数理解析研究所講究録
906 巻 1995 年 64-72

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

徳山高専 義永常宏 (Tsunehiro YOSHINAGA) 山口大学工学部 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 *strong*-$2\Sigma_l$CA $(k, L(n))$ (*strong*-$2\Pi_l$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 $l-1$ alternations with the initial state existential (universal), let *weak*-$2\Sigma_l$CA$(k, L(n))$ (*weak*-$2\Pi_l$CA$(k, L(n))$) denote the class of sets accepted by waekly $L(n)$ space-bounded two-way alternating k-counter automata making at most $l-1$ alternations with the initial state existential (universal), let *strong*-$2\Sigma_l$CA$^*(k, L(n))$ (*strong*-$2\Pi_l$CA$^*(k, L(n))$) denote the class of sets accepted by strongly $L(n)$ space-bounded two-way 1-inkdot alternating k-counter automata making at most $l-1$ alternations with the initial state existential (universal), and let *weak*-$2\Sigma_l$CA$^*(k, L(n))$ (*weak*-$2\Pi_l$CA$^*(k, L(n))$) denote the class of sets accepted by weakly $L(n)$ space-bounded two-way 1-inkdot alternating k-counter automata making at most $l-1$ alternations with the initial state existential (universal). We denote by *strong*-$2\Sigma_l$TM$(L(n))$ (*strong*-$2\Pi_l$TM$(L(n))$) the class of sets accepted by strongly $L(n)$ space-bounded two-way alternating Turing machines making at most $l-1$ alternations with the initial state existential (universal), denote by *weak*-$2\Sigma_l$TM$(L(n))$ (*weak*-$2\Pi_l$TM$(L(n))$) the class of sets accepted by weakly $L(n)$ space-bounded two-way alternating Turing machines making at most $l-1$ alternations with the initial state existential (universal), denote by *strong*-$2\Sigma_l$TM$^*(L(n))$ (*strong*-$2\Pi_l$TM$^*(L(n))$) the class of sets accepted by strongly $L(n)$ space-bounded two-way 1-inkdot alternating Turing machines making at most $l-1$ alternations with the initial state existential (universal), and denote by *weak*-$2\Sigma_l$TM$^*(L(n))$ (*weak*-$2\Pi_l$TM$^*(L(n))$) the class of sets accepted by weakly $L(n)$ space-bounded two-way 1-inkdot alternating Turing machines making at most $l-1$ 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*-$2\Sigma_1$TM$^*(\log n)$ − *weak*-$2\Sigma_1$TM$(L(n)) \neq \phi$ and *strong*-$2\Pi_1$TM$^*(\log n)$ − *weak*-$2\Pi_1$TM$(L(n)) \neq \phi$. 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$^*(1, \log n)$ − $\bigcup_{1\leq k<\infty}$*weak*-$2\Sigma_1$CA$(k, L(n)) \neq \phi$ and *strong*-$2\Pi_1$CA$^*(1, \log n)$ − $\bigcup_{1\leq k<\infty}$*weak*-$2\Pi_1$CA$(k, L(n)) \neq \phi$. 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$(\log n)$ − *weak*-$2\Pi_l$TM$(L(n)) \neq \phi$ and *strong*-$2\Sigma_l$TM$(\log n)$ − *weak*-$2\Pi_l$TM$(L(n)) \neq \phi$. 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*-$2\Pi_l$CA$(1, \log n)$ − $\bigcup_{1\leq k<\infty}$*weak*-$2\Sigma_l$CA$(k, L(n)) \neq \phi$ and *strong*-$2\Sigma_l$CA$(1, \log n)$ − $\bigcup_{1\leq k<\infty}$*weak*-$2\Pi_l$CA$(k, L(n)) \neq \phi$. Section 5 investigates a relationship between m-$2\Sigma_1$CA$^*(k, L(n))$ and m-$2\Pi_1$CA$^*(k, L(n))$ 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$^*(k, L(n))$ is incomparable with m-$2\Pi_1$CA$^*(k, L(n))$. 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^i (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 "¢" and the right endmarker "\$" on the input tape, read the input tape right or left, and can enter an accepting state only when falling off the right endmarker \$. 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

$$\Sigma^* \times (N \cup \{0\}) \times S_M,$$

where Σ ($\not{c}, \$ \notin \Sigma$) 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 $I = (w, i, (q, (\alpha_1, \ldots, \alpha_k)))$ represent the input string and the input head position, respectively.[1] The third component $(q, (\alpha_1, \ldots, \alpha_k))$ 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 λ 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 π of the tree is labeled with an ID, $\ell(\pi)$,
2. if π is an internal node (a non-leaf) of the tree, $\ell(\pi)$ is universal and $\{I | \ell(\pi) \vdash_M I\} = \{I_1, I_2, \ldots, I_r\}$, then π has exactly r children $\rho_1, \rho_2, \ldots, \rho_r$ such that $\ell(\rho_i) = I_i$, and
3. if π is an internal node of the tree and $\ell(\pi)$ is existential, then π has exactly one child ρ such that $\ell(\pi) \vdash_M \ell(\rho)$.

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 1aca(k) denote a one-way alternating k-counter automaton.

Let $L : N \to R$ be a function, where R denotes the set of all nonnegative real numbers. For each x$\in$\{1,2\} and each $k \geq 1$, xaca(k) M 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 π 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 π 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 xaca(k) is denoted by *weak*-xaca($k, L(n)$) (*strong*-xaca($k, L(n)$)).

Let $T : N \to N$ be a function. For each $m \in \{weak, strong\}$, each x$\in$\{1,2\} and each $k \geq 1$, and any function $L : N \to R$, we say that an m-xaca($k, L(n)$) 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-1aca($k, L(n)$) M *operates in realtime* if $T(n) = n + 1$. For operating time, we are only interested in realtime in this paper.

For each $m \in \{weak, strong\}$ and each $k \geq 1$, and any function $L : N \to R$, an m-2aca($k, L(n)$) with 1 inkdot, denoted by m-2aca*($k, L(n)$), 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 action 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 $m \in \{weak, strong\}$, each $k \geq 1$ and each $l \geq 1$, and any function $L : N \to R$, we denote by m-2$\sigma_l$ca($k, L(n)$) (m-2$\pi_l$ca($k, L(n)$)) a m-2aca($k, L(n)$) making at most $l - 1$ alternations with the initial state existential (universal), denote by m-1$\sigma_l$ca($k, L(n)$) (m-1$\pi_l$ca($k, L(n)$)) a m-1aca($k, L(n)$) making at most $l - 1$ alternations with the initial state existential (universal), and denote by m-2$\sigma_l$ca*($k, L(n)$) (m-2$\pi_l$ca*($k, L(n)$)) a m-2$\sigma_l$ca($k, L(n)$) (m-2$\pi_l$ca($k, L(n)$)) with 1 inkdot.

