One-shot learning and the polynomial characterizability barrier

Anna Kasprzik

Technical report 12-4, University of Trier, Germany

Abstract

We survey two existing algorithms for the inference of finite-state tree automata from membership queries and a finite positive sample or equivalence queries, and we suggest a correction for one of them which we deem necessary in order to ensure its termination. We present two algorithms for the same two settings when the underlying description is not a deterministic but a residual canonical finite-state automaton. To this end, we adapt all necessary notions to the residual tree case. From our completed perspective, we discuss the terms on which those four algorithms can be considered to be of polynomial complexity, and also where there may be hidden exponentiality.

Keywords: Grammatical inference, one-shot learning, query learning, learning from finite samples, polynomial characterizability

1 Introduction

The area of Grammatical Inference (GI) has evolved as the formal language part of Computational Learning Theory and is concerned with learning algorithms that infer a description (for example, a grammar or an automaton) for an unknown formal language from given information.

In the present article, we assume a learning model where a learner has access to several information sources and is required to return a single final hypothesis describing the target language after a finite number of steps. This notion has also been termed *one-shot learning* (see [28, 29]). In addition to being finite, we want the number of steps taken by the learner to be bounded by some polynomial measure with respect to the target (*efficient learning*), although that measure will not be revealed to the learner.

The sources of information in a learning process can take various forms. Besides a continuous data stream (as studied in [21]), the most studied ones are probably the following. Let L be the target language.

- *Equivalence queries (EQs):* The learner has access to a teacher, also called *oracle*, who is able to judge the correctness of a description $\mathcal{A}$ for

the target, i.e., answers queries of the form '$L = \mathcal{L}(\mathcal{A})$?', and returns a *counterexample* $c \in (L \setminus \mathcal{L}(\mathcal{A})) \cup (\mathcal{L}(\mathcal{A}) \setminus L)$ in case of non-equivalence.

- *Membership queries (MQs)*: The learner has access to an oracle who is able to answer whether an object w is a member of L ('$w \in L$?').

- *Finite samples of L, which can be positive (subsets of L) or negative (subsets of the complement of L)*: No teacher is required, or rather, the only task of the teacher is to procure such a sample and give it to the learner before the learning process is begun. However, in order to ensure identification, especially if no query-based information source is given, those sets have to fulfil certain significant properties with respect to L. The learner has to rely on the given sample containing all necessary information in some form. We will concentrate on cases where this information notably depends on the structure of the target, and more specifically, on the structure of a canonical description of L.

During the past decades a considerable range of learning algorithms for various settings have been developed and presented in the literature. One of the language classes that have been examined most thoroughly with respect to their algorithmic learnability is the class of regular languages. It has been shown (see [3, 4, 21]) that the regular class cannot be learned efficiently from one kind of query or sample alone. Three well-studied combinations of two such sources that allow the inference of the most notorious canonical representation for a regular language, the state-minimal *deterministic finite-state automaton* (DFA), join MQs and EQs (exploited by Angluin's seminal algorithm LSTAR [2]), MQs and a finite positive sample ([1]), and finite positive and negative samples (RPNI [32, 14]).

The cited learners successfully identify a regular language L in a polynomially bounded number of steps and/or queries depending on the size of the state-minimal DFA for L and/or of the data set received throughout the process, provided that the samples fulfil certain informativeness conditions. Here the size of an automaton refers to the number of its states and the size of a set of strings refers to the sum of the lengths of all its members. Moreover, all those results are based on the retrieval of the correct set of equivalence classes under the Myhill-Nerode relation (see [23, 12]) and as a consequence, the language description returned by the learners is exactly the state-minimal DFA for L itself.

These polynomial learnability results are actually quite spectacular seen against the rather bleak background of Gold's [22] finding that in general the retrieval of the smallest automaton consistent with a finite labeled set of data is NP-complete. And there are more negative results: In [13], de la Higuera defines the notion of *polynomial characterizability* where a description class is polynomially characterizable for a certain learner if for each description we can construct a special set which when included into the available data is

sure to make the learner identify the corresponding language correctly and is of polynomial size with respect to the size of the description. He then goes on to show that among other description classes the classes of non-deterministic finite-state automata (NFA) and of context-free grammars (CFGs) are not polynomially characterizable (given P $\neq$ NP). However, as we will see from the next two paragraphs, this is not a non-inferability result as yet.

DFA are not the only kind of automata where each state can be unambiguously related to an equivalence class of the corresponding language. In [16], Denis et al. introduce a special case of NFA, so-called *residual finite-state automata* (RFSA), where each state represents a residual language of the language recognized by the automaton. A residual language is the set of all suffixes that extend a given string into the language in question. For any formal language there is a natural one-to-one correspondence between the residual languages and the equivalence classes it defines. And contrary to NFA in general, RFSA also have the advantageous property that for every regular language there exists a unique state-minimal (the transition-maximal, or *saturated*) RFSA, which makes them an attractive choice for descriptions in the design of learning algorithms and their applications due to their succinctness since that RFSA can be exponentially smaller than the state-minimal DFA and is at most as big.

In [15], RFSA have been shown not to be polynomially characterizable either. However, the authors of [15] also provide an efficient algorithm learning regular string languages from given data that returns another kind of saturated RFSA which for every regular language is unambiguously defined as well and for many languages has a smaller number of states than the corresponding state-minimal DFA. And some years later, Bollig et al. [7] have presented an efficient algorithm for the setting of MQs and EQs (based on Angluin's LSTAR [2]) which in case of success returns an RFSA that is isomorphic to the canonical state-minimal RFSA mentioned above.

The given learnability results for DFA have soon been extended to *deterministic finite-state tree automata* (DFTA), see [33, 17, 5, 31]. Moreover, the notion of RFSA can be equally extended to trees: *Residual finite-state tree automata* (RFTA) have been defined and studied in [8]. As in the case of strings, for every regular tree language L there is a unique state-minimal and transition-maximal RFTA which can be exponentially more succinct in its number of states than the corresponding state-minimal DFTA. The inference of tree automata is particularly attractive due to the wide range of their applications, from computational linguistics to mark-up languages in the context of the world wide web. On a theoretical level, tree structures represent a reasonable balance between expressivity and tractability.

In the present work, we will concentrate on the effects of shifting from the string to the tree case and from the deterministic to the residual case. We discuss the necessary preliminaries in Section 2. In Subsection 3.1, we describe two prototypical learners (based on existing ones [17, 5] from the

	strings	*trees*
deterministic	[2],[1]	[5],[17]
residual	[7], -	this paper

Figure 1: References for inference from MQs & EQs, MQs & a sample

literature) for the deterministic tree case, and suggest a correction for the algorithm in [5] that we deem necessary to ensure its termination. Then we attempt to reproduce those results under the assumption that the canonical description of choice is not a DFTA but an RFTA in Subsection 3.2. So far, the learnability results for residual finite-state automata were established only for the case of strings ([15, 7]) – the tree case involves a fair amount of additional intricacy when one considers the possible transitions in a conjectured tree automaton due to the interactions between adjacent subtrees. On the one hand, we want to demonstrate that we can assemble algorithms with components resembling the ones developed for the deterministic case since it shows that due to the Myhill-Nerode theorem the inference of DFTA and of RFTA is based on the same fundamental principle. This contributes to the completion of the picture of regular tree language inference in settings involving combinations of the information sources fixed above, see Figure 1. For related work by the author, also see [24, 25, 26, 27]. On the other hand, the impact of this shift from determinism to non-determinism on the general complexity interacts with the impact of the shift from strings to trees, and as one of our main concerns we would like to discuss the conditions for successful identification of the target description using only a polynomial measure of computation resources. The crucial issue is the exact definition of that measure and we argue that there are several points in the two shifts addressed above where exponentiality can be hidden. We conclude with a summarizing discussion in Section 4.

All algorithms described or cited in this article that infer a finite-state automaton for a regular language from given information are based on the retrieval of the correct set of equivalence classes using a substructure-context relation in some way or other. The majority of them recur to the concept of a so-called *observation table*, which is a versatile and relatively abstract means to perform and document the inference process at the same time since it allows to represent the relationship between the available substructures and contexts with respect to a language in a fairly intuitive way without having to deal with the idiosyncrasies of more specific descriptions such as finite-state automata or grammars, which moreover can be easily derived from it. The concept can be traced back to Gold's [22] *state characterization matrix* and has been widely established in the GI community since Angluin [2] has adopted it for the description of her seminal algorithm LSTAR. Therefore, we will use observation tables in order to make our point.

2 Preliminaries

We assume familiarity with the Chomsky Hierarchy and associated notions for strings (see for example [23]). This section also includes some important theoretical observations for the discussion to come.

2.1 Trees

Definition 1 *A* ranked alphabet *is a pair* $\langle \Sigma, \sigma \rangle$ *where* Σ *is a set of symbols and* $\sigma : \Sigma \longrightarrow \mathbb{N}$ *is a total function assigning a* rank *to each element of* Σ*. We will abbreviate* $\langle \Sigma, \sigma \rangle$ *to* Σ*, and we denote by* $\Sigma_n := \{f \in \Sigma \mid \sigma(f) = n\}$ *the set of symbols from* Σ *that are associated with rank* n*.*
The set $\mathbb{T}_\Sigma$ of all trees over a ranked alphabet Σ *is defined as the smallest set such that* $\Sigma_0 \subseteq \mathbb{T}_\Sigma$ *and* $t = f(t_1, \ldots, t_n) \in \mathbb{T}_\Sigma$ *for all* $n \geq 1$*,* $f \in \Sigma_n$*, and* $t_1, \ldots, t_n \in \mathbb{T}_\Sigma$*. Any set* $L \subseteq \mathbb{T}_\Sigma$ *is called a* tree language*.*
Let $t = f(t_1, \ldots, t_n) \in \mathbb{T}_\Sigma$ *for* $n \geq 0$*. We define the* set $Subt(t)$ of all subtrees of t *as the smallest set such that* $t \in Subt(t)$ *and, if* $n \geq 1$*,* $Subt(t_i) \subseteq Subt(t)$ *for all* $i \in \{1, \ldots, n\}$*. We call* $t_1, \ldots, t_n$ *the* direct subtrees *of* t*. We define* $Subt(T) := \bigcup\{Subt(t) \mid t \in T\}$ *for a set* $T \subseteq \mathbb{T}_\Sigma$*.*

Definition 2 *Let* $\Box$ *be a special symbol of rank 0 not contained in* Σ*.*
A tree $c \in \mathbb{T}_{\Sigma \cup \{\Box\}}$ *in which* $\Box$ *occurs exactly once is a* context*, and the set of all contexts over* Σ *is denoted by* $\mathbb{C}_\Sigma$*. For* $c \in \mathbb{C}_\Sigma$ *and* $s \in \mathbb{T}_\Sigma \cup \mathbb{C}_\Sigma$*,* $c[\![s]\!]$ *denotes the tree obtained by substituting* s *for* $\Box$ *in* c*. The depth* $cdp(c)$ *of a context* c *is the length of the path from the root to the leaf labeled by* $\Box$*.*
Let $t \in \mathbb{T}_\Sigma \cup \mathbb{C}_\Sigma$ *and* $T \subseteq \mathbb{T}_\Sigma \cup \mathbb{C}_\Sigma$*. We define*
$Cont(t) := \{c \in \mathbb{C}_\Sigma \mid \exists t' \in Subt(t) : c[\![t']\!] = t\}$ *and*
$Cont(T) := \{c \in \mathbb{C}_\Sigma \mid \exists t' \in Subt(T) : c[\![t']\!] \in T\}$*.*
For a tree $f(t_1, \ldots, t_n) \in \mathbb{T}_\Sigma$ *and* $j \in \{1, \ldots, n\}$ *we define* $f(t_1, \ldots, t_n)^\Box_j$ *as a context* $f(t'_1, \ldots, t'_n) \in \mathbb{C}_\Sigma$ *with* $t'_j = \Box$ *and* $t'_i = t_i$ *for all* $i \in \{1, \ldots, n\} \setminus \{j\}$*.*

The traditional (term-based) tree language theory ([12, 20]) also establishes the concept of finite-state automata for trees. We consider bottom-up tree automata where the automaton assigns a state to each leaf and then works its way up by assigning states to the respective mother nodes according to the states assigned to the children and the admissible transition rules.

Definition 3 *Let* Σ *be finite.*
A finite-state tree automaton (FTA) *is a tuple* $\mathcal{A} = \langle \Sigma, Q, F, \delta \rangle$ *where* Q *is the finite set of states,* $F \subseteq Q$ *the set of accepting states, and the transition relation* δ *is a set of triples of the form* $\langle f, q_1 \cdots q_n, q \rangle$ *for* $n \geq 0$*,* $f \in \Sigma_n$*, and* $q_1, \ldots, q_n, q \in Q$ *where* $q_1 \cdots q_n$ *denotes the sequence of the states* $q_1, \ldots, q_n$ *in the same order which for* $n = 0$ *we will write as* $\langle\rangle$*. We will also use* δ *to denote a function such that* $\delta(\langle f, q_1 \cdots q_n \rangle) = \{q \in Q \mid \exists \langle f, q_1 \cdots q_n, q \rangle \in \delta\}$*.*

From δ we can derive a function $\delta^* : \mathbb{T}_\Sigma \longrightarrow 2^Q$ such that $\delta^*(f(s_1,\ldots,s_n)) =$

$$\{q \in Q \mid \exists\langle f, q_1 \cdots q_n, q\rangle \in \delta : \forall i \in \{1,\ldots,n\} : q_i \in \delta^*(s_i)\},$$

and a function $\delta^+ : Q \times \mathbb{C}_\Sigma \longrightarrow 2^Q$ such that $\delta^+(q,\Box) = \{q\}$ and for $e \neq \Box$,

$$\begin{aligned}\delta^+(q,e) = \{q' \in Q \mid \exists e', e'' \in \mathbb{C}_\Sigma : e = e'[\![e'']\!] \wedge \exists q'' \in Q : q' \in \delta^+(q'',e') \wedge \\ \exists n \geq 1 : \exists f(s_1,\ldots,s_n) \in \mathbb{T}_\Sigma : e'' = f(s_1,\ldots,s_n)^{\Box}_i \wedge \\ \exists\langle f, q_1 \cdots q_n, q''\rangle \in \delta : q_i = q \wedge \forall j \in \{1,\ldots,n\} \setminus \{i\} : q_j \in \delta^*(s_j)\}.\end{aligned}$$

Intuitively, $\delta^*(t)$ is the set of all states that the tree t ends up in and $\delta^+(q,e)$ is the set of all states that can be reached from the state q by the context e.
Let $\mathcal{L}_q := \{s \in \mathbb{T}_\Sigma \mid q \in \delta^*(s)\}$ and $\mathcal{C}_q := \{e \in \mathbb{C}_\Sigma \mid \delta^+(q,e) \cap F \neq \emptyset\}$.
Intuitively, $\mathcal{L}_q$ is the set of all trees that can end up in q and $\mathcal{C}_q$ is the set of all contexts that can lead from q into an accepting state.
We write $\mathcal{A}(s) = 1$ for $s \in \mathbb{T}_\Sigma$ if $\delta^*(s) \cap F \neq \emptyset$, $\mathcal{A}(s) = 0$ if $\delta^*(s) \cap Q \neq \emptyset$ but $\delta^*(s) \cap F = \emptyset$, and $\mathcal{A}(s) = *$ if $\delta^*(s) = \emptyset$.
The language accepted by $\mathcal{A}$ is defined as $\mathcal{L}(\mathcal{A}) := \{s \in \mathbb{T}_\Sigma \mid \mathcal{A}(s) = 1\}$.
Any tree language accepted by an FTA is called recognizable *or* regular.
If for all $n \geq 0$ with $\Sigma_n \neq \emptyset$ and for all $f \in \Sigma_n$ and all $q_1,\ldots,q_n \in Q$ there is a transition $\langle f, q_1 \cdots q_n, q\rangle \in \delta$ then $\mathcal{A}$ is total. *If for no $\langle f, q_1 \cdots q_n, q\rangle \in \delta$ there is $\langle f, q_1 \cdots q_n, q'\rangle \in \delta$ with $q' \neq q$ then $\mathcal{A}$ is* deterministic (a DFTA).

For any tree language $L \subseteq \mathbb{T}_\Sigma$, the equivalence relation $\equiv_L$ is defined such that $t \equiv_L t'$ for $t, t' \in \mathbb{T}_\Sigma$ iff $e[\![t]\!] \in L \Leftrightarrow e[\![t']\!] \in L$ for all contexts $e \in \mathbb{C}_\Sigma$. The *index* I_L of L is the cardinality of the set $\mathcal{E}_L := \{[t]_L \mid t \in \mathbb{T}_\Sigma\}$ where $[t]_L$ denotes the equivalence class containing t. The Myhill-Nerode theorem (see for example [23] for strings, and [12] for trees) states that I_L is finite iff L is recognizable by a finite-state automaton. For every regular tree language $L \subseteq \mathbb{T}_\Sigma$ there is a unique FTA $\mathcal{A}_L = \langle \Sigma, Q_L, F_L, \delta_L\rangle$ with I_L states where

- $Q_L := \mathcal{E}_L$,
- $F_L := \{x \in Q_L \mid x \subseteq L\}$, and
- $\delta_L := \{\langle f, x_1 \cdots x_n, x\rangle \mid x_1,\ldots,x_n, x \in Q_L \wedge f \in \Sigma_n \wedge$
  $\exists t_1,\ldots,t_n \in \mathbb{T}_\Sigma : \forall i \in \{1,\ldots,n\} :$
  $t_i \in x_i \wedge f(t_1,\ldots,t_n) \in x\}$.

