Learning Tree Languages from Positive Examples and Membership Queries

Jérôme Besombes[1] and Jean-Yves Marion[1,2]

[1] LORIA
615, rue du jardin botanique
54602 Villers-lès-Nancy, France
[2] INPL-Ecole des Mines de Nancy
Parc de Saurupt,
54042 Nancy CEDEX, France
{Jerome.Besombes, Jean-Yves.Marion}@loria.fr

Abstract. We investigate regular tree languages exact learning from positive examples and membership queries. Input data are trees of the language to infer. The learner computes from the inputs new trees and ask to the oracle whether they belongs or not to the language. From the answers, the learner may ask further membership until he finds the correct grammar that generates the target language.
Neither negative examples, equivalence queries nor counter examples are allowed in this paradigm. This paradigm was introduced by Angluin in [1] for the case of regular word language. Recall that it is not infer the regular tree (or even word) languages just from positive examples.
We describe an efficient algorithm which is polynomial in the size of the examples for learning the whole class of regular tree languages. The convergence is insured when the set of examples contains a representative sample of the language to guess. Any finite subset $\mathcal{E}$ of a regular tree language $\mathcal{L}$ is representative for $\mathcal{L}$ if every transition of the minimal tree automaton for $\mathcal{L}$ is used at least once for the derivation of an element of $\mathcal{E}$. A representative sample is really minimal for this setting.
Our main motivation is to seek for a mathematical model of natural language acquisition based on the recent advances on this field. Here, the membership queries are seen as a minimal interaction between a child and his environment.

1 Introduction

1.1 Some linguistic motivations

The most astonishing discovery of Chomsky is the universal grammar which is a model of how the human language works. The universal grammar is an innate combinatorial systems from which every language, french, english, Japanese, can be derived. What is the implication of Chomsky's universal grammar on grammatical inference? We think it gives a strong intuition on the mathematical modelling of language learning. Before going further, let us focus on the linguistic aspect of language acquisition process.

Several recent works of psycholinguists like Pinker [16] or Christophe [6] advocate that the universal grammar plays the role of a learning device for children. A child is able to determine whether or not a sentence is grammatically correct, even if we don't know the meaning of each word. Of course, semantics speed up the learning process, but it is not necessary. And, it is fascinating to see that a child needs only few informations in order to learn a language (poverty-of-stimulus hypothesis).

Moreover, the underlying reason seems to be our capacity to guess a tree structure of a phrase. How a child is able to do that is beyond the scope of this paper. However the child language acquisition process is not based on the construction of a huge finite automaton with probabilistic transitions, because there is an infinity of valid sentences and we have the ability to generate them.

To sum up this brief discussion, we take as hypothesis that a child computes a grammar from tree structured sentences.

1.2 A mathematical model

Can we give a mathematical model of the language acquisition which corroborates this theory? The grammatical inference paradigm of Gold [13] is a good candidate. Indeed, the inputs of the learning process are just examples of the target language. And there is no interaction with the environment. But this paradigm is too weak to be plausible. Indeed the class of regular languages is not learnable just from positive examples. For this reason, we add to Gold paradigm an oracle which answers to membership queries. Hence, the grammatical inference is based on positive examples and membership questions of computed elements, as introduced by Angluin in [1]. This learning model agrees with the poverty-of-stimulus hypothesis. Indeed the interaction with the environment is very weak. For example, a child asks something, but nobody understands. From this lack of reaction from the environment, he may deduce that the sentence is wrong, and so not in the language. We insist on the fact that membership queries are the minimal information which can be inferred in a dialog.

On the other hand, negative examples are not necessary because a parent does not say incorrect sentences to a child. One might think about other kind of queries like equivalence queries as suggested by Angluin [2]. But there are not necessary as we shall see and there are unrealistic in a linguistic context. In conclusion, our learning model seems quite adequate with respect to our initial motivation.

Now, we have set the learning paradigm, we have to say what's kind of language are targeted. As we have said, we can assume that a child has a kind of parser which transforms a (linear) sentence into a tree representation. So, we learn tree languages. Regular tree languages are the bare bone of several linguistic formalisms like classical categorial grammars for which learnabilty has been studied by Kanazawa in [15] or dependency languages [8].

1.3 The results

We establish that the whole set of regular tree languages is efficiently identifiable from membership queries and positive examples. The running time of the learning algorithm is polynomial

in the size of the input examples. The efficiency is a necessary property of our model. The difficulty is to construct trees to ask mambership queries and which gives useful information to proceed in the inference process. As far as we know, this result is new.

1.4 A web application

There are other applications of our result. For example, an XML document is a tree, if we forget links. The style-sheet defines a regular tree grammar. Now, say that we try to determine a style-sheet. For this, we can read correct XML documents from a server which forms a set of positive examples. Then, we can build a XML document and make a membership query by sending it to the server. If no error occurs then the the document is in the language, otherwise it is not.

1.5 Related works

Learning from positive examples and membership queries. Angluin considers the same learning paradigm in [1] for the class of regular word languages. The notion of observation table which is defined in [2] is already unformally used in [1]. However, we can not extend in a straight forward way the algorithm of [1] as it is explained by Sakakibara in [18]. In case of words, we can try all possible transitions from a word ω by checking $\omega\alpha$ for every letter α. Indeed, in the case of trees, this approach will lead us to consider an exponential number of cases, depending on the arity of the function symbols. As a consequence, if Angluin's algorithm may be adapted for learning the class of regular tree language, this adaptation loses the property of polynomiality. In order to preserve polynomiality, we introduce the original idea of considering contexts obtained from trees given in input, instead of testing new contexts (suffixs for words are contexts for trees) constructed from the alphabet (like in Angluin's algorithm).

Other paradigms. Sakakibara studied grammatical inference of languages of unlabelled derivation trees of context free grammars. In [19], he extends the result of [2] by learning with membership queries and equivalence queries. The possibility of asking the teacher whether a calculated hypothesis corresponds to the target language seems not to be relevant for the aim of construct a model of natural language process. In [18, 17] Sakakibara uses positive and negative examples with membership query. At the end of the paper [17], it claims that the second algorithm is polynomial time (without proofs), but the main one is exponential in the function symbol arity. Compare with [17], we have showed that negative examples are not necessary and that the running time of our learning algorithm is polynomial. It is worth noticing that the set of all unlabelled context free derivation tree language is a strict subclass of the set of regular tree languages. Therefore, we learn more and in a weaker setting since negative examples are not necessary in our work. This is important when we are learning from structured examples. Indeed, the language of structured examples may not come from a context free grammar but still the word language is context free.

Inference of regular tree languages from positive examples only, has been studied in [12, 14], in [4] and [11] learnable subclasses have been defined and in [5], learning is studied from a stochastic point of view.

Inference of regular tree languages with queries has been studied in [9]; the learning algorithm is based on membership queries and equivalence queries and this result constitutes an extension of Sakakibara's works. In [10], a polynomial version of the former learning algorithm has been developped.

2 Regular tree languages

A ranked alphabet $\mathcal{V}$ is a finite set of symbols with a function *arity* from $\mathcal{V}$ to $\mathbb{N}$, which indicates the arity of a symbol. The set $\mathcal{T}(\mathcal{V})$ of terms is inductively defined as follows. A symbol of arity 0 is in $\mathcal{T}(\mathcal{V})$, and if $\mathtt{f}$ is a symbol of arity n and $\mathtt{t}_1, \ldots, \mathtt{t}_n$ are in $\mathcal{T}(\mathcal{V})$, then $\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$ is in $\mathcal{T}(\mathcal{V})$. Subterms of a term $\mathtt{t}$ are definied by : $\mathtt{t}$ is a subterm of $\mathtt{t}$ and if $\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$ is a subterm of $\mathtt{t}$, then $\mathtt{t}_1, \ldots, \mathtt{t}_n$ are subterms of $\mathtt{t}$. Throughout, labelled ordered trees are represented by terms. For a set of terms $\mathcal{E}$, $\mathcal{S}(\mathcal{E})$ represents the set of subterms of elements of $\mathcal{E}$.

