

Some Accepting Powers of Three-Dimensional Synchronized Alternating Turing Machines

Takao ITO, Dept. of Computer Science and Systems Engineering, University of Miyazaki, JAPAN,
ito@ube-k.ac.jp

Makoto SAKAMOTO, Dept. of Computer Science and Systems Engineering, University of Miyazaki, JAPAN,
sakamoto@cs.miyazaki-u.ac.jp

Abstract

This paper introduces a three-dimensional synchronized alternating Turing machine (*3-SATM*), and investigates fundamental properties of *3-SATM*'s whose input tapes are restricted to cubic ones. The main topics of this paper are: (1) a relationship between the accepting powers of *3-SATM*'s and three-dimensional alternating Turing machines with small space bounds, (2) a relationship between the accepting powers of five-way and six-way *3-SATM*'s, (3) a relationship between the accepting powers of *3-SATM*'s and three-dimensional nondeterministic Turing machines.

Key Words

alternation, computation, finite automaton, space complexity, synchronization, three-dimension, Turing machine

1 Introduction and preliminaries

Synchronized alternating Turing machines were introduced in [4] to study the effect of allowing processes of an alternating Turing machine to communicate via synchronization. Informally, a synchronized alternating machine is an alternating machine with a special subset of internal states called synchronizing states. Each of these synchronizing states is associated with a synchronizing symbol. If, during the course of computation, some process enters a synchronizing state, then it has to wait until all other processes enter either an accepting state or a synchronizing state with the same synchronizing symbol. When this happens, all processes are allowed to continue their computation; otherwise, the machine is said to have a deadlock. A computation is successful if no deadlocks occur and all processes terminate in accepting states. It turns out that synchronization significantly increases the computational power of alternating Turing machines.

On the other hand, recently, due to the advances in many application areas such as computer vision, robotics, and so forth, it has become increasingly apparent that the study of three-dimensional pattern processing has been of crucial importance. Thus, the research of three-dimensional automata as a computational model of three-dimensional pattern processing has also been meaningful. From this viewpoint, we introduced three-dimensional alternating Turing machine [22].

In this paper, we continue the investigations about three-dimensional alternating Turing machines, introduce a three-dimensional synchronized alternating Turing machine (*3-SATM*), and investigate fundamental properties of *3-SATM*'s whose input tapes are restricted to cubic ones.

Let Σ be a finite set of symbols. A *three-dimensional input tape* over Σ is a three-dimensional rectangular array of elements of Σ . The set of all the three-dimensional input tapes over Σ is denoted by $\Sigma^{(3)}$. Given an input tape $x \in \Sigma^{(3)}$, for each j ($1 \leq j \leq 3$), we let $l_j(x)$ be the length of x along the j th axis. The set of all $x \in \Sigma^{(3)}$ with $l_1(x) = m_1, l_2(x) = m_2$ and $l_3(x) = m_3$ is denoted by $\Sigma^{(m_1, m_2, m_3)}$. If $1 \leq i_j \leq l_j(x)$ for each j ($1 \leq j \leq 3$), let $x(i_1, i_2, i_3)$ denote the symbol in x with coordinates (i_1, i_2, i_3) . Furthermore, we define $x[(i_1, i_2, i_3), (i'_1, i'_2, i'_3)]$, when $1 \leq i_j \leq i'_j \leq l_j(x)$ for each integer j ($1 \leq j \leq 3$), as the three-dimensional input tape y satisfying the following;

- (1) for each j ($1 \leq j \leq 3$), $l_j(y) = i'_j - i_j + 1$;
- (2) for each r_1, r_2, r_3 ($1 \leq r_1 \leq l_1(y), 1 \leq r_2 \leq l_2(y), 1 \leq r_3 \leq l_3(y)$), $y(r_1, r_2, r_3) = x(r_1 + i_1 - 1, r_2 + i_2 - 1, r_3 + i_3 - 1)$. (We call $x[(i_1, i_2, i_3), (i'_1, i'_2, i'_3)]$ the $[(i_1, i_2, i_3), (i'_1, i'_2, i'_3)]$ *segment* of x .)

We now introduce a three-dimensional synchronized alternating Turing machine. A three-dimensional synchronized alternating Turing machine (denoted by 3-SATM) is a 10-tuple $M = (Q, q_0, U, E, S, F, \Sigma, \Pi, \Gamma, \delta)$, where

- (1) $Q = U \cup E \cup S$ is a finite set of *states*,
- (2) $q_0 \in Q$ is the *initial state*,
- (3) U is the set of *universal states*,
- (4) E is the set of *existential states*,
- (5) $S \subseteq \{(q, s) : q \in U \cup E, s \in \Pi\}$ is the set of *synchronizing states (s-states)*,
- (6) $F \subseteq Q$ is the set of *accepting states*,
- (7) Σ is a finite *input alphabet* ($\# \notin \Sigma$ is the *boundary symbol*),
- (8) Π is a finite *alphabet of synchronizing symbols*,
- (9) Γ is a finite *storage tape alphabet* containing the special *blank symbol* B ,
- (10) $\delta \subseteq (Q \times (\Sigma \cup \{\#\}) \times \Gamma) \times (Q \times (\Gamma - \{B\})) \times \{east, west, south, north, top, down, no\ move\} \times \{left, right, no\ move\}$ is the *next move relation*.

