On the Computation Power of Finite Automata in Two-dimensional Environments

Oleksiy Kurganskyy[1]* and Igor Potapov[2] **

[1] Institute of Applied Mathematics and Mechanics,
Ukrainian National Academy of Sciences
74 R. Luxemburg St, Donetsk, Ukraine,
kurgansk@gmx.de

[2] Department of Computer Science,
University of Liverpool, Chadwick Building,
Peach St, Liverpool L69 7ZF, U.K.,
igor@csc.liv.ac.uk

Abstract. In this paper we study the model of a finite state automaton interacting with infinite two-dimensional geometric environments. We show that the reachability problem for a finite state automaton interacting with a quadrant of the plane extended by a power function, a polynomial function or a linear function is algorithmically undecidable, by simulating a Minsky machine. We also consider the environment defined by a parabola which impedes the direct simulation of multiplication. However we show that the model of a finite automaton interacting inside a parabola is also universal.

1 Introduction

Finite state automata arose as models of transducers of discrete information, i.e. models interacting with their environments [9]. Automata on picture languages [3, 17], automata in labyrinths [2, 10, 11], communicating automata [14], multicounter automata [15, 7], a model of a computer in the form of interaction of a control automaton [4] are examples of such an interaction.

The fundamental problem for systems where an automaton interacts with (possibly infinite) environment is the reachability problem: "Does a global state S (state of the automaton and state of the environment) belong to the set of states reachable from an initial global state". The reachability problem has connections to many classical problems in automata theory such as diagnostic problems, distinguishability problems, searching in labyrinths, etc. One of the standard methods to show the undecidablity of the reachability problem for some model is to prove that this computational model is universal.

In this paper we consider the computational power of the reactive system, where an input/output automaton interacts with a two-dimensional geometric

* Work partially supported by RDF grant RDF/4417

** Work partially supported by The Nuffield Foundation grant NAL/00684/G.

environment. In particular we consider the reactive system that includes a finite state automaton (FSA) A and an infinite environment E where

- an automaton A is a kind of well-known 4-way finite state automaton introduced by Blum and Hewitt [1] that reads the symbol from the environment and reacts by changing its internal state and then moving up, down, left or right to a neighbouring cell of the environment E;
- an environment E is a two-dimensional language over one-letter alphabet [13, 3] which is defined as a subspace of two-dimensional integer grid bounded by a number of integer value functions.

It was shown in [12, 13] that the problem of checking indistinguishable states for two finite automata in two-dimensional environment E over a one-letter alphabet (geometric environment) is decidable if and only if the reachability problem for a finite state automaton in E is decidable. The geometric environment is called *efficient* if the reachability problem is decidable and *non-efficient* otherwise. It is known from [5, 6, 13] that the environment defined by a set of rectangles of fixed height and the environment represented by a regular or context-free expression are efficient. There are many other classes of efficient environments with or without holes that can be constructed by substitution of one cell in efficient environment by another efficient environment [13].

It was proved in [2] that the sets of input-output words generated by automata interacting with geometric environments without holes, namely, rectangles of unlimited height are, in general, context-sensitive languages. Thus the reachability problem for finite automata interacting with geometric environments is fundamentally difficult and algorithmically unsolvable, in the general case. In spite of known undecidability results it is interesting to identify new classes of non-efficient environments, i.e. where a model of finite automaton interacting with these environments is universal.

We start from the environment defined by a quadrant of the plane that corresponds to the Minsky machine model. First we consider an extension of quadrant of the plane, changing the vertical border by power, polynomial or linear functions. In particular we show that the finite automaton interacting with these environments can simulate a Minsky machine. Then we consider an environment which impedes the direct simulation of multiplication. However we show on example of an environment defined by parabola:

$$D_{n^2,n^2} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq x^2\}$$

that FSA interacting with D_{n^2,n^2} is again a universal model of computation.

The additional motivation for the investigation of automaton dynamics in geometric environments is that the different constraints on the environments can be also seen as constraints on the values of counters in multicounter automata. So we hope that the new results about n-dimensional geometric environments could lead to the new results for n-counter automata.

