Deterministically and Sudoku-Deterministically Recognizable Picture Languages

Bernd Borchert
Klaus Reinhardt

WSI-2006-09

Universität Tübingen
Wilhelm-Schickard-Institut für Informatik
Arbeitsbereich Theoretische Informatik/Formale Sprachen
Sand 13
D-72076 Tübingen

borchert/reinhardt@informatik.uni-tuebingen.de

ISSN 0946-3852

Deterministically and Sudoku-Deterministically Recognizable Picture Languages

Bernd Borchert and Klaus Reinhardt

Universität Tübingen, Sand 13, 72076 Tübingen, Germany

{borchert,reinhard}@informatik.uni-tuebingen.de

Abstract. The recognizable 2-dimensional languages are a robust class with many characterizations, comparable to the regular languages in the 1-dimensional case. One characterization is by tiling systems. The corresponding word problem is NP-complete. Therefore, notions of determinism for tiling systems were suggested. For the notion which was called "deterministically recognizable" it was open since 1998 whether it implies recognizability. By showing that acyclicity of grid graphs is recognizable we answer this question positively. In contrast to that, we show that non-recognizable languages can be accepted by a generalization of this tiling system determinism which we call Sudoku-determinism. Its word problem, however, is still in linear time. We show that Sudoku-determinism even contains the set of 2-dimensional languages which can be recognized by 4-way alternating automata.

1 Introduction

We regard pictures as two-dimensional rectangular arrays of symbols of a given alphabet. It turns out that various formalisms which characterize the regular languages in the one-dimensional case lead to different classes of two-dimensional languages. For example, it was shown already in 1967 [BH67] that nondeterministic 4-way finite automata (4NFA) are strictly more powerful than deterministic ones (4DFA).

In [GR92], a set of pictures L over an alphabet Σ is defined as *recognizable* if it is recognized by a finite tiling system $(\Sigma, \Gamma, \Delta, \pi)$ with $\pi : \Gamma \mapsto \Sigma$ and $L = \pi(L')$ for the *local* language L' over the alphabet Γ determined by the 2x2 tiles from Δ. Here "local" means that L' consists of all pictures which "locally look like Δ", see the formal definition later. The recognizable picture languages REC are characterized in [GRST94] as the ones expressible in existential monadic second order logic over two-dimensional structures. This emphasizes the nondeterministic character of this class: recognizing a given picture p can be viewed as a process of finding a pre-image p' in the local language L' with $\pi(p') = p$. This word problem is indeed NP-complete for certain tiling systems.

A closer link to computational complexity is established in [Bor03] where NP is characterized with the notion of recognizability by padding 1-dimensional words with blanks to form an n-dimensional cube. With a similar idea, the set of

context-sensitive languages is characterized in [LS97a] as the set of "frontiers" of picture languages.

The natural subclass UREC of REC, for which it is required that each picture $p \in L$ has exactly one unique pre-image $p' \in L'$, is different from REC, as recently shown by Anselmo et al. [AGMR06].

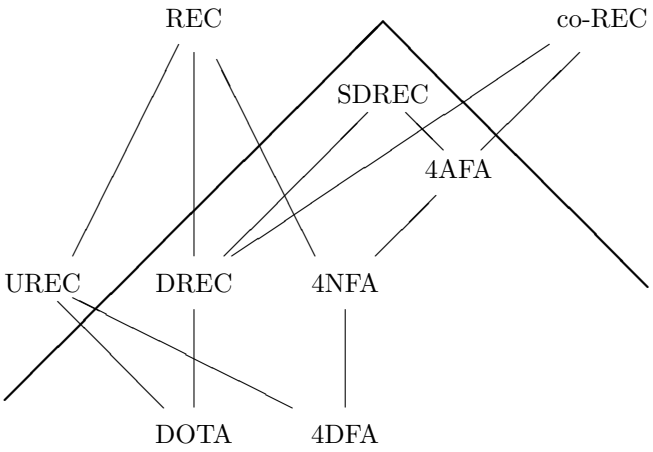
For practical applications, however, we would like to have an appropriate deterministic process starting with a given picture p over Σ and ending with a pre-image p' over Γ. The deterministic 4-way finite automata 4DFA and the deterministic online tessellation automata DOTA are well-known examples of such deterministic processes, see [GR97]. In [AGMR06a] a subclass of UREC was defined with the restriction that the pre-image-symbols on a column respectively row are uniquely determined by the symbols there and pre-image-symbols on the neighbor column respectively row. A more restricted subclass allows to uniquely determine a pre-image-symbol on a postion by the symbols there and the pre-image-symbols on the left and upper (respectively upper and right) neighbor. Both classes allow a deterministic process to find the pre-image proceeding in a certain direction. A bit more general already is the deterministic tiling system in [Chl84], which allows to work either left-determined or right-determined within a row but still only works in one vertical direction.

In this paper we consider a more general deterministic process which was defined in [Rei98]. We call the corresponding class DREC. The intermediate configurations are pictures over $\Sigma \cup \Gamma$. One step is a replacement of an $s \in \Sigma$ by a $g \in \Gamma$ with $s = \pi(g)$ which can be performed only if it is locally the only possible choice. This allows the deterministic process to change its direction within the picture and in this way for example connectivity can be recognized deterministically [Rei98]. On the other hand a new problem arises from situations where the deterministic process is not able to proceed because there are more than one possibilities for each position in Σ but there exists a pre-image which locally looks like having been found deterministically. For that reason it was an open problem whether DREC $\subseteq$ REC.

Directed grid graphs can in an obvious way be seen as pictures over an alphabet containing four arrows. Let L_{dag} be the picture language consisting of the directed grid graphs which do not contain a cycle. It was conjectured in [KM01] that L_{dag} is in REC. As a main result of this paper we prove this conjecture. Moreover, this result will be the key lemma for proving DREC $\subseteq$ REC: the picture language L_{dag} is used to check if the assumed directions of the deterministic steps contain no cycle – which is needed in the simulation for DREC $\subseteq$ REC.

