Necessary spaces for seven-way four-dimensional Turing machines to simulate four-dimensional one-marker automata

Makoto Nagatomo, Shinnosuke Yano, Makoto Sakamoto, Satoshi Ikeda, and Hiroshi Furutani
Faculty of Engineering, University of Miyazaki,
1-1 Gakuen Kibanadai Nishi, Miyazaki, Miyazaki 889-2192, Japan
E-mail: je.suis.makoto@gmail.com, shinchandx@ezweb.ne.jp, sakamoto@cs.miyazaki-u.ac.jp,
bisu@cs.miyazaki-u.ac.jp, furutani@cs.miyazaki-u.ac.jp

Takao Ito and Tsutomu Ito
Institute of Engineering, Hiroshima University, 4-1, Kagamiyama 1-chome
Higashi-Hiroshima, Hiroshima 739-8527, Japan
E-mail: itotakao@horoshima-u.ac.jp, 0va71-2538f211n@ezweb.ne.jp

Yasuo Uchida
Department of Business Administration, Ube National College of Technology, Tokiwadai
Ube, Yamaguchi 755-8555, Japan
E-mail:uchida@ube-k.ac.jp

Tsunehiro Yoshinaga
Department of Computer Science & Electronic Engineering,
National Institute of Technology, Tokuyama College, Gakuendai
Shunan, Yamaguchi 745-8585, Japan
E-mail:yosinaga@tokuyama.ac.jp

Abstract

We think that recently, due to the advances in many application areas such as motion image processing, computer animation, and so on, it is very useful for analyzing computational complexity of multi-dimensional information processing to explicate the properties of four-dimensional automata, i.e., three-dimensional automata with the time axis. As far as we know, there is no investigation about four-dimensional automata. Then, in 2002, we first introduced four-dimensional finite automata in the world. In 2003, we investigated four-dimensional alternating Turing machines. In 2015, we show the sufficient spaces for four-dimensional Turing machines to simulate four-dimensional one-marker automata. In this paper, we continue the investigations, and deal with the necessary spaces for four-dimensional Turing machines to simulate four-dimensional one-marker automata.

Keywords: computational complexity, finite automaton, lower bounds, marker, simulation, Turing machine.

1. Introduction

An improvement of picture recognizability of the finite automaton is the reason why the marker automaton was introduced. That is, a two-dimensional one-marker automaton can recognize connected pictures. This automaton has been widely investigated in the two- or three-dimensional case [2]. A multi-marker automaton is a finite automaton which keeps marks as 'pebbles' in the finite control, and cannot rewrite any input symbols but can make marks on its input with the restriction that only a bounded number of these marks can exist at any given time[1].

As is well known among the researchers of automata theory, one-dimensional one-marker automata are equivalent to ordinary finite state automata. In other words, there is no need of working space usage for one-way Turing machines to simulate one-marker automata, as well as finite state automata.

In the two-dimensional case, the following facts are known : the necessary and sufficient space for three-way two-dimensional deterministic Turing machines *TR*2-*DTM*'s to simulate two-dimensional deterministic (nondeterministic) finite automata 2-*DFA*'s (2-*NFA*'s) is *m*log*m* (m^2) and the corresponding space for three-way two-dimensional nondeterministic Turing machines *TR*2-*NTM*'s is *m* (*m*), whereas the necessary and sufficient space for three-way two-dimensional deterministic Turing machines *TR*2-*DTM*'s to simulate two-dimensional deterministic (nondeterministic) one-marker automata 2-DMA_1's (2-NMA_1's) is $2^{m\log m}$ (2^{m^2}) and the corresponding space for *TR*2-*NTM*'s is *m*log*m* (m^2), where *m* is the number of columns of two-dimensional rectangular input tapes.

