Theoretical Computer Science 125 (1994) 243–257
Elsevier

Some results concerning 2-D on-line tessellation acceptors and 2-D alternating finite automata

Tao Jiang*

Department of Computer Science and Systems, McMaster University, Hamilton, Ont. L8S 4K1, Canada

Oscar H. Ibarra** and Hui Wang**

Department of Computer Science, University of California, Santa Barbara, CA 93106, USA

Communicated by A. Salomaa
Received January 1992
Revised August 1992

Abstract

Jiang, T., O.H. Ibarra and H. Wang, Some results concerning 2-D on-line tessellation acceptors and 2-D alternating finite automata, Theoretical Computer Science 125 (1994) 243–257.

A two-dimensional nondeterministic on-line tessellation acceptor (2-NOTA) is a special type of real-time two-dimensional nondeterministic cellular automaton in which data flows from the upper-left corner to the lower-right corner. A two-dimensional alternating finite automaton (2-AFA) is an alternating finite automaton with a two-dimensional rectangular input whose input head can move in all four directions on the input. In this paper, we show that 2-NOTAs and 2-AFAs are incomparable. This answers in the negative an open question posed by Ito et al. (this journal, 1989). Closure properties of the classes of languages (i.e., sets of two-dimensional patterns) accepted by two-way, three-way, and four-way two-dimensional alternating finite automata and two-dimensional alternating finite automata with only universal states (2-UFAs) are also obtained which answer several open questions posed by Inoue and Takanami (1988).

Correspondence to: H. Wang, Department of Computer Science, University of Alabama, Huntsville, AL 35899, USA.

* Research supported in part by a grant from SERB, McMaster University and NSERC Operating Grant OGP 0046613.

** Research supported in part by NSF Grants CCR89-18403 and CCR90-96221.

SSDI 0304-3975(92)00010-S

1. Introduction

Blum and Hewitt [1] were the first to study finite automata and marker automata operating on a two-dimensional input tape. Since then, some new types of automata with two-dimensional input tape have been introduced. One model is the two-dimensional alternating finite automaton (2-AFA) [7] and another is the two-dimensional on-line tessellation acceptor (2-OTA) [5] which is a special type of real-time rectangular array bounded cellular automaton. There are also restricted versions of these automata, such as the two-way 2-AFA (TW2-AFA) which is a 2-AFA whose input head can only move in two directions (right and down, in addition to no move), the three-way 2-AFA (TR2-AFA) which is a 2-AFA whose input head can only move in three directions (left, right and down) and the 2-AFA with only universal states (2-UFA). Many researchers have investigated the properties and relationship between these automata ([2–4, 6, 9–11, 13]). Recently, Ito et al. [12] showed that the TW2-AFAs are equivalent to the deterministic 2-OTAs (2-DOTAs) through 180° rotation. They conjectured that the class of languages accepted by 2-AFAs is included in the class of languages accepted by nondeterministic 2-OTAs (2-NOTAs). In Section 3, we show that 2-AFAs and 2-NOTAs are incomparable, disproving their conjecture. The proof is rather interesting in that we use the planar embedding of acyclic-directed bipartite graphs to show the incomparability. In Section 4, we show that the class of languages accepted by 2-AFAs is not closed under complementation and that the class of languages accepted by TR2-AFAs is closed under complementation. These results answer two open questions in [8]. In Section 5, we show some closure properties of these automata under rotations.

2. Preliminaries

We adopt most of the definitions and notation in [5, 7, 12]. Below we give brief descriptions of the devices. For more details, the reader is referred to [5, 7, 12].

Definition 2.1. Let Σ be a finite set of input symbols. A pattern over Σ is a rectangular 2-dimensional array of symbols from Σ surrounded by boundary symbols, $\#$. The set of all input patterns over Σ is denoted by $\Sigma^{(2)}$. For a pattern x, we denote $\mathrm{row}(x)$ and $\mathrm{col}(x)$ to be the number of rows and columns of x. A pattern x is square if $\mathrm{row}(x)=\mathrm{col}(x)$. A pattern can be rotated clockwise 90°, 180° and 270° to obtain new patterns.

A two-way nondeterministic on-line tessellation acceptor (2-NOTA) M is an infinite mesh-connected array of cells. Each cell of the array consists of a nondeterministic finite-state machine. The nondeterministic finite-state machines in a given array are all identical. The cells can be identified by a pair of integers denoting its coordinate. A cell is called the (i,j) cell if its coordinate is (i,j). An input to M is

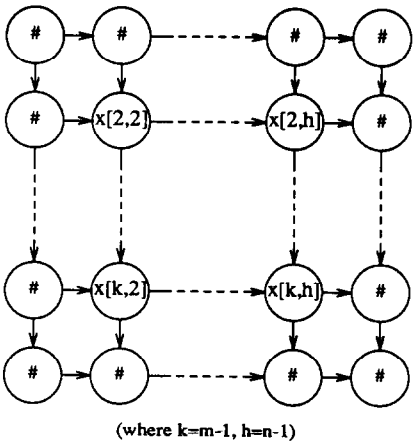


Fig. 1. A two-dimensional nondeterministic on-line tessellation acceptor.

