<table>
<thead>
<tr>
<th>Title</th>
<th>Alternation for Two-Way (Inkdot) Multi-Counter Automata with Sublinear Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>YOSHINAGA, Tsunehiro; INOUE, Katsushi</td>
</tr>
<tr>
<td>Citation</td>
<td>数理解析研究所講究録 (1995), 906: 64-72</td>
</tr>
<tr>
<td>Issue Date</td>
<td>1995-04</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/2433/59460">http://hdl.handle.net/2433/59460</a></td>
</tr>
<tr>
<td>Type</td>
<td>Departmental Bulletin Paper</td>
</tr>
<tr>
<td>Textversion</td>
<td>publisher</td>
</tr>
</tbody>
</table>

Kyoto University
Alternation for Two-Way (Inkdot) Multi-Counter Automata with Sublinear Space

德山高専 義永常宏 (Tsunehiro YOSHINAGA) 井口大学工学部 井上克司 (Katsushi INOUE)

1 Introduction

Alternating Turing machines were introduced in [1] as a mechanism to model parallel computation, and in related papers [8-17], investigations of alternation have been continued. Recently, several properties of Turing machines with small space bounds were given in [6-10,14-18]. For example, von Braunmühl et al. [16] showed that the alternation hierarchy of Turing machines for sublogarithmic space is infinite. Ranjan et al. [18] introduced a slightly modified Turing machine model, called a 1-inkdot Turing machine. The 1-inkdot Turing machine is a Turing machine with the additional power of marking 1 tape-cell in the input (with an inkdot). Inoue et al. [6-8] investigated some accepting powers of 1-inkdot Turing machines and extended this model to that with multi inkdots. It is well known that counter automata without time or space limitations have the same power as Turing machines; however, when time or space restrictions are applied, a different situation emerges. For example, hierarchical properties in the accepting powers of one-way alternating multi-counter automata operating in realtime and alternating multi-counter automata with small space are investigated in [11-13].

In this paper, we investigate an alternation hierarchy of multi-counter automata and 1-inkdot alternating multi-counter automata which have sublinear space. Section 2 gives the definitions and notations necessary for this paper. Let \( \Sigma_{2}\Sigma_{2}^{\ast}Ca(k, L(n)) (strong-2\Sigma_{2}^{\ast}Ca(k, L(n))) \) denote the class of sets accepted by strongly \( L(n) \) space-bounded two-way alternating k-counter automata making at most \( l \) alternations with the initial state existential (universal), let \( weak-2\Sigma_{2}Ca(k, L(n)) (weak-2\Sigma_{2}^{\ast}Ca(k, L(n))) \) denote the class of sets accepted by weakly \( L(n) \) space-bounded two-way alternating k-counter automata making at most \( l \) alternations with the initial state existential (universal), let \( strong-2\Pi_{l}Ca^{\ast}(k, L(n)) \) denote the class of sets accepted by strongly \( k \) space-bounded two-way alternating \( k \)-counter automata making at most \( l \) alternations with the initial state existential (universal), and let \( weak-2\Pi_{l}Ca^{\ast}(k, L(n)) \) denote the class of sets accepted by weakly \( k \) space-bounded two-way alternating \( k \)-counter automata making at most \( l \) alternations with the initial state existential (universal). We denote by \( strong-2\Sigma_{l}TM^{\ast}(L(n)) \) \( (strong-2\Pi_{l}TM^{\ast}(L(n))) \) the class of sets accepted by strongly \( L(n) \) space-bounded two-way alternating Turing machines making at most \( l \) alternations with the initial state existential (universal), denote by \( weak-2\Sigma_{l}TM^{\ast}(L(n)) \) \( (weak-2\Pi_{l}TM^{\ast}(L(n))) \) the class of sets accepted by weakly \( L(n) \) space-bounded two-way alternating Turing machines making at most \( l \) alternations with the initial state existential (universal), and denote by \( weak-2\Sigma_{l}TM^{\ast}(L(n)) \) \( (weak-2\Pi_{l}TM^{\ast}(L(n))) \) the class of sets accepted by weakly \( L(n) \) space-bounded two-way 1-inkdot alternating Turing machines making at most \( l \) alternations with the initial state existential (universal). Section 3 investigates a relationship between the accepting powers of alternating multi-counter automata with and without 1 inkdot. It is shown in [6,8], for example, that for any \( L(n) = o(\log n) \), \( strong-2\Sigma_{l}TM^{\ast}(\log n) \) \( weak-2\Sigma_{l}TM^{\ast}(L(n)) \) \( weak-2\Pi_{l}TM^{\ast}(\log n) \) \( weak-2\Pi_{l}TM^{\ast}(L(n)) \) \( \neq \phi \) and \( strong-2\Pi_{l}TM^{\ast}(\log n) \) \( weak-2\Sigma_{l}TM^{\ast}(L(n)) \) \( weak-2\Pi_{l}TM^{\ast}(\log n) \) \( \neq \phi \). In correspondence to this result, we show, for example, that for any \( L(n) \) such that \( \log L(n) = o(\log n) \), \( strong-2\Sigma_{1}Ca^{\ast}(1, L(n)) \) \( weak-2\Sigma_{1}Ca^{\ast}(1, L(n)) \) \( \neq \phi \) and \( strong-2\Pi_{1}Ca^{\ast}(1, L(n)) \) \( weak-2\Pi_{1}Ca^{\ast}(1, L(n)) \) \( \neq \phi \). Section 4 investigates an infinite alternation hierarchy of alternating multi-counter automata with sublinear space. It is shown in [16], for example, that for each \( l \geq 1 \), and any \( L(n) = o(\log n) \), \( strong-2\Sigma_{l}TM^{\ast}(\log n) \) \( weak-2\Pi_{l}TM^{\ast}(L(n)) \) \( weak-2\Pi_{l}TM^{\ast}(\log n) \) \( \neq \phi \) and \( strong-2\Sigma_{l}TM^{\ast}(\log n) \) \( weak-2\Pi_{l}TM^{\ast}(L(n)) \) \( weak-2\Pi_{l}TM^{\ast}(\log n) \) \( \neq \phi \). In correspondence to this result, we show, for example, that for each \( l \geq 1 \), and any \( L(n) \) such that \( \log L(n) = o(\log n) \), \( strong-2\Pi_{l}TM^{\ast}(1, L(n)) \) \( weak-2\Sigma_{l}Ca^{\ast}(1, L(n)) \) \( weak-2\Sigma_{l}Ca^{\ast}(1, L(n)) \) \( \neq \phi \) and \( strong-2\Pi_{l}TM^{\ast}(1, L(n)) \) \( weak-2\Sigma_{l}Ca^{\ast}(1, L(n)) \) \( weak-2\Sigma_{l}Ca^{\ast}(1, L(n)) \) \( \neq \phi \). Section 5 investigates a relationship between \( m-2\Sigma_{1}Ca^{\ast}(k, L(n)) \) and \( m-2\Pi_{1}Ca^{\ast}(k, L(n)) \) for each \( m \in \{ weak, strong \} \) and each \( k \geq 1 \), and any \( L(n) \) such that \( L(n) \geq \log n \) and \( \log L(n) = o(\log n) \). We show, for example, that \( m-2\Sigma_{1}Ca^{\ast}(k, L(n)) \) \( m-2\Pi_{1}Ca^{\ast}(k, L(n)) \) \( \neq \phi \). Section 6 concludes this paper by giving some open problems.

