On the Complexity of Unary Tiling-Recognizable Picture Languages*

Alberto Bertoni[1], Massimiliano Goldwurm[1], and Violetta Lonati[1]

Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano
Via Comelico 39/41, 20135 Milano – Italy
{bertoni,goldwurm,lonati}@dsi.unimi.it

Abstract. We give a characterization, in terms of computational complexity, of the family REC_1 of the unary picture languages that are tiling recognizable. We introduce quasi-unary strings to represent unary pictures and we prove that any unary picture language L is in REC_1 if and only if the set of all quasi-unary strings encoding the elements of L is recognizable by a one-tape nondeterministic Turing machine that is space and head-reversal linearly bounded. In particular, the result implies that the family of binary string languages corresponding to tiling-recognizable square languages lies between $\text{NTIME}(2^n)$ and $\text{NTIME}(4^n)$. This also implies the existence of a nontiling-recognizable unary square language that corresponds to a binary string language recognizable in nondeterministic time $O(4^n \log n)$.

Classification: automata and formal languages, computational complexity.

Keywords: unary picture languages, tiling systems, Turing machine head reversal.

1 Introduction

Picture languages have been introduced in the literature as two-dimensional extension of traditional string languages, a picture being a two-dimensional array of elements from a finite alphabet. They have been originally considered as formal models for image processing in connection with problems of pattern recognition. Several classical tools and concepts have been used to classify picture languages and study their properties: regular expressions [8], grammars [12], automata [6], logic formulas [5].

One of the main effort in this area is to capture the notion of recognizability. In particular, various notions of two-dimensional finite automaton have been proposed and studied in the literature [6,7]. An interesting formal model for the recognition of picture languages is given by the so-called tiling systems introduced in [3], which are based on projection of local properties. A tiling system τ is defined by a finite set Θ of square pictures of size 2 together with a projection

* This work has been supported by the Project M.I.U.R. COFIN "Automata and formal languages: mathematical and application driven studies".

W. Thomas and P. Weil (Eds.): STACS 2007, LNCS 4393, pp. 381–392, 2007.
© Springer-Verlag Berlin Heidelberg 2007

between alphabets. Roughly speaking, a language is recognized by τ if each of its elements can be obtained as a projection of a picture whose subpictures of size 2 belong to Θ. The class of picture languages recognized by such systems satisfy relevant properties, which resemble classical properties of regular string languages [4].

A special case is represented by pictures over a one-letter alphabet: in this case only the shape of the picture is relevant, and hence a unary picture is simply identified by a pair of positive integers. In this context, a general goal is to define techniques to describe families of recognizable languages, or to construct examples of non-recognizable languages [4,7]. For instance, families of tiling-recognizable unary picture languages are introduced in [4] by means of integer functions or in [2] by means of special regular expressions, whereas in [7] two-dimensional automata are used to recognize unary languages and several strategies to explore pictures are presented.

In this work we give a complexity result concerning the unary picture languages recognized by tiling systems. We characterize such a family by means of non-deterministic Turing machines that are space and head-reversal bounded. More precisely, we introduce a notion of quasi-unary strings to represent pairs of positive numbers and we prove that a unary picture language L is tiling recognizable if and only if the set of all quasi-unary strings encoding the sizes of the elements of L is recognizable by a one-tape non-deterministic Turing machine M that works within $\max(n, m)$ space and executes at most $\min(n, m)$ head reversals, on the input representing the pair (n, m).

In particular for the case of squares, this result allows us to relate the recognizability of unary square pictures to nondeterministic time complexity bounds. Informally, it shows that the complexity of the binary encodings of tiling-recognizable unary square picture languages is located between $\text{NTIME}(2^n)$ and $\text{NTIME}(4^n)$. This yields a large variety of examples of picture languages that are tiling recognizable. For instance, all sets of binary encodings of NP problems correspond to tiling-recognizable (unary square) picture languages.

Also, our characterization allows us to use separating results on time complexity classes as a tool for defining recognizable and non-recognizable unary picture languages. In particular, using a property proved in [11], we show the existence of a unary square language that is not tiling recognizable, but corresponds to a binary string language recognizable in nondeterministic time $O(4^n \log n)$.

2 Preliminaries on Picture Languages

Given a finite alphabet Σ, a *picture* (or *two-dimensional string*) over Σ is either a two-dimensional array (i.e., a matrix) of elements of Σ or the *empty picture* λ. The set of all pictures over Σ is denoted by Σ^{**}; a *picture language* (or *two-dimensional language*)over Σ is a subset of Σ^{**}.

