Note

On 1-inkdot alternating Turing machines with small space

Katsushi Inoue, Akira Ito and Itsuo Takanami

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

Communicated by M.A. Harrison
Received November 1992
Revised June 1993

Abstract


This paper investigates the accepting powers of nondeterministic and alternating 1-inkdot Turing machines using small space. Let NTM (ATM, UTM) denote a nondeterministic Turing machine (alternating Turing machine, alternating Turing machine with only universal states). For each \( X \in \{N, A, U\} \), let STRONG-\(X\)SPACE\([L(n)]\) (STRONG-\(X\)SPACE*\([L(n)]\)) denote the class of languages accepted by strongly \( L(n) \) space-bounded XTM (1-inkdot XTM), and let WEAK-\(X\)SPACE\([L(n)]\) (WEAK-\(X\)SPACE*\([L(n)]\)) denote the class of languages accepted by weakly \( L(n) \) space-bounded XTM (1-inkdot XTM). We show that

1. Introduction and preliminaries

We consider the Turing machine model with a two-way, read-only input tape and a separate two-way, read-write worktape [4]. Recently, Ranjan et al. [11] introduced

\[
\begin{align*}
(1) \text{STRONG-ASPACE}^*\left[\log \log n\right] &\neq \text{WEAK-ASPACE}\left[o(\log n)\right], \\
(2) \text{STRONG-USPACE}^*\left[\log \log n\right] &\neq \text{WEAK-USPACE}\left[o(\log n)\right], \\
(3) \text{STRONG-ASPACE}^*\left[\log \log n\right] &\neq \text{WEAK-NSPACE}^*\left[o(\log n)\right], \text{ and} \\
(4) \text{STRONG-ASPACE}^*\left[\log \log n\right] &\neq \text{WEAK-USPACE}^*\left[o(\log n)\right].
\end{align*}
\]
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 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 and worktape symbols and the presence of the inkdot on the currently scanned tape-cell. The action consists of moving the heads and making appropriate changes on worktape cells (using the finite control). In addition, the inkdot may be used to mark the currently scanned cell on the input tape if it has not been used already.

Let STRONG-NSPACE\([L(n)]\) (STRONG-NSPACE*\([L(n)]\)) denote the class of languages accepted by strongly \(L(n)\) space-bounded nondeterministic Turing machines (1-inkdot Turing machines), and let WEAK-NSPACE\([L(n)]\) (WEAK-NSPACE*\([L(n)]\)) denote the class of languages accepted by weakly \(L(n)\) space-bounded nondeterministic Turing machines (1-inkdot Turing machines).

Ranjan et al. [11] left the following open problem:

\[
\text{STRONG-NSPACE}[\log \log n] = \text{STRONG-NSPACE}^* [\log \log n]?
\]

This problem was solved by Geffert [3], who showed that there is a language which is in STRONG-NSPACE*\([\log \log n]\), but not in STRONG-NSPACE[\(o(\log n)\)]. Recently, Inoue et al. [5] strengthened this result, and showed that there is a language which is in STRONG-NSPACE*\([\log \log n]\), but not in WEAK-NSPACE[\(o(\log n)\)].

A natural question is whether a similar result also holds for alternating Turing machines [1, 2, 7–9, 11, 12]. In this paper, we challenge this problem. We assume that the reader is familiar with the basic concepts and terminology concerning nondeterministic and alternating Turing machines and computational complexity. (If necessary, see [1, 2, 7–9, 11, 12]).

Our alternating Turing machine (ATM) has a two-way, read-only input tape (with the left endmarker \(\varepsilon\) and the right endmarker \(\$\)) and a separate two-way, read–write worktape. A 1-inkdot ATM is an alternating version of the 1-inkdot Turing machine stated above.

