アルゴリズム 30－10
(1992. 11. 20)

2次元オートマトンに対するインクドットの効果

伊藤 暁, 井上 克司, 高浪 五男

山口大学 工学部

〒755宇部市常盤台2557山口大学工学部知能情報システム工学科

あらまし　最近，決定性と非決定性の領域計算量のクラスが分離するかどうかという未解決問題に関連して，インクドットチューリング機械が導入された．インクドット機械とは入力テープ上に目印としてインクドットを落すことが可能なオフライン型チューリング機械である．

本稿では，デジタル図形固有の性質に対する弱い認識機械として，インクドット有限状態機械を導入する．まず最初に，予備的な結果として各種3方向チューリイング機械により2次元インクドット有限オートマトンを模倣するために充分な領域計算量を調べる．次に，2次元インクドット有限オートマトンの基本的な性質を調べる．すなわち，インクドットオートマトンとマーカーオートマトンその他のオートマトンとの関係やインクドットの個数に関する受理能力の階層性である．最後に，2次元インクドット有限オートマトンの連結図形認識可能性の問題を考察する．

和文キーワード　インクドット機械，2次元マーカーオートマトン，オルタネイション，連結図形

The Effect of Inkdots for Two-Dimensional Automata

Akira ITO, Katsushi INOUE, and Itsuo TAKANAMI

Faculty of Engineering, Yamaguchi University

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

Abstract　Recently, related to the open problem whether deterministic and nondeterministic space (especially low-level) complexity classes are separated, inkdot Turing machine was introduced. A inkdot machine is a conventional Turing machine capable to drop a inkdot on the given input tape for a landmark.

In this paper, we introduce finite state version of inkdot machine as a weakest recognizer of the inherent properties of digital pictures, rather than Turing machine supplied with a one-dimensional working tape. We firstly investigate the sufficient spaces of thee-way Turing machines to simulate two-dimensional inkdot finite automaton, as preliminary results. Next, we investigate the basic properties of two-dimensinal inkdot automaton, i.e., the relationship of two-dimensional inkdot automata to marker or other two-dimensional automata and the hierarchy based on the number of inkdots. Finally, we investigate the recognizability of connected pictures of two-dimensional inkdot finite machines.

英文 key words　inkdot machine, two-dimensional marker automaton, alternation, connected pictures

1. Introduction

Recently, related to the historical open problem whether deterministic and nondeterministic space (especially low-level) complexity classes are separated, inkdot Turing machine was introduced in [1]. A inkdot machine is a conventional Turing machine capable to drop a inkdot on the given input tape for a landmark. Against the earlier expectation, it is proved [2,3] that nondeterministic inkdot machines are more powerful than nondeterministic ordinary Turing machines for sublogarithmic space bound.

As the well-known result, in case of two-dimensional input tapes, there is a set of square tapes accepted by nondeterministic finite automaton but not by deterministic Turing machines with sublogarithmic space bound. Thus, it makes no sense to ask the same question for two-dimensional Turing machines.

On the other hand, there is another important aspect for inkdot mechanism: we can see two-dimensional inkdot finite automaton as a weakest recognizer of the inherent properties of digital pictures. From this reason, we introduce finite state version of inkdot machine, rather than Turing machine supplied with a one-dimensional working tape. The main problem here is "what improvement of recognizability is brought about by the addition of inkdot mechanism to ordinary two-dimensional finite automaton.

We also emphasize the standpoint such that inkdot automaton is a restricted version of "marker automaton." In Section 2, we firstly give the definition of two-dimensional marker automaton. Then, the two-dimensional inkdot automaton aimed at here is introduced. We also give several basic definitions needed for our discussions.

In Section 3, we investigate the sufficient spaces of thee-way Turing machines to simulate two-dimensional inkdot finite automaton, as preliminary results.

In Section 4, we investigate the relationship of two-dimensional inkdot automata to marker or other two-dimensional automata.

In Section 5, we investigate the exsitance of the hierarchy based on the number of inkdots. It is shown that no such hierarchy exists for deterministic inkdot finite automata.

In Section 6, we investigate the recognizability of connected pictures of two-dimensional inkdot finite machines.

The final section give the several problems to be solved in the future.

2. Definitions

Definition 2.1. Let Σ be a finite set of symbols. A *two-dimensional tape* over Σ is a two-dimensional rectangular array of elements of Σ.

The set of all two-dimensional Tapes over Σ is denoted by Σ^{2+}. Given a tape $x \in \Sigma^{2+}$, $\ell_1(x)$ denotes the number of rows of x and $\ell_2(x)$ denotes the number of columns of x. If $1 \leq i \leq \ell_1(x)$ and $1 \leq j \leq \ell_2(x)$, we let x(i,j) denote the symbol in x with coordinates (i,j). Furthermore, we define

$$x[(i,j),(i',j')],$$

when $1 \leq i \leq i' \leq \ell_1(x)$ and $1 \leq j \leq j' \leq \ell_2(x)$, as the two-dimensional tape z satisfying the following:

(i) $\ell_1(z)=i'-i+1$ and $\ell_2(z)=j'-j+1$,

(ii) for each k,r $[1 \leq k \leq \ell_1(z)$ and $1 \leq r \leq \ell_2(z)]$,
$$z(k,r)=x(k+i-1,r+j-1).$$

For $x \in \Sigma^{2+}$ with $\ell_2(x)=n$, the *i*th row $x[(i,1),(i,n)]$ of x is simply denoted by $x[i,*]$.