Each state $x \in Q_L$ recognizes the equivalence class under $\equiv_L$ it is associated with, i.e., we have $\mathcal{L}_x = x$. As a consequence, $\mathcal{A}_L$ recognizes L and is the only state-minimal total DFTA recognizing L up to isomorphism (see [12]).
If we have $\mathbb{T}_\Sigma \setminus \mathit{Subt}(L) \neq \emptyset$ then there is a *failure state* q in that DFTA with $\mathcal{C}_q = \emptyset$ and hence there exists a non-total DFTA for L with one less state which is obtained by stripping the total one of the failure state q. We denote any of those two canonical DFTA for L by $\mathcal{A}_L$ in general but in cases where it matters, we will denote the total one by $\mathcal{A}^\bullet_L = \langle \Sigma, Q_\bullet, F_\bullet, \delta_\bullet\rangle$ and the not necessarily total one without a failure state by $\mathcal{A}^\circ_L = \langle \Sigma, Q_\circ, F_\circ, \delta_\circ\rangle$. For $\mathbb{T}_\Sigma \setminus \mathit{Subt}(L) = \emptyset$ the automata $\mathcal{A}^\bullet_L$ and $\mathcal{A}^\circ_L$ coincide.

2.2 Residual languages

The following three definitions are based on [8].

Definition 4 *The* (bottom-up) residual language $t^{-1}L$ *of a tree* $t \in \mathbb{T}_\Sigma$ *with respect to a language* $L \subseteq \mathbb{T}_\Sigma$ *is defined as the set* $\{e \in \mathbb{C}_\Sigma \mid e[\![t]\!] \in L\}$. *The set of all residual languages defined by* L *is denoted by* $\mathcal{C}_L$.

There is a natural correspondence between the equivalence classes and the residual languages of L determined by $s^{-1}L = t^{-1}L \Leftrightarrow s \equiv_L t$ for any $s, t \in \mathbb{T}_\Sigma$ which is due to the fact that the definitions of both concepts are based on the substructure-context relation. As a consequence, each of the I_L equivalence classes under $\equiv_L$ defines a unique residual language with respect to L, and any pair of equivalence classes of L can be distinguished by their differing – but not necessarily disjoint – sets of contexts. Obviously, since the index I_L is finite if and only if L is regular the same holds for $\mathcal{C}_L$. This precise correspondence also accounts for Lemma 1, which has been proven in [8]:

Lemma 1 *Let* $t_1, \ldots, t_n, t'_1 \ldots, t'_n \in \mathbb{T}_\Sigma$ *for some* $n \geq 1$. *If* ${t_i}^{-1}L \subseteq {t'_i}^{-1}L$ *for all* $i \in \{1, \ldots, n\}$ *then* $f(t_1, \ldots, t_n)^{-1}L \subseteq f(t'_1 \ldots, t'_n)^{-1}L$ *for all* $f \in \Sigma_n$.

By the notion of residual languages one can define a special kind of FTA:

Definition 5 *A* residual finite-state tree automaton (RFTA) *is an FTA* $\mathcal{R} = \langle \Sigma, Q, F, \delta \rangle$ *such that for all* $q \in Q$ *there is* $t \in \mathbb{T}_\Sigma$ *with* $\mathcal{C}_q = t^{-1}\mathcal{L}(\mathcal{R})$.

The class of tree languages that are recognized by RFTA corresponds exactly to the class of regular tree languages as a whole (see [8]).

Definition 6 *Let* $L \subseteq \mathbb{T}_\Sigma$. *A residual language* $\gamma \in \mathcal{C}_L$ *is* composed *iff* $\gamma = \bigcup\{\gamma' \in \mathcal{C}_L \mid \gamma' \subsetneq \gamma\}$. *Otherwise we say that it is* prime. *The set of prime residual languages of* L *is denoted by* $\mathcal{P}_L$.

For a regular $L \subseteq \mathbb{T}_\Sigma$, one can define an FTA $\mathcal{R}_L = \langle \Sigma, Q_L, F_L, \delta_L \rangle$ by

- $Q_L := \mathcal{P}_L$,
- $F_L := \{y \in Q_L \mid \Box \in y\}$, and
- $\delta_L := \{\langle f, y_1 \cdots y_n, y \rangle \mid y_1, \ldots, y_n, y \in Q_L \wedge f \in \Sigma_n \wedge$
  $\exists t_1, \ldots, t_n \in \mathbb{T}_\Sigma : \forall i \in \{1, \ldots, n\} :$
  $y_i = {t_i}^{-1}L \wedge y \subseteq f(t_1, \ldots, t_n)^{-1}L\}$.

The transitions in δ_L are determined by the inclusion relations among the residual languages of L. The resulting FTA $\mathcal{R}_L$ recognizes L and meets the definition of an RFTA. Moreover, for each state $y \in Q_L = \mathcal{P}_L$ the set $\mathcal{C}_y$ equals y. The definition of δ_L also entails that $\mathcal{R}_L$ is *saturated*, i.e., any addition to the transitions of δ_L would result in an automaton that recognizes a

superset of L. Thus, $\mathcal{R}_L$ is the unique state-minimal saturated RFTA recognizing L up to isomorphism (see [8]) and is suitable as a canonical description for L. There may be other RFTA recognizing L having the same number of states and less transitions – however, a state-minimal RFTA for L featuring the smallest possible number of transitions may not be unique.

2.3 Observation tables

We fix a regular tree language $L \subseteq \mathbb{T}_\Sigma$. The type of learning algorithm we consider tries to infer a canonical FTA for L from given information. This task is solved principally by means of an *observation table* in which the learner keeps track of the information obtained and processed so far. The rows of the table are labeled by trees, the columns are labeled by contexts.

Definition 7 *A triple $T = \langle S, E, obs\rangle$ containing two finite sets $S \subseteq \mathbb{T}_\Sigma$ and $E \subseteq \mathbb{C}_\Sigma$ with $\Box \in E$ is called an* observation table *if*

- *S is* subtree-closed*, i.e., if $f(s_1, \ldots, s_n) \in S$ implies $s_1, \ldots, s_n \in S$ for all $f(s_1, \ldots, s_n) \in \mathbb{T}_\Sigma$, and*
- *$obs : \mathbb{T}_\Sigma \times \mathbb{C}_\Sigma \longrightarrow \{0, 1\}$ is a total function with*

$$obs(s, e) = \begin{cases} 1 & \text{if } e[\![s]\!] \in L \text{ is confirmed,} \\ 0 & \text{if } e[\![s]\!] \notin L \text{ is confirmed.} \end{cases}$$

For trees $s, s' \in \mathbb{T}_\Sigma$ and a context $e \in \mathbb{C}_\Sigma$, we will say that

- *e is a* positive *context for s if $obs(s, e) = 1$,*
- *e is a* negative *context for s if $obs(s, e) = 0$, and that*
- *e is a* separating *context for s and s' if $obs(s, e) \neq obs(s', e)$.*

For $s \in \mathbb{T}_\Sigma$, let $row(s) := \{\langle e, obs(s, e)\rangle \mid e \in E\}$. Also, $row(X) := \{row(s) \mid s \in X\}$ for a set $X \subseteq \mathbb{T}_\Sigma$, and $row(s)(e) := obs(s, e)$ for $s \in \mathbb{T}_\Sigma$ and $e \in E$. We say that two rows $r_1, r_2 \in row(\mathbb{T}_\Sigma)$ are obviously different *and denote it by $r_1 <> r_2$ if there is a context $e \in E$ with $r_1(e) \neq r_2(e)$. We also say that two trees $s_1, s_2 \in \mathbb{T}_\Sigma$ are* obviously different *and denote it by $s_1 <> s_2$ if $row(s_1) <> row(s_2)$. For $row(s_1) = row(s_2)$ we also write $s_1 \approx s_2$.*

According to criteria proper to the respective type of learner the set S labeling the rows of the table is divided into two sets RED and BLUE with RED $\cup$ BLUE $= S$ and RED $\cap$ BLUE $= \emptyset$. Intuitively, for a pure query learner RED contains the elements the learner has already processed and set down to represent the constituents of his current hypothesis whereas BLUE contains elements the learner has already "in sight" and intends to consider next. During the learning process, elements are moved successively from BLUE to

RED and BLUE is filled up with further trees. A learner from membership queries and a positive sample fills the set RED with elements derived from the sample once at the beginning and does not change it anymore whereas the set BLUE is left empty throughout the process.

Let us fix an observation table $T = \langle S, E, obs\rangle$ with $S = \text{RED} \cup \text{BLUE}$ for the rest of this section.

Definition 8 *T is* closed *if* $\forall s \in \text{BLUE} : \exists s' \in \text{RED} : \neg(s <> s')$.
T is consistent *if, for all* $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$ *and* $1 \leq i \leq n$, $s_i \approx s'_i$ *implies* $f(s_1, \ldots, s_n) \approx f(s'_1, \ldots, s'_n)$.

From T we can construct an FTA $\mathcal{A}_T = \langle \Sigma, Q_T, F_T, \delta_T\rangle$ defined by

- $Q_T := row(\text{RED})$,
- $F_T := \{q \in Q_T \mid q(\Box) = 1\}$, and
- $\delta_T := \{\langle f, q_1 \cdots q_n, q\rangle \mid q_1, \ldots, q_n, q \in Q_T \wedge$
  $\exists s_1, \ldots, s_n, f(s_1, \ldots, s_n) \in S :$
  $\forall i \in \{1, \ldots, n\} : \neg(q_i <> row(s_i)) \wedge$
  $\neg(q <> row(f(s_1, \ldots, s_n)))\}$.

It follows directly from the definition of δ_T and of consistency that if T is consistent then $\mathcal{A}_T$ is deterministic. As S is subtree-closed, if T is closed then there is no $q \in Q_T$ with $\mathcal{L}_q = \emptyset$, i.e., all states can be reached.

Definition 9 *We say that $\mathcal{A}_T$ is* T-consistent *if, for all $s \in S$ and $e \in E$, we have $\mathcal{A}_T(e[\![s]\!]) = 1 \Leftrightarrow obs(s, e) = 1$.*

In any DFTA $\mathcal{A}$, for each state q the set $\mathcal{L}_q$ of trees ending up in q is a subset of some equivalence class under the relation $\equiv_{\mathcal{L}(\mathcal{A})}$ whereas in a non-deterministic FTA the set $\mathcal{L}_q$ may intersect several of those classes. Abstractly speaking, all learning algorithms appearing in this article can be conceived to start out with a provisional set of equivalence classes and then to try and converge to the partition induced by $\equiv_L$ on $\mathbb{T}_\Sigma$ by splitting up these classes according to the obtained information, which effectively translates into inferring a state-minimal DFTA $\mathcal{A}_L$ for L where for each state q the set $\mathcal{L}_q$ corresponds exactly to an equivalence class of L and no class is represented twice. The sets S and E were so named to indicate that the rows of S are candidates for *states* in a DFTA for L and that E contains *experiments* proving that two elements of S belong to distinct classes and should represent different states.

Finally, the finite samples that are given to a learner can have certain useful properties. Definition 10 is taken in a modified form from [5]:

Definition 10 *A finite set* $X_+ \subseteq \mathit{Subt}(L)$ *is* representative *for* L *if for every transition* $\langle f, q_1 \cdots q_n, q\rangle \in \delta_\circ$ *there is* $f(s_1, \ldots, s_n) \in \mathit{Subt}(X_+)$ *with* $\delta_\circ^*(s_i) = q_i$ *for* $1 \leq i \leq n$.

Intuitively, X_+ is representative for L if every transition of $\mathcal{A}_L^\circ$ is needed to assign states to all trees in X_+. Equivalently, for every tree $f(t_1, \ldots, t_n) \in \mathit{Subt}(L)$ there is $f(s_1, \ldots, s_n) \in \mathit{Subt}(X_+)$ such that $s_i \equiv_L t_i$ for $1 \leq i \leq n$.

Definition 11 *A finite set* $C \subseteq \mathbb{C}_\Sigma$ *is* separative *for* L *if for all* $t, t' \in \mathbb{T}_\Sigma$ *with* $\neg(t \equiv_L t')$ *there is* $e \in C$ *such that* $\neg(e[\![t]\!] \in L \Leftrightarrow e[\![t']\!] \in L)$.

Thus, a set is separative for L if for each pair of equivalence classes $\chi, \chi' \in \mathcal{E}_L$ with $\chi \neq \chi'$ it features a context by which the equality can be disproven.

3 Two settings, two canonical representations

In Subsection 3.1 we will describe two algorithms that infer a state-minimal DFTA for the target language in two different settings based on the learners given in [17] and [5]. In Subsection 3.2 we will attempt to reproduce those results for the canonical state-minimal RFTA.

Let L be the regular target tree language over some finite ranked alphabet. Let the table $T = \langle S, E, \mathit{obs}\rangle$ with $S = \text{RED} \cup \text{BLUE}$ maintained by the respective learner always be defined by the current values of its components which we assume to be visible as global variables throughout all procedures. We also assume that the learner is given the *smallest* alphabet Σ such that $L \subseteq \mathbb{T}_\Sigma$. A common procedure needed by all learners is UPDATE which fills in the cells of the table with the aid of a membership oracle $\mathcal{O}$:

```
procedure UPDATE
1     for s ∈ S do
2         for e ∈ E do
3             if O(e[[s]]) = 1 then obs(s, e) := 1; else obs(s, e) := 0.
```

3.1 The deterministic case

We consider two settings: In both settings the learner has access to a membership oracle. In addition, in one setting the learner is given a finite subset of the target language once at the beginning whereas in the other the learner has access to an equivalence oracle instead.

3.1.1 MQs and a positive sample

The algorithm ALTEX [5], given in our own modified version below, relies on a single check-and-mend mechanism: It builds an initial observation table from the given sample and checks if the table is consistent. While this is

Input: A set $X_+ \subseteq \mathbb{T}_\Sigma$, a membership oracle $\mathcal{O}$. **Output**: A DFTA.

1 $S := Subt(X_+)$; $E := Cont(X_+)$; RED $:= S$; BLUE $:= \emptyset$; UPDATE;
2 **while** T is not consistent **do**
3 find $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$, $e \in E$, $j \in \{1, \ldots, n\}$ with
4 $\forall i \in \{1, \ldots, n\} : s_i \approx s'_i \ \wedge$
5 $obs(f(s_1, \ldots, s_n), e) = 1 \ \wedge \ obs(f(s'_1, \ldots, s'_n), e) = 0 \ \wedge$
6 $\mathcal{O}(e[\![f(s_1, \ldots, s_n)^{\square}_j]\!][\![s'_j]\!]) = 0$;
7 $E := E \cup \{e[\![f(s_1, \ldots, s_n)^{\square}_j]\!]\}$; UPDATE;
8 **return** $\mathcal{A}_T$.

Figure 2: Our modified ALTEX

not the case the learner finds suitable elements causing an inconsistency in the table and from them constructs new separating contexts which are then added to E. The process stops when the table is consistent, and the learner returns the FTA represented by the resulting table as the solution. As consistency implies determinism the output is bound to be a DFTA.

We observe that there is a problem with the original version in [5]: The authors claim that in each loop execution their algorithm adds a new separating context to E. This is only true when certain inconsistencies fulfilling additional conditions are processed (see below). As a consequence, we must either transform their while-loop into a for-loop processing the inconsistencies found in the initial table one by one, or check in each execution if the inconsistency is suitable. Both corrections preserve the learner's polynomial complexity but increase its degree. We have chosen the second option, and the corrected code is displayed in Figure 2.

Assume the sample X_+ to be representative for L. If X_+ is representative for L then the set $S = Subt(X_+)$ must be as well, i.e., all equivalence classes of L and all transitions between them are represented in S. The following lemma is essential for the learner's progress because its claim constitutes a necessary condition to ensure that as long as the table is not consistent we can derive a separating context for two elements representing two different equivalence classes directly from the table itself (i.e., that the search in lines 3–6 will always succeed). It is a reformulated version of a corresponding one from [5], and the proof can be found in Appendix A.1. Let us call a pair of trees $s, s' \in S$ *deceiving* if $s \approx s'$ but $\neg(s \equiv_L s')$. We will say that an inconsistency is *simple* if it involves just one deceiving pair of direct subtrees as defined in the text of Lemma 2, and *multiple* otherwise.

Lemma 2 *As long as T is not consistent it contains a simple inconsistency, i.e, there are $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$ with $s_i \approx s'_i$ for all i with $1 \leq i \leq n$ and $\neg(f(s_1, \ldots, s_n) \approx f(s'_1, \ldots, s'_n))$ such that there is an index $j \in \{1, \ldots, n\}$ with $\neg(s_j \equiv_L s'_j)$ but $s_k \equiv_L s'_k$ for all other $k \in \{1, \ldots, n\} \setminus \{j\}$.*

Why does it work? Why are simple inconsistencies essential?

We will give some semi-formal explanations for the termination and correctness of our version of ALTEX, and also for the problems arising with the original version in [5], in preparation of the discussion of the residual case. Then we will prove the algorithm formally.