In the three-dimensional case, the following facts are known : the necessary and sufficient space for five-way three-dimensional deterministic Turing machines *FV3-DTM*'s to simulate three-dimensional deterministic (nondeterministic) finite automata 3-*DFA*'s (3-*NFA*'s) is $m^2\log m$ (m^3) and the corresponding space for five-way three-dimensional nondeterministic Turing machines *FV3-NTM*'s is m^2 (m^2), whereas the necessary and sufficient space for five-way three-dimensional deterministic Turing machines *FV3-DTM*'s to simulate three-dimensional deterministic (nondeterministic) one-marker automata 3-DMA_1's (3-NMA_1's) is $2^{lm\log lm}$ ($2^{l^2m^2}$) and the corresponding space for *FV3-NTM*'s is *lm*log*lm* (l^2m^2), where *l*(*m*) is the number of rows (columns) on each plane of three-dimensional rectangular input tapes.

In the four-dimensional case, we showed the sufficient spaces for four-dimensional Turing machines to simulate four-dimensional one-marker automata [3]. In this paper, we continue the investigations, and deal with the necessary spaces for four-dimensional Turing machines to simulate four-dimensional one-marker automata.

2. Preliminaries

An ordinary finite automaton cannot rewrite any symbols on input tape, but a marker automaton can make a mark on the input tape. We can think of the mark as a 'pebble' that *M* puts down in a specified position. If *M* has already put down the mark, and wants to put it down elsewhere, *M* must first go to the position of the mark and pick it up. Formally, we define it as follows.

Definition 2.1. A *four-dimensional nondeterministic one-marker automaton* (4-NMA_1) is defined by the 6-tuple $M = (Q, q_0, F, \Sigma, \{+, -\}, \delta)$, where

(1) Q is a finite set of *states* ;

(2) $q_0 \in Q$ is the *initial state* ;

(3) $F \subseteq Q$ is the set of *accepting states* ;

(4) Σ is a finite input *alphabet* ($\# \notin \Sigma$ is the *boundary symbol*);

(5) $\{+,-\}$ is the pair of *signs of presence and absence of the marker* ; and

(6) $\delta : (Q \times \{+,-\}) \times ((\Sigma \cup \{\#\}) \times \{+,-\}) \mapsto \times 2^{Q\times\{+,-\}} \times ((\Sigma \cup \{\#\}) \times \{+,-\}) \times \{\text{east, west, south, north, up, down, future, past, no move}\}$ is the *next-move function*, satisfying the following : For any q, $q' \in Q$, any a, $a' \in \Sigma$, any u, u', v, $v' \in \{+,-\}$, and any $d \in$ {east, west, south, north, up, down, future, past, no move}, if $((q', u'), (a', v'), d) \in \delta((q,v), (a,v))$ then $a = a'$ and $(u, v, u', v') \in \{(+,-,+,-), (+,-,-,+), (-,+,-,+), (-,+,+,-), (-,-,-,-)\}$.

We call a pair (q, u) in $Q \times \{+,-\}$ an *extended state*, representing the situation that *M* holds or does not hold the marker in the finite control according to the sign $u = +$ or $u = -$, respectively. A pair (a, v) in $\Sigma \times \{+,-\}$ represents an input tape cell on which the marker exists or does not exist according to the sign $u = +$ or $u = -$, respectively.

Therefore, the restrictions on δ imply the following conditions. (i) When holding the marker, *M* can put it down or keep on holding. (ii) When not holding the marker, and ① if the marker exists on the current cell, *M* can pick it up or leave it there, or ② if the marker does not exist on the current cell, *M* cannot create a new marker any more.

Definition 2.2. Let Σ be the input alphabet of 4-NMA_1 *M*. An *extended input tape* $\tilde{x}$ of *M* is any four-dimensional tape over $\Sigma \times \{+,-\}$ such that for some $x \in \Sigma^{(4)}$,

(i) for each $j(1 \le j \le 4)$, $l_j(\tilde{x}) = l_j(x)$,

(ii) for each $i_1\big(1 \le i_1 \le l_1(\tilde{x})\big)$, $i_2\big(1 \le i_2 \le l_2(\tilde{x})\big)$, $i_3\big(1 \le i_3 \le l_3(\tilde{x})\big)$, and $i_4\big(1 \le i_4 \le l_4(\tilde{x})\big)$, $\tilde{x}(i_1, i_2, i_3, i_4) = x\big((i_1, i_2, i_3, i_4), u\big)$ for some $u \in \{+,-\}$.