Definition 2.2. Let k be a non-negative integer. A *two-dimensional k-marker automaton* (AMk) is a septuple

$$M = (Q, q_0, U, F, \Sigma, \{0,1\}, \delta)$$

where

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

(3) $q_0 \in Q$ is the *initial state*,

(2) $U \subseteq Q$ is the set of *universal states*,

(4) $F \in Q$ is a set of *accepting states*,

(5) Σ is a finite *input alphabet*,

(6) {0,1} is the *presence and absence signs* of markers,

(7) $\delta \subseteq ((Q \times \{0,1\}^k) \times ((\Sigma \cup \{\#\}) \times \{0,1\}^k)) \times ((Q \times \{0,1\}^k) \times ((\Sigma \cup \{\#\}) \times \{0,1\}^k) \times \Delta)$ is the *next move relation* satisfying the following (where $\# \notin \Sigma$ is the *boundary symbol* and $\Delta = \{\text{up},\text{down},\text{left},\text{right},\text{stationary}\}$ is the directional set of input head):

For any $q,q' \in Q$, $a,a' \in \Sigma$, $u=(u_1,\cdots,u_k)$, $u'=(u'_1,\cdots u'_k)$, $v=(v_1,\cdots,v_k)$, $v'=(v'_1,\cdots,v'_k) \in \{0,1\}^k$, and $d \in \Delta$, if $((q',u'),(a',v'),d) \in \delta((q,u),(a,v))$, then (i) a=a' and (ii) for each $i(1 \leq i \leq k)$, $(u_i,v_i,u'_i,v'_i) \in \{(1,0,1,0),(1,0,0,1),(0,0,0,0),(0,1,0,1),(0,1,1,0)\}$.

A state q in Q-U is said to be *existential*. The machine M has a read-only rectangular input tape with boundary symbols "#"s. If the input head falls off boundaries of the input tape, then the machine M can make no further move.

An element of $Q \times \{0,1\}^k$ is called an *extended state*. An element of $\Sigma \times \{0,1\}^k$ is called an *extended input symbol* (the set $\Sigma \times \{0,1\}^k$ itself is called the *extended input alphabet*). An extended state [q,u] represents the situation that M is in state q and M holds or does not hold the *i*th marker in the finite control, according to the value of u_i which is equal to 1 when it holds the marker. An extended input symbol [a,v] represents the situation that the input symbol on the current cell is a and the *i*th marker exists in the same place, according to the value of v_i which is equal to 1 when the marker exists.

Therefore, the condition (ii) of δ implies the following: ① When holding the marker, M can either continue to hold it or put it down on the current cell. ② When not holding the marker, and (a) if there does not exist the marker on the current cell, M cannot create a new marker, but (b) if there exists the marker on the current cell, M can either leave it alone or pick it up.

Definition 2.3. Let $M = (Q, q_0, U, F, \Sigma, \{0,1\}, \delta)$ be an AMk. An *extended input tape* $\tilde{x}$ of M is a two-dimensional tape obtained from the original input x such that (1) $\tilde{x} \in (\Sigma \times \{0,1\}^k)^{2+}$, (2) $\ell_1(\tilde{x}) = \ell_1(x)$ & $\ell_2(\tilde{x}) = \ell_2(x)$, and (3) for each $i, j (1 \leq i \leq \ell_1(\tilde{x}), 1 \leq j \leq \ell_2(\tilde{x}))$, $\tilde{x}(i,j) = (x(i,j), \mathbf{v})$, where $\mathbf{v} \in \{0,1\}^k$. That is, $\tilde{x}$ is a two-dimensional tape over the extended input alphabet each cell of which is a pair of the same symbol as in the given input and a 0-1 vector of length k to indicate the presence or absence of the k markers.

The *initial input tape* x^0 of M is an extended input tape $\tilde{x}$ such that for each $i, j (1 \leq i \leq \ell_1(\tilde{x}), 1 \leq j \leq \ell_2(\tilde{x}))$, $\tilde{x}(i,j) = (x(i,j), \mathbf{0})$, where $\mathbf{0} = (0, 0, \cdots, 0)$.

Definition 2.4. Let $M = (Q, q_0, U, F, \Sigma, \{0,1\}, \delta)$ be an AMk. A **configuration** of M on x is an element of

$$(\Sigma \times \{0,1\}^k)^{2+} \times (Q \times \{0,1\}^k) \times N^2,$$

where N is the set of all natural numbers. The first component of configuration $c = (\tilde{x}, [q, \mathbf{u}], (i,j))$ is an extended input tape of M. The second component of c is an extended state of M. The third component of c is the input head position of M. If q is the state associated with configuration c, then c is said to be universal (existential, accepting) configuration if q is a universal (existential, accepting) state. The initial configuration of M on x is

$$I_M(x) = (x^0, [q_0, \mathbf{1}], (1,1)),$$

where $\mathbf{1} = (1, 1, \cdots, 1)$.