For the following explanations, recall that $\mathcal{E}_L$ is the set of all equivalence classes of L under $\equiv_L$, and let us define an *L-transition* as an expression $\epsilon := f(\chi_1, \ldots, \chi_n)$ with $f \in \Sigma_n$ for some $n \geq 1$, $\chi_j = \{\Box\}$ for some index $j \in \{1, \ldots, n\}$ and $\chi_i \in \mathcal{E}_L$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ such that there is $\chi \in \mathcal{E}_L$ with $\epsilon[\![\chi]\!] \subseteq \mathit{Subt}(L)$, where $\epsilon[\![\chi]\!]$ denotes the set

$$\{f(t_1, \ldots, t_n) \in \mathbb{T}_\Sigma \mid t_j \in \chi \wedge \forall i \in \{1, \ldots, n\} \setminus \{j\} : t_i \in \chi_i\}.$$

Observe that an L-transition can be naturally represented by a context that is obtained by instantiating χ_j by $\Box$ and each χ_i in ϵ for $i \in \{1, \ldots, n\} \setminus \{j\}$ with an arbitrary tree from χ_i.
The *application* of ϵ to a tree $s \in \mathit{Subt}(L)$ is the set $\epsilon[\![[s]_L]\!]$. The *behaviour* of $\epsilon[\![[s]_L]\!]$ with respect to a context $e \in \mathbb{C}_\Sigma$ is the truth value of $e[\![\epsilon[\![[s]_L]\!]]\!] \subseteq L$. We say that s has the L-transition ϵ (or that ϵ is an L-transition of s) if $\epsilon[\![[s]_L]\!] \subseteq \mathit{Subt}(L)$. In terms of the canonical DFTA $\mathcal{A}_L^\circ = \langle \Sigma, Q_\circ, F_\circ, \delta_\circ \rangle$ (without a failure state) this corresponds to the existence of a transition $\langle f, q_1 \cdots q_n, q \rangle \in \delta_\circ$ with $q_j = [s]_L$ and $q_i = \chi_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$. The behaviour of $\epsilon[\![[s]_L]\!]$ with respect to e is the truth of $\delta_\circ^+(q, e) \cap F_\circ \neq \emptyset$.

From here on, assume X_+ to be representative for L. While the table is not consistent it cannot represent the target automaton $\mathcal{A}_L$ since $\mathcal{A}_L$ is deterministic. Moreover, the authors of [5] show that with X_+ representative, as long as T is not consistent the FTA $\mathcal{A}_T$ recognizes a subset of L.

An inconsistency in the table caused by two elements $t = f(s_1, \ldots, s_n)$ and $t' = f(s'_1, \ldots, s'_n)$ in S with $s_i \approx s'_i$ for all $i \in \{1, \ldots, n\}$ but $\neg(t \approx t')$ implies that there must be at least one deceiving pair s_j, s'_j for some index $j \in \{1, \ldots, n\}$ for which there is a separating context exactly of depth $d + 1$ where d is the depth of some context $e' \in \mathbb{C}_\Sigma$ fulfilling $\mathit{obs}(t, e') = 1$ and $\mathit{obs}(t', e') = 0$. However, the actual ability of the learner to construct such a separating context for s_j and s'_j from the (finite!) table fundamentally relies on Lemma 2 which in turn relies on X_+ being representative for L. Lemma 2 implies that for each multiple inconsistency the table also features a simple inconsistency for each deceiving pair in it.

Changing the point of view, such a simple inconsistency can be seen as a comparison of the behaviour with respect to the contexts in E when the *same* L-transition (represented by e_2 in the proof of Lemma 2) is applied to the members of a deceiving pair s_j, s'_j – with the outcome that there is a difference. Note that as X_+ is representative for L the set S contains all L-transitions of s_j, i.e., all possible first steps from $[s_j]_L$ towards acceptance

in $\mathcal{A}_L$, and E contains a positive context for each of those L-transitions. This means that if an inconsistency involving two trees t, t' retrieved by the checks in lines 3–5 is indeed simple then the learner can pick the L-transition represented by $f(s_1, \ldots, s_n)^{\square}_j$ (and $f(s'_1, \ldots, s'_n)^{\square}_j$) and any separating context for t and t' to construct a separating context for s_j and s'_j. In other words, if we have chosen the right index j then any separating context for t and t' will fulfil the condition in line 6 and the learner is sure to add a new separating context to E in line 7. However, if the inconsistency is multiple then this is not ensured since in that case we do not compare the effect of applying the same L-transition to s_j and s'_j. This is why the algorithm as given in [5] may not terminate if in the while-loop the same inconsistency is chosen over and over again without eliminating it. Therefore we search for a suitable context e and index j in lines 3–6 right away, especially making sure that the context added to E would indeed be separating in line 6.
Clearly, when all simple inconsistencies are resolved there cannot remain any multiple inconsistencies in the table either. Since the set S is never modified only a finite number of rows in the table can be identical, and obviously by no addition to the elements of E can this number be increased. Thus, in our version the termination of the while-loop is ensured.

However, the mere achievement of consistency in a table constructed from some positive sample of L would not suffice to ensure its correctness: First of all, if the set of contexts labeling the columns of the final table is to be separative for L then we have yet to verify that for *any* pair of trees $s, s' \in \mathit{Subt}(L)$ fulfilling $\neg(s \equiv_L s')$ but $s \approx s'$ in the initial table there is a corresponding inconsistency which will then be resolved at some point during the process. The truth of this statement also relies on the condition that X_+ be representative for L and is shown by induction over the depth of separating contexts in the proof of Theorem 1 below. The basic idea of this induction is inspired by the original one in [1] for the simpler case of deterministic finite-state string automata in the same setting.

Theorem 1 *If X_+ is representative for L then ALTEX terminates and returns a DFTA which is isomorphic to $\mathcal{A}^{\circ}_L$.*

Proof. Assume that ALTEX performs m executions of the while-loop in total, and let $T_k = \langle S_k, E_k, \mathit{obs}\rangle$ be the table obtained after k executions for $k \geq 0$. Clearly we have $S_{k'-1} = S_{k'}$ and $E_{k'-1} \subsetneq E_{k'}$ for all k' with $1 \leq k' \leq m$. Note that the definition of the relation symbol $\approx$ differs depending on the current set E_k – we will write $\approx_k$ for disambiguation.

When the while-loop terminates, (a) T_m is consistent due to the termination criterion, and (b) the set E_m is separative for L, i.e., for any $t, t' \in \mathbb{T}_\Sigma$ with $\neg(t \equiv_L t')$ there is a context $e \in E_m$ with $\mathit{obs}(t, e) \neq \mathit{obs}(t', e)$.

(b): First of all, observe that if $t' \notin \mathit{Subt}(L)$ then either $t \in \mathit{Subt}(L)$ or t cannot fulfil $\neg(t \equiv_L t')$. In the former case, as X_+ is representative for L

and as we have $X_+ \subseteq L$ there is $s \in S_0$ with $s \equiv_L t$ and $e \in \mathit{Cont}(X_+) = E_0$ with $\mathit{obs}(s, e) = \mathit{obs}(t, e) = 1$, and thus $\neg(t \approx_m t')$ is ensured.

Let $t, t' \in \mathit{Subt}(L)$. As X_+ is representative for L there are $s, s' \in S_0$ with $s \equiv_L t$ and $s' \equiv_L t'$. We prove (b) by induction over separating contexts.

If $s \in L$ but $s' \notin L$ then the claim is true since we have $\Box \in \mathit{Cont}(X_+) = E_0$. If the canonical DFTA $\mathcal{A}_L^\circ$ contains a transition $\langle f, q_1 \cdots q_n, q\rangle \in \delta_\circ$ with $q_j = [s]_L$ for some $j \in \{1, \ldots, n\}$ but no $\langle f, q'_1 \cdots q'_n, q'\rangle \in \delta_\circ$ with $q'_j = [s']_L$ then the claim holds as well due to the fact that X_+ is representative for L and $X_+ \subseteq L$ which implies that there is $e \in \mathit{Cont}(X_+) = E_0$ and $e_1, e_2 \in \mathbb{C}_\Sigma$ with $e = e_1[\![e_2]\!]$ and $e_2 = f(s_1, \ldots, s_n)^{\Box}_j$ and $[s_i]_L = q_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ such that $\mathit{obs}(s, e) = 1$ and $\mathit{obs}(s', e) = 0$.

Remark Intuitively stated, if two elements do not share any L-transition featuring the same root symbol then they cannot be deceiving a priori. $\diamondsuit$

For the remaining cases, let $e_s \in \mathbb{C}_\Sigma$ be a separating context for s and s' with $\mathit{cdp}(e_s) = d+1$ for some $d \geq 0$, and choose $k \leq m$ such that we already have $\neg(s_k \approx_k s'_k)$ for all $s_k, s'_k \in S_k$ with $\neg(s_k \equiv_L s'_k)$ for which there is a separating context $e_k \in \mathbb{C}_\Sigma$ with $\mathit{cdp}(e_k) \leq d$. Moreover, assume $s \approx_k s'$.
Let there be a transition $\langle f, q_1 \cdots q_n, q\rangle \in \delta_\circ$ with $q_j = [s]_L$ for some $j \in \{1, \ldots, n\}$. Since we have $s \approx_k s'$ there is a context $e \in \mathit{Cont}(X_+) \subseteq E_k$ and $e_1, e_2 \in \mathbb{C}_\Sigma$ such that $e = e_1[\![e_2]\!]$ and $e_2 = f(s_1, \ldots, s_n)^{\Box}_j$ with $q_i = [s_i]_L$ for $i \in \{1, \ldots, n\}\setminus\{j\}$ fulfilling $\mathit{obs}(s, e) = \mathit{obs}(s', e) = 1$. With $\mathit{Cont}(X_+) \subseteq E_k$ we also have $e_1 \in E_k$. Since e is a positive context both for s and s' there are $f(t_1, \ldots, t_n), f(t'_1, \ldots, t'_n) \in S_k$ with $t_j \equiv_L s$ and $t'_j \equiv_L s'$ and $t_i \equiv_L s_i \equiv_L t'_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$. We have $\neg(f(t_1, \ldots, t_n) \approx_k f(t'_1, \ldots, t'_n))$ by the induction assumption which implies that T_k must be inconsistent, i.e., there must be $e' \in E_k$ with $\mathit{obs}(f(t_1, \ldots, t_n), e') \neq \mathit{obs}(f(t'_1, \ldots, t'_n), e')$.

Hence, as long as there is a pair $t, t' \in \mathit{Subt}(L)$ with $\neg(t \equiv_L t')$ but $t \approx t'$ the table cannot be consistent. By Lemma 2, as long as the table is not consistent one can derive a context from it that eliminates an identity between two rows of S. ALTEX will find such a context and add it to E. Since S is finite there can be only a finite number of such pairs and thus the while-loop terminates. As X_+ is representative for L and $S_m = \mathit{Subt}(X_+)$, on exiting the while-loop E_m must be separative for all of L.
The claim of Theorem 1 follows directly from the fact that X_+ is representative for L and the definition of $\mathcal{A}_T$ (also see Section 3.2.2 in [27]). ■

Naturally, the complexity of ALTEX crucially depends on the given sample.

Let $m_+ := \sum_{t \in X_+} |t|$ be the size of all trees in X_+ added up, which is also the maximal cardinality of $\mathit{Subt}(X_+)$ and of $\mathit{Cont}(X_+)$. Let ρ be the maximal rank in Σ with $\Sigma_\rho \neq \emptyset$. Assume X_+ to be representative for L. Then ρ is bounded by m_+, and the index I_L is bounded by ρ and thus by m_+ as well.

The initial table can contain at most m_+ elements in S and at most m_+ elements in E. In each loop execution we add a single context to E. This implies that the number of MQs needed to successively fill the final table is bounded by $m_+ \cdot 2m_+ \in O({m_+}^2)$. Moreover, an additional MQ is asked in line 6. Finding suitable elements may require $|S|^2 \cdot 2m_+ \cdot \rho \in O({m_+}^4)$ executions of that MQ, and as the loop itself is executed at most m_+ times, the overall number of additional MQs caused by the executions of that loop is bounded by a function in $O({m_+}^5)$. However, we observe that most cases fulfil $\rho \ll m_+$ and also $|Subt(X_+)| \ll m_+$, and that in line 3 we restrict ourselves to elements from S with the same root symbol.

Note that the authors of [5] do not try to identify the deceiving pair of subtrees in an inconsistency but add contexts for all indices in $\{1, \ldots, n\}$. We prefer to keep the table small and have shifted the query complexity of filling their table to the search for a suitable index j in the loop. This is unrelated to the issue of the missing check which we have added in line 6.

Remark As it is, our learner cannot reliably check equivalence under $\equiv_L$. However, we can eliminate the problem of identifying inconsistencies as simple and moreover significantly reduce the learner's complexity by imposing an even stronger condition on the given positive sample.
We define $mSubt(X_+) := \{s \in Subt(X_+) \mid \forall s' \in Subt(X_+) \cap [s]_L : s \preceq s'\}$ for some total ordering relation $\preceq$ on trees and we formulate the condition

$$\text{(A) } \Sigma(mSubt(X_+)) \cap Subt(L) \subseteq Subt(X_+).$$

In connection with representativeness, this condition ensures that if two non-equivalent elements $t, t' \in \mathbb{T}_\Sigma$ share an L-transition then the application of that L-transition to each of those trees is represented in S using the *same* subtrees at least once, and thus for each deceiving pair s_j, s'_j in the table we can find a pair of trees in S representing the results of applying the exact same L-transition to $[s_j]_L$ and $[s'_j]_L$, respectively. In other words, we can find a simple inconsistency where the non-deceiving pairs are not only equivalent but represented by the same trees. Note that we could even weaken condition (A) by merely requiring the existence of an arbitrary subset of $Subt(X_+)$ containing at least one representative for each equivalence class $\chi \in \mathcal{E}_L$ with $\chi \subseteq Subt(L)$ and fulfilling (A) instead of $mSubt(X_+)$ but we have chosen the set of minimal access trees for convenience.
We can then define a modified version of ALTEX that relies on condition (A) and searches for two trees $s = f(s_1, \ldots, s_n)$ and $s' = f(s'_1, \ldots, s'_n)$ in S with $s_j \approx s'_j$ for some j and $s_i = s'_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ but $\neg(s \approx s')$. This obviously represents a simple inconsistency and we can pick an arbitrary separating context for s and s' in order to construct a separating context for s_j and s'_j without having to ask any additional MQs. The number of MQs is thus reduced to the at most $2{m_+}^2$ needed to fill in the cells of the table.

```
3     while T is not consistent do
4         find f(s_1,...,s_n), f(s'_1,...,s'_n) ∈ S and e ∈ E such that
5'              ∃j ∈ {1,...,n} : s_j ≈ s'_j ∧ ∀i ∈ {1,...,n} \ {j} : s_i = s'_i ∧
6               obs(f(s_1,...,s_n), e) = 1 ∧ obs(f(s'_1,...,s'_n), e) = 0;
8         E := E ∪ {e⟦f(s_1,...,s_n)^□_j⟧}; UPDATE;
```

Condition (A) is inspired by a similar one in [32] where a learner (RPNI) learns from a positive and a negative sample without being allowed to ask MQs at all which implies that the learner has to rely on equivalence classes always being represented by the same trees as well. ♢

3.1.2 MQs and EQs

The algorithm inferring the state-minimal DFTA for a regular tree language from MQs and EQs given in [17] stays quite close to Angluin's LSTAR [2] for strings but adds some improvements for practical complexity.

Let the equivalence oracle $\mathcal{O}_{eq}$ return the undefined tree $\square$ if the learner's hypothesis is correct. The learner starts out with an initial table T_0 defined by $\text{RED} = \{a\}$ for some arbitrary $a \in \Sigma_0$, $\text{BLUE} = \Sigma(\text{RED})$, and $E = \{\square\}$. Provided that an EQ for $\mathcal{A}_{T_0}$ is answered in the negative the learner enters a loop in which it first establishes closedness and consistency by successively promoting elements from BLUE to RED and/or adding suitable elements to the table. When the table is both closed and consistent $\mathcal{A}_T$ is a state-minimal DFTA for $\mathcal{L}(\mathcal{A}_T)$, see [17]. If an EQ for this DFTA is still answered in the negative then the learner uses the given counterexample to modify the table in such a way that the learning process can be continued. The code of this learning algorithm is displayed in Figure 3.

It is easy to see that the learner will always succeed in establishing a closed table after less than I_L executions of the inner while-loop since there can be at most I_L distinct rows. As soon as the table is closed the learner checks for consistency. Considering the difficulties addressed in the previous section, let us give some explanations why in this setting the learner will always be able to establish consistency as well although we cannot be sure that at this stage the sets S and E have the same favourable properties as when they are derived from a representative sample of L.

All tables constructed by this learner fulfil the condition $\text{BLUE} = \Sigma(\text{RED})$. An inconsistency caused by two trees $t = f(s_1, \ldots, s_n)$ and $t' = f(s'_1, \ldots, s'_n)$ in S with $s_i \approx s'_i$ for all $i \in \{1, \ldots, n\}$ but $\neg(t \approx t')$ implies that there must be at least one deceiving pair s_j, s'_j for some $j \in \{1, \ldots, n\}$ for which there is a separating context of depth $d + 1$ where d is the depth of some context $e \in \mathbb{C}_\Sigma$ with $obs(t, e) = 1$ and $obs(t', e) = 0$. Necessarily, $s_i, s'_i \in \text{RED}$ for all $i \in \{1, \ldots, n\}$. How do we find a separating context for at least one deceiving pair? Consider the following argument from [17]: There must be an index

Input: Oracles $\mathcal{O}_{eq}$ and $\mathcal{O}$. **Output:** A DFTA.

