammars

A *ranked alphabet* is a finite set $\mathcal{F}$ of function symbols together with a finite *rank* relation $rk(\mathcal{F}) \subseteq \mathcal{F} \times \mathbb{N}$. We denote the subset $\{f \in \mathcal{F} \mid (f, m) \in rk(\mathcal{F})\}$ by $\mathcal{F}_m$, the set $\{m \mid (f, m) \in rk(\mathcal{F})\}$ by $ar(f)$, and $\bigcup_{f\in\mathcal{F}} ar(f)$ by $ar(\mathcal{F})$. The *terms* of the set $\mathcal{T}(\mathcal{F})$ are the strings of symbols defined recursively by the grammar $t ::= a \mid f(t_1, \ldots, t_m)$ where $a \in \mathcal{F}_0$ and $f \in \mathcal{F}_m$ with $m > 0$. The *yield* of a term $t \in \mathcal{T}(\mathcal{F})$ is the finite string $yield(t) \in \mathcal{F}_0^*$ defined as follows: $\mathit{yield}(a) := a$ if $a \in \mathcal{F}_0$, and $\mathit{yield}(f(t_1, \ldots, t_m)) := w_1 \ldots w_m$ where $w_i = \mathit{yield}(t_i)$ for $1 \leq i \leq m$.

A *finite ordered tree* over a set of labels $\mathcal{F}$ is a mapping t from a nonempty and prefix closed set $\mathit{Pos}(t) \subseteq (\mathbb{N} \setminus \{0\})^*$ into $\mathcal{F}$. Each element in $\mathit{Pos}(t)$ is called a *position.* The tree t is *ranked* if $\mathcal{F}$ is a ranked alphabet, and t satisfies the following additional property: For all $p \in \mathit{Pos}(t)$, there exists $m \in \mathbb{N}$ such that $\{i \in \mathbb{N} \mid pi \in \mathit{Pos}(t)\} = \{1, \ldots, m\}$ and $t(p) \in \mathcal{F}_m$.

Thus, any term $t \in \mathcal{T}(\mathcal{F})$ may be viewed as a finite ordered ranked tree, and we will refer to it by "tree" when we mean the finite ordered tree with the additional property mentioned above. The *depth* of t is $\mathtt{d}(t) := \max\{\|p\| \mid p \in \mathit{Pos}(t)\}$ where $\|p\|$ denotes the length of p as sequence of numbers. The *size* $\mathtt{sz}(t)$ of t is the number of elements of the set $\{p \in \mathit{Pos}(t) \mid \|p\| \neq \mathtt{d}(t)\}$, that is, the number of internal nodes of t.

The *subterm* $t|_p$ of a term t at position $p \in \mathit{Pos}(t)$ is defined by the following: $\mathit{Pos}(t|_p) := \{i \mid pi \in \mathit{Pos}(t)\}$, and $t|_p(p') := t(pp')$ for all $p' \in \mathit{Pos}(t|_p)$. We denote by $t[u]_p$ the term obtained by replacing in t the subterm $t|_p$ with u, that is: $\mathit{Pos}(t[u]_p) = (\mathit{Pos}(t) - \{pp' \mid p' \in \mathit{Pos}(t|_p)\}) \cup \{pp'' \mid p'' \in \mathit{Pos}(u)\}$, and

$$t[u]_p(p') := \begin{cases} u(p'') & \text{if } p' = pp'' \text{ with } p'' \in \mathit{Pos}(u), \\ t(p') & \text{otherwise.} \end{cases}$$

The set $\mathcal{C}(\mathcal{F})$ of *contexts* over $\mathcal{F}$ is the set of terms over $\mathcal{F} \cup \{\bullet\}$, where:

- $\bullet$ is a distinguished fresh symbol with $\mathit{ar}(\bullet) = \{0\}$, called *hole*,
- $\mathit{rk}(\mathcal{F} \cup \{\bullet\}) = \mathit{rk}(\mathcal{F}) \cup \{(\bullet, 0)\}$, and
- every element $C \in \mathcal{C}(\mathcal{F})$ contains only one occurrence of $\bullet$. This is the same as saying that $\{p \in \mathit{Pos}(C) \mid C(p) = \bullet\}$ is a singleton set.

If $C \in \mathcal{C}(\mathcal{F})$ and $u \in \mathcal{C}(\mathcal{F}) \cup \mathcal{T}(\mathcal{F})$ then $C[u]$ stands for the context or term $C[u]_p$, where $C(p) = \bullet$. The *hole depth* of a context $C \in \mathcal{C}(\mathcal{F})$ is $\mathtt{d}_\bullet(C) := \|p\|$ where p is the unique position of C such that $C(p) = \bullet$. From now on, whenever M is a set of terms, P is a set of contexts, and m is a non-negative integer, we define the sets $M_{[m]} := \{t \in M \mid \mathtt{d}(t) \leq m\}$ and $P_{\langle m \rangle} := \{C \in P \mid \mathtt{d}_\bullet(C) \leq m\}$.

We assume that the reader is acquainted with the notions of CFG and the context-free language $\mathcal{L}(G)$ generated by a CFG G, see, e.g., [13]. A CFG is ϵ-free if it has no productions of the form $X \to \epsilon$. It is well known [7] that every ϵ-free context-free language L (that is, $\epsilon \notin L$) is generated by an ϵ-free CFG. The derivation trees of an ϵ-free CFG $G = (N, \Sigma, P, S)$ correspond to terms from $\mathcal{T}(N \cup \Sigma)$ with $\mathit{ar}(a) = \{0\}$ for al $a \in \Sigma$ and $\mathit{ar}(X) = \{m \mid \exists(X \to \alpha) \in P$ with $\|\alpha\| = m\}$ for all $X \in N$. The sets $D_G(U)$ of derivation trees issued from $U \in N \cup \Sigma$, and $D(G)$ of derivation trees of G, are defined recursively as follows:

$$D_G(a) := \{a\} \text{ if } a \in \Sigma,$$

$$D_G(X) := \bigcup_{(X \to U_1 \ldots U_m) \in P} \{X(t_1, \ldots, t_m) \mid t_1 \in D_G(U_1) \wedge \ldots \wedge t_m \in D_G(U_m)\},$$

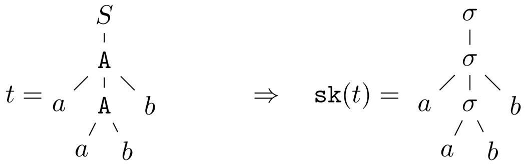
$D(G) := D_G(S)$. Note that $\mathcal{L}(G) = \{\mathit{yield}(t) \mid t \in D(G)\}$.

Structural Descriptions and Cover Context-Free Grammars

A *skeletal alphabet* is a ranked alphabet $Sk = \{\sigma\}$, where σ is a special symbol with $ar(\sigma)$ a finite subset of $\mathbb{N} \setminus \{0\}$, and a *skeletal set* is a ranked alphabet $Sk \cup A$ where $Sk \cap A = \emptyset$ and $ar(a) = \{0\}$ for all $a \in A$. Skeletal alphabets are intended to describe the structures of the derivation trees of ϵ-free CFGs. For an ϵ-free CFG $G = (N, \Sigma, P, S)$ we consider the skeletal alphabet Sk with $ar(\sigma) := \{\|\alpha\| \mid (X \to \alpha) \in P\}$, and the skeletal set $Sk \cup \Sigma$. The *skeletal* (or *structural*) *description* of a derivation tree $t \in D_G(U)$ is the term $\mathtt{sk}(t) \in \mathcal{T}(Sk \cup \Sigma)$ where

$$\mathtt{sk}(t) := \begin{cases} a & \text{if } t = a \in \Sigma, \\ \sigma(\mathtt{sk}(t_1), \ldots, \mathtt{sk}(t_m)) & \text{if } t = X(t_1, \ldots, t_m) \text{ with } m > 0. \end{cases}$$

For example, if G is the grammar $(\{\mathtt{S}, \mathtt{A}\}, \{a, b\}, \{\mathtt{S} \to \mathtt{A}, \mathtt{A} \to a\mathtt{A}b, \mathtt{A} \to ab\}, \mathtt{S})$ then $t = \mathtt{S}(\mathtt{A}(a, \mathtt{A}(a, b), b)) \in D_G(\mathtt{S})$ and $\mathtt{sk}(t) = \sigma(\sigma(a, \sigma(a, b), b)) \in \mathcal{T}(\{\sigma, a, b\})$, where $ar(\sigma) = \{1, 2, 3\}$ and $ar(a) = ar(b) = \{0\}$. Graphically, we have



If M is a set of ranked trees, then the set of its structural descriptions is $K(M) := \{\mathtt{sk}(t) \mid t \in M\}$. Two context-free grammars G_1 and G_2 over the same alphabet of terminals are *structurally equivalent* if $K(D(G)) = K(D(G'))$.

Definition 1 (cover CFG). *Let ℓ be a positive integer and G_U be an ϵ-free CFG of a language $U \subseteq \Sigma^*$. A* cover context-free grammar *of G_U with respect to ℓ is an ϵ-free CFG G' such that $K(D(G'))_{[\ell]} = K(D(G_U))_{[\ell]}$.*

Tree Automata

The definition of tree automaton presented here is equivalent with that given in [12]. It is non-standard in the sense that it cannot accept any tree of depth 0.