A *context* is a term $\mathtt{c}[\diamond]$ containing a special variable $\diamond$ which has only one occurrence. The variable $\diamond$ marks an empty place in a term. In particular, $\diamond$ is a context called the *empty context*. The substitution of $\diamond$ by a term $\mathtt{s}$ is noted $\mathtt{c}[\mathtt{s}]$. In particular, for any set $\mathcal{E}$, $\mathcal{E}[\diamond]$ contains the empty context $\diamond$. $\mathcal{E}[\diamond]$ is the set of contexts obtained by replacing in any element of $\mathcal{E}$, exactly one subterm by $\diamond$.

A *bottom up non-deterministic tree automaton* (NFTA) is a quadruplet $\mathcal{A} = \langle \mathcal{V}, \mathcal{Q}, \mathcal{Q}_F, \underset{\mathcal{A}}{\rightarrow} \rangle$ where $\mathcal{V}$ is ranked alphabet, $\mathcal{Q}$ is a finite set of states, $\mathcal{Q}_F \subseteq \mathcal{Q}$ is the set of final states, and $\underset{\mathcal{A}}{\rightarrow}$ is the set of transitions. A transition is a rewrite rule of the form $\mathtt{f}(q_1, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} q$ where q and $q_1, \ldots, q_n$ are states of $\mathcal{Q}$, and $\mathtt{f}$ is a symbol of arity n. In particular, a transition may be just of the form $\mathtt{a} \underset{\mathcal{A}}{\rightarrow} q$ where $\mathtt{a}$ is a symbol of arity 0.

The single derivation relation $\underset{\mathcal{A}}{\rightarrow}$ is defined so that $\mathtt{t} \underset{\mathcal{A}}{\rightarrow} \mathtt{s}$ if and only if there is a transition $\mathtt{f}(q_1, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} q$ such that for a context $\mathtt{u}[\diamond]$, $\mathtt{t} = \mathtt{u}[\mathtt{f}(q_1, \ldots, q_n)]$ and $\mathtt{s} = \mathtt{u}[q]$.

The derivation relation $\overset{*}{\underset{\mathcal{A}}{\rightarrow}}$ is the reflexive and transitive closure of $\underset{\mathcal{A}}{\rightarrow}$. The tree language *recognized* by $\mathcal{A}$ is $\mathcal{L}_{\mathcal{A}} = \{\mathtt{t} \in \mathcal{T}(\mathcal{V}) : \mathtt{t} \overset{*}{\underset{\mathcal{A}}{\rightarrow}} q_F \text{ and } q_F \in \mathcal{Q}_F\}$.

A state q is *accessible* if there is a term $\mathtt{t}$ such that $\mathtt{t} \overset{*}{\underset{\mathcal{A}}{\rightarrow}} q$. A NFTA is *reduced* if each state is accessible. In the rest of this paper, we will consider only reduced NFTA , since any NFTA can be transformed into an equivalent reduced NFTA .

A set of terms $\mathcal{L}$ is a *regular tree language* if and only if $\mathcal{L} = \mathcal{L}_{\mathcal{A}}$, for a DFTA $\mathcal{A}$.

A finite tree automaton $\mathcal{A} = \langle \mathcal{V}, \mathcal{Q}, \mathcal{Q}_F, \underset{\mathcal{A}}{\rightarrow} \rangle$ is *deterministic* (DFTA) if there are no two different rules with the same left hand side. In this case, $\underset{\mathcal{A}}{\rightarrow}$ induces a transition function $\delta_{\mathcal{A}} : \mathcal{T}(\mathcal{V}) \rightarrow \mathcal{Q}$ where $\delta_{\mathcal{A}}(\mathtt{t}) = q$ iff $\mathtt{t} \overset{*}{\underset{\mathcal{A}}{\rightarrow}} q$. It has been showed that NFTA and DFTA are equivalent notions (see [7] for a transformation algorithm of any NFTA in a equivalent DFTA and a proof of this equivalence). Indeed, a set of terms $\mathcal{L}$ is a regular tree language if $\mathcal{L} = \mathcal{L}_{\mathcal{A}}$, for a DFTA $\mathcal{A}$.

In [7] is proved that for any regular tree language there exists a unique minimal automata in the number of states, that constitutes a tree language version of the Myhill-Nerode theorem.

Example 1. Consider the DFTA $\mathcal{A} = \langle \{a, b, c\}, \{q_F, q_1, q_2, q_3, q_4\}, \{q_F\}, \underset{\mathcal{A}}{\rightarrow} \rangle$, where $\underset{\mathcal{A}}{\rightarrow}$ is the following set of transitions:

$$
\begin{array}{cc}
a(q_2, q_3) \underset{\mathcal{A}}{\rightarrow} q_F & \\
b(q_2) \underset{\mathcal{A}}{\rightarrow} q_1 & c(q_4) \underset{\mathcal{A}}{\rightarrow} q_3 \\
b(q_1) \underset{\mathcal{A}}{\rightarrow} q_2 & c(q_3) \underset{\mathcal{A}}{\rightarrow} q_4 \\
b \underset{\mathcal{A}}{\rightarrow} q_1 & c \underset{\mathcal{A}}{\rightarrow} q_3
\end{array}
$$

The tree $a(b(b), c(c(c)))$ belongs to $\mathcal{L}_{\mathcal{T}}$. Indeed we have:

$$a(b(b), c(c(c))) \underset{\mathcal{A}}{\rightarrow} a(b(q_1), c(c(c))) \underset{\mathcal{A}}{\rightarrow} a(q_2, c(c(c)))$$

$$\underset{\mathcal{A}}{\rightarrow} a(q_2, c(c(q_3))) \underset{\mathcal{A}}{\rightarrow} a(q_2, c(q_4)) \underset{\mathcal{A}}{\rightarrow} a(q_2, q_3) \underset{\mathcal{A}}{\rightarrow} q_F$$

$\mathcal{L}_\mathcal{A}$ is the tree language $\{a(b^{2n+2}, c^{2m+1}) \ : \ n, m \in \mathbb{N}\}$.

Any function from $\mathcal{Q}$ to a set $\mathcal{Q}'$ can be extended to an *automata homomorphism* with $\phi(\mathcal{A}) = \langle \mathcal{V}, \phi(\mathcal{Q}), \phi(\mathcal{Q}_F), \phi(\underset{\mathcal{A}}{\rightarrow}) \rangle$ and $\phi(\underset{\mathcal{A}}{\rightarrow})$ is the set of rules of the form $\mathtt{f}(\phi(q_1), \ldots, \phi(q_n)) \underset{\mathcal{A}'}{\rightarrow} \phi(q)$, for all rule $\mathtt{f}(q_1, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} q$ in $\underset{\mathcal{A}}{\rightarrow}$. For any term $\mathtt{t}$, if $\mathtt{t} \overset{*}{\underset{\mathcal{A}}{\rightarrow}} q$ so $\mathtt{t} \overset{*}{\underset{\mathcal{A}'}{\rightarrow}} \phi(q)$, which implies that $\mathcal{L}_\mathcal{A} \subseteq \mathcal{L}_{\mathcal{A}'}$. If ϕ is bijective, it consists in a renaming of the states and $\mathcal{L}_\mathcal{A} = \mathcal{L}_{\mathcal{A}'}$.

3 Learning regular tree languages

3.1 The learning paradigm

The goal is the identification of any unknown regular tree language $\mathcal{L}$ with help of a teacher. The teacher is an oracle and the learning process begins with the construction of a representative sample who provides the learner with positive examples (a finite subset $\mathcal{E}$ of $\mathcal{L}$). The teacher transmits $\mathcal{E}$ to the learner Communication consisting in a sequence of membership queries is possible. The learner submit a term $\mathtt{t}$ to the teacher whose reply `1` if $\mathtt{t}$ belongs to $\mathcal{L}$ and `0` otherwise. From these data, the learner constructs a `DFTA` $\mathcal{A}$ and the process terminates as soon as he convinces himself that this `DFTA` is correct ($\mathcal{L}_\mathcal{A} = \mathcal{L}$). We say that an algorithm identifies the class of regular tree languages if and only if, for any language, from any representative set given by a teacher, the `DFTA` constructed when the process stop is correct.

