A note on real-time one-way alternating multicounter machines

Katsushi Inoue, Akira Ito and Itsuo Takanami

Department of Computer Science and Systems Engineering, Faculty of Engineering, Yamaguchi University, Ube, 755 Japan

Abstract


This paper investigates several properties of one-way alternating multicounter machines which operate in real time, and shows that (1) for each \( k \geq 1 \), one-way alternating \( k \)-counter machines (1acm(\( k \))’s) which operate in real time are less powerful than 1acm(\( k+1 \))’s which operate in real time, (2) for each \( k \geq 2 \), 1acm(\( k \))’s which operate in real time are less powerful than 1acm(\( k \))’s which operate in linear time, and (3) for each \( k \geq 1 \), the class of sets accepted by 1acm(\( k \))’s which operate in real time is not closed under concatenation with regular sets, Kleene closure, reversal and length-preserving homomorphism.

1. Introduction

Alternating Turing machines were introduced in [2] as a generalization of non-deterministic Turing machines and as a mechanism to model parallel computation. In related papers [6–12], investigations of alternating machines have been continued. Many problems about alternating machines remain to be solved, however.

In [8], it is shown that for each \( k \geq 1 \), two-way alternating finite automata with \( k+1 \) heads are more powerful than two-way alternating finite automata with \( k \) heads, but it is still unknown [8] whether for each \( k \geq 1 \), one-way alternating finite automata with \( k+1 \) heads are more powerful than one-way alternating finite automata with \( k \) heads.
In [6, 7, 10], a relationship among alternating multihead finite automata, alternating
simple multihead finite automata, and alternating multicounter machines whose
counter contents are bounded by the length of the input word is investigated.

This paper investigates several properties of one-way alternating multicounter
machines which operate in real time. We show that (1) for each \( k \geq 1 \), one-way
alternating \( k \)-counter machines (1acm(\( k \))'s) which operate in real time are less
powerful than 1acm(\( k+1 \))'s which operate in real time, (2) for each \( k \geq 2 \), 1acm(\( k \))'s which
operate in real time are less powerful than 1acm(\( k \))'s which operate in linear time, and
(3) for each \( k \geq 1 \), the class of languages accepted by 1acm(\( k \))'s which operate in real
time is not closed under concatenation, Kleene closure, reversal and length-preserving
homomorphism. (1) above is the first hierarchical result, based on the number of
counters or heads, concerning the accepting powers of one-way machines with full
alternation.

2. Preliminaries

A one-way multicounter machine is a one-way multipushdown machine whose
pushdown stores operate as counters, i.e. have a single-letter alphabet. (See [1, 3, 5] for
formal definitions of one-way multicounter machines.)

An \( \text{one-way alternating multicounter machine} \ (1\text{acm}) \ M \) is the generalization of
a one-way nondeterministic multicounter machine in the same sense as in [2, 8, 9].
That is, the state set of \( M \) is divided into two disjoint sets, the set of \text{universal}
states and the set of \text{existential} states. Of course, \( M \) has a specified set of \text{accepting}
states. For each \( k \geq 1 \), we denote a one way alternating \( k \)-counter machine by lacm(\( k \)). We
assume that 1acm's have the right endmarker $ on the input tape, read the input
tape from left to right, and can enter an accepting state only when falling off the right
endmarker $. We also assume that in one step 1acm's can make an increment or
a decrement in the contents of each counter by at most one.