Definition 2.3. A *configuration* of 4-NMA_1 $M = (Q, q_0, F, \Sigma, \{+, -\}, \delta)$ is a pair of an element of $\big((\Sigma \cup \{\#\}) \times \{+,-\}\big)^{(4)}$ and an element of $C_M = (\mathbf{N} \cup \{0\})^{(4)} \times (Q \times \{+,-\})$. The first component of a configuration $c = (\tilde{x}, ((i_1, i_2, i_3, i_4), (q, u)))$ represents the extended input tape of *M*. The second component (i_1, i_2, i_3, i_4) of

c represents the input head position. The third component (q, u) of c represents the extended state. An element of C_M is called a *semi-configuration* of *M*. If q is the state associated with configuration *c*, then *c* is said to be an *accepting configuration* if q is an accepting state. The *initial configuration* of *M* on input *x* is $I_M(x) = (x^-, ((1,1,1,1), (q_0, +)))$, where x^- is the special extended input tape of M such that $x^-(i_1, i_2, i_3, i_4) = (x(i_1, i_2, i_3, i_4), -)$ for each i_1, i_2, i_3, i_4 $(1 \le i_1 \le l_1(x^-), 1 \le i_2 \le l_2(x^-), 1 \le i_3 \le l_3(x^-),\ 1 \le i_4 \le l_4(x^-))$. If *M* moves deterministically, we call *M* a four-dimensional deterministic one-marker automaton (4-*DMA*$_1$).

Definition 2.4. Given a 4-*NMA*$_1$ *M* = (Q, q_0, F, Σ, {+, -}, δ), we write $c \vdash_M c'$ and say c' is a *successor* of *c* if configuration c' follows from configuration c in one step of *M*, according to the transition rules δ. $\vdash_M^*$ denotes the reflexive transitive closure of $\vdash_M$. The relation $\vdash_M$ is not necessarily single-valued, because δ is not. A *computation path* of *M* on x is a sequence $c_0 \vdash_M c_1 \vdash_M \ldots \vdash_M c_n (n \ge 0)$, where $c_0 = I_M(x)$. An *accepting computation path* of *M* on *x* is a computation path of *M* on *x* which ends in an accepting configuration. We say that *M accepts x* if there is an accepting computation path of *M* on input *x*.

Let $L(l, m, n) : \mathbf{N}^3 \mapsto \mathbf{R}$ be a function. A seven-way four-dimensional Turing machine *M* is said to be $L(l, m, n)$ *space-bounded* if for each l, m, n≥ 1 and for each x with $l_1(x) = l$, $l_2(x) = m$, and $l_3(x) = n$, if x is accepted by *M*, then there is an accepting computation path of *M* on *x* in which *M* uses no more than $L(l, m, n)$ cells of the storage tape. We denote an $L(l, m, n)$ space-bounded *SV*4-*DTM* (*SV*4-*NTM*) by *SV*4-*DTM*($L(l, m, n)$) (*SV*4-*NTM*($L(l, m, n)$)).

Definition 2.5. For any four-dimensional automaton *M* with input alphabet Σ, define T(M) = {$x \in \Sigma^{(4)}$ | *M* accepts *x*}. Furthermore, for each $X \in \{D,N\}$, define

$\mathcal{L}$[4-*XMA*$_1$] = {T | $T = T(M)$ for some 4-*XMA*$_1$},

$\mathcal{L}$[*SV*4-*XTM*($S(l,m,n)$)] = {T | $T = T(M)$ for some *SV*4-*XTM*($S(l,m,n)$) *M*}, and

$\mathcal{L}$[*SV*4-*XTM*($L(l,m)$)] = {T | $T = T(M)$ for some *SV*4-*XTM*((l,m)) *M*}.

3. Necessary spaces

In this section, we investigate the necessary spaces (i.e., lower bounds) for seven-way Turing machines to simulate one-marker automata.

