Journal of Universal Computer Science, vol. 6, no. 9 (2000), 881-905
submitted: 6/1/00, accepted: 4/8/00, appeared: 28/9/00 *© Springer Pub. Co.*

Nonlinear Computation with Switching Map Systems

Yuzuru Sato
(Institute of Physics, Graduate School of Arts and Sciences, University of Tokyo
ysato@sacral.c.u-tokyo.ac.jp)

Takashi Ikegami
(Institute of Physics, Graduate School of Arts and Science, University of Tokyo
ikeg@sacral.c.u-tokyo.ac.jp)

Abstract: A dynamical systems based model of computation is studied. We demonstrate the computational capability of a class of dynamical systems called switching map systems. There exists a switching map system with two types of baker's map to emulate any Turing machines. The baker's maps are corresponding to the elementary operations of Turing machines such as left/right head-moving and read/write symbols. A connection between the generalized shifts by C. Moore [Moore 91] and the input-output mappings by L. Blum et al. [Blum, Cucker, Shub and Smale 98] is shown with our model. We present four concrete examples of switching map systems corresponding to the Chomsky hierarchy. Taking non-hyperbolic mappings as elementary operations, it is expected that the switching map systems shows a new model of computation with nonlinearity as an oracle.

Key Words: switching map systems, Smale's horseshoe, baker's map, Hénon map
Category: F.1

1 Introduction

Dynamical systems have recently been actively studied in the view of a computing process in physical systems (see e.g., [Crutchfield and Young 90] [Moore 91] [Siegelmann 99]). In [Moore 91], it is shown that any Turing machine is equivalent to a class of two-dimensional piecewise-linear map (Generalized shifts, GSs) and they can be embedded in smooth flows in $\mathbf{R}^3$. Here we introduce another class of dynamical systems called switching map systems to demonstrate its computational capability.

We show that there exists a switching map system with two types of baker's map to emulate any Turing machines. The baker's maps are corresponding to the elementary operations of Turing machines such as left/right head-moving and read/write symbols. An advantage of this system is that it is a 'programmable' GSs. Thus we can easily construct various concrete examples with it. Since this system is also a class of input-output mappings defined in the BSS model [Blum, Cucker, Shub and Smale 98], a connection between GSs and input-output map is also shown.

We also put forward the importance of 'natural' effective procedures" in computation. Taking non-hyperbolic mappings (for example the Hénon map), as elementary operations, we argue that the switching map system shows a new model of computation with the nonlinearity as an oracle. [1]

[1] Here we call it oracle here in the sense that it can potentially use infinite bits at one time step. See [Sato].

In section 2, we introduce switching map systems and study the correspondence between dynamical systems and Turing machines. In section 3, we show examples of switching map systems which can be recognizers for formal languages of the Chomsky hierarchy. In section 4, we propose a framework of the nonlinear computation by using Hénon maps as elementary operations of computation there. We also give an example of nonlinear computation. In section 5, we summarize the results and the overview. In particular, probabilistic computation and non-deterministic computation are discussed. Continuous flow that sustains switching map system is also presented.

2 Turing machines and switching map systems

Here we consider the correspondence between the switching map systems and Turing machines. We introduce a symbolic dynamics to interpret dynamical systems as Turing machines.

2.1 The switching map systems

Let $f_1, f_2, \ldots, f_M$ be M mappings on $X \subset \mathbf{R}^n$. A set of internal states $S = \{0, \ldots, N-1\}$ and N branching functions labeled by the states $g_0, g_1, \ldots, g_{N-1}$ are given in advance, where $g_n : X \to S$ are mappings from a value of x to a label of the next states. We denote a switching map system with $f_1, f_2, \ldots, f_M$ as $F(f_1, f_2, \ldots, f_M)$:

$$F : S \times X \to S \times X : \ (n, x) \mapsto (n', x') = (g_n(x), h_n(x)) \ \ (n \in S),$$

where $h_n \in \{f_1, f_2, \ldots, f_M\}$. F maps each state/space pair (n, x) to the unique next state/space pair (n', x') deterministically. Here in practice, mappings are switched and applied to an initial configuration $(0, x_0) \in S \times X$ successively as follows:

$$\begin{aligned}(0, x_0) &\mapsto (n_1, x_1) = (g_0(x_0), h_0(x_0)) \\ &\mapsto (n_2, x_2) = (g_{n_1}(x_1), h_{n_1}(x_1)) \\ &\ \ \vdots \\ &\mapsto (n_{t+1}, x_{t+1}) = (g_{n_t}(x_t), h_{n_t}(x_t))\end{aligned}$$

Here g_{n_t} is applied to x_t and $g_{n_t}(x_t)$ is used as function label n_{t+1}, in the same way, h_{n_t} is applied to x_t, and $h_{n_t}(x_t)$ is substituted as initial value x_{t+1} for $h_{n_{t+1}}$, where $n_1, n_2, \ldots$ is the state of the system at time $t = 1, 2, \ldots$ respectively. This process is iteratively applied to determine the successive functional form.

We can regard a switching map system $F(f_1, f_2, \ldots, f_M)$ with suitable branching function g's as a program, and its trajectories as a process of computation. Structures of its attractors are corresponding to the output results. Since we can use periodic orbits or attractors as output results, a halting state is not always required. With this framework, we can define a process of computation as a dynamics and interpret properties of dynamical systems such as measures, dimensions and topological structures from a computational theoretical point of view. Additionally, a switching map system is represented by a skew product of dynamical systems [Cornfeld, Fomin, Sinai 82] and it is also introduced in a view from information processing on high-dimensional dynamical systems [Tsuda 91].

2.2 The horseshoe map and symbolic dynamics

A dynamical system is called chaotic if the set of the orbit in the phase space embeds Smale's topological horse shoe [Smale 67]. The horse shoe map is described in terms of an invertible planar map which can be thought of as a Poincaré map arising from a three dimensional autonomous differential equation or a forced oscillator (Figure 1). The set of non-wandering points of horse shoe map Λ forms a square Cantor set, in which each point corresponds to a sequence in $\{0,1\}^{\mathbf{N}}$. Which half (left or right) of the square the point is belonging to gives a rule of the gray code (Figure 2). The action of the map induces a shift on a bi-infinite sequence of two symbols where $\overline{a_i}$ denotes a bit-flipping:

$$H_0 : \Lambda \to \Lambda : \ \ldots a_{-2}a_{-1}.a_0a_1a_2\ldots \mapsto \ldots a_{-2}a_{-1}a_0.a_1a_2\ldots$$

$$H_1 : \Lambda \to \Lambda : \ \ldots a_{-2}a_{-1}.a_0a_1a_2\ldots \mapsto \ldots a_{-2}a_{-1}\overline{a_0}.a_1a_2\ldots$$

$$H_0^{-1} : \Lambda \to \Lambda : \ \ldots a_{-2}a_{-1}.a_0a_1a_2\ldots \mapsto \ldots a_{-2}.a_{-1}a_0a_1a_2\ldots$$

Regarding the decimal point as the position of the head of Turing machines,

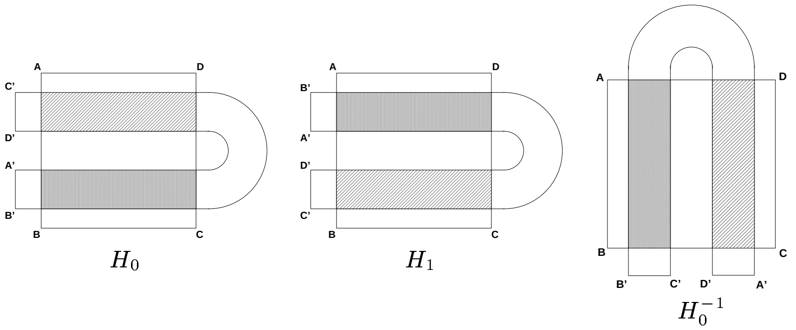


Figure 1: The Smale's horseshoe map

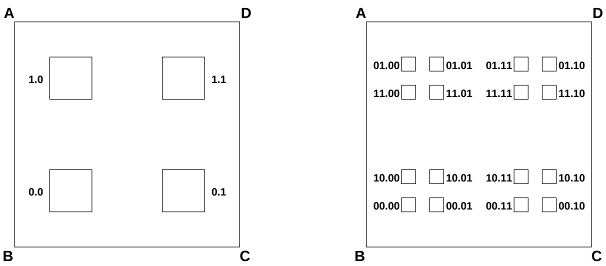


Figure 2: Coding non-wandering points on the invariant set Λ

these actions correspond to the elementary operations of Turing machines, such as (1) left-shifting ($H_0 = H_1 H_0^{-1} H_1$), (2) right-shifting (H_0^{-1}), (3) bit flipping and left-shifting (H_1), and (4) bit-flipping and right-shifting ($H_0^{-1} H_1 H_0^{-1}$). Thus we can use suitable combinations of H_0^{-1} and H_1 to emulate arbitrary computation processes. When a halting state is required, we can use identical maps I as the halting operations.

In two dimensional discrete dynamical systems on a unit space $[0,1]\times[0,1]$, we can substitute the baker's map for the horseshoe map. It is a measure preserving map which conjugates to the shift on two symbols. In this case, the left and right halves of a point's sequence are simply the binary expansions of its x and y coordinates respectively. Here we use a standard metric on the unit space. In practice, Turing machines with N internal states can be emulated by the following set of baker's mappings:

$$B_0(x,y) = \begin{cases} (2x, \frac{y}{2}) & x \in [0, \frac{1}{2}) \\ (2x-1, \frac{y+1}{2}) & x \in [\frac{1}{2}, 1) \end{cases} \tag{1}$$

$$B_1(x,y) = \begin{cases} (2x, \frac{y+1}{2}) & x \in [0, \frac{1}{2}) \\ (2x-1, \frac{y}{2}) & x \in [\frac{1}{2}, 1) \end{cases} \tag{2}$$

$$B_0^{-1}(x,y) = \begin{cases} (\frac{x}{2}, 2y) & y \in [0, \frac{1}{2}) \\ (\frac{x+1}{2}, 2y-1) & y \in [\frac{1}{2}, 1) \end{cases} \tag{3}$$

Theorem 1. *For any Turing machine M, there exists a switching map systems $F(B_0^{-1}, B_1, I)$ conjugate to M via a map Φ.*

Proof. This is done by constructing a map Φ from a Turing machine M with two symbols to a switching map system $F(B_0^{-1}, B_1, I)$.