As shown in Fig.1, M has a read-only cubic input tape with boundary symbols $\#$'s ($\# \notin \Sigma$) and one semi-infinite storage tape, initially filled with the blank symbols. M begins in state q_0 . A *position* is assigned to each cell of the input tape and the storage tape, as shown in Fig.1. A *step* of M consists of reading one symbol from each tape, writing a symbol on the storage tape, moving the input and storage tape heads in specified directions, and entering a new state, according to the next move relation δ . When a process P enters a synchronizing state, it stops and waits until all the parallel processes either enter the states with the same synchronizing element or stop in accepting states.

An *instantaneous description (ID)* of a 3-SATM $M = (Q, q_0, U, E, S, F, \Sigma, \Pi, \Gamma, \delta)$ is a pair of an element of $\Sigma^{(3)}$ and an element of

$$C_M = (N \cup \{0\})^3 \times S_M, S_M = Q \times (\Gamma - \{B\})^* \times \mathbf{N},$$

where \mathbf{N} denotes the set of all positive integers. The first component of an *ID* $I = (x, ((i_1, i_2, i_3), (q, \alpha, k)))$ represents the input to M , and the first component (i_1, i_2, i_3) of the second component of I represents the input head position ($0 \leq i_1 \leq l_1(x) + 1, 0 \leq i_2 \leq l_2(x) + 1, 0 \leq i_3 \leq l_3(x) + 1$), and the second component (q, α, k) of the second component of I represents the state of the finite control, nonblank contents of the storage tape, and the storage head position ($1 \leq k \leq |\alpha| + 1$). An element of C_M is called a *configuration* of M , and an element of S_M

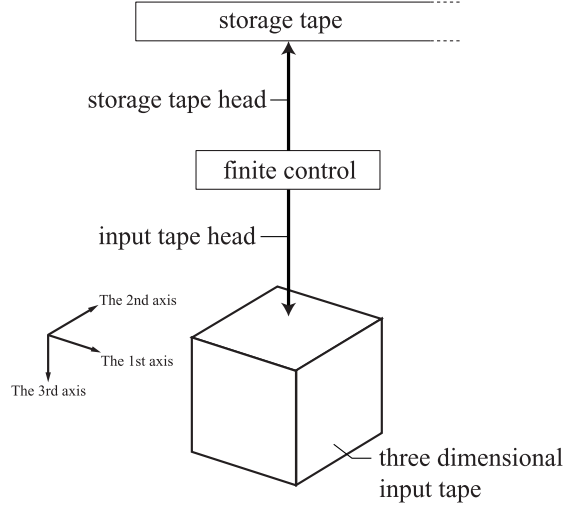


Fig. 1: Three-dimensional synchronized alternating Turing machine.

is called a *storage state* of M .

An *ID* is *universal* (*existential*, *synchronizing*, *accepting*) depending on the type of the state of the finite control. The *initial ID* of M on input x is $I_M(x) = (x, ((1, 1, 1), (q_0, \epsilon, 1)))$, where ϵ is the null word.

Suppose I_1 and I_2 are two *ID*'s of M and I_2 follows from I_1 in one step according to the next move relation δ . Then we write $I_1 \vdash_M I_2$ and say that I_2 is a *successor* of I_1 . The reflexive and transitive closure of \vdash_M is denoted by \vdash_M^* .

A sequence of *ID*'s of M , $I_0, I_1, \dots, I_m (m \geq 0)$, is called a *sequential computation* of M if $I_0 \vdash_M I_1 \vdash_M \dots \vdash_M I_m$. If $I_0 = I_M(x)$ for some x , we call this sequence a *computation path* of M on x .

The *full computation tree* of M on an input tape x is a (possibly infinite) labeled tree \vdash_x^M such that

- (1) each node v is labeled by some *ID* I_v of M ,
 - (2) the root is labeled by $I_M(x)$,
 - (3) v_2 is a direct descendant of v_1 iff $I_{v_1} \vdash_M I_{v_2}$.
- (Each branch of \vdash_x^M is called a *process*.)

The *synchronizing sequence* (*s-sequence*) of a node v in a full computation tree T with root v_0 is the sequence of synchronizing symbols occurring in labels of the nodes on the path from v_0 to v . Two *s-sequences* are *compatible* if one is a prefix of the other. If s_1 and s_2 are two compatible *s-sequences*, and s_2 is longer than s_1 , then we use $s_2 - s_1$ to denote their difference.

