ON INFERRING ZERO-REVERSIBLE LANGUAGES

ERKKI MÄKINEN

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF TAMPERE

REPORT A-1998-7

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER SCIENCE
SERIES OF PUBLICATIONS A
A-1998-7, AUGUST, REVISED OCTOBER 1998

ON INFERRING ZERO-REVERSIBLE LANGUAGES

ERKKI MÄKINEN

University of Tampere
Department of Computer Science
P.O.Box 607
FIN-33101 Tampere, Finland

On inferring zero-reversible languages

Erkki Mäkinen [1]

Department of Computer Science, University of Tampere, P.O. Box 607, FIN-33101 Tampere, Finland

Abstract

We use a language-theoretic result for zero-reversible languages to show that there exists a linear time inference method for this class of languages using positive data only.

Keywords: regular language, grammatical inference, identification in the limit.

1 Introduction

Regular languages cannot be inferred from positive data only [3]. This negative result has initiated a search for subclasses of regular languages having the desirable inference property. Several subclasses of regular languages allow inference algorithms based on merging nonterminals (or states in finite automata); such algorithms are surveyed in [5]. In this paper we consider zero-reversible languages, a well-known subclass of regular languages inferable from positive data only by using a merging algorithm.

We assume a familiarity with the basics of formal language theory and grammatical inference as given e.g. in [4] and [2], respectively. As inference criterion we use "identification in the limit" [3,2].

If not otherwise stated we follow the notations and definitions of [4]. The empty word is denoted by λ, the reverse of a string $w = w_1 w_2 \dots w_n$ by w^R ($= w_n w_{n-1} \dots w_1$), and the left-quotient of L and w by $T_L(w) = \{v \mid wv \in L\}$.

We consider here regular languages and grammars only. We also suppose that grammars are reduced [4], i.e. each terminal and nonterminal appears at least in one derivation from the start symbol to a terminal word. A production of

[1] E-mail: em@cs.uta.fi. Work supported by the Academy of Finland (Project 35025).

the form $A \to b$, where b is a terminal, is said to be *terminating*. A *continuing* production has the form $A \to bB$, where b is a terminal and B is a nonterminal. Other forms of productions are not allowed (except $S \to \lambda$, where S is the start symbol). A production with a nontemninal A on the left hand side is said to be an *A-production*.

2 Zero-reversible languages

Recall that a finite automaton A is zero-reversible if the following conditions hold [1]:

(i) A is deterministic.
(ii) A is reset-free, i.e. for no two distinct states q_1 and q_2 do there exist an input symbol b and a state q_3 such that $\delta(q_1, b) = q_3 = \delta(q_2, b)$, where δ is the transition function of A.
(iii) A has at most one final state.

A regular language L is zero-reversible if there exists a zero-reversible finite automaton accepting L. We denote the class of zero-reversible languages as $\mathcal{R}(0)$.

Angluin's inference algorithm [1] for $\mathcal{R}(0)$ starts with a prefix tree automaton and proceeds by merging states as long as the conditions (i) - (iii) are not satisfied. It follows that the time complexity for outputting the next conjecture is not linear, but it has a small nonlinear factor.

The following purely language-theoretic characterization is also useful.

Proposition 1 *[1] A regular language L is zero-reversible if and only if whenever u_1v and u_2v are in L, then $T_L(u_1) = T_L(u_2)$.*

A regular grammar $G = (V, \Sigma, P, S)$ is said to be *deterministic* if, for each nonterminal A, the right hand sides of A-productions start with unique terminals. Given a nonterminal A and a sequence w of terminal symbols in a deterministic grammar, $A \Rightarrow^+ wB$ is possible for at most one symbol B in $(V \setminus \Sigma) \cup \{\lambda\}$. The concept of *backward determinism* is related to a somewhat opposite situation.

G is said to be backward deterministic, if $A \Rightarrow^+ w$ and $B \Rightarrow^+ w$, where $w \in \Sigma^+$, always implies $A = B$. Hence, in a backward deterministic grammar each terminal string is possible to generate from at most one nonterminal. Notice that a backward deterministic grammar is not necessarily deterministic.

A language L is backward deterministic if there exists a backward deterministic grammar generating L. The class of backward deterministic languages is denoted by $\mathcal{B}$.

Notice that in backward deterministic grammars terminating productions have unique right hand sides. Namely, if we have $A \Rightarrow a$ and $B \Rightarrow a$, where $a \in \Sigma$, then we have $A = B$. Similarly, if we have

$$A = A_0 \Rightarrow a_1 A_1 \Rightarrow \ldots \Rightarrow a_1 \ldots a_{n-1} A_{n-1} \Rightarrow a_1 \ldots a_{n-1} a_n$$

and

$$B = B_0 \Rightarrow a_1 B_1 \Rightarrow \ldots \Rightarrow a_1 \ldots a_{n-1} B_{n-1} \Rightarrow a_1 \ldots a_{n-1} a_n,$$

then we have $A_i = B_i$, for $i = 0, \ldots, n-1$.