1 $\textsc{red} := \{a\}$ for some $a \in \Sigma_0$; $\textsc{blue} := \Sigma(\textsc{red})$; $E := \{\Box\}$; UPDATE;
2 $c := \mathcal{O}_{eq}(\mathcal{A}_T)$;
3 **while** $c \neq \Box$ **do**
4 **while** T is not closed $\vee$ T is not consistent **do**
5 **if** T is not closed **then**
6 find $s \in \textsc{blue}$ such that $\forall s' \in \textsc{red} : s <> s'$; $\textsc{red} := \textsc{red} \cup \{s\}$;
7 $\textsc{blue} := \textsc{blue} \setminus \{s\}$; $\textsc{blue} := \textsc{blue} \cup \Sigma(\textsc{red})$; UPDATE;
8 **else** find $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$, $e \in E$, $j \in \{1, \ldots, n\}$
9 such that $\forall i \in \{1, \ldots, n\} \setminus \{j\} : s_i = s'_i \ \wedge \ s_j \approx s'_j \ \wedge$
10 $obs(f(s_1, \ldots, s_n), e) = 1 \ \wedge \ obs(f(s'_1, \ldots, s'_n), e) = 0$;
11 $E := E \cup \{e[\![f(s_1, \ldots, s_n)^{\Box}_j]\!]\}$; UPDATE;
12 **if** $c := \mathcal{O}_{eq}(\mathcal{A}_T) \neq \Box$ **then** $\textsc{red} := \textsc{red} \cup \text{MINIMIZE}(c)$;
13 $\textsc{blue} := (\textsc{blue} \setminus \textsc{red}) \cup \Sigma(\textsc{red})$; UPDATE;
14 **return** $\mathcal{A}_T$.

procedure MINIMIZE(c)
15 find $s \in \textsc{blue}$ such that $\exists e \in \mathbb{C}_\Sigma : e[\![s]\!] = c$;
16 **if** $\exists s' \in \textsc{red} : \neg(s <> s') \ \wedge \ \mathcal{O}(c) = \mathcal{O}(e[\![s']\!])$ **then return** MINIMIZE($e[\![s']\!]$);
17 **else return** s.

Figure 3: Learning DFTA from MQs and EQs

$k \in \{1, \ldots, n\}$ and a context e' with $obs(f(s_1, \ldots, s_k, s'_{k+1}, \ldots, s'_n), e') \neq obs(f(s_1, \ldots, s_{k-1}, s'_k, \ldots, s'_n), e')$, otherwise we would have $t \approx t'$ by letting k run from 1 to n.[1] The important point here is that all those trees created from t and t' by shifting the index k are also in S due to $\textsc{blue} = \Sigma(\textsc{red})$ and that $e' \in E$, and hence one of those pairs $f(s_1, \ldots, s_k, s'_{k+1}, \ldots, s'_n)$, $f(s_1, \ldots, s_{k-1}, s'_k, \ldots, s'_n)$ forms a simple inconsistency in the table, which moreover can be trivially identified as simple since all pairs of direct subtrees except one are identical. Thus, we can retrieve the necessary ingredients for a context reliably separating that pair s_k, s'_k directly from the table by picking the separating context e' and building $e'[\![f(s_1, \ldots, s_{k-1}, \Box, s'_{k+1}, \ldots, s'_n)]\!]$. Note that we do not even need an additional MQ to check if that context is indeed separating because the situation here has the same effect as condition (A) for the corrected ALTEX in the previous subsection.

Thus, as long as there are new equivalence classes within the reach of a single symbol, the learner continues building a representative sample on its own. Only if this is no longer the case the learner asks for a counterexample (line 12). On the receipt of a counterexample c we could either add $Subt(c)$

[1] This is the easiest way to formulate the argument but may mislead the intuition. Observe that when building t' from t by successively replacing a direct subtree s_i by s'_i in any order, there must be a consecutive pair of stages disrupting the $\approx$-relation.

to RED or $Cont(c)$ to E. The authors of [17] have developed a third option which serves to eliminate the influence of unnecessarily large counterexamples and which is here called procedure MINIMIZE. Basically, MINIMIZE simulates a parse of c by the current hypothesis automaton: It recursively replaces BLUE subtrees by RED ones with the same row provided that the property of being a counterexample stays preserved. As soon as there is no such RED element we know that the BLUE subtree is not yet represented as an equivalence class in RED and can add it. All three methods either result in a new state directly or at least in a new inconsistency (see Appendix A.2). MINIMIZE is based on the principle of *contradiction backtracing* which was described by Shapiro in [34]. For more details, see [17] and [18].

Thus, with the help of counterexamples in order to bridge gaps spanning more than one symbol, the learner successively builds a table in which each equivalence class of the target language L has exactly one representative in RED, and distinct equivalence classes have distinct rows. The authors of [17] show that the algorithm terminates after at most I_L executions of the outer loop and returns the total state-minimal DFTA $\mathcal{A}_L^\bullet$ for L.

The running time of this learner is bounded by $O((I_L{}^\rho + \zeta){I_L}^3)$ where ζ is the size of the biggest counterexamples received and ρ is the greatest rank in Σ with $\Sigma_\rho \neq \emptyset$, see [17].

3.1.3 Discussing complexity aspects

First of all, we observe that polynomial learning of a class of languages does not make sense without reference to a specific class of language descriptions because otherwise the learner's search cannot be adequately bounded.
In [13], de la Higuera discusses several notions of polynomial learnability, of which the most basic one can be summarized as follows.

Definition 12 *Let the underlying alphabet be fixed. A sample is* polynomial *with respect to a description if it is polynomial in size with respect to*

- *the number of states for DFA, and*
- *the number of rules multiplied by the longest rule for CFGs.*

Definition 13 *A class of descriptions is* polynomially characterizable *with respect to some learner if for each member of the class there is a polynomial sample that allows the learner to identify the target description correctly.*

Recall that in the case of trees we have defined the size of a sample as the total number of nodes in it. The algorithm ALTEX described in Subsection 3.1.1 can identify a regular tree language L from a representative sample in a polynomial number of steps with respect to the size of the sample. However, although such a sample can be built in polynomial time with respect

to the number of states in $\mathcal{A}_L^\circ$,[2] there are cases where the minimal size of such a sample is exponential with respect to the number of states in $\mathcal{A}_L^\circ$. The authors of [5] demonstrate this using the language containing a single k-ary tree of height l for $k, l \in \mathbb{N}$ where all inner nodes are labeled with a and all leaves with b. This tree has k^l leaves and the state-minimal DFTA is $\langle \{a, b\}, \{q_1, \ldots, q_l, q_F\}, \{q_F\}, \delta \rangle$ with $\Sigma_l = \{a\}$ and $\Sigma_0 = \{b\}$ and δ containing the transitions $\langle b, \langle\rangle, q_1 \rangle$ and $\langle a, q_i \cdots q_i, q_{i+1} \rangle$ for $1 \leq i \leq l-1$ and $\langle a, q_l \cdots q_l, q_F \rangle$. The size of the minimal representative sample is $k^{l+1} - 1$. Hence, state-minimal DFTA are not polynomially characterizable with respect to ALTEX in the sense of Definition 13.

The authors of [5] also remark that several distinct regular languages can have identical representative samples (see Appendix A.3 for an example) and observe that from this point of view it is not surprising that regular string languages are not learnable from positive data only, as shown by Gold [21]. Hence, for ALTEX it is vital to decompose the given trees and reassemble the parts, and then to be able to verify the membership status of the resulting trees with respect to the target language via membership queries. Note that we can use the MQs and EQs learner to build a representative input sample for ALTEX, namely $S[\![E]\!] \cap L$ for the final table $\langle S, E, obs \rangle$, which moreover would cause ALTEX to terminate immediately and successfully. Also note that we would not even have to intersect $S[\![E]\!]$ with L since for ALTEX any superset of a representative sample works equally well (due to MQs).

The authors of [17] observe that the behaviour of their MQs and EQs learner can only be considered as polynomial if the rank ρ is fixed and not taken as an input parameter. The exponential component stems from the complexity of filling up BLUE with all possible one-symbol extensions of RED. The learner ALTEX is spared this difficulty by the fact that it receives a representative sample and does not have to maintain a set BLUE. Trivially, if we would indicate the complexity of the MQs and EQs learner in terms of the number of possible(!) transitions given the number of states in the target automaton, the difficulty would disappear. This seems appropriate when one considers that Definition 13 also refers to the number of rules in a CFG and not to the number of nonterminals in it. There is an important parallel between CFGs and FTA which distinguishes them from DFA: In both formats, one side of each rule or transition may involve more than one component, i.e., nonterminal or state. However, we remark that since we are learning our descriptions based on the notion of syntactic equivalence, in contrast to the case of CFGs in general, in the case of FTA the number of options for the right-hand side is still finite.

[2] This building process rather depends on the number of transitions than on the number of states but since an FTA with $|Q|$ states has at most $|\Sigma| \cdot |Q|^{\rho+1}$ transitions there can be no exponential gap between those two numbers.

We also remark that while ALTEX directly returns the not necessarily total automaton $\mathcal{A}_L^\circ$ for L, the MQs and EQs learner returns the total $\mathcal{A}_L^\bullet$ and would have to eliminate a possible failure state *ex post*. However, there is at most one failure state in $\mathcal{A}_L^\bullet$ and thus this issue does not relate to the discussion of polynomial versus exponential behaviour.

3.2 The residual case

The following section presents tools and notions that serve to relate observation tables and RFTA, staying as close as possible to the case of DFTA.

3.2.1 Relating observation tables and RFTA

Definition 14 *For two trees $s, s' \in \mathbb{T}_\Sigma$ with $s^{-1}L \not\subseteq s'^{-1}L$ and a context $e \in \mathbb{C}_\Sigma$, we call e an* excluding *context for s and s' if $e \in s^{-1}L$ but $e \notin s'^{-1}L$.*

Definition 15 *A finite set $C \subseteq \mathbb{C}_\Sigma$ is* exclusive *for L if for all $t, t' \in \mathbb{T}_\Sigma$ with $t^{-1}L \not\subseteq t'^{-1}L$ there is $e \in C$ such that $e \in t^{-1}L \setminus t'^{-1}L$.*
A finite set $C \subseteq \mathbb{C}_\Sigma$ is p-exclusive *for L if for each $\gamma \in \mathcal{P}_L$ there is $e \in C$ such that $e \in \gamma$ but $e \notin \bigcup\{\gamma' \in \mathcal{C}_L \mid \gamma' \subsetneq \gamma\}$.*

A set is exclusive for L if for each pair of residual languages γ, γ' of L with $\gamma \not\subseteq \gamma'$ it features a context by which the inclusion can be disproven. A set is p-exclusive for L if for every prime residual language γ of L it features a context e by which it is possible to prove that γ is indeed prime.

For the following definitions (based on [7] where the string case is treated), we fix an observation table $T = \langle S, E, obs\rangle$ with $S = \text{RED} \cup \text{BLUE}$.

Definition 16 *Let $max(X) := x \in X \subseteq \{0,1\}$ such that $\forall x' \in X : x' \leq x$. For $r_1, r_2 \in row(\mathbb{T}_\Sigma)$ and $R \subseteq row(\mathbb{T}_\Sigma)$, we define a* join operation *by*

$r_1 \sqcup r_2 := \{\langle e, x\rangle \mid e \in E \wedge x = max(\{r_1(e)\} \cup \{r_2(e)\})\}$, *and*

$\bigsqcup R := \{\langle e, x\rangle \mid e \in E \wedge x = max(\{r(e) \mid r \in R\})\}$.

For two rows $r, r' \in row(\mathbb{T}_\Sigma)$ we say that r is covered *by r', and denote it by $r \sqsubseteq r'$, if $r(e) = 1$ implies $r'(e) = 1$ for all $e \in E$. For two trees $s, s' \in \mathbb{T}_\Sigma$, if $row(s) \sqsubseteq row(s')$ then we also write $s \sqsubseteq s'$. A row $r \in row(S)$ is* composed *if there are rows $r_1, \ldots, r_n \in row(S) \setminus \{r\}$ such that $r = \bigsqcup\{r_1, \ldots, r_n\}$. If r is not composed and there is at least one $e \in E$ with $r(e) = 1$ then r is said to be* prime. *For a set $X \subseteq \mathbb{T}_\Sigma$, we define $\mathcal{P}_X := \{r \in row(X) \mid r$ is prime$\}$.*

Definition 17 *T is* R-closed *if it fulfils $\mathcal{P}_{\text{BLUE}} \subseteq \mathcal{P}_{\text{RED}}$.*
T is R-consistent *if, for all $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$ and $1 \leq i \leq n$, $s_i \sqsubseteq s'_i$ implies $f(s_1, \ldots, s_n) \sqsubseteq f(s'_1, \ldots, s'_n)$.*

Closedness implies R-closedness whereas R-consistency implies consistency. Let us consider a small example to clarify the definitions given above.

		$\Box$	$f(\Box, b)$	$f(a, \Box)$
RED	a	0	1	0
	b	1	1	1
	$f(a,a)$	1	0	0
BLUE	$f(a,b)$	1	1	0
	$f(b,b)$	1	0	0

Figure 4: An example for an observation table and relations in it

Example 1 In the small observation table given in Figure 4,

- $row(f(a,b))$ is the only one that is not prime because it can be composed from $row(a)$ and $row(f(a,a))$ or $row(f(b,b))$,
- $row(b)$ covers all others, $row(f(a,b))$ covers $row(a)$, $row(f(a,a))$ and $row(f(b,b))$, and $row(f(a,a))$ and $row(f(b,b))$ cover each other.

Moreover, this table is R-closed because the only prime row in $row(\text{BLUE})$, $row(f(b,b))$, is also an element of $row(\text{RED})$ due to the fact that there is $f(a,a) \in \text{RED}$ with $row(f(a,a)) = row(f(b,b))$ but it is not R-consistent because $row(b)$ covers $row(a)$ but $row(f(b,b))$ does not cover $row(f(a,b))$.

From T we can derive an FTA $\mathcal{R}_T = \langle \Sigma, Q_T, F_T, \delta_T \rangle$ defined by

- $Q_T := \mathcal{P}_{\text{RED}}$,
- $F_T := \{r \in Q_T \mid r(\Box) = 1\}$, and
- $\delta_T := \{\langle f, q_1 \cdots q_n, q\rangle \mid q_1, \ldots, q_n, q \in Q_T \wedge \exists s_1, \ldots, s_n, f(s_1, \ldots, s_n) \in S : \forall i \in \{1, \ldots, n\} : q_i = row(s_i) \wedge q \sqsubseteq row(f(s_1, \ldots, s_n))\}$.

If T is R-consistent then the following holds: Consider $s_1, \ldots, s_n, s'_1, \ldots, s'_n \in \mathcal{P}_{\text{RED}}$ with $s_i \approx s'_i$ for $1 \leq i \leq n$. Clearly, we have $s_i \sqsubseteq s'_i$ and $s'_i \sqsubseteq s_i$. For any $f \in \Sigma_n$, if the trees $s = f(s_1, \ldots, s_n)$ and $s' = f(s'_1, \ldots, s'_n)$ are also in S then due to the R-consistency of T we have $s \sqsubseteq s'$ and $s' \sqsubseteq s$ and hence $s \approx s'$ as well such that $row(s)$ and $row(s')$ cover the same (prime) rows, i.e., states in Q_T. Conversely, if T is not R-consistent then different instantiations of $s_1, \ldots, s_n$ as representatives of $q_1, \ldots, q_n$ within the definition of δ_T may contribute different subsets of $\{q \in Q_T \mid \langle f, q_1 \cdots q_n, q\rangle \in \delta_T\}$.

R-closedness is needed to establish all prime rows of T as states in Q_T. In conjunction with R-closedness, R-consistency is a necessary (albeit not sufficient) condition if we want to ensure that the derived automaton $\mathcal{R}_T$ is

Input: A set $X_+ \subseteq \mathbb{T}_\Sigma$, a membership oracle $\mathcal{O}$. **Output**: An FTA.

1 $S := Subt(X_+)$; $E := Cont(X_+)$; RED $:= S$; BLUE $:= \emptyset$; UPDATE;
2 **while** T is not R-consistent **do**
3 find $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$ and $e \in E$ such that
4 $\exists j \in \{1, \ldots, n\} : s_j \sqsubseteq s'_j \ \wedge\ \forall i \in \{1, \ldots, n\} \setminus \{j\} : s_i \approx s'_i \ \wedge$
5 $obs(f(s_1, \ldots, s_n), e) = 1 \ \wedge\ obs(f(s'_1, \ldots, s'_n), e) = 0 \ \wedge$
6 $\mathcal{O}(e[\![f(s_1, \ldots, s_n)^{\square}_j]\!][\![s'_j]\!]) = 0$;
7 $E := E \cup \{e[\![f(s_1, \ldots, s_n)^{\square}_j]\!]\}$; UPDATE;
8 **for** $s \in S$ **do**
9 **if** $row(s) \notin \mathcal{P}_S \ \wedge\ \exists f(t_1, \ldots, t_n) \in S : \exists j \in \{1, \ldots, n\} : \exists e \in E :$
10 $t_j \approx s \ \wedge\ \mathcal{O}(e[\![f(t_1, \ldots, t_n)^{\square}_j]\!][\![s]\!]) = 1 \ \wedge\ \forall s' \in S \setminus \{s\} :$
11 $(s' \sqsubseteq s \ \wedge\ \neg(s' \approx s)) \ \Rightarrow\ \mathcal{O}(e[\![f(t_1, \ldots, t_n)^{\square}_j]\!][\![s']\!]) = 0$ **then**
12 $E := E \cup \{e[\![f(t_1, \ldots, t_n)^{\square}_j]\!]\}$; UPDATE;
13 **return** $\mathcal{R}_T$.

