TM-0498

Learnig Context-Free Grammars from Structural Data in Polynomial Time

by

Y. Sakakibara

May, 1988

Learning Context-Free Grammars from Structural Data in Polynomial Time

Yasubumi SAKAKIBARA[1]

International Institute for Advanced Study of Social Information Science,
FUJITSU LIMITED, 140 Miyamoto, Numazu, Shizuoka 410-03, JAPAN

abstract

We consider the problem of learning a context-free grammar from structural data. Structural data of a context-free grammar are unlabelled derivation trees of the grammar. We present an efficient algorithm for learning context-free grammars using two types of queries: structural equivalence queries and structural membership queries. The learning protocol is based on what is called "minimally adequate teacher", and it is shown that a grammar learned by the algorithm is not only a correct grammar, i.e., equivalent to the unknown grammar but also structurally equivalent to it. Furthermore, the algorithm runs in time polynomial in the number of states of the minimum frontier-to-root tree automaton for the structural description of the unknown grammar and the maximum size of any counter-example.

[1]This is part of the work in the major R&D of FGCP, conducted under program set up by MITI.

1 Introduction

Angluin [1] shows that the regular sets can be learned by an algorithm using equivalence queries and membership queries in time polynomial in the number of states of the minimum deterministic finite automaton for the unknown set and the maximum length of any counter-example. However the question of whether there is an analogous result for the full class of context-free languages is open. Recently, Berman and Roos [3], and Angluin [2] give partial solutions for this question. In [2], Angluin assumes non-terminal membership queries.

In this paper, we give another partial solution by demonstrating an algorithm to learn context-free grammars from structural data. We assume that information on the structure of the unknown grammar is available to the learning algorithm. Structural data of a context-free grammar are unlabelled derivation trees of the grammar. The unlabelled derivation trees, called *structural description*, of a context-free grammar constitutes a rational set of trees. Based on this observation, the problem of learning a context-free grammar from structural data is reduced to the problem of learning a tree automaton. Then by extending Angluin's efficient learning algorithm for finite automata [1] to the one for tree automata, we can get an efficient learning algorithm for context-free grammars.

As for a related early work, Crespi-Reghizzi [5] shows that a subclass of context-free grammars can be learned efficiently from positive structural data. In [6], Levy and Joshi show a theoretical framework for grammatical inference in terms of structural descriptions. However the algorithm described in this paper is the one and only algorithm that learns the full class of context-free grammars from structural data and achieves the polynomial time bound.

2 Preliminaries

A *ranked alphabet* V is a finite set of symbols associated with a relation called the *rank* relation $r_V \subseteq V \times \{0, 1, 2, \dots, m\}$. V_n denotes the subset $\{f \in V \mid (f, n) \in r_V\}$ of V. Especially, we call V_0, denoted Σ (i.e. $\Sigma = V_0$), the *terminal alphabet.* A *tree* over V is a mapping $t : Dom_t \mapsto V$, which labels the nodes of the tree domain Dom_t. V^T denotes the set of all trees over V. A *terminal node* in Dom_t is one which has no descendant. Let \$ be a new symbol of rank 0. $V_\$^T$ denotes the subset of $(V \cup \$)^T$ that is the set of all trees which exactly contains one \$-symbol. For trees $s \in V_\$^T$ and $t \in V^T$, we define an operation "#" to replace the node labelled \$ of s with t by $s\#t = \{(x, A) \mid s(x) = A \text{ and } A \neq \$\} \cup \{(x \cdot y, A) \mid s(x) = \$, t(y) = A \text{ and } y \in Dom_t\}$. For example, if $s \in V_\$^T$ is of the form : $A(c, \$)$, and $t \in V^T$ is of the form : $B(d)$, then $s\#t$ is the tree of the form : $A(c, B(d))$. Note that $s\#t$ is in V^T. For subsets $S \subseteq V_\$^T$ and $T \subseteq V^T$, $S\#T$ ($\subseteq V^T$) is defined to be the set $\{s\#t \mid s \in S \text{ and } t \in T\}$.