An \text{instantaneous description} (ID) of a 1acm(\( k \)) \( M \) is an element of
\[ \Sigma^* \times \mathbb{N} \times S_M, \]
where \( \Sigma \) (\( \Sigma \neq \Sigma \)) is the input alphabet of \( M \), \( \mathbb{N} \) denotes the set of all positive integers and
\( S_M = Q \times (\mathbb{N} \cup \{0\})^k \) (where \( Q \) is the set of states of the finite control of \( M \)). The first and
second components \( x \) and \( i \) of an ID \( I = (x, i, (q, (j_1, \ldots, j_k))) \) represent the input string
and the input head position, respectively.\(^3\) The third component \( (q, (j_1, \ldots, j_k)) \) of \( I \)
represents the state of the finite control and the contents of the \( k \) counters. An
element of \( S_M \) is called a \text{storage state} of \( M \). If \( q \) is the state associated with an ID \( I \),

\[^3\] \text{We note that } 1 \leq i \leq |x|+2, \text{ where for any string } w, |w| \text{ denotes the length of } w. "1", "|x|+1" \text{ and } "|x|+2" \text{ represent the positions of the leftmost symbol of } x, \text{ the right endmarker } $, \text{ and the immediate right to } $.
then $I$ is said to be a universal (existential, accepting) ID if $q$ is a universal (existential, accepting) state. The initial ID of $M$ on $x \in \Sigma^*$ is $I_M(x) = (x, 1, (q_0, (0, \ldots, 0)))$, where $q_0$ is the initial state of $M$. We write $I \vdash_M I'$ and say $I'$ is a successor of $I$ if an ID $I'$ follows from an ID $I$ in one step, according to the transition function of $M$. A computation path of $M$ on input $x$ is a sequence $I_0 \vdash_M I_1 \vdash_M \ldots \vdash_M I_n$ ($n \geq 0$), where $I_0 = I_M(x)$. A computation tree of $M$ is a finite, nonempty labeled tree with the following properties:

1. each node $x$ of the tree is labeled with an ID, $l(x)$,
2. if $\pi$ is an internal node (a nonleaf) of the tree, $l(\pi)$ is universal and \[ \{ l(\pi) \vdash_M I \} = \{ I_1, \ldots, I_r \}, \]
then $\pi$ has exactly $r$ children $\rho_1, \ldots, \rho_r$ such that $Q(\rho_i) = l_i$,
3. if $\pi$ is an internal node of the tree and $l(\pi)$ is existential, then $\pi$ has exactly one child $\rho$ such that $l(\rho) = l_M^r(\rho)$.

A computation tree of $M$ on input $x$ is a computation tree of $M$ whose root is labeled with $I_M(x)$. An accepting computation tree of $M$ on $x$ is a computation tree of $M$ on $x$ whose leaves are all labeled with accepting IDs. We say that $M$ accepts $x$ if there is an accepting computation tree of $M$ on $x$. Define $T(M) = \{ x \in \Sigma^* | M$ accepts $x \}$.

A 1acm($k$) $M$ operates in time $T(n)$ if for each input $x$ accepted by $M$, there is an accepting computation tree of $M$ on $x$ such that the length of each computation path of the tree is at most $T(|x|)$. $M$ operates in real time (linear time) if $T(n) = n + 1$ ($T(n) = cn$ for some positive constant $c$). Define

\[ 1\text{ACM}(k, \text{real}) = \{ T | T = T(M) \text{ for some 1acm($k$) } M \text{ which operates in real time} \}, \]

\[ 1\text{ACM}(k, \text{linear}) = \{ T | T = T(M) \text{ for some 1acm($k$) } M \text{ which operates in linear time} \}. \]

Similarly, we let $1\text{DCM}(k, \text{real})$ ($1\text{DCM}(k, \text{linear})$) denote the class of languages accepted by one-way deterministic $k$-counter machines which operate in real (linear) time, and let $1\text{NCM}(k, \text{real})$ ($1\text{NCM}(k, \text{linear})$) denote the class of languages accepted by one-way nondeterministic $k$-counter machines which operate in real (linear) time.