2 Preliminaries

A multi-counter automaton is a multi-pushdown automaton whose pushdown stores operate as counters, i.e., each storage tape is a pushdown tape of the form \( Z^{*} \) (\( Z \) fixed). (See [4,5] for formal definitions of multi-counter automata.)

A two-way alternating multi-counter automaton (2amca) \( M \) is the generalization of a two-way nondeterministic multi-counter automaton in the same sense as in [1-3]. That is, the state set of \( M \) is divided into two disjoint sets, the set of universal states and the set of existential states. Of course, \( M \) has a specified set of accepting states. We assume that 2amca's have the left endmarker "$\phi" and the right endmarker "$\phi" on the input tape, read the input tape right or left, and can enter an accepting state only when falling off the right endmarker $\phi. We also assume that in one step 2amca's can increment or decrement the contents (i.e., the length) of each counter by at most one. For each \( k \geq 1 \), we denote a two-way alternating \( k \)-counter automaton by \( 2aca(k) \).

An instantaneous description (ID) of \( 2aca(k) \) \( M \) is an element of

\[ \Sigma \times (N \cup \{0\}) \times SM, \]
where $\Sigma (\neq \emptyset \neq \Sigma)$ is the input alphabet of $M$, $N$ denotes the set of all positive integers, and $S_M = Q \times ((Z)^*)^k$, where $Q$ is the set of states of the finite control of $M$, and $Z$ is the storage symbol of $M$. The first and second components, $w$ and $i$, of an ID $I = (w, i, (q_1, (\alpha_1, \ldots, \alpha_k)))$ represent the input string and the input head position, respectively. The third component $(q_1, (\alpha_1, \ldots, \alpha_k))$ of $I$ represents the state of the finite control and the contents of the $k$ counters. An element of $S_M$ is called a storage state of $M$. If $q$ is the state associated with an ID $I$, 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 $w \in \Sigma^*$ is $I_M(w) = (w, 0, (q_0, (\lambda, \ldots, \lambda)))$, where $q_0$ is the initial state of $M$ and $\lambda$ denotes the empty string.

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 $w$ is a sequence $I_0 \vdash_{M} I_1 \vdash_{M} \ldots \vdash_{M} I_n$ ($n \geq 0$), where $I_0 = I_M(w)$. A computation tree of $M$ is a finite, nonempty labeled tree with the following properties:

1. each node $\pi$ of the tree is labeled with an ID, $I(\pi)$,
2. if $\pi$ is an internal node (a non-leaf) of the tree, $I(\pi)$ is universal and $(I(\pi) \vdash_{M} I) = \{I_1, I_2, \ldots, I_L\}$, then $\pi$ has exactly $r$ children $\rho_1, \rho_2, \ldots, \rho_r$, such that $\ell(\rho_i) = I_i$, and
3. if $\pi$ is an internal node of the tree and $\ell(\pi)$ is existential, then $\pi$ has exactly one child $\rho$ such that $\ell(\pi) \vdash_{M} I(\rho)$.

A computation tree of $M$ on input $w$ is a computation tree of $M$ whose root is labeled with $I_M(w)$. An accepting computation tree of $M$ on $w$ is a computation tree of $M$ on $w$ whose leaves are all labeled with accepting ID's. We say that $M$ accepts $w$ if there is an accepting computation tree of $M$ on $w$. We denote the set of input words accepted by $M$ by $T(M)$.

If the state of the finite control of $M$ changes from universal to existential or vice versa, we say that the computation path has an alternation at this point.

A one-way alternating multi-counter automaton (1amca) is a 2amca which reads the input tape from left to right only.

For each $k \geq 1$, let $I_M(k)$ denote a one-way alternating $k$-counter automaton.

Let $L : N \rightarrow R$ be a function, where $R$ denotes the set of all nonnegative real numbers. For each $x \in \{1, 2\}$ and each $k \geq 1$, $x\text{aca}(k)$ is weakly (strongly) $L(n)$ space-bounded if for any $n \geq 1$ and any input $w$ of length $n$ accepted by $M$, there is an accepting computation tree $t$ of $M$ on $w$ such that for each node $\pi$ of $t$, the length of each counter in $\ell(\pi)$ is bounded by $L(n)$ (if for any $n \geq 1$ and any input $w$ of length $n$ accepted or not), and each node $\pi$ of any computation tree of $M$ on $w$, the length of each counter in $\ell(\pi)$ is bounded by $L(n)$. A weakly (strongly) $L(n)$ space-bounded $x\text{aca}(k)$ is denoted by weak-$x\text{aca}(k, (L(n)))$ (strong-$x\text{aca}(k, (L(n)))$).

Let $T : N \rightarrow N$ be a function. For each $m \in \{x\text{weak}, x\text{strong}\}$, each $x \in \{1, 2\}$, and each $k \geq 1$, and any function $L : N \rightarrow R$, we say that an $m\text{aca}(k, (L(n))) M$ operates in time $T(n)$ if for each input $w$ accepted by $M$, there is an accepting computation tree $t$ of $M$ on $w$ such that the length of each computation path of $t$ is at most $T(|w|)$. An $m\text{aca}(k, (L(n))) M$ operates in realtime if $T(n) = n + 1$. For operating time, we are only interested in realtime in this paper.

For each $m \in \{x\text{weak}, x\text{strong}\}$ and each $k \geq 1$, and any function $L : N \rightarrow R$, an $m\text{aca}(k, (L(n)))$ with 1 inkdot, denoted by $m\text{aca}(k, (L(n)))$ can mark 1 tape-cell on the input (with an inkdot). This tape-cell is marked once and for all (no erasing) and no more than one dot of ink is available. The operation of the machine depends on the current state, the currently scanned input, the current contents of counters, and the presence of the inkdot on the currently scanned tape-cell.

For each $m \in \{x\text{weak}, x\text{strong}\}$, each $k \geq 1$ and each $l \geq 1$, and any function $L : N \rightarrow R$, we denote by $m\text{aca}(k, (L(n))) (m\text{xaca}(k, (L(n)))$ a $m\text{aca}(k, (L(n))) making at most $l + 1$ alternations with the initial state existential (universal), denote by $m\text{aca}(k, (L(n))) (m\text{xaca}(k, (L(n)))$ a $m\text{aca}(k, (L(n))) making at most $l + 1$ alternations with the initial state existential (universal), and denote by $m\text{aca}(k, (L(n))) (m\text{xaca}(k, (L(n)))$ a $m\text{xaca}(k, (L(n))) (m\text{xaca}(k, (L(n)))$ with 1 inkdot.