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

m-xACA($k, L(n)$) = $\{L | L = T(M)$ for some m-xaca($k, L(n)$) $M\}$,
m-xΣ_lCA($k, L(n)$) = $\{L | L = T(M)$ for some m-x$\sigma_l$ca($k, L(n)$) $M\}$,
m-xΠ_lCA($k, L(n)$) = $\{L | L = T(M)$ for some m-x$\pi_l$ca($k, L(n)$) $M\}$,
weak-1Σ_lCA($k, L(n)$,real) = $\{L | L = T(M)$ for some *weak*-1$\sigma_l$ca($k, L(n)$) M operating in realtime$\}$,
weak-1Π_lCA($k, L(n)$,real) = $\{L | L = T(M)$ for some *weak*-1$\pi_l$ca($k, L(n)$) M operating in realtime$\}$,
m-2Σ_lCA*($k, L(n)$) = $\{L | L = T(M)$ for some m-2$\sigma_l$ca*($k, L(n)$) $M\}$, and
m-2Π_lCA*($k, L(n)$) = $\{L | L = T(M)$ for some m-2$\pi_l$ca*($k, L(n)$) $M\}$.

An alternating Turing machine (aTm) we consider in this paper has a read-only input tape with the left endmarker $\not{c}$ and the right endmarker \$, and a separate storage tape. (The reader is referred to [9,10] for the formal definition of aTm's.) For any $L : N \to R$, 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 aTm's.) For each $m \in \{weak, strong\}$, and any $L : N \to R$, we denote a m-2aTm($L(n)$) with 1-inkdot by m-2aTm*($L(n)$). (The reader is referred to [6-8,18] for formal definition of m-2aTm*($L(n)$).) For each $m \in \{weak, strong\}$ and each $l \geq 1$, and any function $L : N \to R$, we denote by m-2σ_lTm($L(n)$) (m-2π_lTm($L(n)$)) a m-2aTm($L(n)$) making at most $l - 1$ alternations with the initial state existential (universal), denote by m-1σ_lTm($L(n)$) (m-1π_lTm($L(n)$)) a m-1aTm($L(n)$) making at most $l - 1$ alternations with the initial state existential (universal), and denote by m-2σ_lTm*($L(n)$) (m-2π_lTm*($L(n)$)) a m-2σ_lTm($L(n)$) (m-2π_lTm($L(n)$)) with 1 inkdot.

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

[1] We note that $0 \leq i \leq |w| + 2$, where for any string v, $|v|$ denotes the length of v. "0", "1", "$|w| + 1$" and "$|w| + 2$" represent the positions of the left endmaker $\not{c}$, 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-xΣ_lTM$(L(n)) = \{L|L = T(M)$ for some m-xσ_lTm$(L(n))$ $M\}$,
m-xΠ_lTM$(L(n)) = \{L|L = T(M)$ for some m-xπ_lTm$(L(n))$ $M\}$,
m-2Σ_lTM$^*(L(n)) = \{L|L = T(M)$ for some m-2σ_lTm$^*(L(n))$ $M\}$, and
m-2Π_lTM$^*(L(n)) = \{L|L = T(M)$ for some m-2π_lTm$^*(L(n))$ $M\}$.

The following lemmas can be easily proved.

Lemma 2.1. For each $m \in \{weak, strong\}$, each x $\in \{1,2\}$ and each $l \geq 1$, and any function $L : N \rightarrow R$,
$\bigcup_{1\leq k<\infty} m$-xACA$(k, L(n)) \subseteq m$-xATM$(\log L(n))^2$ and $\bigcup_{1\leq k<\infty} m$-2ACA$^*(k, L(n)) \subseteq m$-2ATM$^*(\log L(n))$,
$\bigcup_{1\leq k<\infty} m$-x$\Sigma_lCA(k, L(n)) \subseteq m$-x$\Sigma_lTM(\log L(n))$ and $\bigcup_{1\leq k<\infty} m-2\Sigma_lCA^*(k, L(n)) \subseteq m-2\Sigma_lTM^*(\log L(n))$, and
$\bigcup_{1\leq k<\infty} m$-x$\Pi_lCA(k, L(n)) \subseteq m$-x$\Pi_lTM(\log L(n))$ and $\bigcup_{1\leq k<\infty} m-2\Pi_lCA^*(k, L(n)) \subseteq m-2\Pi_lTM^*(\log L(n))$.

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

strong-1ATM$(F(n))$ is the class of regular sets, and *weak*-2ATM$(G(n))$ 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 $\log G(n) = o(\log\log n)$,

strong-1ACA$(k, F(n))$ is the class of regular sets and *weak*-2ACA$(k, G(n))$ is the class of regular sets.

3 The power of 1 inkdot

This section investigates a relationship between the accepting powers of space-bonded 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 $L(n) = o(\log n)$,

strong-2Σ_1TM$^*(\log\log n)$ $-$ *weak*-2Σ_1TM$(L(n)) \neq \phi$,
strong-2Π_1TM$^*(\log\log n)$ $-$ *weak*-2Π_1TM$(L(n)) \neq \phi$, and
strong-2Π_3TM$^*(\log\log n)$ $-$ *weak*-2ATM$(L(n)) \neq \phi$.

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 $L_1 = \{B(1)\sharp B(2)\sharp\ldots\sharp B(n)cw_1cw_2c\ldots cw_rccw | n \geq 2\ \&\ r \geq 1\ \&\ w \in \{0,1\}^{\lceil\log n\rceil}\ \&\exists i(1 \leq i \leq r)[w = w_i]\}$, and
$L_2 = \{B(1)\sharp B(2)\sharp\ldots\sharp B(n)cw_1cw_2c\ldots cw_rccw | n \geq 2\ \&\ r \geq 1\ \&\ w \in \{0,1\}^{\lceil\log n\rceil}\ \&\ \forall i(1 \leq i \leq r)[w_i \in \{0,1\}^{\lceil\log n\rceil}\ \&\ w \neq w_i]\}$,
where for each positive integer $i \geq 1$, $B(i)$ denotes the string in $\{0,1\}^+$ that represents the integer i in binary notation (with no leading zeros). Then,
(1) $L_1 \in$ *strong*-2Σ_1CA$^*(1, \log n)$, and
(2) $L_2 \in$ *strong*-2Π_1CA$^*(1, \log n)$,
and for any function $L : N \rightarrow R$ such that $\log L(n) = o(\log n)$,
(3) $L_1 \notin \bigcup_{1\leq k<\infty}$ *weak*-2Σ_1CA$(k, L(n))$, and
(4) $L_2 \notin \bigcup_{1\leq k<\infty}$ *weak*-2Π_1CA$(k, L(n))$.
The proof of (1) (resp., (2)): We can construct a *strong*-2$\sigma_1$ca$^*(1, \log n)$ (resp., *strong*-2$\pi_1$ca$^*(1, \log n)$) M which acts as follows. Suppose that an input string
$¢y_1\sharp y_2\sharp\ldots\sharp y_n cw_1cw_2c\ldots cw_rccw\$$
(where $n \geq 2$, $r \geq 1$, 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 $Z^{\lceil\log n\rceil}$ 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 $Z^{\lceil\log n\rceil}$ stored in the counter that $|w| = \lceil\log n\rceil$ (resp., $|w| = \lceil\log n\rceil$ and $|w_i| = \lceil\log n\rceil$ for each i $(1 \leq i \leq r)$). 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 $Z^{\lceil\log n\rceil}$ in its counter that $|w_j| = \lceil\log n\rceil$, and then deterministically checks by using the inkdot as a pilot that $w = w_j$ (M deterministically checks by using the inkdot as a pilot that $w \neq w_j$). That is, for example, M stores Z^i $(1 \leq i \leq |w| = |w_j| = \lceil\log n\rceil)$ in its counter when M picks up the symbol $w_j(i)$[3] and by using Z^i 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 above 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 shoiwn in [6,8] that $L_1 \notin$ *weak*-2Σ_1TM$(L(n))$ and $L_2 \notin$ *weak*-2Π_1TM$(L(n))$ for any function $L(n) = o(\log n)$. From this result and lemma 2.1, (3) and (4) follow. □