Figure 5: The algorithm RESI

an RFTA that recognizes the target language L. A first intuitive explanation runs as follows: As can be inferred from the presentation of the residual learning algorithms below, if the learning process succeeds then the covering relation $\sqsubseteq$ between the rows of T and the subset relation between the residual languages of L defined by the row labels coincide. An R-inconsistency reveals that some rows in T are still wrongly covered by others. As a consequence, due to the use of $\sqsubseteq$ in the definition of δ_T there may be trees $t \in \mathbb{T}_\Sigma$ for which $\delta^*_T(t)$ contains states q such that $\mathcal{C}_q$ does not constitute any residual language actually included in $t^{-1}L$, i.e., there may be contexts $e \in \mathcal{C}_q$ with $\mathcal{R}_T(e[\![t]\!]) = 1$ but $e[\![t]\!] \notin L$. Thus, R-consistency serves to prevent overgeneralization.

3.2.2 MQs and a positive sample – RESI

In this section we present a learning algorithm that tries to learn a regular tree language $L \subseteq \mathbb{T}_\Sigma$ from MQs and a finite positive sample of L based on the notion of residual languages and returns an FTA. If the given sample is representative then the FTA is isomorphic to $\mathcal{R}_L$. RESI is of polynomial complexity due to a technique based on the one used in Subsection 3.1.1 but which we have adapted to and verified for the more intricate residual case. The code of RESI is displayed in Figure 5.

Assume X_+ to be representative for L. Then due to the correspondence between the equivalence classes and the residual languages of L we can prove the following lemma by modifying the proof of Lemma 2. Lemma 3 implies that as long as the table is not R-consistent we can derive an excluding context for two individual candidates directly from the table itself (i.e., that the search in lines 3–6 will always succeed). See Appendix A.1 for the proof.

Let us now call a pair of trees $s, s' \in S$ *deceiving* if $s \sqsubseteq s'$ but $s^{-1}L \not\subseteq s'^{-1}L$. We will say that an R-inconsistency is *simple* if it involves just one deceiving pair and only genuinely equivalent pairs otherwise (as defined in Lemma 3).

Lemma 3 *As long as T is not R-consistent, there are trees $f(s_1, \ldots, s_n)$, $f(s'_1, \ldots, s'_n) \in S$ with $s_i \sqsubseteq s'_i$ for all i with $1 \leq i \leq n$ and $\neg(f(s_1, \ldots, s_n) \sqsubseteq f(s'_1, \ldots, s'_n))$ such that there is an index $j \in \{1, \ldots, n\}$ with ${s_j}^{-1}L \not\subseteq {s'_j}^{-1}L$ but $s_k \equiv_L s'_k$ for all other $k \in \{1, \ldots, n\} \setminus \{j\}$.*

An R-inconsistency in the table caused by two trees $t = f(s_1, \ldots, s_n)$ and $t' = f(s'_1, \ldots, s'_n)$ in S with $s_i \sqsubseteq s'_i$ for all $i \in \{1, \ldots, n\}$ but $\neg(t \sqsubseteq t')$ implies by Lemma 1 that there must be at least one deceiving pair s_j, s'_j for some index $j \in \{1, \ldots, n\}$ for which there is an excluding context of depth $d + 1$ where d is the depth of some context $e' \in E$ fulfilling $obs(t, e') = 1$ and $obs(t', e') = 0$. The ability of RESI to construct such a context from T relies on Lemma 3 which implies that for each R-inconsistency the table features a simple R-inconsistency for each of the deceiving pairs.

The rest of the argument runs parallel to the deterministic case as well: In order to add an excluding context in each loop execution we have to compare the effects of applying the same L-transition to a deceiving pair. Hence, RESI searches for a constellation that looks like a simple R-inconsistency, i.e., trees $t = f(s_1, \ldots, s_n)$, $t' = f(s'_1, \ldots, s'_n) \in S$ with $s_j \sqsubseteq s'_j$ for some $j \in \{1, \ldots, n\}$ and $s_i \approx s'_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ but $\neg(t \sqsubseteq t')$. However, like the ALTEX learner RESI has to check in addition if there is $e \in E$ such that $e[\![f(s_1, \ldots, s_n)^{\square}_j]\!]$ is actually excluding for s_j, s'_j since it might also be the case that $f(s_1, \ldots, s_n)^{\square}_j$ does not represent a suitable L-transition of s_j due to the fact that some of the other pairs are not equivalent – and for the same reason the pair s_j, s'_j might not be deceiving at all.

On the other hand, if the R-inconsistency retrieved in lines 4–5 is simple then the condition in line 6 will be fulfilled and RESI adds an excluding context to E in line 7. Clearly, when all simple R-inconsistencies are resolved there cannot remain any other R-inconsistencies in the table either. Since S is never modified there is only a finite number of possible covering relations between rows in the table, and by no addition to E can this number be increased. Thus, the termination of the while-loop is ensured.

As in the deterministic case, if the set of contexts labeling the columns of the final table is to be exclusive for L then we have yet to verify that for *any* pair of trees $s, s' \in Subt(L)$ fulfilling $s^{-1}L \not\subseteq s'^{-1}L$ but $s \sqsubseteq s'$ in the initial table there is a corresponding R-inconsistency which will then be resolved at some point during the process. This is shown by induction over the depth of excluding contexts in the proof of Theorem 2.

Moreover, even if E is exclusive for L after the while-loop has been exited E may not be p-exclusive for L and hence the FTA derived from the table at that point may not be isomorphic to $\mathcal{R}_L$. Therefore we add an additional

loop in lines 8–12 in order to retrieve all (non-prime) $s \in S$ for which we can find a suitable representative of an L-transition that has been applied to s in S and a matching positive context in E which when combined into another context and added to E cause the row of s to become prime in the table. Note that if E is exclusive for L then it is also separative for L which implies that we can translate the $\approx$-relation into the $\equiv_L$-relation and that thus any L-transition found in line 10 for some t_i with $t_i \approx s$ is an L-transition of s. If $s^{-1}L$ is indeed a prime residual language of L then s has an L-transition ϵ such that for all contexts $e \in \mathbb{C}_\Sigma$ with $e[\![\epsilon[\![[s]_L]\!]]\!] \subseteq L$ we have $e[\![\epsilon[\![[s']_L]\!]]\!] \nsubseteq L$ for all trees $s' \in \mathbb{T}_\Sigma$ with $s'^{-1}L \subsetneq s^{-1}L$. As X_+ is representative for L the set S contains a tree representing $\epsilon[\![[s]_L]\!]$, and E contains a positive context for that tree. As a consequence, RESI is sure to add enough contexts such that when the for-loop is exited there is an exact one-to-one correspondence between prime rows in the table and prime residual languages of L.

Theorem 2 *If X_+ is representative for L then RESI terminates and returns an FTA which is isomorphic to $\mathcal{R}_L$.*

Proof. See Appendix A.1.

Complexity:
Again, let m_+ be the sum of the sizes of all trees in X_+. The initial table contains at most m_+ elements in S and at most m_+ elements in E. In a worst case we may have to eliminate the covering relation between the rows of each pair of trees in S, and thus the number of while-loop executions needed to make E exclusive for L is bounded by ${m_+}^2$. In each execution of the while-loop a single context is added to E. In order to make E also p-exclusive for L we may have to extend E by another set of contexts whose cardinality is bounded by m_+. We assume that the results of MQs are stored so that no query has to be asked twice. This implies that the number of MQs needed to successively fill the final table is bounded by $m_+(m_+ + m_+^2 + m_+) \in O(m_+^3)$.

Moreover, we ask an additional MQs in line 6. Finding suitable elements may require $|S|^2 \cdot 2m_+ \cdot \rho \in O({m_+}^4)$ executions of that MQ, and thus the overall number of additional MQs caused by the executions of the while-loop is bounded by a function in $O({m_+}^6)$. Furthermore, another two MQs are asked in lines 10–11. During the for-loop the size of E is bounded by $2m_+ + {m_+}^2$, and thus the overall number of additional MQs caused by the executions of the for-loop is bounded by $|S|^3 \cdot \rho \cdot (2m_+ + {m_+}^2) \in O({m_+}^6)$.

Remark As in the deterministic case, we can define a modified version of RESI that relies on condition (A) from Subsection 3.1.1 and searches for two trees $s = f(s_1, \ldots, s_n)$ and $s' = f(s'_1, \ldots, s'_n)$ in S with $s_j \sqsubseteq s'_j$ for some j and $s_i = s'_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ but $\neg(s \sqsubseteq s')$. This obviously represents a simple R-inconsistency and we can pick an arbitrary excluding context for s and s' in order to construct an excluding context for s_j and s'_j without having to ask any additional MQs in the while-loop.

Moreover, we do not need additional MQs in the for-loop in order to identify and mark the representatives of prime residual languages of L either: Define $mr(S) := \{s \in S \mid \forall s' \in S : s' \approx s \Rightarrow s \preceq s'\}$. Since we rely on E being separative for L after the while-loop has been exited we assume that the $\approx$-relation and the $\equiv_L$-relation coincide and that hence the set $mr(S)$ contains exactly one minimal access tree for each equivalence class $\chi \in \mathcal{E}_L$ with $\chi \subseteq Subt(L)$, and that each element of $mr(S)$ is a minimal access tree for some equivalence class of L. Since we also rely on condition (A), for each $s \in mr(S)$ we look for a concrete tree $e'[\![s]\!] \in S$ as a representative for the application of some L-transition ϵ to s and a positive context e for $e'[\![s]\!]$ such that for all other $s' \in mr(S)$ with $s' \sqsubseteq s$ either $e'[\![s']\!]$ is not in S, which due to our conditions implies that s' does not have the L-transition ϵ at all, or we have $obs(s', e') = 0$, which implies that e is not a positive context for any representative of $\epsilon[\![s']\!]$. If s is prime then such an L-transition exists and the table contains a suitable tree $e'[\![s]\!]$ and a positive context e for $e'[\![s]\!]$, and obviously the addition of the context $e[\![e']\!]$ to E causes the row of s to become prime in the table.

---

2 **while** T `is not R-consistent` **do**
3 `find` $f(s_1, \ldots, s_n), f(s'_1, \ldots, s'_n) \in S$ `and` $e \in E$ `such that`
4' $\exists j \in \{1, \ldots, n\} : s_j \sqsubseteq s'_j \ \wedge\ \forall i \in \{1, \ldots, n\} \setminus \{j\} : s_i = s'_i \ \wedge$
5 $obs(f(s_1, \ldots, s_n), e) = 1 \ \wedge\ obs(f(s'_1, \ldots, s'_n), e) = 0$;
7 $E := E \cup \{e[\![f(s_1, \ldots, s_n)^{\square}_j]\!]\}$; `UPDATE;`
8' **for** $s \in mr(S)$ **do**
9' **if** $\exists f(t_1, \ldots, t_n) \in mr(S) \cup (\Sigma(mr(S)) \cap S) : \exists j \in \{1, \ldots, n\} : \exists e \in E :$
10' $t_j = s \ \wedge\ obs(f(t_1, \ldots, t_n), e) = 1 \ \wedge$
11' $\forall s' \in mr(S) \setminus \{s\} : s' \sqsubseteq s \ \Rightarrow\ obs(f(t_1, \ldots, t_n)^{\square}_j [\![s']\!], e) \neq 1$ **then**
12 $E := E \cup \{e[\![f(t_1, \ldots, t_n)^{\square}_j]\!]\}$; `UPDATE;`

Again, the only MQs needed are now those that serve to fill the table. ♢

3.2.3 MQs and EQs – MATRES

In [7], Bollig et al. have given an algorithm inferring RFSA for strings from MQs and EQs which stays as close as possible to Angluin's learner LSTAR [2] for DFA. We reproduce the switch from the deterministic to the residual approach for the tree case by presenting an algorithm that infers the canonical RFTA for a regular tree language in the same setting and is designed to stay as close as possible to the MQs and EQs learner from Subsection 3.1.2. The code of MATRES is displayed in Figure 6.

Given the definition of R-closedness, lines 4–6 are straightforward and can be compared to the corresponding lines in Figure 3. For R-consistency, we can argue in a similar way as before as well: All tables constructed by this learner fulfil BLUE $= \Sigma$(RED). An R-inconsistency caused by two trees

Input: Oracles $\mathcal{O}_{eq}$ and $\mathcal{O}$. **Output**: An FTA.

```
1  RED := {a} for some a ∈ Σ_0; BLUE := Σ(RED); E := {□}; UPDATE;
2  c := O_eq(R_T);
2  while c ≠ □ do
3     while T is not R-closed ∨ T is not R-consistent do
4        if T is not R-closed then
5           find s ∈ BLUE such that row(s) ∈ P_S \ P_RED; RED := RED ∪ {s};
6           BLUE := BLUE \ {s}; BLUE := BLUE ∪ Σ(RED); UPDATE;
7        else find f(s_1,...,s_n), f(s'_1,...,s'_n) ∈ S, e ∈ E, j ∈ {1,...,n}
8                 such that ∀i ∈ {1,...,n} \ {j} : s_i = s'_i ∧ s_j ⊑ s'_j ∧
9                           obs(f(s_1,...,s_n), e) = 1 ∧ obs(f(s'_1,...,s'_n), e) = 0;
10          E := E ∪ {e⟦f(s_1,...,s_n)^□_j⟧}; UPDATE;
11    if c := O_eq(R_T) ≠ □ then E := E ∪ Cont(c); RED := RED ∪ Subt(c);
12                                BLUE := (BLUE \ RED) ∪ Σ(RED); UPDATE;
13 return R_T.
```

Figure 6: The algorithm MATRES

$t = f(s_1, \ldots, s_n)$ and $t' = f(s'_1, \ldots, s'_n)$ in S with $s_i \sqsubseteq s'_i$ for all $i \in \{1, \ldots, n\}$ but $\neg(t \sqsubseteq t')$ implies that there must be a deceiving pair s_j, s'_j for some index $j \in \{1, \ldots, n\}$. Once again, we certainly have $\neg(f(s_1, \ldots, s_k, s'_{k+1}, \ldots, s'_n) \sqsubseteq (f(s_1, \ldots, s_{k-1}, s'_k, \ldots, s'_n))$ for some $k \in \{1, \ldots, n\}$, as otherwise we would have $t \sqsubseteq t'$ by letting k run from 1 to n. All the trees created from t and t' by shifting k are also elements of S due to BLUE $= \Sigma$(RED), and hence some pair $f(s_1, \ldots, s_k, s'_{k+1}, \ldots, s'_n), f(s_1, \ldots, s_{k-1}, s'_k, \ldots, s'_n)$ forms an obviously simple R-inconsistency in the table and has an excluding context $e \in E$. Thus, we can retrieve an excluding context for s_k, s'_k from the table, namely $e[\![f(s_1, \ldots, s_{k-1}, \Box, s'_{k+1}, \ldots, s'_n)]\!]$. As in Subsection 3.1.2, the situation here can be compared to the effects of condition (A).

Remark While filling up BLUE preserves the obvious differences between rows in RED (which is the only relevant aspect for the deterministic case), this may cause the row of a tree that has been promoted to RED because its row was prime at the time to become composed again. This means that we cannot even be sure which of the elements in RED will represent states of our final hypothesis and which will not before the process is completed. $\diamondsuit$

When T is R-closed and R-consistent the learner asks an EQ for $\mathcal{R}_T$. On the receipt of a counterexample c the learner adds all contexts that can be derived from c to E in order to create new states and/or to eliminate at least one more covering relation between rows of the table (see the proof of Theorem 4 below). Note that this may include covering relations between an individual row and a join of several rows, with the consequence that the individual row becomes prime. We also remark on the following facts:

- In the residual case, we have to add at least all those contexts of c that introduce a new distinction into the table due to the now admissible non-determinism in the learner's hypothesis.
- In the residual case, adding any number of subtrees of c to S would not ensure termination, as already shown for the special case of strings in [7]. This is due to the fact that even if all equivalence classes were represented by distinct rows in the table there might still be incorrect covering relations between them, which may cause the counterexample to stay a counterexample.
- Nonetheless, we add $Subt(c)$ to RED in order to achieve RED-composure for E, since it is a precondition of Theorem 3 and serves to ensure that our learner returns exactly the canonical RFTA.
- We could have refrained from adding the inconsequential contexts in $Cont(c)$ to E and we could have used a modified version of MINIMIZE replacing BLUE subtrees by RED ones with the same row as long as the property of being a counterexample stays preserved, in order to reduce the size of the table and to achieve a better average performance. The option above has been chosen for conceptual simplicity.

As soon as the while-loop has been exited MATRES returns the FTA $\mathcal{R}_T$. In order to show the correctness of MATRES we need some more lemmata and a theorem which have been proven in [27] (the proofs can be found in Appendix A.1). We fix an R-closed, R-consistent table $T = \langle S, E, obs \rangle$.

Theorem 3 *Assume T to fulfil the following three conditions.*

(1) E is RED*-composed, i.e.,*
for all $e \in E$ and all $s \in Subt(e)$, if $s \in \mathbb{T}_\Sigma$ then $s \in$ RED*,*

(2) T is saturated*, i.e.,* BLUE $= \Sigma($RED$)$*, and*

(3) $\mathcal{R}_T$ is T-consistent.

Then $\mathcal{R}_T$ is a state-minimal saturated RFTA.