Turing machines M with $N+1$ internal state are defined by a finite set of $(q, s_r, s_w, \gamma, q')$, where $q, q' \in \{0, \ldots, N\}$ represent the present/next internal state, $0/N$ correspond to machine's initial/halting state, $s_r, s_w \in \{0,1\}$ correspond to symbols which a machine reads/writes, and $\gamma \in \{L, R\}$ correspond to left/right head-moving.

Let symbols on tape be $(a_{-i}).a_0(a_i)$ $(i \in \mathbf{N}, a_i \in \{0,1\})$. We represent them by $x = 0.a_0(a_i) = \Sigma_{i=0}^{\infty} a_i 2^{-i} \in \mathbf{R}$, and $y = 0.(a_{-i}) = \Sigma_{i=1}^{\infty} a_{-i} 2^{-i} \in \mathbf{R}$. A branching function g_n is denoted as follows;

$$g_n(x) = \begin{cases} n_0 \; x \in [0, \frac{1}{2}) \; (a_0 = 0) \\ n_1 \; x \in [\frac{1}{2}, 1) \; (a_0 = 1) \end{cases} \; (n, n_0, n_1 \in \{0 \ldots N\}). \tag{4}$$

Φ is so constructed that it can map from any state transitions of Turing machine M to corresponding switching maps as follows;

$$\begin{aligned}
(n,0,0,R,m) &: (n,(x,y)) \mapsto (m, B_1 B_0^{-1} B_1(x,y)) \\
(n,0,1,R,m) &: (n,(x,y)) \mapsto (m, B_1(x,y)) \\
(n,1,0,R,m) &: (n,(x,y)) \mapsto (m, B_1(x,y)) \\
(n,1,1,R,m) &: (n,(x,y)) \mapsto (m, B_1 B_0^{-1} B_1(x,y)) \\
(n,0,0,L,m) &: (n,(x,y)) \mapsto (m, B_0^{-1}(x,y)) \\
(n,0,1,L,m) &: (n,(x,y)) \mapsto (m, B_0^{-1} B_0^{-1} B_1(x,y)) \\
(n,1,0,L,m) &: (n,(x,y)) \mapsto (m, B_0^{-1} B_0^{-1} B_1(x,y)) \\
(n,1,1,L,m) &: (n,(x,y)) \mapsto (m, B_0^{-1}(x,y))
\end{aligned}$$

A corresponding initial state is given by $(0, B_0^{-1}(x,y))$ or $(0, B_1(x,y))$, and the halting state is by $(N, I(x,y))$. Then we have a conjugate relationship via a map Φ which satisfies $F(B_0^{-1}, B_1, I) = \Phi^{-1} M \Phi$. □

$$\begin{array}{ccccc} & \Lambda & \overset{F}{\longrightarrow} & \Lambda & \\ \Phi^{-1} & \downarrow & & \uparrow & \Phi \\ & \{0,1\}^{\mathbf{N}} & \underset{M}{\longrightarrow} & \{0,1\}^{\mathbf{N}} & \end{array}$$

In the next section, we give examples of computation in case that the baker's maps are used as elementary operations. If their space is defined on $\mathbf{R}$, it can be a model of real number computation (See e.g. [Blum, Shub and Smale 90] and [Pour-El and Richards 89]). If the space is restricted to the set of the dyadic rationals with finite binary expansions, it is regarded as a model of classical computation.

3 Examples

We show examples of switching map systems which can be recognizers for formal languages of the Chomsky hierarchy. In order to compare them to classical automata, we use B_0^{-1}, B_1, and I as mappings on $\mathbf{Q}$. Symbols on tape are also embedded onto the dyadic rationals with finite binary expansions in $[0,1] \times [0,1]$. For detailed computational processes, see the examples in the appendix.

3.1 Parity check

Parity check is a problem to decide a parity of the number of '1' in input binary sequences. The accepted language belongs to the class of regular language (RL) and is accepted by a finite state automaton (FSA) (Figure 3). For input sequences $x = .(0{+}1)^*00^*$, this dynamical system converges to (0,0) if the number of '1' in binary expansion of input x is an even number, and converges to (0,1) otherwise. Since FSA is equivalent to the one-way head-moving Turing Machines, the class of $F(B_0, B_1)$ should be equivalent to the class of regular grammar. $F(B_0, B_1)$ is a class of hyperbolic dynamical systems, so that it is easy to show that they have a precise Markov partition, their Lyapunov exponents as $\pm \log 2$, and their topological entropies as $\log 2$.

3.2 Parenthesis check

Parenthesis check is a problem to decide a consistency of the number of left and right parenthesis in input binary sequences by regarding '0' as '(' and 1 as ')'. The accepted language, called Dyck language, belongs to the class of context free language (CFL) and is accepted by a push-down automata (PDA) (Figure 4). PDA can be constructed with $F(B_0^{-1}, B_1)$. We set input sequences and an endmarker on the right side of the initial tape and use the left side as a stack. For input sequences $x = .(0{+}1)^*00^*$ and $y = .100^*$, this dynamical system converges to $(0,0)$ if the sequence of '0' and '1' in input x are inconsistent in the above sense, and to $(0,1)$ otherwise.

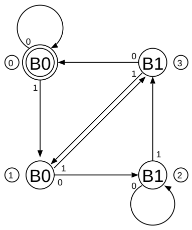


Figure 3: Parity checker: A double circle indicates the initial node. Which half ($x \in [0, \frac{1}{2})$ or $x \in [\frac{1}{2}, 1]$) of the unit space the point is in is denoted on arrows, and next mappings B_0 or B_1 is chosen by it. While arrows with no symbols denote that the map is selected regardless of x. Small numbered circles are the labels of the internal states.

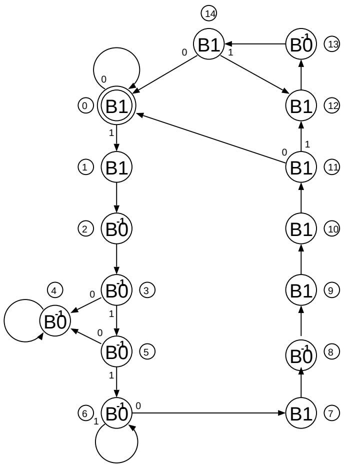
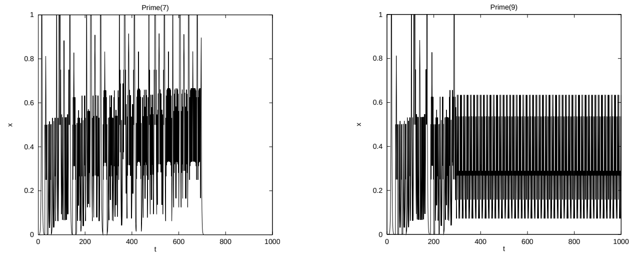


Figure 4: Dyck language recognizer: See the notation in Figure 3.

3.3 Primality check

Primality check is a problem to decide a primality of the number of '1' in input binary sequences. The accepted language belongs to the class of context sensitive language (CSL) and is accepted by a linear bounded automata (LBA) (Figure 6). LBA can be constructed with $F(B_0^{-1}, B_1)$. We set a left endmarker on the left side input sequences and a right endmarker on the right side of the initial tape.

For input sequences $x = .0^{2n}11(0{+}1)^*$ and $y = .11(0{+}1)^*$, this dynamical system converges to $(0,0)$ if n is a prime number, and converges to a $4p+2$-periodic orbit otherwise (Figure 5). Here p is a minimal prime factor of n. We use here the Eratosthenes' sieve as the primarily check algorithm. If the minimal prime factor p is detected, it is trapped to a $4p+2$-periodic orbit.



$Prime(7)$ $Prime(9)$

Figure 5: Computational process of primality checker: The left figure shows the time series of x in case that the input sequence is 7 ($11.0^{14}11$). The dynamics converges to a fixed point (0,0). The right figure shows the time series of x in case that the input sequence is 9 ($11.0^{18}11$) The dynamics converges to an unstable periodic orbit with period 14. Since the prime numbers p exist infinitely, this dynamical system has countably infinite unstable orbits with period $4p+2$.

3.4 Universal Turing machine

A universal Turing machine (UTM) is constructed with $F(B_0^{-1}, B_1)$. Figure 7 shows a UTM found by M. Minsky [Minsky 67]. The accepted language belongs to the class of universal language (UL) (or recursively enumerable set) generated by an unrestricted grammar. This dynamical system can 'emulate' arbitrary two-dimensional discrete dynamical systems consist of computable functions[2].

Each dynamical systems described by switching map systems are embedded as an initial value. Any non-trivial question about this dynamical system, such as sensitive dependency, measures of the basins of attraction, and dimensions of attractors, are undecidable due to Rice's theorem [Moore 91].

[2] Obviously its dynamical structure is different far from the emulated one's, but we can 'trace' the orbits of the emulated dynamical systems with an adequate sampling for the time series.