A *(deterministic frontier-to-root) tree automaton* is a quadruple $A = (Q, V, \delta, F)$ such that Q is a finite set, F is a subset of Q, and $\delta = (\delta_0, \delta_1, \dots, \delta_m)$ consists of the following maps :

$$\delta_k : V_k \times (Q \cup \Sigma)^k \mapsto Q \qquad (k = 1, 2, \dots, m),$$

$$\delta_0(a) = a \qquad \text{for } a \in \Sigma.$$

Q is the set of *states*, F is the set of *final states* of A, and δ is the *state transition function* of A. In this definition, the terminal symbols on the terminal nodes are taken as "initial" states. If δ is a state transition function from $V_k \times (Q \cup \Sigma)^k$ to 2^Q, then A is *nondeterministic.* δ can be extended to V^T by letting :

$$\delta(f(t_1, \ldots, t_k)) = \begin{cases} \delta_k(f, \delta(t_1), \ldots, \delta(t_k)) & \text{for } k > 0 \text{ and } f \in V_k, \\ \delta_0(f) & \text{for } k = 0 \text{ and } f \in \Sigma. \end{cases}$$

The tree t is *accepted* by A iff $\delta(t) \in F$. The set of trees accepted by A, denoted $T(A)$, is defined as $T(A) = \{t \in V^T \mid \delta(t) \in F\}$. Let A be a tree automaton which accepts a set of trees T. A is *minimum* iff A has the minimum number of states among all tree automata which accept T. The minimum tree automaton is unique up to isomorphism [4].

Let $G = (N, \Sigma, P, S)$ be a context-free grammar. If $A \rightarrow \beta$ is a production of P and α and γ are any strings in $(N \cup \Sigma)^*$, then $\alpha A \gamma \Rightarrow \alpha\beta\gamma$. $\overset{*}{\Rightarrow}$ is the reflexive and transitive closure of $\Rightarrow$. The *language generated* by G, denoted $L(G)$, is $\{w \mid w \text{ is in } \Sigma^* \text{ and } S \overset{*}{\Rightarrow} w\}$. Two context-free grammars G_1 and G_2 are said to be *equivalent* if $L(G_1) = L(G_2)$. G is called a *wide-sense context-free grammar* if G is a context-free grammar but S is a subset of N, the set of starting symbols. A *parenthesis grammar* is a context-free grammar $G = (N, \Sigma, P, S)$ such that the productions in P are restricted to the form $A \rightarrow [\alpha]$, where [and] are special symbols not in Σ and α contains neither [nor]. Without loss of generality, we restrict our consideration to only ϵ-free context-free grammars.

For A in $N \cup \Sigma$, the set $D_A(G)$ of trees over $N \cup \Sigma$ is recursively defined as :

$$D_A(G) = \begin{cases} \{a\} & \text{for } A = a \in \Sigma, \\ \{A(t_1, \ldots, t_k) \mid A \Rightarrow B_1 \cdots B_k,\ t_i \in D_{B_i}(G)\ (1 \le i \le k)\} & \text{for } A \in N. \end{cases}$$

A tree in $D_A(G)$ is called a *derivation tree* of G from A. For the set $D_S(G)$ of derivation trees of G from the start symbol S, the S-subscript will be deleted. Further for a wide-sense context-free grammar G, $D(G) = \cup_{S' \in S} D_{S'}(G)$ where S is the set of starting symbols.

A *skeletal alphabet* Sk is a ranked alphabet consisting of only the special symbol σ with the rank relation $r_{Sk} \subseteq \{\sigma\} \times \{1, 2, \ldots, m\}$. A tree defined over $Sk \cup \Sigma$ is called *skeleton*. Let $t \in V^T$. The *skeletal* (or *structural*) *description* of t, denoted $s(t)$, is a skeleton such that

$$s(t)(x) = \begin{cases} t(x) & \text{if } x \text{ is a terminal node,} \\ \sigma & \text{otherwise.} \end{cases}$$

Let T be a set of trees. The *corresponding skeletal set*, denoted $S(T)$, is $\{s(t) \mid t \in T\}$.

For a context-free grammar G, $S(D(G))$ corresponds to the structural description of G. Two context-free grammars G_1 and G_2 are said to be *structurally equivalent* if $S(D(G_1)) = S(D(G_2))$. Note that if G_1 and G_2 are structurally equivalent, they are equivalent, too. Given a context-free grammar G, we can get the skeletal alphabet which $S(D(G))$ is defined over. Let r be the set of the lengths of the right-hand sides of all the productions in G. Then the skeletal alphabet Sk associated with G consists of $\{\sigma\}$ with $r_{Sk} = \{\sigma\} \times r$.

Next we show two important propositions which connect a context-free grammar with a tree automaton.