Definition 2.5. Given $M = (Q, q_0, U, F, \Sigma, \{0,1\}, \delta)$, we write

$$c \vdash_M c'$$

and say c' is a successor of c if configuration c' follows from configuration c in one step of M, according to the transition rules δ. $\overset{*}{\vdash}_M$ denotes the reflexive transitive closure of $\vdash$. A *computation path* of M on x is a sequence

$$c_0 \vdash_M c_1 \vdash_M \cdots \vdash_M c_n \ (n \geq 1).$$

A *computation tree* of M is a nonempty labeled tree with the properties,

(1) each node π of the tree is labeled with a configuration $\ell(\pi)$,

(2) if π is an internal node (a nonleaf) of the tree, $\ell(\pi)$ is universal and

$$\{c \mid \ell(\pi) \vdash_M c\} = \{c_1, \cdots, c_k\},$$

then π has exactly k children $\rho_1, \cdots, \rho_k$ such that $\ell(\rho_i) = c_i$,

(3) if π is an internal node of the tree and $\ell(\pi)$ is existential, then π has exactly one child ρ such that

$$\ell(\pi) \vdash_M \ell(\rho).$$

An *accepting computation tree* of M on x is a finite computation tree whose root is labeled with $I_M(x)$ and whose leaves are all labeled with accepting configurations. We say that M *accepts* x if there is an accepting computation tree of M on input x. Define

$$T(M) = \{x \in \Sigma^{2+} \mid M \text{ accepts } x\}.$$

By DMk (NMk, UMk) we denote a deterministic two-dimensional k-marker automaton (a nondeterministic two-dimensional k-marker automaton, an alternating two-dimensional k-marker automaton with only universal states).

The class of sets accepted by AMks is defined as follows.

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

$\mathcal{L}[\text{DMk}]$, etc. are defined similarly.

Now, we introduce two-dimensional multi-inkdot automaton as follows.

Definition 2.6. Let k be a non-negative integer. A *two-dimensional alternating k-inkdot automaton* (AIk) is an AMk $M = (Q, U, q_0, F, \Sigma, \{1,0\}, \delta)$ whose mext move relation $\delta \subseteq ((Q \times \{0,1\}^k) \times ((\Sigma \cup \{\#\}) \times \{0,1\}^k)) \times ((Q \times \{0,1\}^k) \times ((\Sigma \cup \{\#\}) \times \{0,1\}^k) \times \Delta)$ satisfies the following:

For any $q, q' \in Q$. $a, a' \in \Sigma$, $\mathbf{u} = (u_1, \cdots, u_k)$, $\mathbf{u}' = (u'_1, \cdots u'_k)$, $\mathbf{v} = (v_1, \cdots, v_k)$, $\mathbf{v}' = (v'_1, \cdots, v'_k) \in \{0,1\}^k$, and $d \in \Delta$, if $((q', \mathbf{u}'), (a', \mathbf{v}'), d) \in \delta((q, \mathbf{u}), (a, \mathbf{v}))$, then (i) $a = a'$ and (ii) for each $i (1 \leq i \leq k)$, $(u_i, v_i, u'_i, v'_i) \in \{(1,0,1,0), (1,0,0,1), (0,0,0,0), (0,1,0,1)\}$.

That is, an AIk M is an AMk which cannot pick up the marker any more, once it has been put down on the tape.

By "DIk" ("NIk", "UIk") we denote a deterministic two-dimensional k-inkdot automaton (a nondeterministic two-dimensional k-inkdot automaton, an alternating two-dimensional k-inkdot automaton with only universal states). $\mathcal{L}[\text{AIk}]$, etc. are defined similar to $\mathcal{L}[\text{AMk}]$.

In this paper, we will use many other kinds of two-dimensional automata but omit their definitions. If necessary, see appropriate references mentioned in the text.

At end of this section, we give some useful notations concerning to two-dimensional automata study. For a two-dimensional tape T, the compliment of T is denoted by $T^{\wedge}$. Define co-$\mathcal{L} = \{T^{\wedge} \mid T \in \mathcal{L}\}$. For a two-dimensional tape x with $\ell_1(x) = m$ & $\ell_2(x) = n$, the row reflection of x, denoted by x^{RF}, is the two-dimensional tape with the same size as x and for each $i, j (1 \leq i \leq m, 1 \leq j \leq n)$, $x^{RF}(i,j) = x(m-i, j)$. Define $T^{RF} = \{x^{RF} \mid x \in T\}$ and Ref-$\mathcal{L} = \{T^{RF} \mid T \in \mathcal{L}\}$. Let P(n) be a predicate including integer variable n. We denote "$\exists$!P(n)" if the statement P(n) holds for one and only one integer n.

3. Simulation by Three-Way Machines (Preliminary Results).

In this section, we investigate the sufficient space of three-way Turing machines to simulate inkdot automata. We considered here three-way deterministic Turing machine (3DT), three-way nondeterministic Turing machine (3NT), three-way alternating Turing machine with only universal states (3UT), and three-way alternating Turing machine (3AT) [4].