A *computation tree* of M on an input x is a (possibly infinite) subtree T' of the full computation tree \vdash_x^M satisfying the following conditions:

- (1) if u is an internal (non-leaf) node of the tree T' , I_u is universal and $\{I \mid I_u \vdash_M I\} = \{I_1, \dots, I_m\}$, then u

has exactly m children v_1, \dots, v_m , such that $I_{v_i} = I_i, 1 \leq i \leq m$,

- (2) if u is an internal node of the tree and I_u is existential, then u has exactly one child v such that $I_u \vdash I_v$,
- (3) For arbitrary nodes u and v of T' , the s -sequences of u and v are compatible.

If M on input x has no computation trees, then any subtree of T_x^M that satisfies the first two conditions above must have two processes with incompatible s -sequences. In this case, we say M *deadlocks* on x . The two processes with incompatible s -sequences are called *deadlock processes* and the nonmatching s -states causing the deadlock are called *deadlock states*.

The longest synchronizing sequence of a node in the computation tree T is called the *synchronizing sequence of the computation tree T* .

An *accepting computation tree* of M on an input x is a finite computation tree of M on x such that each leaf node is labeled by an accepting ID . We say that M *accepts* x if there is an accepting computation tree of M on x . Let $T(M) = \{x \in \Sigma^{(3)} \mid M \text{ accepts } x\}$.

We next introduce a five-way three-dimensional synchronized alternating Turing machine which can be considered as a synchronized version of five-way three-dimensional alternating Turing machine [22].

A *five-way three-dimensional synchronized alternating Turing machine* (denoted by *FV3-SATM*) is a *3-SATM* $M = (Q, q_0, U, E, S, F, \Sigma, \Pi, \Gamma, \delta)$, such that

$$\delta \subseteq (Q \times (\Sigma \cup \{\#\}) \times \Gamma) \times (Q \times \Gamma - \{B\}) \times \{east, west, south, north, down, no\ move\} \times \{left, right, no\ move\}.$$

That is, a *FV3-SATM* is a *3-SATM* whose input head can move east, west, south, north, or down, but not up.

Let $L(m) : \mathbf{N} \rightarrow \mathbf{N}$ be a function with one variable m . With each *3-SATM* (or *FV3-SATM*) M we associate a *space complexity function* $SPACE$ which takes ID 's to natural numbers. That is, for each ID $I = (x, ((i_1, i_2, i_3), (q, \alpha, k)))$, let $SPACE(I)$ be the length of α . We say that M is " $L(m)$ *space-bounded*" if for all m and for all x with $l_1(x) = l_2(x) = l_3(x) = m$, if x is accepted by M , then there is an accepting computation tree of M on input x such that for each node π of the tree, $SPACE(I(\pi)) \leq L(m)$. By "*3-SATM*($L(m)$)" ("*FV3-SATM*($L(m)$)") we denote an $L(m)$ *space-bounded 3-SATM* (*FV3-SATM*) whose input tapes are restricted to cubic ones.

Three-dimensional alternating Turing machines (*3-ATM*'s and five-way three-dimensional alternating Turing machines (*FV3-ATM*'s) in [21] are *3-SATM*'s and *FV3-SATM*'s, respectively, which have no synchronizing states. We use *3-SUTM* (*FV3-SUTM*, *3-UTM*, *FV3-UTM*) to denote a *3-SATM* (*FV3-SATM*, *3-ATM*, *FV3-ATM*) which has no existential states. By *3-ATM*($L(m)$) (*FV3-ATM*($L(m)$), *3-SUTM*($L(m)$), *FV3-SUTM*($L(m)$), *3-UTM*($L(m)$), *FV3-UTM*($L(m)$)), we denote an $L(m)$ *space-bounded 3-ATM* (*FV3-ATM*, *3-SUTM*, *FV3-SUTM*, *3-UTM*, *FV3-UTM*).

A *three-dimensional deterministic Turing machine* (*3-DTM*) (*five-way three-dimensional deterministic Turing machine* (*FV3-DTM*)) is a *3-ATM* (*FV3-ATM*) whose ID 's each have at most one successor, and a *three-dimensional nondeterministic Turing machine* (*3-NTM*) (*five-way three-dimensional nondeterministic Turing machine* (*FV3-NTM*)) is a *3-ATM* which has no universal states. We denote an $L(m)$ *space-bounded 3-DTM* (*3-NTM*, *FV3-DTM*, *FV3-NTM*) by *3-DTM*($L(m)$) (*3-NTM*($L(m)$), *FV3-DTM*($L(m)$), *FV3-NTM*($L(m)$)). We use *3-SAFA* (*FV3-SAFA*, *3-AFA*, *FV3-AFA*, *3-NFA*, *FV3-NFA*, *3-DFA*, *FV3-DFA*) to denote a *three-*