Given a picture $p \in \Sigma^{**}$, we use r_p and c_p to denote the number of rows and columns of p, respectively. The pair $(\mathrm{r}_p, \mathrm{c}_p)$ is called the *size* of p. By definition we have $\mathrm{r}_p > 0$ and $\mathrm{c}_p > 0$, except for the empty picture λ that has size $(0, 0)$. The

symbol in p with coordinates (i, j) is denoted by $p(i, j)$, for every $1 \leq i \leq \mathrm{r}_p$ and $1 \leq j \leq \mathrm{c}_p$. If $\mathrm{r}_p = \mathrm{c}_p$, then p is called a *square picture* and the *size* of p is simply r_p. A *square language* is a picture language containing only square pictures. If the alphabet Σ is a singleton, then the pictures over Σ^{**} are called *unary pictures*. A *unary picture language* is a subset of Σ^{**}, where Σ is a singleton.

For any picture $p \in \Sigma^{**}$ of size (m, n), we use $\hat{p}$ to denote a new picture of size $(m + 2, n + 2)$ obtained by surrounding p with a special boundary symbol $\sharp \notin \Sigma$. Such boundary will be useful when describing scanning strategies for pictures.

Many operations can be defined between pictures and picture languages. In particular, we recall the operations of row and column concatenation. Let p and q be pictures over Σ^{**} of size $(\mathrm{r}_p, \mathrm{c}_p)$ and $(\mathrm{r}_q, \mathrm{c}_q)$, respectively. If $\mathrm{r}_p = \mathrm{r}_q$, we define the column concatenation $p \oplus q$ between p and q as the picture of size $(\mathrm{r}_p, \mathrm{c}_p + \mathrm{c}_q)$ whose i-th row equals the concatenation of the i-th rows of p and q, for every $1 \leq i \leq \mathrm{r}_p$. If $\mathrm{c}_p = \mathrm{c}_q$, we define the row concatenation $p \ominus q$ analogously. Clearly, $\oplus$ and $\ominus$ are partial operations over the set Σ^{**}. These definitions can be extended to picture languages and iterated: for every language $L \subseteq \Sigma^{**}$, we set $L^{0\oplus} = L^{0\ominus} = \{\lambda\}$, $L^{i\oplus} = L \oplus L^{(i-1)\oplus}$ and $L^{i\ominus} = L \oplus L^{(i-1)\ominus}$, for every $i \geq 1$. Thus, one can define the *row* and *column closures* as the transitive closures of $\oplus$ and $\ominus$:

$$L^{*\oplus} = \bigcup_{i \geq 0} L^{i\oplus} \qquad L^{*\ominus} = \bigcup_{i \geq 0} L^{i\ominus},$$

which can be seen as a sort of *two-dimensional Kleene star*. Another useful operation is the so-called *rotation*: given $p \in \Sigma^{**}$, its rotation p^R is the picture of size $(\mathrm{c}_p, \mathrm{r}_p)$ defined by $(p^R)_{ij} = p_{\mathrm{r}_p + 1 - j, i}$.

From the recognizability view point, various approaches have been proposed. In particular, here we consider the class REC and its definition in terms of tiling systems [3,4]. First, we recall the definition of *local picture language*.

Definition 1. *A* tile *is a square picture of size 2; for every picture p, $T(p)$ denotes the set of all tiles that are subpictures of p. A picture language $L \subseteq \Gamma^{**}$ is called* local *if there exists a finite set Θ of tiles over the alphabet $\Gamma \cup \{\sharp\}$ such that $L = \{p \in \Gamma^{**} \mid T(\hat{p}) \subseteq \Theta\}$. In this case we write $L = \mathcal{L}(\Theta)$.*

We also need the notion of projection of pictures and picture languages. Let $\pi : \Gamma \to \Sigma$ be a mapping between two alphabets. Given a picture $p \in \Gamma^{**}$, the *projection* of p by π is the picture $\pi(p) \in \Sigma^{**}$ such that $\pi(p)\ (i, j) = \pi(p(i, j))$ for every pair of coordinates i, j. Analogously, the projection of a language $L \subseteq \Gamma^{**}$ by π is the set $\pi(L) = \{\pi(p) \mid p \in \Gamma^{**}\} \subseteq \Sigma^{**}$.

Definition 2. *A* tiling system *is a 4-tuple $\tau = \langle \Sigma, \Gamma, \Theta, \pi \rangle$ where Σ and Γ are two finite alphabets, Θ is a finite set of tiles over the alphabet $\Gamma \cup \{\sharp\}$ and $\pi : \Gamma \to \Sigma$ is a projection. A picture language is* tiling recognizable *if there exists a tiling system $\langle \Sigma, \Gamma, \Theta, \pi \rangle$ such that $L = \pi(\mathcal{L}(\Theta))$.* REC *is the class of picture languages that are tiling recognizable.*

Notice in particular that any local language is tiling recognizable.