Proposition 3.1. For any fully space constructible function $L(n) \geq \log n$, 3DT(L(n)) 3NT(L(n)), 3UT(L(n)), and 3AT(L(n)) can be converted to those machines which always halt, i.e., any computation of which has no loop for any input.
Proof. Omitted here.□

Lemma 3.1. For any fully space constructible function $L(n) \geq \log n$,

(1) $\mathcal{L}[3DT(L(n))] = \text{co-}\mathcal{L}[3DT(L(n))]$,
(2) $\mathcal{L}[3UT(L(n))] = \text{co-}\mathcal{L}[3NT(L(n))]$,
(3) $\mathcal{L}[3AT(L(n))] = \text{co-}\mathcal{L}[3AT(L(n))]$.

Proof. Omitted here.□

Lemma 3.2. For any fully space constructible function $L(n) \geq \log n$,

$$\text{Ref-}\mathcal{L}[3AT(L(n))] \subseteq \bigcup_{c>0} \mathcal{L}[3DT(n \cdot 2^{cL(n)})]$$

Proof. Omitted here.□

Lemma 3.3. (1) $\text{co-}\mathcal{L}[DM1] \subseteq \mathcal{L}[3NT(n \log n)]$,
(2) $\text{co-}\mathcal{L}[NM1] \subseteq \mathcal{L}[3NT(n^2)]$.

Proof. (1): It is shown in Lemma of [5] that $\mathcal{L}[DM1] \subseteq \mathcal{L}[3NT(n \log n)]$. In its proof, a 3NT M' accepting T(M) was constructed, where M is the given DM1. By careful reading of the context, it is easily seen that the machine M' satisfies the property: For any input, only one machine Md among the machines existentially branched from the original can enter an accepting state q_f when M' accepts the input. On the other hand, from Proposition 3.1, we can assume that, when M' does not accept the input, the decision machine Md enters an rejecting state q_r and all the other branching machines enter some halting states which are neither equal to state q_f nor q_r. From these facts, we convert M' to a desired machine M'' by exchanging the accepting state q_f to the rejecting state q_r, and vice versa.
(2): From Lemma in [5] and with the same technique as part (1), we can get the desired result.
□

From Lemma 3.2 and Lemma 3.3, we get the following.

Theorem 3.1. (1) $\mathcal{L}[DM1] \subseteq \mathcal{L}[3UT(n \log n)]$,
(2) $\mathcal{L}[NM1] \subseteq \mathcal{L}[3UT(n^2)]$.

Theorem 3.2. (1) $\mathcal{L}[UM1] \subseteq \mathcal{L}[3NT(n^2)]$,
(2) $\mathcal{L}[AM1] \subseteq \bigcup_c \mathcal{L}[3NT(2^{cn})]$.

Proof. The proof is omitted here. See [6].□

The next proposition will be used not only in this section but also in some places afterward.

Definition 3.1. Let M be an NIk and x be a input tape for M. Suppose that M accepts x and uses all of the k inks. Then, there exists an accepting computation path $P_M(x) = c_0 \vdash c_1 \vdash \cdots \vdash c_f$ $(f \geq 0)$, where c_0 is the initial configurations and c_f is an accepting configuration of M on x. From $P_M(x)$, we draw a sequence of k different extended input tapes $x^0, x^1, \cdots, x^k$, where x^0 is the initial input tape of M. We call x^l the *lth plane* of $P_M(x)$ $(0 \leq l \leq k)$.

The Proposition below is easily derived.

Proposition 3.2. Let M be an NIk and x be a input tape for M. Suppose that M accepts x and uses all of the k inks. For any accepting computation path $P_M(x)$ of M on x, any l $(0 \leq l \leq k)$ and any (i,j) $(0 \leq i \leq \ell_1(x)+1, 0 \leq j \leq \ell_2(x)+1)$, We can say that the number of visits to the position (i,j) on x^l of $P_M(x)$ is at most $|Q|$, where Q is the set of states of M.

Proof. Suppose that, on some plane x^l of $P_M(x) = c_0, c_1, \cdots, c_f$, M visits some position (i,j) more than $|Q|$ times. Then, there exists a configuration c in $P_M(x)$ such that $c_0 \vdash \cdots c \vdash \cdots c \vdash \cdots \vdash c_f$. By removing the subpath (a cycle) from c to c, we get another valid accepting computation path $c_0 \vdash \cdots c \vdash \cdots \vdash c_f$ which is shorter than the original and has no repetition of c.□

In the theorem below, we will show that two-dimensional inkdot automata can be simulated by *two-dimensional on-line tessellation acceptors* (ota) [7]. To simplify our discussion, we adopt "*two-dimensional multipass on-line tessellation acceptor* (mpota)" [8] in stead of ota itself. A mpota is an extension of ota that outputs a 'run' (a two-dimensional tape consisting of the resulting states after one transition of each finite state machines on the cellular space) at each pass regarding the previous run as its own input. The mpota repeats such passes until its accepting cell enters an accepting state.

