

ICGI 2008:

A Learning Algorithm for Multi-dimensional Trees, or: Learning Beyond Context-Freeness

Anna Kasprzik – University of Trier, Germany

kasprzik@informatik.uni-trier.de

Contents

- Part I – Formal groundwork: Multi-dimensional trees
- Part II – The adapted MAT learning algorithm L^* for multi-dimensional trees

Introduction

- It is possible to learn cf string languages via their connection (yield) to regular tree languages.
- We would like to show that this MAT-learnable class extends even further.
- We do that via *multi-dimensional trees* as structural descriptions.
- In order to obtain that result we have to introduce a new term-like notation for multi-dimensional trees which establishes them as a *direct and natural generalization* of classical trees.

Preliminaries – Trees

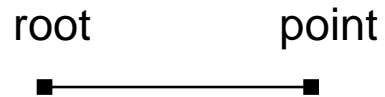
- The set T_Σ of all trees over a ranked alphabet Σ is defined inductively as the smallest set of expressions st
 - $f \in T_\Sigma$ for every $f \in \Sigma_0$ and
 - $f[t_1, \dots, t_n] \in T_\Sigma$ for every $f \in \Sigma_n$ and $t_1, \dots, t_n \in T_\Sigma$.
- \square be a special symbol of rank 0. A tree $c \in T_{\Sigma \cup \{\square\}}$ in which \square occurs exactly once is a *context*, the set of all contexts over Σ is C_Σ . For $c \in C_\Sigma$ and $s \in T_\Sigma$, $c[[s]]$ denotes the tree obtained by substituting s for \square in c .

Multi-dimensional trees (Rogers 2003)

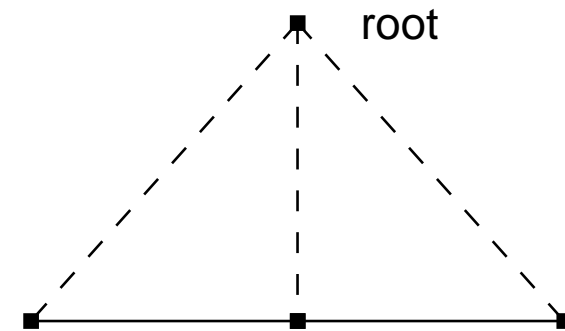
zero-dimensional tree
(point)



one-dimensional tree (string)