Lemma 3.2. Let $L_3 = \{B(1)\sharp B(2)\sharp\ldots\sharp B(n)cw_1cw_2c\ldots cw_rccu_1cu_2c\ldots cu_{r'} | n \geq 2\ \&\ (r, r' \geq 1)\ \&\ \forall i(1 \leq i \leq r)\forall j(1 \leq$

[2]From now on, logarithms are base 2.
[3]For each string w, $w(i)$ denotes the i-th symbol (from the left) of w.

$j \le r')[w_i, u_j \in \{0,1\}^{\lceil \log n \rceil}]$ & $\forall i(1 \le i \le r)[\exists j(1 \le j \le r')[w_i = u_j]]\}$, and
$L_4 = \{B(1)\sharp B(2)\sharp \ldots \sharp B(n)cw_1cw_2c \ldots cw_rccu_1cu_2c \ldots cu_{r'} | n \ge 2$ & $(r, r' \ge 1)$ & $\forall i(1 \le i \le r)\forall j(1 \le j \le r')[w_i, u_j \in \{0,1\}^{\lceil \log n \rceil}]$ & $\exists i(1 \le i \le r)[\forall j(1 \le j \le r')[w_i \ne u_j]]\}$. Then,
(1) $L_3 \in$ *strong*-$2\Pi_3$CA*$(1, \log n)$, and
(2) $L_4 \in$ *strong*-$2\Sigma_3$CA*$(1, \log n)$,
and for any function $L : N \to R$ such that $\log L(n) = o(\log n)$,
(3) $L_3 \notin \bigcup_{1 \le k < \infty}$ *weak*-2ACA$(k, L(n))$,
(4) $L_4 \notin \bigcup_{1 \le k < \infty}$ *weak*-2ACA$(k, L(n))$,
(5) $L_3 \notin \bigcup_{1 \le k < \infty}$ *weak*-$2\Sigma_1$CA*$(k, L(n))$, and
(6) $L_4 \notin \bigcup_{1 \le k < \infty}$ *weak*-$2\Pi_1$CA*$(k, L(n))$.
The proof of (1) (resp., (2)): One can construct a *strong*-$2\pi_3$ca*$(1, \log n)$ (resp., *weak*-$2\sigma_3$ca*$(1, \log n)$) M which accepts L_3 (resp., L_4) as follows. Suppose that an input string

$$¢y_1\sharp y_2\sharp \ldots \sharp y_n cw_1cw_2c \ldots cw_rccu_1cu_2c \ldots cu_{r'}\$$$

(where $n \ge 2$, $(r, r' \ge 1)$, 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.) As in the proof of (1) and (2) of Lemma 3.1, M first stores $Z^{\lceil \log n \rceil}$ in its counter when $y_i = B(i)$ for each $1 \le i \le n$. M then checks by using $Z^{\lceil \log n \rceil}$ stored in the counter that $|w_1| = \ldots = |w_r| = |u_1| = \ldots = |u_{r'}| = \lceil \log n \rceil$. After that, M universally checks that for all i $(1 \le i \le r)$, $w_i = u_j$ for some j $(1 \le j \le r')$ (resp., M existentially checks that for some i $(1 \le i \le r)$, $w_i \ne u_j$ for all j $(1 \le j \le r')$). That is, for example, in order to check that $w_i = u_j$ for some j $(1 \le j \le r')$, M first branches, marks the symbol c just before w_i by the inkdot for each i $(1 \le i \le r)$, and then moves to the right to existentially choose u_j (resp., in order to check that $w_i \ne u_j$ for all j $(1 \le j \le r')$, M first guesses some i $(1 \le i \le r)$, marks the symbol c just before w_i by the inkdot, and then moves to the right to universally choose u_j). After that, by universally cheking that $w_i(l) = u_j(l)$ for all l $(1 \le l \le \lceil \log n \rceil)$ (resp., by existentially cheking that $w_i(l) \ne u_j(l)$ for some l $(1 \le l \le \lceil \log n \rceil)$), M can check that $w_i = u_j$ (resp., $w_i \ne u_j$). For this check, it is sufficient to use only one counter and use only its contents of length $\log n$. It will be obvious that L_3 (resp., L_4) $= T(M)$.
The proofs of (3), (5) and (6): It is shown in [8] that for any function $L(n) = o(\log n)$, $L_3 \notin$ *weak*-2ATM$(L(n))$, $L_3 \notin$ *weak*-$2\Sigma_1$TM*$(L(n))$, and $L_4 \notin$ *weak*-$2\Pi_1$TM*$(L(n))$. From this result and lemma 2.1, (3), (5) and (6) follow.
The proof of (4): For any function $L(n) = o(\log n)$, the proof of '$L_4 \notin$ *weak*-2ATM$(L(n))$' is similar to that of '$L_3 \notin$ *weak*-2ATM$(L(n))$' in [8], and so it is omitted here. From this result and lemma 2.1, (4) follows. □

From these two lemmas, we give the following theorem.

Theorem 3.1. For any function $L : N \to R$ such that $\log L(n) = o(\log n)$,
(1) *strong*-$2\Sigma_1$CA*$(1, \log n) - \bigcup_{1 \le k < \infty}$ *weak*-$2\Sigma_1$CA$(k, L(n)) \ne \phi$,
(2) *strong*-$2\Pi_1$CA*$(1, \log n) - \bigcup_{1 \le k < \infty}$ *weak*-$2\Pi_1$CA$(k, L(n)) \ne \phi$,
(3) *strong*-$2\Sigma_3$TM*$(1, \log n) - \bigcup_{1 \le k < \infty}$ *weak*-2ACA$(k, L(n)) \ne \phi$, and
(4) *strong*-$2\Pi_3$TM*$(1, \log n) - \bigcup_{1 \le k < \infty}$ *weak*-2ACA$(k, L(n)) \ne \phi$.

Corollary 3.1. For each $m \in \{strong, weak\}$, each $k \ge 1$ and each $l \ge 1$ $(l \ne 2)$, and any function $L : N \to R$ such that $L(n) \ge \log n$ and $\log L(n) = o(\log n)$,
m-$2\Sigma_l$CA$(k, L(n)) \subsetneq m$-$2\Sigma_l$CA*$(k, L(n))$ and m-$2\Pi_l$CA$(k, L(n)) \subsetneq m$-$2\Pi_l$CA*$(k, L(n))$.