It is known that mpota's whose passes are restricted to some constant k, denoted by mpota(k)s, accept the same set as ordinary ota's:

Lemma 3.4.[8] $\mathcal{L}[ota] = \bigcup_{k \geq 1} \mathcal{L}[mpota(k)]$.

Theorem 3.3. $\bigcup_{k \geq 0} \mathcal{L}[NIk] \subseteq \mathcal{L}[ota]$.

Proof. Let M be a NIk and x be an input for M. From Proposition, we can assume that for any $l (1 \leq l \leq k)$, the number of visits of M to any cell (i,j) of the lth plane x^l is a constant.

We construct a (k+1)-pass mpota M' accepting

T(M) as follows: At the first pass, M' simulates M from the initial configuration to the configuration in which M drops the first ink. In general, at the lth pass, M' simulates M from the configuration in which M drops the l-1st ink to the configuration in which M drops the lth ink. At the last k+1st pass, M' simulate M from the configuration in which M drops the kth ink to an accepting configuration (if any). If the process above succeeds, then M' accepts x. It is clear that T(M')=T(M). From this and Lemma 3.4, the theorem follows.□

Corollary 3.1. For each $k\geq 1$, the necessary and sufficient space of 3NT to simulate NIk is n.

Proof. Omitted here.□

The following is a key lemma to derive the properties of UIk.

Theorem 3.5. For each $k\geq 0$, co-$\mathcal{L}$[UIk]$\subseteq\mathcal{L}$[NIk+1].

Proof. Without loss of generality, we assume that, when a UIk M rejects the given input x, it always enters a loop. From this, we can say that M does not accept x iff there exists a computation path $R_M(x)=c_0\vdash c_1\vdash\cdots c\vdash\cdots c\vdash$, where the cycle $c\vdash\cdots\vdash c$ represents a loop. NIk+1 M' uses k inks for the trace of subpath $c0\vdash\cdots\vdash c$ and the last ink for the detection of a loop. Note that, M' never enters an accepting state, if any computation of M has no loop.□

Corollary 3.2. co-$\mathcal{L}$[UF]$\subseteq\mathcal{L}$[NI1].

Corollary 3.3. For each $k\geq 1$, the necessary and sufficient space of 3UT to simulate UIk is n.

We describe a sub-procedure used in the proof of Theorem 3.6 below.

Lemma 3.5. $\mathcal{L}$[NF]$\subseteq\mathcal{L}$[3AT(log n)].

Proof. Let M be a NF and x be a two-dimensional tape. For two configuration c,d of M whose input heads are located in the same row of x, let denote "$c\models d$" when (i) the relation $c\overset{*}{\vdash}d$ holds and (ii) for any configuration between c and d, the input head of M never goes up beyond the row of c,d. For simplicity, we assume that it enters an accepting state in leftmost and uppermost position of x, if M accepts x. Then, the acceptance of M is equivalent to the validity of $c_0\models c_f$, where c_0 and c_f are the initial configuration and an accepting configuration of M on x, respectively. A 3AT(log n) M' can verify the relation $c\models d$ as follows [9]. Suppose that two configuration c and d of M locate now the ith row of x At first, M' guesses some other configuration e of M which locates on the ith row. Next, M' universally branches into two machines M_1, M_2. The machine M_2 recursively verifies the relation $e\models d$. The machine M_1 existentially branches into two machines, one to check that the relation $c\overset{*}{\vdash}e$ holds where the intermediated configurations are only those of the ith row. The other machine guesses two configuration c',d' of the i+1st row such that $c\vdash c'\models d'\vdash d$ and goes down to the next row to verify $c'\models d'$. It is clear that T(M')=T(M) and M' is log n space bounded.□

Theorem 3.6. $\bigcup_{k\geq 0}\mathcal{L}$[NIk]$\subseteq\mathcal{L}$[3AT(log n)].

Proof. Let M be a NIk and x be an input for M with $\ell_1(x)=m$ and $\ell_2(x)=n$. In order to simplify our argument, we assume that M uses all of the k inks.

Suppose M accepts x and let PM(x) be an accepting computation path of M on x. From proposition, it follows that the number of the configurations of $P_M(x)$ in which M either drops the ink on the tape or passes over the ink positions is at most $\Sigma_{i=1}{}^{k}(i\cdot s)$, where s is the number of states of finite control of M. Let

$$c_{10},c_{11},\cdots,c_{1r_1},c_{20},c_{21},\cdots,c_{2r_2},\cdots,$$
$$c_{k0},c_{k1},\cdots,c_{kr_k}$$

be the extracted configurations in this way, where c_{l0} is the configuration in which M just drops the lth ink and r_l is a constant such that $r_l\leq l$ $(1\leq l\leq k)$. Let $r=\Sigma(r_l+1)+1$ and let $P_0,P_1,\cdots,P_r$ be the all subpaths "$c_0\overset{*}{\vdash}c_{10}$","$c_{10}\overset{*}{\vdash}c_{11}$",$\cdots$, "$c_{1r_1-1}\overset{*}{\vdash}c_{1r_1}$","$c_{1r_1}\overset{*}{\vdash}c_{20}$","$c_{20}\overset{*}{\vdash}c_{21}$",$\cdots$,"$c_{kr_k-1}\overset{*}{\vdash}c_{kr_k}$", and "$c_{kr_k}\overset{*}{\vdash}c_f$", where c_0,c_f is the initial and an accepting configurations of M, respectively.