An instantaneous description (ID) of an ATM is of the form \((x, i, (q, a, j))\). The first and second components \(x\) and \(i\) represent the input string and the input head position, respectively. The third component \((q, a, j)\) is a storage state which represents a combination of the state of the finite control, the nonblank contents of the worktape, and the worktape head position.

Let \(L(n)\) be a function and \(M\) be an ATM. A computation tree of \(M\) (on some input string) is \(L(n)\) space-bounded if all nodes of the tree are labeled with IDs using at most \(L(n)\) worktape cells. We say that \(M\) is weakly \(L(n)\) space-bounded if for every input string \(x\) of length \(n\), \(n \geq 1\), that is accepted by \(M\), there exists an \(L(n)\) space-bounded accepting computation tree of \(M\) on \(x\). We say that \(M\) is strongly \(L(n)\) space-bounded if for every input string of length \(n\) (accepted by \(M\) or not), \(M\) never uses more than \(L(n)\) worktape cells. A strongly (weakly) \(L(n)\) space-bounded 1-inkdot ATM is defined similarly.
We denote by UT M an ATM with only universal states, i.e., with no existential state. Note that a nondeterministic Turing machine, denoted by NT M, is an ATM which has no universal state.

For any function \( L(n) \) and each \( X \in \{ A, U, N \} \), let \( \text{WEAK-XTSPACE}[L(n)] \) (\( \text{WEAK-XTSPACE}^*[L(n)] \)) denote the class of languages accepted by weakly \( L(n) \) space-bounded XTMs (1-inkdot XTMs) and let \( \text{STRONG-XTSPACE}[L(n)] \) (\( \text{STRONG-XTSPACE}^*[L(n)] \)) denote the class of languages accepted by strongly \( L(n) \) space-bounded XTMs (1-inkdot XTMs). In the next section, we show that

1. \( \text{STRONG-ASPACE}^*[\log \log n] \subseteq \text{WEAK-ASPACE}[o(\log n)] \subseteq \Phi \),
2. \( \text{STRONG-USPACE}^*[\log \log n] \subseteq \text{WEAK-USPACE}[o(\log n)] \subseteq \Phi \),
3. \( \text{STRONG-ASPACE}^*[\log \log n] \subseteq \text{WEAK-NSPACE}^*[o(\log n)] \subseteq \Phi \), and
4. \( \text{STRONG-ASPACE}^*[\log \log n] \subseteq \text{WEAK-USPACE}^*[o(\log n)] \subseteq \Phi \).

In the last section, we discuss our results, and give several open problems.

2. Results

For any alphabet \( \Sigma \) and any \( n \geq 1 \), \( \Sigma^n \) denotes the set of strings of length \( n \) from \( \Sigma \). For any string \( w \), \( |w| \) denotes the length of \( w \), and for any set \( S \), \( |S| \) denotes the number of elements of \( S \). Throughout this paper, we assume that logarithms are base 2.

Our first result is the following theorem.

**Theorem 2.1.** \( \text{STRONG-ASPACE}^*[\log \log n] \subseteq \text{WEAK-ASPACE}[o(\log n)] \neq \Phi \).

**Proof.** Let \( T_1 = \{ B(1) \neq B(2) \neq \cdots \neq B(n) \# w_1 c w_2 c \cdots w_k c u_1 c u_2 c \cdots u_k \in \{0, 1, \#, \}^+ | (n \geq 2) \& (k, k' \geq 1) \& \forall i (1 \leq i \leq k) \forall j (1 \leq j \leq k') [w_i, u_j \in \{0, 1\}^{[\log n]}] \& \forall i (1 \leq i \leq k) [\exists j (1 \leq j \leq k') [w_i = u_j]] \}, \) 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).