We furthermore consider a stronger version of determinism which we call Sudoku-determinism. We call the class SDREC. The idea is that instead of determining the pre-image symbol on a position in one shot, we keep a set of possible pre-image symbols for each position and reduce a set in one step of the process by excluding impossible pre-images – a method similar to the method of solving a Sudoku puzzle.

We show that SDREC contains both DREC and 4AFA, the set of languages recognized by alternating 4-way finite automata. There are non-recognizable languages in 4AFA, as shown by Kari & Moore [KM01].
A diagram below is showing the known inclusions among the classes mentioned above. Note that in the 1-dimensional case each shown class is equal to the set of regular languages REG.



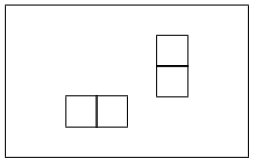
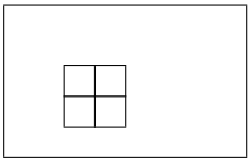
The thick line separates the classes which consist only of picture languages in PTIME from the classes REC, UREC, and co-REC which may contain non-feasible picture languages.

2 Definitions and examples

The following definitions are from [GR92]. A *picture* over an alphabet Σ is a two-dimensional array of elements of Σ. The set of pictures of size (m, n) is denoted by $\Sigma^{m,n}$. A *picture language* is a subset of $\Sigma^{*,*} := \bigcup_{m,n\geq 0} \Sigma^{m,n}$. Let $T_{m,n}(p)$ be the set of all sub-pictures of p with size (m, n). For a $p \in \Sigma^{m,n}$, we define $\hat{p} \in \Sigma^{m+2,n+2}$, adding a frame of symbols $\# \notin \Sigma$.

$\hat{p} :=$

#	#	#	#	#	#
#					#
#		p			#
#					#
#					#
#	#	#	#	#	#

Let Θ be some set of pictures of size (2,2) over the alphabet $\Gamma \cup \{\#\}$, such a set is called a *set of tiles*. A picture language $\mathcal{L}(\Theta) := \{p \in \Gamma^{*,*} | T_{2,2}(\hat{p}) \subseteq \Theta\} \subseteq \Gamma^{*,*}$ is called *local*. A picture language $\mathcal{L}(\Sigma, \Gamma, \Theta, \pi) = \pi(\mathcal{L}(\Theta)) \subseteq \Sigma^{*,*}$ is called *recognizable*, in this definition π is a mapping $\Gamma \to \Sigma$ and is called a *projection*. The quadruple $\tau = (\Sigma, \Gamma, \Theta, \pi)$ is called a *tiling system*.
Let Δ be some set of pictures of size (1,2) or (2,1) over the alphabet $\Gamma \cup \{\#\}$, such a set is called a set of *domino tiles*. A picture language $\mathcal{L}_{hv}(\Delta) := \{p \in \Gamma^{*,*} | T_{1,2}(\hat{p}) \cup T_{2,1}(\hat{p}) \subseteq \Delta\}$ is called *hv-local*. A quadruple $\tau = (\Sigma, \Gamma, \Delta, \pi)$ is called a *domino tiling system*, and the language recognized by τ is defined as $\mathcal{L}(\pi) = \mathcal{L}(\Sigma, \Gamma, \Delta, \pi) = \pi(\mathcal{L}(\Delta))$. According to [LS97b], for every tiling system τ there is a domino tiling system τ' such that $\mathcal{L}(\tau) = \mathcal{L}(\tau')$, and vice versa. This means we can use domino tiles  instead of tiles: 
The domino tiles make is easier and more natural to describe a deterministic process since we have to consider only 4 neighbors.

2.1 The Recognition of a Picture as a Deterministic Process

Let $\tau = (\Sigma, \Gamma, \Delta, \pi)$ be a domino tiling system. Given a picture p over the alphabet Σ we define a picture s_p of the same size in which we initialize every position (i,j) by the set $s_p(i,j) := \pi^{-1}(p(i,j)) \in 2^\Gamma$ of possible pre-image symbols. We first describe one step for a Sudoku-deterministic process: For $s, s' \in (2^\Gamma)^{*,*}$ (of the same size) we allow a step $\hat{s} \underset{sd(\tau)}{\Rightarrow} \hat{s}'$ if for all positions (i,j) in s we have that the set $\hat{s}'(i,j)$ consists of the elements in the set $\hat{s}(i,j)$ which have in each of the four neighboring sets a respective element such that the respective domino tile is in Δ, formally

$$s'(i,j) = \{x \in s(i,j) \mid \exists y_1, y_2, y_3, y_4 \in \Gamma \cup \{\#\} \text{ such that}$$
$$y_1 \in \hat{s}(i+1,j), y_2 \in \hat{s}(i-1,j), y_3 \in \hat{s}(i,j+1), y_4 \in \hat{s}(i,j-1)$$
$$\text{and } \boxed{x}\boxed{y_1}, \boxed{y_2}\boxed{x}, \begin{array}{|c|}\hline x \\ \hline y_3 \\ \hline \end{array}, \begin{array}{|c|}\hline y_4 \\ \hline x \\ \hline \end{array} \in \Delta\}.$$

(Assume in this definition by abuse of notation that $\# = \{\#\}$.) Note that only elements from $s'(i,j)$ are dropped which will never be the symbol of a valid pre-image at this position. In other words: one step excludes one or more possibilities for which one can be sure about that it will not be a solution, and after this step new opportunities for excluding more possibilities may show up – this is how usually a Sudoku puzzle is solved.
The definition in [Rei98] can be formulated as a special case: let $\hat{s} \underset{d(\tau)}{\Rightarrow} \hat{s}'$ if $\hat{s} \underset{sd(\tau)}{\Rightarrow} \hat{s}'$ and for all positions (i,j) the set $\hat{s}'(i,j)$ is either unchanged, i.e. $\hat{s}'(i,j) = \hat{s}(i,j)$, or consists of a single element, i.e. $|\hat{s}'(i,j)| = 1$, or both.