dimensional synchronized alternating finite automaton (five-way three-dimensional synchronized alternating finite automaton, three-dimensional alternating finite automaton, five-way three-dimensional alternating finite automaton, three-dimensional nondeterministic finite automaton, five-way three-dimensional nondeterministic finite automaton, three-dimensional deterministic finite automaton, five-way three-dimensional deterministic finite automaton). That is, a 3-SAFA (FV3-SAFA, 3-AFA, FV3-AFA, 3-NFA, FV3-NFA, 3-DFA, FV3-DFA) is a 3-SATM (FV3-SATM, 3-ATM, FV3-ATM, 3-NTM, FV3-NTM, 3-DTM, FV3-DTM) which doesn't have storage tape. Similarly, we use 3-SUFA (FV3-SUFA, 3-UFA, FV3-UFA) to denote a 3-SUTM (FV3-SUTM, 3-UTM, FV3-UTM) which doesn't have the storage tape. Furthermore, for any integer $k \geq 1$, 3-SATM($L(m)$)[k] is used to denote a 3-SATM($L(m)$) such that any computation tree of M on any input x has at most k leaves. FV3-SATM($L(m)$)[k], 3-SUTM($L(m)$)[k], \dots , 3-SAFA($L(m)$)[k], etc. have the similar meaning. For any integer $k \geq 1$, 3-NFA(k -heads) (3-DFA(k -heads)) is used to denote a 3-NFA (3-DFA) which has k input heads. For any machine class C , let

$$\mathcal{L}[C] = \{ T \mid T = T(M) \text{ for some } M \text{ in } C \}$$

Thus, for example, $\mathcal{L}[3\text{-SATM}(L(m))]$ denotes the class of sets accepted by 3-SATM($L(m)$)'s.

2 Synchronization versus non-synchronization

This section investigates a relationship between the accepting powers of 3-ATM's and 3-SATM's.

Lemma 2.1. Let $T_1 = \{x \in \{0, 1\}^{2m \times 2m \times 2m} \mid m \geq 1 \& x[(1, 1, 1), (2m, 2m, m)] = x[(1, 1, m + 1), (2m, 2m, 2m)]\}$. Then,

- (1) $T_1 \in \mathcal{L}[FV3\text{-SUFA}[2]]$, and
- (2) $T_1 \notin \mathcal{L}[3\text{-ATM}(L(m))]$ for any $L : \mathbf{N} \rightarrow \mathbf{N}$ such that $L(m) = o(\log m)$.

Proof: (1) We can construct an FV3-SUFA[2] M accepting T_1 as follows: Given x with $l_1(x) = l_2(x) = l_3(x) = 2m$ ($m \geq 1$), starting on position (1,1,1) of x , M first splits universally into two processes p_1 and p_2 . Process p_2 moves its head to (1, 1, $m + 1$) and then synchronizes with process p_1 to compare $x(i_1, i_2, i_3)$ and $x(i_1, i_2, i_3 + m)$ for each i_1, i_2, i_3 ($1 \leq i_1 \leq 2m, 1 \leq i_2 \leq 2m, 1 \leq i_3 \leq m$). M accepts x iff $x(i_1, i_2, i_3) = x(i_1, i_2, i_3 + m)$ for each i_1, i_2, i_3 ($1 \leq i_1 \leq 2m, 1 \leq i_2 \leq 2m, 1 \leq i_3 \leq m$).

- (2) This proof is the same as that of Theorem 1 in [21].

From this lemma, we have

Theorem 2.1. For any function $L(m) = o(\log m)$,

- (1) $\mathcal{L}[FV3\text{-UTM}(L(m))] \subsetneq \mathcal{L}[FV3\text{-SUTM}(L(m))]$,
- (2) $\mathcal{L}[FV3\text{-ATM}(L(m))] \subsetneq \mathcal{L}[FV3\text{-SATM}(L(m))]$,
- (3) $\mathcal{L}[3\text{-UTM}(L(m))] \subsetneq \mathcal{L}[3\text{-SUTM}(L(m))]$, and
- (4) $\mathcal{L}[3\text{-ATM}(L(m))] \subsetneq \mathcal{L}[3\text{-SATM}(L(m))]$

Corollary 2.1.

- (1) $\mathcal{L}[FV3-UFA] \subsetneq \mathcal{L}[FV3-SUFA]$,
- (2) $\mathcal{L}[FV3-AFA] \subsetneq \mathcal{L}[FV3-SAFA]$,
- (3) $\mathcal{L}[3-UFA] \subsetneq \mathcal{L}[3-SUFA]$, and
- (4) $\mathcal{L}[3-AFA] \subsetneq \mathcal{L}[3-SAFA]$

Theorem 2.2. For any function $L(m) \geq \log m$,

$$\mathcal{L}[3-SUTM(L(m))] = \mathcal{L}[3-UTM(L(m))].$$

Proof: Given a 3- $SUTM(L(m))$ M where $L(m) \geq \log m$, we construct a 3- $UTM(L(m))$ M' to accept the same set as follows. On input x of side-length $m \geq 1$, M' simulates each process of M with a process of its own. When some process p of M enters an s -state, the corresponding process p' of M' spawns off a process c whose worktape contains the s -symbol associated with the s -state and the number of s -states p has entered so far. Since each process makes at most $d^{L(m)}$ moves (d is a constant), and $L(m) \geq \log m$, there is enough space to store them. Process c restarts the computation of M on x and verifies that the corresponding s -symbols in other processes match with the one stored on its worktape. If a discrepancy occurs, M' rejects. It is easy to see that M and M' accept the same set.