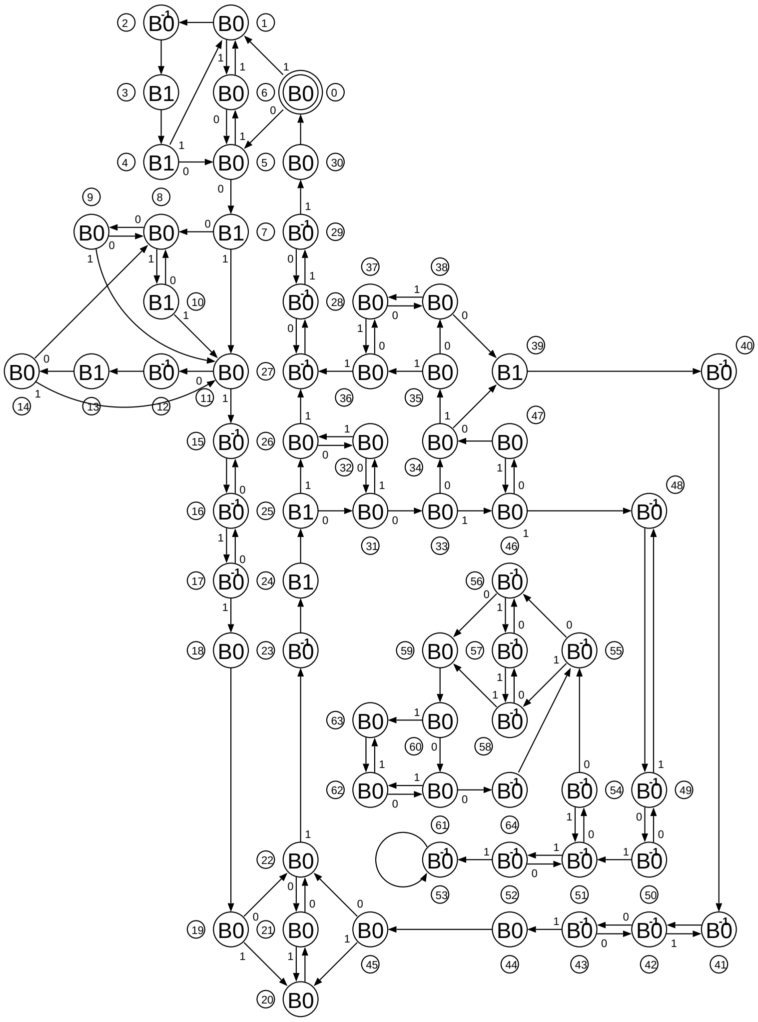


Figure 6: Primality checker: See the notation in Figure 3.

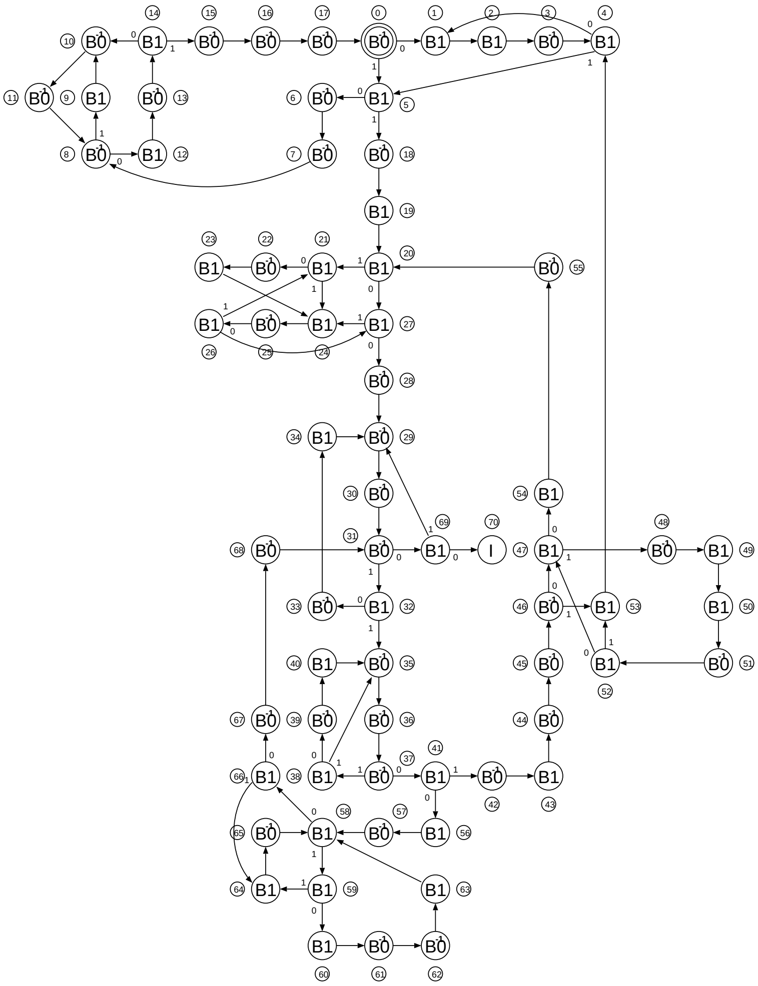


Figure 7: Minsky's universal Turing machine: See the notation in Figure 3.

4 Nonlinear computation

In the above we have adopted the baker's map for emulating Turing machines. However we can take the other types of nonlinear mappings as 'natural' elementary operation from the dynamical systems point of view. We suggest that they can be also basic components of effective procedures. Taking non-hyperbolic mappings with second-order nonlinearity instead of the baker's maps, the present model can show a new model of computation with the nonlinearity as an oracle.

4.1 Hénon map as nonlinear operations

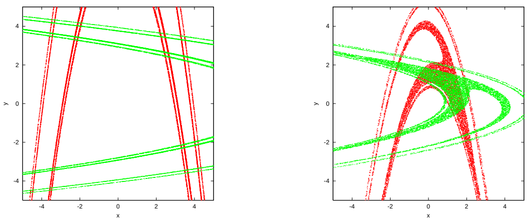
The Hénon map T_0 is given by $T_0(x, y) = (a - x^2 + by, x)$ where $|b| \leq 1$ and a is given as control parameters. It conjugates to Smale's horseshoe and its invariant set is hyperbolic if it has a strong nonlinearity. In practice, the range $a > (5 + 2\sqrt{5})(1 + |b|)^2/4$ corresponds to the hyperbolic case.

We here introduce adjoint Hénon map T_1 corresponding to H_1 in Figure 1 and let the invariant set of T_0 and T_1 be Λ_0 and Λ_1, respectively. Turing machines can be emulated as the symbolic dynamics on the domain including $\Lambda_0 \cap \Lambda_1$ using the Hénon mappings with a large value of a.[3]

$$T_0(x, y) = (a - x^2 + by, x) \tag{5}$$

$$T_1(x, y) = (a - x^2 + by, -x) \tag{6}$$

$$T_0^{-1}(x, y) = (y, (x - a + y^2)/b) \tag{7}$$



$a = 12.0,\ b = -1.0.$ $a = 3.0,\ b = -1.0.$

Figure 8: Hénon map and its inverse map

[3] The invariant set associated with a Hénon map and that with a adjoint Hénon map can certainly have overlaps. But we have not examined whether the overlaps can embed any sequences so that we do not have to constrain the computational ability of this switching map systems with Hénon maps. This problem will be argued elsewhere. Most probable, we have to bound the computation time in accordance with the value of nonlinearity a. The lower the value, the more the time which certificates correct computation is bounded. Apparently, an infinite large value of a case coincides with the switching map systems with the Smale's horseshoe. On the other hand, when we lower the parameter to have homoclinic tangency, the computation processes defined by Turing machines instantly breaks up.

Lowering the control parameter a to the point where the first homoclinic tangency occurs, we observe that the map transits from hyperbolic to non-hyperbolic (Figure 8). This transition can easily break the conjugate relationship between the switching map systems and Turing machines. However, as we show in a simple example in the next section, it rather opens a new model of computation. That is, it is thought that this property shows a potential transition from an ordinary Turing machine to a 'nonlinear' one. We call nonlinear mappings like Hénon maps as **nonlinear operations**, algorithms with them as **nonlinear algorithms**, and computation processes with them as **nonlinear computation**.

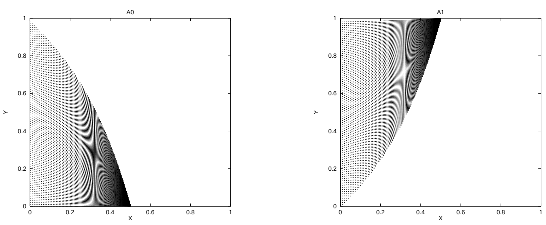
4.2 An example of nonlinear computation: Density estimation

We present here a simple but non-trivial example of nonlinear computation. In the previous section, we proposed an example of a parity checker. Here we consider a related but far difficult computational task. That is, a program which computes a density of 1s of the given input value x, which we name a "density estimator". Instead of merely answering even or odd parity, the density estimator stores the absolute number of 1s, divides it by the total bit length, and outputs the density of 1s at each time steps. This behavior is difficult to be manipulated with any finite state machines with finite memory. We here show that it is however simply manipulated by "nonlinear" switching map systems as below.