3. $k + 1$ counters are better than $k$ in real time

It is well known [1, 3] that for each $k \geq 1$, $1\text{DCM}(k, \text{real}) \subsetneq 1\text{DCM}(k + 1, \text{real})$ and $1\text{NCM}(k, \text{real}) \subsetneq 1\text{NCM}(k + 1, \text{real})$. The main purpose of this section is to show that for each $k \geq 1$, $1\text{ACM}(k, \text{real}) \subsetneq 1\text{ACM}(k + 1, \text{real})$. We first note that the following theorem holds.

**Theorem 3.1.** There exists a language accepted by a 1ACM(1, real), but not accepted by any one-way nondeterministic multicounter machine which operates in time $T(n) = n^r$ for any constant $r$. 


Proof. Let \( L = \{ wcw \mid w \in \{0, 1\}^+ \} \). It is easy to show that \( L \in 1 \text{ ACM}(1, \text{ real}) \). On the other hand, by using a simple counting argument, we can show that \( L \) is not accepted by any one-way nondeterministic multicounter machine which operates in time \( T(n) = n^r \) for any constant \( r \). □

Corollary 3.2. For each \( k \geq 1 \), \( 1 \text{ NCM}(k, \text{ real}) \subseteq 1 \text{ ACM}(k, \text{ real}) \) and \( 1 \text{ NCM}(k, \text{ linear}) \nsubseteq 1 \text{ ACM}(k, \text{ linear}) \).

To prove the main result of this section, we first give some necessary definitions. Let \( M \) be a \( 1 \text{ ACM}(k), k \geq 1 \), and \( \Sigma \) be the input alphabet of \( M \). For each storage state \( (q, (j_1, \ldots, j_k)) \) of \( M \) and for each \( w \in \Sigma^+ \), let a \((q, (j_1, \ldots, j_k))\)-computation tree of \( M \) on \( w \) be a computation tree of \( M \) whose root is labeled with the ID \((w, 1, (q, (j_1, \ldots, j_k)))\). (That is, a \((q, (j_1, \ldots, j_k))\)-computation tree of \( M \) on \( w \) is a computation tree which represents a computation of \( M \) on \( w \$ \) starting with the input head on the leftmost position of \( w \) and with the storage state \((q, (j_1, \ldots, j_k))\).) A \((q, (j_1, \ldots, j_k))\)-accepting computation tree of \( M \) on \( w \) is a \((q, (j_1, \ldots, j_k))\)-computation tree of \( M \) on \( w \) whose leaves are all labeled with accepting IDs.

The following lemma leads to our main theorem.

Lemma 3.3. For each \( k \geq 1 \), let \( L(k) = \{ \#^n w \# w_1 \# w_2 \ldots \# w_r \in \{0, 1, \#\}^+ \mid n \geq 1 \) and \( w \in \{0, 1\}^+ \) and \( |w| = n \) and \( r = (n + 1)^k \) and \( \forall i(1 \leq i \leq r) [w_i \in \{0, 1\}^+ \) and \( |w_i| = n ] \) and \( \exists j(1 \leq j \leq r) [w = w_j] \}. Then
1. \( L(k) \in 1 \text{ ACM}(k + 1, \text{ real}) \), and
2. \( L(k) \nsubseteq 1 \text{ ACM}(k, \text{ real}) \).

Proof. (1) \( L(k) \) is accepted by a \( 1 \text{ ACM}(k + 1, \text{ real}) \) \( M \) which acts as follows. Let \( H \) be the input head of \( M \) and \( C_1, C_2, \ldots, C_{k+1} \) be the counters of \( M \). For each \( n \geq 1 \) and for integers \( b_1, \ldots, b_k \) such that \( 0 \leq b_i \leq n \) (\( 1 \leq i \leq k \)), let \( f_n(b_k, b_{k-1}, \ldots, b_1) \) denote the integer represented by the \((n+1)\)-ary number \( b_k b_{k-1} \ldots b_1 \), i.e.

\[
f_n(b_k, b_{k-1}, \ldots, b_1) = b_k \times (n + 1)^{k-1} + b_{k-1} \times (n + 1)^{k-2} + \cdots + b_1 \times (n + 1)^0.
\]