Definition 3.1. Let x be in $\Sigma^{(4)}$ (Σ is a finite set of symbols) and $l_1(x)=l$, $l_2(x)=m$, $l_3(x)=n$. For each j $(1 \le j \le Q[l_4(x)/lmn])$ (where $Q[l_4(x)/lmn]$ denotes the quotient when $l_4(x)$ is divided by lmn),

$$x[(1, 1, 1, (j\text{-}1)lmn+1), (l, m, n, jlmn)]$$

is called the jth (l, m, n)-block of x. We say that the tape x has exactly k (l, m, n)-blocks if $l_4(x)=klmn$, where k is a positive integer.

Definition 3.2. Let (l_1, m_1, n_1), (l_2, m_2, n_2), ... , be a sequence of points (i.e., pairs of three natural numbers), and let $\{(l_i, m_i, n_i)\}$ denote this sequence. We call a sequence $\{(l_1, m_1, n_1)\}$ the *regular sequence of points* if $(l_i, m_i, n_i) \ne (l_j, m_j, n_j)$ for $i \ne j$.

Lemma 3.1. *Let* T_1={$x \in \{0, 1\}^{(4)}$ | $^{\exists} l \ge 1$, $^{\exists} m \ge 1$, $^{\exists} n \ge 1$[$l_1(x)=l$ *and* $l_2(x)=m$ *and* $l_3(x)=n$ *and* (*each plane of x contains exactly one* '1') *and* $^{\exists} d \ge 2$ [(*x has exactly d* (l, m, n)*-blocks, i.e.,* $l_4(x)=dlmn$) *and* (*the last* (l, m, n)*-block is equal to some other* (l, m, n)*-block*)]]}. *Then,*

(1) $T_1 \in \mathcal{L}$[4-*DMA*$_1$], *but*

(2) $T_1 \notin \mathcal{L}$[*SV*4-*DTM*($2^{L(l, m, n)}$)] (*so,* $T_1 \notin \mathcal{L}$[*SV*4-*NTM*($L(l, m, n)$)]) for any function $L(l, m, n)$ such that $\lim_{i\to\infty}[L(l_i, m_i, n_i)/(l_i m_i n_i \log l_i m_i n_i)]=0$. *for some regular sequence of points* $\{(l_i, m_i, n_i)\}$.

Proof: (1): We construct a 4-*DMA*$_1$ *M* accepting T_1 as follows. Given an input x with $l_1(x) = l$, $l_2(x) = m$, and $l_3(x) = n$, *M* first checks, by sweeping plane by plane, that each plane of x contains exactly one '1', and *M* then checks, by making a zigzag of 45° -direction from top plane to bottom plane, that x has exactly d (l, m, n)-blocks for some integer $d \ge 2$. After that, M tests by utilizing its own marker whether the last (l, m, n)-block is identical to some other (l, m, n)-block. *M* then finds the '1' position on the plane and move up vertically from this position. In this course, each time *M* meets a '1' position, it checks whether or not there is a marker on the plane (containing the '1' position).

(i):If there is a marker on the plane, *M* knows that the kth planes of the hth and the last (l, m, n)-blocks are identical, and so *M* then tries to check whether the next (k+1)th planes of the hth and the last (l, m, n)-blocks are identical.

(ii):If there is no marker on the plane, *M* goes back to the '1' position on the plane, and vertically moves up again to find the next '1' position.

In this case, if *M* eventually encounters the top boundary, *M* knows that the kth planes of the hth and the last (l, m, n)-blocks are different (thus, the hth (l, m, n)-block is not

identical to the last (l, m, n)-block), and so M then tries to check whether the next (h+1)th (l, m, n)-block is identical to the last (l, m, n)-block.

In this way, M enters an accepting state just when it finds out some (l, m, n)-block, each of whose planes is identical to the corresponding plane of the last (l, m, n)-block. It will be obvious that $T(M)= T_1$.