The long-term goal of this work is to characterize the whole class of geometric environments to obtain a better understanding of the border between decidability and undecidability for the reachability problems. While such an ambitious goal is not feasible at the moment, we instead investigate several special cases of geometric environments, that we believe is a sound and reasonable first step towards our ultimate goal.

2 Automata interacting with two-dimensional environments

In what follows we use traditional denotations $\mathbb{N}, \mathbb{Z}, \mathbb{Z}_n, \mathbb{Q}^+$ for the sets of naturals (non-negative integers), integers, bounded integers such that $\{i | i \in \mathbb{Z}, |i| \leq n\}$ and the set of positive rational numbers respectively.

Definition 1. *Let $A = (S, I, O, \delta_A, \lambda_A, s_0)$ be a finite deterministic everywhere defined Mealy automaton, where S, I and O are the sets of states, input symbols, and output symbols, respectively, and $\delta_A : S \times I \to S$ and $\lambda_A : S \times I \to O$ are transition function and function of outputs respectively and $s_0 \in S$ is an initial state.*

Definition 2. *The geometric environment is defined by possibly infinite (countable) Moore automaton $E = (D, O, I, \delta_E, \lambda_E)$, where $D \subseteq \mathbb{Z} \times \mathbb{Z}$ is a set of states, $O = \mathbb{Z}_1 \times \mathbb{Z}_1$ is a set of input symbols, $I = 2^{\mathbb{Z}_1 \times \mathbb{Z}_1}$ is a set of output symbols, $\delta_E : D \times O \to D$ such that*

$$\delta_E((x, y), (d_1, d_2)) = \begin{cases} (x + d_1, y + d_2), & (x + d_1, y + d_2) \in D \\ \quad undefined, & (x + d_1, y + d_2) \notin D \end{cases}$$

is the partial transition function and $\lambda_E : D \to I$ such that

$$\lambda_E(x, y) = \{(d_1, d_2) \in \mathbb{Z}_1 \times \mathbb{Z}_1 | (x + d_1, y + d_2) \in D\}$$

is the function of outputs.

We call the set of states D - the nodes of the environment, the symbols of the output alphabet I - the labels of nodes, and the function of outputs λ_E - the function of labels of nodes. In such case we can say that two-dimensional geometric environment is defined by an automaton E.

Two nodes (x_1, y_1) and (x_2, y_2) of the same environment are neighbours iff $(x_2 - x_1, y_2 - y_1) \in \mathbb{Z}_1 \times \mathbb{Z}_1$. All neighbours of the node (x, y) form the Moore neighbourhood of range 1.

We can also consider finite automaton in two-dimensional environment as a kind of 4-way finite state automaton introduced by Blum and Hewitt [1] that reads the symbol (Moore neighbourhood of range 1) from the environment and reacts by changing its internal state and then moving up, down, left or right to a neighbouring cell of an environment E.

Let an automaton A and environment E interact with each other then an output signal of each of them coincides with an input signal of the other at each instant of time. So in case of 4-way finite state automaton the set of output symbols of an automaton A (set of input symbols of an environment E) is

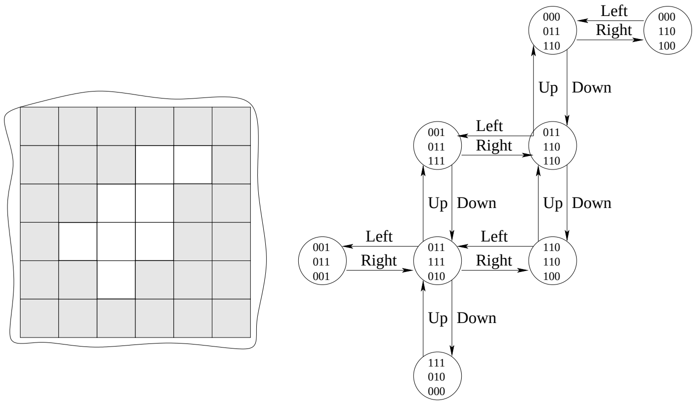

Fig. 1. An example of a geometric environment represented by Moore automaton.