Suppose that an input string

\[
\#^n w \# w_1 \# w_2 \ldots \# w_r S,
\]

(where \( n \geq 1 \), \( r \geq 1 \), \( w \in \{0, 1\}^+ \) and \( w_i \in \{0, 1\}^+ \) (\( 1 \leq i \leq r \)) is presented to \( M \). (Input strings in a form different from the one above can easily be rejected by \( M \).) \( M \) universally branches to check the following three points:

(i) whether \( n = |w| = |w_1| = \cdots = |w_r| \);
(ii) whether \( r = (n + 1)^k \);
(iii) whether \( w = w_j \) for some \( j \) (\( 1 \leq j \leq r \)).
(i) can be easily checked by using two counters. (ii) can be checked by using the following algorithm. (The algorithm below uses only $k$ counters. If we use $k+1$ counters, we can give a simpler algorithm.) For each $i$ ($1 \leq i \leq k$), we let $j_i$ denote the contents of counter $C_i$.

(a) While reading the initial segment $\#^n$ of the input, $M$ stores $n$'s in the first $k$ counters $C_1, \ldots, C_k$. That is, when $H$ reaches the leftmost symbol $w(1)$ of $w$, $j_1 = j_2 = \cdots = j_k = n$ and, thus, $f_k(j_k, \ldots, j_1) = f_k(n, \ldots, n) = (n+1)^k - 1$.

(b) Assuming that point (i) is successfully checked (i.e. $n = |w| = |w_1| = \cdots = |w_r|$), $M$ then checks that $r = (n+1)^k$ as follows. $M$ makes a decrement in $f_k(j_k, \ldots, j_1)$ by one each time $H$ meets the symbol $\#$. In order to do so, $M$ makes a decrement in $j_i$ (the contents of $C_i$) by one each time $H$ meets $\#$. In this case e.g. if $j_1 = 0$ when $H$ meets the $q$th $\#$ (from the left) which appears after $w$, then $M$ makes a decrement in $j_m$ (where $m$ is the smallest integer such that $j_m \neq 0$) by one instead of making a decrement in $j_1$ by one, and then $M$ sets $j_1 = \cdots = j_{m-1} = n$ by using the (assumed) length $n$ of $w_q$. $M$ enters an accepting state only if $H$ meets $\#$ with $j_1 = \cdots = j_k = 0$ (i.e. $f_k(j_k, \ldots, j_1) = 0$) and there exists exactly one string in $L(k)$ after this $\#$.

(iii) can be checked as follows. While reading the initial segment $\#^n$ of the input, $M$ nondeterministically guesses some $j$ ($1 \leq j \leq r$) by using existential states and setting $j_1, \ldots, j_k$ such that $f_k(j_k, \ldots, j_1) = j$. After this, $M$ checks that $w = w_j$. To do so, $M$ universally checks that for each $i$ ($1 \leq i \leq |w|$), $w(i) = w_j(i)$. That is, $M$ stores $i$ in $C_{k+1}$ when it picks up the symbol $w(i)$, and compares the symbol $w(i)$ with the symbol $w_j(i)$ by using $i$ and $j$ ($w_j$ can be identified by using a technique similar to (b) above), and enters an accepting state only if both symbols are identical. It is obvious that $T(M) = L(k)$. This completes the proof of (1).