3.2 Observation table

Following the idea of Angluin [3], information obtained from the queries is stored in a table. Let $\mathcal{L}$ be a tree language, $\mathcal{E}$ be a finite set of terms and F be a finite set of contexts. The *observation table* $T = T_\mathcal{L}(\mathcal{E}, \mathsf{F})$ is the table defined by:

- rows are labelled with terms of $\mathcal{S}(\mathcal{E})$,
- columns are labelled with contexts of F,
- cells $T_\mathcal{L}(\mathsf{t}, \mathsf{c}[\diamond])$ are labelled with `1` or `0` in such a way that

$$T_\mathcal{L}(\mathsf{t}, \mathsf{c}[\diamond]) = \begin{cases} \mathtt{1} \text{ if } \mathsf{c}[\mathsf{t}] \in \mathcal{L} \\ \mathtt{0} \text{ otherwise} \end{cases}$$

For any term $\mathtt{t} \in \mathcal{E}$ and any observation table T, we denote $row_T(t)$ the binary word corresponding to the reading from left to right of the row labelled by $\mathtt{t}$ in T.

Example 2. Consider the tree Language $\mathcal{L}_T$ defined in Example 1, $\mathcal{E}$ the set of terms:

$$\{a(b(b), c(c(c)))\}$$

and F the set of contexts:

$$\{\diamond, a(\diamond, c(c(c))), a(b(\diamond), c(c(c))), a(b(b), \diamond), a(b(b), c(\diamond)), a(b(b), c(c(\diamond)))\}$$

The corresponding observation table $T = T_\mathcal{L}(\mathcal{E}, \mathsf{F})$ is the following:

	$\diamond$	$a(\diamond, c(c(c)))$	$a(b(\diamond), c(c(c)))$	$a(b(b), \diamond)$	$a(b(b), c(\diamond))$	$a(b(b), c(c(\diamond)))$
$a(b(b), c(c(c)))$	1	0	0	0	0	0
$b(b)$	0	1	0	0	0	0
b	0	0	1	0	0	0
$c(c(c))$	0	0	0	1	0	1
$c(c)$	0	0	0	0	1	0
c	0	0	0	1	0	1

An observation table $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$ defines a NFTA

$$\mathcal{A}_T = \langle \mathcal{V}_T, \mathcal{Q}_T, \mathcal{Q}_{F,T}, \underset{T}{\rightarrow} \rangle$$

where:

- $\mathcal{V}_T$ is the set of symbols occuring in $\mathcal{E}$,
- $\mathcal{Q}_T = \{row_T(\mathtt{t}), \mathtt{t} \in \mathcal{S}(\mathcal{E})\}$,
- $\mathcal{Q}_{F,T} = \{row_T(\mathtt{t}), \mathtt{t} \in \mathcal{L}\}$,
- the set $\underset{T}{\rightarrow}$ is the set of of transitions of the form:

$$\mathtt{f}(row_T(\mathtt{t}_1), \ldots, row_T(\mathtt{t}_n)) \rightarrow row_T(\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)),$$

for all $\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n) \in \mathcal{S}(\mathcal{E})$.

Example 3. The table T of example 2 defines the NFTA $\mathcal{A}_T$ as:

- $\mathcal{V}_T = \{a, b, c\}$
- $\mathcal{Q}_T = \{\mathtt{100000}, \mathtt{010000}, \mathtt{001000}, \mathtt{000101}, \mathtt{000010}\}$
- $\mathcal{Q}_{F,T} = \{\mathtt{100000}\}$
- $\underset{T}{\rightarrow}$ is the following set of transitions

$$a(\mathtt{010000}, \mathtt{000101}) \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{100000}$$
$$b(\mathtt{010000}) \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{001000} \qquad c(\mathtt{000010}) \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{000101}$$
$$b(\mathtt{001000}) \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{010000} \qquad c(\mathtt{000101}) \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{000010}$$
$$b \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{001000} \qquad c \underset{\mathcal{A}_T}{\longrightarrow} \mathtt{000101}$$

We remark that $\mathcal{A}_T = \phi(\mathcal{A})$, where $\mathcal{A}$ is the `DFTA` introduced in example 1 and ϕ is the automata homomorphism defined by:

$$\phi(q_F) = \mathtt{1\,0\,0\,0\,0\,0}$$
$$\phi(q_1) = \mathtt{0\,0\,1\,0\,0\,0}$$
$$\phi(q_2) = \mathtt{0\,1\,0\,0\,0\,0}$$
$$\phi(q_3) = \mathtt{0\,0\,0\,1\,0\,1}$$
$$\phi(q_4) = \mathtt{0\,0\,0\,0\,1\,0}$$

Remark 1. In general case, the `DFTA` $\mathcal{A}_T$ is not complete and has no particular reason to be deterministic.

An observation table $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$ is said to be *consistent* if for any terms $\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$ and $\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n)$ in $\mathcal{S}(\mathcal{E})$, if $\forall 1 \leq j \leq n$ we have :

$$row_T(\mathtt{t}_j) = row_T(\mathtt{t}'_j)$$

then

$$row_T(\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)) = row_T(\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n)).$$

Lemma 1. *The NFTA $\mathcal{A}_T$ is deterministic if and only if the table T is consistent.*

Proof. This equivalence is straightforward from definitions of consistency of T and of the NFTA $\mathcal{A}_T$.

3.3 Representative sample

Let $\mathcal{L}$ be a regular tree language and $\mathcal{E}$ a finite subset of $\mathcal{L}$. $\mathcal{E}$ is a *representative sample* of $\mathcal{L}$ if the minimal DFTA $\mathcal{A}$ of $\mathcal{L}$ and any rule $\mathtt{f}(q_1, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} q$ of $\mathcal{A}$, there is a term $\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$ in $\mathcal{S}(\mathcal{E})$ such that $\forall 1 \leq i \leq n$, $\delta_{\mathcal{A}}(\mathtt{t}_i) = q_i$. Informally, a representative sample of a language $\mathcal{L}$ is a finite subset $\mathcal{E}$ such that all the rules of $\mathcal{A}$ are used to produce it. Any finite set of terms which contains a representative sample is a representative sample too.

Example 4. This example illustrates the fact that a set can be a representative set for several distinct languages. Let $\mathcal{A}$ be the DFTA defined in example 1 and $\mathcal{A}'$ the DFTA defined by:

$$\begin{array}{rcl} & a(q_1, q_2) \underset{\mathcal{A}'}{\rightarrow} q_F & \\ b(q_1) \underset{\mathcal{A}'}{\rightarrow} q_1 & & c(q_2) \underset{\mathcal{A}'}{\rightarrow} q_2 \\ b \underset{\mathcal{A}'}{\rightarrow} q_1 & & c \underset{\mathcal{A}'}{\rightarrow} q_2 \end{array}$$

where q_F is the unique final state of $\mathcal{A}'$. From this definition, we have

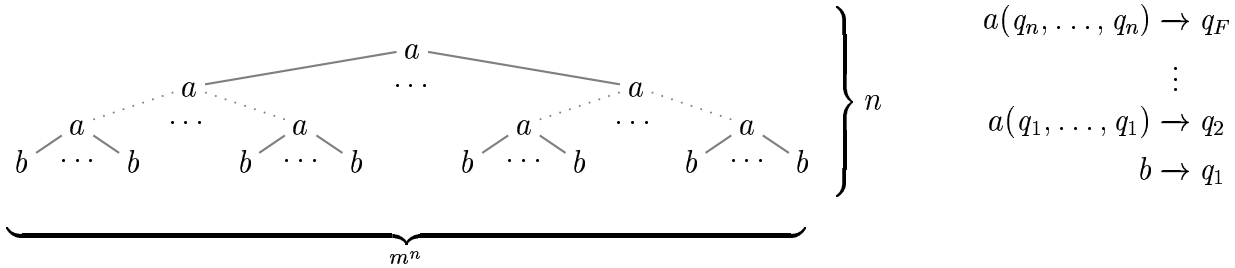
$$\mathcal{L}_{\mathcal{A}'} = \{a(b^n, c^m) : \ n, m \in \mathbb{N}^*\}$$

and the singleton

$$\{a(b(b), c(c(c)))\}$$

is a representative sample for both $\mathcal{L}_{\mathcal{A}}$ and $\mathcal{L}_{\mathcal{A}'}$.

