数理解析研究所講究録
第 591 巻 1986 年 221-227

LA Symposium
in KYOTO 1986 winter

Modified One-way Alternating Pushdown Automata and Indexed Languages

池川将夫
MASAO IKEKAWA

Department of computer science
The University of Electro-Communications

Abstract. A partitioning automaton which is the modified version of the alternating automaton is defined. The machine can partition the input string into some blocks and check them universally. The classes of languages accepted by partitioning finite automata and partitioning pushdown automata are shown to be equivalent to the classes of CFL and Aho's indexed languages, respectivery.

1. Introduction

The concept of alternation was introduced by Chandra, Kozen and Stockmeyer [2] as generalization of nondeterminism. Several interesting applications of alternating Turing machines are known [3, 7, 8, 9]. There are some investigation about the effect of adding alternation to other automata [4, 5]. In this paper, we introduce the concept of *partitioning automata* which are modified alternating automta, and investigate the effect of adding *partition* to (one-way) finite automata and (one-way) pushdown automata.

In the case of alternating machines, it can make "existential branches" and "universal branches." Instead of "universal branches," partitioning machines can make "partitive branches." In partitive branches the machine guesses partition of remained input string into k blocks $(b_1, b_2, \cdots, b_k)$, and then reaches to each member of finite list of configuration $[\beta_1, \beta_2, \cdots, \beta_k]$ universally, where block b_i is assigned to configuration β_i as remained input string.

In Section 3 we show that

(1) the class of languages recognized by *partitioning finite automata* (pfa) is equivalent to the class of CFL, and

(2) the class of languages recognized by *partitioning pushdown automata* (ppda) is equivalent to the class of Aho's indexed languages.

The deterministic complexity hierarchy

$$\text{LOGSPACE} \subseteq \text{PTIME} \subseteq \text{PSPACE} \subseteq \text{EXPTIME} \subseteq \cdots$$

shifted by exactly one level when alternation was introduced. Then the formal language hierarchy

$$\text{REGULAR SET} \subsetneq \text{CFL} \subsetneq \text{INDEXED LANGUAGE} \cdots$$

also shifts exactly one level when partitioning automata are introduced.

In Section 4 we consider about the power of partitioning automata as complexity measures. And we show that

(3) for any polynomial-time bounded one-way alternating pushdown automaton M, there is an indexed language L and $L(M) \prec_{\text{poly}} L$.

2. Partitioning Finite Automata and Partitioning Pushdown Automata

A partitioning machine is similar to an alternating machine except that a subset of the states are designated as partitive states instead of universal states and the range of a transition function is defined by the set of finite lists instead of the power set.

Definition 2.1. A *partitioning finite automaton* (pfa) is a 6-tuple

$$M = (Q,\ U,\ \Sigma,\ \delta,\ q_0,\ F),$$

where

Q is a finite set of states,

$U \subseteq Q$ is a finite set of partitive states ($Q-U$ is a set of existential states),

Σ is a finite input alphabet,

$q_0 \in Q$ is the initial state,

$F \subseteq Q$ is the set of accepting states and

δ is the transition function where

$$\delta : Q \times \Sigma \rightarrow [Q]^*.$$

$[A]^*$ denotes the set of the finite lists, whose elements are elements of A, $[a_1, a_2, \cdots, a_k] (k \geqslant 0)$. In general definitions of nondeterministic or alternating machines, the range of a transition function is represented by the power set. In this paper, because the order of values of transition function is important, the range is represented by the set of lists. In following arguments, "$b \in B$" means also that b is an element of the list B.

Definition 2.2. A *configuration* of pfa M is a pair (q,w) with $q\in Q$ and $w\in\Sigma^*$. An *initial configuration of* M *on input* x is (q_0,x). An *accepting configuration* of M is (q,ε) where $q\in F$ and ε is the empty string.

Definition 2.3. A pfa M *accepts input* x if and only if an *accepting computation tree of* M *on input* x exists. An accepting computation tree of pfa M on input x is a finite rooted tree whose nodes are labeled with configurations of M and with the properties:

(I) A root of the tree is labeled with an initial configuration of M on input x.

(II) All the leaves of the tree are labeled with accepting configurations of M.