For each $m \in \{x\text{weak}, x\text{strong}\}$ and each $x \in \{1, 2\}$, we define

$$m\text{xaca}(k, (L(n))) = \{I L = T(M) \mid \text{for some } m\text{aca}(k, (L(n))) M\},$$

$$m\text{xaca}(k, (L(n))) = \{I L = T(M) \mid \text{for some } m\text{xaca}(k, (L(n))) M\},$$

$$m\text{xaca}(k, (L(n))) = \{I L = T(M) \mid \text{for some } m\text{xaca}(k, (L(n))) M\},$$

$$m\text{xaca}(k, (L(n)) \text{ with } 1 \text{ inkdot}) = \{I L = T(M) \mid \text{for some } m\text{xaca}(k, (L(n)) \text{ with } 1 \text{ inkdot}) M\}.$$

An alternating Turing machine (aTm) we consider in this paper has a read-only input tape with the left endmarker $\#$, the right endmarker $\$, and a separate storage tape. (The reader is referred to [9,10] for the formal definition of $aTm$.) For any $L : N \rightarrow R$, we denote a weakly (strongly) $L(n)$ space-bounded one-way $aTm$ by weak-1aTm$(L(n))$ (strong-1aTm$(L(n))$), and denote a weakly (strongly) $L(n)$ space-bounded two-way $aTm$ by weak-2aTm$(L(n))$ (strong-2aTm$(L(n))$). (See [6-10,14-18] for the definition of weakly (strongly) $L(n)$ space-bounded aTms.) For each $m \in \{x\text{weak}, x\text{strong}\}$, and any $L : N \rightarrow R$, we denote a $m\text{aca}(L(n))$ with 1 inkdot by $m\text{aca}(L(n))$, and any function $L : N \rightarrow R$, we denote by $m\text{aca}(L(n)) (m\text{aca}(L(n))$ a $m\text{aca}(L(n)) making at most $l + 1$ alternations with the initial state existential (universal), denote by $m\text{aca}(L(n)) (m\text{aca}(L(n))$ a $m\text{aca}(L(n)) making at most $l + 1$ alternations with the initial state existential (universal), and denote by $m\text{aca}(L(n)) (m\text{aca}(L(n))$ a $m\text{aca}(L(n)) (m\text{aca}(L(n))$ with 1 inkdot.

For each $m \in \{x\text{weak}, x\text{strong}\}$, each $x \in \{1, 2\}$, we define

$$m\text{xaca}(L(n)) = \{I L = T(M) \mid \text{for some } m\text{xaca}(L(n)) M\},$$

$$m\text{xaca}(L(n)) = \{I L = T(M) \mid \text{for some } m\text{xaca}(L(n)) M\},$$

$$m\text{xaca}(L(n)) = \{I L = T(M) \mid \text{for some } m\text{xaca}(L(n)) M\}.$$

1. We note that $0 \leq i \leq |w| + 2$, where for any string $w$, $|w|$ denotes the length of $w$. "$^0", "^1", "^{|w| + 1}" and "^{|w| + 2}" represent the positions of the left endmarker $\#$, the leftmost symbol of $w$, the right endmarker $\$, and the immediate right to $\$, respectively.
m-xATM(L(n)) = \{L| L = T(M) for some m-xATM(L(n)) M\},
m-xΣTM(L(n)) = \{L| L = T(M) for some m-xΣATM(L(n)) M\},
m-xΠTM(L(n)) = \{L| L = T(M) for some m-xΠATM(L(n)) M\},
m-2ΣTM(L(n)) = \{L| L = T(M) for some m-2ΣTM(L(n)) M\},
and m-2ΠTM(L(n)) = \{L| L = T(M) for some m-2ΠTM(L(n)) M\}.

The following lemmas can be easily proved.

Lemma 2.1. For each \(m \in \{\text{weak, strong}\}\), each \(x \in \{1,2\}\) and each \(i \geq 1\), and any function \(L: N \rightarrow R\),

\[
\bigcup_{1 \leq k < \infty} m-xATM(k, L(n)) \subseteq m-xATM(\log L(n)) \quad \text{and} \quad \bigcup_{1 \leq k < \infty} m-xΣATM^*(k, L(n)) \subseteq m-2ΣTM^*(\log L(n)),
\]

\[
\bigcup_{1 \leq k < \infty} m-xΠATM(k, L(n)) \subseteq m-2ΠTM(\log L(n)),
\]

It is shown in [17] that for any functions \(F(n) = o(\log n)\) and \(G(n) = o(\log \log n)\),

\[\text{strong-1ATM}(F(n)) \quad \text{is the class of regular sets, and} \quad \text{weak-2ATM}(G(n)) \quad \text{is the class of regular sets.}\]

From this fact and Lemma 2.1, we can show that for any \(k \geq 1\), any functions \(F: N \rightarrow R\) such that \(\log F(n) = o(\log n)\) and \(G: N \rightarrow R\) such that \(\log G(n) = o(\log \log n)\),

\[\text{strong-1ACA}(k, F(n)) \quad \text{is the class of regular sets and} \quad \text{weak-2ACA}(k, G(n)) \quad \text{is the class of regular sets.}\]

3 The power of 1 inkdot

This section investigates a relationship between the accepting powers of space-bounded 2amca's with and without 1 inkdot. This investigation is based on the results of 2atm's [6,8]. Inoue et al. [6,8] showed that for any function \(L(n) = o(\log n)\),

\[\text{strong-2ΣTM}^*(\log L(n)) \quad \text{and} \quad \text{weak-2ΣTM}(L(n)) \neq \emptyset,
\]

\[\text{strong-2ΠTM}^*(\log L(n)) \quad \text{and} \quad \text{weak-2ΠTM}(L(n)) \neq \emptyset,
\]

In correspondence to this result, we can show several results for 2amca's. In order to do so, we first give the following two lemmas.