4 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 infinte. 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 $\overset{i}{\sharp}$ is introduced. Let
$D_1 = \{0,1\}^*$ and $D_{i+1} = (D_i\{\overset{i}{\sharp}\})^* \cdot D_i$ for each $i \ge 1$,
$\exists D_1(u) = \{0,1\}^* - \{u\}$ and $\forall D_1(u) = \{u\}$, and for each $i \ge 1$,
$\exists D_{i+1}(u) = \{W_1 \overset{i}{\sharp} \ldots \overset{i}{\sharp} W_m \in D_{i+1} | \exists j(1 \le j \le m)[W_j \in \forall D_i(u)], m \ge 1\}$ and
$\forall D_{i+1}(u) = \{W_1 \overset{i}{\sharp} \ldots \overset{i}{\sharp} W_m \in D_{i+1} | \forall j(1 \le j \le m)[W_j \in \exists D_i(u)], m \ge 1\}$.
Now, let us define the following witness languages: for each $l \ge 2$,
$L_l^{\exists} = \bigcup\{\exists D_l(u) \cdot \{\sharp u \sharp B(n)^R \sharp B(n-1)^R \sharp \ldots \sharp B(1)^R\}\{\sharp\}^* | u \in \{0,1\}^{\lceil \log n \rceil}, n \ge 2\}$,[4] and

$L_l^{\forall} = \bigcup\{\forall D_l(u) \cdot \{\sharp u \sharp B(n)^R \sharp B(n-1)^R \sharp \ldots \sharp B(1)^R\}\{\sharp\}^* | u \in \{0,1\}^{\lceil \log n \rceil}, n \ge 2\}$. The following lemma is shown in [16].

Lemma 4.1. For each $l \ge 2$,
(1) $L_l^{\exists} \in$ *weak*-$1\Sigma_l$TM$(\log \log n)$ and $L_l^{\forall} \in$ *weak*-$1\Pi_l$TM$(\log \log n)$, and
(2) $L_l^{\exists} \in$ *strong*-$2\Sigma_l$TM$(\log \log n)$ and $L_l^{\forall} \in$ *strong*-$2\Pi_l$TM$(\log \log n)$,

[4] For each string w, w^R denotes the reversal of w

and for any function $L(n) = o(\log n)$,
(3) $L_l^{\exists} \notin$ *weak*-$2\Pi_l$TM$(L(n))$ and $L_l^{\forall} \notin$ *weak*-$2\Sigma_l$TM$(L(n))$.

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

Lemma 4.2. For each $l \geq 2$,
(1) $L_l^{\exists} \in$ *weak*-$1\Sigma_l$CA$(1, \log n$,real) and $L_l^{\forall} \in$ *weak*-$1\Pi_l$CA$(1, \log n$,real), and
(2) $L_l^{\exists} \in$ *strong*-$2\Sigma_l$CA$(1, \log n)$ and $L_l^{\forall} \in$ *strong*-$2\Pi_l$CA$(1, \log n)$,
and for any function $L : N \to R$ such that $\log L(n) = o(\log n)$,
(3) $L_l^{\exists} \notin \bigcup_{1 \leq k < \infty}$*weak*-$2\Pi_lCA(k, L(n))$ and $L_l^{\forall} \notin \bigcup_{1 \leq k < \infty}$*weak*-$2\Sigma_lCA(k, L(n))$.
The proof of (1): We prove (1) of this lemma by using induction for l (≥ 2).
1. (I) The following *weak*-$1\sigma_2$ca$(1, \log n)$ $M_2^{\exists}$ operating in realtime accepts the language
$L_2^{\exists} = \{w_1 \overset{1}{\sharp} w_2 \overset{1}{\sharp} \ldots \overset{1}{\sharp} w_m \sharp u \sharp B(n)^R \sharp B(n-1)^R \sharp \ldots \sharp B(1)^R \sharp^t | n \geq 2 \ \& \ m \geq 1 \ \& \ t \geq 1 \ \& \ u \in \{0,1\}^{\lceil \log n \rceil} \ \& \ \forall i (1 \leq i \leq m)[w_i \in D_1] \ \& \ \exists j (1 \leq j \leq m)[w_i = u]\}$.
Suppose that an input string

$$x = w_1 \overset{1}{\sharp} w_2 \overset{1}{\sharp} \ldots \overset{1}{\sharp} w_m \sharp u \sharp y_n \sharp y_{n-1} \sharp \ldots \sharp y_1 \sharp^t$$

(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 $M_2^{\exists}$. (Input strings in the form different from the above can easily be rejected by $M_2^{\exists}$.)

$M_2^{\exists}$ first existentially guesses some j, and runs to w_j. $M_2^{\exists}$ then makes a universal branch.

(A) In one branch, in order to check whether $w_j = u$, $M_2^{\exists}$ universally checks that $w_j(i) = u(i)$ for each i ($1 \leq i \leq |w_j|$). That is, to verify $w_j(i) = u(i)$, $M_2^{\exists}$ 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^i in the counter, and enters an accepting states only if $w_j(i) = u(i)$.

(B) In another branch, $M_2^{\exists}$ branches to check the following two points:

- **(a)** whether $|u| = |y_n|$, 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 $M_2^{\exists}$ enters an accepting state only if (a) is successfully checked. (b) above can be checked as follows. $M_2^{\exists}$ essentially uses the algorithm in [9,13]. By using univeral branches and only one counter, $M_2^{\exists}$ can check in a way described below whether $y_i = B(i)^R$ for each i ($1 \leq i \leq n$). $M_2^{\exists}$ compares y_{i+1} with y_i, and verifies that ${y_{i+1}}^R$ represents in binary notation (with no leading zeros) an integer which is one more than the integer represented by ${y_i}^R$ in binary notation (with no leading zeros). In doing so, $M_2^{\exists}$ will compare the j-th symbols of y_i and y_{i+1}, for all appropriate j. Observe that if ${y_{i+1}}^R$ is one more than ${y_i}^R$, then (i) $y_{i+1} = 0^m 1x$ and $y_i = 1^m 0x$, where x is a string (finishing with 1) over {0,1} and m is some non-negative integer, or (ii) $y_{i+1} = 0^m 1$ and $y_i = 1^m$, where m is some positive integer. Let C be the counter of $M_2^{\exists}$. For each j ($1 \leq j \leq |y_{i+1}|$), M stores the symbol $y_{i+1}(j)$ in its finite control and Z^j in C just after it has read the symbol $y_{i+1}(j)$, and makes a universal branch.

- In one branch, it compares $y_{i+1}(j)$ with the symbol $y_i(j)$ using Z^j 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_{i+1}. If the symbol 1 has already occurred, then $y_{i+1}(j)$ and $y_i(j)$ should be the same; otherwise, $y_i(j)$ and $y_{i+1}(j)$ 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 to C in order to store Z^{j+1} in C.