The class REC satisfies some remarkable properties. For instance it can be defined as the class of languages recognized by online tessellation automata, that are special acceptors related to cellular automata [6]; they can be expressed by formulas of existential monadic second order [5]; they can be defined by means of regular-like expressions based on certain composition rules between pictures [4]. In particular we will use the fact that REC is closed with respect to the operations $\cup, \ominus, \oplus, ^{*\ominus}, ^{*\oplus}, ^{R}$.

Finally, since we are interested in unary pictures, we also introduce the following

Definition 3. REC_1 *is the subclass of* REC *containing the unary picture languages that are tiling recognizable.*

3 Characterization of $\mathbf{Rec_1}$

In this section, we state our main result, that is a characterization of the class of unary picture languages that are tiling recognizable.

To this aim, consider the alphabet $\Sigma = \{\circ\}$ and notice that any unary picture $p \in \{\circ\}^{**}$ is identified by its size, that is by the pair $(\mathrm{r}_p, \mathrm{c}_p)$. Thus, unary pictures (i.e. pairs of positive integers) can be encoded by quasi-unary strings as follows. We consider the set of unary strings over Σ

$$U = \{\circ^n \mid n > 0\}$$

and the following sets of strings that are unary except for one special letter h or v (not occurring in first position):

$$Q_h = \{\circ^n h \circ^k \mid n > 0, k \geq 0\} ,$$
$$Q_v = \{\circ^n v \circ^k \mid n > 0, k \geq 0\} .$$

We call *quasi-unary string* over the alphabet $\{\circ, h, v\}$ any string in $Q = U \cup Q_h \cup Q_v$. The length of any quasi-unary string x is denoted as usual by $|x|$, whereas we use ${}_\circ|x|$ to denote the length of the longest prefix of x in $\circ^+$. The use of symbols h and v allows us to distinguish among squares, horizontal (with more columns than rows), and vertical rectangles. Thus, a quasi-unary string $x \in Q_h$ represents the unary horizontal rectangle of size $({}_\circ|x|, |x|)$; $x \in Q_v$ represents the unary vertical rectangle of size $(|x|, {}_\circ|x|)$; whereas $x \in U$ represents the unary square of size $|x|$.

Summarizing the previous definitions, the encoding ϕ from unary pictures to quasi-unary strings can be stated as follows: for every picture $p \in \{\circ\}^{**}$, we have

$$\phi(p) = \begin{cases} \circ^{\mathrm{r}_p}\, h\, \circ^{\mathrm{c}_p - \mathrm{r}_p - 1} & \text{if } \mathrm{r}_p < \mathrm{c}_p \\ \circ^{\mathrm{r}_p} & \text{if } \mathrm{r}_p = \mathrm{c}_p \\ \circ^{\mathrm{c}_p}\, v\, \circ^{\mathrm{r}_p - \mathrm{c}_p - 1} & \text{if } \mathrm{r}_p > \mathrm{c}_p \end{cases}$$

Notice that $|\phi(p)| = \max(\mathrm{r}_p, \mathrm{c}_p)$, while ${}_\circ|\phi(p)| = \min(\mathrm{r}_p, \mathrm{c}_p)$.

Now, let us introduce the complexity classes of quasi-unary languages that we shall use to characterize the class of tiling-recognizable unary languages.

Definition 4. NSPACEREV_Q *is the class of quasi-unary string languages that can be recognized by 1-tape nondeterministic Turing machines working within* $|x|$ *space and executing at most* ${}_\circ|x|$ *head reversals, for any input* x *in* Q.

Our main theorem can then be stated as follows:

Theorem 1. *A unary picture language* L *is in* REC_1 *if and only if* $\phi(L)$ *belongs to* NSPACEREV_Q.

The proof of Theorem 1 is split into two parts. In section 4 we prove that if L is in REC_1, then $\phi(L)$ belongs to NSPACEREV_Q, whereas in Section 5 we prove the inverse.

4 Recognizability Implies the Complexity Bound

In this section we prove that, if L is a tiling-recognizable unary picture language, then $\phi(L)$ is in NSPACEREV_Q. In order to prove such a result, let Θ be a finite set of tiles over some alphabet Γ, and consider the following problem.

SIZE REPRESENTABILITY (Θ)
Instance: a quasi-unary string $x \in Q$.
Question: does there exist $p \in \mathcal{L}(\Theta)$ whose size is represented by x?

Lemma 1. *The problem* SIZE REPRESENTABILITY (Θ) *is in* NSPACEREV_Q *for every finite set of tiles* Θ.

Proof. We define a Turing machine M for the SIZE REPRESENTABILITY problem, that nondeterministically tries to generate some $p \in \mathcal{L}(\Theta)$ of the required size. First of all, M establishes if $x \in Q_h$, $x \in Q_v$, or $x \in U$. This can be done nondeterministically without head reversals. If $x \in Q_h$ or $x \in U$, then the generation is performed row by row, otherwise the generation has to be done column by column. The input is accepted if and only if such a generating process can be accomplished. We describe in details only the steps executed in the case $x \in Q_h$; the other cases are similar and are left to the reader.