Lemma 3.1. Let \(L_1 = \{B(1)\}B(2)\} \ldots B(n)\}cw_1cw_2c \ldots cw_{\ell}cw_\ell(n) \geq 2 \quad \text{and} \quad \{0,1\}^{\log n} \quad \text{for} \quad w \in \{0,1\}^{\log n} \quad \text{and} \quad \forall i(1 \leq i \leq r)(w = w_i),
\]

and \(L_2 = \{B(1)\}B(2)\} \ldots B(n)\}cw_1cw_2c \ldots cw_{\ell}cw_\ell(n) \geq 2 \quad \text{and} \quad \{0,1\}^{\log n} \quad \text{for} \quad w \in \{0,1\}^{\log n} \quad \forall i(1 \leq i \leq r)(w = w_i)
\]

where each positive integer \(i \geq 1\), \(B(i)\) denotes the string in \(\{0,1\}^{\log n}\) that represents the integer \(i\) in binary notation (with no leading zeros). Then,

\begin{enumerate}
  \item \(L_1 \in \text{strong-2ΣTM}^*(1, \log n)\), and
  \item \(L_2 \in \text{strong-2ΠTM}^*(1, \log n)\), and
\end{enumerate}

and for any function \(L: N \rightarrow R\) such that \(\log L(n) = o(\log n)\),

\begin{enumerate}
  \item \(L \notin \bigcup_{1 \leq k < \infty} \text{weak-2ΣTM}^*(k, L(n))\), and
  \item \(L \notin \bigcup_{1 \leq k < \infty} \text{weak-2ΠTM}^*(k, L(n))\).
\end{enumerate}

The proof of (1) (resp., (2)): We can construct a strong-2ΣTM^*(1, log n) (resp., strong-2ΠTM^*(1, log n)) \(M\) which acts as follows. Suppose that an input string

\[f^{n_1}_{y_1}f^{n_2}_{y_2} \ldots f^{n_\ell}_{y_\ell}c_{w_1}c_{w_2}c \ldots c_{w_\ell}\mathrm{cw}_\ell(n) \geq 2 \quad \text{and} \quad \{0,1\}^{\log n} \quad \text{for} \quad w \in \{0,1\}^{\log n} \quad \text{and} \quad \forall i(1 \leq i \leq r)\}
\]

(\(n \geq 2\), \(r \geq 1\), and \(y_i\)'s, \(w_j\)'s and \(w\) are all in \(\{0,1\}^*\)) is presented to \(M\). (Input strings in the form different from the above can easily be rejected by \(M\).) For each \(i(1 \leq i \leq n)\), \(M\) can first check whether \(y_i = B(i)\) and store \(Z^{\log n}\) in its counter when \(y_i = B(i)\) (for example, by using the algorithm in [13]. (Of course, \(M\) never enters an accepting state if \(y_i \neq B(i)\) for some \(1 \leq i \leq n\).) If \(M\) successfully completes the action above, then it checks by using \(Z^{\log n}\) stored in the counter that \(w = [\log n]\) (resp., \(w = [\log n]\) and \(|w| = [\log n]\) for each \(1 \leq i \leq r)\). After that, \(M\) existentially guesses some \(j(1 \leq j \leq r)\), and marks the symbol \(c\) just before \(w_j\) by the inkdot in order to check whether \(w = w_j\) (resp., \(M\) universally branches and marks the symbol \(c\) just before \(w_j\) by the inkdot in order to check whether \(w \neq w_j\) for each \(j(1 \leq j \leq r)\). Finally, \(M\) checks by using \(Z^{\log n}\) in its counter that \(w_j = [\log n]\), and then deterministically checks by using the inkdot as a pilot that \(w = w_j\) (after deterministically checks by using the inkdot as a pilot that \(w \neq w_j\)). That is, for example, \(M\) stores \(Z'(1 \leq i \leq |w|) = |w| = [\log n]\) in its counter when \(M\) picks up the symbol \(w_j(i)\) and by using \(Z'\) in its counter compares \(w_j(i)\) with \(w(i)\) while moving its input head back and forth. (For the check, it is clear that log \(n\) space is sufficient.) \(M\) enters an accepting state only if these checks are all successful. It will be obvious that \(M\) accepts the language \(L_1\) (resp., \(L_2\)).

The proofs of (3) and (4): It is shown in [6,8] that \(L_1 \notin \text{weak-2ΣTM}(L(n))\) and \(L_2 \notin \text{weak-2ΠTM}(L(n))\) for any function \(L(n) = o(\log n)\). From this result and lemma 2.1, (3) and (4) follow.

\[\square\]

Lemma 3.2. Let \(L_3 = \{B(1)\}B(2)\} \ldots B(n)\}cw_1cw_2c \ldots cw_{\ell}cw_\ell(n) \geq 2 \quad \text{and} \quad \{r, r' \geq 1\} \quad \forall i(1 \leq i \leq r)\}

\[\text{from now on, logarithms are base 2.}\]

\[\text{for each string } w, w(i) \text{ denotes the } i\text{-th symbol (from the left) of } w.\]
An infinite alternation hierarchy without inkdots

It is shown in [16] that the alternation hierarchy for aTM’s with space-bounds between log log n and log n is infinite. This section investigates an infinite alternation hierarchy for amca’s with sublinear space (without inkdots). Throughout this section, we need the languages in [16] described in the following.

For each \( i \in N \), a special symbol \( \# i \) is introduced. Let
\[
D_1 = \{0,1\}^* \quad \text{and} \quad D_{i+1} = (D_i(\#i))^* \cdot D_i \quad \text{for each } i \geq 1,
\]
\[
\forall D_1(u) = \emptyset \quad \text{and} \quad \forall D_i(u) = \{u\} \quad \text{and} \quad \forall D_{i+1}(u) = \{W_1 \# \ldots \# W_m \in D_{i+1} | \exists j (1 \leq j \leq m) \forall D_i(W_j), m \geq 1 \}
\]

Now, let us define the following witness languages: for each \( i \) \geq 2,
\[
L_i^T = \bigcup \{3D_i(u) \cdot \{uB(n)^i\}^* \cdot \{uB(n-1)^i\}^* \ldots \{uB(1)^i\}^* | u \in \{0,1\}^{\lfloor \log n \rfloor}, n \geq 2 \}.
\]

For each \( i \) \geq 2,
\[
L_i^T \subseteq \text{weak-1} \Pi_i \text{TM}(\log \log n) \quad \text{and} \quad L_i^T \subseteq \text{weak-1} \Pi_i \text{TM}(\log \log n), 
\]
\[
\forall D(u) \subseteq \{uB(n)^i\}^* \cdot \{uB(n-1)^i\}^* \ldots \{uB(1)^i\}^* | u \in \{0,1\}^{\lfloor \log n \rfloor}, n \geq 2 \}.
\]

Lemma 4.1. For each \( i \) \geq 2,
\[
(1) L_i^T \subseteq \text{weak-1} \Sigma_i \text{TM}(\log \log n) \quad \text{and} \quad L_i^T \subseteq \text{weak-1} \Sigma_i \text{TM}(\log \log n), 
\]
\[
(2) L_i^T \subseteq \text{strong-2} \Sigma_i \text{TM}(\log \log n) \quad \text{and} \quad L_i^T \subseteq \text{strong-2} \Sigma_i \text{TM}(\log \log n),
\]

\(^4\)For each string \( u \), \( \overline{u} \) denotes the reversal of \( u \).
and for any function $L(n) = o(\log n)$,
(3) $L^\exists \not\in \text{weak-2}\Pi TM(L(n))$ and $L^\forall \not\in \text{weak-2}\Sigma TM(L(n))$.

In correspondence to this result, we can show the following lemma.

