Towards more expressive 2D deterministic automata

Violetta Lonati[1] and Matteo Pradella[2]

[1] Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano
Via Comelico 39/41, 20135 Milano, Italy – lonati@dsi.unimi.it
[2] Dipartimento di Elettronica e Informazione, Politecnico di Milano
Via Ponzio 34/5, 20133 Milano, Italy – matteo.pradella@polimi.it

Abstract. REC defines an important class of picture languages that is considered a 2D analogous of regular languages. In this paper we recall some of the most expressive operational approaches to define deterministic subclasses of REC. We summarize their main characteristics and properties and try to understand if it is possible to combine their main features to define a larger deterministic subclass. We conclude by proposing a convenient generalization based on automata and study some of its formal properties.

1 Introduction

Generalizing string languages and related approaches to two dimensions has always been tempting, as pictures are an important part of our life, as well as text. One of the most successful classes of picture languages introduced in the literature is surely REC, the class of *tiling recognizable* languages [1], that aims at generalizing to 2D the class of regular string languages. REC is a robust class that has various characterizations: e.g. in terms of *online tessellation acceptors* [2], *tiling systems* [3], or *Wang systems* [4].

Unfortunately, many good properties of string languages are modified or are lost in the transition towards the two dimensions. One such property is related to *determinism*: all the proposed 2D analogous of regular languages lose expressivity if constrained to deterministic models.

Essentially two approaches are proposed in the literature for defining a deterministic model of finite state automaton within REC. The first one is presented in the seminal work [5], which clearly predates REC and actually define a subclass. In this approach, the input picture is seen as a read-only tape that can be visited freely, and finite states are exploited to propagate information (see also [6] for an account of the main properties of the model).

Another, orthogonal approach is based on fixing a *scanning strategy* to visit the input picture, and allowing to add marking information to its pixels, so that it is possible to propagate information locally. The first of such models is the one of the deterministic online tessellation acceptors (or DOTAs), a kind of cellular automata [2]. This approach was then generalized and extended by the following subclasses and models, presented in chronological order: class Diag-DREC [7,8], the closure by rotation of the class defined by DOTAs; *tiling automata* [9]; snake-deterministic tiling systems and class Snake-DREC [10], in which scanning strategies follow a boustrophedonic order; then the more recent *μ-directed Wang automata* and class Scan-DREC [11,12,13].

The literature has already considered and studied the relation among subclasses within the same basic approach, but our knowledge of the actual general situation is still quite partial. The aim of this paper is to consider and analyze the inherent characteristics of these two main families of approaches, in order to get a clearer idea of the picture (no pun intended), and obtain a larger deterministic class, yet still in REC.

The paper has the following structure: it first presents the preliminaries and basic notions. In Section 3 it then considers how to add expressivity and the problem of remaining within REC. In it a new deterministic model of automaton is presented. To conclude, Section 4 studies some properties of the model.

2 Preliminaries

The following definitions are taken and adapted from [1]. Let Σ be a finite alphabet. A two-dimensional array of elements of Σ is a *picture* over Σ. A picture having n rows and m columns has *size* (n, m). $\# \notin \Sigma$ is used when needed as a *boundary symbol*; $\hat{p}$ refers to the bordered version of picture p. For instance

$$p = \begin{array}{|c|c|c|}\hline p(1,1) & \dots & p(1,m) \\ \hline \vdots & \ddots & \vdots \\ \hline p(n,1) & \dots & p(n,m) \\ \hline \end{array} \qquad \hat{p} = \begin{array}{|c|c|c|c|c|}\hline \# & \# & \dots & \# & \# \\ \hline \# & p(1,1) & \dots & p(1,m) & \# \\ \hline \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline \# & p(n,1) & \dots & p(n,m) & \# \\ \hline \# & \# & \dots & \# & \# \\ \hline \end{array}.$$

A *pixel* is an element $p(i, j)$ of p. We call (i, j) the *position* in p of the pixel. We say that $(i-1, j)$, $(i+1, j)$, $(i, j-1)$, and $(i, j+1)$ are *adjacent* to position (i, j).

The set of all pictures over Σ is Σ^{++}. A picture language is a subset of Σ^{++}. If D denotes some kind of picture-accepting device, then $\mathcal{L}(\mathrm{D})$ denotes the class of picture languages recognized by such devices.

We will sometimes consider the 90° clockwise *rotation*, the *horizontal mirror*, and the *vertical mirror* of a picture p. E.g. if $p = \begin{array}{|c|c|}\hline a & b \\ \hline c & d \\ \hline \end{array}$, then $\begin{array}{|c|c|}\hline c & a \\ \hline d & b \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline c & d \\ \hline a & b \\ \hline \end{array}$, and $\begin{array}{|c|c|}\hline b & a \\ \hline d & c \\ \hline \end{array}$ are its rotation, horizontal mirror and vertical mirror, respectively. Naturally, the same operations can be applied to languages, and classes of languages, too.