We first show that \( T_1 \) is in \( \text{STRONG-ASPACE}^*[\log \log n] \). We consider a strongly \( \log \log n \) space-bounded 1-inkdot ATM \( M \) which acts as follows. Suppose that an input string

\[
y_1 \neq y_2 \neq \cdots \neq y_n \# w_1 c w_2 c \cdots w_k c u_1 c u_2 c \cdots u_k \in S
\]

(where \( n \geq 2, k, k' \geq 1 \), and \( y_i, \), \( w_j, u_m \) are all in \( \{0, 1\}^+ \)) is presented to \( M \). (Input strings in the form different from the above can easily be rejected by \( M \).) By using the well-known technique (see [4, Problem 10.2]), \( M \) first marks off \( \log \log n \) worktape cells when \( y_i = B(i) \) for each \( 1 \leq i \leq n \). (Of course, \( M \) enters a rejecting state if \( y_i \neq B(i) \) for some \( 1 \leq i \leq n \).) \( M \) then checks, by using \( \log \log n \) worktape cells, that \( |w_1| = \cdots = |w_k| = |u_1| = \cdots = |u_k| = \log n \). After that, \( M \) universally checks that for all \( i (1 \leq i \leq k) \), \( w_i = u_j \) for some \( j (1 \leq j \leq k') \). That is, for example, in order to check that \( w_i = u_j \) for some \( j (1 \leq j \leq k') \), \( M \) first marks the symbol \( c \) just before \( w_i \) by the inkdot, and then moves to the right to existentially choose \( u_j \). After that, by universally
checking that the rth symbol of \( w_i \) is equal to the rth symbol of \( u_j \) for all \( r \) \( (1 \leq r \leq \lceil \log n \rceil) \), \( M \) can check whether \( w_i = u_j \). (For this check, \( \log \log n \) worktape cells are sufficient.) \( M \) enters an accepting state only if these checks are all successful. It will be obvious that \( M \) accepts the language \( T_1 \).

We next show that \( T_1 \) is not in \( \text{WEAK-ASPACE}[o(\log n)] \). Suppose, to the contrary, that there exists a weakly \( L(n) \) space-bounded ATM \( M \) accepting \( T_1 \), where \( L(n) = o(\log n) \). We assume, without loss of generality, that when \( M \) accepts \( x \in T_1 \), it enters an accepting state on the right endmarker \$. For \( n \geq 2 \), let

\[
V(n) = \{ B(1) \# B(2) \# \cdots \# B(n) \} \forall y \in W(n),
\]

where \( W(n) = \{ w_1 c w_2 c \cdots c w_f(n) \} \forall i (1 \leq i \leq f(n) \equiv 2^{\lceil \log n \rceil}) \). \( w_i \in \{ 0, 1 \}^{\lceil \log n \rceil} \}. \) We consider the computations of \( M \) on the strings in \( V(n) \). Clearly, each \( x \in V(n) \) is in \( T_1 \), and so \( x \) is accepted by \( M \). For each \( y \in W(n) \), let

\[
x(y) \equiv B(1) \# B(2) \# \cdots \# B(n) \}
\]

\( t(y) \equiv \) a fixed accepting computation tree of \( M \) on \( x(y) \). (Fix \( t(y) \) that has minimal number of nodes, and, among them, a tree that is minimal according to some lexicographical ordering.)

Note that we can assume that for any \( t(y) \),

(i) each node on any path of \( t(y) \) differs from each other, and
(ii) if two nodes of \( t(y) \) are labeled by the same IDs, then the subtrees (of \( t(y) \)) with these nodes as the roots are the same.

This implies that no computation path in \( t(y) \) enters the same ID twice, i.e., we do not have to worry about cycles in \( t(y) \).

For each ID \( I \),

- whose storage state component is \( s \),
- and that crosses the boundary "B" between the left part \#B(1) \# \cdots \# B(n) \} c y of \( \#x(y) \$ and the right part \} c y \$ from left to right, let \( M_s(s) \) denote the set of storage state components of all IDs in the subtree of \( t(y) \) with root \( I \) such that the input head crosses the boundary \( B \) back from right to left for the first time. (We note that, by (ii), two IDs crossing \( B \) from left to right and having the same storage state component \( s \) must also have the same set \( M_s(s) \). On the other hand, if, for some \( s \), no node of \( t(y) \) is labeled by ID \( I \) with the storage state component \( s \) and crossing \( B \) from left to right, then \( M_s(s) \) is undefined. As a special case, \( M_s(s) \) equals to the empty set, if no computation path of \( t(y) \) beginning with \( I \) returns back to cross the boundary \( B \) from right to left.)