Remark 2. The question of the size of a minimal representative sample relatively to the size of the minimal automaton $\mathcal{A}$ of a language $\mathcal{L}$ is interesting to be discussed. In the contrary of the case of word language, there is no polynomial relation between the size (the total number of nodes) of a minimal characteristic sample and the size (the number of states) of the minimal DFTA. For instance, for two integers n and m, the singleton containing the following tree:



where the arity of a is m and the arity of b is 0, is a regular tree language recognized by the above minimal DFTA. The size of the representative sample (which is unique and corresponds to the language itself) is $m^{n+1} - 1$. This shows that the "compression" corresponding to the representation of a tree language with an automaton is very efficent in the case of tree. Since, for a learning problem, we deal with elements of the language, its more natural to consider the complexity relatively to the size of the language. In our sense, the size of a representative sample can be seen as a good definition of the size of a language.

Lemma 2. *If $\mathcal{E}$ is a representative sample for a regular tree language $\mathcal{L}$, F any finite set of contexts and T the observation table $T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$, then we have $\mathcal{L} \subseteq \mathcal{L}_{\mathcal{A}_T}$.*

Proof. We exhibit an automata homomorphism ϕ such that $\mathcal{A}_T = \phi(\mathcal{A})$, where $\mathcal{A}$ is the minimal automaton for $\mathcal{L}$. ϕ is defined by the function from the set $\mathcal{Q}$ of states of $\mathcal{A}$ to the set $\mathcal{Q}_T$ of states of $\mathcal{A}_T$ by:

$$\begin{aligned}\phi : \mathcal{Q} &\mapsto \mathcal{Q}_T \\ q &\to row_T(\mathsf{t}) \ : \ t \in \mathcal{S}(\mathcal{E}) \text{ and } \delta_{\mathcal{A}}(\mathsf{t}) = q\end{aligned}$$

If t and t' are two terms in $\mathcal{S}(\mathcal{E})$ such that $\delta_{\mathcal{A}}(\mathsf{t}) = \delta_{\mathcal{A}}(\mathsf{t}')$, we have $row_T(\mathsf{t}) = row_T(\mathsf{t}')$. The function ϕ is so well defined. By the definition of a representative sample, we can easily check that $\mathcal{Q}_T = \phi(\mathcal{Q})$ and that for any transition $\mathtt{f}(q_1, \ldots, q_n) \underset{\mathcal{A}}{\to} q$, we have $\mathtt{f}(\phi(q_1), \ldots \phi(q_n)) \underset{T}{\to} \phi(q)$.

Remark 3. The number of states of the automaton $\mathcal{A}_T$ is lower or equal to the number of states of $\mathcal{A}$.

Remark 4. If the autamata homomorphism ϕ defined in the proof of Lemma 2 is bijective, then $\mathcal{L} = \mathcal{L}_{\mathcal{A}_T}$. Consequently, if $\mathcal{L} \neq \mathcal{L}_{\mathcal{A}_T}$, then there are two terms t and t' in $\mathcal{S}(\mathcal{E})$ such that $row_T(\mathsf{t}) = row_T(\mathsf{t}')$ and $\delta_{\mathcal{A}}(\mathsf{t}) \neq \delta_{\mathcal{A}}(\mathsf{t}')$ for $\delta_{\mathcal{A}}$ the transition function of the minimal automaton $\mathcal{A}$ for $\mathcal{L}$.

The next lemma is central for the construction of an algorithm that solves the learning problem. First, it will underline that consistency of the table constructed from a representative sample $\mathcal{E}$ implies that the corresponding automaton is a minimal automaton for the target language $\mathcal{L}$, that is $\mathcal{L}$ is learnt. Moreover, since in the case where the table is not consistent, two subterms t and t' of $\mathcal{E}$ for which the rows are equal in the table should be different, then a new context $\mathsf{c}[\diamond]$ can be constructed and added to F for separate these rows ($\mathsf{c}[\mathsf{t}] \in \mathcal{L}$ and $\mathsf{c}[\mathsf{t}] \notin \mathcal{L}$). Such a context is called *separating-context*.

Lemma 3. *Let $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$ be an observation table where $\mathcal{L}$ is a regular tree language, $\mathcal{E}$ a representative set for $\mathcal{L}$, F a set of context containing $\mathcal{E}[\diamond]$ and $\mathcal{A}$ the minimal automaton for $\mathcal{L}$.*

If $\mathcal{L}_{\mathcal{A}_T} \neq \mathcal{L}$, then there are two terms

$$f(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)$$

and

$$f(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)$$

in $\mathcal{S}(\mathcal{E})$ such that:

- $row_T(f(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)) \neq row_T(f(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n))$,
- $\delta_{\mathcal{A}}(\mathsf{t}_j) = \delta_{\mathcal{A}}(\mathsf{t}'_j)$, $\forall 1 \leq j \neq i \leq n$,
- $row_T(\mathsf{t}) = row_T(\mathsf{t}')$.

The proof of Lemma 3 is given in Appendix B.

The first consequence is that the learning problem is solved when the table is consistent.

Lemma 4. *Let $\mathcal{L}$ be a regular tree language, $\mathcal{E}$ a representative sample for $\mathcal{L}$ and F a set of contexts including $\mathcal{E}[\diamond]$. If the table $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$ is consistent, then $\mathcal{L}_{\mathcal{A}_T} = \mathcal{L}$*

Proof. Immediate from Lemma 3, since the terms $\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)$ and $\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)$ in $\mathcal{S}(\mathcal{E})$ are in contradiction with the definition of consistency of the table.

4 Identification of regular tree languages

The algorithm defined in figure 1 works as follows: $ALTEX$ first recieves a finite subset $\mathcal{E}$ of an unknown language $\mathcal{L}$. $ALTEX$ constructs the first observation table $T = T_{\mathcal{L}}(\mathcal{E}, \mathcal{E}[\diamond])$ with help of queries and checks successively the consistency of the table. Each time $ALTEX$ finds the table non-consistent, new contexts constructed from the terms that contradict the consistency are added in the table which is then completed with queries. The process stops when the table is consistent and the automaton $\mathcal{A}_T$ is output.

Input: a finite set of terms $\mathcal{E}$
Initilization: $\mathsf{F} = \mathcal{E}[\diamond]$;
Construct the table $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$;
while exist two terms $\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$ and $\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n)$ in $\mathcal{S}(\mathcal{E})$ such that
$row_T(\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)) \neq row_T(\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n))$ and $\forall 1 \leq i \leq n$, $row_T(\mathtt{t}_i) = row_T(\mathtt{t}'_i)$ **do**
  Find a context $\mathtt{c}[\diamond]$ in F such that $\mathtt{c}[\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)] \in \mathcal{L}$ and $\mathtt{c}[\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n)] \notin \mathcal{L}$;
  $\mathsf{F} = \mathsf{F} \cup \{\mathtt{c}[\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_{i-1}, \diamond, \mathtt{t}_{i+1}, \ldots, \mathtt{t}_n)], 1 \leq i \leq n\}$;
  Construct $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$;
end while;
Return the automata $\mathcal{A}_T$.