**Lemma 4.2.** For each $l \geq 2$,
(1) $L^\exists \not\in \text{weak-1}\Pi \land CA(1,\log n, \text{real})$ and $L^\forall \not\in \text{weak-1}\Pi \land CA(1,\log n, \text{real})$, and
(2) $L^\exists \not\in \text{strong-2}\Sigma \land CA(1,\log n)$ and $L^\forall \not\in \text{strong-2}\Sigma \land CA(1,\log n)$,
and for any function $L : N \rightarrow R$ such that $\log L(n) = o(\log n)$,
(3) $L^\exists \not\in \bigcup_{\text{such \, weak-2}\Pi \land CA(k,L(n))}$ and $L^\forall \not\in \bigcup_{\text{such \, weak-2}\Sigma \land CA(k,L(n))}$.

**The proof of (1):** We prove (1) of this lemma by using induction for $l \geq 2$.

1. (I) The following weak-1\lor\land CA(1,\log n) $M^2_2$ operating in realtime accepts the language

$L^2_2 = \{w_1 1 w_2 1 \ldots 1 w_m 1 u B(n) 1 \ldots 1 B(1) 1 | \text{n} \geq 2 \ \& \ m \geq 1 \ \& \ t \geq 1 \ \& \ u \in \{0,1\}^{\lceil \log n \rceil} \ \& \ \forall i(1 \leq i \leq m)[w_i \in D_1] \ \& \ \exists j(1 \leq j \leq m)[w_i = u]\}$

Suppose that an input string

$x = w_1 1 w_2 1 \ldots 1 w_m 1 u y_{n-1} 1 \ldots 1 y_1 1$

(where $n \geq 2, m \geq 1$ and $t \geq 1$, and $w_i's, y_j's$ and $u$ are all in \{0,1\}^*) is presented to $M^2_2$. (Input strings in the form different from the above can easily be rejected by $M^2_2$.)

$M^2_2$ first existentially guesses some $j$, and runs to $w_j$. $M^2_2$ then makes a universal branch.

(A) In one branch, in order to check whether $w_j = u = M^2_2$ universally checks that $w_j(i) = u(i)$ for each $i (1 \leq i \leq |w_j|)$. That is, to verify $w_j(i) = u(i)$. $M^2_3$ stores $i$ in its counter when it picks up the symbol $w_j(i)$, compares the symbol $w_j(i)$ with the symbol $u(i)$ by using $Z^l$ in the counter, and enters an accepting state only if $w_j(i) = u(i)$.

(B) In another branch, $M^2_3$ branches to check the following two points:

(a) whether $|u| = |y_{n-1}|$, and
(b) whether $y_i = B(i)^R$ for each $i (1 \leq i \leq n)$.

(a) above can easily be checked by using only one counter, and $M^2_3$ enters an accepting state only if (a) is successfully checked.

(b) above can be checked as follows. $M^2_2$ essentially uses the algorithm in [3,13]. By using universal branches and only one counter, $M^2_2$ can check in a way described below whether $y_i = B(i)^R$ for each $i (1 \leq i \leq n)$. $M^2_2$ compares $y_{n-1}$ with $y_i$, and verifies that $y_{n-1}^R$ represents in binary notation (with no leading zeros) an integer which is one more than the integer represented by $y_{n-1}$ in binary notation (with no leading zeros). In doing so, $M^2_2$ will compare the $j$-th symbols of $y_i$ and $y_{n-1}$, for all appropriate $j$. Observe that if $y_{n-1}^R$ is one more than $y_i^R$, then (i) $y_{n-1}^R = 0^n 1$ and $y_i = 0^m 0^m$, where $x$ is a string (finishing with 1) over \{0,1\} and $m$ is some non-negative integer, or (ii) $y_{n-1}^R = 0^m 1$ and $y_i = 1^m 0^m$, where $m$ is some positive integer. Let $C$ be the counter of $M^2_2$. For each $j (1 \leq j \leq |y_{n-1}^R|)$, $M$ stores the symbol $y_{n-1}^R(j)$ in its finite control and $Z^l$ in $C$ just after it has read the symbol $y_{n-1}^R(j)$, and makes a universal branch.

- In one branch, it compares $y_{n-1}^R(j)$ with the symbol $y_i(j)$ using $Z^l$ stored in $C$, and checks whether both the symbols satisfy (i) or (ii) above. (It determines whether they should be the same or not, by checking the first occurrence of 1 in $y_{n-1}$. If the symbol 1 has already occurred, then $y_{n-1}^R(j)$ and $y_i(j)$ should be the same; otherwise, $y_i(j)$ and $y_{n-1}^R(j)$ should not be the same.)

- In another branch, it reads the next symbol $y_i(j + 1)$, stores it in the finite control, and adds $Z$ in $C$ in order to store $Z^l + 1$ in $C$.

In this way, $M^2_2$ can check that $y_{n-1}^R$ is one more than $y_i^R (1 \leq i \leq n)$ using only universal branches and only one counter, and operating in one-way. It will be obvious that if $y_{n-1} \ldots y_1$ is such a string that $y_i = B(i)^R$ for each $i (1 \leq i \leq n)$, then the length of $y_{n-1} = \Vert B(n)^R \Vert$ is equal to $\lceil \log n \rceil$, and thus the length of $C$ is bounded by $\log n$. Furthermore, it is clear that $M^2_2$ operates in realtime.

(II) $L^2_3 = \{w_1 1 w_2 1 \ldots 1 w_m 1 u B(n) 1 \ldots 1 B(1) 1 | n \geq 2 \ \& \ m \geq 1 \ \& \ t \geq 1 \ \& \ u \in \{0,1\}^{\lceil \log n \rceil} \ \& \ \forall i(1 \leq i \leq m)[w_i \in D_1] \ \& \ \not\exists j(1 \leq j \leq m)[w_i = u]\}$ is accepted by a realtime weak-1\lor\land CA(1,\log n) $M^2_3$ as follows. Suppose that an input string

$x = w_1 1 w_2 1 \ldots 1 w_m 1 u y_{n-1} 1 \ldots 1 y_1 1$

(where $n \geq 2, m \geq 1$ and $t \geq 1$, and $w_i's, y_j's$ and $u$ are all in \{0,1\}^*) is presented to $M^2_3$. (Input strings in the form different from the above can easily be rejected by $M^2_3$.)

$M^2_3$ moves on $x$ while making a universal branch at the first symbol of each $w_i (1 \leq i \leq m)$.

(A) In one branch, $M^2_3$ continues the action above until it reads the first $1$, and then makes a universal branch to check the following two points:

(a) whether $|u| = |y_{n-1}|$, and
(b) whether $y_i = B(i)^R$ for each $i (1 \leq i \leq n)$

(B) In another branch, $M^2_3$ immediately enters an existential state, guesses some $j (1 \leq j \leq |w_i|)$ and compares $w_i(j)$ with $u(j)$ in order to check that $w_i(j) \neq u(j)$. 


(a) and (b) can be checked in a way as described in (1).
2. Assume that assertion (1) of this lemma holds for \(L^i\) and \(L^j\) \((i = 3, 4, \ldots, I - 1)\). We shall prove assertion (1) of the lemma holds for \(L^I_i\) and \(L^I_j\), too.