Theorem 2 *$\mathcal{R}(0) \subseteq \mathcal{B}$.*

Proof. Let L be zero-reversible. Suppose that $A \Rightarrow^+ w$ and $B \Rightarrow^+ w$ are possible in a regular grammar G generating L. Let $S \Rightarrow^+ u_1 A$ and $S \Rightarrow^+ u_2 B$ be derivations in G. We have $u_1 w$ and $u_2 w$ in L, and since L is zero-reversible, $T_L(u_1) = T_L(u_2)$. In other words, everything derivable from A is also derivable from B, and vice versa. Thus, we can replace all appearances of B in G by A without changing the language generated. This can be repeated with all pairs of nonterminals generating a common terminal string. Hence, there is a backward deterministic grammar for L.

The inclusion in Theorem 2 is proper since there are nondeterministic languages in $\mathcal{B}$. However, all deterministic languages in $\mathcal{B}$ are zero-reversible. Namely, if we have $u_1 v$ and $u_2 v$ in a deterministic language L in $\mathcal{B}$, in the corresponding backward deterministic grammar we have

$$S \Rightarrow^+ u_1 A \Rightarrow^+ u_1 v$$

and

$$S \Rightarrow^+ u_2 B \Rightarrow^+ u_2 v,$$

for some nonterminals A and B. Now $A \Rightarrow^+ v$ and $B \Rightarrow^+ v$ must imply $A = B$. And further, since L is deterministic, $T_L(u_1) = T_L(u_2)$. Hence, L is zero-reversible by Proposition 1. We have proved the following theorem.

Theorem 3 *If a deterministic language L is in $\mathcal{B}$, then L is zero-reversible.*

3 The new algorithm

Our new algorithm is based on Theorem 2. Contrary to Angluin's algorithm [1], we start with a suffix automaton (a trie containing the suffixes), since we consider terminal strings derivable from nonterminals. In a reduced regular grammar such strings are always suffixes.

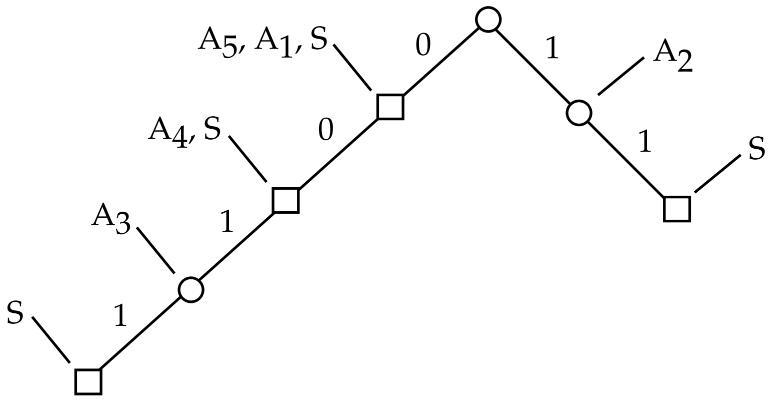

Fig. 1. The trie for the sample { 0, 00, 11, 1100 }.

As an example, consider a sample { 0, 00, 11, 1100 } (cf. [1, Example 29]). We have the following derivations:

$$S \Rightarrow 0$$

$$S \Rightarrow 0A_1 \Rightarrow 00$$
$$S \Rightarrow 1A_2 \Rightarrow 11$$
$$S \Rightarrow 1A_3 \Rightarrow 11A_4 \Rightarrow 110A_5 \Rightarrow 1100.$$

The corresponding trie is shown in Figure 1. Nodes with at least one ending word are drawn as squares. Each node (except the root) has a set of nonterminals associated with it.

The nonterminals associated with the same node are merged. The nonterminal with the smallest subscript is chosen to be the canonical element, i.e. the one used as the representative of the merged nonterminals. We assume that $S = A_0$.

The productions of the resulting grammar can be read by traversing the edges from the leaves to the root. We obtain the productions

$$S \to 1A_3, A_3 \to 1S, S \to 0S, S \to 0, S \to 1A_2, A_2 \to 1.$$

Notice that we do not merge nonterminals A_2 and A_3, although we have productions $S \to 1A_2$ and $S \to 1A_3$. The corresponding states in the resulting finite automaton are merged in Angluin's algorithm [1].

Figure 2 shows the trie after reading the next sample 101. The corresponding derivation is

$$S \Rightarrow 1A_6 \Rightarrow 10A_7 \Rightarrow 101.$$

We merge A_2 and A_7. Notice that merging A_2, A_3, and A_7 would imply a further merge (S and A_2), a finally, a grammar equivalent with the finite automaton shown in Figure 5(d) of [1].

