アルゴリズム 30－9
(1992. 11. 20)

ある制限を加えた同期型交代マルチヘッド有限オートマトン

松野浩嗣† 井上克司‡ 高浪五男‡

†大島商船高等専門学校 ‡山口大学工学部

†742-21 山口県大島郡大島町小松 1091-1
E-mail: matsuno@oshima-k.ac.jp

‡755 山口県宇部市常盤台 2557

あらまし: Hromkovic(1992) らは 1 方向同期型交代マルチヘッド有限オートマトン (1 方向 SAMHFA) と 2 方向 SAMHFA は受理能力が等しいことを証明した. 本論文ではまず, SAMHFA の並列に走るプロセッサの個数を定数個に限定した場合, これとは異なった状況が生じ, 2 方向のものは 1 方向のものよりも受理能力が高くなることを示す. この他, SAMHFA のヘッドの個数とプロセッサの個数に基づく階層性についても述べる. 次に, 1 方向同期型交代シンプルマルチヘッド有限オートマトン (1 方向 SASPMHFA) の受理能力は 2 方向 SAMHFA の受理能力と等価であることを示す. これは先に述べた Hromkovic(1992) らの結果を強めたものになっている. さらに全称状態のみをもつ SAMHFA と SASPMHFA(SUMHFA,SUSPMHFA と略記) について 1 方向 SUMHFA は 1 方向 SUSPMHFA よりも真に受理能力が高いことも示す.

和文キーワード: マルチヘッドオートマトン, シンプルマルチヘッドオートマトン, オールタネイション

Synchronized Alternating Multihead Finite Automata with Some Restrictions

Hiroshi MATSUNO† Katsushi INOUE‡ Itsuo TAKANAMI‡

†Oshima National College of Maritime Technology ‡Yamaguchi University

†1091-1 Komatsu, Oshima-cho, Yamaguchi-Pref., 742-21, Japan
E-mail: matsuno@oshima-k.ac.jp

‡2557 Tokiwadai, Ube-shi, 755, Japan

Abstract: Hromkovic(1992) proved that one-way SAMHFAs and two-way SAMHFAs are equivalent. In this paper, we first show that when we restrict the number of processors to constant, two-way SAMHFAs are more powerful than one-way SAMHFAs. Futhermore, we examine the hierarchies of SAMHFAs based on the number of heads and the processors which run in paralell. This paper shows that one-way synchronized alternating SPMHFAs (one-way SASPMHFAs) and two-way SAMHFAs are equivalent. This is the stronger result than Hromkovic's result stated above. We also show that one-way SUMHFAs are more powerful than SUSPMHFAs, where SUMHFA(SUSPMHFA) is an SAMHFA (SASPMHFA) with only universal states.

英文 **Keywords:** Multihead automaton, Simple multihead automaton, Alternation

1 Introduction

Alternating Turing machines were proposed in [1] to model parallel computation. Informally, an alternating Turing machine is a generalization of a non-deterministic Turing machine which can, at some point during a computation, split into several processes working in parallel and independently;an input is accepted iff all parallel processes finish in accepting configurations. However, the alternating Turing machines is not a realistic model for real-world computers, because it does not allow any communication among its process.

The synchronized alternating Turing machines were introduced in [2] as a generalization of alternating Turing machines enabling a simple, natural form of communication among paralell processes in alternating computation. A synchronized alternating machine is an alternating machine with a special subset of internal states called synchronized state. Each synchronizing state is assoiated with a synchronizing symbol. If during the course of computations some process enter a synchronizing state, then it has to wait until all other processes enter accepting states or synchronizing states with the same synchronizing symbol. When this happens, all processes are allowed to continue this computation. A very interesting property of one-way finite automata is that neither two-way input nor alternation helps to increase their computational power [1]. In comparison with this fact, by adding synchronized alternation to two-way finite automata, we jump exactly two steps in Chomsky hierarchy. That is, two-way synchronized alternating finite automata recognize exactly context-sensitive languages [3].