Definition 2. *A* nondeterministic (bottom-up) finite tree automaton *(NFTA) over $\mathcal{F}$ is a quadruple $\mathcal{A} = (\mathcal{Q}, \mathcal{F}, \mathcal{Q}_{\mathtt{f}}, \Delta)$ where $\mathcal{Q}$ is a finite set of* states*, $\mathcal{Q}_{\mathtt{f}} \subseteq \mathcal{Q}$ is the set of* final states*, and Δ is a set of* transition rules *of the form $f(q_1, \ldots, q_m) \to q$ where $m \geq 1$, $f \in \mathcal{F}_m$, $q_1, \ldots, q_m \in \mathcal{F}_0 \cup \mathcal{Q}$, and $q \in \mathcal{Q}$.*

Such an automaton $\mathcal{A}$ induces a *move* relation $\to_{\mathcal{A}}$ on the set of terms $\mathcal{T}(\mathcal{F} \cup \mathcal{Q})$ where $ar(q) = \{0\}$ for all $q \in \mathcal{Q}$, as follows:

> $t \to_{\mathcal{A}} t'$ if there exist $C \in \mathcal{C}(\mathcal{F} \cup \mathcal{Q})$ and $f(q_1, \ldots, q_m) \to q \in \Delta$ such that $t = C[f(q_1, \ldots, q_m)]$ and $t' = C[q]$.

The *language accepted by* $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}) := \{t \in \mathcal{T}(\mathcal{F}) \mid t \rightarrow^*_{\mathcal{A}} q$ for some $q \in \mathcal{Q}_\mathtt{f}\}$ where $\rightarrow^*_{\mathcal{A}}$ is the reflexive-transitive closure of $\rightarrow_{\mathcal{A}}$. In this paper, a *regular tree language* is a language accepted by such an NFTA. Two NFTAs are *equivalent* if they accept the same language.

$\mathcal{A} = (\mathcal{Q}, \mathcal{F}, \mathcal{Q}_\mathtt{f}, \Delta)$ is *deterministic* (DFTA) if the transition rules of Δ describe a mapping δ which assigns to every $m \in ar(\mathcal{F})$ a function δ_m such that $\delta_0 : \mathcal{F}_0 \rightarrow \mathcal{F}_0$, $\delta_0(a) = a$ for all $a \in \mathcal{F}_0$, and $\delta_m : \mathcal{F}_m \rightarrow (\mathcal{F}_0 \cup \mathcal{Q})^m \rightarrow \mathcal{Q}$ if $m > 0$. This implies that $f(q_1, \ldots, q_m) \rightarrow q \in \Delta$ if and only if $\delta_m(f)(q_1, \ldots, q_m) = q$. The extension δ^* of $\{\delta_m \mid m \in ar(\mathcal{F})\}$ to $\mathcal{T}(\mathcal{F})$ is defined as expected: $\delta^*(a) = a$ if $a \in \mathcal{F}_0$, and $\delta^*(f(t_1, \ldots, t_m)) := \delta_m(f)(\delta^*(t_1), \ldots, \delta^*(t_m))$ otherwise. Note that, if $\mathcal{A}$ is a DFTA then $\mathcal{L}(\mathcal{A}) = \{t \in \mathcal{T}(\mathcal{F}) \mid \delta^*(t) \in \mathcal{Q}_\mathtt{f}\}$.

Two DFTAs $\mathcal{A}_1 = (\mathcal{Q}, \mathcal{F}, \mathcal{Q}_\mathtt{f}, \delta)$ and $\mathcal{A}_2 = (\mathcal{Q}', \mathcal{F}, \mathcal{Q}'_\mathtt{f}, \delta')$ are *isomorphic* if there exists a bijection $\varphi : \mathcal{Q} \rightarrow \mathcal{Q}'$ such that $\varphi(\mathcal{Q}_\mathtt{f}) = \mathcal{Q}'_\mathtt{f}$ and for every $f \in \mathcal{F}_m$, $q_1, \ldots, q_m \in \mathcal{F}_0 \cup \mathcal{Q}$, $\varphi(\delta_m(f)(q_1, \ldots, q_m)) = \delta'_m(f)(\varphi(q_1), \ldots, \varphi(q_m))$. A *minimum DFTA* of a regular tree language $L \subseteq \mathcal{T}(\mathcal{F}) \setminus \mathcal{F}_0$ is a DFTA $\mathcal{A}$ with minimum number of states such that $\mathcal{L}(A) = L$.

There is a strong correspondence between tree automata and ϵ-free CFGs. The NFTA corresponding to an ϵ-free CFG $G = (N, \Sigma, P, S)$ is $NA(G) = (N, Sk \cup \Sigma, \{S\}, \Delta)$ with $\Delta := \{\sigma(U_1, \ldots, U_m) \rightarrow X \mid (X \rightarrow U_1 \ldots U_m) \in P\}$. Conversely, the ϵ-free CFG corresponding to an NFTA $\mathcal{A} = (\mathcal{Q}, Sk \cup \Sigma, \mathcal{Q}_\mathtt{f}, \Delta)$ over the skeletal set $Sk \cup \Sigma$ is $G(\mathcal{A}) = (\mathcal{Q} \cup \{S\}, \Sigma, P, S)$ where S is a fresh symbol and $P := \{q \rightarrow q_1 \ldots q_m \mid (\sigma(q_1, \ldots, q_m) \rightarrow q) \in \Delta\} \cup \{S \rightarrow q_1 \ldots q_m \mid (\sigma(q_1, \ldots, q_m) \rightarrow q) \in \Delta$ with $q \in \mathcal{Q}_\mathtt{f}\}$. These constructs are dual to each other, in the following sense:

(A_1) If G is an ϵ-free CFG then $\mathcal{L}(NA(G)) = K(D(G))$. [12, Prop. 3.4]

(A_2) If $\mathcal{A} = (\mathcal{Q}, Sk \cup \Sigma, \mathcal{Q}_\mathtt{f}, \Delta)$ is an NFTA for the skeletal set $Sk \cup \Sigma$ then $K(D(G(\mathcal{A}))) = \mathcal{L}(\mathcal{A})$. That is, the set of structural descriptions of $G(\mathcal{A})$ coincides with the set of trees accepted by $\mathcal{A}$. [12, Prop. 3.6]

We recall the following well-known results: every NFTA is equivalent to an DFTA [10], and every two minimal DFTAs are isomorphic [3].

Cover Tree Automata

Definition 3 (DCTA). *Let $\ell \in \mathbb{N}^+$ and A be a tree language over the ranked alphabet $\mathcal{F}$. A* deterministic cover tree automaton *(DCTA) of A with respect to ℓ is a DFTA $\mathcal{A}$ over a skeletal set $Sk \cup \mathcal{F}_0$ such that $\mathcal{L}(\mathcal{A})_{[\ell]} = K(A)_{[\ell]}$.*

The correspondence between tree automata and ϵ-free CFGs is carried over to a correspondence between cover tree automata and cover CFGs. More precisely, it can be shown that if G_U is an ϵ-free CFG, then a DFTA $\mathcal{A}$ is a DCTA of $K(D(G_U))$ w.r.t. ℓ if and only if $G(\mathcal{A})$ is a cover CFG of G_U w.r.t. ℓ.

3 Learning Context-Free Grammars

In [12], Sakakibara assumes a *learner* eager to learn a CFG which is structurally equivalent with the CFG G_U of an unknown context-free language $U \subseteq \Sigma^*$ by asking questions to a *teacher*. We assume that the learner and the teacher share the skeletal set $Sk \cup \Sigma$ for the structural descriptions in $K(D(G_U))$. The learner can pose the following types of queries:

1. *Structural membership queries*: the learner asks if some $s \in \mathcal{T}(Sk \cup \Sigma)$ is in $K(D(G_U))$. The answer is *yes* if so, and *no* otherwise.
2. *Structural equivalence queries*: The learner proposes a CFG G' and asks whether G' is structurally equivalent to G_U. If the answer is *yes*, the process stops with the learned answer G. Otherwise, the teacher provides a counterexample s from the symmetric set difference $K(D(G')) \triangle K(D(G_U))$.

This learning protocol is based on what is called *minimal adequate teacher* in [1]. Ultimately, the learner constructs a minimal DFTA $\mathcal{A}$ of $K(D(G_U))$ from which it can infer immediately the CFG $G' = G(\mathcal{A})$ which is structurally equivalent to G_U, that is, $K(D(G')) = K(D(G_U))$. In order to understand how $\mathcal{A}$ gets constructed, we shall introduce a few auxiliary notions.

For any subset S of $\mathcal{T}(Sk \cup \Sigma)$, we define the sets

$$\begin{aligned} \sigma_\bullet\langle S\rangle &:= \bigcup_{m\in ar(\sigma)} \bigcup_{i=1}^{m} \{\sigma(s_1,\ldots,s_m)[\bullet]_i \mid s_1,\ldots,s_m \in S\cup\Sigma\},\\ X(S) &:= \{C_1[s] \mid C_1 \in \sigma_\bullet\langle S\rangle, s \in S\cup\Sigma\} \setminus S. \end{aligned}$$

Note that $\sigma_\bullet\langle S\rangle = \{C \in \mathcal{C}(Sk \cup \Sigma) \setminus \{\bullet\} \mid C|_p \in S \cup \Sigma \cup \{\bullet\}$ for all $p \in Pos(C) \cap \mathbb{N}\}$.