The working alphabet Γ' of M contains the symbols $\circ, h, v, \sharp, \not\sharp$, all the pairs $(a, b) \in (\Gamma \cup \{\sharp\}) \times (\Gamma \cup \{\sharp\})$, and their marked versions $(\overline{a, b})$ and $(\widetilde{a, b})$. The symbols (a, b) shall be used in correspondence with a pair of adjacent symbols in some column of the picture p generated during the computation; the overlined symbols shall be used as bookmarks at the ${}_\circ|x|$-th cell, tildes shall be used to implement a counter.

The machine M works only on the portion of the tape containing the input x, which we call the *working portion* of the tape. The computation behaves as follows:

1. First of all, M reads the tape rightwards until the first blank, nondeterministically replacing each input symbol according to Θ, whenever such a replacement is possible. More precisely:

- the leftmost symbol is replaced by some pair $(\sharp, a)$ such that the tile t_1 in the figure below belongs to Θ;
- any next symbol is replaced by some pair $(\sharp, b)$ in such a way that, for each pair of consecutive pairs $(\sharp, b)$ and $(\sharp, b')$, the tile t_2 in the figure belongs to Θ (the position of the symbol h is preserved by using overlined pairs);
- the rightmost symbol $\circ$ is replaced by some pair $(\sharp, c)$ such that the tile t_3 in the figure belongs to Θ. At any position, if no replacement is allowed, then M halts and rejects.

$t_1 =$  $t_2 =$  $t_3 =$ 

2. M changes direction and reads all the working portion of the tape without head reversals, replacing each symbol (a, b) by (b, c) in such a way that the ending symbols and each pair of consecutive symbols do respect Θ (as in point 1). Such a procedure is repeated $({}_\circ|x| - 1)$-many times. Observe that this task can be performed by using the first $({}_\circ|x|)$ cells of the tape (those that precede some overlined symbol of Γ') and marking one cell with a tilde at each repetition. Also during this phase, if no replacement is allowed, then M halts and rejects.
3. After the $({}_\circ|x| - 1)$-th repetition of step 2, M changes direction and reads all the working portion of the tape again, without head reversals. Now each symbol (a, b) is replaced by $(b, \sharp)$ according to Θ, and whenever no replacement is allowed, then M halts and rejects.

The input x is accepted if and only if the procedure can be concluded, that is, if and only if there exists a picture $p \in \mathcal{L}(\Theta)$ of size $({}_\circ|x|, |x|)$. Since the machine M works exactly in space $|x|$ and executes exactly ${}_\circ|x|$ head reversals, the proof is complete.

Theorem 2. *If L is a unary picture language in* REC_1*, then $\phi(L)$ belongs to* NSPACEREV_Q.

Proof. Let $\langle \{\circ\}, \Gamma, \Theta, \pi \rangle$ be a tiling system for L, and consider the Turing machine M that solves the problem SIZE REPRESENTABILITY(Θ). Now notice that π maps all symbols of Γ to $\circ$, that is π forgets the content of p and preserves only its size. Thus $x \in \phi(L) = \phi(\pi(\mathcal{L}(\Theta)))$ means that there is a picture in $\mathcal{L}(\Theta)$ whose size is represented by x. Therefore M exactly recognizes the set $\phi(L)$ and this concludes the proof.

5 The Complexity Bound Implies Recognizability

To prove the inverse of Theorem 2, we first introduce an auxiliary picture language, associated with the accepting computations of a 1-tape nondeterministic

Turing Machine. A similar approach is used in [3] to prove that the emptiness problem for the family REC is undecidable.

5.1 The Accepting-Computation Language of a Turing Machine

Let M be a 1-tape nondeterministic Turing machine M, and let Σ and Λ be the input and the working alphabet (Λ contains the blank symbol $\not b$). We denote by Q the set of states, which includes the initial state q_0 and a unique accepting state q_{yes}. Also let $\delta : Q \times \Lambda \to 2^{Q\times\Lambda\times\{+,-\}}$ be the transition function of M. Without loss of generality, we assume M can never print the blank symbol $\not b$, and hence $(q, c, x) \in \delta(p, a)$ implies $c \neq \not b$. Then, set $\Lambda_Q = \{\sigma_q \mid \sigma \in \Lambda, q \in Q\}$, a configuration of M is a string $C = x\sigma_q y \in \Lambda^*\Lambda_Q\Lambda^*$ which represents the instantaneous description of the machine where $x\sigma y$ is the work portion of the tape, q is the current state and the head scans the cell containing σ on the right of x. If $q = q_0$ and x is the empty string, then C is the initial configuration of M on input σy. If $q = q_{\text{yes}}$ then C is an accepting configuration. We assume the machine halts in every accepting configuration.