The accepting powers of synchronized alternating multihead finite automata (SAMHFAs) was also investigated in [4], where showed that one-way SAMHFAs are as powerful as two-way ones. From this result, we know that two-way k-head SAMHFAs are less powerful than one-way (k+1)-head SAMHFAs. This result is a little surprising because no anologus result holds for any other type of multihead automata. Furthermore, it is unknown whether or not (k+1)-head one-way alternating multihead automata are more powerful than k-head ones. It is interesting to know that we can get directly from this result that (k+1)-head are more powerful than k-head for both of one-way and two-way SAMHFAs.

In this paper, we put restrictions on the function of SAMHFAs and examine the properties of these restricted type SAMHFAs. In Section 3 some properties of synchronized alternating multihead finite automata with constant leaf-sizes (SAMHFACLs) are establised. Leaf-size is the minimum number of leaves of some accepting computation tree of alternating devices. In other word, it reflects the number of processors which run in parallel in scanning a given input. SAMHFACLs are more realistic parallel computation models than ordinary SAMHFAs because of restriction of the number of process which run in parallel to constant. We show that (k+1)-head (leaf-size s+1) are more powerful than k (leaf-size s) for SAMHFAs whose leaf-size (number of head) are bounded by constant. Section 4 examines the relationship between the accepting powers of synchronized alternating simple multihead finite automata (SASPMHFAs) and SAMHFAs. SASPMHFA is an SAMHFA with the restriction that one head can sense input symbols while the others can only detect the left and right endmarkers. We show that two-way k-head SAMHFAs are less powerful than one-way (k+1)- head SASPMHFAs. This result is stronger than the known result as mentioned above.We also descrive some hierarchical properties of SASPMHFAs.

2 Preliminaries

We refer to [1] for a more formal introduction of the alternation and stress here only the notions important for the subsequent discussions. Given an alternating machine type M, we shall augment it by a finite synchronization alphabet. An internal state of such an augmented (synchronized) machines can be either an internal state of M or a pair (internal state of M, synchronizing symbol). The latter is called a synchronizing state. As usual for alternating machines, the states of M are partitioned into universal, existential, accepting and rejecting states. We use the usual notion of a configuration and the computation step relation $\vdash_M$ for the machine, and we call the configuration universal, existential, or synchronizing in the correspondence to the type of the internal state of this configuration. Initial and accepting configurations are defined as usual for the particular type of the machine. To avoid misunderstandings we give a precise definition of accepting computation trees of a synchronized alternating machine. It is a suitable subtree of the full configuration tree.

Definition 2.1. The *full configuration tree* of a synchronized alternating machine SAM M on a input word w is a (possibly infinite) labelled tree $T_M(w)$ such that

1. each node v of $T_M(w)$ is labeled by some configuration c(v) of M,
2. for the root $v_0, c(v_0)$ is the initial configuration of M on w, and
3. node v_2 is a direct descendant of node v_1 iff $c(v_1) \vdash_M c(v_2)$.

Taking all the descendants of each universal configuration and exactly one descendant of each existential configuration gives a subtree representing a computation of an alternating machine as considered usually. It can be viewed as a set of computations of independent "copies" (sometimes called "processors" in what follows) of the machine, working in paralell and splitting in universal configurations. As informal description of the use of synchronization is the following. Each time one of the machines working in parallel enters a synchronizing state, it must wait until all the other machines working in parallel enter accepting states or synchronizing state with the same synchronizing symbol. When this happens, all the machines are allowed to move from the synchronizing states. We shall now make this more precise.

Definition 2.2. The *synchronizing sequences* of a node v in a full configuration tree T with the root v_0 is the sequence of synchronizing symbols occuring in the labels of the nodes on the path from v_0 to v.

Definition 2.3. A *computation tree* of an sam M on an input word w is a (possibly infinite) subtree T' of the full configuration tree $T_M(w)$ of M on w such that

1. each node in T' labelled by a universal configuration has the same direct descendants as in $T_M(w)$,
2. each node in T' labelled by an existential configuration has at most one descendant, and
3. for arbitrary node v_1 and v_2, the synchronizing sequence of v_1 is an initial subsequnce of the synchronizing sequence of v_2 or vise versa.