Below, we construct a 3AT M' simulating M on x. While the simulation, M' simultaneously performs the two tasks, one is guesses of the k positions $p_1,\cdots,p_k$ where M drops the k inks, and the other is r checks of the existence of $P_1,P_2,\cdots,$and P_r.

In the algorithm, we must use a list *remain_ink* to represent the set of inks which have not yet been located and use 2r pairs of variables (*last_q*$_\ell$,*last_j*$_\ell$) and (*first_q*$_\ell$,*first_j*$_\ell$) to represent the extended state and column number pair of the configuration in which M moves on the current row at first and at last between the way of P_ℓ , respectively.

For each ℓ $(1\leq\ell\leq r)$, do the following: Guess an extended state and column number pair in starting node configuration of P_ℓ . Initialize (*last_q*$_\ell$,*last_j*$_\ell$) to this pair. Also guess an extended state and column number pair in goal node configuration of P_ℓ . Initialize (*first_q*$_\ell$, *first_j*$_\ell$) to this pair.
Initialize *remain_ink* to {1,2,$\cdots$,k}.
Fore i=0 to m+1 do the following:

```
1. Go to the $i$th row (when i=0, assume the
   boundary symbols #s on the 1st row).
2. For each I in remain_ink do the following:
    (1) Guess whether the Ith ink does lo-
        cate at the current row. If yes, guess
        also the real location of I and delete
```

the element I from *remain_ink*.

(2) For each subpath P_ℓ either (or both) end node e of which is associated with I, do step ① if the above guess is yes, else do step ②. (Below, we only describe the case when e is the start node of P_ℓ and insert in brackets the case when e is the goal node.)

① Check $e \models (x^0, last\_q_\ell, (i, last\_j_\ell))$ $[(x^0, first\_q_\ell, (i, last\_j_\ell)) \models e]$.

② Guess a configuration s [g] on the current row and a configuration s' [g'] on the next row such that $s' \vdash s$ $[g \vdash g']$. Check $s \models (x^0, last\_q_\ell, (i, last\_j_\ell))$ $[(x^0, first\_q_\ell, (i, first\_j_\ell)) \models g]$. And, rewrite $(last\_q_\ell, last\_j_\ell)$ $[(first\_q_\ell, first\_j_\ell)]$ to the state and column number pair of s' [g'].

In step ① and ② above, the check $c \models d$ for some configurations c,d is achieved by the method of Lemma 3.5. A universal branching must be carried out if two checks of this kind are needed simultaneously. It is clear that T(M')=T(M) and M' is log n space-bounded.□

Corollary 3.4. $\bigcup_{k\geq 0}\mathcal{L}[NIk] \subsetneq \mathcal{L}[ota]$.

Proof. Omitted here.□

Corollary 3.5. For each $k\geq 1$, the necessary and sufficient space of 3AT to simulate NIk is log n.

From Theorem 3.6 and Lemma, we get the following.

Corollary 3.6. $\bigcup_{k\geq 0}\mathcal{L}[UIk] \subseteq \mathcal{L}[3AT(\log n)]$.

Corollary 3.7. $\bigcup_{k\geq 0}\mathcal{L}[UIk] \subseteq \mathcal{L}[3UT(n^k)]$.

4. Relationship to Marker and other automata.

In this section, we investigate the accepting power of inkdot machines and compare with those of marker and other automata.

Theorem 4.1. (1) $\mathcal{L}[NI1] \subsetneq \mathcal{L}[NM1]$,
(2) $\mathcal{L}[UI1] \subsetneq \mathcal{L}[UM1]$.

Proof. Omitted here.□

Theorem 4.2. $\mathcal{L}[NM1] \subseteq \mathcal{L}[AI1]$.

Proof. Let M be a NM1 and x be an input for M. We construct a AI1 M which acts as follows. While M does not pick up the marker, M' behaves in the same way as M. When M will put down the marker at a configuration $c=(\tilde{x},(q,(i,j)))$, M' first guesses the configuration $d=(\tilde{x},q',(i,j)))$ at which M would pick up the marker again and memorizes the state q' in the finite control. Next, M' universally branches into two machines, one of which continues to simulate behaviors of M after the configuration d, and the other of which verifies the validity of the guess $c \models d$. To the latter end, M' drops an ink as a landmark in this place. It is clear that T(M')=T(M).□

In the following, we show that *two-dimensional alternating finite automata with constant k leaves* (AF(k)) [6] can be simulated by multi-inkdots nondeterministic finite automata. Intuitively, "leaf-size" is the minimum number of leaves among possible accepting computation trees of an alternating machine on the given input.

Theorem 4.3. $\mathcal{L}[AF(k)] \subseteq \mathcal{L}[NI2^{k-1}-1]$.

Proof. Let M be an AF(k) and x be an input for M. Without loss of generality, we assume that any computation tree of M is a binary tree, i.e., out-degree of computation tree is at most two.