By using a technique similar to that in the proof of Theorem 2.2, we have

Theorem 2.3. For any function $L(m) \geq \log m$,

$$\mathcal{L}[FV3-SUTM(L(m))] = \mathcal{L}[FV3-UTM(L(m))].$$

3 Five-way versus six-way

This section investigates a relationship between the accepting powers of five-way and six-way synchronized machines.

It is shown in [5] that three-way two-dimensional synchronized alternating Turing machine are equivalent to two-dimensional synchronized alternating Turing machines. By using the same idea as in the proof of this fact, we can easily show that the following theorem holds.

Theorem 3.1. For any function $L : \mathbf{N} \rightarrow \mathbf{N}$,

$$\mathcal{L}[FV3-SATM(L(m))] = \mathcal{L}[3-SATM(L(m))].$$

Below, we investigate a difference between the accepting powers of space-bounded 3- $SUTM$'s and $FV3-SUTM$'s.

Lemma 3.1. Let $T_2 = \{x \in \{0, 1\}^{m \times m \times m} \mid m \geq 2, \&x[(1, 1, 1), (m, m, 1)] \neq x[(1, 1, 2), (m, m, 2)] \& x[(1, 1, 3), (m, m, 3)] \in \{0\}^{(3)}\}$. Then,

- (1) $T_2 \in \mathcal{L}[3-DFA](= \mathcal{L}[3-SUTM(0)[1]])$, and
- (2) $T_2 \notin \mathcal{L}[FV3-SUTM(L(m))]$ for any $L : \mathbf{N} \rightarrow \mathbf{N}$ such that $L(m) = o(m^2)$.

Proof: (1) We can construct a 3-DFA M accepting T_2 as follows: Given x with $l_1(x) = l_2(x) = l_3(x) = m$ ($m \geq 2$), starting on position $(1,1,1)$ of x , M first checks that $x[(1, 1, 3), (m, m, 3)] \in \{0\}^{(3)}$. Then, M repeats the following process from $j = 1$ to m ; M records the input symbol $x[(1, 1, 1), (m, m, 1)]$ in the finite control and checks that two symbols $x[(1, 1, 1), (m, m, 1)] \neq x[(1, 1, 2), (m, m, 2)]$. If so, M enters an accepting state. It is clear that $T(M) = T_2$.

(2) Suppose that there exists a FV3-SUTM($L(m)$) M accepting T_2 , where $L(m) = o(m)$. By using the technique of counting argument[5,21], we can get the desired result. \square

Lemma 3.2. Let $T_3 = \{x \in \{0, 1\}^{2m \times 2m \times 2m} \mid m \geq 1 \& x[(1, 1, 1), (2m, 2m, m)] \neq x[(1, 1, m+1), (2m, 2m, 2m)]\}$. Then,

(1) $T_3 \in \mathcal{L}[3\text{-DTM}(\log m)] (= \mathcal{L}[3\text{-SUTM}(\log m)[1]])$, and

(2) $T_3 \notin \mathcal{L}[FV3\text{-SUTM}(L(m))]$ for any $L : \mathbf{N} \rightarrow \mathbf{N}$ such that $L(m) = o(m^3)$.

Proof: (1) We can construct a 3-DTM($\log m$) M accepting T_3 as follows: Given x with $l_1(x) = l_2(x) = l_3(x) = 2m$ ($m \geq 1$), starting on position $(1,1,1)$ of x for all i_1, i_2, i_3 ($1 \leq i_1 \leq 2m, 1 \leq i_2 \leq 2m, 1 \leq i_3 \leq m$), M repeats the following process; M records the input symbol $x(i_1, i_2, i_3)$ in the finite control and checks that two symbols $x(i_1, i_2, i_3) \neq x(i_1, i_2, i_3 + m)$. (This can be easily done by using $\log m$ cells of the storage tape.) If so, M and enters an accepting state. It is clear that $T(M) = T_3$.

(2) The idea is almost the same as in the proof of Lemma 3.1 (2). \square

From Lemmas 3.1 and 3.2 we can get the following theorem.

Theorem 3.2. Let $L : \mathbf{N} \rightarrow \mathbf{N}$ be a function such that (i) $L(m) = o(m^2)$, or (ii) $L(m) \geq \log m$ and $L(m) = o(m^3)$. Then,

$$\mathcal{L}[FV3\text{-SUTM}(L(m))] \subsetneq \mathcal{L}[3\text{-SUTM}(L(m))].$$

Corollary 3.1. $\mathcal{L}[FV3\text{-SUF A}] \subsetneq \mathcal{L}[3\text{-SUF A}]$.

It is easy to show that the following theorem holds.

Theorem 3.3. For any function $L(m) \geq m^3$,

$$\mathcal{L}[FV3\text{-SUTM}(L(m))] = \mathcal{L}[3\text{-SUTM}(L(m))].$$

4 Nondeterminism versus synchronized alternation

This section investigates a relationship between the accepting powers of three-dimensional synchronized alternating machines and three-dimensional nondeterministic machines.