(2) Suppose that there exists a 1acm$(k)$ $M$ which operates in real time and accepts $L(k)$. For each $n \geq 1$, let

$$V(n) = \{ \#^nw \# w_1 \# w_2 \cdots \# w_{g(n)} | \forall i(1 \leq i \leq g(n))[w_i \in \{0, 1\}^*$$

$$\& |w_i| = n] \& \exists j(1 \leq j \leq g(n))[w = w_j] \subseteq L(k),$$

where $g(n) = (n+1)^k$ and

$$W(n) = \{ \#_{\sim}w \# w_1 \# w_2 \cdots \# w_{g(n)} | \forall i(1 \leq i \leq g(n))[w_i \in \{0, 1\}^* \& |w_i| = n] \}.$$  

Note that for each $x = \#^nw \# w_1 \# w_2 \cdots \# w_{g(n)}$ in $V(n)$,

(i) $|x| = 2n + (n+1)^k + 1 = r(n)$, and

(ii) there exists an accepting computation tree of $M$ on $x$ which has the following properties:

(a) for each computation path $p$ from the root to a leaf, the length of $p$ is $|x| = r(n) + 1$ and $p$ represents a computation in which the input head moves one square to the right in each step and, thus,

2. For each string $x$ and each integer $i$ ($1 \leq i \leq |x|$), $x(i)$ denotes the $i$th symbol (from the left) of $x$.  


(b) for each node \( \pi \) labeled with an ID which \( M \) enters just after the input head has read the initial segment \( \#^n w \) of \( x \), the contents of each counter in \( l(\pi) \) is bounded by \( 2n \), since \( M \) operates in real time and we assume that \( M \) can enter an accepting state only when falling off the right endmarker \( \$ \).

For each storage state \((q, (j_1, \ldots, j_k))\) of \( M \) and for each \( y \) in \( W(n) \), let

\[
M_s(q, (j_1, \ldots, j_k)) = 1 \quad \text{if there exists a } (q, (j_1, \ldots, j_k))-\text{accepting computation tree of } M \text{ on } y \text{ such that for each computation path } p \text{ from the root to a leaf, the length of } p \text{ is } |yS| = r(n) + 1 - 2n \text{ and } p \text{ represents a computation in which the input head moves one square to the right in each step,}
\]

\[
= 0 \quad \text{otherwise.}
\]

For any two strings \( y, z \) in \( W(n) \), we say that \( y \) and \( z \) are \( M \)-equivalent if for each storage state \((q, (j_1, \ldots, j_k))\) of \( M \) with \( 0 \leq j_i \leq 2n \) \((1 \leq i \leq k)\), \( M_s(q, (j_1, \ldots, j_k)) = M_s(q, (j_1, \ldots, j_k)) \). Clearly, \( M \)-equivalence is an equivalence relation on strings in \( W(n) \), and there are at most

\[
E(n) = 2^{(2n+1)^k}
\]

\( M \)-equivalence classes, where \( s \) denotes the number of states of the finite control of \( M \).

We denote these \( M \)-equivalence classes by \( C_1, C_2, \ldots, C_{E(n)} \).

For each \( y = \# w_1 \# w_2 \ldots \# w_{g(n)} \) in \( W(n) \), let

\[
b(y) = \{ u \in \{0, 1\}^+ \mid \exists i \ (1 \leq i \leq g(n)) \ [u = w_i] \}.
\]

Furthermore, for each \( n \geq 1 \), let \( R(n) = \{ b(y) \mid \exists y \in W(n) \} \). Then,

\[
|R(n)| = \binom{2^n}{1} + \binom{2^n}{2} + \cdots + \binom{2^n}{g(n)}
\]

where for any set \( S \), \( |S| \) denotes the number of elements of \( S \).

We can easily see that \( \log E(n) = O(n^k) \) and \( \log |R(n)| = O(n^{k+1}) \). Thus, we have \( |R(n)| > E(n) \) for large \( n \). For such \( n \), there must be some \( Q, Q' \) \((Q \neq Q')\) in \( R(n) \) and some \( C_i \) \((1 \leq i \leq E(n))\) such that the following statement holds:

“\( \text{There are two words } y, z \in W(n) \text{ such that (i) } b(y) = Q \neq Q' = b(z) \text{ and (ii) } y, z \in C_i \text{ (i.e. } y \text{ and } z \text{ are } M\text{-equivalent).} \)"

Because of (i), we can, without loss of generality, assume that there is some word \( w \in \{0, 1\}^+ \) such that \( |w| = n \) and \( w \in b(y) - b(z) \). Clearly, it implies that \( y' = \#^n wy \in L(k) \) and \( z' = \#^n wz \notin L(k) \). But because of (ii), \( y' \) is accepted by \( M \) iff \( z' \) is accepted by \( M \), which is a contradiction. This completes the proof of (2).