When M behaves existentially, A inkdot automaton M' behaves in the same way as M. When M branches universally at a configuration c, M' begin to performs a depth-first search of computation subtree of M rooted at c. To complete the task, M' drops a ink in this place to be a landmark when it return here in backtrack of the depth-first search. detailed construction of M' is omitted here.

We next see how many inks must be used by M'. In general, if the number of leaves of a binary tree t is k, the depth of t is at most k-1. Consequently, the number of the non-leaf nodes is at most $2^{k-1}-1$. From this fact, it follows that M' uses at most $2^{k-1}-1$ inks in our algorithm.□

This result is useful to derive the properties of AF(k).

Corollary 4.1. $\mathcal{L}[AF(2)] \subseteq \mathcal{L}[NI1]$.

Corollary 4.2. $\bigcup_{k\geq 1}\mathcal{L}[AF(k)] \subseteq \mathcal{L}[ota]$.

5. Hierarchy Based on the Number of Inkdots

In this section, we show that no hierarchy based on the number of inkdots exists for deterministic machines.

Theorem 5.1. $\mathcal{L}[DF] = \bigcup_{k\geq 1}\mathcal{L}[DIk]$.

Proof. The simulation method of Ref.[1], which only treats one-ink machines, is also valid in our two-dimensional case. We recall this technique for the beginning: DF M' behaves in the same way as DI1 M until M uses its own ink. When M drops its ink, M' memorizes the input symbol on the current cell. After that, M' continues to simulate M, except when M' encounters the same input symbol as recorded symbol in the finite control. When such a case happens, M' memorizes the current state of M and performs "depth-first backward search" of the computation toward the initial configuration of M, in order to test whether the encountered cell must be the place of ink drop. If M' reaches the initial configuration of M, it performs "foreword simulation" of M until M will drop the ink. After that, M' continues to simulate M from the memorized state as if there is a ink in this place.

Based on this, we will prove the theorem by induction on the number of inkdots. That is, assuming that DFs can simulate DIk's, we will show that a DIk M' can simulate a given DIk+1 M. Before M will drop the k+1st ink, each time when M' encounters a previously dropped ink, it memorize the pair of ink and current state as an anchor like the initial configuration described above. When M drops the k+1st ink, M' memorizes the current input symbol in the finite control. After that, M' continues to simulate M, except when M' encounters the same input symbol as recorded symbol in the finite control. When such a case happens, M' performs "depth-first backward search" and "foreword simulation" of M between the current configuration and the most recent configuration that encountered the inkdots represented by the newest pair in the finite control.□

6. Recognizability of Connected Pictures

In this section, we investigate the ability of two-dimensional inkdot automaton to recognize some properties of two-dimensional binary pictures.

Definition 6.1. Let x be a two-dimensional tape over {0,1}. An 1-component of x is the maximum subset P of $N \times N$ satisfying the following:

(i) For each $(i,j) \in P$, $1 \leq i \leq \ell_1(x)$, $1 \leq j \leq \ell_2(x)$, and $x(i,j)=1$,

(ii) for any $(i,j),(h,k) \in P$, there exists a sequence $(i_0,j_0),(i_1,j_1),\cdots,(i_n,j_n)$ of elements of P such that

(a) $(i_0,j_0)=(i,j),(i_n,j_n)=(h,k)$, and

(b) for each $m (1 \leq m \leq n)$,

$$|i_m - i_{m-1}| + |j_m - j_{m-1}| \leq 1.$$

If $x \in \{0,1\}^{(2)}$ has exactly one 1-component, we say that x is a connected picture. The set of all connected pictures is denoted by T_c. Similar to 1-component, 0-component of x can be defined. Any 0-component not adjacent to boundary #s is called a hole of x. If x has no hole, we say that x is a simply-connected picture. The set of all simply-connected pictures is denoted by T_{sc}.

Definition 6.2.[10] Let x be a two-dimensional tape over {0,1}. The *tail point* of x is the unique position (i_0,j_0) such that (1) $x(i_0,j_0)=1$ and (2) for each $(i,j) (\neq (i_0,j_0)$, $1 \leq i \leq \ell_1(x), 1 \leq j \leq \ell_2(x))$ with $x(i,j)=1$, it holds that

$$i_0 > i \text{ or } (i_0 = i \ \& \ j_0 > j),$$

i.e., $i \cdot \ell_2(x) + j < i_0 \cdot \ell_2(x) + j_0$.

Further, we need some preliminaries to get the desired results. The reader should recall the standard border-following procedure on binary pictures [11].

Definition 6.3. Let C denote a closed simple curve in 2-dimensional Euclidean space composed entirely of a finite number of horizontal and vertical line segment.

Let *walk*(C) be a walk from some point p_0 of C to p_0 that begins at p_0, initially goes in either direction, and continues in the same direction along C until p_0 is reached. Define

turn⟨*walk*(C)⟩=[total number of right hand turns - total number of left hand turns that must be made in a *walk*(C)]

± 4 Lemma.[10] *turn*⟨*walk*(C)⟩=+4 if *walk*(C) is homotopic to a clockwise loop. *turn*⟨*walk*(C)⟩=-4 if *walk*(C) is homotopic to a counterclockwise loop.