Defining p as the density of the τ bit length, we organize the density estimator by switching map systems with the following two second order nonlinear mappings A_0, A_1. Here the branching action depends on the input binary sequences (see Figure 9, 10). Setting the initial value of $F(A_0, A_1)$ to $x_0 = 1.0, y_0 = 0.0$, it estimates the density p as the values of y at each time steps. Figure 11 shows the time series of p obtained by the program when it is given a binary sequences generated by a baker's map with a finite precise initial value. The branching function g is adopted to the baker's map from equation (4) (see Figure 9, 10).

$$A_0(x, y) = (\frac{x}{1+x}, (1-x)y) \tag{8}$$

$$A_1(x, y) = (\frac{x}{1+x}, (1-x)y + x) \tag{9}$$



A_0 A_1

Figure 9: Elementary operations of density estimator

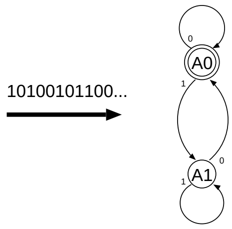
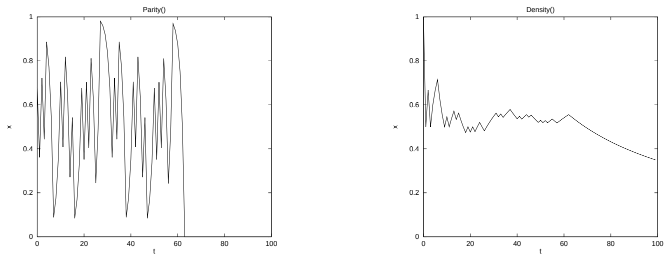


Figure 10: Density estimator



Dynamics of the baker's map   Dynamics of the density estimator

Figure 11: Computational process of density estimator: The left figure shows the time series of x of a baker's map. The right figure shows the time series of y of the density estimator. The density p is estimated at each time steps. p stays around 0.5 for a while because the dynamics of baker's map is completely chaotic. The dynamics of density estimator finally converges to $y = 0.0$ following with the convergence of the dynamics of baker's map due to the finite precision.

5 Concluding remarks and future works

We have investigated a dynamical systems called switching map systems as a new model of computation. We have shown that the switching map system with two types of baker's map (or Smale's horseshoe map) is computationally universal. If their space is defined on $\mathbf{R}$, it can be a model of real number computation and if the space is restricted to the set of the dyadic rationals with finite binary expansions, it is regarded as a model of classical computation.

A connection between the generalized shifts [Moore 91] and the input-output mappings in the BSS model [Blum, Cucker, Shub and Smale 98] has been shown with our model of computation. Our model is a class of the input-output mappings, but at the same time it is a class of GSs if it is 'compiled' to a single two-dimensional piecewise-linear mapping. Note that the BSS model allows for a linearly growing number of variables, while the switching map systems have a fixed number of variables.

We have presented four examples of switching map systems $F(B_0^{-1}, B_1)$ on $\mathbf{Q}$ corresponding to the Chomsky hierarchy. In some examples, they have no halting states and the periodic behavior is taken as the final output results.

Taking the other nonlinear mappings as elementary operations instead of the baker's maps, it is expected that the dynamical system can show a new model of computation with the nonlinearity as an oracle. We have defined nonlinear computation, and presented a density estimator as a simple example of nonlinear computation.

It is thought that the non-hyperbolicity of dynamical systems would possibly gain its computational power. The notion of computational complexity class may be useful for the studies of non-hyperbolic dynamical systems in terms of the pruning front theory [Cvitanovic, Gnaratne and Proccacia 88]. The pseudo-orbit tracing property can be also analyzed with computational theoretic approaches. We can possibly define a new class of the computational complexity based on dynamical systems and analyze the chaos and colored-noises in computational theoretical point of view. To study the computational power of the switching map systems with non-hyperbolic mapping is left to a future work.

In below, we list up some of the future directions of the switching map systems.

Probabilistic computation: The problem of deterministic chaotic maps has a complete solution at the statistical level, where the intrinsic probabilistic aspect of the dynamics is fully understood (see e.g. [Antoniou and Tasaki 93]). Thus we can investigate the probabilistic Turing machines with our deterministic switching map systems with chaotic maps.

Nondeterminism: From a point of view of dynamical systems, it is natural that the initial value in phase space should be given as ϵ-ball including measurement error rather than given as a point. In this case, the system shows non-deterministic behavior and the orbits of the system bifurcates ununiformly caused from the partitioning action of branching function g (Figure 12). It is thought that this dynamics is a kind of non-deterministic computation processes and would be interesting to study the computational class of non-deterministic switching map systems.

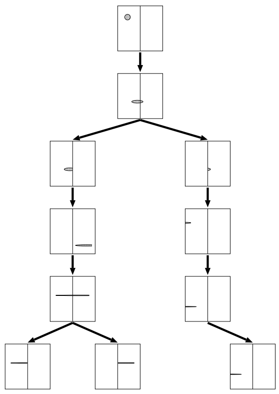


Figure 12: Schematic view of the dynamics of non-deterministic switching map systems: The orbits of the system bifurcates ununiformly with the partitioning action of the branching function.

Computation flow: A GS is topologically embedded into a smooth flow in $\mathbf{R}^3$ with suspension on the extended phase space [Moore 91]. In the same way, the switching map systems can be embedded into the network of ordinary differential equations as "Computation flow." (See Figure 13, 14). Studies on the construction of hyperchaos [Rösler 79] and the hybrid systems [Moore 98] [Asartin, Maler and Pnueli 95] would be related to these scheme.

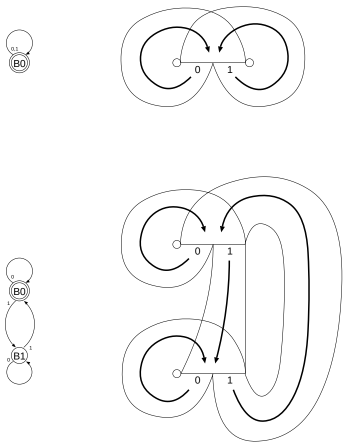


Figure 13: Schematic view of the construction of the 2-dimensional computation flow on the extended phase space.

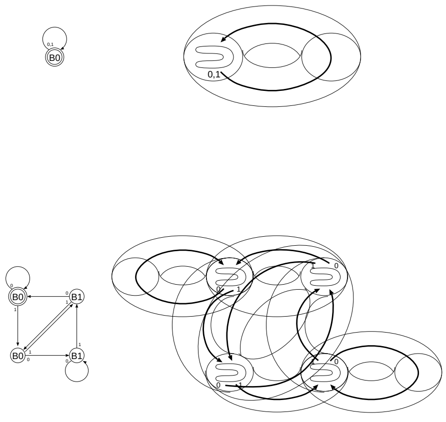


Figure 14: Schematic view of the construction of the 3-dimensional computation flow on the extended phase space.

Acknowledgements

The authors would like to thank Prof. Makoto Taiji (ISM, Japan) and Kentaro Goto (Hokkaido Univ., Japan) for valuable comments. This work was partially supported by research fellowships of the Japan society for the promotion of science for young scientists (Grant No. 09352).

References

[Antoniou and Tasaki 93] Antoniou, I., Tasaki S.: "Spectral decompositions of the Renyi map"; Journal of Physics, A26, (1993), 73-94.

[Asartin, Maler and Pnueli 95] Asartin, E., Maler, O., and Pnueli, A.: "Reachability analysis of dynamical systems with piecewise-constant derivatives"; Theoretical Computer Science, 138, (1995), 35-66.

[Blum, Shub and Smale 90] Blum, L., Shub, M. and Smale, S.: "On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines"; Bull. Amer. Math. Soc., 21, (1990), 1-46.

[Blum, Cucker, Shub and Smale 98] Blum, L., Cucker, F., Shub, M. and Smale, S.: "Complexity and Real Computation"; Springer-Verlag / New York (1998)

[Cornfeld, Fomin, Sinai 82] Cornfeld, I. P., Fomin, S. V. and Sinai, Y. G.: "Ergodic theory,"; Springer Verlag / New York (1982)

[Crutchfield and Young 90] Crutchfield,J. P.,Young,K.: "Computation at the onset of chaos," Complexity, Entropy and the Physics of information, Addison Wesley (1990), 223-269.

[Cvitanovic, Gnaratne and Proccacia 88] Cvitanovic, P., Gnaratne, G. H., Procaccia, I.: "Topological and metric properties of Hénon type strange attractors"; Phys. Rev., A38, 3, (1988), 1503-1520.

[Minsky 67] Minsky, M.: "Computation: Finite and Infinite Machines," Prentice-Hall (1967)

[Moore 91] Moore, C.: "Generalized shifts: unpredictability and Undecidability in Dynamical Systems"; Nonlinearity, 4, (1991), 199-230.

[Moore 98] Moore, C.: "Dynamical recognizers: real-time language recognition by analog computers"; Theoretical Computer Science, 201, (1998), 99-136.

[Pour-El and Richards 89] Pour-El, M. B. and Richards, J. I.: "Computability in Analysis and Physics"; Springer-Verlag / Berlin (1989)

[Rösler 79] Rösler, O. E.: "Chaotic oscillations: An example of hyperchaos in Nonlinear Oscillations in Biology"; Lectures in Applied Mathematics, 17, (1979), 141-156.

[Sato] Y. Sato, M. Taiji, and T. Ikegami: "On the computational power of nonlinear mappings in switching map systems"; in preparation.

[Siegelmann 99] Siegelmann, H. T.: "Neural Networks and Analog Computation"; Birkhauser / Boston (1999)

[Smale 67] Smale, S.: "Differentiable dynamical systems"; Bull. Am. Math. Soc., 73, (1967), 473-487.

[Tsuda 91] Tsuda, I.: "Chaotic itinerancy as a dynamical basis of hermeneutics in brain and mind"; World Futures, 32, (1991), 167-173.

Appendix: Examples of the computational process

We give examples of the computational processes of the recognizers in section 3. In below, the labels of the internal states (left), the symbol sequences on the tape (center) and the points in the phase space (right) are shown. The initial decimal point of x is denoted by $\overline{*}$ and we can trace the left/right-shifts actions. These data are generated with the program made by Kentaro Goto (Hokkaido Univ.).

A.1 Parity Checker

(1) The input is $x = .011101\dot{0}$. The number of '1' in x is an even number and the dynamics converges to (0.0, 0.0).