For a given domino tiling system $\tau = (\Sigma, \Gamma, \Delta, \pi)$ let the accepted language $\mathcal{L}_{sd}(\tau)$ be defined as the set of pictures p for which there exists a way to transform the initialized picture $\hat{s}_p$ in finitely many steps into a picture in which every position consists of exactly one element, and which can not be transformed further, formally

$$\mathcal{L}_{sd}(\tau) := \{p \in \Sigma^{*,*} | \exists p' \in \mathcal{L}(\tau) \text{ s.t. } \hat{s}_p \overset{*}{\underset{sd(\tau)}{\Longrightarrow}} \{\hat{p}'\}\}.$$

In the above formula $\{\hat{p}'\}$ stands for the picture which is of the same size as $\hat{p}'$ and has instead of an letter $p'(i,j)$ the singleton set $\{p'(i,j)\}$ as a letter at position (i,j). The language $\mathcal{L}_d(\tau)$ is defined the same way using $\overset{*}{\underset{d(\tau)}{\Longrightarrow}}$ instead of $\overset{*}{\underset{sd(\tau)}{\Longrightarrow}}$.

The classes of the *Sudoku-deterministically recognizable* and the *deterministically recognizable* picture languages SDREC and DREC are defined as the languages L for which there is a domino tiling system τ recognizing L in that way, i.e.

$$\text{SDREC} := \{L \subseteq \Sigma^{*,*} \mid \text{there is a } \tau = (\Sigma, \Gamma, \Delta, \pi) \text{ such that } L = \mathcal{L}_{sd}(\tau)\},$$

$$\text{DREC} := \{L \subseteq \Sigma^{*,*} \mid \text{there is a } \tau = (\Sigma, \Gamma, \Delta, \pi) \text{ such that } L = \mathcal{L}_{d}(\tau)\}.$$

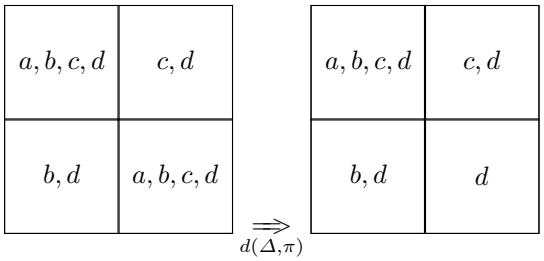
Clearly it holds for every domino tiling system $\tau = (\Sigma, \Gamma, \Delta, \pi)$:

$$\mathcal{L}_{\mathrm{d}}(\tau) \subseteq \mathcal{L}_{\mathrm{sd}}(\tau) \subseteq \mathcal{L}(\tau).$$

Example. Let $\tau = (\Sigma, \Gamma, \Delta, \pi)$ be defined as follows. $\Gamma = \{a, b, c, d\}$ and $\Delta = \{$ $\begin{array}{|c|}\hline a \\ \hline c \\ \hline \end{array}$, $\begin{array}{|c|}\hline b \\ \hline c \\ \hline \end{array}$, $\begin{array}{|c|}\hline a \\ \hline d \\ \hline \end{array}$, $\begin{array}{|c|}\hline b \\ \hline d \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline a & c \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline b & c \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline a & d \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline b & d \\ \hline \end{array}$ $\}$. Then it holds

$$\begin{array}{|c|c|}\hline a,b,c,d & a,b,c,d \\ \hline b,d & a,b,c,d \\ \hline \end{array} \underset{sd(\Delta,\pi)}{\Longrightarrow} \begin{array}{|c|c|}\hline a,b,c,d & a,c,d \\ \hline b,d & b,d \\ \hline \end{array}$$

$$\begin{array}{|c|c|}\hline a,b,c,d & c,d \\ \hline a,b,d & a,b,c,d \\ \hline \end{array} \underset{sd(\Delta,\pi)}{\Longrightarrow} \begin{array}{|c|c|}\hline a,b,c,d & c,d \\ \hline a,b,d & c,d \\ \hline \end{array}$$



Note that the first two examples show that two streams of information can cross during a Sudoku-deterministic process while for a deterministic process this is not possible: once a "line" is colored it cannot be "traversed" later.

The definition of $\underset{d(\Delta,\pi)}{\Rightarrow}$ and $\underset{sd(\Delta,\pi)}{\Rightarrow}$ forces the sets at every position to be reduced as much as possible. It is easy to see that in a "less deterministic" formulation for the two relations where not all informations on neighbors has to be used at once, every derivation will lead in the end to the same s' with $|s'(i,j)| = 1$ for all i, j (if it exists), this means the "less deterministic" versions of $\underset{d(\Delta,\pi)}{\Rightarrow}$ and $\underset{sd(\Delta,\pi)}{\Rightarrow}$ are confluent in successfull cases.

Let a L be one of the languages in DREC and SDREC. Then L is computable in PTIME. The algorithm just initializes an 2-dimensional array of the size of the input p with the the values of s_p and then iterates the stepwise transformation until no changes occur anymore. Finally it checks whether there is exactly one element left in every set of the array. This shows that picture languages in DREC and SDREC, seen as algorithmic problems, are in PTIME. Using the confluence property one can even write an linear time algorithm for this word problem, see [Rei98][Cor. 2].

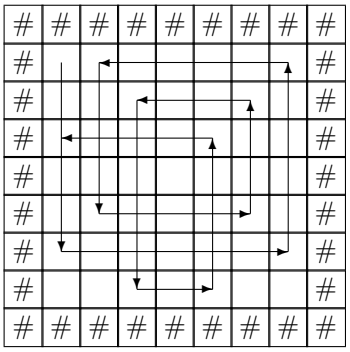
3 The power of Sudoku-determinism

In this section we characterize the power of Sudoku-determinism. In order to demonstrate this power we use a language not in REC as an example for a language which can be Sudoku-deterministically recognized. The language is basically the language L_P from [KM01] (with just some formal changes): it consists of the squares of odd side length over the alphabet $\{0, 1, 2, 3\}$ such that the middle vertical column is filled with 2's, the middle vertical line is filled with 3's, the lower left quarter is a permutation matrix and symmetric to the lower right quarter. The upper right quarter contains only a diagonal of 1's and the

upper left quarter a diagonal of 1's shifted by one to the right.