$\{Left, Right, Up, Down\}$ and the input symbols of A (the output symbols of E) form a matrix $O^{3\times3}$ that represents the Moore neighbourhood of range 1 for a position (x, y):

$$\begin{pmatrix} o_{-1,1} & o_{0,1} & o_{1,1} \\ o_{-1,0} & o_{0,0} & o_{1,0} \\ o_{-1,-1} & o_{0,-1} & o_{1,-1} \end{pmatrix},$$

where $o_{i,j} = 1$ if $(x+i, y+j) \in D$, otherwise $o_{i,j} = 0$ and $i, j \in \mathbb{Z}_1$ (see Figure 1).

Let us describe the process of interaction between automaton and environment. An automaton A initiates the interaction with an environment E starting from an initial state s_0 and a node $r \in D$. Let A be in a state s and a node r, then automaton moves to the state $\delta_A(s, \lambda_E(r))$ and the node $\delta_E(r, \lambda_A(s, \lambda_E(r)))$.

The *configuration* of an automaton in an environment we denote by pair (s, r), where s is state of an automaton, and r is a node of the environment We also say that configuration (s', r') is directly reachable from (s, r) if $s' = \delta_A(s, \lambda_E(r))$ and $r' = \delta_E(r, \lambda_A(s, \lambda_E(r)))$ and we denote it by $(s, r) \to (s', r')$. We define $\to$ as a binary relation over the set configurations.

The *trajectory* of an automaton A in an environment E is a sequence of reachable configurations. By $(s, r)..(s', r')$ we denote a trajectory of an automaton that reach configuration (s', r') from (s, r).

3 From Minsky machine to more exotic models of computation

In this section we start from geometric interpretation of a well-known model of two-counter Minsky machine that can increment and decrement counters by one and test them for zero. It is known that Minsky machine is a universal model of computations and it is equivalent to Turing machine [16].

It is easy to see that the behaviour of Minsky machine can be interpreted as a 4-way finite state automaton that interacts with (or moves in) the quadrant of the plane $D_Q = \{(x,y)|x \in \mathbb{N}, y \in \mathbb{N}\}$, where the border of a quadrant is the following set of nodes: $G_Q = \{(x,y)|x \geq 0, y = 0, x \in \mathbb{N}, y \in \mathbb{N}\} \cup \{(x,y)|x = 0, y \geq 0, x \in \mathbb{N}, y \in \mathbb{N}\}$. The node (x,y) of the environment $E_Q = (D_Q, O, I, \delta_E, \lambda_E)$ represents the values x and y of two counters and the empty counters of Minsky machine corresponds to the situation when the automaton is on the borders of the geometric environment E_Q (i.e. in a cell of the environment that does not allow to move at least in one direction).

Now we can define some exotic models of computation by changing the shape of the geometric environment. Let us consider an extension of the quadrant of the plane, changing the vertical border by a power function, a polynomial function, a linear function or a sublinear function. Let $a \in \mathbb{Q}^+$ then by $D_{a \cdot n}$, D_{n^a} and D_{a^n} we denote the following environments:

$$D_{a \cdot n} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq a \cdot |x|, x < 0; y \geq 0, x \geq 0\}$$

$$D_{n^a} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq |x|^a, x < 0; y \geq 0, x \geq 0\}$$

$$D_{a^n} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq a^{|x|}, x < 0; y \geq 0, x \geq 0\}$$

In particular in the next section we show that a 4-way finite state automaton in the following three types of geometric environments $D_{a \cdot n}$, D_{n^a} and D_{a^n} where $a = 2$ can simulate a Minsky machine (see Figure 2).

It is not difficult to see that if the left border of the quadrant is extended by a sublinear function (in other words it cannot be covered by a sector in the half plane), for example:

$$D_{sub} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq log|x|, x < 0; y \geq 0, x \geq 0\}$$

then a 4-way automaton in this environment cannot simulate a universal computational model. This environment actually is not *essentially two-dimensional* environment, i.e. the dynamics in this model can be simulated by a finite state automaton interacting with one-dimensional environment.

Another interesting case is an environment which impedes the direct simulation of multiplication as it can be done in the environment D_Q, for example the environment bounded by a parabola. However we show that a finite state automaton interacting with the environment defined by parabola:

$$D_{n^2,n^2} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq x^2\}$$

is a universal model of computation.

The main aim of these results is to support our thesis that the reachability problem is undecidable for any *essentially two-dimensional* environments.