a pattern x with symbol $x[i,j]$ placed at the (i,j) cell of the array, where $1 \leqslant i \leqslant \mathrm{row}(x)$ and $1 \leqslant j \leqslant \mathrm{col}(x)$ and subject to the condition that the boundary symbols are "#", i.e., $x[1,j]=x[\mathrm{row}(x),j]=\#$ for $1 \leqslant j \leqslant \mathrm{col}(x)$ and $x[i,1]=x[i,\mathrm{col}(x)]=\#$ for $1 \leqslant i \leqslant \mathrm{row}(x)$. The interior symbols are from Σ, see Fig. 1. The 2-NOTA M on input pattern x works as follows. At time $t=0$, all cells in M except the (1, 1) cell are in the "quiescent state" and the (1, 1) cell is in the "motive state". At time $t=1$, the (1, 1) cell enters an active state which depends on symbol $x[1,1]$. At time $t=k$ $(k>1)$, each (i,j) cell such that $(i-1)+(j-1)=k-1$ enters an active state which depends on the active states of $(i-1,j)$ cell and $(i,j-1)$ cell at time $t=k-1$, and on the symbol $x[i,j]$. We assume that if a cell's neighbor does not exist, the state of that neighbor is quiescent. M accepts input pattern x if and only if the active state of the (m,n) cell at time $t=m+n-1$ is one of the specified final states, where $m=\mathrm{row}(x)$ and $n=\mathrm{col}(x)$.

Definition 2.2. A two-dimensional nondeterministic on-line tessellation acceptor (2-NOTA) is a 7-tuple $M=(Q,E^2,\Sigma\cup\{\#\},\delta,q_e,q_0,F)$, where Q is the finite set of states, E^2 is the set of all pairs of integers, Σ is the finite set of input symbols, $\#\notin\Sigma$ is the boundary symbol, and $\delta: Q^3\times(\Sigma\cup\{\#\})\to 2^{Q'}\cup\{\{q_0\}\}$ is the state transition function, where $Q'=Q-\{q_e,q_0\}$, $q_e\in Q$ is the motive state, $q_0\in Q$ is the quiescent state, and $F\subseteq Q-\{q_e,q_0\}$ is the set of final (accepting) states.

The cell state transition function δ prescribes state transitions of cells with coordinates in E^2. The state transition function is defined as follows. Let $q_{(i,j)}(t)\in Q$ denote the state of the (i,j) cell at time t. Then

$$q_{(i,j)}(t+1)\in\delta(q_{(i,j)}(t),q_{(i-1,j)}(t),q_{(i,j-1)}(t),a),$$

where a is the input symbol on the (i,j) cell. In addition, δ has the property that, for any $a\in\Sigma\cup\{\#\}$ and any $p_i\in Q(1\leqslant i\leqslant 3)$,

$$\delta(p_1,p_2,p_3,a)=q_0 \text{ iff } p_1=q_0 \text{ and } p_2,p_3\in\{q_e,q_0\}.$$

This property allows each cell to remain in the quiescent state q_0 before being activated.

A two-way on-line tessellation acceptor is called deterministic (2-DOTA) if the finite-state machine in each cell of the array is deterministic.

Definition 2.3. Let $M=(Q,E^2,\Sigma\cup\{\#\},\delta,q_e,q_0,F)$ be a 2-NOTA and $x\in\Sigma^{(2)}$ be an input pattern to M. Then a run of M on x is a two-dimensional pattern z over $Q-\{q_e,q_0\}$ which satisfies the following conditions:

(1) $\text{row}(z)=\text{row}(x)$ and $\text{col}(z)=\text{col}(x)$;

(2) $z(i,0)=z(0,j)=q_0$ for $1\leqslant i\leqslant\text{row}(z)$ and $1\leqslant j\leqslant\text{col}(z)$;

(3) $z(1,1)\in\delta(q_e,q_0,q_0,x[1,1])$;

(4) $z(i,j)\in\delta(q_0,z(i-1,j),z(i,j-1),x[i,j])$ for $i+j\geqslant 3$, $1\leqslant i\leqslant\text{row}(z)$, and $1\leqslant j\leqslant\text{col}(z)$.

A run of M on x is a state configuration of M's computation on x.

Definition 2.4. Let $M=(Q,E^2,\Sigma\cup\{\#\},\delta,q_e,q_0,F)$ be a 2-NOTA. The language accepted by M, $L(M)$, is the set of all patterns over $\Sigma^{(2)}$ that are accepted by M, i.e.,

$$L(M)=\{x\in\Sigma^{(2)}\mid \text{there is a run } z \text{ of } M \text{ on } x \text{ such that } z(\text{row}(z),\text{col}(z))\in F\}.$$

A two-dimensional alternating finite automaton (2-AFA) is an alternating finite automaton with a two-dimensional pattern (as defined in Definition 2.1) as its input. A 2-AFA has a read-only input head attached to a finite control as shown in Fig. 2.

Definition 2.5. A two-dimensional alternating finite-automaton (2-AFA) is a six-tuple $M=(Q,\Sigma\cup\{\#\},\delta,q_0,U,F)$, where Q is the finite set of states, Σ is the finite input alphabet ($\#\notin\Sigma$ is the boundary symbol), $\delta:(Q\times(\Sigma\cup\{\#\}))\to(2^{Q\times(\text{left, right, up, down, no move})})$ is the transition function, $q_0\in Q$ is the initial state, $U\subseteq Q$ is the set of universal states, $Q-U$ is the set of existential states, and $F\subseteq Q$ is the set of final (accepting) states.

A 2-AFA is called 2-UFA if it does not have existential states.

A configuration of M is a triple $c=(x,(i,j),q)$, where x is the input pattern to M, (i,j) is the input head position and q is the state of the finite control when the input head is at position (i,j). The initial configuration of M on x is $I_M(x)=(x,(1,1),q_0)$. A configuration is called universal or existential if the state associated with it is universal or existential, respectively. A configuration is called accepting if the state associated with it is both final and halting. We assume, without loss of generality, that the input head never falls off the input tape. Let c and c' be two configurations of M. We write $c\vdash_M c'$ if configuration c' follows from configuration c in one step of M.