In this way, $M_2^{\exists}$ can check that ${y_{i+1}}^R$ 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 $y_n \sharp y_{n-1} \sharp \ldots \sharp y_1$ is such a string that $y_i = B(i)^R$ for each i ($1 \leq i \leq n$), then the length of y_n ($= B(n)^R$) is equal to $\lceil \log n \rceil$, and thus the length of C is bounded by $\log n$. Furthermore, it is clear that $M_2^{\exists}$ operates in realtime.
(II) $L_2^{\forall} = \{w_1 \overset{1}{\sharp} w_2 \overset{1}{\sharp} \ldots \overset{1}{\sharp} w_m \sharp u \sharp B(n)^R \sharp B(n-1)^R \sharp \ldots \sharp B(1)^R \sharp^t | n \geq 2 \ \& \ m \geq 1 \ \& \ t \geq 1 \ \& \ u \in \{0,1\}^{\lceil \log n \rceil} \ \& \ \forall i (1 \leq i \leq m)[w_i \in D_1 \ \& \ w_i \neq u]\}$ is accepted by a realtime *weak*-$1\pi_2$ca$(1, \log n)$ $M_2^{\forall}$ as follows. Suppose that an input string

$$x = w_1 \overset{1}{\sharp} w_2 \overset{1}{\sharp} \ldots \overset{1}{\sharp} w_m \sharp u \sharp y_n \sharp y_{n-1} \sharp \ldots \sharp y_1 \sharp^t$$

(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 $M_2^{\forall}$. (Input strings in the form different from the above can easily be rejected by $M_2^{\forall}$.)

$M_2^{\forall}$ 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, $M_2^{\forall}$ continues the action above until it reads the first $\sharp$, and then makes a univresal branch to check the following two points:

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

(B) In another branch, $M_2^{\forall}$ 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 (I).
2. Assume that assertion (1) of this lemma holds for $L_i^{\exists}$ and $L_i^{\forall}$ ($i = 3,4,\ldots,l-1$). We shall prove assertion (1) of the lemma holds for $L_l^{\exists}$ and $L_l^{\forall}$, too.

(I) An input string x in $L_l^{\exists}$ has the form $x = WS$ with W in $\exists D_l(u)$, $W = W_1 \overset{l-1}{\sharp} W_2 \overset{l-1}{\sharp} \ldots \overset{l-1}{\sharp} W_m$ with W_i in $\forall D_{l-1}(u)$ for some i ($1 \le i \le m$), and $S = \sharp u \sharp B(n)^R \sharp \ldots \sharp B(1)^R \sharp^t$, where u is in $\{0,1\}^{\lceil \log n \rceil}$, $t \ge 1$, $m \ge 1$ and $n \ge 2$. By the assumption above, there is a realtime *weak*-$1\pi_{l-1}$ca$(1,\log n)$ $M_{l-1}^{\forall}$ which accepts W_iS iff W_i is in $\forall D_{l-1}(u)$. $L_l^{\exists}$ is accepted by a realtime *weak*-$1\sigma_l$ca$(1,\log n)$ $M_l^{\exists}$ acts as follows. Suppose that an input string

$$x = W_1 \overset{l-1}{\sharp} W_2 \overset{l-1}{\sharp} \ldots \overset{l-1}{\sharp} W_m \sharp u \sharp y_n \sharp y_{n-1} \sharp \ldots \sharp y_1 \sharp^t$$

(where $n \ge 2$, $m \ge 1$, $t \ge 1$ and W_i's are all in $\{0,1,\overset{1}{\sharp},\overset{2}{\sharp},\ldots,\overset{l-2}{\sharp}\}^+$, and y_j's are all in $\{0,1\}^+$) is presented to $M_l^{\exists}$. (Input strings in the form different from the above can easily be rejected by $M_l^{\exists}$.) $M_l^{\exists}$ first guesses some i and runs on x to W_i. $M_l^{\exists}$ then enters a universal state, and acts just like $M_{l-1}^{\forall}$ above, but ignores any symbols between the next $\overset{l-1}{\sharp}$ (just after W_i) and the first $\sharp$.

(II) An input string x in $L_l^{\forall}$ has the form $x = WS$ with W in $\forall D_l(u)$, $W = W_1 \overset{l-1}{\sharp} W_2 \overset{l-1}{\sharp} \ldots \overset{l-1}{\sharp} W_m$ with W_i in $\exists D_{l-1}(u)$ for all i ($1 \le i \le m$), and $S = \sharp u \sharp B(n)^R \sharp \ldots \sharp B(1)^R \sharp^t$, where u is in $\{0,1\}^{\lceil \log n \rceil}$, $t \ge 1$, $m \ge 1$ and $n \ge 2$. By the assumption above, there is a realtime *weak*-$1\sigma_{l-1}$ca$(1,\log n)$ $M_{l-1}^{\exists}$ which accepts W_iS iff W_i is in $\exists D_{l-1}(u)$. There is a realtime *weak*-$1\pi_l$ca$(1,\log n)$ $M_l^{\forall}$ which accepts $L_l^{\forall}$ as follows. Suppose that an input string x described in (I) above is presented to $M_l^{\forall}$. $M_l^{\forall}$ moves on x while making a universal branch at the first symbol of each w_i ($1 \le i \le m$).

(a) In one branch, $M_l^{\forall}$ continues the action above until it reaches the first $\sharp$. After that, $M_l^{\forall}$ runs to the right endmarker \$, and enters an accepting state.

(b) In another branch, $M_l^{\forall}$ enters an existential state, and acts just like $M_{l-1}^{\exists}$ above, but ignores all the segments between the next $\overset{l-1}{\sharp}$ and the first $\sharp$.

Clearly, the lengthes of the counters of $M_l^{\exists}$ and $M_l^{\forall}$ are bounded by $\lceil \log n \rceil$, because those used in the computation are basically equal to the lengthes of the counters of $M_2^{\exists}$ and $M_2^{\forall}$ when they enter accepting states. $M_2^{\exists}$ and $M_2^{\forall}$ on accepted inputs use no more than $\lceil \log n \rceil$ space, which is shown as in 1 above.
The proof of (2): It is shown in [13] that the language $\{B(1)\sharp B(2)\sharp \ldots \sharp B(n) | n \ge 1\}$ is accepted by a strongly $\log n$ space-bounded two-way deterministic 1-counter automaton. For each $l \ge 2$, $L_l^{\exists}$ (resp., $L_l^{\forall}$) can be accepted by *strong*-$2\sigma_l$ca$(1,\log n)$ (resp., *strong*-$2\pi_l$ca$(1,\log n)$) M as follows. M begins by examining whether the suffix of a given input is of the form $B(n)^R \sharp B(n-1)^R \sharp \ldots \sharp B(1)^R$ $(= (B(1)\sharp B(2)\sharp \ldots \sharp B(n))^R)$ in a way as in [13]. If this examination is successful and M stores $Z^{\lceil \log n \rceil}$ in its counter, then M 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 $L(n) = o(\log n)$, $L_l^{\exists}$ and $L_l^{\forall}$ are not in *weak*-$2\Pi_l$TM$(L(n))$ and *weak*-$2\Sigma_l$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 $l \ge 2$, and any function $L : N \to R$ such that $\log L(n) = o(\log n)$,
(1) *weak*-$1\Sigma_l$CA$(1,\log n$,real$)$ $\cap$ *strong*-$2\Sigma_l$CA$(1,\log n)$ $-$ $\bigcup_{1 \le k < \infty}$*weak*-$2\Pi_lCA(k, L(n)) \ne \phi$, and
(2) *weak*-$1\Pi_l$CA$(1,\log n$,real$)$ $\cap$ *strong*-$2\Pi_l$CA$(1,\log n)$ $-$ $\bigcup_{1 \le k < \infty}$*weak*-$2\Sigma_lCA(k, L(n)) \ne \phi$.