Let \( a(n) \) be the length of each string in \( V(n) \). Then \( a(n) = O(n \log n) \). We say that two strings \( x(y), x(y') \) in \( V(n) \) are \( M \)-equivalent if \( M_s(s) = M_{s'}(s') \) for each storage state \( s = (q, \sigma, k) \) of \( M \) such that \( |s| \leq L(a(n)) \). Clearly, \( M \)-equivalence is an equivalence relation on \( V(n) \). Let \( e(n) \) be the number of \( M \)-equivalence classes of strings in \( V(n) \). Then \( e(n) = O(c^{u(n) \times u(n)}) \) for some constant \( c > 0 \), where \( u(n) = rL(a(n))tL(a(n)) \) (where \( r \) and \( t \) are the number of states and storage tape symbols of \( M \), respectively). For each
$y = w_1cw_2c \ldots cw_{r_{\text{fin}}}$ in $W(n)$, let $b(y) = \{w \in \{0, 1\}^{\log n} \mid \exists i \ (1 \leq i \leq f(n)) [w = w_i] \}$. Furthermore, for each $n \geq 2$, let $R(n) = \{ b(y) \mid y \in W(n) \}$. Intuitively, $R(n)$ denotes the family of all the different sets of strings from $\{0, 1\}^{\log n}$ occurring in elements of $W(n)$. Clearly, $|R(n)| = 2^{\log n}$ and from the fact that $a(n) = O(n \log n)$ and from the assumption that $L(n) = o(\log n)$, it follows that, for $n$ large enough, $|R(n)| = e(n)$. For such a large $n$, there must exist two different strings $y, y'$ in $W(n)$ such that (1) $b(y) \neq b(y')$ (thus, say, $b(y) - b(y') \neq 0$) and (2) $x(y)$ and $x(y')$ are $M$-equivalent. Consider the string $z = B(1) \# \ldots \# B(n)ccyccy'$. We can easily construct an $L(z)$ space-bounded accepting computation tree of $M$ on $z$ from the fixed accepting computation trees $t(y)$ and $t(y')$. Thus, $z$ is also accepted by $M$. This is a contradiction, because $z$ is not in $T_1$. This completes the proof of "$T_1 \notin \text{WEAK-ASPACE}[o(\log n)]". 

Our second result is the following theorem.

**Theorem 2.2.** STRONG-USPACE*[$\log \log n$] = WEAK-USPACE*[o(\log n)] ≠ \emptyset.

**Proof.** Let $T_2 = \{ B(1) \# B(2) \# \ldots \# B(n) \mid w_1cw_2c \ldots cw_kccw \in \{0, 1, \#, \}^+ \mid n \geq 2 \ & k \geq 1 \ & w \in \{0, 1\}^{\log n} \ & \forall i \ (1 \leq i \leq k) \ [w_i \in \{0, 1\}^{\log n} \ & w_i \neq w] \}$. We first show that $T_2$ is in STRONG-USPACE*[log \log n]. We consider a strongly log \log n space-bounded 1-inkdot UTM $M$ which acts as follows. Consider the input $y_1 \# y_2 \# \ldots \# y_n \# B(n) \# cccw$. If $y_i = B(i)$ for each $1 \leq i \leq n$, then $M$ can mark off log \log n worktape cells and check if $|w_1| = \ldots = |w_k| = |w| = [\log n]$ (for details, see the proof of Theorem 2.1). After that, $M$ universally checks that $w_i \neq w$ for all $i \ (1 \leq i \leq k)$. For example, by marking the symbol $c$ just before $w_i$ by the inkdot and by using log \log n worktape cells as a counter, $M$ can check that $w_i \neq w$. It will be obvious that $M$ accepts the language $T_2$.

