Two-Way Alternating Counter Automata with Only Universal States

Tsunehiro YOSHINAGA[*1] and Katsushi INOUE[*2]

Abstract

A counter automaton is a pushdown automaton with only one pushdown symbol. A two-way alternating counter automaton (2aca) is a generalization of a two-way nondeterministic counter automaton (2nca). The state set of 2aca is partitioned into universal and existential states. In this paper, from a theoretical interest, we investigate a relationship between the accepting powers of 2aca's with only universal states (2uca's) and two-way deterministic counter automata (2dca's), and show that 2uca's are more powerful than 2dca's. Since 2nca's are 2aca's with only existential states, our result can be regarded as the pair to Chrobak's one in Refs. 2), 3) that 2nca's are better than 2dca's.

Key Words: alternating counter automata, two-way counter automata, alternation, universal states, computational complexity

1. Introduction and Preliminaries

A *counter automaton* is a pushdown automaton with only one pushdown symbol Z. That is, its pushdown tape is of the form Z^i.

A *two-way alternating counter automaton* (2aca) M is the generalization of a two-way nondeterministic counter automaton in the same sense as in Ref. 1). That is, the state set of M is divided into two disjoint sets, the set of *universal* states and the set of *existential* states. Intuitively, in a universal state M splits into some submachines which act in parallel, and in an existential state M nondeterministically chooses one of possible subsequent actions.

We assume that M has the left endmarker "¢" and the right endmarker "\$" on the input tape, reads the input tape in two directions (that is, right or left). We also assume that in one step M can increment or decrement the contents (that is, the length) of the counter by at most one.

We denote by 2uca (2nca) a 2aca with only universal states (with only existential states, i.e., a two-way nondeterministic counter automaton), and denote a two-way deterministic counter automaton by 2dca.

Let denote by 2dsfa(2) a two-way deterministic simple 2-head finite automaton. That is, the first head (called the *seeing head*) can sense input symbols while the second head (called the *blind head*) can only detect the endmarkers ¢ and \$.

2ACA, 2UCA, 2NCA, 2DCA, and 2DSFA(2) are the classes of languages accepted by the corresponding automata.

Chrobak solved the open problem posed in Ref. 4) and showed in Refs. 2), 3) that

"2DCA $\subsetneq$ 2NCA."

In this paper, from a theoretical interest, we investigate the relationship between 2DCA and 2UCA, and show that

"2DCA $\subsetneq$ 2UCA."

Since 2uca's and 2nca's are 2aca's with only universal and existential states, respectively, ours can be regarded as the pair to the result of Chrobak above.

2. Results

The results obtained in this paper are based on

[*1] Department of Computer Science and Electronics Engineering

[*2] Yamaguchi University

those in Refs. 2), 3). Let

$$E = \{x_1\#x_2\#\ldots\#x_k \mid k \geq 1 \;\&\; \exists l(l \geq 0)[\forall i(1 \leq i \leq k)[x_i \in \{0,1\}^l]]\}.$$

Let Σ be a finite alphabet and h: $\mathrm{E} \to \Sigma^+$ be a function such that for every $w_1, w_2 \in E$,
(a) if $w_1 \neq w_2$ then $h(w_1) \neq h(w_2)$, and
(b) if $|w_1| = |w_2|$ then $|h(w_1)| = |h(w_2)|$,
where for any string v, $|v|$ denotes the length of v. Now, the following languages are defined.

$$L^h_n = \{x_0\#h(x_1\#x_2\#\ldots\#x_k) \mid x_1\#x_2\#\ldots\#x_k \in E \;\&\; \exists j(1 \leq j \leq k)[x_j = x_0]\}, \text{ and}$$

$$L^h_n = \{x_0\#h(x_1\#x_2\#\ldots\#x_k) \mid x_1\#x_2\#\ldots\#x_k \in E \;\&\; \forall j(1 \leq j \leq k)[x_j \neq x_0]\}.$$

Then, Chrobak modified the proof in Ref. 5) and proved in Refs. 2), 3) that "$L^h_n \notin$ 2DSFA(2)."

In correspondence to this result, we can show:

Theorem 1. $L^h_u \notin$ 2DSFA(2).

Especially, to obtain the result that "$L^h_n \notin$ 2NCA", as a function h, Chrobak defined in Refs. 2), 3)

$$h(x_1\#x_2\#\ldots\#x_k) = \mathrm{s}(x_1)\#\mathrm{s}(x_2)\#\ldots\#\mathrm{s}(x_k)$$

for any $x_1\#x_2\#\ldots\#x_k \in E$, where

$$s(a_1a_2\ldots a_{l-1}a_l) = a_1 2^{l+1} a_2 2^{l+2} \ldots a_{l-1} 2^{2l-1} a_l,$$

where $a_i \in \{0, 1\}$ for each $1 \leq i \leq l$.

We can gain, using the same function h defined as above, our corresponding result:

Theorem 2. $L^h_u \in$ 2UCA.