Corollary 4.1. For each $l \ge 2$, each $k \ge 1$ and each $m \in \{weak, strong\}$, and any function $L : N \to R$ such that $L(n) \ge \log n$ and $\log L(n) = o(\log n)$,
(1) m-$2\Sigma_l$CA$(k, L(n))$ is incomparable with m-$2\Pi_l$CA$(k, L(n))$,
(2) *weak*-$1\Sigma_l$CA$(k, L(n))$ is incomparable with *weak*-$1\Pi_l$CA$(k, L(n))$, and
(3) *weak*-$1\Sigma_l$CA$(k, L(n), real)$ is incomparable with *weak*-$1\Pi_l$CA$(k, L(n), real)$.

Corollary 4.2. For each $l \ge 1$, each $k \ge 1$, each $m \in \{weak, strong\}$ and each X,Y$\in\{\Sigma, \Pi\}$, and any function $L : N \to R$ such that $L(n) \ge \log n$ and $\log L(n) = o(\log n)$,
(1) m-2X$_l$CA$(k, L(n)) \subsetneq m$-2Y$_{l+1}$CA$(k, L(n))$,
(2) *weak*-1X$_l$CA$(k, L(n)) \subsetneq$ *weak*-1Y$_{l+1}$CA$(k, L(n))$, and
(3) *weak*-1X$_l$CA$(k, L(n)$,real$) \subsetneq$ *weak*-1Y$_{l+1}$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$\in\{\Sigma, \Pi\}$, and any function $L : N \to R$ such that $L(n) \ge \log n$ and $\log L(n) = o(\log n)$,
strong-2X$_2$CA$(1,\log n)$ $-$ $\bigcup_{1 \le k < \infty}$*weak*-1ACA$(k, L(n)) \ne \phi$.
proof. Let $L_5 = \{B(1)\sharp B(2)\sharp \ldots \sharp B(n) 2wcw_1cw_2c \ldots cw_r | n \ge 2 \ \& \ r \ge 1 \ \& \ w \in \{0,1\}^{\lceil \log n \rceil} \ \& \ \forall i(1 \le i \le r)[w_i \in \{0,1\}^+] \ \& \ \exists j(1 \le j \le r)[w = w_j]\}$, and $L_6 = \{B(1)\sharp B(2)\sharp \ldots \sharp B(n) 2wcw_1cw_2c \ldots cw_r | n \ge 2 \ \& \ r \ge 1 \ \& \ w \in \{0,1\}^{\lceil \log n \rceil} \ \& \ \forall i(1 \le i \le r)[w_i \in \{0,1\}^{\lceil \log n \rceil} \ \& \ w \ne w_i]\}$. Then, (1) $L_5 \in$ *strong*-$2\Sigma_2$CA$(1,\log n)$ and (2) $L_5 \notin$ *weak*-1ACA$(k, L(n))$ are essentially proved in [13]. (3) $L_6 \in$ *strong*-$2\Pi_2$CA$(1,\log n)$ and (4) $L_6 \notin$ *weak*-1ACA$(1, L(n))$ 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 $L: N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$, *weak*-$1\Sigma_l$CA$(k, L(n)) \subsetneq$ *weak*-$2\Sigma_l$CA$(k, L(n))$, and *weak*-$1\Pi_l$CA$(k, L(n)) \subsetneq$ *weak*-$2\Pi_l$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 $L: N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$,

weak-2DCA$(4, \log n) - \bigcup_{1 \leq k < \infty}$*strong*-2ACA$(k, L(n)) \neq \phi$, and
weak-$2\Sigma_1$CA$(3, \log n) - \bigcup_{1 \leq k < \infty}$*strong*-2ACA$(k, L(n)) \neq \phi$.

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 $L: N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$,
(1) *strong*-$2\Sigma_l$CA$(i, L(n)) \subsetneq$ *weak*-$2\Sigma_l$CA$(i, L(n))$, and
(2) *strong*-$2\Pi_l$CA$(j, L(n)) \subsetneq$ *weak*-$2\Pi_l$CA$(j, L(n))$.

5 An alternation hierarchy on the first level with 1 inkdot

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

strong-$2\Pi_3$TM*$(\log\log n)$ − *weak*-$2\Sigma_1$TM*$(L(n)) \neq \phi$, and
strong-$2\Sigma_3$TM*$(\log\log n)$ − *weak*-$2\Pi_1$TM*$(L(n)) \neq \phi$.

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 $L: N \to R$ such that $\log L(n) = o(\log n)$,

strong-$2\Pi_3$CA*$(1, \log n) - \bigcup_{1 \leq k < \infty}$*weak*-$2\Sigma_1$CA*$(k, L(n)) \neq \phi$, and
strong-$2\Sigma_3$CA*$(1, \log n) - \bigcup_{1 \leq k < \infty}$*weak*-$2\Pi_1$CA*$(k, L(n)) \neq \phi$.