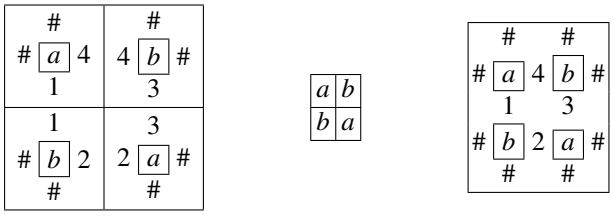
2.1 Tiling recognizable picture languages

An important class of two-dimensional languages is REC, i.e., the class of *tiling-recognizable languages*, originally defined in terms of tiling systems [3]. Another equivalent definition [14] is given by using online tessellation acceptors, OTAs, first introduced in [2]. Here we define REC by using the equivalent notation introduced in [4], which is based on a variant of Wang tiles.

Labeled Wang tiles. Let Σ be a finite alphabet and K be a set of colors, containing the special color # representing borders. A *labeled Wang tile* (or *tile* for short) is a unitary square with colored edges and a *label* in Σ. Formally, a tile is an element

$A = (a, t, l, r, b) \in \Sigma \times K^4$, where t, b, r, l represent the colors at top, bottom, right and left edges, respectively. For better readability, we represent labeled Wang tiles as $A = \begin{matrix} & t & \\ l & \boxed{a} & r \\ & b & \end{matrix}$. For any direction $d \in Dirs = \{\uparrow, \rightarrow, \leftarrow, \downarrow\}$, A_d is the color of the edge of A towards direction d. We also use $-d$ for referring to the direction opposite to d. The set of tiles with labels in Σ and colors in K is Σ_{4K}. We also consider *partial* tiles, where some colors may be undefined: the set of partial tiles is denoted by Σ_K. The *domain* of a tile A is the set Δ_A of directions where A is defined. Given two partial tiles A, B bearing the same label, we say that B extends A if $B_d = A_d$ for every $d \in \Delta_A$. When we need to emphasize the fact that a tile is not partial, we will call it *complete*.

Wang pictures. Labeled Wang tiles in Σ_{4K} can be used to build pictures over Σ, by using colors to check compatibility: two tiles may be adjacent only if the color of the touching edges is the same. A picture $P \in {\Sigma_{4K}}^{++}$ is called a *Wang picture* if all borders are colored with # and $P(i, j)_\downarrow = P(i + 1, j)_\uparrow$ for every $1 \leq i < n$, and $P(i, j)_\rightarrow = P(i, j + 1)_\leftarrow$ for every $1 \leq j < m$, where (n, m) is the size of P. The *label of a Wang picture P* over Σ_{4K} is the picture having for pixels the labels of pixels of P. Next (on the left), the reader may find the example of a Wang picture of size $(2, 2)$ with its label (in the middle). For better readability, we represent Wang pictures by writing each common color only once, as in the figure on the right.



Sometimes we need to consider *partial* Wang pictures, whose pixel are partial tiles with compatible edges (some colors may be undefined). Any (partial) Wang picture is called a *(partial) Wang tiling* of its label.

Wang systems. A *Wang system* is a triple $\omega = \langle \Sigma, K, \Theta \rangle$, where Σ is a finite alphabet, K is a set of colors, Θ is a subset of Σ_{4K}. The language generated by ω is the language $L(\omega) \subseteq \Sigma^{++}$ of the labels of all Wang pictures built with tiles in Θ. Notice that a picture $p \in L(\omega)$ may have more than one Wang tiling in ω. REC is the class of picture languages generated by Wang systems.

Example 1. Let $L_{\exists r=1r}$ be the language of all pictures that have a row which equals the first row. L is recognized by the Wang system producing tilings as in Figure 1. In it, symbols from the first row are propagated downwards, and each row is examined from left to right to check its compatibility with the first row: if a wrong symbol is found, color $\times$ is propagated rightwards till the end of the row. If a row is found to be equal to the first one, a primed version of its rightmost symbol is propagated downwards. The picture is recognized only if in the bottom-right corner is colored by a primed symbol (or the last row is checked as compatible and its rightmost symbol matches).

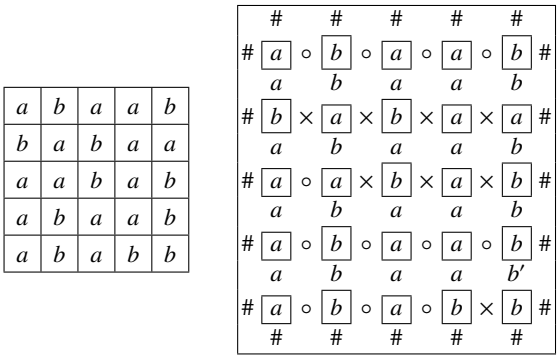


Fig. 1. A picture recognized by the Wang system of Example 1 and the corresponding tiling.

2.2 2D automata models

In the literature, several models of 2D automata have been proposed.

4-way automata. Historically, the first generalization of finite state automata to two dimensions is given by 4-way automata [5]. Soon after this paper, several other similar models have been proposed: a survey can be found in [6]. As the standard model of finite-state automata on strings, a 4-way automaton can be seen as a finite control having a head that visits the positions of a picture and can move in four directions. At each step, it reads the input symbol under the head, then it enters a new state and move to an adjacent position: the direction to move towards and the new state to enter are determined by a transition function, according to the read symbol and the current state. The input picture is accepted if, starting from position $(1, 1)$ in state q_0, the automaton eventually halts in state q_{yes}.