Here, we convert "Mod 3 Corollary" in [10,12] to more convenient form.

Proposition 6.1. The component surrounded by the closed curve C is a hole iff [*turn*⟨*walk*(C)⟩=1 mod3 & symbol '1's are seen in the left hand side on the walk] or [*turn*⟨*walk*(C)⟩=2 mod3 & symbol '1's are seen in the right hand side on the walk]

Theorem 6.1. $T_c\hat{}, T_{sc}\hat{} \in \mathcal{L}$[NI1].

Proof. Let x be a two-dimensional tape over {0,1}. We construct an NI1 M accepting $T_c\hat{}$ as follows. First, M existentially chooses some 0-1 border point p_0, and drop an inkdot. Next, it resets a mod3 counter and begins to perform the border-following algorithm which starts from p_0. In this walk, M increments the mod3 counter at each turn of direction. If M meets the tail position of x along the walk, then it halts in a non-accepting state. If M does not meet the tail position on the walk and finally returns to the starting point p_0, it follows that the component surrounded by C is either a hole or a 1-component different from the 1-component on which the tail position locates. The latter shows that x is not connected. M enters an accepting state only when the component is not a hole, which is easily verified by the criterion of Proposition 6.1. It is clear that $T(M)=T_c\hat{}$.

In case of $T_{sc}\hat{}$, M enters an accepting state even if there is a hole in the input.□

Corollary 6.1.() $T_c\hat{}, T_{sc}\hat{} \in \mathcal{L}$[ota].

Theorem 6.2. $T_c \in \mathcal{L}$[UI1].

Proof. In the previous theorem, the constructed NI1 M accepting $T_c\hat{}$ always halts, i.e., it never enters a loop for any time for any input. Based on this fact, it follows that, by exchanging the existential states of M to universal states, non-accepting states to accepting states, and accepting states to non-accepting states conversely, we obtain a UI1 accepting T_c. (we can get the same result by parallelization of the raster scan method of deterministic 1-marker in [13]) □

Corollary 6.2. $T_c, T_c\hat{} \in \mathcal{L}[3AT(\log n)]$.

7. Final Remarks

We conclude this paper giving the open problems concerning two-dimensional inkdot automata.

- $\mathcal{L}[NIk]$ is not closed under intersection?
- $co\text{-}\mathcal{L}[NF] \subseteq \cup \mathcal{L}[AIk]$?
- $co\text{-}\mathcal{L}[UI2] \subseteq \mathcal{L}[NI2]$?

References

[1] D.Ranjan,R.Chang,and J.Hartmanis, Space bounded computations: review and new separation results, *Theoret.Comput.Sci.* **80**, pp.289-302,(1991)

[2] V.Geffert, Nondeterministic computations in sublogarithmic space and space constructibity, *SIAM J.Comput.* **20**, No.3, pp.484-498 (1989).

[3] K.Inoue,A.Ito, and I.Takanami, A Note on 1-inkdot Alternating Turing Machines with Small Space, *Tech.Rep.IEICE Japan* **COMP92-35**, pp.55-61(1992).

[4] A.Ito,K.Inoue,and I.Takanami, The simulation of two-dimensional one-marker automata by three-way Turing machines, *Inter. J. of Patt. Recog. & Art.Intel.* **3**, No.3&4, pp.393-404 (1989).

[4] A.Ito,K.Inoue,I.Takanami,and H.Taniguchi, Two-Dimensional Alternating Turing Machines with Only Universal States, *Inform. & Contr.* **55**, Nos1-3 (1982) 193-221.

[5] A.Ito,K.Inoue,and I.Takanami, The simulation of two-dimensional one-marker automata by three-way Turing machines, *Inter. J. of Patt. Recog. & Art.Intel.* **3**, No.3&4, pp.393-404 (1989).

[6] A.Ito,K.Inoue,and I.Takanami, Constant leaf-size hierarchy of two-dimensional alternating Turing machines, *2nd Int.Conf. on Para.Image. Ana.*, to appear in *Lect.Notes on Comp.Sci.*

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

[8] K.Inoue and A.Nakamura, Two-Dimensional multipass on-line tessellation acceptors, *Inform. & Contr.* **41**, 305-323 (1979).

[9] S.A.Cook, Characterization of pushdown machines in terms of time-bounded computers, *J.Ass.Comp.Mach.* **18**, No.1, pp.4-18 (Jan. 1971).

[10] M.Blum and D.Kozen, On the power of the compass (or why mazes are easier to search than graphs), *Proc. 19th Ann. Sym. on Found. of Comp. Sci.*, pp.132-142 (1978).

[11] A.Rosenfeld, *Picture Languages -- Formal models for Picture Recognition*, Academic Press (1979).

[12] K.Nakazono,K.Morita,and K.Sugata, Accepting Ability of linear time nondeterministic bottom-up pyramid cellular automata, *Trans. IEICE Japan* Vol.**J71-D**, No.2 pp.458-461 (1988).

[13] M.Blum and C.Hewitt, Automata on a two-dimensional tape, *IEEE symp. of Switching and Automata Theory* (1967) 155-160.