Let $f : \mathbf{N} \rightarrow \mathbf{N}$ be a function. The function f is said to be *three-dimensionally fully space constructible* if there is a 3-DTM which for any input tape x with $l_1(x) = l_2(x) = l_3(x) = m$ ($m \geq 1$) makes use of exactly $f(n)$ cells of the storage tape and halts.

Theorem 4.1. For any function $L(m) \geq \log m$,

$$\mathcal{L}[3\text{-SATM}(L(m))] = \bigcup_{c \geq 0} \mathcal{L}[3\text{-NTM}(m^3 c^{L(m)})].$$

Proof: We first show that $\mathcal{L}[3\text{-SATM}(L(m))] \subseteq U_{c>0} \mathcal{L}[3\text{-NTM}(m^3 c^{L(m)})]$. Given a 3-SATM($L(m)$) M , we can construct a 3-NTM($m^3 c^{L(m)}$) M' to simulate M by doing a breadth-first-like traversal of the computation tree of M on input x of side-length m . Each process of M is simulated until it enters an s -state; M' will compare the corresponding s -states to make sure that no deadlock occurs before continuing the simulation. Since there are at most $m^3 d^{L(m)}$ distinct configurations of M on an input x of side-length m , M' needs at most $m^3 e^{L(m)}$ space, for some constants d and e , at any time to maintain the current ID 's of all processes of M on x . Then on any input x of side-length m , M uses at most $L(m)$ space iff M' uses at most $m^3 e^{L(m)}$ space.

On the other hand, by using same idea described in Lemma 3.4 in [6], we can show that $\bigcup_{c \geq 0} \mathcal{L}[3\text{-NTM}(m^3 c^{L(m)})] \subseteq \mathcal{L}[3\text{-SATM}(L(m))]$. \square

Theorem 4.2. For any integer $k \geq 1$,

- (1) $\mathcal{L}[3\text{-SAFA}[k]] = \mathcal{L}[3\text{-NFA}(k\text{-heads})]$, and
- (2) $\mathcal{L}[FV3\text{-SAFA}[k]] = \mathcal{L}[3\text{-NFA}(k\text{-heads})]$

Proof: We only prove (1). Given a 3-NFA(k -heads) M where $k \geq 1$, we can construct a 3-SAFA[k] M' . Let H_1, H_2, \dots, H_k denote the input heads of M . These heads are simulated by a single input head of 3-SAFA[k] M' in the following way. The computation of M' branches from the initial configuration in a universal manner into k processes. Note that, the initial configuration is the only universal configuration which occurs in the computation. The states of M' (in all processes of M') store the simulated state of M . If the stored state is an accepting (rejecting) state, then the state of M is also an accepting (rejecting) one. In the i th process, for any i ($1 \leq i \leq k$), the input head of M' is at the same position as H_i .

One step of M is simulated by two steps of M' . Besides the state of M the symbols scanned by all heads of M have to be known to M' . Every process in the computation of M' has only a part of the necessary information. The processes can share this information via the synchronization. The synchronizing element consists of k components and represents the symbols scanned by the input heads of M . The i th process, for $1 \leq i \leq k$, sets the i th component according to the symbol scanned by the input head of M' . The other components are set nondeterministically. The synchronization is successful only in the case when every process has correctly guessed the remaining components.

The next synchronization is necessary because of nondeterminism. One configuration of M has several potential successors. All of the processes of M' must agree on the next step of M (they must simulate the same successor of the currently simulated configuration). The synchronizing element represents the new state of M and the actions of the heads of M . The successful synchronization means that all processes choose the same element of the next move relation of M . After that, M' moves its heads and enters a new state in accordance with the synchronizing element (i.e., in the i th process (for $1 \leq i \leq k$), M' moves by its input head like M by H_i). It will be obvious that M' can simulate M .

Conversely, given a 3-SAFA[k] M , we can construct a 3-NFA(k -heads) M' such that $T(M) = T(M')$. The proof is omitted here. \square

We next investigate a relationship between the accepting powers of five-way nondeterministic machines and five-way synchronized machines with only universal states.

Theorem 4.3.

- (1) $\mathcal{L}[FV3\text{-SUFA}[2]] - \mathcal{L}[FV3\text{-NTM}(o(m^3))] \neq \emptyset$,

(2) $\mathcal{L}[FV3-NFA] - \mathcal{L}[FV3-SUTM(o(m^2))] \neq \phi$, and

(3) $\mathcal{L}[FV3-NTM(\log m)] - \mathcal{L}[FV3-SUTM(o(m^3))] \neq \phi$.

Proof: (1) Let T_1 be the set described in Lemma 2.1. By using the technique of counting argument, we can show that $T_1 \notin \mathcal{L}[FV3-NTM(o(m^3))]$. (1) follows from this fact and Lemma 2.1 (1).

(2) Let T_2 be the set of described in Lemma 3.1. It is easy to see that $T_2 \in \mathcal{L}[FV3-NFA]$. (2) follows from this fact and Lemma 3.1 (2).