simple (local) two-dimensional tree

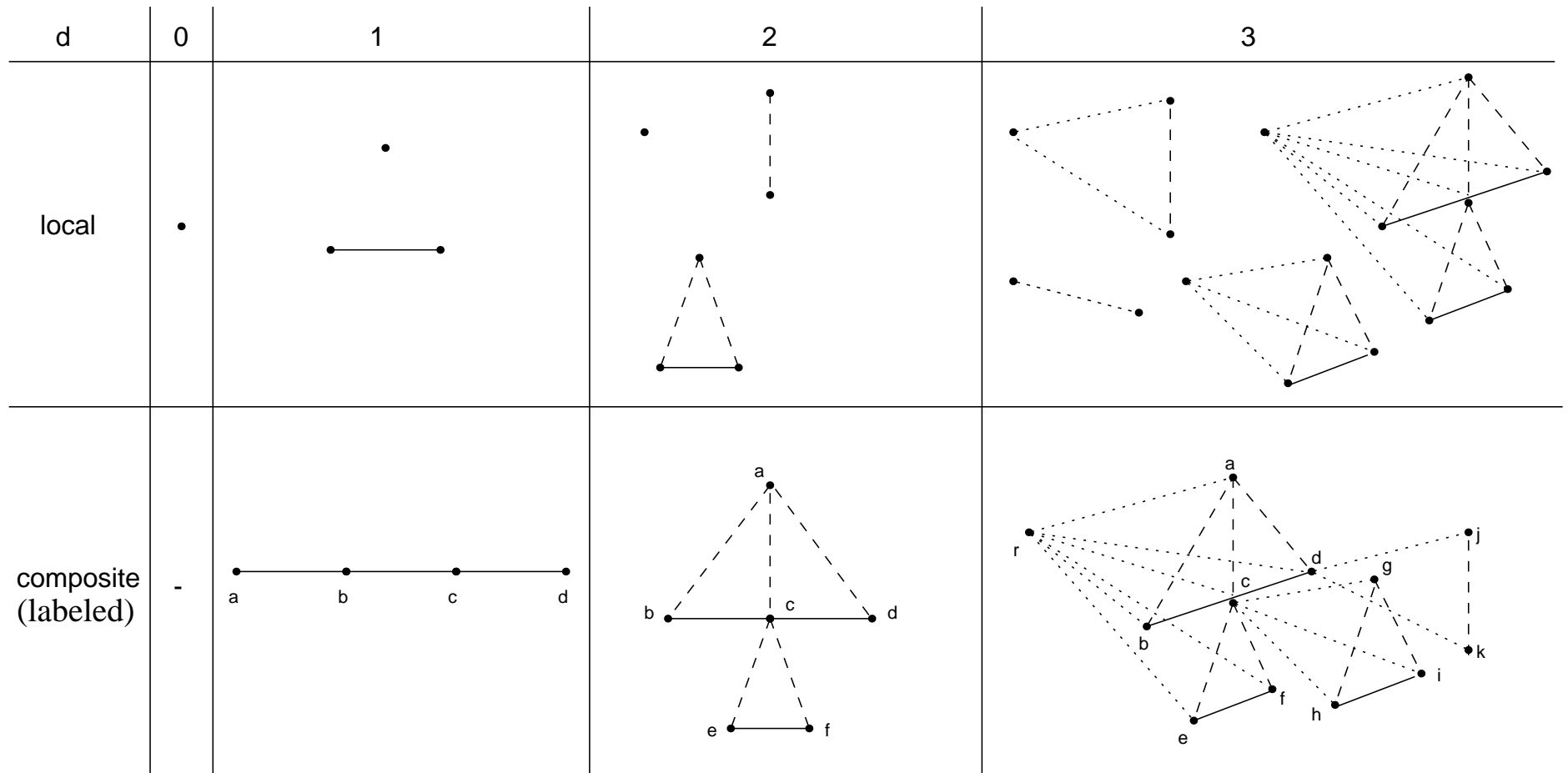


complex one-dimensional tree

complex one-dimensional tree (string)

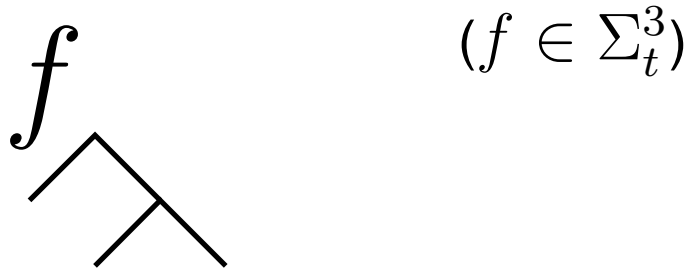


Multi-dimensional trees (Rogers 2003)



d -dimensional tree labeling alphabets

We will use finite d -dimensional tree labeling alphabets Σ^d where each symbol $f \in \Sigma^d$ is associated with at least one unlabeled $(d - 1)$ -dimensional tree t specifying the admissible child structure for a root labeled with f .



Let Σ_t^d for $d \geq 1$ be the set of all symbols associated with t .

Multi-dimensional trees (over Σ_t^d)

Let Σ^0 be a set of constant symbols. The set \mathbb{T}_{Σ^d} of all d -dimensional trees can be defined inductively as follows:

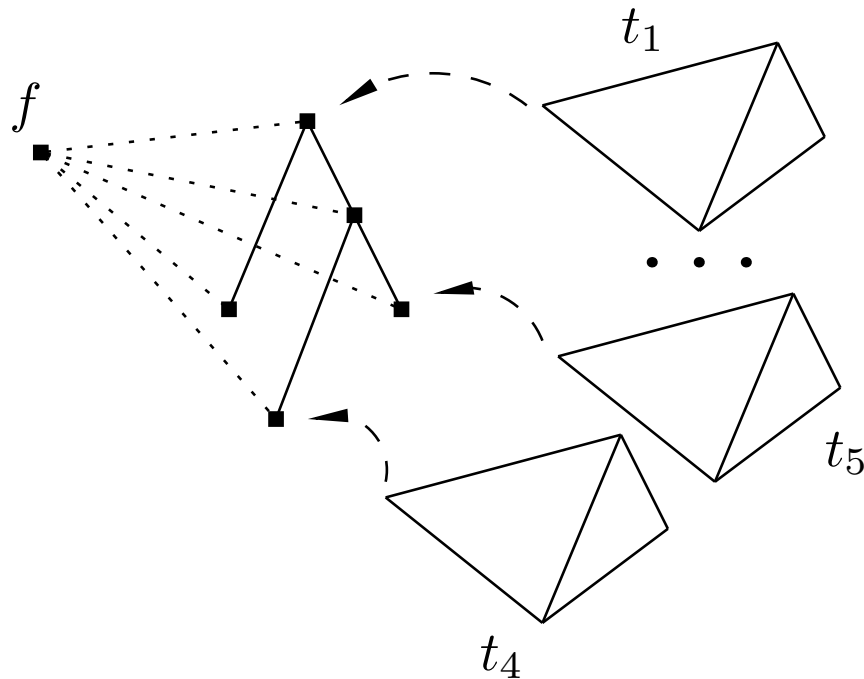
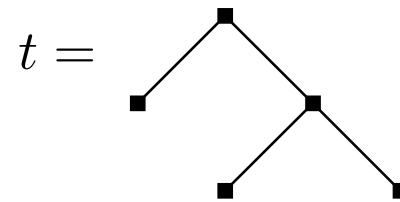
Definition 1 *Let ε^d be the empty d -dimensional tree. Then*