Definition 2.4. An *accepting computation tree* of an sam M on an input word w is a finite computation tree of M on w such that each leaf node is labelled by an accepting configuration. We say that an sam M *accepts* an input word w if there exists an accepting computation tree of M on w. We denote the set of words accepted by M by T(M).

We then introduce 'leaf-size' which reflects the number of processors which run in parallel in scanning a given input.

Definition 2.5. Let L→R be a function, where N denotes the set of all positive integers and R denotes the set of all non-negative real numbers. For each tree t, let LEAF(t) denotes the leaf-size of t (that is ,the number of leaves of t). We say that a synchronized alternating machine sam M is L(n) leaf-size bounded if when we give an input x of length n to M there is no computation tree of M on x such that LEAF(t)> $\lceil$L(n)$\rceil$. [1]

In this paper, we consider synchronized alternating multihead finite automata (SAMHFAs) whose leaf-sizes are bounded by constant. This restriction make SAMHFAs more realistic models of real-world parallel computers.

We then put another restriction on the function of SAMHFAs. We introduce synchronized alternating simple multihead finite automata (SASPMHFAs). SASPMHFAs are SAMHFAs with the restriction that one head (called *reading head*) can sense input symbols while the others (called *counting heads*) can only detect the left endmarker '¢'and right endmarkers '$'. When the heads of SASPMHFA are allowed to sense the presence of other heads on the same input position, we call such SASPMHFA a *sensing* SASPMHFA. The reader is reffered to [6] and [7] for formal definitions of simple multihead finite automata.

In this paper, to represent the different kinds of SAMHFAs systematically, we use the notations XYk-HFA, k≥1, where

- X∈ {1,2}
  - 1: one-way
  - 2: two-way
- Y∈ $\{SA, SU, SASP, SUSP\}$
  - SA: synchronized alternating multihead automata
  - SU: synchronized alternating multihead automata with only universal states
  - SASP: synchronized alternating simple multihead automata
  - SUSP: synchronized alternating simple multihead automata with only universal state
- k-H: k-head (the number of heads is k)

For example,

- 2SAk-HFA: two-way synchronized alternating k-head finite automaton,
- 1SUSPk-HFA: one-way synchronized alternating k-head finite automaton with only universal states

Futher, we denote an L(n) leaf-size bounded XYk-HFA by XYk-HFA(L(n)). For example, we denote L(n) leaf-size bounded 1SAk-HFA by 1SAk-HFA(L(n)).

We denote the class of sets accepted by XYk-HFA and XYk-HFA(L(n)) by $\mathcal{L}$[XYk-HFA] and $\mathcal{L}$[XYk-HFA(L(n))]. For example, we denote the class of sets accepted by 1SUk-HFA by $\mathcal{L}$[1SUk-HFA].

[1] $\lceil r \rceil$ means the smallest interger greater than or equal to r.

3 Constant Leaf-Sizes

In this section, we investigate the accepting powers of synchronized alternating multihead finite automata (SAMHFAs) with constant leaf- size. That is, we restrict the number of processes of SAMHFA available (which corresponts to the number of processors) to a constant.

In [8], Matsuno et.al. introduced alternating multihead finite automata with constant leaf-sizes (AMHFACLs) and showed the following results.

- For one-way AMHFACLs,
  - (k+1 head, leaf-size s) are powerful than (k head, leaf-size s)
  - (k head, leaf-size s+1) are powerful than (k head, leaf-size s)
- Two-way AMHFACLs are powerful than one-way ones.

It is natural to ask whether similar characterizations of SAMHFACLs can be found. In order to solve this problem, we first show the following lamma. For each $X \in \{1,2\}$ and $k > 1$, XNk-HFA denotes X-way nondeterministic k-head finite automaton.

Lemma 3.1 *For each* $X \in \{1,2\}$, $k \geq 1$, *and* $s \geq 1$, $\mathcal{L}$*[XSAk-HFA(s)]=*$\mathcal{L}$*[XN(ks)-HFA].*

Proof. We demonstrate the proof for the case of one-way. (A similar idea is applicable for two-way case.) A 1N(ks)-HFA can be considered as s parallel nondeterministic k-head finite automaton with total information exchange among them. This implies $\mathcal{L}$[1SAk-HFA(s)] $\subseteq$ $\mathcal{L}$[1N(ks)-HFA]. To prove the opposite inclusion, we shall use the same technique as in Theorem 3.1 in [9].