Given two configurations C and D of M, we write $C \rhd D$ whenever M can go from C to D without head reversals, possibly by several distinct moves. We call *run* such a sequence of moves.

We define an accepting computation[1] of M on input $x \in \Sigma^*$ as a string of the form

$$W = W_1 \rhd W_2 \rhd \cdots \rhd W_n$$

such that all W_j's are configurations of M, W_1 is the initial configuration on input x, W_n is an accepting configuration, $W_i \rhd W_{i+1}$ holds for each $i = 1, \ldots, n-1$, and there is a head reversal at W_i for every $1 < i < n$, that is, in the runs from W_{i-1} to W_i and from W_i to W_{i+1}, the head moves to opposite directions.

Given an accepting computation W, let $m = \max_i |W_i|$ and consider the picture of size $n \times m$ containing the string W_i (possibly followed by $\not b$'s) on the i-th row, for $1 \leq i \leq n$. Notice that, from such a picture, one can recover the input and the sequence of runs but not the complete step-by-step computation on the same input.

The *accepting-computation language* of M is defined as the set $A(M)$ of all pictures corresponding to any accepting computation of M. Note that every accepting computation W of M corresponds to a picture $w \in A(M)$ such that $\mathrm{r}_w - 2$ equals the number of head reversals executed in W (corresponding to $W_2, \cdots, W_{n-1}$) and c_w is the space used in W.

Example 1. Let M be a Turing machine such that $\{a, b, c\}$ is the input and working alphabet, $Q = \{1, 2, 3, 4, 5, y\}$ is the set of states, y is the accepting state. Then, consider the sequence of moves represented in the following table, where $(\sigma', q', *) \in \delta(\sigma, q)$:

[1] We remark that usually the term computation refers to a description of the sequence of all single moves the machine executes. Rather, here we refer to this concept using the expression *step-by-step computation*.

q	0	1	4	2	1	4	5	1	4	0	2	3	2	4	2	0
σ	a	b	a	c	b	c	a	b	c	b	c	a	b	a	a	c
q'	1	4	2	1	4	5	1	4	0	2	3	2	4	2	0	y
σ'	c	a	c	b	a	b	c	a	b	a	b	b	a	c	b	b
$*$	+	+	+	+	+	−	−	+	+	+	+	−	−	+	−	−

The picture w associated to such a computation $W = W_1 \rhd W_2 \rhd \cdots \rhd W_7$ is given by

$$w = \begin{array}{|c|c|c|c|c|c|c|c|l} \hline a_0 & b & a & c & b & c & c & a & \to W_1 \\ \hline c & a & c & b & a & c_4 & c & a & \to W_2 \\ \hline c & a & c & b_1 & c & b & c & a & \to W_3 \\ \hline c & a & c & a & b & a & b & a_3 & \to W_4 \\ \hline c & a & c & a & b & a_4 & a & b & \to W_5 \\ \hline c & a & c & a & b & c & a_2 & b & \to W_6 \\ \hline c & a & c & a & b_y & b & b & b & \to W_7 \\ \hline \end{array}$$

Proposition 1. *The accepting-computation language of a 1-tape nondeterministic Turing machine is in* REC.

Sketch of the proof. One can prove that, for every given 1-tape nondeterministic Turing machine M, the accepting-computation language L of M is the projection of a suitable language L' in REC. The complete proof of this fact is omitted because of space constraints. Here we just say that, given M, for every picture $w \in L$ it is possible to define a new picture $w' \in L'$ by marking some symbols of w, so that w' encodes all information about the step-by-step computation of M on input w. Then, since such a computation can be described *locally* (the head touches only two cells at each step), L' can be recognized by a tiling system. Hence, L is in REC, too. □

5.2 Overlap of Picture Languages

We now introduce a partial operation in the set of all picture languages (over all alphabets). Given two picture languages L_1 and L_2, we consider every pair of pictures $p \in L_1$ and $q \in L_2$ with the same size and having the first row in common, and we glue them along the first row. The collection of all these pairs is called the *overlap* $L_1 \diamond L_2$.

More formally, given two pictures p and q of the same size (n, m), let $p \times q$ be the picture such that $(p \times q)(i, j) = (p(i, j), q(i, j))$ for every $1 \le i \le n$ and $1 \le j \le m$. Then, the overlap of L_1 and L_2 is defined as

$$L_1 \diamond L_2 = \{p \times q \mid p \in L_1, q \in L_2, \mathrm{r}_p = \mathrm{r}_q, \mathrm{c}_p = \mathrm{c}_q, \\ p(1, j) = q(1, j) \text{ for every } 1 \le j \le \mathrm{c}_p\}$$

Proposition 2. *Given two picture languages in* REC*, their overlap is still in* REC.