Definition 4. *A subset E of $\mathcal{C}(Sk \cup \Sigma)$ is* **$\bullet$-prefix closed** *with respect to a set $S \subseteq \mathcal{T}(Sk \cup \Sigma)$ if $C \in E \setminus \{\bullet\}$ implies the existence of $C' \in E$ and $C_1 \in \sigma_\bullet\langle S\rangle$ such that $C = C'[C_1]$. If $E \subseteq \mathcal{C}(Sk \cup \Sigma)$ and $S \subseteq \mathcal{T}(Sk \cup \Sigma)$ then $E[S]$ denotes the set of structural descriptions defined by $E[S] = \{C[s] \mid C \in E, s \in S\}$.*

We say that $S \subseteq \mathcal{T}(Sk \cup \Sigma)$ is **subterm closed** *if $\mathtt{d}(s) \geq 1$ for all $s \in S$, and $s' \in S$ whenever s' is a subterm of some $s \in S$ with $\mathtt{d}(s') \geq 1$.*

An *observation table* for $K(D(G_U))$, denoted by (S, E, T), is a tabular representation of the finitary function $T : E[S \cup X(S)] \to \{0, 1\}$ defined by $T(t) := 1$ if $t \in K(D(G_U))$, and 0 otherwise, where S is a finite nonempty subterm closed subset S of $\mathcal{T}(Sk \cup \Sigma)$, and E is a finite nonempty subset of $\mathcal{C}(Sk \cup \Sigma)$ which is $\bullet$-prefix closed with respect to S. Such an observation table is visualised as a matrix with rows labeled by elements from $S \cup X(S)$, columns labeled by elements from E, and the entry for row of s and column of C equal to $T(C[s])$. If we fix a listing $\langle C_1, \ldots, C_r\rangle$ of all elements of E, then the row of values of some $s \in S \cup X(S)$ corresponds to the vector $row(s) = \langle T(C_1[s]), \ldots, T(C_r[s])\rangle$. In fact, for every such s, $row(s)$ is a finitary representation of the function $f_s : E \to \{0, 1\}$ defined by $f_s(C) = T(C[s])$.

The observation table (S, E, T) is *closed* if every $row(x)$ with $x \in X(S)$ is identical to some $row(s)$ of $s \in S$. It is *consistent* if whenever $s_1, s_2 \in S$ such that $row(s_1) = row(s_2)$, we have $row(C_1[s_1]) = row(C_1[s_2])$ for all $C_1 \in \sigma_\bullet\langle S\rangle$.

The DFTA *corresponding to a closed and consistent observation table* (S, E, T) is $\mathcal{A}(S, E, T) = (\mathcal{Q}, Sk \cup \Sigma, \mathcal{Q}_{\mathtt{f}}, \delta)$ where $\mathcal{Q} := \{row(s) \mid s \in S\}$, $\mathcal{Q}_{\mathtt{f}} := \{row(s) \mid s \in S \text{ and } T(s) = 1\}$, and δ is uniquely defined by

$$\delta_m(\sigma)(q_1, \ldots, q_m) := row(\sigma(r_1, \ldots, r_m)) \quad \text{for all } m \in ar(\sigma),$$

where $r_i := a$ if $q_i = a \in \Sigma$, and $r_i := s_i$ if $q_i = row(s_i) \in \mathcal{Q}$.

It is easy to check that, under these assumptions, $\mathcal{A}(S, E, T)$ is well-defined, and that $\delta^*(s) = row(s)$. Furthermore, Sakakibara proved that the following properties hold whenever (S, E, T) is a closed and consistent observation table:

1. $\mathcal{A}(S, E, T)$ is consistent with T, that is, for all $s \in S \cup X(S)$ and $C \in E$ we have $\delta^*(C[s]) \in \mathcal{Q}_{\mathtt{f}}$ iff $T(C[s]) = 1$. [12, Lemma 4.2]
2. If $\mathcal{A}(S, E, T) = (\mathcal{Q}, Sk \cup \Sigma, \delta, \mathcal{Q}_{\mathtt{f}})$ has n states, and $\mathcal{A}' = (\mathcal{Q}', Sk \cup \Sigma, \delta', \mathcal{Q}'_{\mathtt{f}})$ is any DFTA consistent with T that has n or fewer states, then $\mathcal{A}'$ is isomorphic to $\mathcal{A}(S, E, T)$. [12, Lemma 4.3]

The *LA* Algorithm

In this subsection we briefly recall Sakakibara's algorithm LA. *LA* extends the observation table whenever one of the following situations occurs: the table is not consistent, the table is not closed, or the table is both consistent and closed but the CFG corresponding to the resulting automaton $\mathcal{A}(S, E, T)$ is not structurally equivalent to G_U (in which case a counterexample is produced). The first two situations trigger an extension of the observation table with one distinct row. From properties (A_1) and (A_2), if n is the number of states of the minimum bottom-up tree automaton for the structural descriptions of G_U, then the number of unsuccessful consistency and closedness checks during the whole run of this algorithm is at most $n - 1$. For each counterexample of size at most m returned by a structural equivalence query, at most m subtrees are added to S. Since the algorithm encounters at most n counterexamples, the total number of elements in S cannot exceed $n + m \cdot n$, thus *LA* must terminate. It also follows that the number of elements of the domain $E[S \cup X(S)]$ of the function T is at most $(n + m \cdot n + (l + m \cdot n + k)^d) \cdot n = O(m^d \cdot n^{d+1})$, where l is the number of distinct ranks of $\sigma \in Sk$, and d is the maximum rank of a symbol in Sk. A careful analysis of *LA* reveals that its time complexity is indeed bounded by a polynomial in m and n [12, Thm. 5,3].

4 Learning Cover Context-Free Grammars

We assume we are given a teacher who knows an ϵ-free CFG G_U for a language $U \subseteq \Sigma^*$, and a learner who knows the skeletal set $Sk \cup \Sigma$ for $K(D(G_U))$. The teacher and learner both know a positive integer ℓ, and the learner is interested to learn a cover CFG G' of G_U w.r.t. ℓ or, equivalently, a cover DCTA of $K(D(G_U))$ w.r.t. ℓ. The learner is allowed to pose the following types of questions:

1. *Structural membership queries*: the learner asks if some $s \in \mathcal{T}(Sk \cup \Sigma)_{[\ell]}$ is in $K(D(G_U))$. The answer is *yes* if so, and *no* otherwise.
2. *Structural equivalence queries*: The learner proposes a CFG G', and asks if G' is a cover CFG of G_U w.r.t. ℓ. If the answer is yes, the process stops with the learned answer G'. Otherwise, the teacher provides a counterexample from the set $(K(D(G_U))_{[\ell]} - K(D(G'))) \cup (K(D(G'))_{[\ell]} - K(D(G_U)))$.

We will describe an algorithm LA^ℓ that learns a cover CFG of G_U with respect to ℓ in time that is polynomial in the number of states of a minimal DCTA of the rational tree language $K(D(G_U))$.

4.1 The Observation Table

LA^ℓ is a generalisation of the learning algorithm L^ℓ proposed by Ipate [8]. Ipate's algorithm is designed to learn a minimal finite cover automaton of an unknown finite language of words in polynomial time, using membership queries and language equivalence queries that refer to words and languages of words with length at most ℓ. Similarly, LA^ℓ is designed to learn a minimal DCTA $\mathcal{A}'$ for $K(D(G_U))$ with respect to ℓ by maintaining an observation table (S, E, T, ℓ) for $K(D(G_U))$ which differs from the observation table of LA in the following respects:

1. S is a finite nonempty subterm closed subset of $\mathcal{T}(Sk \cup \Sigma)_{[\ell]}$.
2. E is a finite nonempty subset of $\mathcal{C}(Sk \cup \Sigma)_{\langle \ell-1\rangle} \cap \mathcal{C}(Sk \cup \Sigma)_{[\ell]}$ which is $\bullet$-prefix closed with respect to S.
3. $T : E[S \cup X(S)_{[\ell]}] \to \{1, 0, -1\}$ is defined by

$$T(t) := \begin{cases} 1 & \text{if } t \in K(D(G_U))_{[\ell]}, \\ 0 & \text{if } t \in \mathcal{T}(Sk \cup \Sigma)_{[\ell]} \setminus K(D(G_U)), \\ -1 & \text{if } t \notin \mathcal{T}(Sk \cup \Sigma)_{[\ell]}. \end{cases}$$

In a tabular representation, the observation table (S, E, T, ℓ) is a two-dimensional matrix with rows labeled by elements from $S \cup X(S)_{[\ell]}$, columns labeled by elements from E, and the entry corresponding to the row of t and column of C equal to $T(C[t])$. If we fix a listing $\langle C_1, \ldots, C_k\rangle$ of all elements from E, then the row of t in the observation table is described by the vector $\langle T(C_1[t]), \ldots, T(C_k[t])\rangle$ of values from $\{-1, 0, 1\}$. The rows of an observation table are used to identify the states a a minimal DCTA for $K(D(G_U))$ with respect to ℓ. But, like Ipate [8], we do not compare rows by equality but by a similarity relation.

4.2 The Similarity Relation

This time, the rows in the observation table correspond to terms from $S \cup X(S)_{[\ell]}$, and the comparison of rows should take into account only terms of depth at most ℓ. For this purpose, we define a relation $\sim_k$ of *k-similarity*, which is a generalisation to terms of Ipate's relation of k-similarity on strings [8].