(2):Suppose to the contrary that there exists an *SV*4-*DTM*($2^{L(l,\ m,\ n)}$) M accepting T_1, where $L(l, m, n)$ is a function such that

$$\lim_{i\to\infty}[L(l_i, m_i, n_i)/(l_i m_i n_i \log l_i m_i n_i)]=0.$$

For some regular sequence of points $\{(l_i, m_i, n_i)\}$. Let s and t be the numbers of states in the finite control and storage tape symbols of M, respectively. We assume without loss of generality that if M accepts an input, then M enters an accepting state on the bottom boundary. For each $l\geq 1$, $m\geq 1$, $n\geq 1$, let

$V(l, m, n)= \{x\in T_1 \mid l_1(x)=l$ and $l_2(x)=$m and $l_3(x)=n$ and (x has exactly $((lmn)^{lmn}+1)$ (l, m, n)-blocks)$\}$. For each $x\in V(l, m, n)$, let $B(x)=\{b\in\{0, 1\}^{(4)} \mid {}^{\exists}h(1\leq h\leq (lmn)^{lmn})$ [b is the hth (l, m, n)-block of x] $\}$, and let $S(l, m, n)= \{B(x) \mid x\in V(l, m, n)\}$. Note that for each $x\in V(l, m, n)$, there is a sequence of configurations of M which leads M to an accepting state. Let *conf*(x) be the semi-configuration just after M leaves the second-to-last (l, m, n)-block of x. Then, we get following proposition.

Proposition 3.1. *For any two tapes $x, y\in V(l, m, n)$, if $B(x) \neq B(y)$, then conf(x) ≠ conf(y).*

Proof of Lemma 3.1(*continued*)**:** There are at most

$$E(l, m, n)=(l+2)(m+2)(n+2)s2^{L(l,\ m,\ n)}t^{2L(l,\ m,\ n)}$$

different semi-configurations of M just when M enters the last (l, m, n)-block of tapes in $V(l, m, n)$. On the other hand,

$$|S(l, m, n)|=2^r-1 \ (r=(lmn)^{lmn}).$$

Thus, from the assumption concerning the function $L(l, m, n)$, it follows that there exists a point (l_i, m_i, n_i) such that $|S(l_i, m_i, n_i)|>E(l_i, m_i, n_i)$. For such (l_i, m_i, n_i), there exist two tapes x, y in $V(l_i, m_i, n_i)$ such that $B(x) \neq B(y)$ and $conf(x) \neq conf(y)$. This contradicts Proposition 3.1. This completes the proof of (2). □

From Lemma 3.1, we can conclude as follows.

Theorem 3.1. *To simulate* 4-*DMA*$_1$*'s,*

(1) *SV*4-*NTM*'*s require* $\Omega(lmn\log lmn)$ *space, and*

(2) *SV*4-*DTM*'*s require* $2^{\Omega(lmn\log lmn)}$ *space.*

By using the same technique as in the proof of Theorem 3.1, we can get as follows.

Theorem 3.2. *To simulate* 4-*NMA*$_1$*'s,*

(1) *SV*4-*NTM*'*s require* $\Omega(l^2m^2n^2)$ *space, and*

(2) *SV*4-*DTM*'*s require* $2^{\Omega(l2m2n2)}$ *space.*

4. Conclusion

In this paper, we showed the necessary spaces for four-dimensional Turing machines to simulate four-dimensional one-marker automata. It will be interesting to investigate how much space is necessary and sufficient for seven-way four-dimensional deterministic or nondeterministic Turing machines to simulate four-dimensional 'alternating' one-marker automata.

References

1. M. Blum and C. Hewitt, "Automata on a two-dimensional tape", IEEE Symposium on Switching and Automata Theory, pp.155-160 (1967).
2. M. Sakamoto, "Three-dimensional alternating Turing machines", Ph.D. Thesis, Yamaguchi University (1999).
3. M.Nagatomo, M.Sakamoto, H.Susaki, T. Zhang, S.Ikeda, H.Furutani, T.Ito, Y.Uchida, and T.Yoshinaga, "Sufficient spaces for seven-way four-dimensional Turing machines to simulate four-dimensional one-marker automata", Proceedings of the International Conference on Artificial Life and Robotics（ICAROB2015）, Horuto Hall, Oita, Japan, OS6-1, pp.340-343(CD-ROM) (2015).