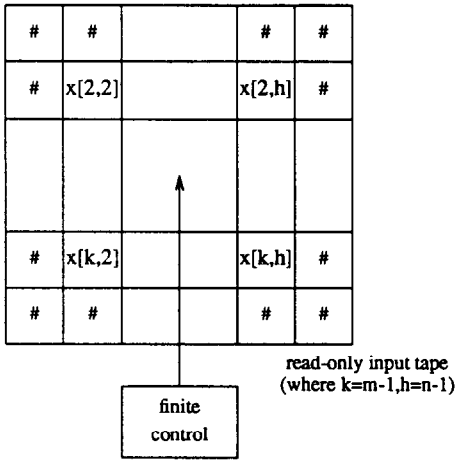


Fig. 2. A two-dimensional alternating finite automaton.

Definition 2.6. Let $M=(Q, \Sigma \cup \{\#\}, \delta, q_0, U, F)$ be a 2-AFA and x be an input pattern to M. A computation tree of M on x is a (possibly infinite) nonempty labeled tree which satisfies the following conditions:

(1) each node u is labeled with a configuration $c(u)$;

(2) if u is an internal node of the tree, $c(u)$ a universal configuration and $\{c \mid c(u) \vdash_M c\} = \{c_1, \ldots, c_k\}$, then u has exactly k children $v_1, \ldots, v_k$ such that $c(v_i) = c_i$ for $1 \leqslant i \leqslant k$;

(3) if u is an internal node of the tree and $c(u)$ is an existential configuration, then u has exactly one child v such that $c(u) \vdash_M c(v)$.

For any configuration c, a c-computation tree of M is a computation tree of M whose root is labeled with configuration c, A c-accepting computation tree of M is a finite c-computation tree whose leaves are all labeled with accepting configurations.

Definition 2.7. A three-way two-dimensional alternating finite automaton (TR2-AFA) is a 2-AFA whose input head movement is restricted to left, right, down or no move. Similarly, a two-way two-dimensional alternating finite automaton (TW2-AFA) is a 2-AFA with its input head movement restricted to right, down or no move.

Definition 2.8. A two-dimensional nondeterministic finite automaton (2-NFA) is a nondeterministic finite automaton whose input is two-dimensional and whose input head can move in all four directions. A two-dimensional deterministic finite automaton is denoted by 2-DFA.

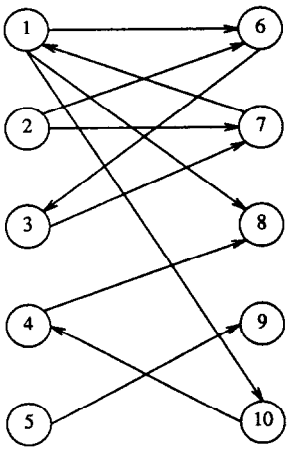
Definition 2.9. Let M be a 2-AFA (TR2-AFA, TW2-AFA, 2-UFA, 2-DFA, 2-NFA). The language accepted by M, $L(M)$, is the set of all patterns accepted by M. The class

of languages accepted by 2-NOTAs, 2-AFAs, 2-UFAs, TR2-AFAs, TW2-AFAs, 2-DFAs and 2-NFAs are denoted by £(2-NOTA), £(2-AFA), £(2-UFA), £(TR2-AFA), £(TW2-AFA), £(2-DFA) and £(2-NFA), respectively.

3. £(2-NOTA) and £(2-AFA) are incomparable

Lemma 3.1. £(2-*AFA*) $\not\subseteq$ £(2-*NOTA*).

Proof. Consider the planar embedding of directed bipartite graphs with equal number of vertices on both sides. Let $\Sigma = \{v, l, r, u, d, +, x, 0\}$ be the alphabet used for the embedding. We use the following embedding rule. The symbol v represents a vertex, symbol + means an intersection of two edges (i.e., where they join or split), symbol x is for a cross-over of two edges, symbol 0 represents a blank space, and symbols u, d, l and r are the symbols needed to form upward, downward, leftward and rightward edges, respectively. Let pattern P be a planar embedding of a directed bipartite graph with n vertices on both sides. The size of P will be $(4n+3) \times (4n+2)$ (including the boundary markers). Row $2n+2$ of P defines $2n$ vertices of a bipartite graph, where the left n vertices form one group and the right n vertices form the other. The $2n$ v's are placed such that there are two blanks separating the first n vertices from the second n vertices, and there is a blank between consecutive v's in both the left and




```
######################
#+rrrrrrrrrr+rrr+rrr+#
#u          d   d   d#
#u +rrrrrrrr+r+ d   d#
#u u        d d d   d#
#u u +rrrrrrxr+ d   d#
#u u u      d d d   d#
#u u u +rrrrxrxr+   d#
#u u u u    d d d   d#
#u u u u +rrxrxrxr+ d#
#u u u u u  d d d d d#
#v0v0v0v0v00v0v0v0v0v#
#u   u u    d d     d#
#u   +lxllll+ d     d#
#u     u      d     d#
#+llllxllllll+      d#
#      u            d#
#      u            d#
#      u            d#
#      u            d#
#      u            d#
#      +lllllllllll+#
######################
```

Fig. 3. An example of embedding: (a) A directed bipartite graph G; (b) The planar embedding of G.

right groups. The upper $2n+1$ rows of P specify the set of (directed) edges from the left group of vertices to the right group of vertices and the lower $2n+1$ rows specify the set of edges from the right group of vertices to the left group of vertices. An example of such embedding is given in Fig. 3. It is easy to see that every directed bipartite graph with equal number of vertices on both sides can be embedded in the plane following the above rule.

Consider language $L_1 = \{P \mid P \in \Sigma^{(2)}$ and P is a planar embedding of some acyclic directed bipartite graph with equal number of vertices on both sides$\}$. We show that L_1 can be accepted by a 2-AFA but not by any 2-NOTA.