From Lemma 3.3, we have the following theorem.

**Theorem 3.4.** For each \( k \geq 1 \), \( 1 \text{ ACM}(k, \text{ real}) \subsetneq 1 \text{ ACM}(k+1, \text{ real}) \).
4. Real time versus linear time

In [4], it is shown that for each \( k \geq 2 \), \( 1\text{DCM}(k, \text{real}) \subseteq 1\text{DCM}(k, \text{linear}) \) and \( 1\text{NCM}(k, \text{real}) \subseteq 1\text{NCM}(k, \text{linear}) \). This section shows that a similar fact holds for the alternating version. In fact, we can show a stronger result as follows.

**Theorem 4.1.** There exists a language in \( 1\text{DCM}(2, \text{linear}) \), but not in \( \bigcup_{1 \leq k < \infty} 1\text{ACM}(k, \text{real}) \).

**Proof.** Let \( L_s = \{ w \# 0^{m_1} \# 0^{m_2} \ldots \# 0^{m_r} \mid w \in \{0, 1\}^+ \) & \( r \geq 1 \) & \( \forall i (1 \leq i \leq r) \ [m_i \geq 1] \) & \( \exists j (1 \leq j \leq r) \ [m_j = N(w) + 1] \} \), where \( N(w) \) denotes the integer represented by \( w \) as a binary number (with the least significant bit in the rightmost position). The language \( L_s \) can be accepted by a \( 1\text{DCM}(2, \text{linear}) \) \( M_s \) which acts as follows. Suppose that an input string

\[
w \# 0^{m_1} \# 0^{m_2} \ldots \# 0^{m_r} S,
\]

(where \( r \geq 1 \), \( w \in \{0, 1\}^+ \) and \( m_i \geq 1 \) \((1 \leq i \leq r)\)) is presented to \( M_s \). (Input strings in a form different from the one above can easily be rejected by \( M_s \).) While reading the initial segment \( w \) of the input, \( M_s \) stores the integer \( N(w) + 1 \) in one counter. (It is an easy exercise to see that this action is possible in time \( O(\max \{m_i \mid i = 1, \ldots, r\}) \). \( M_s \) then checks by using two counters that \( m_j = N(w) + 1 \) for some \( j (1 \leq j \leq r) \), and accepts the input only if this check is successful. It is obvious that \( T(M_s) = L_s \).

We show below, by using the same technique as in the proof of Lemma 3.3(2), that \( L_s \notin \bigcup_{1 \leq k < \infty} 1\text{ACM}(k, \text{real}) \). Suppose that for some \( k \geq 1 \), there exists a \( 1\text{ACM}(k) \) \( M \) which operates in real time and accepts \( L_s \). For each \( n \geq 1 \), let

\[
V(n) = \{ w \# 0^{m_1} \# 0^{m_2} \ldots \# 0^{m_r} \mid |w| = n \) & \( w \in \{0, 1\}^+ \) & \( \forall i (1 \leq i \leq f(n)) \ [1 \leq m_i \leq 2^n] \) & \( \exists j (1 \leq j \leq f(n)) \ [m_j = N(w) + 1] \} \subseteq L_s,
\]

where \( f(n) = 2^n \) and

\[
W(n) = \{ \# 0^{m_1} \# 0^{m_2} \ldots \# 0^{m_r} \mid \forall i (1 \leq i \leq f(n)) \ [1 \leq m_i \leq 2^n] \}.
\]

Similarly, as in the proof of Lemma 3.3(2), we can divide \( W(n) \) into at most \( E(n) = 2^{2^n + 1} \) \( M \)-equivalence classes, where \( s \) denotes the number of states of the finite control of \( M \).