3. The proof of Theorem 1

This proof is almost the same as that of "2DSFA(2) $\in L^h_n$" in Ref. 2). But we will give a sketch of it below, because the present paper is made to be self-contained and we believe that this proof is instructive. For the convenience of the reader, the notation is taken from Refs 2), 3), 5).

We take now arbitrary 2dsfa(2) M and show that M cannot accept L^h_u. We construct two languages $L_1 \subseteq \{0, 1\}^*$ and $L_2 \subseteq \#\Sigma^*$ such that

(A) for every different strings w, $\ddot{w} \in L_2$, there exists $y \in L_1$ such that exactly one of yw, $y\ddot{w}$ belongs to L^h_u, and

(B) there exist different strings $\hat{w}$, $\acute{w} \in L_2$ such that for every $y \in L_1$, $y\hat{w} \in L(M)$ iff $y\acute{w} \in L(M)$,

where $L(M)$ denotes the set of input strings accepted by an automaton M. From (A) and (B), we easily derive that $L^h_u \neq L(M)$.

Let Q be the set of states of M and w be an input of length n. A *configuration of M on w* is a triple (q, i, j), where $q \in Q$, $0 \leq i \leq n+1$ and $0 \leq j \leq n+1$ are the positions of the seeing head and the blind head, respectively. We assume that the computation starts and halts in configurations of the form $(q, 0, 0)$ for some $q \in Q$.

Let $x, y, z \in \{0, 1\}^*$ and consider the computation of M on xyz. Let C_0, C_r be configurations of M on xyz such that in C_0 the seeing head is on the first or on the last symbol of y, and in C_r the seeing head is on the symbol immediately to the right or left of y. The computation of M from C_0 to C_r is called an *internal computation on the triple* (x, y, z) if during it the seeing head is always on y and the blind head never reaches the endmarkers.

Lemma 1. If m is large enough then there are two different strings $u, v \in \{0, 1\}^m$ such that for every pair of strings x, z and every pair of configurations C_0, C_r of M, there is an internal computation of M from C_0 to C_r on the triple (x, u, z) if and only if there is an internal computation of M from C_0 to C_r on the triple (x, v, z).

Let m, u, v be fixed, satisfying Lemma 1. By n we denote an integer whose value will be specified later. Let

$$L_1 = \{u, v\}^n \text{ and}$$

$$L_2 = \{\#\,h(z_1\#z_2\#\ldots\#z_{2^{n-1}}) \mid \forall i(1 \leq i \leq 2^{n-1}) [z_i \in L_1] \;\&\; z_1 < z_2 < \ldots < z_{2^{n-1}}\},$$

where $z_1 < z_2$ means that the binary number represented by z_1 is smaller than the one represented by z_2. From the condition (a) of function h in the previous section and the definition of L_2, we obtain:

Lemma 2. For every two different strings $w, \ddot{w} \in L_2$, there is a string $y \in L_1$ such that exactly one of yw, $y\ddot{w}$

is in L^h_u.

Let $y = y_1y_2\ldots y_n \in L_1$, where each y_i is either u or v. Then, y_i is called the *i-th block of y*. Let e be the length of the strings in L_2 (by the condition (b), all strings in L_2 are of equal length). We set $d = mn + e$ and by N_d we denote the set $\{0, 1,\ldots, d+1\}$. Then $Q \times N_d \times N_d$ is the set of all possible configurations of M on the strings from L_1L_2.

Let σ be the set of all configurations C in $Q \times N_d \times N_d$ such that the position p of the seeing head in C satisfies $1 \leq p \leq mn$, and the blind head is in C on one of the endmarkers.

Let $y \in L_1$, $w \in L_2$ and $C_0, C_r \in \sigma$. If there is a computation of M on yw from C_0 to C_r then this computation is called a *computation segment* if every configuration between C_0 and C_r is not in σ.

Lemma 3. Let $y \in L_1$, $w \in L_2$ and C_0, $C_r \in \sigma$, and assume that the seeing head is on the i-th block of y in C_0 and on the j-th block of y in C_r. If there is a computation segment of M on yw from C_0 to C_r then, for all $y' \in L_1$ with the same i-th and j-th blocks as y, there is a computation segment of M from C_0 to C_r on $y'w$.

For every string $w \in L_2$, we define a partial function g_w: $\sigma \times \{u, v\}^2 \rightarrow \sigma$ as follows: Let $y \in L_1$ have the i-th block p and the j-th block q, where $p, q \in \{u, v\}$. Let C, $C' \in \sigma$ be such that the seeing head is on the i-th block of y in C and on the j-th block of y in C'. If there is a computation segment of M on yw from C to C' then $g_w(C, p, q) = C'$, otherwise $g_w(C, p, q)$ is undefined. By Lemma 3, g_w is well defined for any $w \in L_2$.

Lemma 4. If n is large enough then there are two strings $\hat{w}$, $\acute{w} \in L_2$ such that for every y $\in L_1$, y$\hat{w} \in L(M)$ iff y$\acute{w} \in L(M)$.