(III) For each nonleaf π labeled with (q,aw) where $q\in Q$, $a\in\Sigma\cup\{\varepsilon\}$ and $w\in\Sigma^*$,

(a) if $q\in Q-U$ and $[q_1,q_2,\cdots,q_k]=\delta(q,a)$ then π has exactly one child ρ with label (q_i,w) $1\leqslant i\leqslant k$.

(b) if $q\in U$ and $[q_1,q_2,\cdots,q_k]=\delta(q,a)$ then π has exactly k children $\rho_1,\rho_2,\cdots,\rho_k$ with labels $(q_1,w_1),(q_2,w_2),\cdots,(q_k,w_k)$ and $w_1w_2\cdots w_k=w$.

Definition 2.4. A *partitioning pushdown automaton* (ppda) is a 8-tuple

$$M = (Q,U,\Sigma,\Gamma,z,\delta,q_0,F),$$

where
Γ is the pushdown store alphabet,
$z\in\Gamma$ is the bottom symbol on the pushdown store and
δ is the transition function where

$$\delta: Q\times(\Sigma\cup\{\varepsilon\})\times\Gamma \to [Q\times\Gamma^*]^*.$$

Definition 2.5. A *configuration* of ppda M is a triple (q,w,θ) with $q\in Q$, $w\in\Sigma^*$ and $\theta\in\Gamma^*$. An *initial configuration of* M *on input* x is (q_0,x,z). An *accepting configuration of* M is (q,ε,θ) with $q\in F$ and $\theta\in\Gamma^*$.

Definition 2.6. A ppda M *accepts input* x if and only if an *accepting computation tree of* M *on input* x exists. An accepting computation tree of ppda M on input x is a finite rooted tree whose nodes are labeled with configurations of M and with properties:

(I)(II) similarly with *Definition* 2.3.

(III) For each nonleaf π labeled with $(q, aw, f\theta)$ where $q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $w \in \Sigma^*$, $f \in \Gamma$ and $\theta \in \Gamma^*$,

(a) if $q \in Q-U$ and $[(q_1, \xi_1), (q_2, \xi_2), \cdots, (q_k, \xi_k)] = \delta(q, a, f)$ then π has exactly one child ρ with label (q_i, w, ξ_i) $1 \leqslant i \leqslant k$.

(b) if $q \in U$ and $[(q_1, \xi_1), (q_2, \xi_2), \cdots, (q_k, \xi_k)] = \delta(q, a, f)$ then π has exactly k children $\rho_1, \rho_2, \cdots, \rho_k$ with labels $(q_1, w_1, \xi_1\theta)$, $(q_2, w_2, \xi_2\theta), \cdots, (q_k, w_k, \xi_k\theta)$ and $w_1 w_2 \cdots w_k = w$.

3. Relationships Between Partitioning Automata and Formal Languages

In this section we establish fundamental relationships between partitioning automata and formal languages. First we study about partitioning finite automata and context-free languages.

LEMMA 3.1. *Every context-free language L can be generated by a grammar for which every production is of the form $A \rightarrow ab\alpha$ or $A \rightarrow c$ where A is a variable, a and b are teminals, c is a terminal or ε, and α is a(possibly empty)string of variables.*

PROOF. Let $G = (N, T, P, S)$ be a Greibach normal form grammar generating the CFL L. Now consider a production in P, of the form $A \rightarrow aB_1B_2 \cdots B_k (k \geqslant 1)$. We can make new productions by replacing B_1 by all productions of the form $B_1 \rightarrow bC_1C_2 \cdots C_\ell$ $(\ell \geqslant 0)$. □

THEOREM 3.2. *A language L is accepted by a pfa iff L is a context-free language.*

PROOF. Let L be a CFL. There is a CFG $G = (N, T, P, S)$ which generates L. We assume that G satisfies Lemma 3.1. A pfa $M = (Q, U, T, \delta, S, F)$ is defined as follows:

$Q = N \cup \{\langle a \rangle \mid a \in T\} \cup \{S\} \cup U \cup F$.

$U = \{\langle a, X \rangle \mid a \in T,\ X \in N^+,\ |X| \leqslant \ell$ where ℓ is the largest length of strings which appear in righthand of all productions in P $\}$.