Proof. Let L_1 and L_2 be two picture languages over the alphabets Σ_1 and Σ_2, respectively, and assume that they are in REC. Then, for each $i \in \{1,2\}$, there exists a tiling system $\langle \Sigma_i, \Gamma_i, \Theta_i, \pi_i \rangle$ recognizing L_i. Set

$$\mathrm{Top}(\Theta_i) = \{t \in \Theta_i \mid t = \begin{array}{|c|c|}\hline \sharp & \sharp \\ \hline a & b \\ \hline \end{array} \text{ where } a, b \in \Gamma_i \cup \{\sharp\}\}$$

and let $\mathrm{Left}(\Theta_i)$, $\mathrm{Right}(\Theta_i)$, and $\mathrm{Bottom}(\Theta_i)$ be defined analogously. Also, define $\mathrm{Inner}(\Theta_i)$ as the set of tiles of Θ_i that do not belong to any of the previous set. Now, let $\Gamma = \Gamma_1 \times \Gamma_2$ and define Θ as the union of the sets $\mathrm{Inner}(\Theta)$, $\mathrm{Left}(\Theta)$, $\mathrm{Right}(\Theta)$, $\mathrm{Bottom}(\Theta)$, $\mathrm{Top}(\Theta)$, where:

$$\mathrm{Inner}(\Theta) = \{ \begin{array}{|c|c|}\hline (a_1,a_2) & (b_1,b_2) \\ \hline (c_1,c_2) & (d_1,d_2) \\ \hline \end{array} \mid \begin{array}{|c|c|}\hline a_i & b_i \\ \hline c_i & d_i \\ \hline \end{array} \in \mathrm{Inner}\ (\Theta_i),\ i \in \{1,2\}\},$$

$$\mathrm{Left}(\Theta) = \{ \begin{array}{|c|c|}\hline \sharp & (a_1,a_2) \\ \hline \sharp & (b_1,b_2) \\ \hline \end{array} \mid \begin{array}{|c|c|}\hline \sharp & a_i \\ \hline \sharp & b_i \\ \hline \end{array} \in \mathrm{Left}(\Theta_i),\ i \in \{1,2\}\ \},$$

$\mathrm{Bottom}(\Theta)$ and $\mathrm{Right}(\Theta)$ are defined similarly, whereas $\mathrm{Top}(\Theta)$ is given by

$$\mathrm{Top}(\Theta) = \{ \begin{array}{|c|c|}\hline \sharp & \sharp \\ \hline (a_1,a_2) & (b_1,b_2) \\ \hline \end{array} \mid \begin{array}{|c|c|}\hline \sharp & \sharp \\ \hline a_i & b_i \\ \hline \end{array} \in \mathrm{Top}(\Theta_i),\ i \in \{1,2\} \text{ and } \pi_1(a_1) = \pi_2(a_2), \pi_1(b_1) = \pi_2(b_2)\}.$$

Finally, set $\pi = \pi_1 \times \pi_2$, that is, for each pair $(a_1, a_2) \in \Gamma$, set $\pi(a_1, a_2) = (\pi_1(a_1), \pi_2(a_2))$. Clearly, $\langle \Sigma_1 \times \Sigma_2, \Gamma, \Theta, \pi \rangle$ is a tiling system recognizing the overlap of L_1 and L_2.

We are now able to prove the second part of Theorem 1.

Theorem 3. *Given any unary picture language L, if the quasi-unary string language $\phi(L)$ is in* NSPACEREV$_Q$*, then L is tiling recognizable.*

Proof. Since $\phi(L)$ is in NSPACEREV$_Q$, it is recognized by a 1-tape nondeterministic Turing machine M that works in $|x|$ space for any input $x \in Q$, and executes at most $_\circ|x|$ head reversals during each computation. Thus, the accepting-computation language $A(M)$ of such a Turing machine is in REC, by Proposition 1, and so is the language $\bar{A}$ obtained from $A(M)$ by replacing the symbol $\circ_{q_0}$ by $\circ$ in the upper-leftmost cell of each picture in $A(M)$. As a consequence, the following language is in REC, too:

$$A' = \bar{A} \ominus \left(\not\sharp^{*\Theta}\right)^{*\Phi}$$

(observe that any picture in A' can be seen as a picture in $\bar{A}$ possibly extended downwards with rows of blanks).