Let A be a 1N(ks)-HFA accepting a set T(A) over an alphabet Σ. Without loss of generality, we may assume that A makes its computation step deterministically without moving any of its heads for each input word. Let Q_A be the set of states of A with the initial state q_0. The transition function of A is as follows.

$$\delta_A \subseteq (Q_A \times (\Sigma \cup \{\not\epsilon, \$\})^{ks}) \times (Q_A \times \{right, no\ move\}^{ks})$$

Futhermore let

$$b_{max}=\max_{v\in D}\{card(\{u\in Q_A\times\{right,\ no\ move\}^{ks}) |\ (v,u) \in \delta_A\})\}$$

be the upper bound on the number of nondeterministic branches (possible actions) from any input argument v∈D of A. Let us also assume that for each input argument v∈D all possible actions (which belong to $Q_A \times \{right, no\ move\}^{ks}$) from v are sequenctially ordered, i.e., we can assign to each such action its order number.

Now, let us construct a 1SAk-HFA(s) B simulating A. Let the finite set of synchronizing symbol of B be

$$S_B=\{(a_{11}, a_{12}, \ldots, a_{1k}), \ldots, (a_{s1}, a_{s2}, \ldots, a_{sk})\ , d) |$$
$$\forall i(1\leq i\leq s)\ \forall j(1\leq j\leq k)[a_{ij}\in\Sigma\cup\{\not\epsilon, \$\}\ \&$$
$$d\in\{1, \ldots, b_{max}\}\}.$$

Let $Q_B = Q_A \times \{1,2,\ldots,s\} \cup Q_A \times \{1,2,\ldots,s\} \times S_B$ be the set of states of B, and let the only one universal states of B be the initial state $(q_0, 1)$. We partition ks heads of A into s blocks (one block consists of k heads) and assign to each such block ordered number. The idea of the simulation of A by B is as follows. Assume that after reading the left endmarker '$\not\epsilon$' A goes determinisitically from the initial state q_0 to a state p without moving its heads. B simulates this step by a universal branching of B into s processes $B_1, B_2, \ldots, B_s$ and each B_i has k heads which corresponding to i-th block of heads of A, where B_i is in the state (p,i) for each $i \in \{1, \ldots, s\}$. The idea of the simulation consists in simulating the action of the i-th block of heads of A by the heads of B_i during the whole computation.

Let us assume that at some time of computation of A, A is in state q and each of i-th block of heads of A reads symbols $(a_{i1}, a_{i2}, \ldots, a_{ik})$. Futher, assume that for each $j \in \{1, \ldots, s\}$, B_j reads the symbols $(b_{j1}, b_{j2}, \ldots, b_{jk})$ in a state (q,j,t). Now, each B_j existentially choose one of at most $\Sigma^{k(s-1)}\times b_{max}$ possible actions, each corresponding to one element in the following subset, S_j, of S_B:

$$S_j=\{((a_{11}, a_{12}, \ldots, a_{1k}), \ldots, (b_{j1}, b_{j2}, \ldots, b_{jk}), \ldots,$$
$$(a_{s1}, a_{s2}, \ldots, a_{sk}), r) | \forall f(1\leq f\leq s, f\neq j)\forall g(1\leq$$
$$g\leq h)[a_{fg}\in\Sigma\cup\{\not\epsilon, \$\}]\&(r\in\{1, \ldots, b_{max}\})\}.$$

If B_i choose one action in which B_i takes the synchronizing symbol

$$t=((a_{11}, a_{12}, \ldots, a_{1k}), \ldots, (b_{j1}, b_{j2}, \ldots, b_{jk}),$$
$$\ldots, (a_{s1}, a_{s2}, \ldots, a_{sk}), r) \in S_j$$

then B_j follows the r-th nondeterministic choice of A (δ_A) for the argument

$$u=(q, (a_{11}, a_{12}, \ldots, a_{1k}), (b_{j1}, b_{j2}, \ldots, b_{jk}), \ldots,$$
$$(a_{s1}, a_{s2}, \ldots, a_{sk})).$$