The proof of "$T_2 \notin \text{WEAK-USPACE[O(\log n)]}" is very similar to that of Theorem 1 in [5], and so omitted here (the full proof is given in [6]). 

Our third result is the following theorem.

**Theorem 2.3.** STRONG-ASPACE*[log \log n] = WEAK-NSPACE*[o(\log n)] ≠ \emptyset.

**Proof.** Let $T_1$ be the language stated in the proof of Theorem 2.1. As shown in the proof of Theorem 2.1, $T_1$ is in STRONG-ASPACE*[log \log n]. We now show that $T_1 \notin \text{WEAK-NSPACE*[O(\log n)]}$. Suppose, to the contrary, that there exists a weakly $L(n)$ space-bounded 1-inkdot NTM $M$ accepting $T_1$, where $L(n) = o(\log n)$. For each $n \geq 2$, let $V(n)$ and $W(n)$ be the sets defined in the proof of Theorem 2.1. We consider the computations of $M$ on the strings in $V(n)$. Clearly, each $x$ in $V(n)$ is in $T_1$, and so $x$ is accepted by $M$. For each $y \in W(n)$, let

$x(y) \equiv B(1) \# B(2) \# \ldots \# B(n)ccyccy \in V(n)$

$\text{comp}(y) \equiv \text{a fixed accepting computation of } M \text{ on } x(y)$. 

We can assume, without loss of generality, that \( \text{comp}(y) \) is loop-free. For each \( y \in W(n) \), let

\[
\text{cross}(y) \equiv \text{the crossing sequence when } M \text{ crosses the boundary } B \text{ between the left part } \notin B(1) \# B(2) \# \cdots \# B(n)cy \text{ of } \notin x(y)\& \text{ and the right part } ccy\& \text{ according to } \text{comp}(y), \text{i.e. the sequence of pairs } (s, \text{flag}) \text{ when } M \text{ crosses the boundary } B \text{ according to } \text{comp}(y) \text{ (where } s \text{ is a storage state of } M, \text{ and } \text{flag}=0 \text{ if } M \text{ has not yet used the inkdot, and } \text{flag}=1 \text{ otherwise}).
\]

Let \( a(n) \) be the length of each string in \( V(n) \). Then \( a(n) = O(n \log n) \). For each \( n \geq 2 \), let \( C(n) = \{ \text{cross}(y) \mid y \in W(n) \} \). Clearly, \( |C(n)| = O(u(n)! \) where \( u(n) = 2rL(a(n))^{kL(a(n))} \) (where \( r \) and \( k \) are the number of states and storage tape symbols of \( M \), respectively).

As in the proof of Theorem 2.1, let \( R(n) \) denote the family of all the different sets of strings from \( \{0, 1\}^{\log n} \) occurring in elements of \( W(n) \). From the assumption that \( L(n) = o(\log n) \) and from the fact that \( a(n) = O(n \log n) \) and \( |R(n)| = \Omega(2^n) \), it follows that for \( n \) large enough, \( |R(n)| > |C(n)| \). For such a large \( n \), there must exist two different elements \( y \) and \( y' \) in \( W(n) \) such that (1) \( \text{cross}(y) = \text{cross}(y') \) and (2) there is a string \( w \in \{0, 1\}^{\log n} \) which occurs in \( y \) but not in \( y' \). Applying now "cut-and-paste" technique (as in the proof of Theorem 2.1), one can obtain an input which is not in \( T_1 \), but is accepted by \( M \), a contradiction. This completes the proof of "\( T_1 \notin \text{WEAK-NSPACE}^*[O(\log n)] \)".