Lemma 4 *We have $q \sqsubseteq row(s)$ for all $s \in S$ and all $q \in \delta_T^*(s)$.*

Lemma 5 *For all $q \in Q_T$ and all $e \in Cont(E)$ we have $q(e) = 1 \Leftrightarrow e \in \mathcal{C}_q$.*

Lemma 6 *For all $q, q' \in Q_T$ we have $q \sqsubseteq q' \Leftrightarrow \mathcal{C}_q \subseteq \mathcal{C}_{q'}$.*

Lemma 7 *For all $s \in S$ with $row(s) \in Q_T$ we have $row(s) \in \delta_T^*(s)$.*

Let ζ be the size of the biggest counterexamples received, and let ρ be the maximal rank in Σ. The proof below has been adapted from [7] to trees.

Theorem 4 *On termination MATRES has asked a number of EQs bounded by ${I_L}^2$, filled a number of cells bounded by $(\zeta I_L + (\zeta I_L)^\rho |\Sigma|) \cdot \zeta {I_L}^2$ via MQs, and returns an FTA which is isomorphic to $\mathcal{R}_L$.*

Proof. First of all, the output FTA is the state-minimal saturated RFTA for L by Theorem 3 since the components of T obviously fulfil conditions (1) and (2), and $\mathcal{R}_T$ recognizes L and thus ensures condition (3) due to the fact that the last EQ must have been answered in the positive.

We show that MATRES terminates, and that it asks at most ${I_L}^2$ EQs.

Define values $\alpha := |row(S)|$, $\beta := |row(\text{RED})|$, $\pi := |\mathcal{P}_S|$, and $\iota := |\{\langle r, r' \rangle \mid r, r' \in row(S) \wedge r \sqsubseteq r' \wedge r <> r'\}|$, i.e., ι is the number of row pairings such that the second row strictly covers the first.

We examine how these values evolve during a run of our algorithm. Clearly, the values of α, β, and π cannot increase above I_L. Moreover, α and β can never decrease as we do not delete elements of S or E, and no subsequent extension of the table can make two distinct rows identical again. Each new distinction causes at least one of those values to change. We will show that none of those values can change infinitely often and that each EQ leads to a new distinction of at most ${I_L}^2$ possible ones.

If T is not R-closed then after an execution of the inner while-loop where line 5 is entered β increases by 1. Simultaneously, α may increase by some measure $k > 0$ due to line 6, and the value of ι may increase by at most k times the old value of α (the maximal number of strict covering relations between new rows and old rows) plus $k(k-1)/2$ (the maximal number of strict covering relations between new rows).
If T is not R-consistent then after an execution of the inner while-loop where line 7 is entered β stays unchanged. However, α may increase by some measure $k' > 0$, and ι may increase by at most k' times the old value of α plus $k'(k'-1)/2$ (see above). If α does not increase then this means that any pair of elements $s, s' \in S$ with $s \approx s'$ in the old table still fulfils $s \approx s'$ in the new table. Thus, no new strict covering relation can have been introduced and ι cannot increase. However, since in that loop execution we have added a context to E *eliminating* a covering relation (recall the argument above) ι must decrease by at least 1.

When the inner while-loop terminates the current table $T = \langle S, E, obs \rangle$ is R-closed and R-consistent, and the learner submits an EQ for $\mathcal{R}_T$. If this results in a counterexample $c \neq \square$ then MATRES constructs a new table $T' = \langle S', E', obs \rangle$ with $S' = S \cup Subt(c)$ and $E' = E \cup Cont(c)$. Either α increases or it does not. If α increases by some measure $k'' > 0$ then ι may increase by at most k'' times the old value of α plus $k''(k''-1)/2$.
If α does not increase then ι cannot increase either (as explained in the previous paragraph). We add $Cont(c)$ to E. Note that T' must be R-consistent as the introduction of an R-inconsistency would entail an increase of α.

If c is a positive counterexample: Observe that for each symbol $a \in \Sigma_0$ and context $e \in E$ with $e[\![a]\!] \in L$ the tree $e[\![a]\!]$ is accepted by $\mathcal{R}_T$ due to the fact that $\delta_T^*(a) = \{q \in Q_T \mid q \sqsubseteq \mathit{row}(a)\}$ and Lemma 5. Thus, if we had $\mathit{Cont}(c) \subseteq E$ then $\mathcal{R}_T$ would accept c. Therefore, adding $\mathit{Cont}(c)$ to E must change the values of ι and/or π: Suppose that both values stay unchanged. Then $\mathcal{R}_T$ and $\mathcal{R}_{T'}$ would be isomorphic since α has not changed either. However, as we have $\mathit{Cont}(c) \subseteq E'$ the FTA $\mathcal{R}_{T'}$ must correctly accept c whereas $\mathcal{R}_T$ rejects it, a contradiction to our assumption.
If c is a negative counterexample: The fact that c is wrongly accepted by $\mathcal{R}_T$ implies that there is a context $e' \in \mathit{Cont}(c)$ and some state $q \in Q_T$ such that $e' \in \mathcal{C}_q$ but $q(e') = 0$. Moreover, we have $q(e) = 0 \Rightarrow e \notin \mathcal{C}_q$ for all $e \in E$ due to the "$\Leftarrow$"-direction in Lemma 5. Again, adding $\mathit{Cont}(c)$ to E must change the values of ι and/or π: Suppose that both stay unchanged. Then $\mathcal{R}_T$ and $\mathcal{R}_{T'}$ would be isomorphic since α has not changed either. However, as we have $e' \in E'$ the automaton $\mathcal{R}_{T'}$ must correctly reject c by Lemma 5 whereas $\mathcal{R}_T$ accepts it, a contradiction to our assumption.
Therefore, ι decreases or π increases or both.

In summary we observe that after each extension of the table, either (a) β is increased or (b) α is increased by some measure $k > 0$ and simultaneously ι is increased by at most $k\alpha + k(k-1)/2$ or (c) α stays the same and ι does not increase but ι decreases or π increases. Recall that the values of α, β, and π cannot increase beyond I_L. When α and β cannot increase anymore then ι decreases or π increases and hence MATRES terminates. Furthermore, we have shown that each EQ leads to a change of at least one of the values α, ι, or π. Since each of those changes indicates a new distinction of at most ${I_L}^2$ possible ones between rows the number of EQs is bounded by ${I_L}^2$.

Concerning MQs, we fill a table containing at most $m_S = \zeta I_L + (\zeta I_L)^\rho |\Sigma|$ elements in S and $m_E = \zeta {I_L}^2$ elements in E since for each distinction between two rows of the table at most ζ elements were added to E and, given that the distinction had to be obtained via an EQ, also to S. ■

Remark The authors of [7] show that for many example runs their learner needs a much smaller number of EQs than in the worst case given above (${I_L}^2$) and even than Angluin's theoretically superior learner LSTAR [2] (I_L). This is due to the fact that we do not need all equivalence classes in order to represent the canonical residual automaton. ♢

A very simple example run for our learners can be found in Appendix A.4.

4 Discussion

For the four learners studied above, the most important mechanism in order to make progress is the elimination of inconsistencies of the respective kind. During the learning process, the strategy of the learners for MQs and a finite sample is solely based on that operation whereas the learners for MQs and EQs use closure and counterexamples in between in order to introduce more elements into the table, thereby creating more inconsistencies (which in a broader sense includes obvious differences), the resolution of which will then reliably cause them to make progress towards the target.

The importance of consistency and closedness stems from the fact that they can be directly linked to specific properties of the underlying descriptions we are trying to learn, i.e., deterministic or residual automata in which all states are reachable. For an MQs and EQs learner, it is crucial that between EQs the learner must be able to compute its next hypothesis in a polynomially bounded number of steps with respect to I_L. An eligible strategy to ensure this seems to make sure that any hypothesis fulfils essential properties of the agreed canonical description before submitting it to the teacher – even if the oracle would also answer an EQ for an arbitrary FTA.

We remark that in the deterministic case the eventuality of inconsistencies can be avoided altogether, for example if we choose to process counterexamples by simply adding the contexts that can be derived from them to E such that the only way to promote elements to RED is by closure, with the outcome that all RED elements are pairwise obviously different at any time (see [27]). This is possible because in the case of DFTA a single representative for each state in RED is enough. In the residual case the table cannot be guaranteed to be R-consistent a priori: For the representation of an RFTA we generally have to allow and even need rows covering other rows in S and hence the preconditions for an R-inconsistency are given.

This brings us to the second most important mechanism for the polynomial progress of our learners, the acquisition of suitable candidates for states in S. This is also a point where one can hide a lot of exponentiality. The learners for MQs and a sample enjoy the luxury of having a representative sample built *for* them by the teacher, and if the number of steps needed to build the sample or its size are exponential with respect to I_L then this is "none of the learner's concerns" – the learner's complexity is specified with respect to the size of the given sample. Learners for MQs and EQs have to be more prudent: They fill up BLUE with new elements but only ever care to look one symbol further (an operation which is already exponential with respect to the maximal rank ρ), and if they cannot make progress anymore then they recur to a counterexample. Again, the size of the counterexample is none of the learner's concerns although it may be or even may have to be exponential with respect to I_L, and its size is taken mitigatingly into account when specifying the learner's worst case complexity.

Also mind the potentially exponential gap in size between the canonical deterministic and the canonical residual automaton. This is usually solved by indicating the complexity of learners for residual automata with respect to I_L as well, and not to the number of states in the target automaton. As far as we know, this "trick" has been used in order to achieve polynomiality by all results for non-deterministic automata so far, also see [35]. However, there is a new result [6] co-written by the author for the inference of universal string automata from MQs and EQs in polynomial time with respect to the size of the target automaton, which can be exponentially smaller than the corresponding DFA. Universal automata are yet another kind of canonical description for regular languages based on the substructure-context relation, and our result is based on the tool of an observation table as well.

In [27], we have also presented a learner for the inference of the canonical RFTA in a third setting not involving any queries, namely from a positive and a negative sample where the latter serves to control overgeneralization. In order to ensure the learner's success, the two finite samples have to fulfil certain properties which are even more restrictive than representativeness – however, in this setting the given positive sample does not even have to be fully representative for the target language as in general we do not need all equivalence classes to represent the canonical RFTA. For similar reasons, an MQs and EQs learner may even benefit from the absence of certain representatives since they may introduce misleading covering relations. Although the learner does not know which of the representatives in the current table are indispensable and which are not, due to the powerful device of an EQ the learner is notified immediately in case of success. As a consequence, in many individual cases the residual learner will terminate after having asked even less queries than its deterministic archetype (also see [7]).

The issue of polynomial inferability can be raised beyond regularity: An entire series of approaches towards the inference of context-free grammars (CFGs) based on the substructure-context relation has been developed by Alexander Clark and his coauthors, see [9, 10, 11, 36]. In the case of CFGs we face the problem that in general the number of equivalence classes under the Myhill-Nerode relation is infinite. However, positive results can still be achieved if each nonterminal of the target CFG can be characterized by some finite set of substructures or of contexts. In [37], we have been able to show that those approaches carry over to context-free tree grammars.
A learner that infers a CFG from MQs and EQs in polynomial time with respect to the size of the target grammar provided that each nonterminal in the CFG generates some equivalence class of the corresponding language (and can thus be characterized by the finite set of contexts that separate it from all other nonterminals) has been given in [9]. The authors remark that the complexity of a consistency check cannot be bounded polynomially as it may introduce strings of exponential length into the table but also that the check can be omitted since non-determinism is an admissible property

of the target CFG. However, for less restrictive characterizations polynomial results for MQs and EQs seem to be difficult to obtain, and we conjecture that this can be explained by the lack of polynomial characterizability as well, although the exact correlations remain yet to be spelled out.
In [27], we also sketch an algorithm for the inference of a context-free tree grammar (based on a draft with Alexander Clark and Ryo Yoshinaka for the string case) from a positive and a negative finite sample. We emphasize that this is possible without reference to a canonical description – as long as the sample is built with reference to the learner's inference strategy, and as long as the complexity of this building process is not taken into account.

We conclude that in order to obtain polynomial learning results beyond deterministic finite-state string automata we need to control the effects of the shift from strings to trees, which seems to introduce what we may call a vertical exponentiality (and reflects the relationship between the depth of a tree and the number of its leaves), and from deterministic automata to non-deterministic automata and to context-free grammars, which seems to introduce what we could call a horizontal exponentiality (and reflects the relationship between the equality of sets and the subset relation).

We remark that since strings can be seen as a special case of trees, our paper covers the setting represented by a dash in Figure 1. We also remark that other data structures than the observation table have been developed in order to improve the average complexity, see for example [19].

References

[1] D. Angluin. A note on the number of queries needed to identify regular languages. *Information and Control*, 51(1):76–87, 1981.

[2] D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

[3] D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

[4] D. Angluin. Negative results for equivalence queries. *Machine Learning*, 5:121–150, 1990.

[5] J. Besombes and J.-Y. Marion. Learning tree languages from positive examples and membership queries. *Theoretical Computer Science*, 382:183–197, 2007.

[6] J. Björklund, H. Fernau, and A. Kasprzik. Polynomial inference of universal automata from membership and equivalence queries. Technical report, 12-3, University of Trier, Germany, 2012.

[7] B. Bollig, P. Habermehl, C. Kern, and M. Leucker. Angluin-style learning of NFA. In *Online Proceedings of IJCAI 21*, 2009.

[8] J. Carme, R. Gilleron, A. Lemay, A. Terlutte, and M. Tommasi. Residual finite tree automata. In *Proceedings of DLT 2003*, volume 2710 of *LNCS*, pages 171–182. Springer, 2003.

[9] A. Clark. Distributional learning of some context-free languages with a minimally adequate teacher. In *Proceedings of ICGI 2010*, volume 6339 of *LNCS*, pages 24–37. Springer, 2010.

[10] A. Clark. Learning context-free grammars with the syntactic concept lattice. In *Proceedings of ICGI 2010*, volume 6339 of *LNCS*, pages 38–51. Springer, 2010.

[11] A. Clark. Towards general algorithms for grammatical inference. In *Proceedings of ALT 2010*, volume 6331 of *LNAI*, pages 11–30. Springer, 2010.

[12] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. *Tree Automata Techniques and Applications*. Online publication, 2007.

[13] C. de la Higuera. Characteristic sets for polynomial grammatical inference. *Machine Learning*, 27:125–138, 1997.

[14] C. de la Higuera. *Grammatical Inference: Learning Automata and Grammars*. Cambridge University Press, 2010.

[15] F. Denis, A. Lemay, and A. Terlutte. Learning regular languages using RFSA. In *Proceedings of ALT 2001*, volume 2225 of *LNCS*, pages 348–363. Springer, 2001.

[16] F. Denis, A. Lemay, and A. Terlutte. Residual finite state automata. *Fundamentae Informaticae*, 51:339–368, 2002.

[17] F. Drewes and J. Högberg. Learning a regular tree language from a teacher. In *Proceedings of DLT 2003*, volume 2710 of *LNCS*, pages 279–291. Springer, 2003.

[18] F. Drewes and J. Högberg. Query learning of regular tree languages: How to avoid dead states. *Theory of Computing Systems*, 40(2):163–185, 2007.

[19] F. Drewes, J. Högberg, and A. Maletti. MAT learners for tree series: an abstract data type and two realizations. *Acta Informatica*, 48:165–189, 2011.

[20] F. Gécseg and M. Steinby. *Tree Automata*. Akadémiai Kiadó, Budapest, Hungary, 1984.

[21] E. M. Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967.

[22] E. M. Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1978.

[23] J. E. Hopcroft and J. D. Ullmann. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Longman, 1990.

[24] A. Kasprzik. Learning residual finite-state automata using observation tables. Technical report, 09-3, University of Trier, 2009.

[25] A. Kasprzik. Learning residual finite-state automata using observation tables. In *Proceedings of DCFS 2010*, volume 31 of *EPTCS*, pages 205–212, 2010.

[26] A. Kasprzik. Learning residual finite-state tree automata from membership queries and finite positive data. Technical report, 11-1, University of Trier, 2011.

[27] A. Kasprzik. *Formal tree languages and their algorithmic learnability*. PhD thesis, University of Trier, Germany, 2012.

[28] S. Lange and S. Zilles. Formal language identification: Query learning vs. Gold-style learning. *Information Processing Letters*, 91:285–292, 2004.

[29] S. Lange and S. Zilles. Relations between Gold-style learning and query learning. *Information and Computation*, 203:211–237, 2005.

[30] O. Maler and A. Pnueli. On the learnability of infinitary regular sets. *Information and Computation*, 118(2):316–326, 1995.

[31] J. Oncina and P. García. Inference of recognizable tree sets. Technical report, DSIC II/47/93, Universidad de Valencia, 1993.

[32] J. Oncina and P. García. Identifying regular languages in polynomial time. In *Advances in Structural and Syntactic Pattern Recognition*, volume 5 of *Machine Perception and Artificial Intelligence*, pages 99–108. World Scientific, 2002.

[33] Y. Sakakibara. Learning context-free grammars from structural data in polynomial time. *Theoretical Computer Science*, 76(2–3):223–242, 1990.

[34] E. Y. Shapiro. *Algorithmic Program Debugging*. MIT Press, 1983.

[35] T. Yokomori. Learning non-deterministic finite automata from queries and counterexamples. In *Machine Intelligence 13*, pages 169–189, 1994.

[36] R. Yoshinaka. Towards dual approaches for learning context-free grammars based on syntactic concept lattices. In *Proceedings of DLT 2011*, volume 6795 of *LNCS*, pages 429–440. Springer, 2011.