Now let us show that B simulates A correctly. From the definition of the computation tree of SAM and from the fact that B has only one universal branching into $B_1, \ldots, B_s$, we see that each computation tree of B consists exactly k disjoint paths leading from the root of full configuration tree of B to some leaves, and each path corresponds to one of the nondeterministically chosen computations of some B_i. Since all the vertices of the m-th level of the full configuration tree are labelled by synchronizing configuration for $m \geq 3$, each level of a computation tree of B has to

be labelled by synchronizing configuration with the same synchronizing symbol. Thus, the only synchronizing symbol (which can be the same for all the s vertices in each level) labelled by the actual configurations of the processes $B_1, B_2, \ldots, B_s$ are synchronizing symbol in

$$\cap_{1\leq j\leq s} S_j = \{((b_{11}, b_{12}, \ldots, b_{1k}), \ldots, (b_{s1}, b_{s2}, \ldots, b_{sk}), r) | \\ r \in \{1, \ldots, b_{max}\}\}.$$

It implies that only correct guessings $(b_{11}, b_{12}, \ldots, b_{1k})$, $\ldots, (b_{s1}, b_{s2}, \ldots, b_{sk})$ of the actual argument of A can appear in any computation tree of B. On the other hand, B is able to simulate any of possible nondeterministic choices of A in the j-th step by using the synchronizing symbol $((b_{11}, b_{12}, \ldots, b_{1k}), \ldots, (b_{s1}, b_{s2}, \ldots, b_{sk}), r)$ for the r-th possible choise. Since no two vertices in the some label of computation tree of B may be labelled by two different synchronizing symbols $((b_{11}, b_{12}, \ldots, b_{1k}), \ldots, (b_{s1}, b_{s2}, \ldots, b_{sk}), u)$ $((b_{11}, b_{12}, \ldots, b_{1k}), \ldots, (b_{s1}, b_{s2}, \ldots, b_{sk}), v)$ for $u \neq v$, it is clear that all processes $B_1, B_2, \ldots, B_s$ in an computation tree of B simulate the same nondeterministic decision of A made in the each step. Clearly, in this way we have exactly one computation tree of B for each one of the computation paths of A. So, there is an accepting computation of A on the given input w iff there is an accepting computation of B on w.

Q.E.D.

In [4], it is shown that one-way SAMHFAs are as powerful as two-way ones. The theorem below shows that different situation occurs if we bound leaf-sizes of SAMHFAs.

Theorem 3.1 *For each* $X \in \{1, 2\}$, $k \geq 1$, *and* $s \geq 1$, *$\mathcal{L}$[1SAk-HFA(s)] $\subset$ $\mathcal{L}$[2SAk-HFA(s)].*

Proof. Let L(b)=$\{w_1 2 w_2 2 \cdots 2 w_b 2 w_b 2 \cdots 2 w_2 2 w_1 | \ \forall i (1 \leq i \leq b) w_i \in \{0, 1\}^*\}$. It is easily seen that L(ks(ks+1)/2) is accepted by two-way deterministirc 2-head finite automaton. On the other hand, it is shown in [10] that L(k(k+1)/2)$\notin$$\mathcal{L}$[1Nk-HFA], where 1Nk-HFA denotes a one-way nondeterministic k-head finite automaton. It follows that L(ks(ks+1)/2)$\notin$ $\mathcal{L}$[1N(ks)-HFA]. From this fact and Lemma 3.1 above this collorary completes.

Q.E.D.

One of the major problems in the thoery of automata and complexity is to determine whether additional computation resources increase the computational power of the investigeted machine. The following theorems show the results concerning to this problem for SAMHFAs cases.