Proposition 1 *Let $G = (N, \Sigma, P, S)$ be a context-free grammar. The corresponding nondeterministic tree automaton $A(G) = (Q, Sk \cup \Sigma, \delta, F)$ is defined as follows.*

$$
\begin{array}{ll}
Q = N, & \\
F = \{S\}, & \\
\delta_k(\sigma, B_1, \ldots, B_k) = A & \text{for each production of the form } A \rightarrow B_1 \cdots B_k, \\
\delta_0(a) = a & \text{for } a \in \Sigma.
\end{array}
$$

Then $T(A(G)) = S(D(G))$. That is, the set of trees accepted by $A(G)$ is equal to the structural description of G.

Proposition 2 *Let $A = (Q, Sk \cup \Sigma, \delta, F)$ be a tree automaton. The corresponding wide-sense context-free grammar $G(A) = (N, \Sigma, P, S)$ is defined as follows.*

$$
\begin{array}{l}
N = Q, \\
S = F, \\
P = \{\delta_k(\sigma, x_1, \ldots, x_k) \rightarrow x_1 \cdots x_k \mid \sigma \in Sk_k \text{ and } x_1, \ldots, x_n \in Q \cup \Sigma\}.
\end{array}
$$

Then $S(D(G(A))) = T(A)$. That is, the structural description of $G(A)$ is equal to the set of trees accepted by A.

3 The learning algorithm

Suppose G is the unknown grammar to be learned (up to structural equivalence).

A *structural membership query* proposes a skeleton s and asks whether it is in $S(D(G))$. The answer is either *yes* or *no*. A *structural equivalence query* proposes a grammar G' and asks whether $S(D(G)) = S(D(G'))$. The answer is *yes* or *no*. If it is *no*, then *counter-example* is also provided, that is, a skeleton s in the symmetric difference of $S(D(G))$ and $S(D(G'))$.

Now we extend Angluin's learning algorithm [1] to the one for tree automata over skeletons. Let A be a finite set of skeletons and B be a finite subset of $(Sk \cup \Sigma)_\$^T$. A is called *subtree-closed* if $s \in A$ implies all subtrees with depth at least 1 of s are elements of A. B is called *\$-prefix-closed with respect to A* if $e \in B - \{\$\}$ implies there exists an e' in B such that $e = e' \# \sigma(s_1, \ldots, s_{i-1}, \$, s_i, \ldots, s_{k-1})$ for some $s_1, \ldots, s_{k-1} \in A \cup \Sigma$. Firstly the *observation table* (S, E, T) is extended so that S is a nonempty finite subtree-closed set of skeletons with depth at least 1, $X(S) = \{\sigma(u_1, \ldots, u_k) \mid \sigma \in Sk_k,\ u_1, \ldots, u_k \in S \cup \Sigma \text{ and } \sigma(u_1, \ldots, u_k) \notin S \text{ for } k \geq 1\}$, E is a nonempty finite subset of $(Sk \cup \Sigma)_\$^T$ which is \$-prefix-closed with respect to S, and the finite function T is a mapping $(E \# (S \cup X(S)))$ to $\{0, 1\}$. The interpretation of this is that $T(s)$ is 1 iff s is a member of the structural description of the unknown grammar G. An observation table can be visualized as a two-dimensional matrix with rows labelled by elements of $(S \cup X(S))$, columns labelled by elements of E, and the entry for row s and column e equal to $T(e\#s)$. The learning algorithm uses the observation table to build a tree automaton. If s is an element of $(S \cup X(S))$, $row(s)$ denotes the finite function f from E to $\{0,1\}$ defined by $f(e) = T(e\#s)$. An observation table is called *closed* if every $row(x)$ of $x \in X(S)$ is identical to some $row(s)$ of $s \in S$. An observation table is called *consistent* if whenever s_1 and s_2 are in S such that $row(s_1) = row(s_2)$, $row(\sigma(u_1, \ldots, u_{i-1}, s_1, u_i, \ldots, u_{k-1})) = row(\sigma(u_1, \ldots, u_{i-1}, s_2, u_i, \ldots, u_{k-1}))$ for all $u_1, \ldots, u_{k-1} \in S \cup \Sigma$ and $1 \leq i \leq k$.

Let (S, E, T) be a closed, consistent observation table. The *corresponding tree automaton* $A(S, E, T)$ over $Sk \cup \Sigma$ constructed from (S, E, T) is defined with state set Q, final states F, and state transition function δ as follows.

$$Q = \{row(s) \mid s \in S\},$$

$$F = \{row(s) \mid s \in S \text{ and } T(s) = 1\},$$

$$\delta_k(\sigma, row(s_1), \ldots, row(s_k)) = row(\sigma(s_1, \ldots, s_k)) \quad \text{for } s_1, \ldots, s_k \in S \cup \Sigma,$$

$$\delta_0(a) = a \quad \text{for } a \in \Sigma,$$

where the function row is augmented to be $row(a) = a$ for $a \in \Sigma$. We can see that this is a well-defined deterministic tree automaton.