0: 00000000000000000000.$\overline{0}$1110100000000000000	(0.453125, 0.000000)
0: 0000000000000000000$\overline{0}$.11101000000000000000	(0.906250, 0.000000)
1: 000000000000000000$\overline{0}$1.11010000000000000000	(0.812500, 0.500000)
3: 00000000000000000$\overline{0}$10.10100000000000000000	(0.625000, 0.250000)
1: 0000000000000000$\overline{0}$101.01000000000000000000	(0.250000, 0.625000)
2: 000000000000000$\overline{0}$1011.10000000000000000000	(0.500000, 0.812500)
3: 00000000000000$\overline{0}$10110.00000000000000000000	(0.000000, 0.406250)
0: 0000000000000$\overline{0}$101100.00000000000000000000	(0.000000, 0.203125)
0: 000000000000$\overline{0}$1011000.00000000000000000000	(0.000000, 0.101562)
0: 00000000000$\overline{0}$10110000.00000000000000000000	(0.000000, 0.050781)
0: 0000000000$\overline{0}$101100000.00000000000000000000	(0.000000, 0.025391)
0: 000000000$\overline{0}$1011000000.00000000000000000000	(0.000000, 0.012695)
0: 00000000$\overline{0}$10110000000.00000000000000000000	(0.000000, 0.006348)
0: 0000000$\overline{0}$101100000000.00000000000000000000	(0.000000, 0.003174)
	$\vdots$
	$\to (0.0, 0.0)$

(2) The input is $x = .001101\dot{0}$. The number of '1' in x is an odd number and the dynamics converges to (0.0, 1.0).

0: 00000000000000000000.$\overline{0}$0110100000000000000	(0.203125, 0.000000)
0: 0000000000000000000$\overline{0}$.01101000000000000000	(0.406250, 0.000000)
0: 000000000000000000$\overline{0}$0.11010000000000000000	(0.812500, 0.000000)
1: 00000000000000000$\overline{0}$01.10100000000000000000	(0.625000, 0.500000)
3: 0000000000000000$\overline{0}$010.01000000000000000000	(0.250000, 0.250000)
0: 000000000000000$\overline{0}$0100.10000000000000000000	(0.500000, 0.125000)
1: 00000000000000$\overline{0}$01001.00000000000000000000	(0.000000, 0.562500)
2: 0000000000000$\overline{0}$010011.00000000000000000000	(0.000000, 0.781250)
2: 000000000000$\overline{0}$0100111.00000000000000000000	(0.000000, 0.890625)
2: 00000000000$\overline{0}$01001111.00000000000000000000	(0.000000, 0.945312)
2: 0000000000$\overline{0}$010011111.00000000000000000000	(0.000000, 0.972656)
2: 000000000$\overline{0}$0100111111.00000000000000000000	(0.000000, 0.986328)
2: 00000000$\overline{0}$01001111111.00000000000000000000	(0.000000, 0.993164)
2: 0000000$\overline{0}$010011111111.00000000000000000000	(0.000000, 0.996582)
	$\vdots$
	$\to (0.0, 1.0)$

A.2 Parenthesis Checker

(1) The input is $x = .010111\dot{0}$ and $y = .1\dot{0}$, that is ()()))(((.... '(' and ')' in x is inconsistent and the dynamics converges to (0.0, 0.0).

00: 00000000000000000001.$\overline{0}$1011100000000000000	(0.359375, 0.500000)
00: 0000000000000000001$\overline{1}$.10111000000000000000	(0.718750, 0.750000)
01: 000000000000000001$\overline{1}$0.01110000000000000000	(0.437500, 0.375000)
02: 0000000000000000001$\overline{1}$.00111000000000000000	(0.218750, 0.750000)
03: 00000000000000000001.$\overline{1}$0011100000000000000	(0.609375, 0.500000)
05: 00000000000000000000.1$\overline{1}$001110000000000000	(0.804688, 0.000000)
06: 00000000000000000000.01$\overline{1}$00111000000000000	(0.402344, 0.000000)
07: 00000000000000000001.1$\overline{1}$001110000000000000	(0.804688, 0.500000)
08: 00000000000000000000.11$\overline{1}$00111000000000000	(0.902344, 0.000000)
09: 00000000000000000000.1$\overline{1}$001110000000000000	(0.804688, 0.000000)
10: 00000000000000000000.$\overline{1}$0011100000000000000	(0.609375, 0.000000)
11: 0000000000000000000$\overline{0}$.00111000000000000000	(0.218750, 0.000000)
00: 000000000000000000$\overline{0}$1.01110000000000000000	(0.437500, 0.500000)
00: 00000000000000000$\overline{0}$11.11100000000000000000	(0.875000, 0.750000)
01: 0000000000000000$\overline{0}$110.11000000000000000000	(0.750000, 0.375000)
02: 00000000000000000$\overline{0}$11.01100000000000000000	(0.375000, 0.750000)
03: 000000000000000000$\overline{0}$1.10110000000000000000	(0.687500, 0.500000)
05: 0000000000000000000$\overline{0}$.11011000000000000000	(0.843750, 0.000000)
06: 00000000000000000000.$\overline{0}$1101100000000000000	(0.421875, 0.000000)
07: 0000000000000000000$\overline{1}$.11011000000000000000	(0.843750, 0.500000)
08: 00000000000000000000.$\overline{1}$1101100000000000000	(0.921875, 0.000000)
09: 0000000000000000000$\overline{0}$.11011000000000000000	(0.843750, 0.000000)
10: 000000000000000000$\overline{0}$0.10110000000000000000	(0.687500, 0.000000)
11: 00000000000000000$\overline{0}$00.01100000000000000000	(0.375000, 0.000000)
00: 0000000000000000$\overline{0}$001.11000000000000000000	(0.750000, 0.500000)
01: 000000000000000$\overline{0}$0010.10000000000000000000	(0.500000, 0.250000)
02: 0000000000000000$\overline{0}$001.01000000000000000000	(0.250000, 0.500000)
03: 00000000000000000$\overline{0}$00.10100000000000000000	(0.625000, 0.000000)
05: 000000000000000000$\overline{0}$0.01010000000000000000	(0.312500, 0.000000)
04: 0000000000000000000$\overline{0}$.00101000000000000000	(0.156250, 0.000000)
04: 00000000000000000000.$\overline{0}$0010100000000000000	(0.078125, 0.000000)
04: 00000000000000000000.0$\overline{0}$001010000000000000	(0.039062, 0.000000)
04: 00000000000000000000.00$\overline{0}$00101000000000000	(0.019531, 0.000000)
04: 00000000000000000000.000$\overline{0}$0010100000000000	(0.009766, 0.000000)
04: 00000000000000000000.0000$\overline{0}$001010000000000	(0.004883, 0.000000)
04: 00000000000000000000.00000$\overline{0}$00101000000000	(0.002441, 0.000000)
04: 00000000000000000000.000000$\overline{0}$0010100000000	(0.001221, 0.000000)
04: 00000000000000000000.0000000$\overline{0}$001010000000	(0.000610, 0.000000)
04: 00000000000000000000.00000000$\overline{0}$00101000000	(0.000305, 0.000000)
04: 00000000000000000000.000000000$\overline{0}$0010100000	(0.000153, 0.000000)
04: 00000000000000000000.0000000000$\overline{0}$001010000	(0.000076, 0.000000)
04: 00000000000000000000.00000000000$\overline{0}$00101000	(0.000038, 0.000000)
04: 00000000000000000000.000000000000$\overline{0}$0010100	(0.000019, 0.000000)

$$\vdots$$

$$\to (0.0, 0.0)$$

(2) The input is $x = .0011\dot{0}$ and $y = .1\dot{0}$, that is (())(((. . . . '(' and ')' in x is consistent (or undecided) and the dynamics converges to (0.0, 1.0).

00:	$00000000000000000001.\overline{0}0110000000000000000$	(0.187500, 0.500000)
00:	$0000000000000000001\overline{1}.01100000000000000000$	(0.375000, 0.750000)
00:	$000000000000000001\overline{1}1.11000000000000000000$	(0.750000, 0.875000)
01:	$00000000000000001\overline{1}10.10000000000000000000$	(0.500000, 0.437500)
02:	$000000000000000001\overline{1}1.01000000000000000000$	(0.250000, 0.875000)
03:	$0000000000000000001\overline{1}.10100000000000000000$	(0.625000, 0.750000)
05:	$00000000000000000001.\overline{1}1010000000000000000$	(0.812500, 0.500000)
06:	$00000000000000000000.1\overline{1}101000000000000000$	(0.906250, 0.000000)
06:	$00000000000000000000.01\overline{1}10100000000000000$	(0.453125, 0.000000)
07:	$00000000000000000001.1\overline{1}101000000000000000$	(0.906250, 0.500000)
08:	$00000000000000000000.11\overline{1}10100000000000000$	(0.953125, 0.000000)
09:	$00000000000000000000.1\overline{1}101000000000000000$	(0.906250, 0.000000)
10:	$00000000000000000000.\overline{1}1010000000000000000$	(0.812500, 0.000000)
11:	$0000000000000000000\overline{0}.10100000000000000000$	(0.625000, 0.000000)
12:	$000000000000000000\overline{0}0.01000000000000000000$	(0.250000, 0.000000)
13:	$0000000000000000000\overline{0}.00100000000000000000$	(0.125000, 0.000000)
14:	$000000000000000000\overline{0}1.01000000000000000000$	(0.250000, 0.500000)
00:	$00000000000000000\overline{0}11.10000000000000000000$	(0.500000, 0.750000)
01:	$0000000000000000\overline{0}110.00000000000000000000$	(0.000000, 0.375000)
02:	$00000000000000000\overline{0}11.00000000000000000000$	(0.000000, 0.750000)
03:	$000000000000000000\overline{0}1.10000000000000000000$	(0.500000, 0.500000)
05:	$0000000000000000000\overline{0}.11000000000000000000$	(0.750000, 0.000000)
06:	$00000000000000000000.\overline{0}1100000000000000000$	(0.375000, 0.000000)
07:	$0000000000000000000\overline{1}.11000000000000000000$	(0.750000, 0.500000)
08:	$00000000000000000000.\overline{1}1100000000000000000$	(0.875000, 0.000000)
09:	$0000000000000000000\overline{0}.11000000000000000000$	(0.750000, 0.000000)
10:	$000000000000000000\overline{0}0.10000000000000000000$	(0.500000, 0.000000)
11:	$00000000000000000\overline{0}00.00000000000000000000$	(0.000000, 0.000000)
00:	$0000000000000000\overline{0}001.00000000000000000000$	(0.000000, 0.500000)
00:	$000000000000000\overline{0}0011.00000000000000000000$	(0.000000, 0.750000)
00:	$00000000000000\overline{0}00111.00000000000000000000$	(0.000000, 0.875000)
00:	$0000000000000\overline{0}001111.00000000000000000000$	(0.000000, 0.937500)
00:	$000000000000\overline{0}0011111.00000000000000000000$	(0.000000, 0.968750)
00:	$00000000000\overline{0}00111111.00000000000000000000$	(0.000000, 0.984375)
00:	$0000000000\overline{0}001111111.00000000000000000000$	(0.000000, 0.992188)
00:	$000000000\overline{0}0011111111.00000000000000000000$	(0.000000, 0.996094)
00:	$00000000\overline{0}00111111111.00000000000000000000$	(0.000000, 0.998047)
00:	$0000000\overline{0}001111111111.00000000000000000000$	(0.000000, 0.999023)
00:	$000000\overline{0}0011111111111.00000000000000000000$	(0.000000, 0.999512)
00:	$00000\overline{0}00111111111111.00000000000000000000$	(0.000000, 0.999756)
00:	$0000\overline{0}001111111111111.00000000000000000000$	(0.000000, 0.999878)
00:	$000\overline{0}0011111111111111.00000000000000000000$	(0.000000, 0.999939)
00:	$00\overline{0}00111111111111111.00000000000000000000$	(0.000000, 0.999969)
00:	$0\overline{0}001111111111111111.00000000000000000000$	(0.000000, 0.999985)