(I) An input string \(x\) in \(L^I_i\) has the form \(x = WSW_{i}^{1-i}W_{j}^{1-i}W_{m}\) with \(W_{i}\) in \(\forall D_{i}(u)\) for some \(i(1 \leq i \leq m)\), and \(S = \{B(1)B(2)\ldots B(n)\}^{i}\), where \(u\) is in \(\{0, 1\}^{log n}\), \(\ell \geq 1\), \(m \geq 1\) and \(n \geq 2\). By the assumption above, there is a real-time weak-1\(\Sigma_{1}CA(1, log n)\) \(M_{i}^{\prime}\) which accepts \(W_{i}\) if \(W_{i}\) is in \(\forall D_{i}(u)\). \(L^I_i\) is accepted by a real-time weak-1\(\Sigma_{1}CA(1, log n)\) \(M^I\) acts as follows. Suppose that an input string \(x = W_{i}^{1-i}W_{j}^{1-i}W_{m}\)

(II) An input string \(x\) in \(L^I_j\) has the form \(x = WSW_{i}^{1-i}W_{j}^{1-i}W_{m}\) with \(W_{i}\) in \(\forall D_{i}(u)\) for all \(i(1 \leq i \leq m)\), and \(S = \{B(1)B(2)\ldots B(n)\}^{i}\), where \(u\) is in \(\{0, 1\}^{log n}\), \(\ell \geq 1\), \(m \geq 1\) and \(n \geq 2\). By the assumption above, there is a real-time weak-1\(\Sigma_{1}CA(1, log n)\) \(M^I\) which accepts \(W_{i}\) if \(W_{i}\) is in \(\forall D_{i}(u)\). There is a real-time weak-1\(\Sigma_{1}CA(1, log n)\) \(M_{i}^{\prime}\) which acts \(L^I_j\) as follows. Suppose that an input string \(x\) described in (I) above is presented to \(M^I_{i}\). \(M_{i}^{\prime}\) moves on \(x\) while making a universal branch at the first symbol of each \(w_{i}\) \((1 \leq i \leq m)\).

(a) In one branch, \(M^I_{i}\) continues the action above until it reaches the first \(S\). After that, \(M^I_{i}\) runs to the right endmarker \(S\), and enters an accepting state.

(b) In another branch, \(M^I_{j}\) enters an existential state, and acts just like \(M^I_{2i-1}\) above, but ignores all the segments between the next \(\ell\) and the first \(w_{a}\).

Clearly, the lengths of the counters of \(M^I_{i}\) and \(M^I_{j}\) are bounded by \([log n]\), because those used in the computation are basically equal to the lengths of the counters of \(M^I_{i}\) and \(M^I_{j}\) when they enter accepting states. \(M^I_{i}\) and \(M^I_{j}\) on accepted inputs use no more than \([log n]\) space, which is shown as in 1 above.

The proof of (2): It is shown in [13] that the language \(\{B(1)B(2)\ldots B(n)\}^{i}\) is accepted by a strongly \(log n\) space-bounded two-way deterministic 1-counter automaton. For each \(i \geq 2\), \(L^I_{i}\) (resp., \(L^I_{j}\)) can be accepted by \(strong-2\Sigma_{1}CA(1, log n)\) (resp., \(strong-2\Sigma_{1}CA(1, log n)\)) \(M^I\) as follows. \(M^I\) begins by examining whether the suffix of a given input is of the form \(B^{i}(n)^{i}B^{i}(n-1)^{i}...B^{i}(1)^{i}\) \((\ell = 1)\) in a way as in [13]. If this examination is successful and \(M^I\) stores \(\sum_{1}log n\) in its counter, then \(M^I\) can check by using the same technique as in the proof of (1) of this lemma whether the given string is a desired one.

The proof of (3): It is shown in [16] that for any function \(L(n) = o(log n)\), \(L^I_{i}\) and \(L^I_{j}\) are not in \(weak-2\Sigma_{1}TM(L(n))\) and \(weak-2\Sigma_{1}TM(L(n))\), respectively. (3) follows from this result and Lemma 2.1.

From this lemma, we have the following theorem and corollaries.

Theorem 4.1. For each \(i \geq 2\), and any function \(L : N \rightarrow R\) such that \(log L(n) = o(log n)\),

(1) \(weak-2\Sigma_{1}CA(k, log n, real) \cap \Sigma_{2}CA(1, log n) = \Sigma_{2}CA(k, log n, real) \neq \emptyset\),

(2) \(weak-2\Pi_{1}CA(k, log n, real) \cap \Sigma_{2}CA(1, log n) = \Sigma_{2}CA(k, log n, real) \neq \emptyset\).

Corollary 4.1. For each \(i \geq 2\), each \(k \geq 1\) and each \(m \in \{weak, strong\}\), and any function \(L : N \rightarrow R\) such that \(\Sigma_{1}\) \(\geq log n\) and \(log L(n) = o(log n)\),

(1) \(\Sigma_{2}\Sigma_{1}CA(k, L(n))\) is incomparable with \(\Sigma_{2}\Pi_{1}CA(k, L(n))\),

(2) \(\Sigma_{2}\Pi_{1}CA(k, L(n))\) is incomparable with \(\Sigma_{2}\Pi_{1}CA(k, L(n))\), and

(3) \(\Sigma_{2}Pi_{1}CA(k, L(n), real) \neq \Sigma_{2}Pi_{1}CA(k, L(n), real)\).

Corollary 4.2. For each \(i \geq 1\), each \(k \geq 1\), each \(m \in \{weak, strong\}\) and each \(X, Y \in \{\Sigma, \Pi\}\), and any function \(L : N \rightarrow R\) such that \(L(n) \geq log n\) and \(log L(n) = o(log n)\),

(1) \(\Sigma_{2}X_{1}CA(k, L(n)) \subseteq \Sigma_{2}Y_{1}CA(k, L(n))\),

(2) \(\Sigma_{2}X_{1}CA(k, L(n)) \subseteq \Sigma_{2}Y_{1}CA(k, L(n))\),

(3) \(\Sigma_{2}X_{1}CA(k, L(n), real) \subseteq \Sigma_{2}Y_{1}CA(k, L(n), real)\).

We then show a relationship between one-way and two-way, and strongly and weakly space-bounds.

Theorem 4.2. For each \(X \subseteq \{\Sigma, \Pi\}\), and any function \(L : N \rightarrow R\) such that \(L(n) \geq log n\) and \(log L(n) = o(log n)\),

\(\Sigma_{2}X_{1}CA(1, log n) \neq \Sigma_{2}X_{1}CA(1, log n)\).