Fig. 1. The learning algorithm $ALTEX$

From definition of consistency, $ALTEX$ can easily verify if the table T constructed with help of membership queries is consistent or not. In the case where T is not consistent, the problem is that the algorithm has to find by itself (no counter example is allowed) a new context that will separate two equivalent rows. The key point is that the fact that the input set of terms $\mathcal{E}$ is representative provides the possibility of calculating such a context.

Lemma 5. *If the table T is not consistent, $ALTEX$ calculates a separating-context.*

Proof. $ALTEX$ collects every couples of terms

$$\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$$

and

$$\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n)$$

in $\mathcal{S}(\mathcal{E})$, such that exists a context $\mathtt{c}[\diamond]$ in F with

- $\forall 1 \leq i \leq n$, $row_T(\mathtt{t}_i) = row_T(\mathtt{t}'_i)$,
- $\mathtt{c}[\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_n)] \in \mathcal{L}$
- $\mathtt{c}[\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_n)] \notin \mathcal{L}$.

From Lemma 3, among these couples, there is a particular one

$$\mathtt{f}(\mathtt{t}_1, \ldots, \mathtt{t}_{i-1}, \mathtt{t}, \mathtt{t}_{i+1}, \ldots, \mathtt{t}_n)$$

and

$$\mathtt{f}(\mathtt{t}'_1, \ldots, \mathtt{t}'_{i-1}, \mathtt{t}', \mathtt{t}'_{i+1}, \ldots, \mathtt{t}'_n)$$

such that

$$\forall 1 \leq j \neq i \leq n, \delta_{\mathcal{A}}(\mathtt{t}_j) = \delta_{\mathcal{A}}(\mathtt{t}'_j)$$

where $\mathcal{A}$ is the minimal automaton for $\mathcal{L}$.

$$\mathsf{c}[\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)] \notin \mathcal{L}$$

is so equivalent to

$$\mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)] \notin \mathcal{L}$$

By definition of $ALTEX$, the context

$$\mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \diamond, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)]$$

is added to F, that makes the rows of t and t' different in

$$T_{\mathcal{L}}(\mathcal{E}, \mathsf{F} \cup \{\mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \diamond, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)]\})$$

Theorem 1. *The algorithm $ALTEX$ identifies the class of regular tree languages in polynomial time.*

Proof. The algorithm $ALTEX$ starts with the construction of $T_{\mathcal{L}}(\mathcal{E}, \mathcal{E}[\diamond])$ and enters the loop. If the program leaves this loop, the observation table is consistent and by Lemma 4, the automaton given as output is correct. It remains to show that $ALTEX$ terminates. From Lemma 5, each time the loop is processed, a new separating-context is added to the table. Since two rows that were different are still different in the new table, the number of states of the automaton $\mathcal{A}_T$ is strictly increased. From Lemma 3, this number is always lower or equal to the number of states of the minimal automaton for $\mathcal{L}$. This implies that the loop may be processed only a finite number of times and by consequence, $ALTEX$ terminates.

Example process of $ALTEX$ are given in Appendix A.

We noticed that any finite set containing a representative sample is also a representative sample. This implies that if we consider an incremental version of *Altex* (the table is completed as the set of positive exemples increases during the process), the algorithm converges. Now, if the input set doesn't contain a representative sample yet, the algorithm calculates a sub-automaton (the recognized language is strictly contained in the target language). The algorithm terminates on any input and the success of learning is guaranted if a representative sample is presented as input, that constitutes a weak hypothesis.

The time complexity of $ALTEX$ depends on the size n of the representative sample $\mathcal{E}$ gave by the teacher (total number of nodes) and the size m of the minimal automaton (number of states) for the language $\mathcal{L}$ to identify. The first observation table has n^2 cells. Then, the number of rows doesn't change and the number of columns increases as the table is found not to be consistent. Each time this happens, less than n new contexts are added to the table and this may happen no more than m times. This gives at most a $n \times (m \times n + n)$ sized table to construct with queries and so $k \times n \times (m \times n + n)$ steps of calculation, where k is the time needed for a single query (the problem of membership of a term to a language is linear [7]). To this is added the time of checking the consistency of the table which is polynomial in its size and finally gives a polynomial in n and m algorithm.

Then, let compare $ALTEX$ with Angluin works for learning the class of regular languages. In [1], Angluin studies the paradigm of learning regular (word) languages from positive examples and queries and in [3], the idea of observation table is introduced. It's interseting to see whether these results may be applied in the case of tree languages. Angluin's algorithms try any possible transition by considering the set of words $\omega\alpha$, where ω is a prefix and α an element of the alphabet. To apply this technique in the case of trees, we have to construct, for any subterm t of $\mathcal{E}$, all terms of the form $\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}, \ldots, \mathsf{t}_n)$, for each subterm t_i is of $\mathcal{E}$ and each element $\mathtt{f}$ of arity n in the alphabet. This method highly increases the size of the constructed table

(an exponential in the maximimum arity of the alphabet appears) and, as a consequence, the time complexity too. This justifies the original choice of considering representative sample for input and the contexts $\mathcal{E}[\diamond]$ to construct the first table. The key point is that $ALTEX$ determines useful contexts very efficiently from the non-consistent table. For similar reason, taking into account counter example for the construction of the table is not relevant in the case of trees. Counter examples are terms gave be the teacher, terms that are in the language $\mathcal{L}_{\mathcal{A}_T}$ corresponding to the table, but not in $\mathcal{L}$ (from Lemma 2, the language is not yet learned if and only if such terms exist). There are no particular reason for a counter example to easily provide a context able to separate two mistakely equals rows (in the case where several mystakes appear in different branchs of the negative example, unlike in the case of words where considering all suffixes of a counter example gives such a context). Finally, it's worth noticing that $ALTEX$ terminates even if the positive examples given in input do not contain a representative sample of the targert language. In this case, the language corresponding to the output automaton contains at least the entries.

5 Conclusion

We have showed that the whole class of regular tree language is learnable from positive examples and membership queries, that extends Angluin's works. We gave a polynomial algorithm for this learning model and proved its correctnes. It is worth noticing that, with the same proofs, $ALTEX$ is able to learn from partial examples, that is subtrees of positive examples. Indeed, in the definition of representative sample, we stipulate that a representative sample must be a subset of the language; there is no need for this restriction and the property that any transition of the minimal automaton is necessary is enough. This constitutes a new paradigm that can be even more realistic for the problem of natural language acquisition modelling. A part of a sentence can be used by a child, even if the sentence in which it appears is not totally understood. Moreover, this may partially answer the problem of noise in the signal processing.

References

1. D. Angluin. A note on the number of queries needed to identify regular languages. *Information and Control*, 51:76–87, 1981.
2. D. Angluin. Learning regular sets from queries and counter examples. *Information and Control*, 75:87–106, 1987.
3. D. Angluin. Queries and concept learning. *Machine learning*, 2:319–342, 1988.
4. J. Besombes and J.Y. Marion. Apprentissage des langages réguliers d'arbres et applications. *Conférence d'Apprentissage, Orléans 17, 18 et 19 juin 2002*, pages 55–70, 2002.
5. R.C. Carrasco, J. Oncina, and J. Calera. Stochastic inference of regular tree languages. *Lecture Notes in Computer Science*, 1433:185–197, 1998.
6. A. Christophe. L'apprentissage du langage. In *Université de tous les savoirs*, volume 2, pages 41–51. Odile Jacob, 2000.
7. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://www.grappa.univ-lille3.fr/tata, 1997.
8. A. Dikovsky and L. Modina. Dependencies on the other side of the curtain. *Traitement automatique des langues*, 41(1):67–96, 2000.
9. F. Drewes and J. Högberg. Learning a regular tree language from a teacher. In Z. Ésik and Z. Fülöp, editors, *Proc. Developments in Language Theory 2003*, volume 2710 of *Lecture Notes in Computer Science*, pages 279–291. Springer, 2003.
10. F. Drewes and J. Högberg. Learning a regular tree language from a teacher even more efficiently. Technical Report 03.11, Umea University, 2003.
11. H. Fernau. Learning tree languages from text. 2375:153–168, 2002.
12. H. Fukuda and K. Kamata. Inference of tree automata from sample set of trees. *International Journal of Computer and Information Sciences*, 13(3):177–196, 1984.
13. M.E. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
14. K. Kamata. Inference methods for tree automata from sample set of trees. *IEEE International Conference on Systems, Man and Cybernetics*, pages 490–493, 1988.
15. M. Kanazawa. *Learnable classes of Categorial Grammars*. CSLI, 1998.