$$\vdots$$

$$\rightarrow (0.0, 1.0)$$

A.3 Primality Checker

(1) The input is $x = .00000011\dot{0}$ and $y = .11\dot{0}$. The number of '00' in x is 3. It is a prime number and the dynamics converges to (0.0, 0.0).

00: $00000000000000000011.\overline{0}0000011000000000000$	(0.011719, 0.750000)
05: $0000000000000000011\overline{0}.00000110000000000000$	(0.023438, 0.375000)
07: $000000000000000011\overline{0}1.00001100000000000000$	(0.046875, 0.687500)
08: $00000000000000011\overline{0}10.00011000000000000000$	(0.093750, 0.343750)
09: $0000000000000011\overline{0}100.00110000000000000000$	(0.187500, 0.171875)
08: $000000000000011\overline{0}1000.01100000000000000000$	(0.375000, 0.085938)
09: $00000000000011\overline{0}10000.11000000000000000000$	(0.750000, 0.042969)
11: $0000000000011\overline{0}100001.10000000000000000000$	(0.500000, 0.521484)
15: $00000000000011\overline{0}10000.11000000000000000000$	(0.750000, 0.042969)
16: $000000000000011\overline{0}1000.01100000000000000000$	(0.375000, 0.085938)
15: $0000000000000011\overline{0}100.00110000000000000000$	(0.187500, 0.171875)
16: $00000000000000011\overline{0}10.00011000000000000000$	(0.093750, 0.343750)
15: $000000000000000011\overline{0}1.00001100000000000000$	(0.046875, 0.687500)
16: $0000000000000000011\overline{0}.10000110000000000000$	(0.523438, 0.375000)
17: $00000000000000000011.\overline{0}1000011000000000000$	(0.261719, 0.750000)
16: $00000000000000000001.1\overline{0}100001100000000000$	(0.630859, 0.500000)
17: $00000000000000000000.11\overline{0}10000110000000000$	(0.815430, 0.000000)
18: $00000000000000000001.1\overline{0}100001100000000000$	(0.630859, 0.500000)
19: $00000000000000000011.\overline{0}1000011000000000000$	(0.261719, 0.750000)
22: $0000000000000000011\overline{0}.10000110000000000000$	(0.523438, 0.375000)
23: $00000000000000000011.\overline{0}1000011000000000000$	(0.261719, 0.750000)
24: $0000000000000000011\overline{1}.10000110000000000000$	(0.523438, 0.875000)
25: $000000000000000011\overline{1}0.00001100000000000000$	(0.046875, 0.437500)
31: $00000000000000011\overline{1}00.00011000000000000000$	(0.093750, 0.218750)
33: $0000000000000011\overline{1}000.00110000000000000000$	(0.187500, 0.109375)
34: $000000000000011\overline{1}0000.01100000000000000000$	(0.375000, 0.054688)
39: $00000000000011\overline{1}00001.11000000000000000000$	(0.750000, 0.527344)
40: $000000000000011\overline{1}0000.11100000000000000000$	(0.875000, 0.054688)
41: $0000000000000011\overline{1}000.01110000000000000000$	(0.437500, 0.109375)
42: $00000000000000011\overline{1}00.00111000000000000000$	(0.218750, 0.218750)
43: $000000000000000011\overline{1}0.00011100000000000000$	(0.109375, 0.437500)
42: $0000000000000000011\overline{1}.00001110000000000000$	(0.054688, 0.875000)
43: $00000000000000000011.\overline{1}0000111000000000000$	(0.527344, 0.750000)
44: $0000000000000000011\overline{1}.00001110000000000000$	(0.054688, 0.875000)
45: $000000000000000011\overline{1}0.00011100000000000000$	(0.109375, 0.437500)
22: $00000000000000011\overline{1}00.00111000000000000000$	(0.218750, 0.218750)
21: $0000000000000011\overline{1}000.01110000000000000000$	(0.437500, 0.109375)
22: $000000000000011\overline{1}0000.11100000000000000000$	(0.875000, 0.054688)
23: $0000000000000011\overline{1}000.01110000000000000000$	(0.437500, 0.109375)
24: $000000000000011\overline{1}0001.11100000000000000000$	(0.875000, 0.554688)
25: $00000000000011\overline{1}00010.11000000000000000000$	(0.750000, 0.277344)
26: $0000000000011\overline{1}000101.10000000000000000000$	(0.500000, 0.638672)
27: $00000000000011\overline{1}00010.11000000000000000000$	(0.750000, 0.277344)
28: $000000000000011\overline{1}0001.01100000000000000000$	(0.375000, 0.554688)
27: $0000000000000011\overline{1}000.10110000000000000000$	(0.687500, 0.109375)

28: 00000000000000011$\overline{1}$00.01011000000000000000	(0.343750, 0.218750)
27: 000000000000000011$\overline{1}$0.00101100000000000000	(0.171875, 0.437500)
28: 0000000000000000011$\overline{1}$.00010110000000000000	(0.085938, 0.875000)
27: 00000000000000000011.$\overline{1}$0001011000000000000	(0.542969, 0.750000)
28: 00000000000000000001.1$\overline{1}$000101100000000000	(0.771484, 0.500000)
29: 00000000000000000000.11$\overline{1}$00010110000000000	(0.885742, 0.000000)
30: 00000000000000000001.1$\overline{1}$000101100000000000	(0.771484, 0.500000)
00: 00000000000000000011.$\overline{1}$0001011000000000000	(0.542969, 0.750000)
01: 0000000000000000011$\overline{1}$.00010110000000000000	(0.085938, 0.875000)
02: 00000000000000000011.$\overline{1}$0001011000000000000	(0.542969, 0.750000)
03: 0000000000000000011$\overline{0}$.00010110000000000000	(0.085938, 0.375000)
04: 000000000000000011$\overline{0}$1.00101100000000000000	(0.171875, 0.687500)
05: 00000000000000011$\overline{0}$10.01011000000000000000	(0.343750, 0.343750)
07: 0000000000000011$\overline{0}$101.10110000000000000000	(0.687500, 0.671875)
11: 000000000000011$\overline{0}$1011.01100000000000000000	(0.375000, 0.835938)
12: 0000000000000011$\overline{0}$101.10110000000000000000	(0.687500, 0.671875)
13: 000000000000011$\overline{0}$1010.01100000000000000000	(0.375000, 0.335938)
14: 00000000000011$\overline{0}$10100.11000000000000000000	(0.750000, 0.167969)
11: 0000000000011$\overline{0}$101001.10000000000000000000	(0.500000, 0.583984)
15: 00000000000011$\overline{0}$10100.11000000000000000000	(0.750000, 0.167969)
16: 000000000000011$\overline{0}$1010.01100000000000000000	(0.375000, 0.335938)
15: 0000000000000011$\overline{0}$101.00110000000000000000	(0.187500, 0.671875)
16: 00000000000000011$\overline{0}$10.10011000000000000000	(0.593750, 0.343750)
17: 000000000000000011$\overline{0}$1.01001100000000000000	(0.296875, 0.687500)
16: 0000000000000000011$\overline{0}$.10100110000000000000	(0.648438, 0.375000)
17: 00000000000000000011.$\overline{0}$1010011000000000000	(0.324219, 0.750000)
16: 00000000000000000001.1$\overline{0}$101001100000000000	(0.662109, 0.500000)
17: 00000000000000000000.11$\overline{0}$10100110000000000	(0.831055, 0.000000)
18: 00000000000000000001.1$\overline{0}$101001100000000000	(0.662109, 0.500000)
19: 00000000000000000011.$\overline{0}$1010011000000000000	(0.324219, 0.750000)
22: 0000000000000000011$\overline{0}$.10100110000000000000	(0.648438, 0.375000)
23: 00000000000000000011.$\overline{0}$1010011000000000000	(0.324219, 0.750000)
24: 0000000000000000011$\overline{1}$.10100110000000000000	(0.648438, 0.875000)
25: 000000000000000011$\overline{1}$0.01001100000000000000	(0.296875, 0.437500)
31: 00000000000000011$\overline{1}$00.10011000000000000000	(0.593750, 0.218750)
32: 0000000000000011$\overline{1}$001.00110000000000000000	(0.187500, 0.609375)
31: 000000000000011$\overline{1}$0010.01100000000000000000	(0.375000, 0.304688)
33: 00000000000011$\overline{1}$00100.11000000000000000000	(0.750000, 0.152344)
46: 0000000000011$\overline{1}$001001.10000000000000000000	(0.500000, 0.576172)
48: 00000000000011$\overline{1}$00100.11000000000000000000	(0.750000, 0.152344)
49: 000000000000011$\overline{1}$0010.01100000000000000000	(0.375000, 0.304688)
50: 0000000000000011$\overline{1}$001.00110000000000000000	(0.187500, 0.609375)
49: 00000000000000011$\overline{1}$00.10011000000000000000	(0.593750, 0.218750)
48: 000000000000000011$\overline{1}$0.01001100000000000000	(0.296875, 0.437500)
49: 0000000000000000011$\overline{1}$.00100110000000000000	(0.148438, 0.875000)
50: 00000000000000000011.$\overline{1}$0010011000000000000	(0.574219, 0.750000)
51: 00000000000000000001.1$\overline{1}$001001100000000000	(0.787109, 0.500000)
52: 00000000000000000000.11$\overline{1}$00100110000000000	(0.893555, 0.000000)
53: 00000000000000000000.011$\overline{1}$0010011000000000	(0.446777, 0.000000)
53: 00000000000000000000.0011$\overline{1}$001001100000000	(0.223389, 0.000000)