Proof. Let \(L_{S} = \{(B(1)B(2))\ldots B(n)\}^{i}c_{1}c_{2}c_{3}\ldots c_{n}|n \geq 2\) and \(r \geq 1\) and \(w \in \{0, 1\}^{log n}\) \& \(\forall i(1 \leq i \leq r)|w_{i}| \in \{0, 1\}^{*}\) \& \(\exists j(1 \leq j \leq r)|w_{j}| \geq 2\) \& \(r \geq 1\) and \(w \in \{0, 1\}^{log n}\) \& \(\forall i(1 \leq i \leq r)|w_{i}| \in \{0, 1\}^{*}\) \& \(w \neq w_{i}\). Then, \(L_{S} \in \Sigma_{2}CA(1, log n)\) and \(L_{S} \notin \Sigma_{2}CA(1, log n)\) are essentially proved in [13]. (3) \(L_{S} \in \Sigma_{2}CA(1, log n)\) and (4) \(L_{S} \notin \Sigma_{2}CA(1, log n)\) can be proved in the same way as the
proof of Lemma 4.1 in [13]. So, the proof is omitted here.

Corollary 4.3. For each $l \geq 2$ and each $k \geq 1$, and any function $L : N \rightarrow R$ such that $L(n) \geq \log n$ and $log L(n) = o(\log n)$, weak-$1\Sigma(CA(k, L(n))) \subsetneq$ weak-$2\Sigma(CA(k, L(n)))$, and weak-$1\Pi(CA(k, L(n))) \subsetneq$ weak-$2\Pi(CA(k, L(n)))$.

Let weak-2DCA($k, L(n)$) denote the class of sets accepted by weakly $L(n)$ space-bounded two-way deterministic $k$-counter automata. It is shown in [13] that for any function $L : N \rightarrow R$ such that $L(n) \geq \log n$ and $log L(n) = o(\log n)$,

$$\text{weak-2DCA}(4, \log n) - \bigcup_{1 \leq k < \infty} \text{strong-2ACA}(k, L(n)) \neq \phi,$$

and

$$\text{weak-2\Sigma}(3, \log n) - \bigcup_{1 \leq k < \infty} \text{strong-2\Sigma}(k, L(n)) \neq \phi.$$

From this result and Theorem 4.1, the following corollary is shown.

Corollary 4.4. For each $l \geq 1$, each $i \geq 3$ and each $j \geq 4$, and any function $L : N \rightarrow R$ such that $L(n) \geq \log n$ and $log L(n) = o(\log n)$,

(1) $\text{strong-2\Sigma}CA(i, L(n)) \subsetneq \text{weak-2\Sigma}CA(i, L(n))$, and

(2) $\text{strong-2\Pi}CA(j, L(n)) \subsetneq \text{weak-2\Pi}CA(j, L(n))$.

5 An alternation hierarchy on the first level with 1 inkdot

Inoue et al. [8] showed that for any function $L(n) = o(\log n)$,

$$\text{strong-}2\Pi_3\mathrm{T}M^*(\log \log n) - \bigcup_{1 \leq k < \infty} \text{strong-2\Sigma}CA(k, L(n)) \neq \phi,$$

$$\text{strong-}2\Sigma_3\mathrm{T}M^*(\log \log n) - \bigcup_{1 \leq k < \infty} \text{weak-2\Sigma}CA(k, L(n)) \neq \phi.$$

In correspondence to this result, from Lemma 3.2 (1), (2), (5) and (6), it follows that for each $k \geq 1$, and any function $L : N \rightarrow R$ such that $log L(n) = o(\log n)$,

$$\text{strong-}2\Pi_k\mathrm{CA}^*(1, \log n) - \bigcup_{1 \leq k < \infty} \text{strong-2\Sigma}_k\mathrm{CA}^*(1, \log n) \neq \phi,$$

$$\text{strong-}2\Sigma_k\mathrm{CA}^*(1, \log n) - \bigcup_{1 \leq k < \infty} \text{weak-2\Sigma}_k\mathrm{CA}^*(1, \log n) \neq \phi.$$

We can strengthen this result as Theorem 5.1 below shows. We need the following key lemma.

Lemma 5.1. Let $L_l = \{B(1)^j B(2)^j \ldots B(n)^j \mid w_{1j} \in \{0, 1\}^{\{0, 1\}} \} \subset \text{strong-2\Pi}_3\mathrm{T}M^*(\log \log n)$ and $L_l = \{w_{1j} \in \{0, 1\}^{\{0, 1\}} \}$ and $L_l = \{B(1)^j B(2)^j \ldots B(n)^j \mid w_{1j} \in \{0, 1\}^{\{0, 1\}} \}$ for each $l \geq 1$.

We consider the computations of $M$ on the strings in $V(n)$. Let $l(n)$ be the length of each element in $V(n)$. Then $\text{strong-}2\Pi_k\mathrm{CA}(1, \log n, \text{real})$, and

$$\text{weak-}2\Pi_k\mathrm{CA}(1, \log n, \text{real}),$$

and for any function $L : N \rightarrow R$ such that $log L(n) = o(\log n)$,

$$\text{strong-}2\Pi_k\mathrm{CA}^*(1, \log n),$$

$$\text{weak-}2\Pi_k\mathrm{CA}^*(1, \log n),$$

The proofs of (1), (2), (3) and (4): The proofs are similar to those of Lemmas 3.1, 3.2, and 4.2. We leave the proofs to the reader as an exercise.

The proof of (5): Suppose that there exists a weak-$2\Pi_1\mathrm{CA}^*(1, L(n))$ accepting $L_l$ for some $k \geq 1$, where $log L(n) = o(\log n)$. For each $n \geq 2$, let

$$V(n) = \{B(1)^j B(2)^j \ldots B(n)^j \mid w_{1j} \in \{0, 1\}^{\log n} \},$$

where

$$W(n) = \{w_{1j} \in \{0, 1\}^{\log n} \mid w_1 \in \{0, 1\}^{\log n} \}.$$
$b(y) \neq b(y')$. We can, without loss of generality, assume that there is a string $u \in \{0,1\}^{\log n}$ such that $u \in b(y') - b(y)$. Consider the following string $z$:

$$z = B(1)^{y_1}B(2)^{y_2} \cdots B(n)^{\infty},$$

where $y_1 = y_2 = y$. As is easily seen, $z$ is in $L$, and so there exists an $L((n))$ space-bounded accepting computation of $M$ on $z$. We denote the accepting computation by $\text{comp}(z)$. Since $M$ has 1 inkdot, it follows that there is some string $y_i (i \in \{1,2\})$ such that $M$ never consumes the inkdot on $y_i$ in $\text{comp}(z)$. Let $y'$ be the string obtained from $z$ by replacing $y_i (= y)$ by $y'$. From $\text{comp}(z)$, we can construct an $L((n))$ space-bounded accepting computation of $M$ on $y'$. This is a contradiction, since $z$ is not in $L$.