16. S. Pinker. *The language instinct*. Harper, 1994.
17. Y. Sakakibara. Inductive inference of logic programs based on algebraic semantics. Technical Report ICOT, 79, 1987.
18. Y. Sakakibara. Inferring parsers of context-free languages from structural examples. Technical Report ICOT, 81, 1987.
19. Y. Sakakibara. Learning context-free grammars from structural data in polynomial time. *Theoretical Computer Science*, 76:223–242, 1990.

Appendix A - Examples

In example 4, we saw that the tree $a(b(b), c(c(c)))$ is a representative sample for the tree language

$$\mathcal{L}_{\mathcal{A}} = \{a(b^{2n+2}, c^{2m+1}) \ : \ n, m \in \mathbb{N}\}$$

Now suppose that the singleton is given to $ALTEX$; the constructed table with help of membership queries is the table of example 2. Thiq table is consistent a then, $ALTEX$ output the automaton of example 3. This automaton is a renaming of $\mathcal{A}$ and the language is learnt. If we suppose that, with the same input, the language to learn is

$$\mathcal{L}_{\mathcal{A}'} = \{a(b^{n}, c^{m}) \ : \ n, m \in \mathbb{N}^*\}$$

($\{a(b(b), c(c(c))\}$ is representative for this language too), the table is now:

	$\diamond$	$a(\diamond, c(c(c)))$	$a(b(\diamond), c(c(c)))$	$a(b(b), \diamond)$	$a(b(b), c(\diamond))$	$a(b(b), c(c(\diamond)))$
$a(b(b), c(c(c)))$	1	0	0	0	0	0
$b(b)$	0	1	1	0	0	0
b	0	1	1	0	0	0
$c(c(c))$	0	0	0	1	1	1
$c(c)$	0	0	0	1	1	1
c	0	0	0	1	1	1

$ALTEX$ checks again that this table is directly consistent and output the automaton $\phi'(\mathcal{A}')$, where ϕ' is the automata homomorphism defined by:

$$\phi'(q_F) = 1\ 0\ 0\ 0\ 0\ 0$$
$$\phi'(q_1) = 0\ 1\ 1\ 0\ 0\ 0$$
$$\phi'(q_2) = 0\ 0\ 0\ 1\ 1\ 1$$

$ALTEX$ is defined as an iterative algorithm; if during the process, the observation table is found not to be consistent, the minimal automaton for the target language must have some particular rules: rules which are identical except in a single state of its left hand side (from the first lemma given in Appendix B). If the minimal automaton has no such rules, the first table constructed by $ALTEX$ is consistent and the language is learned immediatly. With the aim of illustrating the iterative behaviour, we now propose an automaton specially constructed for having this property. Let so be $\mathcal{L}_{\mathcal{A}''}$ the language defined by the following minimal automaton $\mathcal{A}''$:

$$\begin{array}{rrrrrrrr}
a(q_1) \underset{\mathcal{A}}{\rightarrow} q_F & d(q_3, q_5) \underset{\mathcal{A}}{\rightarrow} q_1 & e(q_7) \underset{\mathcal{A}}{\rightarrow} q_3 & g \underset{\mathcal{A}}{\rightarrow} q_7 \\
a(q_2) \underset{\mathcal{A}}{\rightarrow} q_F & d(q_4, q_5) \underset{\mathcal{A}}{\rightarrow} q_1 & e(q_8) \underset{\mathcal{A}}{\rightarrow} q_4 & h \underset{\mathcal{A}}{\rightarrow} q_8 \\
b(q_1) \underset{\mathcal{A}}{\rightarrow} q_F & d(q_3, q_6) \underset{\mathcal{A}}{\rightarrow} q_1 & f(q_9) \underset{\mathcal{A}}{\rightarrow} q_5 & i \underset{\mathcal{A}}{\rightarrow} q_9 \\
c \underset{\mathcal{A}}{\rightarrow} q_1 & d(q_4, q_6) \underset{\mathcal{A}}{\rightarrow} q_2 & f(q_{10}) \underset{\mathcal{A}}{\rightarrow} q_6 & j \underset{\mathcal{A}}{\rightarrow} q_{10}
\end{array}$$

where q_F is the unique final state. From this definition, we establish that $\mathcal{L}_{\mathcal{A}''}$ is the finite language corresponding to the following set of terms:

$$\mathcal{L}_{\mathcal{A}''} = \{a(c), b(c), a(d(e(g), f(i))), a(d(e(h), f(i))), a(d(e(g), f(j))),$$

$$a(d(e(h), f(j))), b(d(e(g), f(i))), b(d(e(h), f(i))), b(d(e(g), f(j)))\}.$$

Let suppose that a teacher constructs the representative sample:

$$\mathcal{E} = \{b(c), a(d(e(g), f(i))), a(d(e(h), f(i))), a(d(e(g), f(j))), a(d(e(h), f(j)))\}.$$

Table 1. $T_{\mathcal{L}_{\mathcal{A}''}}(\mathcal{S}(\mathcal{E}), \mathcal{E}[\diamond])$

$\mathcal{S}(\mathcal{E})$ \ F	$\diamond$	$a(d(e(\diamond), f(i)))$	$a(d(e(g), f(\diamond)))$	$a(d(\diamond, f(i)))$	$a(d(e(g), \diamond))$	$a(\diamond)$	$a(d(e(h), f(\diamond)))$	$a(d(e(h), \diamond))$	$a(d(e(\diamond), f(j)))$	$a(d(\diamond, f(j)))$	$b(\diamond)$
$a(d(e(g), f(i)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(g), f(i))$	0	0	0	0	0	1	0	0	0	0	1
$e(g)$	0	0	0	1	0	0	0	0	0	1	0
$f(i)$	0	0	0	0	1	0	0	1	0	0	0
g	0	1	0	0	0	0	0	0	1	0	0
i	0	0	1	0	0	0	1	0	0	0	0
$a(d(e(h), f(i)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(h), f(i))$	0	0	0	0	0	1	0	0	0	0	1
$e(h)$	0	0	0	1	0	0	0	0	0	1	0
h	0	1	0	0	0	0	0	0	1	0	0
$a(d(e(g), f(j)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(g), f(j))$	0	0	0	0	0	1	0	0	0	0	1
$f(j)$	0	0	0	0	1	0	0	1	0	0	0
j	0	0	1	0	0	0	1	0	0	0	0
$a(d(e(h), f(j)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(h), f(j))$	0	0	0	0	0	1	0	0	0	0	0
$b(c)$	1	0	0	0	0	0	0	0	0	0	0
c	0	0	0	0	0	1	0	0	0	0	1

The learner $ALTEX$ starts with the construction of $\mathsf{F} = \mathcal{E}[\diamond]$ and $T = T_{\mathcal{L}_{\mathcal{A}''}}(\mathcal{S}(\mathcal{E}), \mathsf{F})$ (table 1).

$ALTEX$ now notices the three problematics couples of terms

$$d(e(h), f(i)) \text{ and } d(e(h), f(j)),$$

$$d(e(g), f(i)) \text{ and } d(e(h), f(j))$$

and

$$d(e(g), f(j)) \text{ and } d(e(h), f(j)).$$

Indeed

$$row_T(e(g)) = row_T(e(h))$$

and

$$row_T(f(i)) = row_T(f(j))$$

but

$$b(d(e(g), f(i))) \in \mathcal{L},$$
$$b(d(e(g), f(j))) \in \mathcal{L},$$
$$b(d(e(h), f(i))) \in \mathcal{L}$$

and

$$b(d(e(h), f(j))) \notin \mathcal{L}.$$

$ALTEX$ adds the contexts

$$b(d(e(g), \diamond)),$$
$$b(d(e(h), \diamond)),$$
$$b(d(\diamond, f(i)))$$

and

$$b(d(\diamond, f(j)))$$

to F and complete the table T with help of the teacher (table 2).