Theorem 3.2 *For each* $k \geq 1$, *and* $s \geq 1$, *$\mathcal{L}$[1SAk-HFA(s)] $\subset$ $\mathcal{L}$[1SA(k+1)-HFA(s)].*

Proof. It is shown in [10] that $\mathcal{L}$[1Nk-HFA]$\subset$ $\mathcal{L}$[1N(k+1)-HFA]. From this fact we get that $\mathcal{L}$[1N(ks)-HFA]$\subset$ $\mathcal{L}$[1N(ks+1)-HFA] $\subset$ $\mathcal{L}$[1N(k+1)s-HFA] for each $k \geq 1$ and $s \geq 2$. On the other hand, from Lemma 3.1, we can show that $\mathcal{L}$[1N(ks)-HFA]= $\mathcal{L}$[1SAk-HFA(s)] and $\mathcal{L}$[1N(k+1)s-HFA]=$\mathcal{L}$[1SA(k+1)-HFA(s)]. This completes the proof of the theorem.

Q.E.D.

The following thorem can be shown by using a same technique as in the proof of Theorem 3.2 above.

Theorem 3.3 *For each* $k \geq 1$, *and* $s \geq 1$, *$\mathcal{L}$[1SAk-HFA(s)] $\subset$ $\mathcal{L}$[1SAk-HFA(s+1)].*

4 Simple Multihead Automata

The purpose of this section is to investigate the relationship between the accepting power of synchronized SASPMHFAs and SAMHFAs.

It is shown in [7] that the accepting powers of alternating multihead finite automata and the accepting powers of alternating simple multihead finite automata are equivalent. We first show that a similar result holds for synchronized alternation cases.

When the head of SASPMHFA are allowed to sense the presence of other heads on the same input position, we call such SASPMHFA *sensing* SASPMHFA. We denote one-way (two-way) k-head sensing SASPMHFA by 1SASNSPk-HFA (2SASNSPk-HFA).

Lemma 4.1 *For each* $X \in \{1, 2\}$ *and* $k \geq 1$, *$\mathcal{L}$[XSASNSPk-HFA]=$\mathcal{L}$[XSASPk-HFA].*

Proof. The proof is similar to the proof of Lemma 3.1 in [7]. The outline of the proof of Lemma 3.1 in [7] in two-way cases is as follows. If some head of 2ASPk-HFA M want to sence exisetence of other head in the same position during the simulation of 2ASNSPk-HFA N, then M enters an existential state to guess whether the heads coincide and enters a universal state to choose whether to continue the simulation of N or move the two heads to the right simultaneously, entering an accepting state if and only if both heads reach the right endmarker at the same time.

We note that the automaton under consideration here can do 'synchronization'. We call the process of computation of M to check the coincidence of head 'sensing process'. Assume that when some process P enter a synchronizing state, some sensing process continue its computation. Then the process P waits until all other processes (including sensing process) enter synchronizing state with the same synchronizing symbol. Since we can construct a sensing process which has no synchronizing states, the

process P waits until sensing process enter an accepting states.

It will be obvious that M accepts T(N). We can make a proof in one-way case by similar technique described above.

Q.E.D.

Theorem 4.1 *For each $X \in \{1,2\}$ and $k \geq 1$, $\mathcal{L}$[XSASPk-HFA]= $\mathcal{L}$[XSAk-HFA].*

Proof. To prove this by Lemma 4.1 above, it is sufficient to show that $\mathcal{L}$[XSAk-HFA] $\subseteq$ $\mathcal{L}$[XSASNSPk-HFA] for $X \in \{1,2\}$. Since the proof is similar to the proof of Theorem 3.2 in [7], we only give outline of the proof of this theorem. The action of k input heads of 2SAk-HFA N is simulated by one reading head and k-1 counting heads of 2SASPk-HFA M. M existentially guesses the symbols under counting heads, and enters a universal state to choose whether to continue the simulation of N or move the reading head to the position of one of counting heads, entering an accepting state if and only if the symbol guessed above is correct.

Q.E.D.

In [4], it is shown that $\mathcal{L}$[1SAk-HFA]=$\mathcal{L}$[2SAk-HFA] and $\mathcal{L}$[2SAk-HFA] $\subset$ $\mathcal{L}$[1SA(k+1)-HFA] for each $k \geq 1$. By using Theorem 4.1, we can strengthen this result as follows.