The Learning Algorithm LA

```
S:={$\sigma(a_1, \ldots, a_k) \mid \sigma \in Sk_k$ and $a_1, \ldots, a_k \in \Sigma$ for $k \geq 1$}; E:={$}; 
construct the initial observation table $(S, E, T)$ using structural membership queries;
Repeat
   While $(S, E, T)$ is not closed or not consistent;
      If $(S, E, T)$ is not consistent then
         find $s_1$ and $s_2$ in $S$, $e \in E$, $u_1, \ldots, u_{k-1} \in S \cup \Sigma$, and $i$ such that
            $row(s_1)$ is equal to $row(s_2)$ and
            $T(e\#\sigma(u_1, \ldots, u_{i-1}, s_1, u_i, \ldots, u_{k-1})) \neq T(e\#\sigma(u_1, \ldots, u_{i-1}, s_2, u_i, \ldots, u_{k-1}))$;
         add $e\#\sigma(u_1, \ldots, u_{i-1}, \$, u_i, \ldots, u_{k-1})$ to $E$;
         extend $(S, E, T)$ to $E\#(S \cup X(S))$ using structural membership queries;
      If $(S, E, T)$ is not closed then
         find $s_1 \in X(S)$ such that $row(s_1)$ is different from $row(s)$ for all $s \in S$;
         add $s_1$ to $S$;
         extend $(S, E, T)$ to $E\#(S \cup X(S))$ using structural membership queries;
   Once $(S, E, T)$ is closed and consistent, let G:=$G(A(S, E, T))$;
   Make a structural equivalence query proposing G;
   If the reply is no with a counter-example $t$ then
      add $t$ and all its subtrees with depth at least 1 to $S$;
      extend $(S, E, T)$ to $E\#(S \cup X(S))$ using structural membership queries;
Until the reply is yes to the conjecture G;
Halt and output G.
```

Theorem 3 *There is an algorithm that learns a grammar structurally equivalent to any context-free grammar G using structural equivalence and structural membership queries that runs in time polynomial in the number of states of the minimum tree automaton for the structural description of G and the maximum size of any counter-example.*

Proof. (Sketch) By Proposition 1, $S(D(G))$ can be accepted by some tree automaton. The observation table constructed during the running of LA always become closed, consistent one such that S is subtree-closed and E is \$-prefix-closed with respect to S. The corresponding tree automaton $A(S, E, T)$ constructed from it is consistent with T. Then LA eventually terminates and constructs a tree automaton isomorphic to the minimum tree automaton for $S(D(G))$. By Proposition 2, LA outputs a wide-sense context-free grammar structurally equivalent to G.

Let n be the number of states of the minimum tree automaton for $S(D(G))$, m be the maximum size of any counter-example, and d be the maximum length of the right-hand sides of the productions in G. The observation table is discovered to be not consistent or not closed at most $n - 1$ times. A counter-example requires the addition of at most m subtrees to S, and this can happen at most $n - 1$ times. Thus the maximum cardinality of $E\#(S \cup X(S))$ is at most $O(m^d n^{d+1})$. Hence, the total running time of LA can be bounded by a polynomial function of m and n. Q.E.D.

Since the structural information can be obtained from sentences of a parenthesis grammar, parenthesis languages can be learned efficiently from a *minimally adequate teacher.*

Corollary 4 *There is an algorithm that learns a grammar structurally equivalent to any parenthesis grammar G using equivalence and membership queries that runs in time polynomial in the number of states of the minimum tree automaton for the structural description of G and the maximum length of any counter-example.*

References

[1] D. Angluin. Learning regular sets from queries and counter-examples. *Information and Computation*, 75:87–106, 1987.

[2] D. Angluin. *Learning k-bounded context-free grammars.* YALEU/DCS/RR-557, 1987.

[3] P. Berman and R. Roos. Learning one-counter languages in polynomial time. In *Proceedings of IEEE FOCS '87*, pages 61–67, 1987.

[4] W. S. Brainerd. The minimalization of tree automata. *Information and Control*, 13:484–491, 1968.

[5] S. Crespi-Reghizzi. An effective model for grammar inference. In *Information Processing 71*, pages 524–529, Elsevier North-Holland, 1972.

[6] L. S. Levy and A. K. Joshi. Skeletal structural descriptions. *Information and Control*, 39:192–211, 1978.