$ALTEX$ now remarks that

$$b(d(e(h), f(i))) \in \mathcal{L},$$
$$b(d(e(h), f(j))) \notin \mathcal{L}$$

but

$$row_T(i) = row_T(j)$$

and that

$$b(d(e(g), f(j))) \in \mathcal{L},$$
$$b(d(e(h), f(j))) \notin \mathcal{L}$$

but

$$row_T(g) = row_T(h).$$

It takes the decision to consider the new contexts

$$b(d(e(\diamond), f(j)))$$

and

$$b(d(e(h), f(\diamond)))$$

in F and complete one last time the table T with help of the teacher (table 3). T is now consistent and $ALTEX$ output $\mathcal{A}_T$ which verify $\mathcal{L}_{\mathcal{A}_T} = \mathcal{L}_{\mathcal{A}''}$.

Table 2. $T_{\mathcal{L}_{\mathcal{A}''}}(\mathcal{S}(\mathcal{E}), \mathcal{E}[\diamond] \cup \{b(d(e(g),\diamond)), b(d(e(h),\diamond)), b(d(\diamond, f(i)))\}), b(d(\diamond, f(j)))\})$

$\mathcal{S}(\mathcal{E})$ \ F	$\diamond$	$a(d(e(\diamond), f(i)))$	$a(d(e(g), f(\diamond)))$	$a(d(\diamond, f(i)))$	$a(d(e(g),\diamond))$	$a(\diamond)$	$a(d(e(h), f(\diamond)))$	$a(d(e(h),\diamond))$	$a(d(e(\diamond), f(j)))$	$a(d(\diamond, f(j)))$	$b(\diamond)$
$a(d(e(g), f(i)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(g), f(i))$	0	0	0	0	0	1	0	0	0	0	1
$e(g)$	0	0	0	1	0	0	0	0	0	1	0
$f(i)$	0	0	0	0	1	0	0	1	0	0	0
g	0	1	0	0	0	0	0	0	1	0	0
i	0	0	1	0	0	0	1	0	0	0	0
$a(d(e(h), f(i)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(h), f(i))$	0	0	0	0	0	1	0	0	0	0	1
$e(h)$	0	0	0	1	0	0	0	0	0	1	0
h	0	1	0	0	0	0	0	0	1	0	0
$a(d(e(g), f(j)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(g), f(j))$	0	0	0	0	0	1	0	0	0	0	1
$f(j)$	0	0	0	0	1	0	0	1	0	0	0
j	0	0	1	0	0	0	1	0	0	0	0
$a(d(e(h), f(j)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(h), f(j))$	0	0	0	0	0	1	0	0	0	0	0
$b(c)$	1	0	0	0	0	0	0	0	0	0	0
c	0	0	0	0	0	1	0	0	0	0	1

$\mathcal{S}(\mathcal{E})$ \ F	$b(d(e(g),\diamond))$	$b(d(e(h),\diamond))$	$b(d(\diamond, f(i)))$	$b(d(\diamond, f(j)))$
$a(d(e(g), f(i)))$	0	0	0	0
$d(e(g), f(i))$	0	0	0	0
$e(g)$	0	0	1	1
$f(i)$	1	1	0	0
g	0	0	0	0
i	0	0	0	0
$a(d(e(h), f(i)))$	0	0	0	0
$d(e(h), f(i))$	0	0	0	0
$e(h)$	0	0	1	0
h	0	0	0	0
$a(d(e(g), f(j)))$	0	0	0	0
$d(e(g), f(j))$	0	0	0	0
$f(j)$	1	0	0	0
j	0	0	0	0
$a(d(e(h), f(j)))$	0	0	0	0
$d(e(h), f(j))$	0	0	0	0
$b(c)$	0	0	0	0
c	0	0	0	0

Table 3. $T_{\mathcal{L}_{\mathcal{A}''}}(\mathcal{S}(\mathcal{E}), \mathcal{E}[\diamond] \cup \{b(d(e(g),\diamond)), b(d(e(h),\diamond)), b(d(\diamond, f(j))), b(d(e(\diamond), f(j))), b(d(e(h), f(\diamond)))\})$

$\mathcal{S}(\mathcal{E})$ \ F	$\diamond$	$a(d(e(\diamond), f(i)))$	$a(d(e(g), f(\diamond)))$	$a(d(\diamond, f(i)))$	$a(d(e(g),\diamond))$	$a(\diamond)$	$a(d(e(h), f(\diamond)))$	$a(d(e(h),\diamond))$	$a(d(e(\diamond), f(j)))$	$a(d(\diamond, f(j)))$	$b(\diamond)$
$a(d(e(g), f(i)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(g), f(i))$	0	0	0	0	0	1	0	0	0	0	1
$e(g)$	0	0	0	1	0	0	0	0	0	1	0
$f(i)$	0	0	0	0	1	0	0	1	0	0	0
g	0	1	0	0	0	0	0	0	1	0	0
i	0	0	1	0	0	0	1	0	0	0	0
$a(d(e(h), f(i)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(h), f(i))$	0	0	0	0	0	1	0	0	0	0	1
$e(h)$	0	0	0	1	0	0	0	0	0	1	0
h	0	1	0	0	0	0	0	0	1	0	0
$a(d(e(g), f(j)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(g), f(j))$	0	0	0	0	0	1	0	0	0	0	1
$f(j)$	0	0	0	0	1	0	0	1	0	0	0
j	0	0	1	0	0	0	1	0	0	0	0
$a(d(e(h), f(j)))$	1	0	0	0	0	0	0	0	0	0	0
$d(e(h), f(j))$	0	0	0	0	0	1	0	0	0	0	0
$b(c)$	1	0	0	0	0	0	0	0	0	0	0
c	0	0	0	0	0	1	0	0	0	0	1

$\mathcal{S}(\mathcal{E})$ \ F	$b(d(e(g),\diamond))$	$b(d(e(h),\diamond))$	$b(d(\diamond, f(i)))$	$b(d(\diamond, f(j)))$	$b(d(e(\diamond), f(j)))$	$b(d(e(h), f(\diamond)))$
$a(d(e(g), f(i)))$	0	0	0	0	0	0
$d(e(g), f(i))$	0	0	0	0	0	0
$e(g)$	0	0	1	1	0	0
$f(i)$	1	1	0	0	0	0
g	0	0	0	0	1	0
i	0	0	0	0	0	1
$a(d(e(h), f(i)))$	0	0	0	0	0	0
$d(e(h), f(i))$	0	0	0	0	0	0
$e(h)$	0	0	1	0	0	0
h	0	0	0	0	0	0
$a(d(e(g), f(j)))$	0	0	0	0	0	0
$d(e(g), f(j))$	0	0	0	0	0	0
$f(j)$	1	0	0	0	0	0
j	0	0	0	0	0	0
$a(d(e(h), f(j)))$	0	0	0	0	0	0
$d(e(h), f(j))$	0	0	0	0	0	0
$b(c)$	0	0	0	0	0	0
c	0	0	0	0	0	0

Appendix B

In this section, we proove technical lemma used to establish the correctness of $ALTEX$.

Lemma. Let $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$ be an observation table where $\mathcal{L}$ is a regular tree language, $\mathcal{E}$ a representative set for $\mathcal{L}$, F a set of context containing $\mathcal{E}[\diamond]$ and $\mathcal{A}$ the minimal automaton for $\mathcal{L}$.