Definition 5 (k-similarity). *For $1 \leq k \leq \ell$ we define the relation $\sim_k$ on the elements of the set $S \cup X(S)$ of an observation table (S, E, T, ℓ) as follows:*

$$s \sim_k t \text{ if, for every } C \in E_{\langle k - \max\{\mathtt{d}(s), \mathtt{d}(t)\}\rangle},\ T(C[s]) = T(C[t]).$$

When the relation $\sim_k$ does not hold between two terms $s, t \in S \cup X(S)$, we write $s \nsim_k t$ and say that s and t are k-dissimilar. *When $k = \ell$ we simply say that s and t are* similar *or* dissimilar *and write $s \sim t$ or $s \nsim t$, respectively.*

We say that a context C **ℓ-distinguishes** *s_1 and s_2, where $s_1, s_2 \in S$, if $C \in E_{\langle \ell - \max\{\mathtt{d}(s_1), \mathtt{d}(s_2)\}\rangle}$ and $T(C[s_1]) \neq T(C[s_2])$.*

Note that only the contexts $C \in E_{\langle k - \max\{\mathtt{d}(s), \mathtt{d}(t)\}\rangle}$ with $\mathtt{d}(C) \leq \ell$ are relevant to check whether $s \sim_k t$, because if $\mathtt{d}(C) > \ell$ then $\mathtt{d}(C[s]) > \ell$ and $\mathtt{d}(C[t]) > \ell$, and therefore $T(C[s]) = -1 = T(C[t])$. Also, if $t \in S \cup X(S)$ with $\mathtt{d}(t) > \ell$ then it must be the case that $t \in X(S)$, and then $t \sim_k s$ for all $s \in S \cup X(S)$ and $1 \leq k \leq \ell$ because $E_{\langle k - \max \mathtt{d}(t), \mathtt{d}(s)\}\rangle} = \emptyset$.

The relation of k-similarity is obviously reflexive and symmetric, but not transitive. The following example illustrates this fact.

Example 1. Let $\Sigma = \{a, b\}$, $k = 1$, $\ell = 2$, $S = \{\sigma(a), \sigma(b), \sigma(\sigma(a), b)\}$, $E = \{\bullet, \sigma(\bullet, b)\}$, $t_1 = \sigma(a)$, $t_2 = \sigma(\sigma(a), b)$, $t_3 = \sigma(b)$, and

$$G_U = (\{\mathtt{S}, \mathtt{A}\}, \{a, b\}, \{\mathtt{S} \rightarrow a, \mathtt{S} \rightarrow b, \mathtt{S} \rightarrow \mathtt{A}b, \mathtt{A} \rightarrow a, \mathtt{A} \rightarrow \mathtt{A}b\}, \mathtt{S}).$$

S is a nonempty subterm closed subset of $\mathcal{T}(Sk \cup \Sigma)_{[\ell]}$, and E is a nonempty subset of $\mathcal{C}(Sk \cup \Sigma)_{\langle \ell - 1\rangle}$ which is $\bullet$-prefix closed with respect to S. We have $K(D(G_U))_{[\ell]} = \{t_1, t_2, t_3\}$, $t_1 \sim_\ell t_2$ because $E_{\langle \ell - \max\{\mathtt{d}(t_1), \mathtt{d}(t_2)\}\rangle} = \{\bullet\}$ and $T(\bullet[t_1]) = 1 = T(\bullet[t_2])$, and $t_2 \sim_\ell t_3$ because $E_{\langle \ell - \max\{\mathtt{d}(t_2), \mathtt{d}(t_3)\}\rangle} = \{\bullet\}$ and $T(\bullet[t_2]) = 1 = T(\bullet[t_3])$, However, $t_1 \nsim_\ell t_3$ because $C = \sigma(\bullet, b) \in E_{\langle 1\rangle} = E_{\langle \ell - \max\{\mathtt{d}(t_1), \mathtt{d}(t_3)\}\rangle}$ and $T(C[t_1]) = T(\sigma(\sigma(a), b)) = T(t_2) = 1$, but $T(C[t_3]) = T(\sigma(\sigma(b), b)) = 0$. □

Still, k-similarity has a useful property, captured in the following lemma.

Lemma 1. *Let (S, E, T, ℓ) be an observation table. If $s, t, x \in S \cup X(S)$ such that $\mathtt{d}(x) \leq \max\{\mathtt{d}(s), \mathtt{d}(t)\}$, then $s \sim_k t$ whenever $s \sim_k x$ and $x \sim_k t$.*

In addition, we will also assume a total order $\prec$ on the alphabet Σ, and the following total orders induced by $\prec$ on $\mathcal{T}(Sk \cup \Sigma)$ and $\mathcal{C}(Sk \cup \Sigma)$.

Definition 6. *The total order $\prec_{\mathtt{T}}$ on $\mathcal{T}(Sk \cup \Sigma)$ induced by a total order $\prec$ on Σ is defined as follows: $s \prec_{\mathtt{T}} t$ if either (a) $\mathtt{d}(s) < \mathtt{d}(t)$, or (b) $\mathtt{d}(s) = \mathtt{d}(t)$ and*

1. *$s, t \in \Sigma$ and $s \prec t$, or else*
2. *$s \in \Sigma$ and $t \notin \Sigma$, or else*
3. *$s = \sigma(s_1, \ldots, s_m)$, $t = \sigma(t_1, \ldots, t_n)$ and there exists $1 \leq k \leq \min(m, n)$ such that $s_k \prec_{\mathtt{T}} t_k$ and $s_i = t_i$ for all $1 \leq i < k$, or else*
4. *$s = \sigma(s_1, \ldots, s_m)$ and $t = \sigma(t_1, \ldots, t_n)$, $m < n$, and $s_i = t_i$ for $1 \leq i \leq m$.*

The total order $\prec_{\mathtt{C}}$ on $\mathcal{C}(Sk \cup \Sigma)$ induced by a total order $\prec$ on Σ is defined as follows: $C_1 \prec_{\mathtt{C}} C_2$ if either (a) $\mathtt{d}_\bullet(C_1) < \mathtt{d}_\bullet(C_2)$, or (b) $\mathtt{d}_\bullet(C_1) = \mathtt{d}_\bullet(C_2)$ and $C_1 \prec_{\mathtt{T}} C_2$ where C_1, C_2 are interpreted as terms over the signature with Σ extended with the constant $\bullet$ such that $\bullet \prec a$ for all $a \in \Sigma$.

Definition 7 (Representative). *Let (S, E, T, ℓ) be an observation table and $x \in S \cup X(S)$. We say x* has a representative *in S if $\{s \in S \mid s \sim x\} \neq \emptyset$. If so, the* representative *of x is $\mathtt{r}(x) := \min_{\prec_{\mathrm{T}}}\{s \in S \mid x \sim s\}$.*

We will show later that the construction of an observation table (S, E, T, ℓ) is instrumental to the construction of a cover tree automaton, and the states of the automaton correspond to representatives of the elements from $S \cup X(S)$. Note that, if (S, E, T, ℓ) is an observation table and $x \in S \cup X(S)$ has $\mathtt{d}(x) > \ell$ then $x \in X(S)$ and $x \sim s$ for all $s \in S$. Then $s \prec_{\mathrm{T}} x$ because $\mathtt{d}(s) \leq \ell < \mathtt{d}(x)$ for all $s \in S$. Thus x has a representative in S, and $\mathtt{r}(x) = \min_{\prec_{\mathrm{T}}} S$. For this reason, only the rows for elements $x \in S \cup X(S)_{[\ell]}$ are kept in an observation table.

4.3 Consistency and Closedness

The consistency and closedness of an observation table are defined as follows.

Definition 8 (Consistency). *An observation table (S, E, T, ℓ) is consistent if, for every $k \in \{1, \ldots, \ell\}$, $s_1, s_2 \in S$, and $C_1 \in \sigma_{\bullet}\langle S\rangle$, the following implication holds: If $s_1 \sim_k s_2$ then $C_1[s_1] \sim_k C_1[s_2]$.*

The following lemma captures a useful property of consistent observation tables.

Lemma 2. *Let (S, E, T, ℓ) be a consistent observation table. Let $m \in ar(\sigma)$, $1 \leq k \leq \ell$, and $s_1, \ldots, s_m, t_1, \ldots, t_m \in S \cup \Sigma$ such that, for all $1 \leq i \leq m$, either $s_i = t_i \in \Sigma$, or $s_i, t_i \in S$, $s_i \sim_k t_i$, and $\mathtt{d}(s_i) \leq \mathtt{d}(t_i)$, and $s = \sigma(s_1, \ldots, s_m)$, $t = \sigma(t_1, \ldots, t_m)$. Then $s \sim_k t$.*

Definition 9 (Closedness). *An observation table (S, E, T, ℓ) is closed if, for all $x \in X(S)$, there exists $s \in S$ with $\mathtt{d}(s) \leq \mathtt{d}(x)$ such that $x \sim s$.*

The next five lemmata capture important properties of closed observation tables, which will be used to justify the correctness of the learning algorithm we are about to introduce.