3.1 Undecidability results

In this subsection we consider several types of geometric environments with borders defined by integral functions $\lceil a \cdot x \rceil$, $\lceil x^a \rceil$ and $\lceil a^x \rceil$.

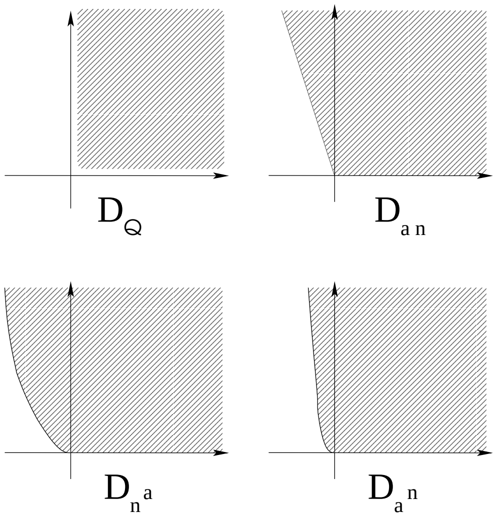

Fig. 2. Universal models of computation represented by different geometric environments.

Theorem 1. *A finite state machine in the geometric environments D_{2n}, D_{n^2}, D_{2^n} can simulate a Minsky machine.*

Proof: The proof of this fact is based on a simulation of Minsky machine by a 4-way finite state automaton in the above geometric environments.

First of all we use a well-known trick [19] to get an equivalent model of two counter machine where one of the counters is used as a scratchpad. Another, counter holds an integer whose prime factorization is $2^a \cdot 3^b$. The exponents a, b can be thought of as two virtual counters that are being simulated. If the real counter is set to zero then incremented once, that is equivalent to setting all the virtual counters to zero. If the real counter is doubled, that is equivalent to incrementing a, and if it's halved, that's equivalent to decrementing a. By a similar procedure, it can be multiplied or divided by 3, which is equivalent to incrementing or decrementing b.

Let the finite state machine A interacting with D_Q can reach the configuration (s, a', b') from configuration (s, a, b) by one step. Now we can construct another FSA A' interacting with D_Q, that can reach the configuration $(s, 2^{a'} \cdot 3^{b'}, 0)$ from configuration $(s, 2^a \cdot 2^b, 0)$ by a finite number of states.

To check if a virtual counter such as a (b) is equal to zero, just divide the real counter by 2 (3), see what the remainder is, then multiply by 2 (3) and add back the remainder. That leaves the real counter unchanged. The remainder will have been nonzero if and only if a (b) was zero.

The geometric environment D_{2n}. The straightforward modification of the FSA A' gives the result for any sector environment formed by two lines in the half plane. Since the integral line defines the regular shifts of the border, we only need to amortise these shifts by adding a fixed number of right or left moves according to the chosen direction.

The geometric environment D_{n^2}. The case where one of the borders does not have a periodic shifts is less trivial. However we use again the same scheme and prove that interaction of finite state automaton A' with the quadrant environment D_Q can be reduced in some sense to the interaction of finite state automaton B with environment D_{n^2}. In case of nonperiodic border we need to choose another method to code the counter in the new environment, and more sophisticated method of amortisation during the operations of multiplication and division.

Let the boundary point $(-x, x^2) \in G_{n^2}$ represents a number x. Now let us show that we can convert any point $(-x, x^2)$ to the point $(-2x, (2x)^2)$, that stands for multiplication of x by 2.

First, FSA converts point $(-x, x^2)$ to point $(x^2 - x, 0)$ by moving right and down until it reaches the border. Then it converts the point $(x^2 - x, 0)$ to the $(-2x, 4x^2 + 4x)$ by repeating the following pattern "moving four times up and one time left" until it reaches the border. Since the inequality $(2x)^2 \le 4x^2 + 4x \le (2x+1)^2$ holds for any natural number we can state that the point $(-2x, 4x^2 + 4x)$ belongs to the border G_{n^2}. Finally the point $(-2x, 4x^2 + 4x)$ can be converted to the point $(-2x, 4x^2)$ by a finite number of moves down along the border, since all points $\{(-2x, w) | (2x)^2 < w \le (2x+1)^2\}$ are the boundary points. Let