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

Lemma 5.1. Let $L_7 = \{B(1)\sharp B(2)\sharp \ldots \sharp B(n) c w_{11} c w_{12} c \ldots c w_{1r_1} c c u_1 c w_{21} c w_{22} c \ldots c w_{2r_2} c c u_2 | n \geq 2 \;\&\; (r_1, r_2 \geq 1) \;\&\; (u_1, u_2 \in \{0,1\}^{\lceil \log n \rceil}) \;\&\; \forall i (1 \leq i \leq 2)[\forall j (1 \leq j \leq r_i)[w_{ij} \in \{0,1\}^{\lceil \log n \rceil} \;\&\; u_i \neq w_{ij}]]\}$, and
$L_8 = \{w_{11} c w_{12} c \ldots c w_{1r_1} c c u_1 c w_{21} c w_{22} c \ldots c w_{2r_2} c c u_2 c B(n)^R B(n-1)^R \ldots B(1)^R | n \geq 2 \;\&\; (r_1, r_2 \geq 1) \;\&\; (u_1, u_2 \in \{0,1\}^+) \;\&\; \forall i (1 \leq i \leq 2)[\forall j (1 \leq j \leq r_i)[w_{ij} \in \{0,1\}^+]] \;\&\; \exists i (1 \leq i \leq 2)[\exists j (1 \leq j \leq r_i)[u_i = w_{ij} \;\&\; |u_i| = |w_{ij}| = \lceil \log n \rceil]]\}$. Then,
(1) $L_7 \in$ *strong*-$2\Pi_1$CA*$(1, \log n)$,
(2) $L_8 \in$ *strong*-$2\Sigma_1$CA*$(1, \log n)$,
(3) $L_7 \in$ *strong*-$2\Pi_2$CA$(1, \log n)$ and $L_7 \in$ *weak*-$1\Pi_2$CA$(1, \log n$,real), and
(4) $L_8 \in$ *strong*-$2\Sigma_2$CA$(1, \log n)$ and $L_8 \in$ *weak*-$1\Sigma_2$CA$(1, \log n$,real),
and for any function $L: N \to R$ such that $\log L(n) = o(\log n)$,
(5) $L_7 \notin \bigcup_{1 \leq k < \infty}$*weak*-$2\Sigma_1$CA*$(k, L(n))$, and
(6) $L_8 \notin \bigcup_{1 \leq k < \infty}$*weak*-$2\Pi_1$CA*$(k, L(n))$.
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 exersice.
The proof of (5): Suppose that there exists a *weak*-$2\sigma_1$ca*$(k, L(n))$ M accepting L_7 for some $k \geq 1$, where $\log L(n) = o(\log n)$. For each $n \geq 2$, let
$V(n) = \{B(1)\sharp B(2)\sharp \ldots \sharp B(n) c y_1 c c u_1 c y_2 c c u_2 | (y_1, y_2 \in W(n)) \;\&\; (u_1, u_2 \in \{0,1\}^{\lceil \log n \rceil})\}$, where
$W(n) = \{w_1 c w_2 c \ldots c w_{2^{\lceil \log n \rceil}} | \forall i (1 \leq i \leq 2^{\lceil \log n \rceil})[w_i \in \{0,1\}^{\lceil \log n \rceil}]\}$.
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 $l(n) = O(n \log n)$. Let $C(n)$ denote the set of all possible storage states of M when the length of each counter of M in the computation is bounded by $L(l(n))$, and let $u(n)$ be the number of elements of $C(n)$. Then, $u(n) = tL(l(n))^k$, where t denotes the number of states of the finite control of M. Let x, y be any two strings in $W(n)$. We say that x and y are *M-equivalent*, if for each pair of storage states q, $q' \in C(n)$, and d, $d' \in \{$right,left$\}$,

there exists an $L(l(n))$ space-bounded computation in which M enters x in q with 1 inkdot from the d edge, and afterwards exits x in q' from d' edge without consuming the inkdot on the way.
$\Longleftrightarrow$
there exists an $L(l(n))$ space-bounded computation in which M enters y in q with 1 inkdot from the d edge, and afterwards exits y in q' from d' edge without consuming the inkdot on the way.

Clearly there are at most $e(n) = O(s^{u(n) \times u(n)})$ M-equivalent classes of strings in $W(n)$, where s is a constant. For each $y = w_1 c \ldots c w_{2^{\lceil \log n \rceil}}$ in $W(n)$, let $b(y) = \{w | \exists i (1 \leq i \leq 2^{\lceil \log n \rceil})[w = w_i]\}$. Furthermore, for each $n \geq 2$, let $R(n) = \{b(y) | y \in W(n)\}$. Then, $|R(n)| = 2^{2^{\lceil \log n \rceil}} - 1$. From the assumption that $\log L(n) = o(\log n)$ and from the fact that $l(n) = O(n \log n)$, it follows that $|R(n)| > e(n)$ for n large enough, and so there must exist two M-equivalent elements y, y' in $W(n)$ such that

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

$$z = B(1)\sharp B(2)\sharp \dots \sharp B(n) c y_1 ccucy_2 ccu,$$

where $y_1 = y_2 = y$. As is easily seen, z is in L_7, and so there exists an $L(l(n))$ space-bounded accepting computation of M on z. We denote the accepting computation by *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 *comp*(z). Let z' be the string obtained from z by replacing y_i $(= y)$ by y'. From *comp*(z), we can construct an $L(l(n))$ space-bounded accepting computation of M on z'. This is a contradiction, since z' is not in L_7.

The proof of (6): The proof is similar to that of (5) of this lemma. Suppose, to the contrary, that there exists a *weak*-2πca$^*(k, L(n))$ M accepting L_8 for some $k \geq 1$, where $\log L(n) = o(\log n)$. For each $n \geq 2$, let

$$V(n) = \{y_1 ccu_1 cy_2 ccu_2 cB(n)^R \sharp B(n-1)^R \sharp \dots \sharp B(1)^R | (y_1, y_2 \in W(n)) \ \&\ (u_1, u_2 \in \{0,1\}^{\lceil \log n \rceil})\},$$

and $W(n)$, $l(n)$, $C(n)$ and $u(n)$ be defined as in the proof of (5) of this lemma. We consider the computations of M on the strings in $V(n)$. Furthermore, for each $y \in W(n)$, let $b(y)$ be the set described in the proof of (5). Let x, y be any two strings in $W(n)$. We say that x and y are *M-equivalent*, if for each pair of storage states q, $q' \in C(n)$, and d, $d' \in$ {right,left},

(1) there exists an $L(l(n))$ space-bounded computation in which M enters x in q with 1 inkdot from the d edge, and afterwards exits x in q' from d' edge without consuming the inkdot on the way.

$\Longleftrightarrow$

there exists an $L(l(n))$ space-bounded computation in which M enters y in q with 1 inkdot from the d edge, and afterwards exits y in q' from d' edge without consuming the inkdot on the way.

(2) there exists a computation in which M enters x in q with 1 inkdot from the d edge, and afterwards exits x in q' from d' edge using some counters of length larger than $L(l(n))$ without consuming the inkdot on the way.

$\Longleftrightarrow$

there exists a computation in which M enters y in q with 1 inkdot from the d edge, and afterwards exits y in q' from d' edge using some counters of length larger than $L(l(n))$ without consuming the inkdot on the way.

(3) there exists a computation in which M enters x in q with 1 inkdot from the d edge, and never exits x without consuming the inkdot.

$\Longleftrightarrow$

there exists a computation in which M enters y in q with 1 inkdot from the d edge, and never exits y without consuming the inkdot.