53: 00000000000000000000.00011$\overline{1}$00100110000000	(0.111694, 0.000000)
53: 00000000000000000000.000011$\overline{1}$0010011000000	(0.055847, 0.000000)
53: 00000000000000000000.0000011$\overline{1}$001001100000	(0.027924, 0.000000)
53: 00000000000000000000.00000011$\overline{1}$00100110000	(0.013962, 0.000000)
53: 00000000000000000000.000000011$\overline{1}$0010011000	(0.006981, 0.000000)
53: 00000000000000000000.0000000011$\overline{1}$001001100	(0.003490, 0.000000)
53: 00000000000000000000.00000000011$\overline{1}$00100110	(0.001745, 0.000000)
53: 00000000000000000000.000000000011$\overline{1}$0010011	(0.000873, 0.000000)
53: 00000000000000000000.0000000000011$\overline{1}$001001	(0.000436, 0.000000)
53: 00000000000000000000.00000000000011$\overline{1}$00100	(0.000218, 0.000000)
	$\vdots$
	$\rightarrow (0.0, 0.0)$

(2) The input is $x = .0000000011\dot{0}$ and $y = .11\dot{0}$. The number of '00' in x is 4 and the minimal prime factor is 2. The dynamics converges to a 10-periodic orbit.

00: 00000000000000000011.$\overline{0}$00000001100000000000	(0.002930, 0.750000)
05: 0000000000000000011$\overline{0}$.00000001100000000000	(0.005859, 0.375000)
07: 000000000000000011$\overline{0}$1.00000011000000000000	(0.011719, 0.687500)
08: 00000000000000011$\overline{0}$10.00000110000000000000	(0.023438, 0.343750)
09: 0000000000000011$\overline{0}$100.00001100000000000000	(0.046875, 0.171875)
08: 000000000000011$\overline{0}$1000.00011000000000000000	(0.093750, 0.085938)
09: 00000000000011$\overline{0}$10000.00110000000000000000	(0.187500, 0.042969)
08: 0000000000011$\overline{0}$100000.01100000000000000000	(0.375000, 0.021484)
09: 000000000011$\overline{0}$1000000.11000000000000000000	(0.750000, 0.010742)
11: 00000000011$\overline{0}$10000001.10000000000000000000	(0.500000, 0.505371)
15: 000000000011$\overline{0}$1000000.11000000000000000000	(0.750000, 0.010742)
16: 0000000000011$\overline{0}$100000.01100000000000000000	(0.375000, 0.021484)
15: 00000000000011$\overline{0}$10000.00110000000000000000	(0.187500, 0.042969)
16: 000000000000011$\overline{0}$1000.00011000000000000000	(0.093750, 0.085938)
15: 0000000000000011$\overline{0}$100.00001100000000000000	(0.046875, 0.171875)
16: 00000000000000011$\overline{0}$10.00000110000000000000	(0.023438, 0.343750)
15: 000000000000000011$\overline{0}$1.00000011000000000000	(0.011719, 0.687500)
16: 0000000000000000011$\overline{0}$.10000001100000000000	(0.505859, 0.375000)
17: 00000000000000000011.$\overline{0}$10000001100000000000	(0.252930, 0.750000)
16: 00000000000000000001.1$\overline{0}$1000000110000000000	(0.626465, 0.500000)
17: 00000000000000000000.11$\overline{0}$100000011000000000	(0.813232, 0.000000)
18: 00000000000000000001.1$\overline{0}$1000000110000000000	(0.626465, 0.500000)
19: 00000000000000000011.$\overline{0}$10000001100000000000	(0.252930, 0.750000)
22: 0000000000000000011$\overline{0}$.10000001100000000000	(0.505859, 0.375000)
23: 00000000000000000011.$\overline{0}$10000001100000000000	(0.252930, 0.750000)
24: 0000000000000000011$\overline{1}$.10000001100000000000	(0.505859, 0.875000)
25: 000000000000000011$\overline{1}$0.00000011000000000000	(0.011719, 0.437500)
31: 00000000000000011$\overline{1}$00.00000110000000000000	(0.023438, 0.218750)
33: 0000000000000011$\overline{1}$000.00001100000000000000	(0.046875, 0.109375)
34: 000000000000011$\overline{1}$0000.00011000000000000000	(0.093750, 0.054688)
39: 00000000000011$\overline{1}$00001.00110000000000000000	(0.187500, 0.527344)
40: 000000000000011$\overline{1}$0000.10011000000000000000	(0.593750, 0.054688)
41: 0000000000000011$\overline{1}$000.01001100000000000000	(0.296875, 0.109375)

42: $00000000000000011\bar{1}00.00100110000000000000$ $(0.148438, 0.218750)$
43: $00000000000000001 1\bar{1}0.00010011000000000000$ $(0.074219, 0.437500)$
42: $0000000000000000011\bar{1}.00001001100000000000$ $(0.037109, 0.875000)$
43: $00000000000000000011.\bar{1}0000100110000000000$ $(0.518555, 0.750000)$
44: $0000000000000000011\bar{1}.00001001100000000000$ $(0.037109, 0.875000)$
45: $000000000000000011\bar{1}0.00010011000000000000$ $(0.074219, 0.437500)$
22: $00000000000000011\bar{1}00.00100110000000000000$ $(0.148438, 0.218750)$
21: $0000000000000011\bar{1}000.01001100000000000000$ $(0.296875, 0.109375)$
22: $000000000000011\bar{1}0000.10011000000000000000$ $(0.593750, 0.054688)$
23: $0000000000000011\bar{1}000.01001100000000000000$ $(0.296875, 0.109375)$
24: $000000000000011\bar{1}0001.10011000000000000000$ $(0.593750, 0.554688)$
25: $00000000000011\bar{1}00010.00110000000000000000$ $(0.187500, 0.277344)$
31: $0000000000011\bar{1}000100.01100000000000000000$ $(0.375000, 0.138672)$
33: $000000000011\bar{1}0001000.11000000000000000000$ $(0.750000, 0.069336)$
46: $00000000011\bar{1}00010001.10000000000000000000$ $(0.500000, 0.534668)$
48: $000000000011\bar{1}0001000.11000000000000000000$ $(0.750000, 0.069336)$
49: $0000000000011\bar{1}000100.01100000000000000000$ $(0.375000, 0.138672)$
50: $00000000000011\bar{1}00010.00110000000000000000$ $(0.187500, 0.277344)$
49: $000000000000011\bar{1}0001.00011000000000000000$ $(0.093750, 0.554688)$
50: $0000000000000011\bar{1}000.10001100000000000000$ $(0.546875, 0.109375)$
51: $00000000000000011\bar{1}00.01000110000000000000$ $(0.273438, 0.218750)$
54: $000000000000000011\bar{1}0.00100011000000000000$ $(0.136719, 0.437500)$

55: $0000000000000000011\bar{1}.00010001100000000000$ $(0.068359, 0.875000)$
56: $00000000000000000011.\bar{1}0001000110000000000$ $(0.534180, 0.750000)$
57: $00000000000000000001.1\bar{1}000100011000000000$ $(0.767090, 0.500000)$
58: $00000000000000000000.11\bar{1}00010001100000000$ $(0.883545, 0.000000)$
59: $00000000000000000001.1\bar{1}000100011000000000$ $(0.767090, 0.500000)$
60: $00000000000000000011.\bar{1}0001000110000000000$ $(0.534180, 0.750000)$
63: $0000000000000000011\bar{1}.00010001100000000000$ $(0.068359, 0.875000)$
62: $000000000000000011\bar{1}0.00100011000000000000$ $(0.136719, 0.437500)$
61: $00000000000000011\bar{1}00.01000110000000000000$ $(0.273438, 0.218750)$
64: $000000000000000011\bar{1}0.00100011000000000000$ $(0.136719, 0.437500)$

55: $0000000000000000011\bar{1}.00010001100000000000$ $(0.068359, 0.875000)$
56: $00000000000000000011.\bar{1}0001000110000000000$ $(0.534180, 0.750000)$
57: $00000000000000000001.1\bar{1}000100011000000000$ $(0.767090, 0.500000)$
58: $00000000000000000000.11\bar{1}00010001100000000$ $(0.883545, 0.000000)$
59: $00000000000000000001.1\bar{1}000100011000000000$ $(0.767090, 0.500000)$
60: $00000000000000000011.\bar{1}0001000110000000000$ $(0.534180, 0.750000)$
63: $0000000000000000011\bar{1}.00010001100000000000$ $(0.068359, 0.875000)$
62: $000000000000000011\bar{1}0.00100011000000000000$ $(0.136719, 0.437500)$
61: $00000000000000011\bar{1}00.01000110000000000000$ $(0.273438, 0.218750)$
64: $000000000000000011\bar{1}0.00100011000000000000$ $(0.136719, 0.437500)$

$\vdots$

A.4 Minsky's universal Turing machine

(1) The input is $x = .110111\dot{0}$ and $y = .11\dot{0}$. The dynamics reaches the halting state.