Lemma 3. *If (S, E, T, ℓ) is closed then every $x \in S \cup X(S)$ has a representative, and $\mathtt{d}(\mathtt{r}(x)) \leq \mathtt{d}(x)$.*

Lemma 4. *If (S, E, T, ℓ) is closed, $r_1, r_2 \in \{\mathtt{r}(x) \mid x \in S \cup X(S)\}$, and $r_1 \sim r_2$ then $r_1 = r_2$.*

Lemma 5. *If (S, E, T, ℓ) is closed and $r \in \{\mathtt{r}(x) \mid x \in S \cup X(S)\}$, then $\mathtt{r}(r) = r$.*

Proof. Let $r_1 = \mathtt{r}(r)$. Then $r_1 \sim r$ and $r_1, r \in \{\mathtt{r}(x) \mid x \in S \cup X(S)\}$. By Lemma 4, $r = r_1$. □

Lemma 6. *If (S, E, T, ℓ) is closed, then for every $x \in S \cup X(S)$ and $C_1 \in \sigma_{\bullet}\langle S\rangle$, there exists $s \in S$ such that $\mathtt{r}(C_1[\mathtt{r}(x)]) = \mathtt{r}(s)$.*

Lemma 7. *Let (S, E, T, ℓ) be closed, $r \in \{\mathtt{r}(x) \mid x \in S \cup X(S)\}$, $C_1 \in \sigma_{\bullet}\langle S\rangle$, and $s \in S$. If $C_1[s] \sim r$ then $\mathtt{d}(r) \leq \mathtt{d}(C_1[s])$.*

The Automaton $\mathcal{A}(\mathbb{T})$

Like L^ℓ, our algorithm relies on the construction of a consistent and closed observation table of the unknown context-free grammar. The table is used to build an automaton which, in the end, turns out to be a minimal DCTA for the structural descriptions of the unknown grammar.

Definition 10. *Suppose $\mathbb{T} = (S, E, T, \ell)$ is a closed and consistent observation table. The automaton corresponding to this table, denoted by $\mathcal{A}(\mathbb{T})$, is the DFTA $(\mathcal{Q}, Sk \cup \Sigma, \mathcal{Q}_{\mathtt{f}}, \delta)$ where $\mathcal{Q} := \{\mathtt{r}(s) \mid s \in S\}$, $\mathcal{Q}_{\mathtt{f}} = \{q \in \mathcal{Q} \mid T(q) = 1\}$, and δ is uniquely defined by $\delta_m(\sigma)(q_1, \ldots, q_m) := \mathtt{r}(\sigma(q_1, \ldots, q_m))$ for all $m \in ar(\sigma)$.*

The transition function δ is well defined because, for all $m \in ar(\sigma)$ and $q_1, \ldots, q_m$ from $\mathcal{Q}$, $C_1 := \sigma(\bullet, q_2, \ldots, q_m) \in \sigma_\bullet\langle S\rangle$, thus $\sigma(q_1, \ldots, q_m) = C_1[q_1] \in S \cup X(S)$ and $\mathtt{r}(C_1[q_1]) = \mathtt{r}(s)$ for some $s \in S$, by Lemma 6. Hence, $\mathtt{r}(\sigma(q_1, \ldots, q_m)) \in \mathcal{Q}$. Also, the set $\mathcal{Q}_{\mathtt{f}}$ can be read off directly from the observation table because $\bullet \in E$ (since E is $\bullet$-prefix closed), thus $q = \bullet[q] \in E[(S \cup X(S)_{[\ell]}]$ for all $q \in \mathcal{Q}$, and we can read off from the observation table all $q \in \mathcal{Q}$ with $T(q) = 1$.

In the rest of this subsection we assume that $\mathbb{T} = (S, E, T, \ell)$ is closed and consistent, and δ is the transition function of the corresponding DFTA $\mathcal{A}(\mathbb{T})$.

Lemma 8. *$\delta^*(x) \sim x$ and $\mathtt{d}(\delta^*(x)) \leq \mathtt{d}(x)$ for every $x \in S \cup X(S)$.*

Corollary 1. *$\delta^*(x) = x$ for all $x \in \{\mathtt{r}(s) \mid s \in S \cup X(S)\}$.*

Proof. By Lemma 8, $x \sim \delta^*(x)$. Since both $\delta^*(x)$ and x belong to the set of representatives $\{\mathtt{r}(s) \mid s \in X \cup X(S)\}$, $x = \delta^*(x)$ by Lemma 4. □

The following theorem shows that the DFTA of a closed and consistent observation table is consistent with the function T on terms with depth at most ℓ.

Theorem 1. *Let $\mathbb{T} = (S, E, T, \ell)$ be a closed and consistent observation table. For every $s \in S \cup X(S)$ and $C \in E$ such that $\mathtt{d}(C[s]) \leq \ell$ we have $\delta^*(C[s]) \in \mathcal{Q}_{\mathtt{f}}$ if and only if $T(C[s]) = 1$.*

Theorem 2. *Let $\mathbb{T} = (S, E, T, \ell)$ be a closed and consistent observation table, and N be the number of states of $\mathcal{A}(\mathbb{T})$. If $\mathcal{A}'$ is any other DFTA with N or fewer states, that is consistent with T on terms with depth at most ℓ, then $\mathcal{A}'$ has exactly N states and $\mathcal{L}(\mathcal{A}(\mathbb{T}))_{[\ell]} = \mathcal{L}(\mathcal{A}')_{[\ell]}$.*

Corollary 2. *Let $\mathcal{A}$ be the automaton corresponding to a closed and consistent observation table (S, E, T, ℓ) of the skeletons of a CFG G_U of an unknown language U, and N be its number of states. Let n be the number of states of a minimal DCTA of $K(D(G_U))$ with respect to ℓ. If $N \geq n$ then $N = n$ and $\mathcal{A}$ is a minimal DCTA of $K(D(G_U))$ with respect to ℓ.*

The LA^ℓ Algorithm

The algorithm LA^ℓ extends the observation table $\mathbb{T} = (S, E, T, \ell)$ whenever one of the following situations occurs: the table is not consistent, the table is not closed, or the table is both consistent and closed but the resulting automaton $\mathcal{A}(\mathbb{T})$ is not a cover tree automaton of $K(D(G_U))$ with respect to ℓ.

The pseudocode of the algorithm is shown below.

```
ask if $(\{\mathtt{S}\}, \Sigma, \emptyset, \mathtt{S})$ is a cover CFG of $G_U$ w.r.t. $\ell$
**if** answer is *yes* **then** halt and output the CFG $(\{\mathtt{S}\}, \Sigma, \emptyset, \mathtt{S})$
**if** answer is *no* with counterexample $t$ **then**
  set $S := \{s \mid s$ is a subterm of $t$ with depth at least $1\}$ and $E = \{\bullet\}$
  construct the table $\mathbb{T} = (S, E, T, \ell)$ using structural membership queries
  **repeat**
    **repeat**
      /* check consistency */
      **for** every $C \in E$, in increasing order of $i = \mathtt{d}_\bullet(C)$ **do**
          search for $s_1, s_2 \in S$ with $\mathtt{d}(s_1), \mathtt{d}(s_2) \leq \ell - i - 1$ and $C_1 \in \sigma_\bullet\langle S\rangle$
                  such that $C[C_1[s_1]]), C[C_1[s_2]] \in \mathcal{T}(Sk \cup \Sigma)_{[\ell]}$,
                             $s_1 \sim_k s_2$ where $k = \max\{\mathtt{d}(s_1), \mathtt{d}(s_2)\} + i + 1$,
                             and $T(C[C_1[s_1]]) \neq T(C[C_1[s_2]])$
          **if** found **then**
            add $C[C_1]$ to $E$
            extend $T$ to $E[S \cup X(S)_{[\ell]}]$ using structural membership queries
      /* check closedness */
      $\mathit{new\_row\_added} := \mathtt{false}$
      **repeat** for every $s \in S$, in increasing order of $\mathtt{d}(s)$
          search for $C_1 \in \sigma_\bullet\langle S\rangle$ such that $C_1[s] \nsim t$ for all $t \in S_{[\mathtt{d}(C_1[s])]}$
          **if** found **then**
            add $C_1[s]$ to $S$
            extend $T$ to $E[S \cup X(S)_{[\ell]}]$ using structural membership queries
            $\mathit{new\_row\_added} := \mathtt{true}$
      **until** $\mathit{new\_row\_added} = \mathtt{true}$ or all elements of $S$ have been processed
    **until** $\mathit{new\_row\_added} = \mathtt{false}$
    /* $\mathbb{T}$ is now closed and consistent */
    make the query whether $G(\mathcal{A}(\mathbb{T}))$ is a cover CFG of $G_U$ w.r.t. $\ell$
    **if** the reply is *no* with a counterexample $t$ **then**
      add to $S$ all subterms of $t$, including $t$, with depth at least 1,
                  in the increasing order given by $\prec_{\mathtt{T}}$
      extend $T$ to $E[S \cup X(S)_{[\ell]}]$ using structural membership queries
  **until** the reply is *yes* to the query if $G(\mathcal{A}(\mathbb{T}))$ is a cover CFG of $G_U$ w.r.t. $\ell$
  halt and output $G(\mathcal{A}(\mathbb{T}))$.
```

Consistency is checked by searching for $C \in E$ and $C_1 \in \sigma_\bullet\langle S\rangle$ such that $C[C_1]$ will ℓ-distinguish two terms $s_1, s_2 \in S$ not distinguished by any other context $C' \in E$ with $\mathtt{d}_\bullet(C') \leq \mathtt{d}_\bullet(C[C_1])$. Whenever such a pair of contexts (C, C_1) is found, $C[C_1]$ is added to E. Note that $C[C_1] \in \mathcal{C}(Sk \cup \Sigma)_{\langle \ell-1\rangle} \cap \mathcal{C}(Sk \cup \Sigma)_{[\ell]}$

because only such contexts can distinguish terms from S, and the addition of $C[C_1]$ to E yields a $\bullet$-prefix closed subset of $\mathcal{C}(Sk \cup \Sigma)_{\langle \ell-1\rangle} \cap \mathcal{C}(Sk \cup \Sigma)_{[\ell]}$.