Collorary 4.1 *For each $k \geq 1$,*

(1) $\mathcal{L}$[1SASPk-HFA]=$\mathcal{L}$[2SAk-HFA] and

(2) $\mathcal{L}$[2SAk-HFA] $\subset$ $\mathcal{L}$[1SASP(k+1)-HFA].

We next show that the situation in different from Theorem 4.1 occurs if we exclude existential states from the state set of SAMHFA and SASPMHFA.

Theorem 4.2 *For each $k \geq 1$,*

(1) $\mathcal{L}$[1SUSPk-HFA] $\subset$ $\mathcal{L}$[1SUk-HFA].

(2) $\bigcup_{1 \leq k < \infty}$ $\mathcal{L}$[1SUSPk-HFA] $\subset$ $\bigcup_{1 \leq k < \infty}$ $\mathcal{L}$[1SUk-HFA].

Proof. Let L=$\{w2w' | \ w, w' \in \{0,1\}^*, w \neq w'\}$. It is easily seen that L is in $\mathcal{L}$[1SU2-HFA]. Below we show that L is not in $\bigcup_{1 \leq k < \infty}$ $\mathcal{L}$ [1USPk-HFA].

Suppose that there exists a 1SUSPk-HFA M for some k. Let s be the number of states of M. For each $n \geq 1$, let V(n)=$\{w2w | \ w \in \{0,1\}^*, |w|=n\}$. For each x in V(n), let S(x) be the set of semi-configurations of M defined as follows.

> S(x)=$\{(q, i_1, i_2, \ldots, i_{k-1}) |$ there exists a computation path $I_M(x) \vdash_M^* (x, n+1, (q', i'_1, i'_2, \ldots, i'_{k-1})) \vdash_M (x, n+2, (q, i_1, i_2, \ldots, i_{k-1}))$ (that is, $(x, n+2, (q, i_1, i_2, \ldots, i_{k-1}))$ is a configuration of M just after the point where the input head left the symbol "2" of x)$\}$

and let

> C(x)=$\{(\sigma_1, \sigma_2) | \sigma_1$ and σ_2 are semi-configurations in S(x) such that
>
> - (i) in case of $\sigma_1 = \sigma_2$, there exists a sequential computation of M which starts with the configuration $(x, n+2, \sigma_1)$ and either terminates in a rejecting configuration, or enters an infinite loop, and
> - (ii) in case of $\sigma_1 \neq \sigma_2$, there exist two sequential computations of M which start with the configurations $(x, n+2, \sigma_1)$ and $(x, n+2, \sigma_2)$ respectively, and terminate in sync configurations with different sync elements.$\}$

(Note that, for each x in V(n), C(x) is not empty, since x is not in L, and so not accepted by M.) Then the following proposition must hold.

[Proposition 4.1] For any two different string x,y in V(x), C(x)$\cap$C(y) $=\phi$.

[For otherwise, suppose that x=w2w, y=w'2w', C(x)$\cap$C(y)$\neq \phi$, and $\{\sigma_1, \sigma_2\} \in$ C(x)$\cap$C(y). Let z=w2w'. Since $\{\sigma_1, \sigma_2\} \in$ C(x), there exist computation paths $I_M(z) \vdash_M^* (z, n+2, \sigma_1)$ and $I_M(z) \vdash_M^* (z, n+2, \sigma_2)$. Since $\{\sigma_1, \sigma_2\} \in$ C(y), in case of $\sigma_1 = \sigma_2$, there exists a sequential computation of M which starts with the configuration $(z, n+2, \sigma_1)$ and either terminates in rejecting configuration, or enters an infinite loop, and in case of $\sigma_1 \neq \sigma_2$, there exist two sequential computation of M which starts with the configurations $(z, n+2, \sigma_1)$ and $(z, n+2, \sigma_2)$, respectively, and terminate in sync configurations with different sync elements. This means that z is not accepted by M. This contradicts the fact that z is in L=T(M).]