Definition 1. *A* 4-way nondeterministic automaton *(*4NA*) is a tuple* $\langle \Sigma, Q, q_0, q_{yes}, q_{no}, \delta \rangle$ *where:* Σ *is a finite input alphabet;* Q *is a finite set of states, containing in particular the initial state* q_0*, the accepting state* q_{yes}*, and the rejecting state* q_{no}*;* $\delta : \Sigma \times \left(Q \setminus \{q_{yes}, q_{no}\}\right) \to 2^{Q \times Dirs}$ *is the transition function.*

Example 2. In [15] it is proved that the language L of square pictures with the first row of the form $w\bar{w}$, where $\bar{w}$ is the reverse of w, is recognizable by a 4-way automaton. One can see that the same holds if L is generalized to pictures of size (n, m), with $n \geq m \geq 4$. We will call the latter language L_{half}.

Online tessellation acceptors. OTAs are defined in [2] as a restricted type of 2D cellular automaton in which cells do not make transitions at every time-step: rather a "transition wave" sweeps diagonally across the array. Each cell changes its state depending on the two neighbors to the top and to the left. A *run* of a OTA on a picture p of size (n, m) assigns a state (from a finite set) to each position (i, j) of p. Such state depends on the states already associated with positions $(i - 1, j)$ and $(i, j - 1)$ and on symbol $p(i, j)$. At

time $t = 0$ an initial state q_0 is associated with all the positions of the first row and of the first column of $\hat{p}$. The computation starts at time $t = 1$ by reading $p(1,1)$; at time $t = 2$, states are simultaneously assigned to positions $(1,2)$ and $(2,1)$, and so on, to the next diagonals. Picture p is recognized if there exists a run such that the state assigned to position (n,m) is final.

Wang automata directed by polite scanning strategies. Recently [11,12], we introduced μ-directed Wang automata (μ-NWA), a model of automata based on Wang tiles and leaded by a prefixed scanning strategy μ. A Wang automaton can be seen as having a head that visits the input picture, coloring at each step the edges of the position it is visiting. For each accepting computation, the automaton produces a complete Wang picture whose label is equal to the input picture. The coloring operations the automaton performs are determined by a finite control, whereas the movements of the head are lead by the scanning strategy μ; we requires that μ is *polite*, i.e., it has to satisfy some further properties. Fix any starting corner c_s and any starting direction $d_s \in Dirs$, and consider a *next-position function*, i.e., a partial function $\eta : 2^{Dirs} \times Dirs \to Dirs$ such that $\eta(D,d) = \perp$ if $-d \notin D$. The *scanning strategy* $\mu = \langle \eta, c_s, d_s \rangle$ determines how to visit any input picture. More precisely, let d be the current direction, representing the direction from the last considered position, and D represent the set of edges on the picture border together with the edges common with other already visited positions; then $\eta(D,d)$ is the direction towards the position to visit next.

In [13] we proved that any polite scanning strategy has to follow, except for some bootstrap steps, one of four kinds of movements, or their rotations and symmetrical, intuitively exemplified by the following pictures, where the number in each pixel denotes its scanning order.

1	6	7	12
2	5	8	11
3	4	9	10

(a) snake ($\mathcal{S}$)

1	10	11	12
2	9	8	7
3	4	5	6

(b) L-like ($\mathcal{J}$)

1	12	9	8
2	11	10	7
3	4	5	6

(c) U-like ($\mathcal{U}$)

1	10	9	8
2	11	12	7
3	4	5	6

(d) spiral ($\mathcal{C}$)

In the rest of the paper we will sometimes refer to rotations of the snake-like strategy (i.e. $\mathcal{S}$). The one depicted here is called the *left-to-right* version (denoted by $\mathcal{S}^{\mathrm{l2r}}$), while its rotation is called *top-to-bottom* ($\mathcal{S}^{\mathrm{t2b}}$). Their respective 180° rotations are called $\mathcal{S}^{\mathrm{r2l}}$ and $\mathcal{S}^{\mathrm{b2t}}$.

Definition 2. *A* μ-directed nondeterministic Wang automaton *(μ-NWA) is a tuple* $\langle \Sigma, K, \delta, \mu, F \rangle$ *where:* Σ *is a finite input alphabet;* K *is a finite set of colors;* $\delta : \Sigma_K \times Dirs \to 2^{\Sigma_{4K}}$ *is a partial function such that each tile in* $\delta(A,d)$ *extends* A*;* $\mu = \langle \eta, c_s, d_s \rangle$ *is a polite scanning strategy such that* $\delta(A,d) \neq \emptyset$ *implies* $\eta(\Delta_A, d) \neq \perp$*;* $F \subset \Sigma_{4K}$ *is the set of final tiles.*

Intuitively, the behavior of a μ-directed Wang automaton over an input picture p is the following. At the beginning, the head of the automaton points at the position in the starting corner c_s and the current direction is set to d_s. When the current direction is d, the head is at position x, the pixel and the colors of edges of $p(x)$ are summarized by A, then let $d' = \eta(\Delta_A, d)$ and $A' \in \delta(A,d)$. The automaton can execute this move: color

the edges of x according to A', set the current direction to d', and move to the position adjacent to x following direction d'. If no move is possible, the automaton halts. The input picture p is accepted if there exists a computation such that the edges of the final position are colored according to some tile in F.