$q_F \in F$.

For each productions in P of the form

(1) $A \rightarrow abB_1B_2 \cdots B_k (k \geqslant 1)$ set

$\langle b, B_1B_2 \cdots B_k \rangle \in \delta(A, a)$ and

$[B_1, B_2, \cdots B_k] = \delta(\langle b, B_1B_2 \cdots B_k \rangle, b)$.

(2) $A \rightarrow ab$ set

$\langle b \rangle \in \delta(A, a)$ and

$q_F \in \delta(\langle b \rangle, b)$.

(3) $A \rightarrow a$ set

$q_F \in \delta(A, a)$.

(4) $A \rightarrow \varepsilon$ set

$A \in F$.

Obviously $L = L(M)$ holds. Now for the converse, let pfa $M = (Q, U, \Sigma, \delta, q_0, F)$.

Let $G = (Q, \Sigma, P, q_0)$ be a context-free grammar where P is defined as follows:

(1) For each $q \in Q-U$ and $a \in \Sigma$, if $q' \in \delta(q,a)$ then set

$$q \rightarrow aq' \in P.$$

(2) For each $q \in U$ and $a \in \Sigma$, if $[q_1, q_2, \cdots q_k] = \delta(q,a)$ $(k \geqslant 1)$ then set

$$q \rightarrow aq_1q_2 \cdots q_k \in P.$$

(3) For each $q \in F$, set

$$q \rightarrow \varepsilon \in P.$$

It is easy to show that $L(M) = L(G)$. □

Now we study about partitioning pushdown automata and indexed language. For the definition of indexed grammars and indexed languages see Aho[1].

THEOREM 3.3. *A language L is accepted by a ppda iff L is an indexed language.*

PROOF. Let L be an indexed language. There is an indexed grammar $G=(N,T,I,P,S)$ in reduced form, which generates L. Each index production in each index $f \in I$ is of the form $A \rightarrow B$, where $A, B \in N$. Each production in P is of the forms

(1) $A \rightarrow BC$,

(2) $A \rightarrow Bf$ or

(3) $A \rightarrow a$.

with $A, B, C \in N$, $f \in I$ and $a \in T \cup \{\varepsilon\}$. A ppda $M=(Q,U,T,\Gamma,z,\delta,S,F)$ is defined as follows:

$Q = N \cup \{S\} \cup U \cup F$.

$U = \{\langle X,Y \rangle \mid X,Y \in N\}$.

$F = \{q_F\}$.

$\Gamma = I \cup \{z\}$.

For each index production $A \rightarrow B$ in each index $f \in I$ set

$$(B,\varepsilon) \in \delta(A,\varepsilon,f).$$

For each production in P of the form

(1) $A \rightarrow BC$ set for all $f \in \Gamma$,

$$(\langle B,C \rangle, f) \in \delta(A,\varepsilon,f) \text{ and}$$
$$[(B,f),(C,f)] = \delta(\langle B,C \rangle,\varepsilon,f).$$

(2) $A \rightarrow Bf$ set for all $g \in \Gamma$,

$$(B,fg) \in \delta(A,\varepsilon,g).$$

(3) $A \rightarrow a$ set for all $f \in \Gamma$,

$$(q_F,f) \in \delta(A,a,f).$$

Obviously $L = L(M)$ holds. now for the converse, let $M=(Q,U,\Sigma,\Gamma,z,\delta,q_0,F)$ be a ppda. Let $G = (N,\Sigma,\Gamma,P,S)$ be an indexed grammar which is defined as follows:

$N = Q \cup \{S\} \cup \{\langle X,\xi \rangle \mid X \in Q,\ \xi \in \Gamma^*,\ |\xi| \leqslant \ell$ where ℓ is the largest length of strings which M can push in one move$\}$.