**The proof of (6):** The proof is similar to that of (5) of this lemma. Suppose, to the contrary, that there exists a weak-$2\text{ca}^*(k, L(n))$ accepting $L_d$ for some $k \geq 1$, where $\log L(n) = o(\log n)$. For each $n \geq 2$, let

$$V(n) = \{y_1 \in \{0,1\}^{\log n} : (y_1, y_2) \in W(n) \} \text{ and } (y_1, y_2) \in \{0, 1\}^{\log n} \},$$

where $y_1 = y_2 = y$. Clearly, $z$ is not in $L_d$, and so $z$ is never accepted by $M$, that is, there is at least one computation path

$$I_{M}^{(z)} = I_0 \uparrow M_1 \uparrow M_2 \cdots \uparrow M_m \text{ such that } m \geq 1 \text{ of } M \text{ on } z \text{ with the following properties (P1), (P2) and (P3):}

(P1) For each $i (0 \leq i \leq m-1)$, $I_i$ is not an accepting ID and the length of each counter in $I_i$ is bounded by $L(|z|)$.

(P2) For each $i,j$ (0 \leq i,j \leq m-1, i \neq j), I_i \neq I_j$ and

(P3) $I_m$ is such that (i) $I_m$ is a non-accepting halting ID, (ii) $I_m = I_i$ for some $i (0 \leq i \leq m-1)$, or (iii) the length of some counter is larger than $L(|z|)$.

Fix such a computation path of $M$ on $z$ with the properties (P1), (P2) and (P3) above, and denote it by $\text{comp}(z)$. Since $M$ has 1 inkdot, it follows that there is some string $y_i (i \in \{1,2\})$ such that $M$ never consumes the inkdot on $y_i$ in $\text{comp}(z)$. Let $y'$ be the string obtained from $z$ by replacing $y_i (= y)$ by $y'$. From $\text{comp}(z)$, we can easily construct a computation path of $M$ on $y'$ with the properties (P1), (P2) and (P3). Thus, $y'$ is rejected by $M$. This is a contradiction, because $z$ is in $L_d$. 

From this lemma, we have the following theorem.

**Theorem 5.1.** For any function $L : N \to R$ such that $\log L(n) = o(\log n)$,

(1) $\text{strong-2I} \Pi_1 \text{CA}^*(k, L(n)) - \bigcup_{k \leq \log n} \text{weak-2I} \Pi_1 \text{CA}^*(k, L(n)) \neq \phi$, 

(2) $\text{strong-2I} \Sigma_1 \text{CA}^*(k, L(n)) - \bigcup_{k \leq \log n} \text{weak-2I} \Sigma_1 \text{CA}^*(k, L(n)) \neq \phi$, 

(3) $\text{strong-2I} \Pi_1 \text{CA}(1, \log n)$ \cap weak-1I \Pi_1 \text{CA}(1, \log n, \text{real}) - \bigcup_{k \leq \log n} \text{weak-2I} \Pi_1 \text{CA}^*(k, L(n)) \neq \phi$, and 

(4) $\text{strong-2I} \Sigma_1 \text{CA}(1, \log n)$ \cap weak-1I \Sigma_1 \text{CA}(1, \log n, \text{real}) - \bigcup_{k \leq \log n} \text{weak-2I} \Sigma_1 \text{CA}^*(k, L(n)) \neq \phi$.

**Corollary 5.1.** For each $k \geq 1$, each $m \in \{\text{strong}, \text{weak}\}$, and each $X \in \{\Pi_1, \Sigma_1\}$, and any function $L : N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$,

(1) $m-2I \Pi_1 \text{CA}^*(k, L(n)) \nsubseteq m-2I \Sigma_1 \text{CA}^*(k, L(n))$, and 

(2) $m-2I \Sigma_1 \text{CA}^*(k, L(n)) \nsubseteq m-2I \Pi_1 \text{CA}^*(k, L(n))$.

For the standard 2amca model, a relationship between $m-2I \Pi_1 \text{CA}^*(k, L(n))$ and $m-2I \Sigma_1 \text{CA}^*(k, L(n))$ is unknown for each $k \geq 1$, each $m \in \{\text{strong}, \text{weak}\}$, and any function $L : N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$. On the other
hand, for the nkdot 2amca model, we have the following result.

**Corollary 5.2.** For each $k \geq 1$ and each $m \in \{\text{strong, weak}\}$, and any function $L : N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$, m-2\Pi_1CA^*(k, L(n))$ is incomparable with $m-2\Sigma_1CA^*(k, L(n))$.

### 6 Conclusion

We conclude this paper by listing up some open problems.

For each $X \in \{\Sigma, \Pi\}$, and any function $L : N \to R$ such that $\log L(n) = o(\log n)$,

1. strong-2XCA^*(1, \log n) - \bigcup_{l \leq \infty} weak-2ACA^*(k, L(n)) \neq \phi$ for each $X \in \{\Sigma, \Pi\}$ and each $l = 1, 2$ ?
2. m-2\Pi_1CA^*(1, \log n) - \bigcup_{l \leq \infty} m-2\Sigma_1CA^*(k, L(n)) \neq \phi$, and
   
   m-2\Sigma_1CA^*(1, \log n) - \bigcup_{l \leq \infty} m-2\Pi_1CA^*(k, L(n)) \neq \phi$ for each $l \geq 2$ and each $m \in \{\text{weak, strong}\}$ ?

Let $m \in \{\text{weak, strong}\}$, $X \in \{\Sigma, \Pi\}$, $l \geq 1$, and $L : N \to R$ be a function such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$.

3. What is a relationship between $m-2X\Pi_1CA^*(k, L(n))$ and $m-2X\Sigma_1CA^*(k, L(n))$ ?

4. $\text{strong-2\Pi}_l\text{CA}(i, L(n)) \subseteq \text{weak-2\Pi}_l\text{CA}(i, L(n))$ for each $i = 1, 2 ?$, and
   
   $\text{strong-2\Pi}_l\text{CA}(j, L(n)) \subsetneq \text{weak-2\Pi}_l\text{CA}(j, L(n))$ for each $j = 1, 2, 3 ?$

Let $\text{strong-2DCA}^*(k, L(n))$ denote the class of sets accepted by strongly $L(n)$ space-bounded two-way deterministic $k$-counter automata, let $\text{weak-2DCA}^*(k, L(n))$ denote the class of sets accepted by weakly $L(n)$ space-bounded two-way 1-inkdot deterministic $k$-counter automata, let $\text{weak-2DTM}^*(L(n))$ denote the class of sets accepted by weakly $L(n)$ space-bounded two-way deterministic Turing machines, and let $\text{weak-2DTM}^*(L(n))$ denote the class of sets accepted by weakly $L(n)$ space-bounded two-way 1-inkdot deterministic Turing machines. It is shown in [18] that

$m-2\text{DTM}^*(L(n)) = m-2\text{DTM}(L(n))$

for each $m \in \{\text{weak, strong}\}$, and any $L : N \to R$ such that $L(n) \geq \log \log n$ and $\log L(n) = o(\log n)$.

For each $m \in \{\text{weak, strong}\}$ and each $k \geq 1$, and any $L : N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$,

5. $m-2\text{DCA}^*(k, L(n)) = m-2\text{DCA}^*(k, L(n))$ ?

### References