[37] R. Yoshinaka and A. Kasprzik. Distributional learning of simple context-free tree grammars. In *Proceedings of ALT 2011*, volume 6925 of *LNAI*, pages 398–412. Springer, 2011.

A Appendix

A.1 Proofs

Lemma 2 *As long as T is not consistent it contains a simple inconsistency, i.e, there are $f(s_1,\dots,s_n), f(s'_1,\dots,s'_n) \in S$ with $s_i \approx s'_i$ for all i with $1 \leq i \leq n$ and $\neg(f(s_1,\dots,s_n) \approx f(s'_1,\dots,s'_n))$ such that there is an index $j \in \{1,\dots,n\}$ with $\neg(s_j \equiv_L s'_j)$ but $s_k \equiv_L s'_k$ for all other $k \in \{1,\dots,n\} \setminus \{j\}$.*

Proof. We prove this by a contradiction. Assume $e \in \mathbb{C}_\Sigma$ to be a context such that $e[\![s]\!] \in L$ and $e[\![s']\!] \notin L$ for some $s, s' \in S$ with $s \approx s'$ but $\neg(s \equiv_L s')$ (which must exist due to the fact that T is not consistent and to the definition of $\equiv_L$) and the depth $cdp(e)$ to be minimal. The fact that e cannot be in E but $\Box$ is implies $e \neq \Box$ and thus there are contexts $e_1, e_2 \in \mathbb{C}_\Sigma$ with $e_1[\![e_2]\!] = e$ and $e_2 = f(s_1,\dots,s_n)^{\Box}_j$ for some trees $s_1,\dots,s_n \in \mathbb{T}_\Sigma$, $f \in \Sigma_n$, and $n \geq 1$. Clearly, $cdp(e_1) < cdp(e)$. Since we require the depth of e to be minimal the context e_1 cannot fulfil the role of a separating context for any pair of trees $t, t' \in S$ with $t \approx t'$ but $\neg(t \equiv_L t')$.
As $e[\![s]\!] = e_1[\![e_2[\![s]\!]]\!]$ is in L we know that $e_2[\![s]\!]$ is in $Subt(L)$, and as X_+ is representative for L there is an element $t = f(t_1,\dots,t_n) \in S$ with $t_i \equiv_L s_i$ for all $i \in \{1,\dots,n\} \setminus \{j\}$ and $t_j \equiv_L s$ and $t \equiv_L e_2[\![s]\!]$. Since $t \in Subt(X_+)$ there is a context $e_3 \in Cont(X_+) \subseteq E$ such that $e_3[\![t]\!] \in X_+ \subseteq L$. Moreover, the context $e_3[\![e_4]\!]$ with $e_4 = f(t_1,\dots,t_n)^{\Box}_j$ is also in $Cont(X_+) \subseteq E$. Obviously, $e_3[\![e_4[\![s]\!]]\!] \in L$ and $obs(s, e_3[\![e_4]\!]) = 1$. The precondition $s \approx s'$ implies that we also have $obs(s', e_3[\![e_4]\!]) = 1$ and $e_3[\![e_4[\![s']\!]]\!] \in L$.
Note that $e_4[\![s']\!] \equiv_L e_2[\![s']\!]$ and hence $e_2[\![s']\!]$ must be a subtree of L as well. The fact that e is separating for s and s' entails $\neg(e_2[\![s]\!] \equiv_L e_2[\![s']\!])$. Again, as X_+ is representative for L there is $t' = f(t'_1,\dots,t'_n) \in S$ with $t'_i \equiv_L t_i$ for all $i \in \{1,\dots,n\} \setminus \{j\}$ and $t'_j \equiv_L s'$ and $t' \equiv_L e_2[\![s']\!]$. In contradiction to the claim, let us suppose $f(t_1,\dots,t_n) \approx f(t'_1,\dots,t'_n)$. In that case the context e_1 would fulfil the role of a separating context for the pair t, t' with $t \approx t'$ but $\neg(t \equiv_L t')$ since with $t \equiv_L e_2[\![s]\!]$ and $t' \equiv_L e_2[\![s']\!]$ we have $e_1[\![t]\!] \in L$ but

$e_1[\![t']\!] \notin L$. However, this would violate the assumption that the depth of e fulfilling such a role be minimal and thus the claim is proven. ■

Lemma 3 *As long as T is not R-consistent, there are trees $f(s_1, \ldots, s_n)$, $f(s'_1, \ldots, s'_n) \in S$ with $s_i \sqsubseteq s'_i$ for all i with $1 \leq i \leq n$ and $\neg(f(s_1, \ldots, s_n) \sqsubseteq f(s'_1, \ldots, s'_n))$ such that there is an index $j \in \{1, \ldots, n\}$ with ${s_j}^{-1}L \not\subseteq {s'_j}^{-1}L$ but $s_k \equiv_L s'_k$ for all other $k \in \{1, \ldots, n\} \setminus \{j\}$.*

Proof. We prove this by a contradiction. Assume $e \in \mathbb{C}_\Sigma$ to be a context such that $e[\![s]\!] \in L$ and $e[\![s']\!] \notin L$ for some $s, s' \in S$ with $s \sqsubseteq s'$ but $s^{-1}L \not\subseteq s'^{-1}L$ (which must exist due to the fact that T is not R-consistent and Lemma 1) and the depth $cdp(e)$ to be minimal. The fact that e cannot be in E but $\Box$ is implies $e \neq \Box$ and thus there are contexts $e_1, e_2 \in \mathbb{C}_\Sigma$ with $e_1[\![e_2]\!] = e$ and $e_2 = f(s_1, \ldots, s_n)^{\Box}_j$ for some trees $s_1, \ldots, s_n \in \mathbb{T}_\Sigma$, $f \in \Sigma_n$, and $n \geq 1$. Clearly, $cdp(e_1) < cdp(e)$. Since we require the depth of e to be minimal the context e_1 cannot fulfil the role of an excluding context for any pair of trees $t, t' \in S$ with $t \sqsubseteq t'$ but $t^{-1}L \not\subseteq t'^{-1}L$.
As $e[\![s]\!] = e_1[\![e_2[\![s]\!]]\!]$ is in L we know that $e_2[\![s]\!]$ is in $Subt(L)$, and as X_+ is representative for L there is an element $t = f(t_1, \ldots, t_n) \in S$ with $t_i \equiv_L s_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ and $t_j \equiv_L s$ and $t \equiv_L e_2[\![s]\!]$. Since $t \in Subt(X_+)$ there is a context $e_3 \in Cont(X_+) \subseteq E$ such that $e_3[\![t]\!] \in X_+ \subseteq L$. Moreover, the context $e_3[\![e_4]\!]$ with $e_4 = f(t_1, \ldots, t_n)^{\Box}_j$ is also in $Cont(X_+) \subseteq E$. Obviously, $e_3[\![e_4[\![s]\!]]\!] \in L$ and $obs(s, e_3[\![e_4]\!]) = 1$. The precondition $s \sqsubseteq s'$ implies that we also have $obs(s', e_3[\![e_4]\!]) = 1$ and $e_3[\![e_4[\![s']\!]]\!] \in L$.
Note that $e_4[\![s']\!] \equiv_L e_2[\![s']\!]$ and hence $e_2[\![s']\!]$ must be a subtree of L as well. The fact that e is excluding for s and s' entails $e_2[\![s]\!]^{-1}L \not\subseteq e_2[\![s']\!]^{-1}L$. Again, as X_+ is representative for L there is $t' = f(t'_1, \ldots, t'_n) \in S$ with $t'_i \equiv_L t_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ and $t'_j \equiv_L s'$ and $t' \equiv_L e_2[\![s']\!]$. In contradiction to the claim, let us suppose $f(t_1, \ldots, t_n) \sqsubseteq f(t'_1, \ldots, t'_n)$. In that case the context e_1 would fulfil the role of an excluding context for the pair t, t' with $t \sqsubseteq t'$ but $t^{-1}L \not\subseteq t'^{-1}L$ since with $t \equiv_L e_2[\![s]\!]$ and $t' \equiv_L e_2[\![s']\!]$ we have $e_1[\![t]\!] \in L$ but $e_1[\![t']\!] \notin L$. However, this would violate the assumption that the depth of e fulfilling such a role be minimal and thus the claim is proven. ■

Theorem 2 *If X_+ is representative for L then RESI terminates and returns an FTA which is isomorphic to $\mathcal{R}_L$.*

Proof. Assume that RESI performs m executions of the while-loop in total, and let $T_k = \langle S_k, E_k, obs \rangle$ be the table obtained after k executions for $k \geq 0$. Clearly we have $S_{k'-1} = S_{k'}$ and $E_{k'-1} \subsetneq E_{k'}$ for all k' with $1 \leq k' \leq m$. Note that the definition of the relation symbol $\sqsubseteq$ differs depending on the set E_k currently in question – we will write $\sqsubseteq_k$ for disambiguation.

When the while-loop terminates, (a) T_m is R-consistent due to the termination criterion, and (b) the set E_m is exclusive for L, i.e., for any $t, t' \in \mathbb{T}_\Sigma$ with $t^{-1}L \not\subseteq t'^{-1}L$ there is $e \in E_m$ with $obs(t, e) = 1$ but $obs(t', e) = 0$.

(b): First of all, observe that if $t \notin Subt(L)$ then there is no $t' \in \mathbb{T}_\Sigma$ such that $t^{-1}L \not\subseteq t'^{-1}L$. If $t' \notin Subt(L)$ then either $t \in Subt(L)$ or t cannot fulfil $t^{-1}L \not\subseteq t'^{-1}L$. In the former case, as X_+ is representative for L and as we have $X_+ \subseteq L$ there is $s \in S_0$ with $s \equiv_L t$ and $e \in Cont(X_+) = E_0$ with $obs(s, e) = obs(t, e) = 1$, and thus $\neg(t \sqsubseteq_m t')$ is ensured.

Let $t, t' \in Subt(L)$. As X_+ is representative for L there are $s, s' \in S_0$ with $s \equiv_L t$ and $s' \equiv_L t'$. We prove (b) by induction over excluding contexts.

If $s \in L$ but $s' \notin L$ then the claim is true since we have $\Box \in Cont(X_+) = E_0$. If the canonical DFTA $\mathcal{A}^\circ_L$ contains a transition $\langle f, q_1 \cdots q_n, q\rangle \in \delta_\circ$ with $q_j = [s]_L$ for some $j \in \{1, \ldots, n\}$ but no $\langle f, q'_1 \cdots q'_n, q'\rangle \in \delta_\circ$ with $q'_j = [s']_L$ then the claim holds as well due to the fact that X_+ is representative for L and $X_+ \subseteq L$ which implies that there is $e \in Cont(X_+) = E_0$ and $e_1, e_2 \in \mathbb{C}_\Sigma$ with $e = e_1[\![e_2]\!]$ and $e_2 = f(s_1, \ldots, s_n)^\Box_j$ and $[s_i]_L = q_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ such that $obs(s, e) = 1$ and $obs(s', e) = 0$.

For the remaining cases, let $e_s \in \mathbb{C}_\Sigma$ be an excluding context for s and s' with $cdp(e_s) = d+1$ for some $d \geq 0$, and choose $k \leq m$ such that we already have $\neg(s_k \sqsubseteq_k s'_k)$ for all $s_k, s'_k \in S_k$ with ${s_k}^{-1}L \not\subseteq {s'_k}^{-1}L$ for which there is an excluding context $e_k \in \mathbb{C}_\Sigma$ with $cdp(e_k) \leq d$. Moreover, assume $s \sqsubseteq_k s'$. Let there be a transition $\langle f, q_1 \cdots q_n, q\rangle \in \delta_\circ$ with $q_j = [s]_L$ for some $j \in \{1, \ldots, n\}$. Since we have $s \sqsubseteq_k s'$ there is a context $e \in Cont(X_+) \subseteq E_k$ and $e_1, e_2 \in \mathbb{C}_\Sigma$ such that $e = e_1[\![e_2]\!]$ and $e_2 = f(s_1, \ldots, s_n)^\Box_j$ with $q_i = [s_i]_L$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ fulfilling $obs(s, e) = obs(s', e) = 1$. With $Cont(X_+) \subseteq E_k$ we also have $e_1 \in E_k$. Since e is a positive context both for s and s' there are $f(t_1, \ldots, t_n), f(t'_1, \ldots, t'_n) \in S_k$ with $t_j \equiv_L s$ and $t'_j \equiv_L s'$ and $t_i \equiv_L s_i \equiv_L t'_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$. We have $\neg(f(t_1, \ldots, t_n) \sqsubseteq_k f(t'_1, \ldots, t'_n))$ by the induction assumption which implies that T_k must be R-inconsistent, i.e., there must be $e' \in E_k$ with $obs(f(t_1, \ldots, t_n), e') = 1$ but $obs(f(t'_1, \ldots, t'_n), e') = 0$.

Hence, as long as there is a pair $t, t' \in Subt(L)$ with $t^{-1}L \not\subseteq t'^{-1}L$ but $t \sqsubseteq t'$ the table cannot be R-consistent. By Lemma 3, as long as the table is not R-consistent one can derive a context from it that eliminates a covering relation between two rows of S. RESI will find such a context and add it to E. Since S is finite there can be only a finite number of such pairs and thus the while-loop terminates. As X_+ is representative for L and $S_m = Subt(X_+)$, on exiting the while-loop E_m must be exclusive for all of L.

Lines 10–15: For every $t \in \mathbb{T}_\Sigma$ with $t^{-1}L \in \mathcal{P}_L$ the canonical DFTA $\mathcal{A}^\circ_L$ features a transition $\langle f, q_1 \cdots q_n, q\rangle \in \delta_\circ$ with $q_j = [t]_L$ for some $j \in \{1, \ldots, n\}$ such that for no $t' \in \mathbb{T}_\Sigma$ with $t'^{-1}L \subsetneq t^{-1}L$ there is $\langle f, q'_1 \cdots q'_n, q'\rangle \in \delta_\circ$ with $q_j = [t']_L$ and $q'_i = q_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$.

As X_+ is representative for L there is $f(s_1,\ldots,s_n) \in S$ with $s_j \equiv_L t$ and $s_i \in q_i$ for $i \in \{1,\ldots,n\} \setminus \{j\}$, and there is $e \in E$ with $obs(f(s_1,\ldots,s_n),e) = 1$. Obviously, the context $e[\![f(s_1,\ldots,s_n)^{\Box}_j]\!]$ is such that it causes the row of s to be prime when included in E. As E_m is exclusive for L in the present table the $\equiv_L$-relation and the $\approx$-relation coincide and RESI is able to construct such a context and add it to E. Therefore, the final set E is also p-exclusive for L and we obtain $s \in \mathcal{P}_S \Leftrightarrow s^{-1}L \in \mathcal{P}_L$ for all elements $s \in S$.
The claim of Theorem 2 follows from the fact that X_+ is representative for L and the definition of $\mathcal{R}_T$ (for a more explicit explanation, see [27]). ■

Theorem 3 *Assume T to be R-closed and R-consistent and to fulfil:*

(1) E is RED*-composed,*

(2) T is saturated*, i.e.,* BLUE $= \Sigma($RED$)$*, and*

(3) $\mathcal{R}_T$ is T-consistent.

Then $\mathcal{R}_T$ is a state-minimal saturated RFTA.

Lemma 4 *We have $q \sqsubseteq row(s)$ for all $s \in S$ and all $q \in \delta^*_T(s)$.*

Proof. By induction over the depth of s. If $s = a$ for some $a \in \Sigma_0$ then the claim follows directly from the definition of δ_T. Let $s = f(s_1,\ldots,s_n)$. Then $\delta^*_T(s) = \{q \in Q_T \mid \exists\langle f, q_1 \cdots q_n, q\rangle \in \delta_T : \forall i \in \{1,\ldots,n\} : q_i \in \delta^*_T(s_i)\}$. Let $q \in \delta^*_T(s)$ and assume the claim to hold for $s_1,\ldots,s_n$. By the definition of δ_T there are trees $s'_1,\ldots,s'_n \in S$ and $q_i \in \delta^*_T(s_i)$ such that $q_i = row(s'_i)$ for $1 \le i \le n$ and $q \sqsubseteq row(f(s'_1,\ldots,s'_n))$. By the induction assumption we have $q_i \sqsubseteq row(s_i)$. This yields $s'_i \sqsubseteq s_i$, and also $f(s'_1,\ldots,s'_n) \sqsubseteq f(s_1,\ldots,s_n)$ due to the R-consistency of T and hence $q \sqsubseteq row(f(s_1,\ldots,s_n)) = row(s)$. ■

For Lemmata 5–6 assume that T fulfils conditions (1)–(3) as stipulated in Theorem 3. Intuitively, Lemma 5 states the following: Provided that we have reached a certain state of Q_T then $\mathcal{R}_T$ correctly classifies the contexts and their subcontexts in E. Recall that we had defined $q(e) := obs(s,e)$ for $q \in row(\mathbb{T}_\Sigma)$, $e \in E$, and any $s \in \mathbb{T}_\Sigma$ with $row(s) = q$.

Lemma 5 *For all $q \in Q_T$ and all $e \in Cont(E)$ we have $q(e) = 1 \Leftrightarrow e \in \mathcal{C}_q$.*

Proof. Let $q \in Q_T$. We prove this by induction over the depth of e.
For $e = \Box$ we have $q(\Box) = 1 \Leftrightarrow q \in F_T \Leftrightarrow \Box \in \mathcal{C}_q$, and the claim is shown.