Now, let us introduce some special picture languages that shall be used to bind the size of a picture, (i.e., they play the role of *mask languages*). Let E_s be the set of all unary squares and set

$$E_h = E_s \oplus h^{*\ominus} \oplus \circ^{**} \qquad \text{and} \qquad E_v = E_s \oplus v^{*\ominus} \oplus \circ^{**}.$$

In other words, any $p \in E_s \cup E_h$ contains, on each row, the quasi-unary string representing its own size, while if $p \in E_v$, then p contains, on each row, the quasi-unary string representing the size of p^R. Moreover consider the picture languages

$$L_s = A' \diamond E_s, \qquad L_h = A' \diamond E_h \qquad \text{and} \qquad L_v = (A' \diamond E_v)^R.$$

and set $L' = L_s \cup L_h \cup L_v$.

By Proposition 2, also L' is tiling recognizable, and it turns out that $L = \pi(L')$. Indeed, by the previous definition, we have that any quasi-unary string x representing a picture of $\pi(L')$ is an accepted input of M, and hence it also represents a picture in L. Thus, L and $\pi(L')$ being unary, we get $\pi(L') \subseteq L$.

On the other hand, assume $p \in L$. First of all, notice that $\phi(p)$ is accepted by M, hence there exists $a \in \bar{A}$ having $\phi(p)$ on the first row and such that $\mathrm{c}_a = \max(\mathrm{r}_p, \mathrm{c}_p)$ and $\mathrm{r}_a \leq \min(\mathrm{r}_p, \mathrm{c}_p)$. Let $a' \in A'$ be the extension of a that has exactly $\min(\mathrm{r}_p, \mathrm{c}_p)$ rows, and notice that a' is a horizontal rectangle or a square, independently of the shape of p. Moreover, consider the picture $u_p = \phi(p)^{\circ|x|\ominus}$. Notice that, if p is a horizontal rectangle, then $u_p \in E_h$; if p is a vertical rectangle, then $u_p \in E_v$, otherwise, if p is a square, $u_p \in E_s$. In any case, u_p has the same size as a'. Hence, if p is a square or a horizontal rectangle, then we have $p = \pi(a' \diamond u_p)$; otherwise we have $p = \pi\left((a' \diamond u_p)^R\right)$. In all cases, $p \in \pi(L')$ and hence $L \subseteq \pi(L')$. Thus, $L = \pi(L')$ is in REC_1 and this concludes the proof.

6 Square Languages

In this last section we focus on unary square languages, that is on unary picture languages whose elements are all squares. As should be clear at this moment of the exposition, square languages are nothing but sets of positive integers, and so far we represented them by unary strings over the alphabet $\{\circ\}$. In the following definition, we introduce a subclass of $\mathrm{NSPACEREV}_Q$ that concerns only square languages and their representation.

Definition 5. $\mathrm{NSPACEREV}_U$ *is the class of unary string languages that can be recognized by 1-tape nondeterministic Turing machines working within n space and executing at most n head reversals, for any input of length n.*

Integers can also be represented with the classical binary encoding and this suggest to define the binary complexity class corresponding to the previous definition.

Definition 6. $\mathrm{NSPACEREV}_B$ *is the class of binary string languages that can be recognized by 1-tape nondeterministic Turing machines working within 2^n space and executing at most 2^n head reversals, for any input of length n.*

Notice that the families NSPACEREV_U and NSPACEREV_B are related to the well-known time complexity classification. In particular, denoting by $\text{NTIME}_U(f(n))$ (resp. $\text{NTIME}_B(f(n))$) the class of unary (resp. binary) string languages that can be recognized by 1-tape nondeterministic Turing machines working within $f(n)$ time for any input of length n, we have the following relations:

$$\text{NTIME}_U(n) \subseteq \text{NSPACEREV}_U \subseteq \text{NTIME}_U(n^2),$$

$$\text{NTIME}_B(2^n) \subseteq \text{NSPACEREV}_B \subseteq \text{NTIME}_B(4^n). \quad (1)$$

Theorem 1 can then be re-stated using these new classes, obtaining the following corollary.

Corollary 1. *Given a unary square language L, the following statements are equivalent:*

- *L is in REC_1,*
- *$\{\circ^{\text{r}_p} \mid p \in L\} \in \text{NSPACEREV}_U$,*
- *$\{Bin(\text{r}_p) \mid p \in L\} \in \text{NSPACEREV}_B$,*

where $Bin(n)$ is the binary encoding of the positive integer n.

The previous corollary provides a useful tool to verify whether a unary square language is tiling recognizable. For instance, it proves that the set of unary square pictures whose size is a prime number is in REC_1, since it is well-known that the set of prime numbers is recognizable in polynomial time[1]. More generally, if π is a NP problem, let L_π be the language of all binary encodings of positive instances of π. Then, the picture language $\{p \in \circ^{**} \mid \exists x \in L_\pi \text{ such that Bin } (\text{r}_p) = 1x\}$ belongs to REC_1.