(3) Let T_3 be the set of described in Lemma 3.2. It is easy to see that $T_3 \in \mathcal{L}[FV3-NTM(\log m)]$. (3) follows from this fact and Lemma 3.2 (2). \square

Corollary 4.1. *For any function $L(m) = o(m^3)$, $\mathcal{L}[FV3-SUTM(L(m))]$ and $\mathcal{L}[FV3-NTM(L(m))]$ are incomparable.*

5 Conclusions

Turing machine is a simple mathematical model of computers which was introduced by Turing in 1936 [30]. In theoretical computer science, Turing machine has played many important roles in understanding and exploiting basic concepts and mechanisms in computing and information processing. If the restrictions in its structure and move are placed on the Turing machine, the restricted Turing machine is less powerful than the original one. However, it has become increasingly apparent that the characterization and classification of powers of the restricted Turing machines should be of great importance. Such a study was active in 1950's and 1960's, and was called the automata theory [3]. On the other hand, in the study of computational complexity, the complexity measures are very important. In general, it is well known that the computational complexity has originated in a study of considering how the computational powers of various types of automata are characterized by the complexity measures such as space complexity, time complexity, or some other related measures. Especially, the concept of space complexity is very useful to characterize various types of automata from a point of view of memory requirements [13]. After that, the growth of the processing of pictorial information by computer was rapid in those days. Therefore, the problem of computational complexity was also arisen in the two-dimensional information processing. Blum and Hewitt first proposed two-dimensional automata, and their pattern recognition abilities in 1967 [1]. Since then, many researchers in this field have been investigating a lot of properties about automata on a two-dimensional tape [7]. By the way, the question of whether processing three-dimensional digital patterns is much difficult than two-dimensional ones is of great interest from the theoretical and practical standpoints. In recent years, due to the advances in many application areas such as computer graphics, computer-aided design / manufacturing, computer vision, image processing, robotics, and so on, the study of three-dimensional pattern processing has been of crucial importance. Thus, the study of three-dimensional automata as the computational model of three-dimensional pattern processing has been meaningful. However, it is conjectured that the three-dimensional pattern processing has its own difficulties not arising in two-dimensional case. One of these difficulties occurs in recognizing topological properties of three-dimensional pattern because the three-dimensional neighborhood is more complicated than two-dimensional case [18,19]. During the past about thirty years, automata on a three-dimensional tape have been proposed [2,14,16-17,20,29]. We have proposed several three-dimensional automata as three-dimensional computational models, and have showed many properties of three-dimensional automata for about twenty years [8-12,15,22-28]. Especially, three-dimensional alternating Turing machine was introduced and investigated as a generalization of three-dimensional nondeterministic Turing machine and as a mechanism to model parallel computation [22]. In this paper, we continued the study about this parallel computational model, and investigated synchronized version of three-dimensional alternating Turing machines whose input tapes are restricted to cubic ones [8-10]. We first showed a relationship between the accepting powers of 3-SATM's and 3-ATM's with small space bounds. We next showed a relationship between the accepting powers of five-way and six-way 3-SATM's. Moreover, we showed a relationship between the accepting powers of 3-SATM's and 3-NTM's. Finally, we conclude this paper by giving two open problems.

(1) For any function $L(m) \geq \log m$, $\mathcal{L}[3-ATM(L(m))] \subsetneq \mathcal{L}[3-SATM(L(m))]$? and $\mathcal{L}[FV3-ATM(L(m))] \subsetneq \mathcal{L}[FV3-SATM(L(m))]$?

(2) For any integer $k \geq 1$, $\mathcal{L}[3\text{-SUF}A[k]] \subsetneq \mathcal{L}[3\text{-SUF}A[k+1]]$? and $\mathcal{L}[FV3\text{-SUF}A[k]] \subsetneq \mathcal{L}[FV3\text{-SUF}A[k+1]]$?

Acknowledgement

The authors would like to thank the anonymous reviewers for their useful comments that improved our paper.