Let $e = e'[\![e'']\!]$ with $e', e'' \in \mathbb{C}_\Sigma$ and $e'' = f(s_1,\ldots,s_n)$ such that $s_j = \Box$ for some $j \in \{1,\ldots,n\}$, and assume the claim to hold for e'.
Let $s \in S$ with $row(s) = q$. Note that we have $(\{s_1,\ldots,s_n\} \setminus \{s_j\}) \subseteq S$ due to condition (1) and $e''[\![s]\!] \in S$ due to condition (2).

"⇒": Let $q(e) = 1$. Since $\mathcal{R}_T$ is T-consistent there has to be a transition $\langle f, q_1 \cdots q_n, q' \rangle \in \delta_T$ with states $q_1, \ldots, q_n, q' \in Q_T$ such that $q_i \in \delta_T^*(s_i)$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $q_j \in \delta_T^*(s)$ and $e' \in \mathcal{C}_{q'}$. Let $s'_1, \ldots, s'_n \in \text{RED}$ with $row(s'_i) = q_i$ for $1 \leq i \leq n$. We have $f(s'_1, \ldots, s'_n) \in S$ by condition (2), $s'_i \sqsubseteq s_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ and $s'_j \sqsubseteq s$ due to Lemma 4, and $q' \sqsubseteq row(f(s'_1, \ldots, s'_n))$ due to the definition of δ_T. By condition (2) there is also an element $f(t'_1, \ldots, t'_n) \in S$ with $t'_i = s'_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $t'_j = s$ and we have $row(f(s'_1, \ldots, s'_n)) \sqsubseteq row(f(t'_1, \ldots, t'_n))$ since T is R-consistent. This entails $q' \sqsubseteq row(f(t'_1, \ldots, t'_n))$ which in turn implies that there is a transition $\langle f, q'_1 \cdots q'_n, q' \rangle \in \delta_T$ with $q'_i = q_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $q'_j = q$ as well, and we can conclude that $e'[\![e'']\!] = e$ is in $\mathcal{C}_q$.

"⇐": Let $q(e) = 0$. Choose states $q_1, \ldots, q_n \in Q_T$ with $q_i \in \delta_T^*(s_i)$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $q_j = q$, and let $s'_1, \ldots, s'_n \in \text{RED}$ such that $row(s'_i) = q_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $s'_j = s$. We have $f(s'_1, \ldots, s'_n) \in S$ by condition (2) and $s'_i \sqsubseteq s_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ due to Lemma 4. Moreover, we have $row(f(s'_1, \ldots, s'_n)) \sqsubseteq row(e''[\![s]\!])$ due to the R-consistency of T.
If there is no row $r \in \mathcal{P}_{\text{RED}}$ with $r \sqsubseteq row(f(s'_1, \ldots, s'_n))$ then clearly there is no transition $\langle f, q_1 \cdots q_n, q' \rangle \in \delta_T$ for any $q' \in Q_T$. If there is $r \in \mathcal{P}_{\text{RED}}$ with $r \sqsubseteq row(f(s'_1, \ldots, s'_n))$ then there exists a transition $\langle f, q_1 \cdots q_n, r \rangle \in \delta_T$. Obviously, $row(s)(e) = 0$ implies $row(e''[\![s]\!])(e') = 0$, which in turn entails $row(f(s'_1, \ldots, s'_n))(e') = 0$ and $r(e') = 0$. We obtain $e' \notin \mathcal{C}_r$ by the induction assumption. Since $q_1, \ldots, q_n$ were chosen arbitrarily except for $q_j = q$ we can deduce $q'(e') = 0$ and $e' \notin \mathcal{C}_{q'}$ for all states q' that are reachable from q via the symbol f, and consequently e cannot be an element of $\mathcal{C}_q$. ■

Lemma 6 *For all $q, q' \in Q_T$ we have $q \sqsubseteq q' \Leftrightarrow \mathcal{C}_q \subseteq \mathcal{C}_{q'}$.*

Proof: Let $q, q \in Q_T$ and $s, s' \in S$ with $row(s) \sqsubseteq q$ and $row(s') \sqsubseteq q'$.

"⇒": Let $q \sqsubseteq q'$. Choose $e \in \mathcal{C}_q$. For $e = \square$ we have $q(\square) = 1$ which implies $q'(\square) = 1$ due to $q \sqsubseteq q'$ and thus $q' \in F_T$ and $\square \in \mathcal{C}_{q'}$.

Now let $e = e'[\![e'']\!]$ with $e', e'' \in \mathbb{C}_\Sigma$ and $e'' = f(s_1, \ldots, s_n)$ and $s_j = \square$ for some $j \in \{1, \ldots, n\}$. Note that $e''[\![s]\!], e''[\![s']\!] \in S$ by condition (2).
We have $e'[\![e'']\!] \in \mathcal{C}_q$ and thus there is a transition $\langle f, q_1 \cdots q_n, q'' \rangle \in \delta_T$ with $q_1, \ldots, q_n, q'' \in Q_T$ such that $q_i \in \delta_T^*(s_i)$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $q_j = q$ and $e' \in \mathcal{C}_{q''}$. Let $s'_1, \ldots, s'_n \in \text{RED}$ with $row(s'_i) = q_i$ for $i \in \{1, \ldots, n\}$. We have $f(s'_1, \ldots, s'_n) \in S$ by condition (2), $s'_i \sqsubseteq s_i$ for all $i \in \{1, \ldots, n\} \setminus \{j\}$ and $s'_j \sqsubseteq s$ due to Lemma 4, and $q'' \sqsubseteq row(f(s'_1, \ldots, s'_n))$ due to the definition of δ_T. By condition (2) there is $f(t'_1, \ldots, t'_n) \in S$ with $t'_i = s'_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $t'_j = s'$. Moreover, we have $row(f(s'_1, \ldots, s'_n)) \sqsubseteq row(f(t'_1, \ldots, t'_n))$ due to the fact that T is R-consistent and $q \sqsubseteq q'$. This entails $q'' \sqsubseteq row(f(t'_1, \ldots, t'_n))$ which in turn implies the existence of a transition $\langle f, q'_1 \cdots q'_n, q'' \rangle \in \delta_T$ with $q'_i = q_i$ for $i \in \{1, \ldots, n\} \setminus \{j\}$ and $q'_j = q'$, and we can conclude that $e'[\![e'']\!] = e$ is in $\mathcal{C}_{q'}$ as well.

The inclusion $\mathcal{C}_q \subseteq \mathcal{C}_{q'}$ follows from the fact that e was chosen arbitrarily.

"$\Leftarrow$": Let $\neg(q \sqsubseteq q')$. By the definition of $\sqsubseteq$ there must be a context $e \in E$ with $q(e) = 1$ but $q'(e) = 0$. We obtain $e \in \mathcal{C}_q$ and $e \notin \mathcal{C}_{q'}$ by Lemma 5, which immediately proves $\mathcal{C}_q \nsubseteq \mathcal{C}_{q'}$. ■

Lemma 7 *For all $s \in S$ with $row(s) \in Q_T$ we have $row(s) \in \delta_T^*(s)$.*

Proof. Suppose $row(s) \notin \delta_T^*(s)$. We get a contradiction as follows.
We have $q \sqsubseteq row(s)$ and $\mathcal{C}_q \subseteq \mathcal{C}_{row(s)}$ for all $q \in \delta_T^*(s)$ by Lemmata 4 and 6. As $row(s)$ is prime there is $e \in E$ such that $row(s)(e) = 1$ but $row(s')(e) = 0$ for all $s' \in \mathcal{P}_{\text{RED}}$ with $s' \sqsubseteq s$ and $s' <> s$. As we have assumed that $row(s)$ is not in $\delta_T^*(s)$ itself this implies that e cannot be in $\mathcal{C}_{q'}$ for any $q' \in \delta_T^*(s)$ by Lemma 5, and hence $\mathcal{R}_T$ would not accept $e[\![s]\!]$. However, this contradicts condition (3) requiring $\mathcal{R}_T$ to be T-consistent, and the claim is shown. ■

The **Proof of Theorem 3** can now be concluded as follows.

We abbreviate $\mathcal{L}(\mathcal{R}_T)$ to L_R. First we show that $\mathcal{R}_T$ is an RFTA by proving the stronger claim $\mathcal{C}_{row(s)} = s^{-1}L_R$ for all $s \in S$ with $row(s) \in Q_T$:

"$\subseteq$": We have $row(s) \in \delta_T^*(s)$ by Lemma 7, which implies $\mathcal{C}_{row(s)} \subseteq s^{-1}L_R$.

"$\supseteq$": We have $q \sqsubseteq row(s)$ and $\mathcal{C}_q \subseteq \mathcal{C}_{row(s)}$ for all $q \in \delta_T^*(s)$ by Lemmata 4 and 6, which yields $s^{-1}L \subseteq \mathcal{C}_{row(s)}$, and thus $\mathcal{C}_{row(s)} = s^{-1}L_R$.

It remains to show that $s^{-1}L_R$ is prime to ensure the state-minimality of $\mathcal{R}_T$: We have $\mathcal{C}_{row(s')} \subseteq \mathcal{C}_{row(s)}$ for all $s' \in \mathcal{P}_{\text{RED}}$ with $s' \sqsubseteq s$ by Lemma 6 and $s^{-1}L_R \supsetneq \bigcup\{s'^{-1}L_R \subseteq \mathbb{C}_\Sigma \mid s' \in \mathcal{P}_{\text{RED}} \setminus \{s\} : s' \sqsubseteq s\}$ due to the fact that there is a context $e \in E$ such that $row(s)(e) = 1$ but $row(s')(e) = 0$ for all $s' \in \mathcal{P}_{\text{RED}}$ with $s' \sqsubseteq s$ and $s' <> s$.

Finally, $\mathcal{R}_T$ is saturated by condition (2) and hence isomorphic to $\mathcal{R}_{L_R}$. ■

A.2 Four different ways of using a counterexample

There are various options of how to "milk" a counterexample. We compare four of them: Let $T = \langle S, E, obs\rangle$ with $S = \text{RED} \cup \text{BLUE}$, and let c be some counterexample for $\mathcal{A}_T$. Each of the following methods allows the creation of at least one more distinct RED row in T in at most one more step:

a. Join $Subt(c)$ to RED.

b1. Join $Cont(c)$ to E.

b2. Find $s \in \text{BLUE}$ and $e \in \mathbb{C}_\Sigma$ with $c = e[\![s]\!]$ and join
$Cont(c) \setminus \{e[\![s']\!] \in \mathbb{C}_\Sigma \mid s' \in Cont(s)\}$ to E.

c. (Procedure MINIMIZE) If there is $s \in$ BLUE and $e \in \mathbb{C}_\Sigma$ with $c = e[\![s]\!]$ but no $r \in$ RED with $\neg(r <> s)$ such that $e[\![r]\!]$ is a counterexample for $\mathcal{A}_T$ as well then add s to RED. Otherwise find $s' \in$ BLUE, $e' \in \mathbb{C}_\Sigma$, and $r' \in$ RED with $c = e'[\![s']\!]$ and $\neg(r' <> s')$ such that $e'[\![r']\!]$ is also a counterexample for $\mathcal{A}_T$ and repeat the procedure with input $e'[\![r']\!]$.

(a): Either such a row is created directly if E already contains a suitable separating context, or the table must become inconsistent. To see the latter, assume that no element of $Subt(c)$ is obviously different from all RED ones. As the automaton derived from the new table including the set $Subt(c)$ can evidently assign a different state to the counterexample c than $\mathcal{A}_T$ although no new distinct row representing a separate state has been created this automaton must be non-deterministic, and the new table inconsistent. This would be repaired by the consistency check in one of the following loop executions by adding to E a separating context for two RED elements that have not been obviously different before, thus creating another distinct row.
(c): Suppose MINIMIZE returns $s \in$ BLUE, then consider (a) and the fact that $s \in Subt(c)$. Remark: As T is closed there is $r \in$ RED with $\neg(r <> s)$ but $e[\![s]\!]$ is a counterexample for some $e \in \mathbb{C}_\Sigma$ whereas $e[\![r]\!]$ is not. Consequently s and r should represent distinct states such that the context e leads to an accepting state from one of them but there is no such accepting state for the other. We add s to RED but we could also have added e to E since e obviously separates r and s. This is the option chosen in [27].
(b1)/(b2): Consider (c) and the fact that the context alternatively added to E in (c) is an element of $Cont(c) \setminus \{e[\![s']\!] \in \mathbb{C}_\Sigma \mid s' \in Cont(s)\} \subseteq Cont(c)$ for some $s \in$ BLUE and $e \in \mathbb{C}_\Sigma$ with $c = e[\![s]\!]$. ■

Remark: Method (b1) introduces all separating contexts that can be derived from c into the table at once but this may also lead to redundant columns. The string equivalent of method (a) is the one used in the original description of LSTAR by Angluin [2]. Method (b1) was suggested for strings in a footnote in [30]. Method (b2) is based on the reflection that there is at least one suitable context in $Cont(c)$ that does not have an element of $Cont(s)$ as a subtree and thus we can avoid creating redundant columns by excluding contexts with subtrees in $Cont(s)$ from the set that we join to E. Method (c) has been first described in [17] and is based on the technique of *contradiction backtracing* developed by Shapiro [34].

A.3 An example for an identical representative sample

Let $\mathcal{A} = \langle\{a, b, c\}, \{q_1, q_2, q_3, q_4, q_F\}, \{q_F\}, \delta\rangle$ with $\delta =$

$$\{\langle a, q_2q_3, q_F\rangle, \langle b, q_2, q_1\rangle, \langle b, q_1, q_2\rangle, \langle b, \langle\rangle, q_1\rangle, \langle c, q_4, q_3\rangle, \langle c, q_3, q_4\rangle, \langle c, \langle\rangle, q_3\rangle\}.$$

We have $\mathcal{L}(\mathcal{A}) = \{a(b^{2k+2}, c^{2l+1}) \mid k, l \in \mathbb{N}\}$ where x^m for $x \in \{b, c\}$ denotes a non-branching tree of the form $x(x(\ldots x(x)\ldots))$ with x occurring m times, i.e., each tree recognized by $\mathcal{A}$ must contain an even number of b's greater than 0 and an uneven number of c's.

Let $\mathcal{A}' = \langle\{a, b, c\}, \{q_1, q_2, q_F\}, \{q_F\}, \delta'\rangle$ with

$$\delta' = \{\langle a, q_1q_2, q_F\rangle, \langle b, q_1, q_1\rangle, \langle b, \langle\rangle, q_1\rangle, \langle c, q_2, q_2\rangle, \langle c, \langle\rangle, q_2\rangle\}.$$

We have $\mathcal{L}(\mathcal{A}') = \{a(b^k, c^l) \mid k, l \in \mathbb{N}\}$.

The set $\{a(b(b), c(c(c)))\}$ is representative for both $\mathcal{L}(\mathcal{A})$ and $\mathcal{L}(\mathcal{A}')$.

A.4 A simple example run

Let the target language be $L = \{a(b^{2k+2}, c^{2l+1}) \mid k, l \in \mathbb{N}\}$. The canonical RFTA for L is isomorphic to the canonical DFTA, that is, the FTA $\mathcal{A}$ in Appendix A.3 above. Consider the representative sample $\{a(b(b), c(c(c)))\}$. Both ALTEX and RESI build from it the table

	$\Box$	e_1	e_2	e_3	e_4	e_5
b	0	0	0	1	0	0
c	0	0	0	1	0	1
$b(b)$	0	1	0	0	0	0
$c(c)$	0	0	0	0	1	0
$c(c(c))$	0	0	0	1	0	1
$a(b(b), c(c(c)))$	1	0	0	0	0	0

where we have $e_1 = a(\Box, c(c(c)))$, $e_2 = a(b(\Box), c(c(c)))$, $e_3 = a(b(b), \Box)$, $e_4 = a(b(b), c(\Box))$, and $e_5 = a(b(b), c(c(\Box)))$.
This table has five distinct rows which correspond to the five states of the target automaton. Moreover, the table is both consistent and R-consistent, so both ALTEX and RESI immediately return the corresponding automaton as their final and correct solution.

The learners for MQs and EQs would both start out with a table

	$\Box$
b	0
c	0
$b(b)$	0
$c(b)$	0
$a(b, b)$	0

which is trivially closed, consistent, R-closed, and R-consistent, and represents the all-rejecting automaton. Upon the receipt of the positive counterexample $a(b(b), c(c(c)))$ both learners could add all subtrees of the example to RED and all contexts that can be derived from it to E and would obtain

	$\Box$	e_1	e_2	e_3	e_4	e_5
b	0	0	0	1	0	0
c	0	0	0	1	0	1
$b(b)$	0	1	0	0	0	0
$c(c)$	0	0	0	0	1	0
$c(c(c))$	0	0	0	1	0	1
$a(b(b), c(c(c)))$	1	0	0	0	0	0
$b(c)$	0	0	0	0	0	0
$a(b(b), c)$	1	0	0	0	0	0

where in the BLUE part we show only one representative $b(c)$ of the failure state and the only other row not representing a failure state.
This table has six distinct rows of which all are prime, the table is consistent and R-consistent and represents the total version of the target automaton $\mathcal{A}$ which is also saturated. Hence, the next EQ will be answered in the positive and both learners terminate successfully.

Bigger examples are rather space-consuming – we refer the reader to [5] and [18] where they may run our learners RESI and MATRES on the regular tree languages given in Section 5 and on page 175, respectively.