A further, more complex, tiling-recognizable picture language can be built by considering INEQ(RE,2), i.e. the inequality problem of regular expressions with squaring, studied by Meyer and Stockmeyer in [9,10]. It is known that this problem is complete in the class $\text{NEXPTIME} = \bigcup_{c \geq 1} \text{NTIME}(2^{cn})$ and hence it is not even included in NP by well-known separation results [11]. It is not difficult to prove that a rather natural binary encoding of INEQ(RE,2) belongs to $\text{NTIME}_B(2^n)$ and hence, by the previous corollary and Equation 1, the corresponding family of unary square pictures is tiling recognizable.

Another consequence of Corollary 1 concerns the construction of unary square languages that are not tiling recognizable. For instance one can prove the existence of a unary square language that is not tiling recognizable, but such that the set of binary encoding of its sizes is not too far (from a complexity view point) from the class NSPACEREV_B. In order to present such an example, for any function $f : \mathbb{N} \to \mathbb{R}^+$, let us define 2t-$\text{NTIME}_B(f)$ as the class of binary string languages that are recognizable by 2-tape nondeterministic Turing machines working within time $f(n)$ on every input of length n.

Proposition 3. *There exists a unary square picture language $L \notin \text{REC}_1$ such that the string language $S = \{x \in \{0,1\}^* \mid 1x = Bin(\text{r}_p)$ for a picture $p \in L\}$ belongs to 2t-$\text{NTIME}_B(4^n \log n)$.*

Proof. The existence of such language is guaranteed by a property proved in [11]. If $T_1, T_2 : \mathbb{N} \to \mathbb{R}^+$ are two running-time functions such that $T_1(n+1)/T_2(n)$ tends to 0 as n goes to infinity, then there exists a language $S \subseteq \{0,1\}^*$ that belongs to 2t-NTIME$_B(T_2(n))$ but does not belong to 2t-NTIME$_B(T_1(n))$. Setting $T_1(n) = 4^n$, $T_2(n) = 4^n \log n$, and observing that 2t-NTIME$_B(4^n) \supseteq$ NSPACEREV$_B$, by Theorem 1 we have that S is in 2t-NTIME$_B(4^n \log n)$ whereas L cannot be tiling recognizable.

Concluding, we observe that a natural problem arising from our characterization result is whether a separation property, similar to the one proved in [11], also holds for complexity classes defined by bounding the number of head reversals. This would lead to simpler unary picture languages that are not tiling recognizable.

References

1. M. Agrawal, N. Kayal, N. Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2): 781-793, 2004.
2. M. Anselmo, D. Giammarresi, M. Madonia. Regular expressions for two-dimensional languages over one-letter alphabet. In *Proc. 8th DLT*, C.S. Calude, E. Calude and M.J. Dinneen (Eds.), LNCS 3340, 63–75, Springer-Verlag, 2004.
3. D. Giammarresi, A. Restivo. Recognizable picture languages. *Int. J. Pattern Recognition and Artificial Intelligence*, Special Issue on Parallel Image Processing, 31–42, 1992.
4. D. Giammarresi, A. Restivo. Two-dimensional languages. In *Handbook of Formal Languages*, G. Rosenberg and A. Salomaa (Eds.), Vol. III, 215 – 268, Springer-Verlag, 1997.
5. D. Giammarresi, A. Restivo, S. Seibert, W. Thomas. Monadic second order logic over rectangular pictures and recognizability by tiling system. *Information and Computation*, 125(1):32–45, 1996.
6. K. Inoue, I. Takanami. A survey of two-dimensional automata theory. In *Proc. 5th Int. Meeting of Young Computer Scientists*, J. Dasson, J. Kelemen (Eds.), LNCS 381, 72–91, Springer-Verlag, 1990.
7. J. Kari, C. Moore. New results on alternating and non-deterministic two-dimensional finite state automata. In *Proc. 18th STACS*, A. Ferreira, H. Reichel (Eds.), LNCS 2010, 396–406, Springer-Verlag, 2001.
8. O. Matz. Regular expressions and context-free grammars for picture languages. In *Proc. 14th STACS*, LNCS 1200, 283–294, Springer-Verlag, 1997.
9. A.R. Meyer and L.J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. Proc. 13th Annual IEEE Symp. on Switching and Automata Theory 125-129, 1972.
10. A.R. Meyer and L.J. Stockmeyer. Words problems requiring exponential time. Proc. 5th ACM Symp. on Theory of Computing 1-9, 1973.
11. J. I. Seiferas, M. J. Fischer, A. R. Meyer. Separating nondeterministic time complexity classes. *Journal of ACM*, 25(1): 146–167, 1978.
12. R. Siromoney. Advances in array languages. In *Graph-grammars and their applications to Computer Science*, Ehrig et al. Eds., LNCS 291, 549–563, Springer-Verlag, 1987.