- $\mathbb{T}_{\Sigma^0} := \{\varepsilon^0\} \cup \Sigma^0$, and
- for $d \geq 1$: \mathbb{T}_{Σ^d} is the smallest set such that $\varepsilon^d \in \mathbb{T}_{\Sigma^d}$ and $f[t_1, \dots, t_n]_t \in \mathbb{T}_{\Sigma^d}$ for every $f \in \Sigma_t^d$, n the number of nodes in t , $t_1, \dots, t_n \in \mathbb{T}_{\Sigma^d}$ and t_1, \dots, t_n are rooted breadth-first in that order at the nodes of t .

Contexts are defined as before (\square being a symbol associated with ε^{d-1}).

Multi-dimensional trees – intuition

$$f \in \Sigma_t^3$$



Multi-dim. finite-state tree automata

Definition 2 A finite-state d -dimensional tree automaton is a quadruple $\mathcal{A}^d = (\Sigma^d, Q, \delta, F)$ with

- input alphabet Σ^d ,
- finite set of states Q ,
- set of accepting states $F \subseteq Q$ and
- transition function δ with $\delta(t(q_1, \dots, q_n), f) \in Q$ for every $f \in \Sigma_t^d$ where $t(q_1, \dots, q_n)$ encodes the assignment of states to the nodes of t .

$\delta : \mathbb{T}_{\Sigma^d} \longrightarrow Q$ is defined such that if $t_p = f[t_1, \dots, t_n]_t \in \mathbb{T}_{\Sigma^d}$ then $\delta(t_p) = \delta(t(\delta(t_1), \dots, \delta(t_n)), f)$.

Myhill-Nerode (for multi-dim. trees)

$L \subseteq \mathbb{T}_{\Sigma^d}$. For $s, s' \in \mathbb{T}_{\Sigma^d}$, $s \sim_L s'$ iff for every $c \in C_{\Sigma^d}$, $c[[s]] \in L \Leftrightarrow c[[s']] \in L$. The *index* of L is $|\{[s]_L | s \in \mathbb{T}_{\Sigma^d}\}|$.

Theorem 3 L is regular iff L is of finite index.

Corollary. If a tree language is of finite index, we can build an fta A_L^d recognizing L , with $Q = \{[s]_L | s \in \mathbb{T}_{\Sigma^d}\}$, $F = \{[s]_L | s \in L\}$, and, given some $f \in \Sigma_t^d$ and states $[s_1]_L, \dots, [s_n]_L$, $\delta_L(t([s_1]_L, \dots, [s_n]_L), f) = [f[s_1, \dots, s_n]t]_L$.

A_L^d is the unique minimal fta recognizing L (up to a bijective renaming of states).

Multi-dimensional trees – yield

- The (direct) yield of a d -dimensional tree is a projection on the $(d - 1)$ -dimensional level.
- The string yield of a d -dimensional tree can be obtained by taking the direct yield $(d - 1)$ times.

Part II: Learning algorithm (multidim.)

The learner is helped by a teacher who can answer membership and equivalence queries (and return a counterexample). He maintains an *observation table*.

Definition 4 The pair (S, C) ($S \subseteq \mathbb{T}_{\Sigma^d}$, $C \subseteq C_{\Sigma^d}$ finite, $C \neq \emptyset$) is called an observation table if the following holds:

- For every (*d-dimensional*) tree $f[s_1, \dots, s_n]_t \in S$: $s_1, \dots, s_n \in S$ as well – S is subtree-closed, and
- for every context $c_0 \in C$ of the form $c[[f[s_1, \dots, s_{i-1}, \square, s_{i+1}, \dots, s_n]_t]] \in C$: $c \in C$ and $s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n \in S$ – we say that C is generalization-closed.

Learning algorithm (multidim.)