The choice of the scanning strategy μ is not relevant from the point of view of the recognizing power of μ-directed Wang automata: for every polite scanning strategy μ, the class of picture languages recognized by μ-NWA equals REC [12].

Deterministic models. The deterministic versions DOTA, 4DA, μ-DWA of OTA, 4NA, and μ-NWA, respectively, are defined in the usual way, by making the transition functions deterministic. $\mathcal{L}$(DOTA) is characterized in terms of diagonal-deterministic tiling systems [7]: here we use Diag-DREC to denote its closure under rotation. For each polite scanning strategy μ, a subclass $\mathcal{L}(\mu\text{-DWA})$ of REC is obtained; Scan-DREC is the union of all such deterministic classes [12].

All these models are strictly less powerful than their nondeterministic counterparts, i.e., the corresponding classes of languages are properly included in REC. Their inclusion relations are summarized as follows. First, DOTAs are incomparable with both 4DA and 4NA [2]. Second, in [12], $\mathcal{L}(\mathcal{S}^{\mathrm{t2b}}\text{-DWA})$ is proved to coincide with t2b-UREC, a class introduced in [7] and, more generally, one can see that $\mathcal{L}(\mathcal{S}^{d}\text{-DWA})$ coincides with d-UREC for any direction $d \in \{\mathrm{t2b}, \mathrm{b2t}, \mathrm{l2r}, \mathrm{r2l}\}$. Since Diag-DREC is properly contained in the union of all classes d-UREC [7], we have that Scan-DREC properly extends Diag-DREC and hence also $\mathcal{L}$(DOTA).

The relation between Wang automata and 4-way automata is not clear yet. We do not know any example of language in 4DA that does not belong to Scan-DREC; on the other hand, there exists a language $L \in$ Scan-DREC that cannot be recognized by a 4NA (and a fortiori not by a 4DA), as shown in the following example.

Example 3. Consider again $L_{\exists r=1r}$ as defined in Example 1. Such language cannot be recognized by a 4-way automaton [14]. But it is both in t2b-UREC and in l2r-UREC [7], hence the corresponding $\mathcal{S}^{\mathrm{t2b}}$-DWA and $\mathcal{S}^{\mathrm{l2r}}$-DWA can be built: they basically produce the tiling of Figure 1, except for a delay of one row in case of $\mathcal{S}^{\mathrm{t2b}}$-DWA, and one column in case of $\mathcal{S}^{\mathrm{l2r}}$-DWA.

It is interesting to note that, for every direction d, a necessary condition is known for a language to belong to d-UREC, and hence to $\mathcal{L}(\mathcal{S}^{d}\text{-DWA})$, since those classes coincide [12]. Such condition is based on Matz's technique [16], that suggests to consider a picture as a string over the alphabet of columns (or rows), and Hankel matrices.

The Hankel matrix of a *string* language $S \subseteq \Omega^*$ is an infinite boolean matrix M_S indexed by words $\alpha, \beta \in \Omega^*$. M_S is defined by setting $M_S(\alpha, \beta) = 1$ if and only if $\alpha\beta \in S$. Let L be a picture language and, for every m, let $L(m)$ be the language of pictures in L having m rows. Then $L(m)$ can be seen as a string language over the alphabet $\Omega = \Sigma^m$ of columns of size m. In [7] it is proved that $L \in$ t2b-UREC implies that there exists an integer k such that, for every m, the number of distinct rows of the Hankel matrix $M_{L(m)}$ is lower than k^m. Similar properties can be given for any direction d.

Example 4. The 4NA cited in Example 2 and recognizing $L = L_{\mathrm{half}}$ is deterministic. On the other hand, here we prove that L cannot be recognized by any $\mathcal{S}^{\mathrm{l2r}}$-DWA. For sake of

simplicity, we assume that all rows except the first one are filled with symbol 0. Let us study the Hankel matrix $M_{L(m)}$ for a fixed m. One can verify that $\alpha \neq \alpha'$ implies that the rows of $M_{L(m)}$ indexed by α and α' differ. In other words, the number of distinct rows in $M_{L(m)}$ is not bounded w.r.t. m. Hence, the necessary condition stated above does not hold and this means that $L_{\text{half}} \notin \mathcal{L}(\mathcal{S}^{\text{l2r}}\text{-DWA})$. Similarly, one has $L_{\text{half}} \notin \mathcal{L}(\mathcal{S}^{\text{r2l}}\text{-DWA})$.

3 Adding expressivity

The first one considers the input picture as a read-only tape that can be visited freely, and uses finite states to propagate information [5,6]. The second one fixes a scanning strategy to visit the input picture, but allows the possibility to mark its positions, and use this markers to propagate information locally: each position is marked with a state by DOTAs [2]; it is rewritten with a symbol from a new alphabet in Diag-DREC [7,8]; its edges are colored by Wang automata [12,13].