Our last result is the following theorem.

**Theorem 2.4.** \( \text{STRONG-ASPACE}^*[\log \log n] \) \( \text{WEAK-USPACE}^*[O(\log n)] \) \( \neq \emptyset \).

**Proof.** Let \( T_3 = \{ B(1) \# B(2) \# \cdots \# B(n) \# w_1 \# w_2 \# \cdots \# w_k \in \{0, 1, c, \# \}^* \mid (n \geq 2) \& (k, k' \geq 1) \& \forall i (1 \leq i \leq k) \forall j (1 \leq j \leq k) [w_i, u_j \in \{0, 1\}^{\log n}] \& \exists i (1 \leq i \leq k) [\forall j (1 \leq j \leq k') [w_i \neq u_j]] \} \).

We first show that \( T_3 \) is in \( \text{STRONG-ASPACE}^*[\log \log n] \). We consider a strongly \( \log \log n \) space-bounded 1-inkdot ATM \( M \) which acts as follows. Consider the input \( \notin y_1 \# y_2 \# \cdots \# y_n \# w_1 \# w_2 \# \cdots \# w_k \$. As in the proof of Theorem 2.1, if \( y_i = B(t) \) for each \( 1 \leq t \leq n \), then \( M \) can mark off \( \log n \) worktape cells and check if \( |w_1| = \cdots = |w_k| = |u_1| = \cdots = |u_k| = \lceil \log n \rceil \). After that, \( M \) existentially chooses some \( i (1 \leq i \leq k) \), and marks the symbol \( c \) just before \( w_i \) by the inkdot. Then, by using this inkdot as a pilot, \( M \) universally checks that \( w_i \neq u_j \), for all \( j (1 \leq j \leq k') \). That is, for example, in order to check that \( w_i \neq u_j \), \( M \) has only to existentially pick up some symbol, say the \( r \)th symbol, of \( u_j \) and check that the \( r \)th symbol of \( u_j \) is different from the \( r \)th symbol of \( w_i \). It will be obvious that \( M \) accepts the language \( T_3 \).

The proof of "\( T_3 \notin \text{WEAK-USPACE}^*[O(\log n)] \)" is similar to that of Theorem 2.3, and so left to the reader (the full proof is given in [6]).

From Theorems 2.1–2.4, we get the following corollary.
Corollary 2.5. For any $X \in \{\text{STRONG, WEAK}\}$, any $Y \in \{A, U, N\}$, and any $L(n)$ such that $L(n) \geq \log \log n$ and $L(n) = o(\log n)$,

1. $X$-YSPACE[$L(n)$] $\subseteq$ $X$-YSPACE*[$L(n)$],
2. $X$-NSPACE*[$L(n)$] $\not\subseteq$ $X$-ASPACE*[$L(n)$], and
3. $X$-USPACE*[$L(n)$] $\not\subseteq$ $X$-ASPACE*[$L(n)$].

3. Conclusions

This paper actually shows more than presented in the formulation of the theorems. The 1-inkdot ATM presented in the proof of Theorem 2.1 is actually $\Pi_3$-alternating, which gives a new corollary:

(a) $\text{STRONG-}\Pi_3$-ASPACE*[$\log \log n$] – $\text{WEAK-ASPACE}[o(\log n)] \neq \emptyset$, and hence
(b) $\text{STRONG-}\Sigma_4$-ASPACE*[$\log \log n$] – $\text{WEAK-ASPACE}[o(\log n)] \neq \emptyset$,
(c) $\text{STRONG-}\Pi_k$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Pi_k$-ASPACE*[$o(\log n)$] $\neq \emptyset$, for each $k \geq 3$, and
(d) $\text{STRONG-}\Sigma_k$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Sigma_k$-ASPACE*[$o(\log n)$] $\neq \emptyset$, for each $k \geq 4$.