The search of such a pair of contexts (C, C_1) is repeated in increasing order of the hole depth of C, until all contexts from E have been processed. Therefore, any context $C[C_1]$ with $C \in E$ and $C_1 \in \sigma_\bullet\langle S\rangle$ that was added to E because of a failed consistency check will be processed itself in the same **for** loop.

The algorithm checks closedness by searching for $s \in S$ and $C_1 \in \sigma_\bullet\langle S\rangle$ such that $C_1[s] \not\sim t$ for all $t \in S$ for which $\mathtt{d}(t) \leq \mathtt{d}(C_1[s])$. The search is performed in increasing order of the depth of s. If s and C_1 are found, $C_1[s]$ is added to the S component of the observation table, and the algorithm checks again consistency. Note that adding $C_1[s]$ to S yields a subterm closed subset of $\mathcal{T}(Sk \cup \Sigma)_{[\ell]}$. Also, closedness checks are performed only on consistent observation tables.

When the observation table is both consistent and closed, the corresponding DFTA is constructed and it is checked whether the language accepted by the constructed automaton coincides with the set of skeletal descriptions of the unknown context-free grammar G_U (this is called a *structural equivalence query*). If this query fails, a counterexample from $\mathcal{L}(\mathcal{A}(\mathbb{T}))_{[\ell]} \triangle K(D(G_U))_{[\ell]}$ is produced, the component S of the observation table is expanded to include t and all its subterms with depth at least 1, and the consistency and closedness checks are performed once more. At the end of this step, the component S of the observation table is subterm closed, and E is unchanged, thus $\bullet$-prefix closed.

Thus, at any time during the execution of algorithm LA^ℓ, the defining properties of an observation table are preserved: the component S is a subterm closed subset of $\mathcal{T}(Sk \cup \Sigma)_{[\ell]}$, and the component E is a $\bullet$-prefix closed subset of $\mathcal{C}(Sk \cup \Sigma)_{\langle \ell-1\rangle} \cap \mathcal{C}(Sk \cup \Sigma)_{[\ell]}$.

5 Algorithm Analysis

We notice that the number of states of the DFTA constructed by algorithm LA^ℓ will always increase between two successive structural equivalence queries. When this number of states reaches the number of states of a minimal DCTA of $K(D(G_U))$, the constructed DFTA is actually a minimal DCTA of $K(D(G_U))$ (Corollary 2) and the algorithm terminates.

From now on we assume implicitly that n is the number of states of a minimal DCTA of $K(D(G_U))$ with respect to ℓ, and that $\mathbb{T}(\mathtt{t})$ is the observation table $(S^{\mathtt{t}}, E^{\mathtt{t}}, T, \ell)$ before execution step $\mathtt{t}$ of the algorithm. By Corollary 2, $\mathcal{Q}^{\mathtt{t}}$ will always have between 1 and n elements. Note that the representative of an element $s \in S$ in $\mathcal{Q}^{\mathtt{t}}$ is a notion that depends on the observation table $\mathbb{T}(\mathtt{t})$. Therefore, we will use the notation $\mathtt{r}_{\mathtt{t}}(s)$ to refer to the representative of $s \in S^{\mathtt{t}}$ in the observation table $\mathbb{T}(\mathtt{t})$. With this notation, $\mathcal{Q}^{\mathtt{t}} = \{\mathtt{r}_{\mathtt{t}}(s) \mid s \in S^{\mathtt{t}}\}$.

Note that the execution of algorithm LA^ℓ is a sequence of steps characterised by the detection of three kinds of failure: closedness, consistency, and structural equivalence query. The $\mathtt{t}$-th execution step is

1. a failed closedness check when the algorithm finds $C_1 \in \sigma_\bullet\langle S^{\mathtt{t}}\rangle$ and $s \in S^{\mathtt{t}}$ such that $C_1[s] \not\sim t$ for all $t \in S^{\mathtt{t}}$ with $\mathtt{d}(t) \leq \mathtt{d}(C_1[s])$,

2. a failed consistency check when the algorithm finds $C \in E^{\mathtt{t}}$ with $\mathtt{d}_\bullet(C) = i$, $s_1, s_2 \in S^{\mathtt{t}}$ with $\mathtt{d}(s_1), \mathtt{d}(s_2) \leq \ell - i - 1$, and $C_1 \in \sigma_\bullet\langle S^{\mathtt{t}}\rangle$, such that $C[C_1[s_1]]$, $C[C_1[s_2]] \in \mathcal{T}(Sk \cup \Sigma)_{[\ell]}$, $s_1 \sim_k s_2$ where $k = \max\{\mathtt{d}(s_1), \mathtt{d}(s_2)\} + i + 1$, and $T(C[C_1[s_1]]) \neq T(C[C_1[s_2]])$,
3. a failed structural equivalence query when the observation table $\mathbb{T}(\mathtt{t})$ is closed and consistent, and the learning algorithm receives from the teacher a counterexample $t \in \mathcal{T}(Sk \cup \Sigma)_{[\ell]}$ as answer to the structural equivalence query with the grammar $G(\mathcal{A}(S^{\mathtt{t}}, E^{\mathtt{t}}, T, \ell))$.

In the following subsections we perform a complexity analysis of the algorithm by identifying upper bound estimates to the computations due to failed consistency checks, failed closeness checks, and failed structural equivalence queries.

5.1 Failed Closedness Checks

We recall that the $\mathtt{t}$-th execution step is a failed closedness check if the algorithm finds a context $C_1 \in \sigma_\bullet\langle S\rangle$ and a term $s \in S^{\mathtt{t}}$ such that $C_1[s] \nsim t$ for all $t \in S^{\mathtt{t}}$ with $\mathtt{d}(t) \leq \mathtt{d}(C_1[s])$. We will show that the number of failed closedness checks performed by algorithm LA^ℓ has an upper bound which is a polynomial in n. To prove this fact, we will rely on the following auxiliary notions:

- For $r, r' \in \mathcal{Q}^{\mathtt{t}}$, we define $r \prec_{\mathtt{T}}^{\mathtt{t}} r'$ if either $\mathtt{d}(r) < \mathtt{d}(r')$ or $\mathtt{d}(r) = \mathtt{d}(r')$ and there exists $\mathtt{t}' < \mathtt{t}$ such that $r \in \mathcal{Q}^{\mathtt{t}'}$ but $r' \notin \mathcal{Q}^{\mathtt{t}'}$ (that is, r became a representative in the observation table before r').
- To every set of representatives $\mathcal{Q}^{\mathtt{t}} = \{r_1, \ldots, r_m\}$ with $r_1 \prec_{\mathtt{T}}^{\mathtt{t}} \ldots \prec_{\mathtt{T}}^{\mathtt{t}} r_m$ we associate the tuple $\mathtt{tpl}(\mathcal{Q}^{\mathtt{t}}) := (d_1, \ldots, d_n) \in \{1, \ldots, \ell + 1\}^n$ where $d_i := \mathtt{d}(r_i)$ if $1 \leq i \leq m$, and $d_i := \ell + 1$ if $m < i \leq n$.
- We consider the following partial order on $\mathbb{N}^n$: $(x_1, \ldots, x_n) < (x'_1, \ldots, x'_n)$ iff there exists $i \in \{1, \ldots, n\}$ such that $x_i < x'_i$ and $x_j \leq x'_j$ for all $1 \leq j \leq n$.
- We denote by $\mathtt{st}_{\mathtt{t}}(i)$ the i-th component of $\mathcal{Q}^{\mathtt{t}}$ in the order given by $\prec_{\mathtt{T}}^{\mathtt{t}}$.

Lemma 9. *Suppose s has been introduced in $S^{\mathtt{t}+1}$ as a result of a failed closedness check. There exists $p \in Pos(s)$ such that $\|p\| = \mathtt{d}(s)$ and for every prefix p' of p different from p, $\mathtt{d}(\mathtt{r}_{\mathtt{t}+1}(s|_{p'})) = \mathtt{d}(s|_{p'})$.*

Corollary 3. *Whenever the $\mathtt{t}$-th execution step is a failed closedness check, the term introduced in $S^{\mathtt{t}+1}$ is in $\mathcal{Q}^{\mathtt{t}+1} \setminus \mathcal{Q}^{\mathtt{t}}$ and its depth is at most j, where j is the position in $\mathcal{Q}^{\mathtt{t}+1}$ of the newly introduced element according to ordering $\prec_{\mathtt{T}}^{\mathtt{t}}$.*

Corollary 4. *$\mathtt{d}(s) \leq n$ for all $s \in S^{\mathtt{t}}$ which was introduced in the table by a failed closedness check.*

Proof. $\mathtt{d}(s) \leq j$ by Cor. 3, and $j \leq n$ because $|\mathcal{Q}^{\mathtt{t}}| \leq n$ for all $\mathtt{t}$. Thus $\mathtt{d}(s) \leq n$.