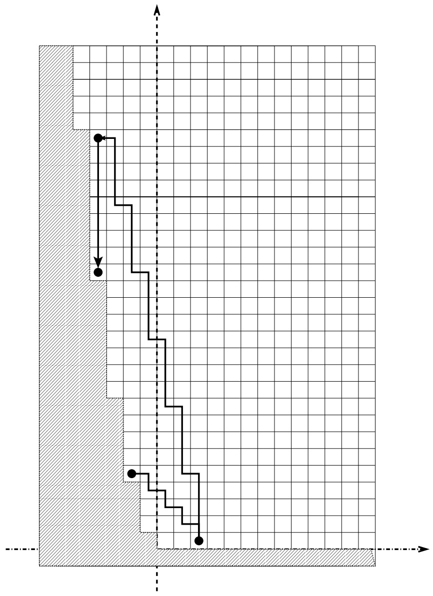

Fig. 3. Simulation of the multiplication by 2 in the geometric environment D_{n^2}.

us consider how to perform multiplication on 3 in this environment. In other words we need to show that from any point $(-x, x^2)$ we can reach the point $(-3x, (3x)^2)$. We start just as in previous case. We convert point $(-x, x^2)$ into $(x^2 - x, 0)$ and then to $(-3x-3, 9x^2+18x+27)$, which is the boundary point according to the inequality $(3x+3)^2 < 9x^2+18x+27 < (3x+4)^2$, which holds for all $x \in \mathbb{N}$. After that FSA moves from point $(-3x-3, 9x^2+18x+27)$ to point $(-3x, (3x)^2)$ along the border via 3 corners $(-3x-3, (3x+3)^2)$, $(-3x-2, (3x+2)^2)$ and $(-3x-1, (3x+1)^2)$. In a similar way we construct automaton that can divide on 2 and 3 in D_{n^2} .

The geometric environment D_{2^n}. Let us prove that we can perform the operations of multiplication and division by a finite state automaton in the geometric environment D_{2^n}.

Let the boundary point $p = (-\lfloor log_2 x \rfloor, x)$ stands for a positive number x in the geometric environment D_{2^n}. The finite automaton can move from the point $(-\lfloor log_2 x \rfloor, x)$ to the new point $(0, x - \lfloor log_2 x \rfloor)$ by repeating a pair of operations move left and down until it reaches the border. Then the finite automaton can reach the border point r by repeating two moves up and one move left. Thus, r is either $(-\lfloor log_2 x \rfloor, 2x)$ or $(-\lfloor log_2 x \rfloor - 1, 2x+2)$.

Now let us show how to check by a finite automaton in which part of E it reaches the boundary point r from the initial point p. We use the simple property that if n is an even number then $(2n \bmod 4) = 0$ and $(2n+2 \bmod 4) \neq 0$, if n is an odd number then $(2n \bmod 4) \neq 0$ and $(2n+2 \bmod 4) = 0$. We first check by finite automaton if the ordinate of the point p is odd or even and then we check the ordinate of the point r on divisibility by 4. So if r's ordinate is divisible on 4 and p's ordinate is odd then the automaton is in the point $(-\lfloor log_2 x \rfloor - 1, 2x+2)$ and it can move down to the point $(-\lfloor log_2 x \rfloor - 1, 2x)$ $=(-\lfloor log_2 2x \rfloor, 2x)$, otherwise it is in the point $(-\lfloor log_x \rfloor, 2x)$ $=(-\lfloor log_2 2x \rfloor, 2x)$.

In a similar way we can show that finite automaton can multiply on 3 and divide on 2 and 3 in the geometric environment D_{2^n}.

Theorem 2. *A finite state automaton in the geometric environment*

$$D_{n^2,n^2} = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} | y \geq x^2\}$$

can simulate a Minsky machine.

Proof: Let the boundary point (x, x^2) stands for a positive number x in the geometric environment D_{n^2,n^2}. A finite automaton cannot multiply in D_{n^2,n^2} if it touches the border only by a constant number of times as we have in case of D_{n^2} or D_{2^n}. So in case of D_{n^2,n^2} we introduce some kind of cycle that will be used for multiplication and division.