($\Sigma_k/\Pi_k$-ASPACE denote the classes of languages accepted by ATMs making at most $k-1$ alternations with the initial state existential/universal, respectively. $\Sigma_k/\Pi_k$-ASPACE* correspond to their inkdot variants.) Note that Theorem 2.2 and the result in [5] show

\begin{align*}
\text{STRONG-}\Pi_1$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Pi_1$-ASPACE*[$o(\log n)$] $\neq \emptyset \quad \text{and} \\
\text{STRONG-}\Sigma_1$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Sigma_1$-ASPACE*[$o(\log n)$] $\neq \emptyset,
\end{align*}

respectively. Natural open questions are:

1. What is the minimum of $k$'s ($k = 1, 2$) such that
\begin{align*}
\text{STRONG-}\Pi_k$-ASPACE*[$\log \log n$] – $\text{WEAK-ASPACE}[o(\log n)] \neq \emptyset?
\end{align*}
2. What is the minimum of $k$'s ($k = 1, 2, 3$) such that
\begin{align*}
\text{STRONG-}\Sigma_k$-ASPACE*[$\log \log n$] – $\text{WEAK-ASPACE}[o(\log n)] \neq \emptyset?
\end{align*}

3. $\text{STRONG-}\Pi_2$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Pi_2$-ASPACE*[$o(\log n)$] $\neq \emptyset$?
4. $\text{STRONG-}\Pi_k$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Sigma_k$-ASPACE*[$o(\log n)$] $\neq \emptyset$ for each $k = 2, 3$.?

Theorems 2.3 and 2.4 show that the alternating hierarchy of 1-inkdot ATMs $\Delta_k/\Sigma_k$ does not collapse to the first level $\Sigma_1/\Pi_1$. More precisely, the proof of Theorem 2.3 shows

$\text{STRONG-}\Pi_3$-ASPACE*[$\log \log n$] – $\text{WEAK-}\Sigma_3$-ASPACE*[$o(\log n)$] $\neq \emptyset$. 

and hence

\[ \text{STRONG-}\Sigma_4\text{-ASPACE}*[\log \log n] - \text{WEAK-}\Sigma_1\text{-ASPACE}*[o(\log n)] \neq \emptyset, \]

and the proof of Theorem 2.4 shows

\[ \text{STRONG-}\Sigma_2\text{-ASPACE}*[\log \log n] - \text{WEAK-}\Pi_1\text{-ASPACE}*[o(\log n)] \neq \emptyset, \]

and hence

\[ \text{STRONG-}\Pi_2\text{-ASPACE}*[\log \log n] - \text{WEAK-}\Pi_1\text{-ASPACE}*[o(\log n)] \neq \emptyset. \]

We leave the following problems open:

5. What is the minimum of \( k \)'s \((k = 1,2)\) such that

\[ \text{STRONG-}\Pi_k\text{-ASPACE}*[\log \log n] - \text{WEAK-}\Sigma_k\text{-ASPACE}*[o(\log n)] \neq \emptyset? \]

6. What is the minimum of \( k \)'s \((k = 1,2)\) such that

\[ \text{STRONG-}\Sigma_k\text{-ASPACE}*[\log \log n] - \text{WEAK-}\Pi_1\text{-ASPACE}*[o(\log n)] \neq \emptyset? \]

7. \( \text{STRONG-}X_{k+1}\text{-ASPACE}*[\log \log n] - \text{WEAK-}X_k\text{-ASPACE}*[o(\log n)] \neq \emptyset \)

for each \( X \in \{\Pi, \Sigma\} \) and each \( k \geq 1 \).

It will also be interesting to investigate properties of a multi-inkdot Turing machine which is able to use a fixed number of inkdots on the input tape. We will present several properties of this machine in a future paper.

Acknowledgment

We sincerely thank the two anonymous referees whose comments have improved the presentation of the paper and enabled us to discuss our results in the last section.

References

I-inkdot alternating Turing machines with small space