For any pair of terms t and t' of $\mathcal{E}$, if $row_T(\mathsf{t}) = row_T(\mathsf{t}')$, then for any rule of the form

$$\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{t}), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r$$

in $\mathcal{A}$, there is also a rule of the form

$$\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{t}'), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r'$$

in $\mathcal{A}$.

Proof. Suppose $row_T(\mathsf{t}) = row_T(\mathsf{t}')$ and let $\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{t}), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r$ be a rule of $\mathcal{A}$. Since F contains $\mathcal{E}[\diamond]$, there is a context of the form $\mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \diamond, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)]$ in F, where $\delta_{\mathcal{A}}(\mathsf{t}_j) = q_j, \forall 1 \leq j \neq i \leq n$. From the definition of $T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$, we have $T_{\mathcal{L}}(\mathsf{t}, \mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \diamond, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)]) = 1$. Now $row_T(\mathsf{t}) = row_T(\mathsf{t}')$ implies that $T_{\mathcal{L}}(\mathsf{t}', \mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \diamond, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)]) = 1$. As $\mathsf{c}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}', \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)]$ is an element of $\mathcal{L}$, there must be a rule $\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{t}), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r'$ in $\mathcal{A}$.

Lemma. Let $T = T_{\mathcal{L}}(\mathcal{E}, \mathsf{F})$ be an observation table where $\mathcal{L}$ is a regular tree language, $\mathcal{E}$ a representative set for $\mathcal{L}$, F a set of context containing $\mathcal{E}[\diamond]$ and $\mathcal{A}$ the minimal automaton for $\mathcal{L}$.

If $\mathcal{L}_{\mathcal{A}_T} \neq \mathcal{L}$, then exist two terms

$$\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)$$

and

$$\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)$$

in $\mathcal{S}(\mathcal{E})$ such that:

- $row_T(\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)) \neq row_T(\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n))$,
- $\delta_{\mathcal{A}}(\mathsf{t}_j) = \delta_{\mathcal{A}}(\mathsf{t}'_j), \forall 1 \leq j \neq i \leq n$,
- $row_T(\mathsf{t}) = row_T(\mathsf{t}')$.

Proof. $\mathcal{L}_{\mathcal{A}_T} \neq \mathcal{L}$ implies that the automata homomorphism ϕ is not an isomorphism; there are so two terms t and t' in $\mathcal{E}$ such that:

$$row_T(\mathsf{t}) = row_T(\mathsf{t}')$$

and

$$\delta_{\mathcal{A}}(\mathsf{t}) \neq \delta_{\mathcal{A}}(\mathsf{t}')$$

From Myhill-Nerode theorem [7], $\delta_{\mathcal{A}}(\mathsf{t}) \neq \delta_{\mathcal{A}}(\mathsf{t}')$ implies the existence of a context $\mathsf{c}[\diamond]$ such that $\mathsf{c}[\mathsf{t}] \in \mathcal{L}$ and $\mathsf{c}[\mathsf{t}'] \notin \mathcal{L}$.

Let now define the context $\mathsf{c}[\diamond]$ as:

- there are two terms s and s' in $\mathcal{S}(\mathcal{E})$ such that
  - $row_T(\mathsf{s}) = row_T(\mathsf{s}')$
  - $\mathsf{c}[\mathsf{s}] \in \mathcal{L}$ and $\mathsf{c}[\mathsf{s}'] \notin \mathcal{L}$

– $\mathsf{depth}(\mathsf{c}[\diamond]) = \min\{\mathsf{depth}(\mathsf{c}'[\diamond]), \exists \mathsf{u}, \mathsf{u}' \in \mathcal{S}(\mathcal{E}), row_T(\mathsf{u}) = row_T(\mathsf{u}'),$
$\mathcal{L}$ and $\mathsf{c}'[\mathsf{u}'] \notin \mathcal{L}\}$

Let first show that $\mathsf{depth}(\mathsf{c}[\diamond]) > 0$. If $\mathsf{depth}(\mathsf{c}[\diamond]) = 0$, we have $\mathsf{c}[\diamond] = \diamond$. Now $\diamond$ is a context of F, so if $\mathsf{s} \in \mathcal{L}$ and $\mathsf{s}' \notin \mathcal{L}$, then $row_T(\mathsf{s}) \neq row_T(\mathsf{s}')$ which provides a contradiction. So $\mathsf{c}[\diamond]$ can be written $\mathsf{d}[\mathtt{f}(\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \diamond, \mathsf{s}_i, \ldots, \mathsf{s}_n)]$, with some terms $\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \mathsf{s}_{i+1}, \ldots, \mathsf{s}_n$ and $\mathsf{d}[\diamond]$ a context such that $\mathsf{depth}(\mathsf{c}[\diamond]) = \mathsf{depth}(\mathsf{d}[\diamond]) + 1$. Since $\mathsf{d}[\mathtt{f}(\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \mathsf{s}, \mathsf{s}_{i+1}, \ldots, \mathsf{s}_n)]$ is in $\mathcal{L}$, there exist in $\mathcal{A}$ a rule $\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{s}), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r$, with $q_j = \delta_{\mathcal{A}}(\mathsf{s}_j)$, $\forall 1 \leq j \neq i \leq n$. From the former Lemma, there is also a rule $\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{s}'), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r'$ in $\mathcal{A}$. If $r = r'$, then we have

$$\delta_{\mathcal{A}}(\mathtt{f}(\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \mathsf{s}, \mathsf{s}_{i+1}, \ldots, \mathsf{s}_n)) = \delta_{\mathcal{A}}(\mathtt{f}(\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \mathsf{s}', \mathsf{s}_{i+1}, \ldots, \mathsf{s}_n))$$

and $\mathsf{d}[\mathtt{f}(\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \mathsf{s}, \mathsf{s}_{i+1}, \ldots, \mathsf{s}_n)]$ is in $\mathcal{L}$ implies $\mathsf{d}[\mathtt{f}(\mathsf{s}_1, \ldots, \mathsf{s}_{i-1}, \mathsf{s}, \mathsf{s}_{i+1}, \ldots, \mathsf{s}_n)]$ is in $\mathcal{L}$, that contradicts the hypothesis. $\mathcal{E}$ is a representative sample for $\mathcal{L}$, so to the two rules

$$\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{s}), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r$$

and

$$\mathtt{f}(q_1, \ldots, q_{i-1}, \delta_{\mathcal{A}}(\mathsf{s}'), q_{i+1}, \ldots, q_n) \underset{\mathcal{A}}{\rightarrow} r'$$

correspond two terms

$$\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)$$

and

$$\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)$$

in $\mathcal{S}(\mathcal{E})$, with

$$\delta_{\mathcal{A}}(\mathsf{t}_j) = \delta_{\mathcal{A}}(\mathsf{t}'_j) = q_j,$$

$\forall 1 \leq j \neq i \leq n$ and

$$\delta_{\mathcal{A}}(\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)) = r$$

and

$$\delta_{\mathcal{A}}(\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)) = r'.$$

Let finally suppose that

$$row_T(\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)) = row_T(\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n))$$

and show that the context $\mathsf{d}[\diamond]$ contradicts the minimality of $\mathsf{c}[\diamond]$. By construction :

– $row_T(\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)) = row_T(\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n))$
– $\mathsf{d}[\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)] \in \mathcal{L}$ and $\mathsf{d}[\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n)] \notin \mathcal{L}$
– $\mathsf{depth}(\mathsf{c}[\diamond]) > \mathsf{depth}(\mathsf{d}[\diamond])$

This contradiction implies that

$$row_T(\mathtt{f}(\mathsf{t}_1, \ldots, \mathsf{t}_{i-1}, \mathsf{t}, \mathsf{t}_{i+1}, \ldots, \mathsf{t}_n)) \neq row_T(\mathtt{f}(\mathsf{t}'_1, \ldots, \mathsf{t}'_{i-1}, \mathsf{t}', \mathsf{t}'_{i+1}, \ldots, \mathsf{t}'_n))$$

and provides the conclusion.