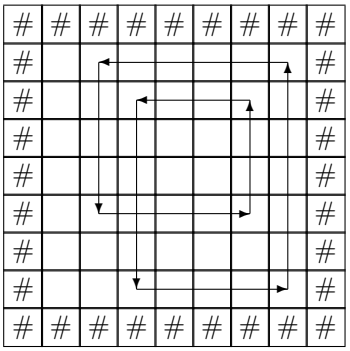
#	#	#	#	#	#	#	#	#
#		1		2			1	#
#			1	2		1		#
#	1			2	1			#
#	3	3	3	2	3	3	3	#
#		1		2		1		#
#	1			2			1	#
#			1	2	1			#
#	#	#	#	#	#	#	#	#

The tiling Δ is constructed in a way such that information can only be transported along the following direction (which bends a positions with a 1):



Any deviation from the symmetry would lead to a cycle for which the deterministic process can not determine the pre-image symbol.

#	#	#	#	#	#	#	#	#
#		1		2			1	#
#			1	2		1		#
#	1			2	1			#
#	3	3	3	2	3	3	3	#
#		1		2		1		#
#	1			2	1			#
#			1	2			1	#
#	#	#	#	#	#	#	#	#



Crossing directions at several positions positions corresponds to a crossproduct of the information which means that a Sudoku-deterministic step can reduce the set to two of four possible symbols going in one direction and doing the other reduction at some other time, see the examples of the previous section.

Remark. A deterministic process using 2×2-tiles would allow processing information over one diagonal and to cross later over the other diagonal. In this way a non-recognizable picture language similar to the one where two permutation matrices are compared (see above) could be recognized deterministically. This means the result of Theorem 3 strongly depends on the planarity (and finite degree) of the graph describing the direction of the deterministic conclusions. This is in contrast to SDREC which is (like REC) robust against changing the form of the tiles.

The preceding example shows that SDREC contains non-recognizable problems. This also follows from the result below that SDREC contains 4AFA.

Theorem 1. 4AFA $\subseteq$ SDREC

Proof. Let M be a 4-way finite automaton for a picture language L over the alphabet Σ. Assume w.l.o.g. that M moves in every step from an existential state i.e. in the next step M is always on one of the neighboring positions but M stays on the same position in a transition from an universal state. Moreover, assume w.l.o.g. that M has besides its unique initial state q_0, which can be assumed to be universal, a unique final state q_e and that M may start and end in an arbitrary position of the picture. Let S be the set of states of M and let f be some element standing for "flooding" – what is explained later. The set Γ for the domino tiling system $\tau = (\Sigma, \Gamma, \Delta, \pi)$ used by the Sudoku process is defined as the subset of $\Sigma \times (2^{S \setminus \{q_0\}} \cup \{f\})$ such that for a set (a, A) with $A \subset S \setminus \{q_0\}$ a universal state q is in A iff for all transitions of M from state q – seeing letter a – the next state q' is in A as well. The projection π is defined as $\pi(a, ...) = a$. It remains to define the set Δ of domino tiles. A horizontal domino tile $\boxed{(a, A)}\boxed{(b, B)}$ with $a, b \in \Sigma$ and $A, B \subseteq S \setminus \{q_0\}$ is in Δ if both A and B contain q_e and it is perfectly consistent with the moves of M, i.e. the following holds:

1. each existential state e is in A if there is a transition of M from state e – seeing letter a – to the right and the next state s of M is in B.
2. each existential state e is in B if there is a transition of M from state e – seeing letter b – to the left and the next state s of M is in A.

Moreover, for all $a, b \in \Sigma$ the tile $((a, f), (b, f))$ for the flooding state f is in Γ (these are the only tiles the flooding state f is in). The vertical domino tiles of Δ are defined analogously (turned by 90 degree).

As an example, let e_2, e_3, q_e be following states of the universal state u_1, and let e_5 be following state of the existential state e_3 – seeing letter a – moving to the right (in state e_2 – seeing letter a – the machine M moves in some other

direction, say upwards). Then in the following horizontal tile these conditions are fulfilled (besides others):

$(a, \{u_1, e_2, e_3, q_e\})$	$(b, \{u_4, e_5, q_e\})$

Now that we have defined $\tau = (\Sigma, \Gamma, \Delta, \pi)$ we claim that $\mathcal{L}_{sd}(\tau) = L(M)$. The Sudoku process will simulate the computation of M starting at the leaves of the computation of M. Note the following inductive argument: If there is an accepting computation subtree of M starting in a state q at some position then all subsets A not containing q will be eliminated at that position at some stage in the Sudoku-process. This means if there is an accepting computation tree of M starting in a state q_0 then all subsets A will be eliminated at that position. Subsequently the elimination continues on all other positions by a flooding process which only leaves the symbol f on every position. On the other hand, if there is no accepting computation tree of M starting in a state q_0, then every position will maintain at least the subset A of states which have an accepting computation tree of M (which might be $\{q_e\}$ ...); these can not be eliminated because they are consistent with their neighbors.