Let p(n) denote the number of pairs of possible configurations of M just after the point where the reading head left the symbol "2" of strings in V(n). Then,

> p(n)=K(K-1)/2+K, where K=s$\cdot n^{k-1}$.

On the other hand, $|V(n)| >$p(n) for large n, so it follows that for large n there must be two different strings x,y in V(n) such that C(x)$\cap$C(y)$\neq \phi$. This contradicts Proposition 4.1.

Q.E.D.

5 Conclusions

In this paper, we put two types of restrictions on synchronized alternating multihead finite automata (SAMHFAs) and examined the properties of there restricted SAMHFAs. One is SAMHFAs with constant leaf-sizes (SAMHFACLs) and the other is synchronized alternating simple multihead finite automata (SASPMHFAs).

It is shown in [4] that one-way SAMHFAs are as powerful as two-way ones. Theorem 3.1 shows that when we bound the leaf-size of SAMHFAs, different situation occurs, that is, two-way SAMHFAs are more powerful than one-way ones. Futhermore, Theorem 4.1 implies the stronger result than Hromkovic's result as stated above. That is, this theorem shows that one-way SASPMHFAs are as powerful as two-way SAMHFAs.

SAMHFACL is interesting model bacause it has two type parallel resources, a constant number of heads and a constant number of processors. Hromkovic [4] and Ibarra [11] gave some hierarchical properties on the number of heads of SAMHFA and hiererchical properties on the number of processes of SAMHFA, but they did not consider the interaction among these two parallel resources. Theorem 3.2 and 3.3 establish tight hierarchies on both of the number of heads and the number of processes.

To restrict the number of processes available to constant is a one way to make SAMHFA more realistic model of real-world parallel computers. Another way to restrict the power of SAMHFA is to allow only universal states so that reflect the deterministic nature of real-world computers. Theorem 4.2 gave a result on SAMHFA with only universal states (SUMHFA). Ibarra [11] showed many interesting properties of SUMHFAs. We would like to investigate the properties of SUMHFAs with constant leaf-sizes from now on.

References

[1] A.K.Chandra, D.C.Kozen and L.J.Stockmeyer, Alternation, *J.ACM 28 (1),* pp.114-133 (1981).

[2] A.Slobodova, "On the power of communication in alternating machines," *Proc. 13th MFCS'88, LNCS 324, Springer-Verlag,* pp.518-528 (1988).

[3] J.Dassow, J.Hromkovic, J.Karhumaki, B.Rovan, and A.Slobodova, "On the power of synchronization in parallel computations," *Proc. 14th MFCS'89, Lecture Notes in Computer Science, Springer-Verlag 1989.*

[4] J.Hromkovic, K.Inoue, B.Rovan, A.Slobodova, I.Takanami, and K.W.Wagner,"On the power of one-way synchronized alternating machines with small-space," *International Journal of Foundations of Computer Science,*to appear.

[5] K.Inoue, I.Takanami, A.Nakamura, and T.Ae, "One-way simple multihead finite automata," *Theoret. Comput. Sci. 9* pp.311-328 (1979).

[6] O.H.Ibarra and C.E.Kim, "A useful device for showing the solvability of some decision problems," *J. Comput. System Sci. 13* pp.153-160 (1976).

[7] H.Matsuno, K.Inoue, H.Taniguchi and I.Takanami, "Alternating simple multihead finite automata," *Theoret. Comput. Sci. 36* pp.291-308 (1985).

[8] H.Matsuno, K.Inoue, I.Takanami and H.Taniguchi, "Alternating multihead finite automata with constant leaf-sizes," *Trans. IEICE of Japan (Section E), E71, 10* pp.1006-1011 (1988).

[9] J.Hromkovic and K.Inoue, "A note on realtime one-way synchronized alternating one-counter automata," *Thoeret. Comput. Sci.* to appear.

[10] A.C.Yao and R.L.Rivest,"K+1 heada are better than k," *J.ACM, 25, 2* pp.337-328 (1978).

[11] O.H.Ibarra and N.Q.Tran, "New results concerning synchronized finite automata," *19th International Colloquium on Automata Languages, and Programming* (1992).