For large n, and two M-equivalent elements y, y' in $W(n)$, and $u \in b(y') - b(y)$, let

$$z = y_1 ccucy_2 ccucB(n)^R \sharp B(n-1)^R \sharp \dots \sharp B(1)^R,$$

where $y_1 = y_2 = y$. Clearly, z is not in L_8, and so z is never accepted by M, that is, there is at least one computation path $I_M(z) = I_0 \vdash_M I_1 \vdash_M I_2 \dots \vdash_M I_m$ $(m \geq 1)$ of M on z 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 *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 *comp*(z). Let z' be the string obtained from z by replacing y_i $(= y)$ by y'. From *comp*(z), we can easily construct a computation path of M on z' with the properties (P1), (P2) and (P3). Thus, z' is rejected by M. This is a contradiction, because z' is in L_8. □

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) *strong*-$2\Pi_1$CA$^*(1, \log n) - \bigcup_{1 \leq k < \infty}$ *weak*-$2\Sigma_1$CA$^*(k, L(n)) \neq \phi$,
(2) *strong*-$2\Sigma_1$CA$^*(1, \log n) - \bigcup_{1 \leq k < \infty}$ *weak*-$2\Pi_1$CA$^*(k, L(n)) \neq \phi$,
(3) *strong*-$2\Pi_2$CA$(1, \log n) \cap$ *weak*-$1\Pi_2$CA$(1, \log n$,real$) - \bigcup_{1 \leq k < \infty}$ *weak*-$2\Sigma_1$CA$^*(k, L(n)) \neq \phi$, and
(4) *strong*-$2\Sigma_2$CA$(1, \log n) \cap$ *weak*-$1\Sigma_2$CA$(1, \log n$,real$) - \bigcup_{1 \leq k < \infty}$ *weak*-$2\Pi_1$CA$^*(k, L(n)) \neq \phi$.

Corollary 5.1. For each $k \geq 1$, each $m \in \{strong, weak\}$, and each X $\in \{\Sigma, \Pi\}$, and any function $L : N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$,
(1) m-$2\Pi_1$CA$^*(k, L(n)) \subsetneq m$-2X$_2$CA$^*(k, L(n))$, and
(2) m-$2\Sigma_1$CA$^*(k, L(n)) \subsetneq m$-2X$_2$CA$^*(k, L(n))$.

For the standard 2amca model, a relationship between m-$2\Pi_1$CA$(k, L(n))$ and m-$2\Sigma_1$CA$(k, L(n))$ is unknown for each $k \geq 1$, each $m \in \{strong, weak\}$, 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 inkdot 2amca model, we have the following result.

Corollary 5.2. For each $k \geq 1$ and each $m \in \{strong, weak\}$, and any function $L : N \to R$ such that $L(n) \geq \log n$ and $\log L(n) = o(\log n)$, m-$2\Pi_1 CA^*(k, L(n))$ is incomparable with m-$2\Sigma_1 CA^*(k, L(n))$.

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) $strong$-$2X_l CA^*(1, \log n) - \bigcup_{1 \leq k < \infty} weak$-$2ACA(k, L(n)) \neq \phi$ for each $X \in \{\Sigma, \Pi\}$ and each $l = 1, 2$?

(2) m-$2\Pi_l CA^*(1, \log n) - \bigcup_{1 \leq k < \infty} m$-$2\Sigma_l CA^*(k, L(n)) \neq \phi$, and
m-$2\Sigma_l CA^*(1, \log n) - \bigcup_{1 \leq k < \infty} m$-$2\Pi_l CA^*(k, L(n)) \neq \phi$ for each $l \geq 2$ and each $m \in \{weak, strong\}$?

Let $m \in \{weak, strong\}$, $X \in \{\Sigma, \Pi\}$, $l \geq 1$, 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-$2X_{l+1} CA(k, L(n))$ and m-$2X_l CA^*(k, L(n))$?

(4) $strong$-$2\Sigma_l CA(i, L(n)) \subsetneq weak$-$2\Sigma_l CA(i, L(n))$ for each $i = 1, 2$?, and
$strong$-$2\Pi_l CA(j, L(n)) \subsetneq weak$-$2\Pi_l CA(j, L(n))$ for each $j = 1, 2, 3$?

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

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

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

For each $m \in \{weak, strong\}$ 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-2DCA$^*(k, L(n)) = m$-2DCA$(k, L(n))$?

References

[1] A.K. Chandra, D.C. Kozen and L.J. Stockmeyer, "Alternation", *J. ACM*, vol. 28, no. 1, pp. 114–133, 1981.

[2] R.E. Ladner, R.J. Liptom and L.J. Stockmeyer, "Alternating pushdown automata", in *Proc. 19th IEEE Symp. on Found. Comput. Sci.*, (Ann Arbor, MI.), pp. 92–106, 1978.

[3] K.N. King, "Alternating multihead finite automata", in *Theoret. Comput. Sci.*, vol. 61, pp. 149–174, 1985.

[4] P.C. Fisher, A.R. Meyer and A.L. Rosenberg, "Counter machines and counter languages", *Math. Systems Theory*, vol. 2, pp. 265–283, 1968.

[5] R. Book and S. Ginsburg, "Multi-stack-counter languages", *Math. Systems Theory*, vol. 6, pp. 37–48, 1972.

[6] K. Inoue, A. Ito, and I. Takanami, "A relationship between nondeterministic Turing machines and 1-inkdot Turing machines with small space", *Inform. Process. Lett.*, vol. 43, pp. 225–227, 1992.

[7] K. Inoue, A. Ito, I. Takanami and T. Yoshinaga, "A note on multi-inkdot nondeterministic Turing machines with small space", *Inform. Process. Lett.*, vol. 48, pp. 285–288, 1993.

[8] K. Inoue, A. Ito and I. Takanami, "On 1-inkdot alternating Turing machines with small space", *Theoret. Comput. Sci.*, vol. 127, pp. 171–179, 1994.

[9] K. Inoue, I. Takanami and R. Vollmar, "Alternating on-line Turing machines with only universal states and small space bounds", *Theoret. Comput. Sci.*, vol. 41, pp. 331–339, 1985.

[10] A. Ito, K. Inoue and I. Takanami, "A note on alternating Turing machines using small space", *Trans. IECE Japan*, E70, 10, pp. 990–996, 1987.

[11] K. Inoue, A. Ito and I. Takanami, "A note on real-time one-way alternating multicounter machines", *Theoret. Comput. Sci.*, vol. 88, pp. 287–296, 1991.

[12] T. Yoshinaga, K. Inoue and I. Takanami, "Hierarchical properties of realtime one-way altrnating multi-stack-counter automata", *Trans. IEICE Japan*, E77–A, 4, pp. 621–629, 1994.

[13] T. Yoshinaga and K. Inoue, "A note on alternating multi-counter automata with small space", *Tec. Report of IEICE*, COMP94–36, pp. 1–10, 1994.

[14] J.H. Chang, O.H. Ibarra, B. Ravikumar and L. Berman, "Some observations concerning alternating Turing machines using small space", *Inform. Process. Lett.*, vol. 25, pp. 1–9, 1987.

[15] M. Liśkiewicz and R. Reischuk, "Separating the lower levels of the sublogarithmic space hierarchy", *STACS*, pp. 17–27, 1993.

[16] B. von Braunmühl, R. Gengler and R. Rettinger, "The alternation hierarchy for sublogarithmic space is infinite", *Comput. Complexity*, 3, pp. 207–230, 1993.

[17] K. Iwama, "ASPACE($o(\log\log n)$) is regular", *SIAM J. Comput.*, vol. 22, pp. 136–146, 1993.

[18] D. Ranjan, R. Chang and J. Hartmanis, "Space bounded computations : review and new separation results, *Thoret. Comput. Sci.*, vol. 80, pp. 289–302, 1991.