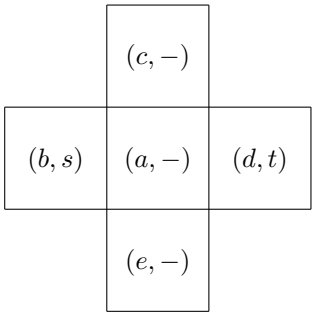
The reader may be disappointed that in the end the Sudoku process shows only the flooding state in every position and does not show a trace of the tree of the alternating computation of M. Note that this is no surprise: the latter can't be expected because 4AFA contains non-recognizable languages [KM01]. □

Theorem 2. DREC ⊆ SDREC

Proof. Let $\tau = (\Sigma, \Gamma, \Delta, \pi)$ be a domino tiling system. It suffices to construct a domino tiling system $\tau' = (\Sigma, \Gamma', \Delta', \pi')$ such that

$$\mathcal{L}_{sd}(\tau') = \mathcal{L}_d(\tau).$$

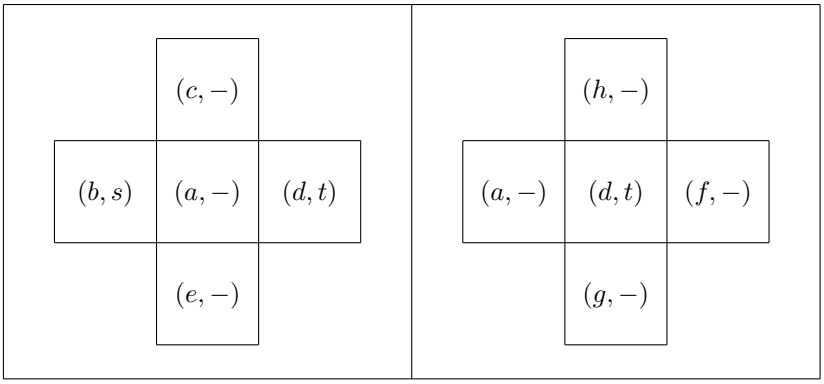
Let the minus symbol "$-$" be an element neither in Γ nor Σ, and define Γ' to be the set of 5-tuples of the pairs from $\Sigma \times (\Gamma \cup \{-\})$. Imagine the five pairs from an element of Γ to be arranged as a "star": the first pair is in the center of the star, the second is on the top of the center, the third is right to the center, etc. For example, the 5-tuple $((a,-),(c,-),(d,t),(e,-),(b,s))$ should be imagined this way:



Define π' to map a star to the letter of the central position, i.e.

$$\pi'(((a,\ ),(\ ,\ ),(\ ,\ ),(\ ,\ ),(\ ,\ ))) = a.$$

It remains to define the set of domino tiles Δ'. The two conditions (A) and (B) for a domino tile for being in Δ' are the following:
(A) The two stars on the tile are *consistent*. This means for example for a horizontal tile: the central pair of the left star is equal to the left pair of the right star, and the right pair of the left star is equal to the central pair of the right star, for example the following horizontal tile fulfills this condition.



For vertical domino tiles the analogous condition has to be fulfilled in the vertical direction.
(B) The two stars on a tile are both from the following subset Γ_r of Γ. Γ_r is the set of stars such that if the color in the central position is uniquely determined via the domino tiling system $(\Sigma, \Gamma, \Delta, \pi)$ by the four surrounding pairs and the left coordinate (= letter) of the pair in the central position (like in a step of a DREC process), then the second coordinate (= color) of the pair in the central position has to be that uniquely determined color, otherwise it has to be $-$. Examples: The left star in the example domino tile above is in Γ_r if the color at the central position is not yet determined via the domino tiling system $(\Sigma, \Gamma, \Delta, \pi)$ by the

surrounding pairs, i.e. even the already determined colors s to the left and t to the right do not yet determine the color in the center because it shows a $-$. The second coordinate of the central position of the right star, on the other hand, is already be determined without having any determined neighbor. Note that such positions are necessary for a DREC process to start.

Starting on a given picture, the Sudoku process using the domino tiling system τ' will first of all eliminate all colors (= stars) which are are not consistent with the letters at the four surrounding positions – this is done via the condition (A) for the tiles. Now assume that a picture is in $\mathcal{L}_d(\Sigma, \Gamma, \Delta, \pi)$. Then the DREC process of finding deterministically the coloring of the picture is simulated 1-1 by the Sudoku deterministic process using the domino tiling system $(\Sigma, \Gamma', \Delta', \pi')$. On the other hand, if there is a Sudoku deterministic process using tiling system $(\Sigma, \Gamma', \Delta', \pi')$ ending in a unique coloring of every position then this process shows a DREC process using the tiling system $(\Sigma, \Gamma, \Delta, \pi)$ which finds the same coloring. □

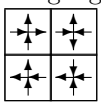
4 Directed acyclic graphs

In this section we will define the picture language L_{dag} consisting of the acyclic grid graphs, and will show that L_{dag} is recognizable. This will be the key result for showing DREC ⊆ REC.

First we prepare the definition of graphs on the grid: Let $\Gamma = \mathbb{Z}^4$ where each component represents one of the four directions (left, right, up, down) $\gamma = (l(\gamma), r(\gamma), u(\gamma), d(\gamma) \in \Gamma$ and let $L_{con}, L'_{con} \subseteq \Gamma^{*,*}$ be the local picture languages defined by the set of tiles $\Delta_{con} :=$

$$\{\begin{array}{|c|c|}\hline \gamma & \gamma' \\ \hline \delta & \delta' \\ \hline \end{array} \in (\{\#\} \cup \Gamma)^{2,2} \mid (\gamma = (l, r, u, d) \wedge \gamma' = (l', r', u', d')) \rightarrow r = -l',$$
$$(\delta = (l, r, u, d) \wedge \delta' = (l', r', u', d')) \rightarrow r = -l',$$
$$(\gamma = (l, r, u, d) \wedge \delta = (l', r', u', d')) \rightarrow d = -u',$$
$$(\gamma' = (l, r, u, d) \wedge \delta' = (l', r', u', d')) \rightarrow d = -u'\}$$

checking that outgoing directed arcs correspond to an ingoing arc at the corresponding neighbor (which means consistency of directions).

Let $\Gamma = \{-1, 1\}^4$ and we identify for example $(-1, 1, 1, 1) =$  . Now we define an local picture language $L_{locdag} \subseteq L_{con} \cap \Gamma^{*,*}$ where we do not allow local 4-cycles like i. e.  and we do not allow sources $(1, 1, 1, 1)$ or sinks $(-1, -1, -1, -1)$.