Let us prove that a finite automaton will always reach a point $(2x, (2x)^2)$ from a point (x, x^2) using th following procedure:

Multiplication by 2 in D_{n^2,n^2}

Input: A point (x, x^2)
Output: A point $(2x, (2x)^2)$

```
repeat
  repeat
      Do 1 move Up and 1 move Left;
  until FSA is on the border
  repeat
      Do move Right;
  until FSA is on the border
  Do 4 moves Up;
until FSA is on the cell (1 1 1 / 1 1 0 / 1 1 0)
repeat
    Do move Down;
until FSA is on the cell (1 1 0 / 1 1 0 / 1 0 0)
```

repeat
  repeat
    Do 1 move Up and 1 move Left;
  until FSA is on the border
  repeat
    Do move Right;
  until FSA is on the border
  Do 4 moves Up;
until FSA is on the cell $\begin{pmatrix} 1\,1\,1 \\ 1\,1\,0 \\ 1\,1\,0 \end{pmatrix}$
repeat
  Do move Down;
until FSA is on the cell $\begin{pmatrix} 1\,1\,0 \\ 1\,1\,0 \\ 1\,0\,0 \end{pmatrix}$

A finite automaton can reach the boundary point $(-(x+1), (x+1)^2) = (-(x+1), x^2+2x+1)$ in D_{n^2,n^2} from a point (x, x^2) by repeating the pattern "up and left", since it will move left for exactly $2x+1$ cells. It is easy to see that an automaton will reach the point $(-(x+1), (x+1)^2+c)$ from (x, x^2+c) for any $0 \le c < 2x+1$ by repeating the same pattern "up and left". Then it can reach the point $(x+1, (x+1)^2+c)$ by moving "right" until it reaches the border and the point $(x+1, (x+1)^2+c+4)$ doing additional 4 moves "up".

Let us call the sequence of moves from the point (x, x^2+c) to the point $(x+1, (x+1)^2+c+4)$ a *cycle*. In the above procedure of multiplication by 2, starting from the point (x, x^2) the finite automaton will meet the cell

$$\begin{pmatrix} 1\,1\,1 \\ 1\,1\,0 \\ 1\,1\,0 \end{pmatrix}$$

for the first time exactly after x cycles since it corresponds to the point $(2x, 4x^2+4x)$. If the automaton will move down from the cell

$$\begin{pmatrix} 1\,1\,1 \\ 1\,1\,0 \\ 1\,1\,0 \end{pmatrix} \text{ to } \begin{pmatrix} 1\,1\,0 \\ 1\,1\,0 \\ 1\,0\,0 \end{pmatrix}$$

it reaches the point $(2x, 4x^2)$ and the multiplication by 2 is completed (see Figure 4). Similarly we can perform multiplication by 3 if we slightly change the previous procedure. If we change our cycle in a such way that the automaton will do 3 additional moves up instead of 4 it reaches the cell

$$\begin{pmatrix} 1\,1\,1 \\ 1\,1\,0 \\ 1\,1\,0 \end{pmatrix}$$

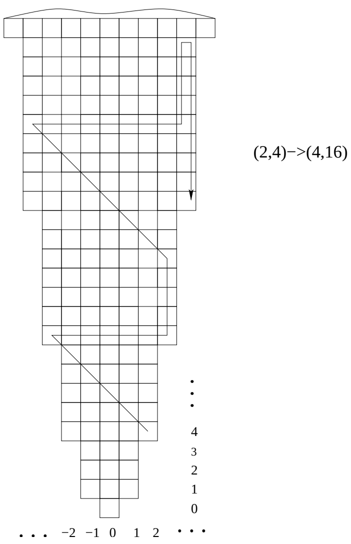

Fig. 4. An example of multiplication of $x = 2$ by 2 in the geometric environment D_{n^2,n^2}. The operation of multiplication is simulated by the movement of a finite automaton from (x, x^2) to $(2x, (2x)^2)$.

for the first time after $2x$ cycles and will be in the position $(3x, 9x^2 + 6x)$. After that it moves down in the same way as in previous case to reach the point $(3x, (3x)^2)$. Another pair of operation such that division by 2 and 3 an automaton can perform in the following way. The automaton moves from point (x, x^2) to the point $(-x, x^2)$ and back. In such way it perform $4x$ steps or moves. In order to check the divisibility of x by 2 (or 3) it needs to check the divisibility of $4x$ by 8 (or 12) that can be done using a finite memory.