Proof. Because of the condition (a) there are $\begin{pmatrix} 2^n \\ 2^{n-1} \end{pmatrix}$ strings in L_2. The number of partial functions from $\sigma \times \{u, v\}^2$ into σ is $(2|Q|mn+1)^{8|Q|mn}$. So if n is large enough then there must be two strings $\hat{w}$, $\acute{w} \in L_2$ such that $g_{\hat{w}}=g_{\acute{w}}$. But then y$\hat{w} \in L(M)$ iff $y\acute{w} \in L(M)$ for every $y \in L_1$. □

Thus, from Lemmas 2 and 4, we obtain the conditions (A) and (B), respectively, what completes the proof of Theorem 1. For the proofs of Lemmas 1 and 3, see Ref. 5).

4. The proof of Theorem 2

We can construct a 2uca M which accepts L^h_u. Suppose that an input string

$w = x_0\#y_1\#y_2\#\ldots\#y_k$

(where $x_0 \in \{0, 1\}^*$ and $y_i \in \{0, 1, 2\}^*$ for each $1 \leq i \leq k$) is presented to M. (Input strings in the form different from the above can easily be rejected by M.)

M first checks if for each $1 \leq i \leq k$, $y_i = s(x_i)$ for some $x_i \in \{0, 1\}^+$, and if all x_i's $(0 \leq i \leq k)$ are of equal length. (Note that the function s has been already defined in the section 2.) It is clear that these checks can be done deterministically using one counter.

M then checks universally if $x_0 \neq x_i$ for each $1 \leq i \leq k$. To do so, M acts as follows. M stores $x_0(1)$ in the finite control and l $(=|x_0|)$ on the counter, where for each string v, $v(i)$ denotes the i-th symbol (from the left) of v.

Afterward, M moves right while making a universal branch at the first symbol of each x_j $(1 \leq j \leq k)$. M checks whether $x_0(1) = x_j(1)$ or not. If $x_0(1) \neq x_j(1)$ then M goes ahead to enter an accepting state, otherwise M executes the following step.

Suppose that M has already verified that $x_0(t) = x_j(t)$ for each $1 \leq t \leq i$-1, the head is on the symbol $x_j(i$-1$)$, and the counter is empty. Now, M checks if $x_0(i) \neq x_j(i)$. Let d be the distance between the first # and $x_j(i$-1$)$. M moves left increasing the counter one by one until it reaches the left endmarker ¢. Then, the counter stores d+1.

After that, M moves right on x_0 while increasing the counter and making a universal branch at each symbol $x_0(p)$ $(2 \leq p \leq l)$. M stores $x_0(p)$ in its finite control. At this time, it is easily observed that M stores $d + l + p$ on the counter.

At last, M moves to the first #, and again moves right decreasing now the counter until it becomes empty. Let c be the symbol scanned by the head when the counter becomes empty. The distance between the position of the head and $x_j(i$-1$)$ is $l + p$. Hence, the equivalence holds:

$$p = i \text{ iff } c \in \{0, 1\} \text{ iff } c = x_j(i).$$

If $(c = 2)$ or $(c \neq 2$ and $c \neq x_0(i))$ then M accepts the input w, otherwise (i.e., $c \neq 2$ and $c = x_0(i)$) M continues the computation above. After M sees that $x_0 = x_j$, it rejects w.

5. Conclusions

Since 2DCA $\subset$ 2DSFA(2), from Chrobak's results in Refs. 2), 3) and our theorems described in the section 2, it follows that

$$2DCA \subsetneq 2NCA \text{ and } 2DCA \subsetneq 2UCA,$$

respectively. We can consider the results above a dual and a complementary pair.

Unfortunately, the following problems are still open:

(1) Is 2NCA incomparable with 2UCA?,
(2) 2NCA $\subsetneq$ 2ACA?,
(3) 2UCA $\subsetneq$ 2ACA?,
(4) 2DCA $\subsetneq$ 2NCA $\cup$ 2UCA?, and
(5) 2NCA $\cap$ 2UCA $\subsetneq$ 2ACA?

References

1) Chandra, A. K., Kozen, D. C. and Stockmeyer, L. J.: Alternation, J. ACM, Vol. 28, PP. 114-133 (1981).
2) Chrobak, M.: Variations on the technique of Ďuriš and Galil, J. Comput. Syst. Sci., Vol. 30, PP. 77-85 (1985).
3) Chrobak, M.: Nondeterminism is essential for two-way counter machines, in MFCS'84, 11th Symp., LNSC 176, Springer-Verlarg, PP. 240-244 (1984).
4) Galil, Z.: Some open problems in the theory of computation as questions about two-way deterministic pushdown automata languages, Math. Systems Theory, Vol. 10, PP. 211-228 (1977).
5) Ďuriš, P. and Galil, Z.: Fooling a two way automata or one pushdown store is better than one counter for two way machines, Theor. Comput. Sci., Vol. 21, PP. 39-53 (1982).

(Received September 2, 2002)