This is done by set of tiles $\Delta_{locdag} :=$

$$\{\begin{array}{|c|c|}\hline \gamma & \gamma' \\ \hline \delta & \delta' \\ \hline \end{array} \in \Delta_{con} \cap (\{\#\} \cup \Gamma)^{2,2} \mid (\ \gamma = (l, x_1, u, -x_4) \wedge \gamma' = (-x_1, r, u', x_2) \wedge$$
$$\delta = (l', -x_3, x_4, d') \wedge \delta' = (x_3, r', -x_2, d)) \rightarrow$$
$$\exists i, j\ x_i \neq x_j\}$$

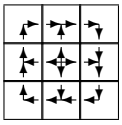
In other words:

$$\Delta_{locdag} := \{\begin{array}{|c|c|}\hline \# & \gamma \\ \hline \# & \gamma' \\ \hline \end{array}, \begin{array}{|c|c|}\hline \gamma & \# \\ \hline \gamma' & \# \\ \hline \end{array} \mid \gamma = (l, r, u, x), \gamma = (l', r', -x, d)\}$$
$$\cup\ \{\begin{array}{|c|c|}\hline \# & \# \\ \hline \gamma & \gamma' \\ \hline \end{array}, \begin{array}{|c|c|}\hline \gamma & \gamma' \\ \hline \# & \# \\ \hline \end{array} \mid \gamma = (l, x, u, d), \gamma' = (-x, r, u', d')\}$$
$$\cup\ \{\begin{array}{|c|c|}\hline \gamma & \gamma' \\ \hline \delta & \delta' \\ \hline \end{array} \mid \gamma = (l, x_1, u, -x_4), \gamma' = (-x_1, r, u', x_2), \delta = (l', -x_3, x_4, d'),$$
$$\delta' = (x_3, r', -x_2, d), \exists i, j\ x_i \neq x_j\}$$

Lemma 1. *A picture in L_{locdag} describes a directed acyclic graph.*

Proof. Assume by contradiction that the picture contains a cycle. Then there must be a cycle with a minimal enclosed part of the plane. Since the cycle can not be a local 4-cycle, there must be an edge of the grid connecting a point on the cycle towards the inside. Let w.l.o.g the direction of this edge lead inside. Then nondeterministically follow directed edges until you reach a point on the cycle or a point where you have been before. This leads to a cycle enclosing a smaller part of the plane in contradiction to the assumption. □

Let $L_{dag} \subset \{-1, 0, 1\}^{4^*}$ be the language of pictures describing a directed acyclic graph, $L'_{dag} := L_{dag} \cap \{-1, 1\}^{4^*}$, $L^-_{dag} := L_{dag} \cap \{-1, 1\}^4 \setminus \{(1, 1, 1, 1)\}^*$ and $L^+_{dag} := L_{dag} \cap \{-1, 1\}^4 \setminus \{(-1, -1, -1, -1)\}^*$. Since sources or sinks can be contained inside a cycle, we can not detect a cycle locally anymore:



We will now prove the following chain of implications: Lemma 1 $\Rightarrow$ $L^-_{dag} \in$REC (Theorem 6) $\Rightarrow$ $L'_{dag} \in$ REC (Theorem 5) $\Rightarrow$ $L_{dag} \in$ REC (Theorem 4) $\Rightarrow$ $DREC \subseteq REC$

Theorem 3. $DREC \subseteq REC$

Proof. For a given picture $p \in \Sigma^{*,*}$ guess a $p' \in (\Sigma \times \{-1, 0, 1\}^4)^{*,*}$ such that $p = \pi(p')$ is the projection to the first component and the second component

is in L_{dag} using Theorem 4. Then check if for each position in the picture the symbols on these neighbors from which an edge leads to this position are together sufficient to determine the symbol on the position deterministically. This can w.l.o.g. be done locally with 3×3-tiles. □

The following solves an open problem in [KM01]:

Theorem 4. $L_{dag} \in REC$

Proof. For a given picture $p \in \{-1, 0, 1\}^{4^*,*}$ guess for every occurring 0 either -1 or 1 and check if the resulting p' is in L'_{dag} using Theorem 5. If p had a cycle, p' must have the same cycle. If p had no cycle, then the existence of a p' without a cycle follows by a stepwise construction replacing a 0 by either -1 or 1 as long as this does not lead to a cycle. If both possibilities would lead to a cycle, there must have been a cycle before. □

Theorem 5. $L'_{dag} \in REC$

Proof. The idea to prove that a picture p contains no cycle is to guess and locally verify a set S of edges where we turn around the direction of arrows obtaining a picture in L^-_{dag} without destroying a cycle. This set S forms a graph containing all sources in p and in all other vertices with outgoing edges in S, all ingoing edges in p are also in S. This makes sure that a cycle in p lies either completely outside or completely inside S (thus turning it around in the later case preserves it).
We have a pre-image alphabet $\Gamma \subseteq \{-1, 1\}^4 \times \{-1, 1\}^4$ and a projection $\pi : \Gamma \mapsto \{-1, 1\}^4$ with $(l, r, u, d) = \pi((l', r', u', d'), (l_S, r_S, u_S, d_S))$, $l_S \cdot l' = l$, $r_S \cdot r' = r$, $u_S \cdot u' = u$ and $d_S \cdot d' = d$. Furthermore, if there is a $x \in \{l, r, u, d\}$ with $x_S = -1$ (that means the edge is in S) and $x = 1$ then for all $y \in \{l, r, u, d\}$ with $y = -1$ it holds $y_S = -1$. Let Δ be the set of tiles in $\Gamma^{2,2}$ for which the projection to the first component is in Δ^-_{dag} and the projections to the second component is in $\Delta'_{con} :=$

$$\{\begin{array}{|c|c|}\hline \gamma & \gamma' \\ \hline \delta & \delta' \\ \hline \end{array} \in (\{\#\} \cup \Gamma)^{2,2} \mid (\gamma = (l, r, u, d) \wedge \gamma' = (l', r', u', d')) \rightarrow r = l',$$
$$(\delta = (l, r, u, d) \wedge \delta' = (l', r', u', d')) \rightarrow r = l',$$
$$(\gamma = (l, r, u, d) \wedge \delta = (l', r', u', d')) \rightarrow d = u',$$
$$(\gamma' = (l, r, u, d) \wedge \delta' = (l', r', u', d')) \rightarrow d = u'\}.$$