We describe a 2-AFA A which accepts L_1. For a given pattern P, A first checks the correctness of embedding so that the nonboundary symbols of P are all from Σ and that the vertices are placed according to the embedding rule. A also verifies that every edge above the $(2n+2)$th row connects one vertex in the left group with one vertex in the right group and that every edge below the $(2n+2)$th row connects one vertex in the right group with one vertex in the left group. A then systematically scans the $(2n+2)$th row of the pattern P. For every vertex v encountered, A checks that each directed path leading from vertex v does not enter any loop. If P is not a planar embedding of any acyclic directed bipartite graph, A will eventually enter an infinite loop and thus will reject P; otherwise, A will accept P. Hence, L_1 can be accepted by a 2-AFA.

Suppose that L_1 is accepted by a 2-NOTA M. For each pattern P that is the embedding of an acyclic directed bipartite graph with equal number of vertices on both sides, fix an accepting computation of M on P and let $c(P)$ denote the configuration of active states of M at row $2n+2$. Clearly, there are k^{4n+2} possible $c(P)$'s, where k is the number of states of M.

Let P_U (P_L) be the upper (lower) half (i.e., $2n+1$ rows) of a pattern as described above. Note that the $(2n+2)$th row is for vertices. We fix an ordering on the edges from the right group of vertices to the left group of vertices as follows: for edges (a, b) and (c, d) in a graph, $(a, b) < (c, d)$ if a is to the left of c or $a = c$ and b is to the left of d in the embedding. We also fix an ordering on the P_L's of the patterns of the same size as follows. Let $P_L^1 = \{e_1^1, e_2^1, \ldots, e_s^1\}$ and $P_L^2 = \{e_1^2, e_2^2, \ldots, e_t^2\}$, where $e_1^1 > e_2^1 > \cdots > e_s^1$ and $e_1^2 > e_2^2 > \cdots > e_t^2$. Then $P_L^1 < P_L^2$ if there exists some m such that $m < t$, $e_1^1 = e_1^2, \ldots, e_m^1 = e_m^2$, and either $m = s$ or $m < s$ and $e_{m+1}^1 < e_{m+1}^2$.

For each P_U, let $L(P_U) = \max\{P_L \mid$ pattern P formed by P_U, $(2n+2)$th row and P_L is acyclic$\}$. $L(P_U)$ will be called the maximum match of P_U. Let $\mathrm{MAXLP}(n) = \{L(P_U) \mid$ for some P_U of size $(2n+1) \times (4n+2)\}$. The following proposition can be shown.

Proposition 3.2. $|\mathrm{MAXLP}(n)| \geq n!$.

Proof. We will construct $n!$ P_U's whose maximum matches are pairwise different. Let $n \geq 1$ be any integer and $\mathrm{VL} = \{1, 2, \ldots, n\}$ and $\mathrm{VR} = \{n+1, n+2, \ldots, 2n\}$ denote the two groups of vertices, where vertex i is placed to the left of vertex $i+1$. Let $\pi = v_1, v_2, \ldots, v_n$ be any permutation of VL. Construct a P_U, denoted by $P_U(\pi)$, as

follows:

$$P_U(\pi)=\{(v_n,n+1),(v_n,n+2),(v_n,n+3),\ldots,(v_n,2n-1),$$
$$(v_{n-1},n+1),(v_{n-1},n+2),\ldots,(v_{n-1},2n-2),$$
$$(v_{n-2},n+1),\ldots,(v_{n-2},2n-3),\ldots,$$
$$(v_2,n+1)\}.$$

Then one can easily verify that

$$L(P_U(\pi))=\{(2n,v_1),(2n,v_2),(2n,v_3),\ldots,(2n,v_n),$$
$$(2n-1,v_1),(2n-1,v_2),\ldots,(2n-1,v_{n-1}),$$
$$(2n-2,v_1),\ldots,(2n-2,v_{n-2}),\ldots,$$
$$(n+1,v_1)\}.$$

Clearly, for any two different permutations π_1 and π_2 of VL, $L(P_U(\pi_1))\neq L(P_U(\pi_2))$. Since there are $n!$ different permutations of VL, $|\mathrm{MAXLP}(n)|\geq n!$. □

Proof of Lemma 3.1 (*continued*). Let n be sufficiently large such that $n!>k^{4n+2}$. For each P_U, define $P(P_U)$ as the pattern formed by P_U, the $(2n+2)$th row and $L(P_U)$. Then there exist P_U^1 and P_U^2 such that $L(P_U^1)<L(P_U^2)$ and $c(P(P_U^1))=c(P(P_U^2))$. Now let P be the pattern formed by P_U^1, the $(2n+2)$th row and $L(P_U^2)$. Then P must also be accepted by M. Since M accepts L_1, P is acyclic. But this contradicts the definition of $L(P_U^1)$. Hence, L_1 is not accepted by any 2-NOTA. □

Lemma 3.1 answers in the negative the open question in [12, 8].

Lemma 3.3. *For every* 2-AFA A, $\bar{L}(A)$ *is accepted by a* 2-*NOTA.*

Proof. Let $A=(Q,\Sigma\cup\{\#\},\delta,q_0,U,F)$ be a 2-AFA. Define the complement 2-AFA of A to be $\bar{A}=(Q,\Sigma\cup\{\#\},\delta,q_0,Q-U,Q-F)$. That is, $\bar{A}$ is obtained by swapping the universal and existential states and the accepting and nonaccepting states of A. Note that in general, $\bar{L}(A)\neq L(\bar{A})$, since A may reject an input by entering an infinite loop.