(1) For each $q \in Q-U$, $a \in \Sigma \cup \{\varepsilon\}$ and $f \in \Gamma$, if $(q',\xi) \in \delta(q,a,f)$ then set

$$q \rightarrow a\langle q',\xi \rangle \in f \text{ and}$$
$$\langle q',\xi \rangle \rightarrow q'\xi \in P.$$

(2) For each $q \in U$, $a \in \Sigma \cup \{\varepsilon\}$ and $f \in \Gamma$, if $[(q_1,\xi_1),(q_2,\xi_2),\cdots,(q_k,\xi_k)] = \delta(q,a,f)$ then set

$q \to a\langle q_1,\xi_1\rangle\langle q_2,\xi_2\rangle\cdots\langle q_k,\xi_k\rangle \in f$ and

for all $1 \leqslant i \leqslant k$,

$\langle q_i,\xi_i\rangle \to q_i\xi_i \in P$.

(3) Set $S \to q_0 z \in P$.

(4) For each $q \in F$, set

$q \to \varepsilon \in P$.

It is easy to show that $L(M) = L(G)$. □

4. The Power of Partitioning Automata

In this section we investigate the power of partitioning automata as complexity measures.

The class of languages accepted by pfa's (ppda's) is denoted by PFA (PPDA). The class of languages accepted by one-way alternating finite (pushdown) automata is denoted by ALT-FA (ALT-PDA). Aho [1] has shown that the class of indexed languages is a proper subset of the class of context-sensitive languages. Chandra, Kozen, and Stockmeyer [2] has shown that ALT-FA is equivalent to the class of regular sets, and ASPACE(n) $\subseteq$ ALT-PDA. Thus by Theorem 3.2 and Theorem 3.3:

Corollary 4.1. ALT-FA $\subsetneq$ PFA.

Corollary 4.2. PPDA $\subsetneq$ ALT-PDA.

We will consider time bounded one-way alternating pushdown automata. We denote the class of languages accepted by real(polynomial)-time bounded one-way alternating pushdown automata by real-ALT-PDA (poly-ALT-PDA).

Theorem 4.3. *For each* $L \in$ real-ALT-PDA, *there is* $L' \in$ PPDA *and* $L \prec_{\text{poly}} L'$.

Proof. Let M be a real-time bounded one-way pushdown automata which accepts L. Let k be the maximum number of branches which M can make in one move in univaersal states. The transducer f is defined as follows:

$f : \Sigma^* \to (\Sigma \cup \{[,]\})^*$,

$f(\varepsilon) = \varepsilon$ and

$f(aw) = a[f(w)][f(w)]\cdots[f(w)]$ (k times).

It is easy to show that there is a ppda M' and x is accepted by M iff $f(x)$ is accepted by M'. □

Corollary 4.4. *For each* $L \in$ poly-ALT-PDA, *there is* $L' \in$ PPDA *and* $L \prec_{\text{poly}} L'$.

Proof. Every $L \in$ poly-ALT-PDA is polynomial-time reducible to $L' \in$ real-ALT-PDA. □

Acknowledgments

Thanks to Associate Professer Takumi Kasai for his most helpful suggestions and discussions.

REFERENCES

1. AHO, A. V. Indexed grammars-an extension of context-free grammars. *J. ACM* 15, 4 (1968), 647-671.
2. CHANDRA, A. K., KOZEN, D. C., AND STOCMEYER, L. J. Alternation. *J. ACM* 28, 1 (1981), 114-133.
3. KANNAN, R. Alternation and the power of nondeterminism. Proc. 15th ACM Symp. on Theory of Computing, (1983), 344-346.
4. LADNER, R. E., LIPTON, R. J. AND STOCKMEYER, L. J. Alternating pushdown and stack automata. *SIAM J. Comput.* 13, 1 (1984), 135-155.
5. LADNER, R. J., STOCKMEYER, L. J. AND LIPTON, R. J. Alternation bounded auxiliary pushdown automata. *Inform. Contr.* 62 (1984), 93-108.
6. PARCHMAN, R., DUSKE, J., AND SPECHT, J. On deterministic indexed languages. *Inform. Contr.* 45 (1980), 48-67.
7. PAUL, W. J., PRAUSS, E. J., AND REISCHUK, R. On alternation. *Acta Informat.* 14 (1980), 243-255.
8. PAUL, W., AND REISCHUK, R. On alternation II. *Acta Informat.* 14 (1980), 391-403.
9. RUZZO, W. L. Tree-size bounded alternation. *J. Comput. Syst. Sci.* 21 (1980), 218-235.