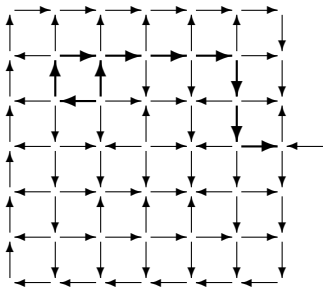
Thus, if $p = \pi(p'')$ has a cycle and p'' has the projections $p' \in L^-_{dag}$ and $p_S \in L'_{con}$ then either p' would have the same cycle which is not possible in L^-_{dag} according to Theorem 6 or w.l.o.g. $x = -x_S = 1 = -x'$ for some x on the cycle thus for the ingoing $y \in \{l, r, u, d\}$ it holds $y = y_S = -1 = -y'$ and since $p'\, p_S \in L'_{con}$, $\overline{y} = \overline{y}_S = 1 = -\overline{y}'$ for the corresponding neighbor and following the cycle in this way means that p' would have the same cycle in opposite direction which

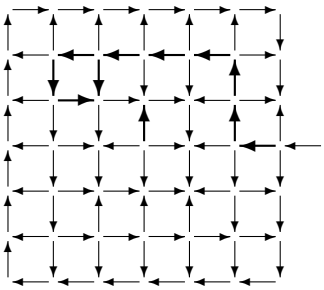
is not possible in L^-_{dag} according to Theorem 6. This means the tiling system $(\{-1,1\}^4, \Gamma, \Delta, \pi)$ generates only pictures in L'_{dag}.

On the other hand, for every $p \in L'_{dag}$ we can find a p'' having the projections $p' \in L^-_{dag}$ and $p_S \in L'_{con}$ by constructing S as follows: Initialize $S := \emptyset$ (that means $p' := p$) and iterate the procedure

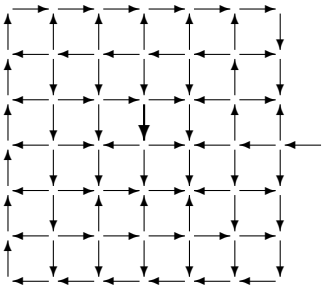
> Initialize $S' := \emptyset$ then inductively look for vertices (i.e. sources) having all ingoing edges in S' and put all their outgoing edges in S' as well. Let B be the set of vertices having an ingoing edge in S' and another ingoing edge not in S'. Let v be a vertex in B such that v can not be reached by a path from any other vertex in B. (The existence of v follows from p' not having a cycle.) Add inductively all edges in S' going to v or to the origin of an edge in S to S.

until p' contains no sources anymore. Since v can not be reached from any source in p' (only from the border of the picture), no cycle is created. □

Example: Consider the grid graph  The first execution of the procedure puts all inner edges into S' (turning them all around would cause the sink to be a new source), all surrounding vertices are in B, the middle vertex at the right side is the only possible v and 10 edges are added to S and actually turned around:



This would only turn around one of the two sources; the second iteration of the procedure has four possible choices of v and will add one more edge to S:



Note that there are cases (for example a spiral-like graph from a source) where we cannot avoid that all four edges of a source belong to S and thus this source becomes a sink. For that reason we cannot apply the same proof directly to reduce L^+_{dag} to L_{locdag}. Observe, however, that if we continue to apply the same method another time in the opposite direction removing the generated sinks, that S' is now smaller; more precisely: one layer got peeled off. We will use this in the proof of the folllowing theorem: its proof idea is an encoding of the iteration of the idea for the previous proof in layers.

Theorem 6. $L^-_{dag}, L^+_{dag} \in REC$

Proof. We prove $L^+_{dag} \in REC$ in a similar way as in Theorem 5 by guessing and locally verifying six sets $S_1, ..., S_6$ of edges where we turn around the direction of the arrows in the odd cases obtaining a picture in L_{locdag} without destroying a cycle.
This means we use a pre-image alphabet $\Gamma \subseteq \{-1,1\}^4 \times \{0,...,6\}^4$ and a projection $\pi : \Gamma \mapsto \{-1,1\}^4$ with $(l,r,u,d) = \pi((l',r',u',d'),(l_S,r_S,u_S,d_S))$, $(-1)^{l_S} \cdot l' = l$, $(-1)^{r_S} \cdot r' = r, (-1)^{u_S} \cdot u' = u$ and $(-1)^{d_S} \cdot d' = d$. Furthermore, if there is a $x \in \{l,r,u,d\}$ with $x_S > 0$ (i.e. $x_S \in \{1,2,5\}$ resp. $x_S \in \{3,4,6\}$)and $x = 1$ then for all $y \in \{l,r,u,d\}$ with $y = -1$ it holds $y_S => 0$ (i.e. $y_S \in \{5,3,4\}$ resp. $x_S \in \{6,1,2\}$). Let Δ be the set of tiles in $\Gamma^{2,2}$ for which the projection to the first component is in Δ_{locdag} and the projections to the second component is in Δ'_{con} and, furthermore, we only allow tiles having either none or two of the four internal edges in $S_1 \cup S_2$ and in the later case both of them either coming from the same vertex or going to the same vertex or going parallel. Analogously we have the same condition with $S_3 \cup S_4$. Since an edge in $S_1 \cup S_2$ also occurs in the overlapping neighbor tile, these edges form a cycle on the faces (or a line from boarder to boarder). This means they divide the picture in two parts; and paths can only lead from one side to the other but no cycle can use an edge in $S_1 \cup S_2$.
Thus, if $p = \pi(p'')$ has a cycle and p'' has the projections $p' \in L_{locdag}$ and $p_S \in L'_{con}$ then either p' would have the same cycle which is not possible in L_{locdag} according to Lemma 1 or w.l.o.g. $x = 1 = -x'$ and $x_S = 5$ (resp. $x_S = 6$) for some x on the cycle thus for the ingoing $y \in \{l,r,u,d\}$ it holds $y = -1 = -y'$ and since $p'\,p_S \in L'_{con}$, $\overline{y} = 1 = -\overline{y}'$ for the corresponding neighbor and $\overline{y}_S = 5$ (resp. $\overline{y}_S = 6$) and following the cycle in this way means that p' would have the same cycle (in opposite direction in case of $x_S = 5$), which is not possible