4 Future work and conclusion

As we mentioned before the long-term goal of this work is to characterize the class of non-efficient geometric environments. It follows from the developed methods of simulating a Minsky machine in the considered geometric environments that there are several factors that have an influence on the status of reachability problem: border markers such as corners in the environments, the continued growth expansion of the environment, etc. These factors create essentially two-dimensional environment that cannot be modelled by one-dimensional systems, like push-down or one-counter automata. So it could be interesting to define more formal condition for essentially two-dimensional systems. On the other hand we still have unanswered questions about the reachability problem for a number of specific environments. One of these cases is the environment in $\mathbb{N}^2$ bounded by logarithmic function:

$$D_{log} = \{(x, y) \in \mathbb{N} \times \mathbb{N} | y \leq log(x)\}.$$

References

1. Manuel Blum, Carl Hewitt: Automata on a 2-Dimensional Tape. FOCS (1967) 155-160
2. V.I. Grunskaya. Some properties of trajectories of automata in labyrinths, Thesis [in Russian]. Moscow State University, Moscow (1994)
3. Dora Giammarresi: Finite State Recognizability for Two-Dimensional Languages: A Brief Survey. Developments in Language Theory (1995) 299-308
4. V.M. Glushkov, G.E. Tseitlin and E.L.Yushchenko. Algebra, Languages and Programming [in Russian]. Naukova Dumka, Kiev (1989)
5. Grunsky I.S., Kurganskyy A.N. Indistinguishability of finite automata interacting with an environment [in Russian]. Dokl. AN Ukraine, v11 (1993), 31-33
6. Grunsky I.S., Kurganskyy A.N. Indistinguishability of finite automata with constrained behaviour, Cybernetics and systems analysis,v5, (1996) 58-72
7. M. Jantzen and O. Kurganskyy. Refining the hierarchy of blind multicounter languages and twist-closed trios. *Information and Computation*, 185 (2003) 159-181.
8. Katsushi Inoue, Itsuo Takanami. A survey of two-dimensional automata theory. Inf. Sci. 55(1-3) (1991) 99-121
9. V.B. Kudryavtsev, S.V. Alyoshin and A.S.Podkolzin. An introduction to the Theory of Automata [in Russian], Nauka, Moscow, (1985)
10. V. B. Kudryavtsev, Sh. Ushchumlich, and G. Kilibarda. On the behavior of automata in labyrinths. Discrete Math. and Applications, 3 (1993) 1-28
11. G. Kilibarda, V. B. Kudryavtsev and Sh. Ushchumlich. Collectives of automata in labyrinths Discrete Mathematics and Applications 13(5) (2003) 429-466
12. Kurganskyy A.N. Indistinguishability of the finite automata interacting with an environment. Thesis [in Russian]. Saratov State University, Russia (1997)
13. Kurganskij A.N. Indistinguishability of finite-state automata with respect to some environments. (Russian, English) [J] 37, No.1, (2001) 33-41 ; translation from Kibern. Sist. Anal. 2001, No.1, (2001) 43-55
14. A. Kurganskyy, I. Potapov. On the bound of algorithmic resolvability of correctness problems of automaton interaction through communication channels. *Cybernetics and System Analysis*, 3 (1999) 49-57
15. Maurice Mergenstern. Frontier between decidability and undecidability: a survey. Theoretical Computer Science, v.231 (2000) 217-251.
16. M. Minsky. Computation: Finite and Infinite Machines. Englewood Cliffs, N.J.: Prentice-Hall (1967)
17. C.Moore and J. Kari. New Results on Alternating and Non-Deterministic Two-Dimensional Finite-State Automata. International Symposium on Theoretical Aspects of Computer Science, STACS (2001) 396-406
18. Tokio Okazaki, Katsushi Inoue, Akira Ito, Yue Wang. Space Hierarchies of Two-Dimensional Alternating Turing Machines, Pushdown Automata and Counter Automata. IJPRAI 13(4) (1999) 503-521
19. Wikipedia, the free encyclopedia - Counter. http://en.wikipedia.org/wiki/Counter