Lemma 10. *Let j be the position of the element introduced in $\mathcal{Q}^{\mathtt{t}+1}$ by a failed closedness check. Then* $\mathtt{tpl}(\mathcal{Q}^{\mathtt{t}+1}) < \mathtt{tpl}(\mathcal{Q}^{\mathtt{t}})$ *and* $\mathtt{d}(\mathtt{st}_{\mathtt{t}+1}(j)) < \mathtt{d}(\mathtt{st}_{\mathtt{t}}(j))$.

Theorem 3. *The number of failed closedness checks performed during the entire run of LA^ℓ is at most $n(n + 1)/2$.*

5.2 Failed Consistency Checks

The $\mathtt{t}$-th execution step is a failed consistency check if the algorithm finds $C \in E^{\mathtt{t}}$ with $\mathtt{d}_{\bullet}(C) = i$, $s_1, s_2 \in S^{\mathtt{t}}$ with $\mathtt{d}(s_1), \mathtt{d}(s_2) \leq \ell - i - 1$, and $C_1 \in \sigma_{\bullet}\langle S^{\mathtt{t}}\rangle$, such that $C[C_1[s_1]], C[C_1[s_2]] \in \mathcal{T}(Sk \cup \Sigma)_{[\ell]}$, $s_1 \sim_k s_2$ where $k = \max\{\mathtt{d}(s_1), \mathtt{d}(s_2)\} + i + 1$, and $T(C[C_1[s_1]]) \neq T(C[C_1[s_2]])$. In this case, the context $C[C_1]$ is newly introduced in the component $E^{\mathtt{t}+1}$ of the observation table $\mathbb{T}(\mathtt{t}+1)$.

We will show that the number of failed consistency checks performed by the learning algorithm LA^{ℓ} has an upper bound which is a polynomial in n. To prove this fact, we rely on the following auxiliary notions:

- For $C, C' \in E^{\mathtt{t}}$, we define $C \prec_{\mathtt{C}}^{\mathtt{t}} C'$ if either $\mathtt{d}_{\bullet}(C) < \mathtt{d}_{\bullet}(C')$ or $\mathtt{d}_{\bullet}(C) = \mathtt{d}_{\bullet}(C')$ and there exists $\mathtt{t}' < \mathtt{t}$ such that $C \in E^{\mathtt{t}'}$ but $C' \notin E^{\mathtt{t}'}$ (that is, C became an experiment in the observation table before C').
- We define $\delta_{\mathtt{t}}(s_1, s_2) := \min_{\prec_{\mathtt{C}}}\{C \in E^{\mathtt{t}} \mid C\ \ell\text{-distinguishes } s_1 \text{ and } s_2\}$ for every $s_1, s_2 \in S^{\mathtt{t}}$ such that $s_1 \not\sim s_2$.
- A nonempty subset U of $E^{\mathtt{t}}$ induces a partition of a subset R of $S^{\mathtt{t}}$ into equivalence classes $Q_1, \ldots, Q_m$ if the following conditions are satisfied:
  1. $\bigcup_{j=1}^{m} Q_j = R$ and $Q_i \cap Q_j = \emptyset$ whenever $1 \leq i \neq j \leq m$,
  2. Whenever $1 \leq i \neq j \leq m$, $s_1 \in Q_i$, and $s_2 \in Q_j$, there exists $C \in U$ that ℓ-distinguishes s_1 and s_2.
  3. Whenever $s_1, s_2 \in Q_j$ for some $1 \leq j \leq m$, there is no $C \in U$ that ℓ-distinguishes s_1 and s_2.

Let $\mathcal{E}^{\mathtt{t}} := \{\delta_{\mathtt{t}}(s_1, s_2) \mid s_1, s_2 \in S^{\mathtt{t}}, s_1 \not\sim s_2\}$. Since $\sim$ is not an equivalence, not every subset of $E^{\mathtt{t}}$ induces a partition of $S^{\mathtt{t}}$ into equivalence classes. However, the next lemma shows that $\mathcal{E}^{\mathtt{t}}$ induces a partition of $\mathcal{Q}^{\mathtt{t}}$ into at least $|\mathcal{E}^{\mathtt{t}}|$ classes.

Theorem 4. *If $\mathcal{E}^{\mathtt{t}} = \{C_1, \ldots, C_k\}$ with $C_1 \prec_{\mathtt{C}} \ldots \prec_{\mathtt{C}} C_k$ then, for every $1 \leq i \leq k$, $\{C_1, \ldots, C_i\}$ induces a partition of $\mathcal{Q}^{\mathtt{t}}$ into at least i classes.*

Corollary 5. *For any $\mathtt{t}$, $\mathcal{E}^{\mathtt{t}}$ has at most n elements.*

We will compute an upper bound on the number of failed consistency checks by examining the evolution of $\mathcal{E}^{\mathtt{t}}$ during the execution of LA^{ℓ}. Initially, $\mathcal{E}^0 = \{\bullet\}$.

Lemma 11. *At any time during the execution of the algorithm, if $\mathcal{Q}^{\mathtt{t}}$ has $i \geq 2$ elements, then the hole depth of any context in $E^{\mathtt{t}}$ is less than or equal to $i - 2$.*

Let $\mathcal{E}^{\mathtt{t}} = \{C'_1, \ldots, C'_k\}$ before some execution step $\mathtt{t}$ of the algorithm LA^{ℓ}, where $C'_1 \prec_{\mathtt{C}} \ldots \prec_{\mathtt{C}} C'_k$. Then $k \leq n$ by Cor. 5. We associate to every such $\mathcal{E}^{\mathtt{t}}$ the n-tuple $\mathtt{tpl}(\mathcal{Q}^{\mathtt{t}}) = (y_1, \ldots, y_n) \in \{0, 1, \ldots, n-1\}^n$, where, for every $1 \leq j \leq n$, y_j is defined as follows:

- If $\mathcal{Q}^{\mathtt{t}}$ has at least $j + 1$ elements then, if i is the minimum integer such that $\{C'_1, \ldots, C'_i\}$ partitions $\mathcal{Q}^{\mathtt{t}}$ into at least $j + 1$ classes then $y_j = \mathtt{d}_{\bullet}(C'_j)$. Since every $\{C'_1, \ldots, C'_i\}$ partitions $\mathcal{Q}^{\mathtt{t}}$ into at least i classes (by Lemma 4) and we assume that $\mathcal{E}^{\mathtt{t}} = \{C'_1, \ldots, C'_k\}$ partitions $\mathcal{Q}^{\mathtt{t}}$ into $|\mathcal{Q}^{\mathtt{t}}| \geq j + 1$ classes, we conclude that such i exists.

- otherwise $y_j = n - 1$.

For $1 \leq j \leq n$ we denote the j-th component of $\mathtt{tpl}(\mathcal{E}^{\mathsf{t}})$ by $\mathtt{dh_t}(j)$. Note that, for all $1 \leq i \leq k$, $\mathtt{d}_\bullet(C_i') \leq |\mathcal{Q}^{\mathsf{t}}| - 2$ by Theorem 11, and $|\mathcal{Q}^{\mathsf{t}}| \leq n$, hence $\mathtt{d}_\bullet(C_i') \leq n-2$. Therefore, we can always distinguish the components y_i of $\mathtt{tpl}(\mathcal{Q}^{\mathsf{t}})$ that correspond to the defining case (1) from those in case (2).

Lemma 12. $\mathtt{dh_t}(j) \leq j-1$ *whenever* $2 \leq j \leq n$ *and* $\mathtt{dh_t}(j) \neq n-1$.

Theorem 5. *If* $\mathcal{Q}^{\mathsf{t}}$ *has at least 2 elements then the number of failed consistency checks over the entire run of* LA^ℓ *is at most* $n(n-1)/2$.

5.3 Failed Structural Equivalence Queries

Every failed structural equivalence query yields a counterexample which increases the number of representatives in $\mathcal{Q}^{\mathsf{t}}$. Thus

Theorem 6. *The number of failed structural equivalence queries is at most* n.

5.4 Space and Time Complexity

We are ready now to express the space and time complexity of LA^ℓ in terms of the following parameters:

- n = the number of states of a minimal DFCA for the language of structural descriptions of the unknown grammar with respect to ℓ,

- m = the maximum size of a counterexample returned by a failed structural equivalence query,

- p = the cardinality of the alphabet Σ of terminal symbols, and

- d = the maximum rank (or arity) of the symbol $\sigma \in Sk$.