in L_{locdag} according to Lemma 1. This means the tiling system (Δ, π) generates only pictures in L^+_{dag}.
On the other hand, for every $p \in L^+_{dag}$ we can find a p'' having the projections $p' \in L_{locdag}$ and $p_S \in L'_{con}$ by constructing $S_1, ..., S_6$ as follows: Initialize $S_1 = ... = S_6 := \emptyset$ (that means $p' := p$) and iterate the procedure

> Determine S', B and v as in the previous proof. Set $S_6 := S_6 \setminus S'$. Add all edges in S' going to v to S_1, all edges in S' going to any other vertex in B to S_2 and the remaining edges in S' to S_5. Now determine S', B and v for the inverse of p' (all edges in opposite direction). Set $S_5 := S_5 \setminus S'$. Add all edges in S' going to v to S_4, all edges in S' going to any other vertex in B to S_3 and the remaining edges in S' to S_6.

until p' contains no sources anymore. The choice of v in each step prevents creating a cycle. □

We list some corollaries of the preceding theorems. By the results of [GRST94] Th. 4 can be formulated equivalently as follows:

Corollary 1. *Acyclicity of grid graphs is expressible in existential monadic second order logic.*

Corollary 2. DREC $\subseteq$ co-REC.

Proof. We show co-DREC $\subseteq$ REC: For a given picture $p \in \Sigma^{*,*}$ we guess the result of a deterministic process like in the proof of Theorem 3 but instead of checking its success, we check whether it either leads to a contradiction or stays incomplete at some point. □

The connectivity problem CONN on pictures consists of the pictures on alphabet $\{a, b\}$ such that all occurring b's are connected. An obvious algorithm [Rei98] shows that CONN is in DREC. By Th. 3, we have:

Corollary 3. CONN $\in$ REC.

A direct proof of this result was given in [Rei98] via a "tentacle" construction. In a way the proofs of this section can be seen as a generalization of the "tentacle" construction.

5 Summary, Open Problems, and Acknowledgments

This paper studies the *deterministically recognizable picture languages* DREC and the *Sudoku deterministically recognizable picture languages* SDREC. It is shown that all DREC languages are recognizable. The proof uses as a main lemma that the picture language consisting of the grid graphs which are acyclic is recognizable. It is shown that SDREC contains 4AFA and therefore is incomparable with the recognizable picture languages.
The open problems suggested by the class diagram shown at the end of the Introduction are the following:

– is DREC incomparable with UREC, and with 4NFA?
– is DREC or even SDREC in 4AFA?

Moreover, it is an interesting question whether there is some "computational model" on pictures which captures exactly REC ∩ co-REC. No such model is known to the authors, it would have to include DREC and 4NFA at the same time.

The concept of Sudoku determinism was inspired by discussions about determinism concepts for picture languages with Marcella Anselmo, Dora Giammarresi, Maria Madonia, Ina Mäurer, and Jarkko Kari at the ESF Meeting 2006 about picture languages in Salerno.

References

[AGMR06] M. Anselmo, D. Giammarresi, M. Madonia, and A. Restivo. Unambiguous recognizable two-dimensional languages. RAIRO – Inf. Theor. Appl. 40, pp. 277–293, 2006.

[AGMR06a] M. Anselmo, D. Giammarresi, M. Madonia, and A. Restivo. *Presentation at ESF Science Meeting, Salerno*, 2006.

[BH67] M. Blum, C. Hewitt. Automata on a 2-dimensional tape. FOCS 1967, pages 155-160, 1967

[Bor03] B. Borchert. Formal Language characterizations of P, NP, and PSPACE, submitted, 2003

[Chl84] B. Chlebus. From domino tilings to a new model of computation. In Skowkron, editor, *Computation Theory*, LNCS 208, 1984

[GR92] D. Giammarresi and A. Restivo. Recognizable picture languages. *Int. Journal Pattern Recognition and Artificial Intelligence*, 6:241–256, 1992.

[GR97] D. Giammarresi and A. Restivo. Two-dimensional languages. In Handbook of Formal Languages G. Rosenberg and A. Salomaa Eds. Vol. III, pag. 215–268. Springer Verlag, 1997

[GRST94] D. Giammarresi, A. Restivo, S. Seibert, and W. Thomas. Monadic second-order logic over pictures and recognizability by tiling systems. In P. Enjalbert, E.W. Mayr, and K.W. Wagner, editors, *Proceedings of the 11th STACS 94*, LNCS 775, pages 365–375, 1994. Springer-Verlag.

[KM01] J. Kari, C. Moore. New results on alternating and non-deterministic two-dimensional finite-state automata. In Afonso Ferreira and Horst Reichel, editors, *STACS*, LNCS 2010, pages 396–406. Springer, 2001.

[LS97a] M. Latteux and D. Simplot. Context-sensitive string languages and recognizable picture languages. *Information and Computation*, 138(2):160–169, 1 November 1997.

[LS97b] M. Latteux and D. Simplot. Recognizable picture languages and domino tiling. *Theoretical Computer Science*, 178(1-2):275–283, 1997.

[Rei98] K. Reinhardt. On some recognizable picture-languages. In L. Brim, editor, *Proceedings of the 23th Conference on Mathematical Foundations of Computer Science*, LNCS 1450, pages 760–770. Springer-Verlag, August 1998.