In this section we try to combine these two apparently orthogonal approaches, in order to improve their expressive power: the idea is to use both states and colors assigned to positions. Clearly we want to stay inside REC, hence this combination must be done carefully.

The first idea is to imagine an automaton with a head that is able to move through the input picture according to its content, and depending on a finite control, changing its state at each step. Such head should move from a position to an adjacent one, and color at each step some edges of the position it is visiting.

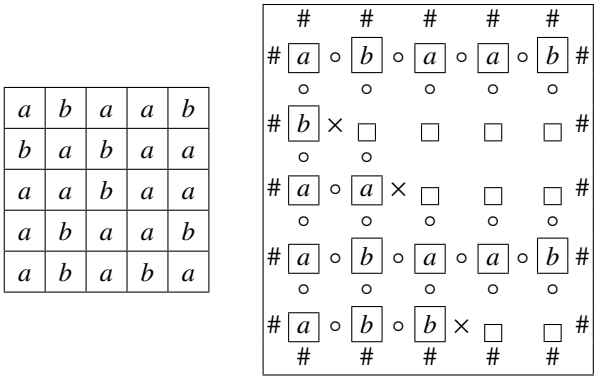
In all models presented in the literature, the coloring operation in a given position are done once and for ever; similarly we do not admit the possibility to change a color previously assigned to a position. This leads to the following tentative definition. A 2D *free* deterministic automaton is defined by a tuple $\langle \Sigma, K, Q, q_0, q_{yes}, q_{no}, \delta \rangle$ where: Σ is a finite input alphabet; K is a finite set of colors; Q is a finite set of states, containing in particular the initial state q_0, the accepting state q_{yes}, and the rejecting state q_{no}; finally $\delta : \Sigma_K \times \left(Q \setminus \{q_{yes}, q_{no}\}\right) \to \Sigma_K \times Q \times \mathit{Dirs}$ is a partial function such that $\delta(A, q) \ni (A', q', d)$ implies that A' extends A.

The behavior of a 2D free deterministic automaton $\langle \Sigma, K, Q, q_0, q_{yes}, q_{no}, \delta \rangle$ over an input picture $p \in \Sigma^{++}$ would be described informally as follows. At the beginning, the head of the automaton points at the top-left corner and the current state is set to q_0. When the current state is q, the head is placed at position x, the pixel and the colors of edges of p at position x are summarized by $P(x)$, then let $(A', q', d) \in \delta(P(x), q)$. Hence the automaton may execute this move: if A is partial, color edges at position x according to A', then enter state q', move to the position adjacent to x towards direction d, and extend P to the Wang picture P' with $P'(x) = A'$.

Notice that the head can visit any cell any number of times, but colors cannot be changed (A' is a Wang tile that extends A). If no move is possible, the automaton halts. The input picture p is accepted if there is a computation such that the automaton eventually enters state q_{yes}.

Example 5. Language $L_{\exists r=1^r}$ can be recognized by an automaton that visits the input picture following a sort of "comb-like" movement. Next you find an accepted input

picture and the partial Wang picture obtained by the computation; the symbols never read by the automaton are omitted.



The automaton uses only two colors: the *reject* color × when it finds out that a row is different from the first one, and another symbol ∘ to mark the edges of the position it is visiting for the first time. The set of states is $Q = \{q_0, q_{yes}, q_{no}\} \cup \Sigma \cup \bar{\Sigma}$, where symbols in $\bar{\Sigma}$ are a barred version of symbols in Σ, and they are used to distinguish the part of the computation when the head moves leftwards. The transition function is summarized in Figure 2.

Considering an input picture of size (n, m), the automaton repeats the following sequence of moves m times. For every $j < m$, it visits the top row from left to right until it reaches the first unvisited position, i.e. $(1, j)$. The symbol found there is saved in the state. Then, the automaton goes back to $(1, 1)$ and moves downward, to find all rows that are compatible with the first j symbols of the first row.

This task is performed as follows. Each row, starting from the second, is scanned from left to right until one of the following two cases occurs. (1) A reject color is found, so the row was already marked as unsuitable, and the automaton has to go back to the first column and then move to the next row. (2) There is an unvisited position that, by construction, is position (i, j), so either it contains the symbol saved in the state and it has to be marked just as visited, or it contains a wrong symbol and it is hence marked with the reject color. As in the previous case, the automaton has to go back to the first column and then move to the next row. Once the bottom row has been examined, the automaton enters state q_0 and moves leftward to the first column, then goes back up to the first row and then repeats the cycle by considering position $(1, j + 1)$.

When the last position in the first row is scanned, to accept the input picture the automaton has only to check if there is at least one non-rejected row ending with the right symbol.