We construct a 2-NOTA M to accept $\bar{L}(A)$. Let x be an input pattern. Given x, M tries to guess and verify the existence of a (possibly infinite) computation tree of $\bar{A}$ on x whose leaves are all labeled with accepting configurations. Let π denote the computation tree of $\bar{A}$ on x that M will guess. Let $R(i,j)$ denote the set of all states of $\bar{A}$ when its input head is at the (i,j) cell, $1\leq i\leq \mathrm{row}(x)$, $1\leq j\leq \mathrm{col}(x)$, in the guessed computation tree π. For each $q\in R(i,j)$, call $(x,(i,j),q)$ a configuration (of $\bar{A}$) represented by q. For convenience, let $R(i,0)=R(i,\mathrm{col}(x)+1)=\emptyset$, $1\leq i\leq \mathrm{row}(x)$ and $R(0,j)=R(\mathrm{row}(x)+1,j)=\emptyset$, $1\leq j\leq \mathrm{col}(x)$.

Generally, the (i,j) cell of M operates as follows. It receives the sets $R(i,j-1)$ and $R(i,j)$ from the $(i,j-1)$ cell and the sets $R(i-1,j)$ and $R(i,j)$ from the $(i-1,j)$ cell. It guesses the sets $R(i,j+1)$ and $R(i+1,j)$ and verifies that $R(i,j)$ is consistent with the neighboring sets $R(i-1,j)$, $R(i,j-1)$, $R(i+1,j)$, $R(i,j+1)$. That is, the following

conditions must hold: (a) none of the members of $R(i,j)$ represents a terminating nonaccepting configuration; (b) if $q \in R(i,j)$ and q is universal, then all immediate successors of the configuration $(x,(i,j),q)$ are represented by the states contained in $R(i-1,j) \cup R(i,j-1) \cup R(i+1,j) \cup R(i,j+1) \cup R(i,j)$; and (c) if $q \in R(i,j)$ and q is existential, then at least one of the immediate successors of the configuration $(x,(i,j),q)$ is represented by the states contained in $R(i-1,j) \cup R(i,j-1) \cup R(i+1,j) \cup R(i,j+1) \cup R(i,j)$. Also, the (i,j) cell passes the sets $R(i,j)$ and $R(i,j+1)$ to the $(i,j+1)$ cell and the sets $R(i,j)$ and $R(i+1,j)$ to the $(i+1,j)$ cell. In addition, the $(1,1)$ cell makes sure that $R(1,1)$ contains q_0.

The 2-NOTA M constructed above verifies that for every configuration in the guessed tree π, either it is a terminating accepting configuration or it is nonterminating and all (or at least one, depending on whether it is universal or existential) of its immediate successor configurations exist. In other words, M verifies that π is a (possibly infinite) computation tree of $\bar{A}$ on x whose leaves are all labeled with accepting configurations. It is easy to see that, if x is rejected by A, then there exists a (possibly infinite) computation tree of $\bar{A}$ on x whose leaves are all labeled with accepting configurations, and vice versa. Hence, M accepts $\bar{L}(A)$. □

Lemma 3.4. £(2-*NOTA*) $\not\subseteq$ £(2-*AFA*).

Proof. Suppose that £(2-NOTA) $\subseteq$ £(2-AFA). Let L_1 be the same language that we considered in the proof of Lemma 3.1. Since L_1 is accepted by a 2-AFA, $\bar{L}_1$ is also accepted by some 2-NOTA, by Lemma 3.3. Thus, by hypothesis, $\bar{L}_1$ is accepted by a 2-AFA. By Lemma 3.3, L_1 is accepted by a 2-NOTA. But we have already shown in the proof of Lemma 3.1 that L_1 is not accepted by any 2-NOTA. Hence, £(2-NOTA) $\not\subseteq$ £(2-AFA). □

Combining Lemmas 3.1 and 3.4, we have the following result, which answers in the negative the open question in [12].

Theorem 3.5. £(2-*NOTA*) *and* £(2-*AFA*) *are incomparable.*

Denote a three-way two-dimensional nondeterministic TM by TR2-NTM. Let TR2-NTM($s(n)$) stand for the class of $s(n)$ space-bounded TR2-NTMs and £(TR2-NTM($s(n)$)) be the class of languages accepted by TR2-NTM($s(n)$). It was an open question in [8] whether £(2-AFA) $\subseteq$ £(TR2-NTM(n)). Clearly, £(2-NOTA) $\subseteq$ £(TR2-NTM(n)). By Lemma 3.4, £(TR2-NTM(n)) $\not\subseteq$ £(2-AFA). It is easy to see that the proof of Lemma 3.1 still works if the 2-NOTA is replaced by a TR2-NTM($s(n)$) for any $s(n) = o(n \log n)$. Thus, L_1 cannot be accepted by any $s(n)$ space-bounded TR2-NTM and £(2-AFA) $\not\subseteq$ £(TR2-NTM($s(n)$)) for any $s(n) = o(n \log n)$. Hence, we have the following theorem, which answers in the negative the open question in [8].

Theorem 3.6. *For any function* $s(n)$ *such that* $s(n) = \Omega(n)$ *and* $s(n) = o(n \log n)$, £(*TR2-NTM*($s(n)$)) *and* £(2-*AFA*) *are incomparable.*

Note that $o(n \log n)$ is the tight bound since it has been shown that £(2-AFA) $\subseteq$ £(TR2-NTM($n \log n$)) [8]. Also, the following theorem is easy to prove.

Theorem 3.7. £(*TR2-AFA*) $\subsetneq$ £(*2-NOTA*).