For each \( y = \# 0^{m_1} \# 0^{m_2} \ldots \# 0^{m_r} \) in \( W(n) \), let

\[
b(y) = \{ m \in N \mid \exists i (1 \leq i \leq f(n)) \ [m = m_i] \}.
\]

Furthermore, for each \( n \geq 1 \), let \( R(n) = \{ b(y) \mid y \in W(n) \} \). Then

\[
| R(n) | = \left( \binom{2^n}{1} + \binom{2^n}{2} + \cdots + \binom{2^n}{f(n)} \right) - 2^{2^n} - 1.
\]
We can easily see that \(|R(n)| > E(n)|\) for large \(n\). Now the proof that \(L_k \notin \bigcup_{1 \leq k < \infty} 1 \text{ACM}(k, \text{real})\) can be completed in the same way as in the proof of Lemma 3.3(2).

From Theorem 4.1, we have the following corollary.

**Corollary 4.2.** (1) For each \(k \geq 2\), \(1 \text{ ACM}(k, \text{ real}) \subset \subset 1 \text{ ACM}(k, \text{ linear})\) and (2) \(\bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{ real}) \subset \subset \bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{ linear})\).

**Remark 4.3.** In [4], it is shown that \(1 \text{ NCM}(1, \text{ real}) = 1 \text{ NCM}(1, \text{ linear})\). We conjecture that \(1 \text{ ACM}(1, \text{ real}) = 1 \text{ ACM}(1, \text{ linear})\), but we have no proof of this conjecture.

## 5. Closure properties

This section investigates several closure properties of one-way alternating multi-counter machines which operate in real time.

**Lemma 5.1.** For each \(k \geq 1\), let \(L'(k) = \{w \# w_1 \# w_2 \ldots \# w_r \mid w \in \{0, 1\}^+ \land r = (|w| + 1)^k \land \forall i (1 \leq i \leq r) \{w_i \in \{0, 1\}^+ \land |w_i| = |w_r| \land \exists j (1 \leq j \leq r) [w = w_j]\}\). Then \(L'(k) \notin 1 \text{ ACM}(k, \text{ real})\) for each \(k \geq 1\).

**Proof.** The proof is almost the same as that of Lemma 3.3(2).

**Lemma 5.2.** Let \(L_1 = \{w \# w_1 \# w_2 \ldots \# w_r \mid w \in \{0, 1\}^+ \land r \geq 1 \land \forall i (1 \leq i \leq r) \{w_i \in \{0, 1\}^+ \land |w_i| = |w| \land \exists j (1 \leq j \leq r) [w = w_j]\}\). Then \(L_1 \notin \bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{ real})\).

**Proof.** Suppose that \(L_1 \in 1 \text{ ACM}(k, \text{ real})\) for some \(k \geq 1\). For each \(s \geq 1\), let \(L''(s) = \{w \# w_1 \# w_2 \ldots \# w_r \mid w \in \{0, 1\}^+ \land r = (|w| + 1)^s \land \forall i (1 \leq i \leq r) \{w_i \in \{0, 1\}^+ \land |w_i| = |w| \land \exists j (1 \leq j \leq r) [w = w_j]\}\). By using a technique similar to that in the proof of Lemma 3.3(1), we can easily show that \(L''(k) \in 1 \text{ ACM}(k, \text{ real})\). It is easy to see that \(L''(k) \cap L_1 = L'(k)\), where \(L'(k)\) is the set described in Lemma 5.1. Further, it is obvious that \(1 \text{ ACM}(k, \text{ real})\) is closed under intersection. From these facts, it follows that \(L'(k) \notin 1 \text{ ACM}(k, \text{ real})\). This contradicts Lemma 5.1.