The previous example shows a critical feature of the model: whenever the symbol at position $(1, j)$ is considered, the automaton enters a sort of loop (it goes across each row i until it reaches position (i, j) or finds the reject color, then it comes back to the first column), whose outcome is different according to a piece of information which is not locally propagated (i.e., whether the row has already been rejected or not). Clearly this sort of cyclic computation cannot be removed and this prevents to apply a construction

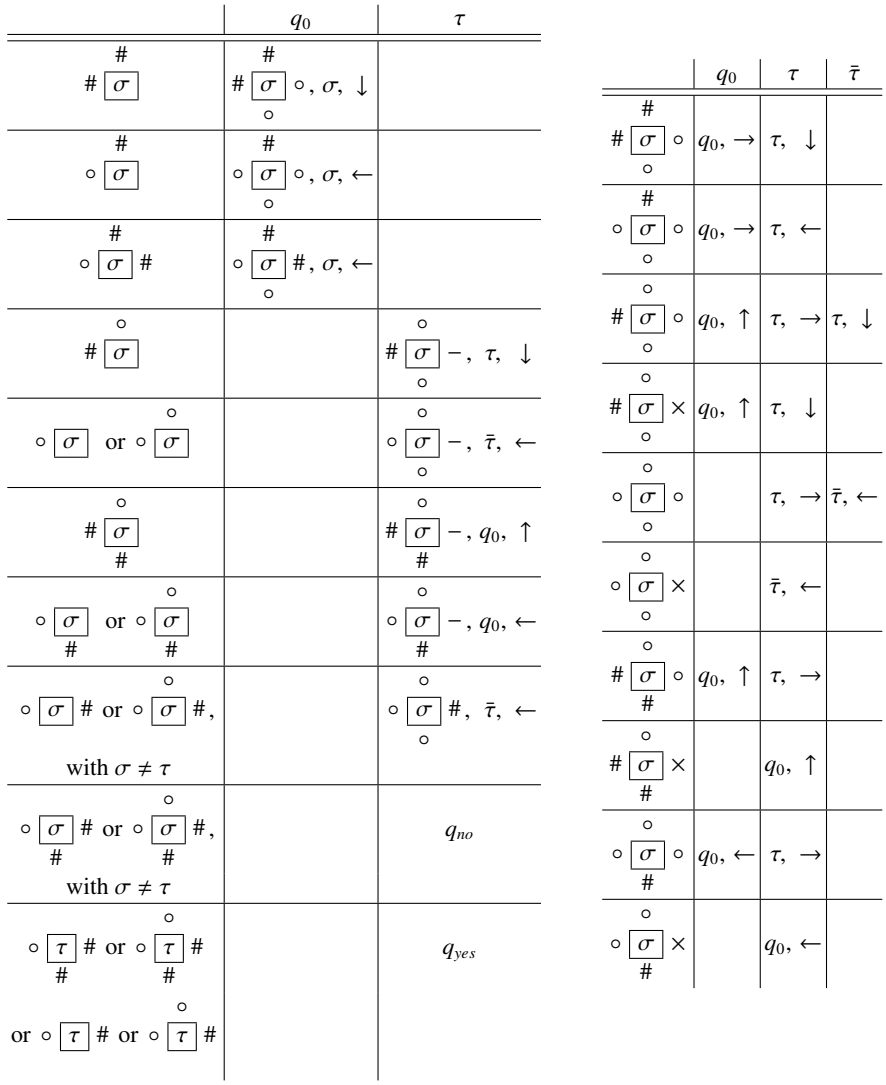


Fig. 2. The transition function of the automaton for language $L_{\exists r=1r}$: coloring steps (left), revisiting steps (right). Rows are indexed by partial Wang tiles, columns are indexed by states. For revisiting steps, the complete Wang tile in the codomain is omitted since equals the tile in the domain. Notation $-$ stays for $\times$ if $\tau \neq \sigma$, for $\circ$ otherwise.

similar to the one used in [2] to prove that $\mathcal{L}$(4NA) is included in REC. Actually, it turns out that this model is really too permissive, as next example illustrates.

Example 6. Consider the language $L_{a^n b^n}$ of pictures with one row of the form $a^n b^n$. Since $L_{a^n b^n}$, seen as a string language, is not regular, clearly $L \notin$ REC. However $L_{a^n b^n}$ is recognizable by the following free automaton: the set of states is $Q = \{q_0, q_1, q_{yes}, q_{no}\}$ and the set of colors is $K = \{ok\}$ it starts from the top-left border, if the current symbol is

a, then it marks its right edge by color *yes*, enters state q_1, and move rightwards without changing states nor coloring, until it reaches a position having the right edge already marked (or bordered); if the current symbol is b, then it marks its left edge by color *ok*, enters state q_0, and move leftwards without changing states nor coloring, until it reaches a position having the left edge already marked (or bordered). Such sequence of moves is repeated until all position are marked, and in this case the input picture is accepted. Whenever one of the previous conditions is not satisfied, the input picture is rejected.

Hence, the definition of free automaton needs to be somehow constrained. For instance one should require that the first time a position is visited, a coloring step must be performed. This is obtained simply by replacing the codomain of δ by $\Sigma_{4K} \times Q \times Dirs$. The new condition would prevent the behavior of the automaton in the previous example. However this is not sufficient yet, as next example illustrates.