Proof. We shall show in Theorem 4.4 that £(TR2-AFA) is closed under complementation. The inclusion follows from Lemma 3.3. That it is proper follows since £(TR2-AFA) $\subseteq$ £(2-AFA) and £(2-AFA) is incomparable with £(2-NOTA). □

4. Closure properties under complementation

Now we consider closure properties under complementation for 2-NOTAs, 2-AFAs, TR2-AFAs and 2-UFAs.

The following result has already been shown in [6]. Using Lemmas 3.1 and 3.3, we can give a very simple proof.

Theorem 4.1 (Inoue et al. [6]). £(*2-NOTA*) *is not closed under complementation.*

Proof. Suppose that £(2-NOTA) is closed under complementation. Let L_1 be the language used in the proof of Lemma 3.1. Since $L_1 \in$ £(2-AFA), $\bar{L}_1 \in$ £(2-NOTA) by Lemma 3.3. Thus, by the hypothesis, $L_1 \in$ £(2-NOTA). This contradicts Lemma 3.1. □

The next result answers an open question in [8].

Theorem 4.2. £(*2-AFA*) *is not closed under complementation.*

Proof. Suppose that £(2-AFA) is closed under complementation. Let L_1 be the language used in the proof of Lemma 3.1. Then $L_1 \in$ £(2-AFA). By the assumption, $\bar{L}_1 \in$ £(2-AFA). Thus, $L_1 \in$ £(2-NOTA) by Lemma 3.3. This contradicts Lemma 3.1. □

The results in [9] imply that £(TR2-UFA) is not closed under complementation. Using the above results, we can easily show that £(2-UFA) is not closed under complementation.

Theorem 4.3. £(*2-UFA*) *is not closed under complementation.*

Proof. Consider the language L_1 used in the proof of Lemma 3.1. One can easily see that the 2-AFA A that accepts L_1 has only universal states. Thus, L_1 is accepted by a 2-UFA. If £(2-UFA) is closed under complementation then $\bar{L}_1$ is also in £(2-UFA). By Lemma 3.3, L_1 is also in £(2-NOTA). But we know that $L_1 \notin$ £(2-NOTA). □

It is obvious that the class of languages accepted by TW2-AFAs is closed under complementation. We can show that the same result holds even for TR2-AFAs. This answers another open question in [8].

Theorem 4.4. *£(TR2-AFA) is closed under complementation.*

Proof. Let A be a TR2-AFA and $\bar{A}$ be the complement of A as in the proof of Lemma 3.3. Note that, in general, $L(\bar{A}) \neq \bar{L}(A)$. The problem again is that $\bar{A}$ may enter an infinite loop when it should accept. But we can construct a TR2-AFA A' from $\bar{A}$ such that $L(A') = \bar{L}(A)$. The basic idea is similar to the proof of Lemma 3.3, i.e., A' should accept when $\bar{A}$ either accepts or loops.

Let x be the input pattern. Again, A' tries to guess and verify the existence of a (possibly infinite) computation tree π of $\bar{A}$ on x whose leaves are all labeled with accepting configurations. For each i and j, let $D(i,j)$ denote the set of all states of $\bar{A}$ immediately after its input head shifts to symbol $x[i,j]$ from symbol $x[i-1,j]$, in the guessed computation tree π. For convenience, define $D(1,1) = \{q_0\}$, where q_0 is the initial state of $\bar{A}$ and $D(1,j) = \emptyset$ for all $j > 1$. For each i,j and j', let $R(i,j,j')$ be the set $\{p \mid (x,(i,j'),p)$ is a descendent of $(x,(i,j),q)$ in the tree π for some state $q \in D(i,j)\}$.

Now we describe a general step of the operations of A'. Suppose that A' has just arrived at symbol $x[i,j]$ from the above with set $D(i,j)$. Now, for each state $q \in D(i,j)$, A' tries to guess and verify that there exists a (possibly infinite) computation tree whose root is labeled with $(x,(i,j),q)$ and whose leaves are all labeled with accepting configurations. To do this, A' first guesses four sets $R(i,j,j-1)$, $R(i,j,j)$, $R(i,j,j+1)$, and $D(i+1,j)$ such that $D(i,j) \subset R(i,j,j)$. A' also makes sure that the sets $R(i,j,j-1)$, $R(i,j,j+1)$ and $D(i+1,j)$ are consistent with the set $R(i,j,j)$, i.e., the following conditions must hold: (a) none of the members of these sets represent a terminating nonaccepting configuration; (b) if $q \in R(i,j,j)$ and q is universal, then all immediate successors of the configuration $(x,(i,j),q)$ are represented by the states contained in $R(i,j,j-1) \cup R(i,j,j) \cup R(i,j,j+1) \cup D(i+1,j)$; and (c) if $q \in R(i,j,j)$ and q is existential, then at least one of the immediate successors of the configuration $(x,(i,j),q)$ is represented by the states contained in $R(i,j,j-1) \cup R(i,j,j) \cup R(i,j,j+1) \cup D(i+1,j)$. Then A' universally generates three branches: left, right, and down. The downward branch moves to symbol $x[i+1,j]$ with the set $D(i+1,j)$. The right branch moves to the symbol $x[i,j+1]$ with the sets $R(i,j,j)$ and $R(i,j,j+1)$ and guesses sets $R(i,j,j+2)$ and $D(i+1,j+1)$ such that $R(i,j,j)$, $R(i,j,j+2)$ and $D(i+1,j+1)$ are consistent with $R(q,i,j,j+1)$. Then it universally moves to the right with the sets $R(i,j,j+1)$ and $R(i,j,j+2)$ and down to the next row with the set $D(i+1,j+1)$ and so on. The left branch is symmetric.