**Theorem 5.3.** \(1 \text{ ACM}(k, \text{ real}), k \geq 1\) and \(\bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{ real})\) are not closed under

(1) concatenation with regular sets,
(2) Kleene closure,
(3) reversal, and
(4) length-preserving homomorphism.

**Proof.** (1) Let \(L_2 = \{w \# w_1 \# w_2 \ldots \# w_r \mid w \in \{0, 1\}^+ \land r \geq 1 \land \forall i (1 \leq i \leq r) \{w_i \in \{0, 1\}^+ \land w = w_r\}\). We can easily show that \(L_2 \in 1 \text{ ACM}(1, \text{ real})\). Further, it is
easily seen that (i) \( L_3 = \{ \# w | w \in \{0, 1\}^+ \}^* \) is regular and (ii) \( L_2 L_3 = L_1 \), where \( L_1 \) is the set described in Lemma 5.2. From these facts and from Lemma 5.2, (1) follows.

(2) Let \( L_4 = \{ 2 \} L_2 \), \( L_5 = \{ 2 \} \{ w \# | w \in \{0, 1\}^+ \}^* \), and \( L_6 = (L_3 \cup L_4)^* \cap L_5 = \{ 2w \# w_1 \# w_2 \ldots \# w_r \in \{0, 1\}^+ \text{ and } r \geq 1 \text{ and } \forall i (1 \leq i \leq r) \{ w_i \in \{0, 1\}^+ \} \} \). By using the same idea as in the proof of Lemma 5.2, we can show that \( L_6 \notin \bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{real}) \). On the other hand, both \( L_3 \cup L_4 \) and \( L_5 \) are in \( 1 \text{ ACM}(1, \text{real}) \). From these facts and from the fact that \( 1 \text{ ACM}(k, \text{real}), k \geq 1 \) and \( \bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{real}) \) are closed under intersection, (2) follows.

(3) It is not so difficult to show that \( L_1^R (= \text{the reversal of } L_1) \) is in \( 1 \text{ ACM}(1, \text{real}) \). From this fact and Lemma 5.2, (3) follows.

(4) Let \( L_7 = \{ w_1 w_2 \ldots \# w_r \in \{0, 1, 2, \# \}^+ \text{ and } r \geq 1 \text{ and } \forall i (1 \leq i \leq r) \{ w_i \in \{0, 1\}^+ \} \} \). We can easily show that \( L_7 \in 1 \text{ ACM}(1, \text{real}) \). Further, \( h(L_7) = L_1 \), where \( h \) is a length-preserving homomorphism such that \( h(0) = 0, h(1) = 1, h(2) = h(\#) = \# \). From these facts and from Lemma 5.2, (4) follows.

6. Conclusion

In this paper, we investigated several properties of one-way alternating multicounter machines that operate in real time. We note that also for the one-way alternating multi-stack-counter automaton, which is an alternating version of the one-way multi-stack-counter automaton [1], we can get results similar to those in this paper. For example, we can show by using the same technique as in the proof of Lemma 3.3 that for each \( k \geq 1 \), one-way alternating automata with \( k \) stack-counters are less powerful than one-way alternating automata with \( k + 1 \) stack-counters.

We conclude this paper by stating some open problems.

(1) For each \( k \geq 1 \), \( 1 \text{ ACM}(k, \text{linear}) \subseteq 1 \text{ ACM}(k + 1, \text{linear})? \)

(2) Are \( 1 \text{ ACM}(k, \text{real}), k \geq 1 \) and \( \bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{real}) \) closed under complementation?

(3) Are \( 1 \text{ ACM}(k, \text{linear}), k \geq 1 \) and \( \bigcup_{1 \leq k < \infty} 1 \text{ ACM}(k, \text{linear}) \) closed under concatenation, Kleene closure, reversal and length-preserving homomorphism?

Acknowledgment

The authors thank the referees for their useful comments.

References