Example 7. Consider the language $L_{2a^nb^n}$ of pictures with two rows, the first one having the form a^nb^n. Again, it is easy to see that $L_{2a^nb^n} \notin$ REC (otherwise it would be easy to build a Wang system for $L_{a^nb^n}$, too). However $L_{2a^nb^n}$ is recognizable by a free automaton that further respects the above condition. The behavior of such automaton is similar to the one described in Example 6, except that the first row is used only to mark the position under consideration, and the second row allows to go back and forth from left to right.

These problems suggest that the combination of coloring and revisiting steps should be simplified: first the automaton executes a sequence of coloring steps, then it performs a sequence of revisiting steps, using the information enclosed in the colors placed before. We will call the two phases *tiling* and *roaming*, respectively. In other words, the automaton simulates first the behavior of a μ-DWA (in the tiling phase), for some prefixed scanning strategy μ, and then the behavior of a 4DA (in the roaming phase).

This leads to this new definition:

Definition 3. *A* 4-way deterministic μ-directed Wang automaton *(μ-*4DWA*) is a tuple* $\langle \Sigma, K, \gamma, \mu, Q, q_0, q_{yes}, q_{no}, \delta \rangle$ *where:*

- Σ *is a finite input alphabet;*
- K *is a finite set of colors;*
- $\gamma : \Sigma_K \times Dirs \to \Sigma_{4K}$*, the* tiling *transition function, is a partial function such that each tile in* $\gamma(A, d)$ *extends* A*;*
- $\mu = \langle \eta, c_s, d_s \rangle$ *is a polite scanning strategy such that* $\gamma(A, d) \neq \emptyset$ *implies* $\eta(\Delta_A, d) \neq \perp$*;*
- Q *is a finite set of states, containing in particular the initial roaming state* q_0*, the accepting state* q_{yes}*, and the rejecting state* q_{no}*;*
- $\delta : \Sigma_{4K} \times Q \to Q \times Dirs$ *is the* roaming *transition function.*

The formal semantics is not presented but it is a straightforward combination of μ-DWA and 4DA: when the μ-DWA component ends its picture scanning, the 4DA component starts working from the current position.

Example 8. For instance, a $\mathcal{S}^{\mathrm{t2b}}$-4DWA for the language $L_{\exists r=1^r} \cap L_{\mathrm{half}}$ can be defined as follows: first visit the input picture row by row from top to bottom, simulating the $\mathcal{S}^{\mathrm{t2b}}$-DWA defined in Example 3, to check if the input picture is in $L_{\exists r=1^r}$; then simulate the 4DA cited in Example 2, to check if the input picture is in L_{half}.

4 Properties of 4-way deterministic Wang automata

Theorem 1. *For every polite μ, $\mathcal{L}$(4DA) $\cup$ $\mathcal{L}$(μ-DWA) $\subseteq$ $\mathcal{L}$(μ-4DWA) $\subseteq$ REC.*

Proof. It is easy to verify that both 4DA and μ-DWA (for μ polite) are special kinds of μ-4DWA. On the one hand, 4DA are obtained by reducing the set of colors used in the tiling part to the empty set, i.e., when Σ_K is simply Σ. On the other hand, μ-DWA are obtained by omitting the roaming part of the transition function.

Let L be accepted by some μ-4DWA $\mathcal{A}$. Clearly, $\mathcal{A}$ can be seen as the combination of a μ-4DWA $\mathcal{A}_1$ over alphabet Σ and a 4DA $\mathcal{A}_2$ over alphabet Σ_K. Now, let $L_1, L_2 \subseteq {\Sigma_K}^{++}$ be defined as follows. L_1 the set of (partial) Wang tilings produced by all accepting computation of $\mathcal{A}_1$; L_2 is the language accepted by $\mathcal{A}_2$. Then, by definition $L = \lambda(L_1 \cap L_2)$, where $\lambda : \Sigma_K \to \Sigma$ is the projection that maps each tile onto its label. Since REC is closed under intersection and alphabetic projection, we have that $L \in$ REC. □

Theorem 2. *For every polite μ, $\mathcal{L}$(μ-4DWA) is a boolean algebra.*

Proof. By definition, since both $\mathcal{L}$(4DA) and $\mathcal{L}$(μ-DWA) are boolean algebras [12]. □

Theorem 3. *There exists a language in $\mathcal{L}$($\mathcal{S}$-4DWA) which is not in $\mathcal{L}$(4DA) nor in* Scan-DREC.

Proof. Consider the language $L_{\text{half}} \cap L_{\exists r=2r}$ of pictures having the first row of the form $w\bar{w}$, and some row that equals the second row. Let L be the intersection of such language with its horizontal mirror.

Language L can be recognized by a $\mathcal{S}^{\text{l2r}}$-4DWA that simulates first a variant of the $\mathcal{S}^{\text{l2r}}$-DWA of Example 3 (such variant examines at the same time both the second and the second-last rows instead of only the first one), then the 4DA of Example 2, and then again a rotated version of the same 4DA.