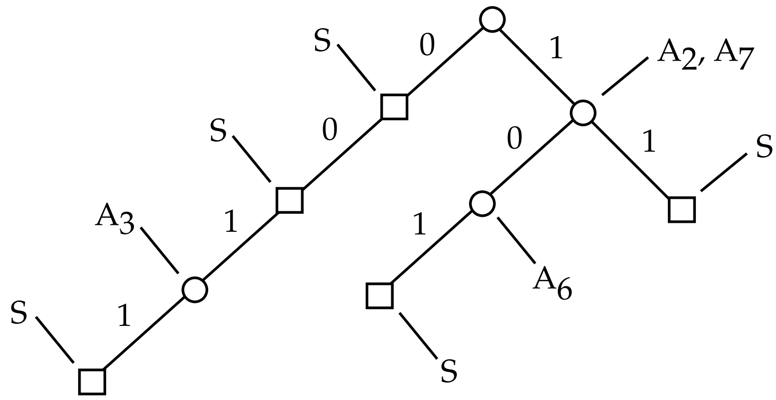

Fig. 2. The trie after reading the next input word 101.

We can formulate our algorithm as follows.

Algorithm 1 (BZR) ***Input:*** *A non-empty sample* $T = \{w_1, w_2, \ldots, w_n\}$. ***Output:*** *A backward deterministic grammar* G.

(i) *Insert the strings* $w_1^R, w_2^R, \ldots, w_n^R$ *to an initially empty string.*
(ii) *Associate the nonterminals from the derivations corresponding to the sample words to the nodes of the trie.*
(iii) *Merge the nonterminals appearing in each node. Choose the nonterminal with the smallest subscript as the the canoninal nonterminal (where* $S = A_0$*).*
(iv) *Read the resulting productions from the trie by traversing the edges from the leaves to the root. If a node is associated with* A_i*, its parent is associated with* A_j*, and the edge between the two nodes is labelled with* a*, the production obtained is* $A_i \to aA_j$*. If a child of the root is associated with* A *and the edge between the nodes is labelled with* a*, we obtain a terminating production* $A \to a$*.*
(v) *If* λ *is in* T*, then insert the production* $S \to \lambda$ *to* G*.*

If the input sample contains words u_1v and u_2v, the algorithm guarantees that in the output grammar G, v is derivable from a single nonterminal only. Hence, G is backward deterministic. Moreover, by Proposition 1 and by the fact that sample words are from a zero-reversible language, $L(G)$ is zero-reversible. It is clear that $L(G)$ is the smallest zero-reversible language containing the sample. Hence, $L(G)$ coincides with the language produced by Angluin's inference algorithm [1] for zero-reversible languages.

Notice that the grammar outputted by BZR is not necessarily deterministic. However, a corresponding deterministic grammar must exist since the language generated is in $\mathcal{R}(0)$. We have simply left some of the merges of Angluin's algorithm undone.

BZR runs in time $O(n)$, where n is the sum of the lengths of the input words.

Hence, we have the following theorem.

Theorem 4 *$\mathcal{R}(0)$ is inferable in linear time from positive data only.*

The space complexity of BZR is also linear. The trie contains less than n nodes, and it is sufficient to maintain one nonterminal (the canonical one) associated with a node.

Grammars obtained by BZR are bigger (have more productions) than those corresponding the finite automata produced by Angluin's algorithm. The bigger size of the resulting grammar seems to be the cost of dropping the nonlinear factor from the time complexity.

4 k-reversible languages

Proposition 1 has the following generalization in the case $k \geq 0$:

Proposition 5 *[1] A regular language L is k-reversible if and only if whenever u_1vw and u_2vw are in L and $lg(v) = k$, then $T_L(u_1v) = T_L(u_2v)$.*

It is possible to apply the approach of the previous section also to the case $k > 0$. However, the simplicity of the algorithm is lost in this case. Namely, we should maintain links between the derivations corresponding to the sample words and the nonterminals associated with the nodes in the trie, since merging is possible only when the condition of Proposition 5 is fulfilled.

References

[1] D. Angluin, Inference of reversible languages, *J. ACM* **29** (1982) 741–765.

[2] D. Angluin and C.H. Smith, Inductive inference: theory and methods, *ACM Comput. Surv.* **15** (1983) 237–269.

[3] E.M. Gold, Language identification in the limit, *Inform. Contr.* **10** (1967) 447–474.

[4] M.A. Harrison, *Introduction to Formal Language Theory* (Addison-Wesley, 1978).

[5] E. Mäkinen, Inferring regular languages by merging nonterminals, To appear in *Intern. J. Computer Math.*