Definition 5 *Observation table* $T = (S, C)$ is closed if $obs_T(\Sigma^d(S)) \subseteq obs_T(S)$, and consistent if, for all $f \in \Sigma_t^d$ and all $s_1, \dots, s_n, s'_1, \dots, s'_n \in S$, if $obs_T(s_i) = obs_T(s'_i)$ for all i with $1 \leq i \leq n$ then $obs_T(f[s_1, \dots, s_n]_t) = obs_T(f[s'_1, \dots, s'_n]_t)$.

From a closed and consistent OT $T = (S, C)$ one can synthesize an fta \mathcal{A}_T^d with $Q_T = \{obs_T(s) | s \in S\}$ as set of states, $F_T = \{obs_T(s) | s \in S \cap U\}$ as set of accepting states, and $\delta_T(t(obs_T(s_1) \cdots obs_T(s_n)), f) = obs_T(f[s_1, \dots, s_n]_t)$ for all $f \in \Sigma_t^d$ and $s_1, \dots, s_n \in S$.

The algorithm

```
 $T = (S, C) := (\{a\}, \{\square\})$  for some arbitrary  $a \in \Sigma_{\varepsilon^{d-1}}^d$ ;  
while  $|\{obs_T(s) \mid s \in S\}| < I$  do  
  if  $T$  is not closed then  $T := \text{CLOSURE}(T)$   
  else if  $T$  is not consistent then  $T := \text{RESOLVE}(T)$   
  else  $T := \text{EXTEND}(T)$   
end while;  
return  $\mathcal{A}_T^d$ ;
```

```
procedure  $\text{CLOSURE}(T)$  where  $T = (S, C)$   
  find  $s \in \Sigma^d(S)$  such that  $obs_T(s) \notin obs_T(S)$ ;  
  return  $(S \cup \{s\}, C)$ ;
```

The algorithm

```
procedure RESOLVE( $T$ ) where  $T = (S, C)$ 
  find  $c[[s]], c[[s']] \in \Sigma^d(S)$  where  $s, s' \in S$  and
   $depth(c) = 1$  such that
     $obs_T(c[[s]]) \neq obs_T(c[[s']])$  and  $obs_T(s) = obs_T(s')$ ;
  find  $t, t' \in S$  such that
     $obs_T(t) = obs_T(c[[s]])$  and  $obs_T(t') = obs_T(c[[s']])$ ;
  find  $c' \in C$  such that  $obs_T(t)(c') \neq obs_T(t')(c')$ ;
  return  $(S, C \cup \{c'[[c]]\})$ ;
```


The algorithm

procedure EXTEND(T) where $T = (S, C)$

$\mathcal{A}_T^d := \text{synthesize}(T);$

return EXTRACT($T, \text{counterexample}(\mathcal{A}_T^d)$);

procedure EXTRACT(T, t) where $T = (S, C)$

choose $c \in C_{\Sigma^d}$ and $s \in \text{subtrees}(t) \cap (\Sigma^d(S) \setminus S)$

such

that $t = c[[s]];$

if there exists $s' \in S$ such that

$\text{obs}_T(s') = \text{obs}_T(s)$ and $t \in U \Leftrightarrow c[[s']] \in U$ then

return EXTRACT($T, c[[s']]$);

else return $(S \cup \{s\}, C)$

end if;

Conclusion

Theorem 6 *The learner returns the unique minimal automaton \mathcal{A}_U^d for U (up to a bijective renaming of states) after less than $2I$ loop executions.*

- We have shown that the algorithm by Drewes and Högberg [2] can be used in an almost unchanged form to learn multi-dimensional trees, with the new notation.
- Consequently the algorithm is able to learn even string languages beyond the cf class, via the yield function.

Thank you!

References

- [1] Angluin, D.: Learning regular sets from queries and counterexamples. *Information and Computation* 75(2), 87–106 (1987)
- [2] Drewes, F., Högberg, J.: Learning a Regular Tree Language from a Teacher. In: *Developments in Lang. Th.* 2003. LNCS, vol. 2710, pp. 279–291. Springer (2003)
- [3] Rogers, J.: Syntactic Structures as Multi-dimensional Trees. *Research on Language and Computation* 1, 265–305 (2003)
- [4] Rogers, J.: wMSO Theories as Grammar Formalisms. *Theoretical Computer Science* 293, 291–320 (2003)
- [5] Kasprzik, A.: Making Finite-State Methods Applicable to Languages Beyond Context-Freeness via Multi-dimensional Trees. Technical Report 08-3, University of Trier. Available on: urts117.uni-trier.de/cms/index.php?id=15939
(to appear in the Post-Proceedings of FSMNLP 2008)