On the other hand, it is known [14] that 4NA cannot recognize $L_{\exists r=2r}$, hence a fortiori L cannot be recognized by any 4DA. Now we show that L is not in Scan-DREC. Since a $\mathcal{S}^{\text{t2b}}$-DWA cannot recognize ${L_{\exists r=2r}}^{h}$, as proved in [12], then L cannot be recognized by neither a $\mathcal{S}^{\text{t2b}}$-DWA nor a $\mathcal{S}^{\text{b2t}}$-DWA. Moreover, since no $\mathcal{C}$-DWA can recognize $L_{\exists r=2r}$ (see [12]), then clearly L cannot be recognized by any $\mathcal{C}$-DWA. By a similar reasoning one can prove that L is neither in $\mathcal{L}$($\mathcal{U}$-DWA), nor in $\mathcal{L}$($\mathcal{J}$-DWA), or in any of their rotations. Finally, reasoning as in Example 4 we can verify that L cannot be recognized by neither a $\mathcal{S}^{\text{r2l}}$-DWA nor a $\mathcal{S}^{\text{l2r}}$-DWA.

Hence, the result follows from the fact that all polite scanning strategies are basically limited to $\mathcal{C}, \mathcal{U}, \mathcal{J}, \mathcal{S}$ and their rotations and mirrors [13]. □

Notice that the language used in the previous proof to separate classes Scan-DREC and $\mathcal{L}$(4DA) from $\mathcal{L}$($\mathcal{S}$-4DWA) actually separates also $\mathcal{L}$(4NA) from $\mathcal{L}$($\mathcal{S}$-4DWA). It is not clear if there exists any language recognizable nondeterministically by a 4NA but not by μ-4DWA.

Concludingly, these last results clearly show that the proposed approach of combining the free roaming of 4-way automata and coloring based on predefined scanning strategies is effective. Indeed, this yields to a concept of determinism which extends those presented in [5,2,7,8,10,11,12,13].

References

1. Giammarresi, D., Restivo, A.: Two-dimensional languages. In Salomaa, A., Rozenberg, G., eds.: Handbook of Formal Languages. Volume 3, Beyond Words. Springer-Verlag, Berlin (1997) 215–267
2. Inoue, K., Nakamura, A.: Some properties of two-dimensional on-line tessellation acceptors. Information Sciences **13** (1977) 95–121
3. Giammarresi, D., Restivo, A.: Recognizable picture languages. International Journal Pattern Recognition and Artificial Intelligence **6** (1992) 241–256 Special Issue on *Parallel Image Processing*.
4. de Prophetis, L., Varricchio, S.: Recognizability of rectangular pictures by Wang systems. Journal of Automata, Languages and Combinatorics **2** (1997) 269–288
5. Blum, M., Hewitt, C.: Automata on a 2-dimensional tape. In: Conference Record of 1967 Eighth Annual Symposium on Switching and Automata Theory, IEEE (1967) 155–160
6. Inoue, K., Takanami, I.: A survey of two-dimensional automata theory. Information Sciences **55** (1991) 99–121
7. Anselmo, M., Giammarresi, D., Madonia, M.: From determinism to non-determinism in recognizable two-dimensional languages. In: Proc. DLT 2007. Volume 4588 of Lecture Notes in Computer Science., Springer (2007) 36–47
8. Anselmo, M., Giammarresi, D., Madonia, M.: Deterministic and unambiguous families within recognizable two-dimensional languages. Fundamenta Informaticae **98** (2010) 143–166
9. Anselmo, M., Giammarresi, D., Madonia, M.: Tiling automaton: A computational model for recognizable two-dimensional languages. In: Proc. CIAA 2007. Volume 4783 of Lecture Notes in Computer Science., Springer (2007) 290–302
10. Lonati, V., Pradella, M.: Snake-deterministic tiling systems. In: Proc. MFCS 2009. Volume 5734 of Lecture Notes in Computer Science., Springer (2009) 549–560
11. Lonati, V., Pradella, M.: Picture recognizability with automata based on Wang tiles. In: Proc. SOFSEM 2010. Volume 5901 of Lecture Notes in Computer Science., Springer (2010) 576–587
12. Lonati, V., Pradella, M.: Deterministic recognizability of picture languages with Wang automata. Discrete Mathematics and Theoretical Computer Science **4** (2010) 73–94
13. Lonati, V., Pradella, M.: Strategies to scan pictures with automata based on Wang tiles. R.A.I.R.O. Theoretical Informatics and Applications **44** (2010)
14. Inoue, K., Takanami, I.: A characterization of recognizable picture languages. In Nakamura, A., Nivat, M., Saoudi, A., Wang, P.S.P., Inoue, K., eds.: Parallel Image Analysis, Second International Conference, ICPIA '92, Ube, Japan, December 21-23, 1992, Proceedings. Volume 654 of Lecture Notes in Computer Science., Springer (1992) 133–143
15. Ito, A., Inoue, K., Takanami, I.: A note on three-way two-dimensional alternating Turing machines. Inf. Sci. **45** (1988) 1–22
16. Matz, O.: On piecewise testable, starfree, and recognizable picture languages. In Nivat, M., ed.: Proc. FoSSaCS 1998. Volume 1378 of Lecture Notes in Computer Science., Springer (1998) 203–210