It is easy to see that the above A' accepts x iff A rejects x. □

Corollary 4.5. (1) *Every 2-DFA can be converted to a halting one*;

(2) *Every TR2-AFA can be converted to a halting one*;

(3) *There are 2-AFAs that cannot be converted to halting ones*;

(4) *There are 2-UFAs that cannot be converted to halting ones.*

Proof. (1) A 2-DFA can be converted to a halting one by using Sipser's technique [15]. Given a 2-DFA M, modify M such that it has a unique accepting configuration. Then construct a 2-DFA N which performs a depth first search, on the finite directed graph formed by M's configurations, starting from M's accepting configuration, to determine if M's starting configuration can be reached. Since M is deterministic, the component of the directed graph which contains M's accepting configuration is a tree rooted at the accepting configuration. Hence N will halt.

(2) This is implied by the proof of Theorem 4.4.

(3) It is easy to see that halting 2-AFAs can be simulated by 2-NOTAs (by using the technique in the proof of Lemma 3.3). By Lemma 3.1, we know that there are 2-AFAs that cannot be simulated by 2-NOTAs.

(4) It is similar to the proof of (3). □

5. Closure properties under rotation

As mentioned in Definition 2.1 that input patterns can be rotated clockwise 90°, 180° and 270°. In this section, we consider some closure properties under rotation of input patterns. Note that all computations start from the upper-left corner of the input pattern.

Theorem 5.1. £(2-*DFA*), £(2-*NFA*), £(2-*UFA*), £(2-*AFA*), *and* £(2-*NOTA*) *are closed under* 90°, 180° *and* 270° *rotations.*

Proof. The claim holds trivially for 2-DFAs, 2-NFAs, 2-UFAs, and 2-AFAs. For 2-NOTAs, closure under 180° rotation is also trivial. We only have to show that £(2-NOTA) is closed under 90° rotation (270° rotation is similar).

Let $M=(Q, E^2, \Sigma \cup \{\#\}, \delta, q_e, q_0, F)$ be a 2-NOTA and x be an input pattern with $\text{row}(x)=n$ and $\text{col}(x)=m$. Let x' be the input pattern obtained through 90° rotation of x. We construct a 2-NOTA M' such that M' accepts x' iff M accepts x. The 2-NOTA M' simulates M as follows. The (i,j) cell of M' guesses the first active state of the $(i,j+1)$ cell while it receives the first active state of the $(i-1,j)$ cell. Note that the (i,j) cell, the $(i,j+1)$ cell and the $(i-1,j)$ cell of M' correspond to the $(n-j,i)$ cell, the $(n-j-1,i)$ cell and the $(n-j,i-1)$ cell of M, respectively. Thus, the first active state of the (i,j) cell of M' can be obtained by applying M's δ function. The (i,j) cell of M' then passes its guessed state to the $(i,j+1)$ cell. It also verifies the $(i,j-1)$ cell's guess and passes its state to the $(i+1,j)$ cell. M' accepts x' iff the first active state of the $(1,n)$ cell is q_0 and the first active state of the $(m,1)$ cell is in F. □

Theorem 5.2. £(TR2-AFA) *is not closed under* 90°, 180° *and* 270° *rotations.*

Proof. The proof that £(TR2-AFA) is not closed under 90° rotation was given in [10]. For the completeness of the paper, we include the proof here.

For 90° rotation, let

$$L_2=\{x[1..n,1..n]\mid x\in\{0,1,\#\}^{(2)}, x[1..n,2]=x[n..1,2], x[i,j]=0,$$
$$\text{for } 2\leqslant i\leqslant n-1 \text{ and } 2<j\leqslant n-1\}.$$

Since £(TR2-AFA) is closed under complementation, it is sufficient to show that $\bar{L}_2$ is accepted by a TR2-AFA A. Let x be an input pattern. The TR2-AFA A scans along the second column of the input pattern x. It guesses the cell $x[i,2]$ which will be the discrepancy (i.e., $x[i]\neq x[n-i+1,2]$). The symbol $x[i,2]$ is remembered. A then moves the input head along the diagonal until the middle row is reached. A enters a universal state with two branches. One verifies that it is indeed the middle row. Another moves the input head along the opposite diagonal until the second column is reached, and when this happens, A's input head will be under the cell $x[n-i+1]$. The discrepancy can then be verified. Hence, L_2 is accepted by a TR2-AFA.

Let L_2' be the language obtained through 90° rotation of L_2. It was shown in [10] that L_2' is not accepted by any TR2-AFAs. (The proof is based on the well-known crossing sequence technique.)

For 180° rotation, let

$$L_3=\{x[1..n,1..n]\mid x\in\{0,1\,\#\}^{(2)}, x[n-1,1..n]=x[n-1,n..1],$$
$$x[i,j]=0, \text{ for } 2\leqslant i<n-1 \text{ and } 2\leqslant j\leqslant n-1\}.$$

Similarly, L_3 is accepted by a TR2-AFA, but the 180° rotation of L_3 is not accepted by any TR2-AFAs.

The candidate language for 270° rotation is

$$L_4=\{x[1..n,1..n]\mid x\in\{0,1,\#\}^{(2)}, x[1..n,n-1]=x[n..1,n-1],$$
$$x[i,j]=0, \text{ for } 2\leqslant i\leqslant n-1 \text{ and } 2\leqslant j<n-1\}.$$

One can easily see that L_4 is accepted by a TR2-AFA, but the 270° rotation of L_4 is not accepted by any TR2-AFAs. □

Theorem 5.3. £(*TW2-AFA*) *is not closed under* 90°, 180° *and* 270° *rotations.*

Proof. For simplicity, we only consider 180° rotation. The 90° and 270° rotations are similar and we leave the proofs to the reader.

For 180° rotation, consider language

$$L_5=\{x\mid x\in\{0,1,\#\}^{(2)}, \text{ there is some } i\geqslant 2,\ x[2,i]=x[i,2]=1,$$
$$\text{all other entries of } x \text{ are 0, except for the boundaries}\}.$$

It can easily be shown that the 180° rotation of L_5 is accepted by a TW2-AFA.