References

- [1] M. Blum and C. Hewitt, Automata on a two-dimensional tape, *Proceedings of IEEE Symposium on Switching and Automata Theory*, pp.155-160, 1967.
- [2] M. Blum and W. J. Sakoda, On the capability of finite automata in 2 and 3 dimensional space, *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pp.147-161, 1977.
- [3] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Mass., 1979.
- [4] J. Hromkovič, How to organize the communication among parallel proceses in alternating computations, *manuscript*, 1986.
- [5] J. Hromkovič, J. Karhumäki, B. Rován and A. Slobodová, On the power of synchronization in parallel computations, *Tech. Report*, Comenius Univ., Bratislava, Czechoslovakia, Dept. of Theoretical Cybernetics and Institute of Computer Science, 1989.
- [6] C.M. Ibara and N.Q. Trân, On space-bounded synchronizing alternating Turing machines, *Theoretical Computer Science* 99, pp.243-264, 1992.
- [7] K. Inoue and I. Takanami, A survey of two-dimensional automata theory, *Information Sciences*, Vol.55, pp.99-121, 1991.
- [8] T. Ito, M. Sakamoto, M. Saito, K. Iihoshi, H. Furutani, M. Kono, and K. Inoue, Three-dimensional synchronized alternating Turing machines, *Proceedings of the 11th International Symposium on Artificial Life and Robotics*, OS 9-4, Oita, Japan, 2006.
- [9] T. Ito, M. Sakamoto, N. Tomozoe, K. Iihoshi, H. Furutani, M. Kono, T. Tamaki, and K. Inoue, Some properties of three-dimensional synchronized alternating Turing machines, *Proceedings of the 10th WSEAS International Conference on COMPUTERS*, pp.665-670, Athens, Greece, 2006.
- [10] T. Ito, M. Sakamoto, N. Tomozoe, K. Iihoshi, H. Furutani, M. Kono, T. Tamaki, and K. Inoue, Two topics concerning three-dimensional synchronized alternating Turing machines, *WSEAS Transactions on Computers*, Issue 10, Vol.5, pp.2149-2155, 2006.
- [11] T. Ito, M. Sakamoto, Y. Nagamizu, K. Iihoshi, N. Tomozoe, H. Furutani, M. Kono, S. Ikeda, T. Tamaki, and K. Inoue, Three-dimensional parallel Turing machines, *Proceedings of the 12th International Symposium on Artificial Life and Robotics*, pp.131-134, Oita, Japan, 2007.
- [12] T. Makino, H. Okabe, S. Taniguchi, M. Sakamoto, and K. Inoue, Three-dimensional multi-inkdot automata, *International Journal of Artificial Life and Robotics*, Vol.9, No.2, pp.99-101, 2005.
- [13] P. Michel, A survey of space complexity, *Theoretical Computer Science*, Vol.101, pp.99-132, 1992.
- [14] D. G. Morgenthaler, Three-dimensional digital topology: the genus, *Technical Report TR-980*, Computer Science Center, University of Maryland, 1980.

- [15] Y. Nagamizu, M. Sakamoto, T. Ito, K. Iihoshi, N. Tomozoe, H. Furutani, M. Kono, S. Ikeda, T. Takanami, and K. Inoue, Remarks on recognizability of topological components by three-dimensional automata, *Proceedings of The 12th International Symposium on Artificial Life and Robotics*, pp.135-139, Oita, Japan, 2007.
- [16] A. Nakamura and K. Aizawa, On the recognition of properties of three-dimensional pictures, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.PAMI-7, pp.708-713, 1985.
- [17] A. Nakamura and K. Aizawa, Detection of interlocking components in three-dimensional digital pictures, *Information Sciences*, Vol.40, pp.143-153, 1986.
- [18] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, Academic Press, New York, 1976.
- [19] A. Rosenfeld, *Picture Languages—Formal Models for Picture Recognition*, Academic Press, 1979.
- [20] A. Rosenfeld, Three-dimensional digital topology, *Information and Control*, Vol.50, pp.119-127, 1981.
- [21] M. Sakamoto, I. Sakuramoto, K. Inoue and I. Takanami, A space lower-bound technique for two-dimensional alternating Turing machines, *IEICE Trans. D-I*, J74-DI(4), pp.311-314, 1991.
- [22] M. Sakamoto, K. Inoue, and I. Takanami, A note on three-dimensional alternating Turing machines with space smaller than $\log m$, *Information Sciences* Vol.72, pp.225-249, 1993.
- [23] M. Sakamoto, A. Ito, K. Inoue and I. Takanami, Simulation of three-dimensional one-marker automata by five-way Turing machines, *Information Sciences*, Vol.77, pp.77-99, 1994.
- [24] M. Sakamoto, K. Inoue and I. Takanami, Three-dimensionally fully space constructible functions, *IEICE Transactions on Information and Systems*, Vol.E77-D, No.6, pp.723-725, 1994.
- [25] M. Sakamoto and K. Inoue, Three-dimensional alternating Turing machines with only universal states, *Information Sciences—Information and Computer Science*, Vol.95, pp.155-190, 1996.
- [26] M. Sakamoto, T. Okazaki and K. Inoue, Three-dimensional Multicounter Automata, in *Proceedings of the 6th International Workshop on Parallel Image Processing and Analysis—Theory and Applications*, pp.267-280, 1999.
- [27] M. Sakamoto, Three-Dimensional Alternating Turing Machines, *Ph.D Thesis Yamaguchi University*, 1999.
- [28] M. Sakamoto, S. Nogami, K. Inoue and M. Kono, A relationship between the accepting powers of three-dimensional finite automata and time-bounded bottom-up pyramid cellular acceptors with three-dimensional layers, *Trans. of SCI (Japan)*, Vol.17, No.10, pp.451-458, 2004.
- [29] H. Taniguchi, K. Inoue and I. Takanami, A note on three-dimensional finite automata, *Information Sciences*, Vol.26, pp.65-85, 1982.
- [30] A. M. Turing, On computable number, with an application to the Entscheidungsproblem, *Proceedings of the London Math. Soc.*, Vol.2, No.42, pp.230-265, 1936.