First, we determine the space needed by the observation table. The number of elements in S^{t} is initially 0 (i.e., $|S^0| = 0$) and is increased either by a failed closedness check or by a failed structural membership query. By Theorem 3, the number of failed closedness checks is at most $n(n+1)/2$, and each of them adds one element to S. By Theorem 6, the number of failed structural equivalence queries is at most n. A failed structural equivalence query which produces a counterexample t with $\mathtt{sz}(t) \leq m$, adds at most m terms to S^{t}. Thus, $|S^{\mathsf{t}}| \leq n(n+1)/2 + n\,m = O(m\,n + n^2)$ and $|S^{\mathsf{t}} \cup \Sigma| = O(m\,n + n^2 + p)$, therefore $|\sigma_\bullet\langle S^{\mathsf{t}}\rangle| \leq \sum_{j=0}^{d-1}(j+1)\,|S^{\mathsf{t}} \cup \Sigma|^j = O((d+1)\,(m\,n + n^2 + p)^d)$ and $|X(S)| \leq \sum_{j=1}^{d} |S^{\mathsf{t}} \cup \Sigma|^j = O(d\,(m\,n + n^2 + p)^d)$. Thus $S^{\mathsf{t}} \cup X(S^{\mathsf{t}})_{[\ell]}$ has $O(d\,(m\,n + n^2 + p)^d)$ elements. By Theorem 5, there may be at most $n(n-1)/2$ failed consistency checks, and each of them adds a context to E^{t}. Thus E^{t} has $O(n^2)$ elements and $E^{\mathsf{t}}[S^{\mathsf{t}} \cup X(S^{\mathsf{t}})_{[\ell]}]$ has $O(n^2 d\,(m\,n + n^2 + p)^d)$ elements. By Lemma 12, $\mathtt{d}_\bullet(C) \leq n-1$ for all $C \in E^{\mathsf{t}}$. We also know that, if $s \in S^{\mathsf{t}}$, then $\mathtt{d}(s) \leq m$ if it originates from a failed structural equivalence query, and $\mathtt{d}(s) \leq n$ if it originates from a failed closedness check (by Cor. 4). Therefore $\mathtt{d}(s) \leq \max(m,n)$ for all $s \in S^{\mathsf{t}}$, and thus $\mathtt{d}(x) \leq 1 + \max(m,n) \leq 1 + m + n$ for all $x \in S^{\mathsf{t}} \cup X(S^{\mathsf{t}})$ and $\mathtt{d}(t) \leq m + 2\,n$ for all $t \in E^{\mathsf{t}}[S^{\mathsf{t}} \cup X(S^{\mathsf{t}})_{[\ell]}]$. Since the number of positions of such a term t is $\sum_{j=0}^{m+2\,n} d^j = O((m + 2\,n + 1)d^{m+2\,n})$, we conclude that the total space occupied by an observation table at any time is $O\left(n^2(m\,n+n^2+p)^d(m+2\,n+1)d^{m+2\,n+1}\right)$.

Next, we examine the time complexity of the algorithm by looking at the time needed to perform each kind of operation.

Since the consistency checks of the observation table are performed in a **for** loop which checks the result produced by $s_1 \sim_k s_2$ (where $s_1, s_2 \in S^{\mathtt{t}}$) in increasing order of k, the result produced by $s_1 \sim_k s_2$ can be reused in checking $s_1 \sim_{k+1} s_2$ and so the corresponding elements in the rows of s_1 and s_2 are compared only once. Thus, the total time needed to check if the observation table is consistent involves at most $(|S^{\mathtt{t}}| \cdot (|S^{\mathtt{t}}|-1)/2) \cdot |E^{\mathtt{t}}| \cdot (1+|\sigma_\bullet\langle S^{\mathtt{t}}\rangle|)$, comparisons. As $\sigma_\bullet\langle S^{\mathtt{t}}\rangle$ has $O(d\,(m\,n+n^2+p)^d)$ elements, a consistency check of the table takes $O((m\,n+n^2)^2n^2d\,(m\,n+n^2+p)^d) = O(n^2d\,(m\,n+n^2+p)^{d+2})$ time. As there are at most $(n\,(n+1)/2+1)\,(n+1) = O(n^3)$ consistency checks, the total time needed to check if the table is consistent is $O(n^5d\,(m\,n+n^2+p)^{d+2})$.

Checking if the observation table is closed takes at most $|S^{\mathtt{t}}|^2 \cdot |\sigma_\bullet\langle S^{\mathtt{t}}\rangle| \cdot |E^{\mathtt{t}}|$ time, which is $O((m\,n+n^2)^2d\,(m\,n+n^2+p)^dn^2) = O(n^2d\,(m\,n+n^2+p)^{d+2})$.

Extending an observation table $\mathbb{T}(\mathtt{t})$ with a new element in $S^{\mathtt{t}+1}$ requires the addition of $\sum_{k=2}^{d}(2^{k-1}-1) = 2^d-d-1$ contexts to $\sigma_\bullet\langle S^{\mathtt{t}+1}\rangle \setminus \sigma_\bullet\langle S^{\mathtt{t}}\rangle$, thus the addition of at most 2^d-d new rows for the new elements of $S^{\mathtt{t}+1} \cup X(S^{\mathtt{t}+1})$ in the observation table $\mathbb{T}(\mathtt{t}+1)$. This extension requires at most $(2^d-d) \cdot |E^{\mathtt{t}}| \cdot (1+|\sigma_\bullet\langle S^{\mathtt{t}}\rangle|) = O(n^2d\,(2^d-d)\,(m\,n+n^2+p)^d)$ membership queries. The number of elements added to $S^{\mathtt{t}}$ as a result of a failed structural equivalence query is at most m. As there will be at most n failed structural equivalence queries and at most $n(n+1)/2$ failed closedness checks, the maximum number of elements added to $S^{\mathtt{t}}$ is $n(n+1)/2+m\,n = O(m\,n+n^2)$. Thus the total time spent on inserting new elements in the S-component of the observation table is $O(n^2d\,(2^d-d)\,(m\,n+n^2)(m\,n+n^2+p)^d)$. Adding a context to $E^{\mathtt{t}}$ requires at most $|S^{\mathtt{t}}+X(S^{\mathtt{t}})_{[\ell]}| = O(d\,(m\,n+n^2+p)^d)$ membership queries. These additions are performed only by failed consistency checks, and there are at most $n(n-1)/2$ of them. Thus, the total time spent to insert new contexts in the E-component of the observation table is $O(n^2d\,(m\,n+n^2+p)^d)$. We conclude that the total time spent to add elements to the components S and E of the observation table is $O(n^2d\,(2^d-d)\,(m\,n+n^2)(m\,n+n^2+p)^d)$, which is polynomial.

The identification of the representative $\mathtt{r}_{\mathtt{t}}(s)$ for every $s \in S^{\mathtt{t}}$ can be done by performing $((|S^{\mathtt{t}}|)(S^{\mathtt{t}}-1)/2)\,|E^{\mathtt{t}}| = O((m\,n+n^2)^2n^2)$ comparisons.

Thus, all DFCAs $\mathcal{A}(\mathbb{T}(\mathtt{t}))$ corresponding to consistent and closed observation tables $\mathbb{T}(\mathtt{t})$ can be constructed in time polynomial in m and n. Since the algorithm encounters at most n consistent and closed observation tables, the total running time of the algorithm is polynomial in m and n.

6 Conclusions and Acknowledgments

We have presented an algorithm, called LA^ℓ, for learning cover context-free grammars from structural descriptions of languages of interest. LA^ℓ is an adaptation of Sakakibara's algorithm LA for learning context-free grammars from structural descriptions, by following a methodology similar to the design of Ipate's algorithm L^ℓ as a nontrivial adaptation of Angluin's algorithm L^*. Like L^*, our algorithm synthesizes a minimal deterministic cover automaton consistent with an observation table maintained via a learning protocol based on what is called in the literature a "minimally adequate teacher" [1]. And again, like algorithm L^*, our algorithm is guaranteed to synthesize the desired automaton in time polynomial in n and m, where n is its number of states and m is the maximum size of a counterexample to a structural equivalence query. As the size of a minimal finite cover automaton is usually much smaller than that of a minimal automaton that accepts that language, the algorithm LA^ℓ is a better choice than algorithm LA for applications where we are interested only in an accurate characterisation of the structural descriptions with depth at most ℓ.

This work has been supported by CNCS IDEI Grant PN-II-ID-PCE-2011-3-0981 "Structure and computational difficulty in combinatorial optimization: an interdisciplinary approach."

References

1. Angluin, D.: Learning regular sets from queries and counterexamples. Information and Computation 75, 87–106 (1987)
2. Angluin, D., Kharitonov, M.: When won't membership queries help? Journal of Computer and System Sciences 50(2), 336–355 (1995)
3. Brainerd, W.S.: The minimalization of tree automata. Information and Control 13(5), 484–491 (1968)
4. Farzan, A., Chen, Y.-F., Clarke, E.M., Tsay, Y.-K., Wang, B.-Y.: Extending automated compositional verification to the full class of omega-regular languages. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 2–17. Springer, Heidelberg (2008)
5. De la Higuera, C.: Grammatical inference: learning automata and grammars. Cambridge University Press (2010)
6. Holzer, M., Jakobi, S.: From equivalence to almost-equivalence, and beyond—minimizing automata with errors. In: Yen, H.-C., Ibarra, O.H. (eds.) DLT 2012. LNCS, vol. 7410, pp. 190–201. Springer, Heidelberg (2012)
7. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation, 2nd edn. Pearson Addison Wesley (2003)
8. Ipate, F.: Learning finite cover automata from queries. Journal of Computer and System Sciences 78(1), 221–244 (2012)
9. Kumar, V., Madhusudan, P., Viswanathan, M.: Minimization, learning, and conformance testing of boolean programs. In: Baier, C., Hermanns, H. (eds.) CONCUR 2006. LNCS, vol. 4137, pp. 203–217. Springer, Heidelberg (2006)
10. Levy, L.S., Joshi, A.K.: Skeletal structural descriptions. Information and Control 39(3), 192–211 (1978)
11. Maler, O., Pnueli, A.: On the learnability of infinitary regular sets. Information and Computation 118(2), 316–326 (1995)
12. Sakakibara, Y.: Learning context-free grammars from structural data in polynomial time. Theoretical Computer Science 76, 223–242 (1990)
13. Sipser, M.: Introduction to the Theory of Computation, 2nd edn. Thomson (2006)