Suppose that L_5 is accepted by TW2-AFA M. Let x be a pattern in L_5 with $\mathrm{col}(x)=m$ and $\mathrm{row}(x)=n$. Without loss of generality, assume that M moves on every step. Let T be an accepting computation tree of M on x. Consider M's operations on the first and the second column of x and let T' be the part of T corresponding to the second column of x. Note, since M is two-way (input head can only move right or down), once the input head leaves the second column, it cannot come back. Thus, T' is a subtree of T obtained by purging nodes involving columns $3, \ldots, m$. Let $S(h)$ denote the set of states at the $(h+1)$th level of T', i.e., $S(h)$ is the set of all states that M will enter after h steps when the input head is shifted to symbol $x[h, 2]$. Let k be the size of M's state set. Let n be sufficiently large (e.g., $n=2(2^{2k}+3)$) and choose $i=n/2$. Consider the following sequence of pairs of sets of states:

$$(S(2), S(i+1)), (S(3), S(i+2)), \ldots, (S(2^{2k}+2), S(i+2^{2k}+1)).$$

Clearly, since there are only 2^{2k} distinct pairs of sets of states, two of the pairs in the above sequence must be equal. Assume that $S(k)=S(k+t)$ and $S(i+k-1)=S(i+k-1+t)$. Then if we cut off levels k to $k+t$ and duplicate levels $i+k-1$ to $i+k-1+t$ in the subtree T', we still get a valid subtree. Let T'' denote the new subtree. Clearly, if we replace T' by T'' in tree T, we get an accepting computation tree for the following pattern x':

$$x'[2, i]=x'[i-t, 2], \text{ and all other interior entries of } x' \text{ are } 0.$$

But x' is not in L_5, a contradiction! □

It was shown in [12] that £(2-DOTA) is equivalent to £(TW2-AFA) through 180° rotation. Combining this result with Theorem 5.3, we have the following corollary.

Corollary 5.4. £(2-*DOTA*) *is not closed under* 90°, 180° *and* 270° *rotations.*

6. Conclusion

It is straightforward to see that all of the above results hold for square input patterns. It is still unknown whether £(2-NFA) is closed under complementation. We have shown in Section 3 that the complement of each set in £(2-AFA) is included in £(2-NOTA). Whether or not this inclusion is proper is open. It also remains an open question whether connected pictures (the definition of connected pictures is given in, e.g., [14]) can be accepted by 2-DFAs, 2-NFAs or 2-NOTAs.

Acknowledgment

The authors wish to thank Prof. K. Inoue for many helpful comments. They would also like to thank an anonymous referee for pointing out an error in the original proof of Theorem 4.4 and for suggestions which help simplify the proof of Theorem 5.3.

References

[1] M. Blum and C. Hewitt, Automata on a 2-dimensional tape, in: *Proc. IEEE Symp. on Switching and Automata Theory* (1967) 155–160.

[2] J. Hromkovic, K. Inoue and I. Takanami, Lower bounds for language recognition on two-dimensional alternating multihead machines, *J. Comput. System Sci.* **38** (1989) 431–451.

[3] O. Ibarra and R. Melson, Some results concerning automata on two-dimensional tapes, *Internat. J. Comput. Math.* **4A** (1974) 269–279.

[4] K. Inoue, A. Ito, I. Takanami and H. Taniguchi, A space hierarchy result on two-dimensional alternating Turing machines with only universal states, *Inform. Sci.* **35** (1985) 79–90.

[5] K. Inoue and A. Nakamura, Some properties of two-dimensional on-line tessellation acceptors, *Inform. Sci.* **13** (1977) 95–121.

[6] K. Inoue and A. Nakamura, Nonclosure properties of two-dimensional on-line tessellation acceptors and one-way parallel sequential array acceptors, in: *Proc. IECE of Japan Trans. (E)* (1977) 475–476.

[7] K. Inoue, I. Takanami and H. Taniguchi, Two-dimensional alternating Turing machines, *Theoret. Comput. Sci.* **27** (1983) 61–83.

[8] K. Inoue and I. Takanami, A survey of two-dimensional automata theory, *IMYCS* (1988) 21–35.

[9] A. Ito, K. Inoue, I. Takanami and H. Taniguchi, Two-dimensional alternating Turing machines with only universal states, *Inform. and Control* **55** (1982) 193–221.

[10] A. Ito, K. Inoue and I. Takanami, A note on three-way two-dimensional alternating Turing machines, *Inform. Sci.* **45** (1988) 1–22.

[11] A. Ito, K. Inoue and I. Takanami, A relationship between one-dimensional bounded cellular acceptors and two-dimensional alternating finite automata, *Informatik-Skripten* **21** (1988) 60–76.

[12] A. Ito, K. Inoue and I. Takanami, Deterministic two-dimensional on-line tessellation acceptors are equivalent to two-way two-dimensional alternating finite automata through 180°-rotation, *Theoret. Comput. Sci.* **66** (1989) 273–287.

[13] A. Ito, K. Inoue and I. Takanami, Some closure properties of the class of sets accepted by three-way two-dimensional alternating finite automata, *Trans. IEICE E* **72** (1989) 348–350.

[14] S. Selkow, One-pass complexity of digital picture properties, *J. ACM* **19** (1972) 283–295.

[15] M. Sipser, Halting space-bounded computations, in: *Proc. Symp. on Foundations of Comput. Sci.* (1978) 73–74.

[16] A. Szepietowski, On three-way two-dimensional Turing machines, *Inform. Sci.* **47** (1989) 135–147.