00: $00000000000000000011.\bar{1}1011100000000000000$	(0.859375, 0.750000)
05: $0000000000000000011\bar{0}.10111000000000000000$	(0.718750, 0.375000)
18: $00000000000000000011.\bar{0}1011100000000000000$	(0.359375, 0.750000)
19: $0000000000000000011\bar{1}.10111000000000000000$	(0.718750, 0.875000)
20: $000000000000000011\bar{1}0.01110000000000000000$	(0.437500, 0.437500)
27: $00000000000000011\bar{1}01.11100000000000000000$	(0.875000, 0.718750)
28: $000000000000000011\bar{1}0.11110000000000000000$	(0.937500, 0.437500)
29: $0000000000000000011\bar{1}.01111000000000000000$	(0.468750, 0.875000)
30: $00000000000000000011.\bar{1}0111100000000000000$	(0.734375, 0.750000)
31: $00000000000000000001.1\bar{1}011110000000000000$	(0.867188, 0.500000)
32: $00000000000000000010.\bar{1}0111100000000000000$	(0.734375, 0.250000)
35: $00000000000000000001.0\bar{1}011110000000000000$	(0.367188, 0.500000)
36: $00000000000000000000.10\bar{1}01111000000000000$	(0.683594, 0.000000)
37: $00000000000000000000.010\bar{1}0111100000000000$	(0.341797, 0.000000)
41: $00000000000000000001.10\bar{1}01111000000000000$	(0.683594, 0.500000)
42: $00000000000000000000.110\bar{1}0111100000000000$	(0.841797, 0.000000)
43: $00000000000000000000.10\bar{1}01111000000000000$	(0.683594, 0.000000)
44: $00000000000000000000.010\bar{1}0111100000000000$	(0.341797, 0.000000)
45: $00000000000000000000.0010\bar{1}011110000000000$	(0.170898, 0.000000)
46: $00000000000000000000.00010\bar{1}01111000000000$	(0.085449, 0.000000)
47: $00000000000000000001.0010\bar{1}011110000000000$	(0.170898, 0.500000)
54: $00000000000000000011.010\bar{1}0111100000000000$	(0.341797, 0.750000)
55: $00000000000000000001.1010\bar{1}011110000000000$	(0.670898, 0.500000)
20: $00000000000000000010.010\bar{1}0111100000000000$	(0.341797, 0.250000)
27: $00000000000000000101.10\bar{1}01111000000000000$	(0.683594, 0.625000)
28: $00000000000000000010.110\bar{1}0111100000000000$	(0.841797, 0.250000)
29: $00000000000000000001.0110\bar{1}011110000000000$	(0.420898, 0.500000)
30: $00000000000000000000.10110\bar{1}01111000000000$	(0.710449, 0.000000)
31: $00000000000000000000.010110\bar{1}0111100000000$	(0.355225, 0.000000)
69: $00000000000000000001.10110\bar{1}01111000000000$	(0.710449, 0.500000)
29: $00000000000000000000.110110\bar{1}0111100000000$	(0.855225, 0.000000)
30: $00000000000000000000.0110110\bar{1}011110000000$	(0.427612, 0.000000)
31: $00000000000000000000.00110110\bar{1}01111000000$	(0.213806, 0.000000)
69: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)
70: $00000000000000000001.0110110\bar{1}011110000000$	(0.427612, 0.500000)

$$\vdots$$

$$\rightarrow (0.427612, 0.500000)$$

(2) The input is $x = .100111\dot{0}$ and $y = .11\dot{0}$. The dynamics traps a loop of internal states.

00: $00000000000000000011.\overline{1}0011100000000000000$ $(0.609375, 0.750000)$
05: $0000000000000000011\overline{0}.00111000000000000000$ $(0.218750, 0.375000)$
06: $00000000000000000011.\overline{0}0011100000000000000$ $(0.109375, 0.750000)$
07: $00000000000000000001.1\overline{0}001110000000000000$ $(0.554688, 0.500000)$
08: $00000000000000000000.11\overline{0}00111000000000000$ $(0.777344, 0.000000)$
09: $00000000000000000000.1\overline{0}001110000000000000$ $(0.554688, 0.000000)$

10: $00000000000000000000.01\overline{0}00111000000000000$ $(0.277344, 0.000000)$
11: $00000000000000000000.001\overline{0}0011100000000000$ $(0.138672, 0.000000)$
08: $00000000000000000000.0001\overline{0}001110000000000$ $(0.069336, 0.000000)$
12: $00000000000000000001.001\overline{0}0011100000000000$ $(0.138672, 0.500000)$
13: $00000000000000000000.1001\overline{0}001110000000000$ $(0.569336, 0.000000)$
14: $00000000000000000000.001\overline{0}0011100000000000$ $(0.138672, 0.000000)$

10: $00000000000000000000.0001\overline{0}001110000000000$ $(0.069336, 0.000000)$
11: $00000000000000000000.00001\overline{0}00111000000000$ $(0.034668, 0.000000)$
08: $00000000000000000000.000001\overline{0}0011100000000$ $(0.017334, 0.000000)$
12: $00000000000000000001.00001\overline{0}00111000000000$ $(0.034668, 0.500000)$
13: $00000000000000000000.100001\overline{0}0011100000000$ $(0.517334, 0.000000)$
14: $00000000000000000000.00001\overline{0}00111000000000$ $(0.034668, 0.000000)$

10: $00000000000000000000.000001\overline{0}0011100000000$ $(0.017334, 0.000000)$
11: $00000000000000000000.0000001\overline{0}001110000000$ $(0.008667, 0.000000)$
08: $00000000000000000000.00000001\overline{0}00111000000$ $(0.004333, 0.000000)$
12: $00000000000000000001.0000001\overline{0}001110000000$ $(0.008667, 0.500000)$
13: $00000000000000000000.10000001\overline{0}00111000000$ $(0.504333, 0.000000)$
14: $00000000000000000000.0000001\overline{0}001110000000$ $(0.008667, 0.000000)$

10: $00000000000000000000.00000001\overline{0}00111000000$ $(0.004333, 0.000000)$
11: $00000000000000000000.000000001\overline{0}0011100000$ $(0.002167, 0.000000)$
08: $00000000000000000000.0000000001\overline{0}001110000$ $(0.001083, 0.000000)$
12: $00000000000000000001.000000001\overline{0}0011100000$ $(0.002167, 0.500000)$
13: $00000000000000000000.1000000001\overline{0}001110000$ $(0.501083, 0.000000)$
14: $00000000000000000000.000000001\overline{0}0011100000$ $(0.002167, 0.000000)$

10: $00000000000000000000.0000000001\overline{0}001110000$ $(0.001083, 0.000000)$
11: $00000000000000000000.00000000001\overline{0}00111000$ $(0.000542, 0.000000)$
08: $00000000000000000000.000000000001\overline{0}0011100$ $(0.000271, 0.000000)$
12: $00000000000000000001.00000000001\overline{0}00111000$ $(0.000542, 0.500000)$
13: $00000000000000000000.100000000001\overline{0}0011100$ $(0.500271, 0.000000)$
14: $00000000000000000000.00000000001\overline{0}00111000$ $(0.000542, 0.000000)$

$\vdots$
$\to (0.0, 0.0)$
$\to (0.0, 0.5)$
$\to \cdots$

(3) The input is $x = .110111\dot{0}$ and $y = .10\dot{0}$. The behavior of the dynamics is undecided yet.

00: $00000000000000000001.\bar{1}1011100000000000000$	(0.859375, 0.500000)
05: $0000000000000000001\bar{0}.101110000000000000000$	(0.718750, 0.250000)
18: $00000000000000000001.\bar{0}1011100000000000000$	(0.359375, 0.500000)
19: $0000000000000000001\bar{1}.101110000000000000000$	(0.718750, 0.750000)
20: $000000000000000001\bar{1}0.01110000000000000000$	(0.437500, 0.375000)
27: $00000000000000001\bar{1}01.11100000000000000000$	(0.875000, 0.687500)
28: $000000000000000001\bar{1}0.11110000000000000000$	(0.937500, 0.375000)
29: $0000000000000000001\bar{1}.01111000000000000000$	(0.468750, 0.750000)
30: $00000000000000000001.\bar{1}0111100000000000000$	(0.734375, 0.500000)
31: $00000000000000000000.1\bar{1}011110000000000000$	(0.867188, 0.000000)
32: $00000000000000000000.\bar{1}0111100000000000000$	(0.734375, 0.000000)
35: $00000000000000000000.0\bar{1}011110000000000000$	(0.367188, 0.000000)
36: $00000000000000000000.00\bar{1}01111000000000000$	(0.183594, 0.000000)
37: $00000000000000000000.000\bar{1}0111100000000000$	(0.091797, 0.000000)
41: $00000000000000000001.00\bar{1}01111000000000000$	(0.183594, 0.500000)
56: $00000000000000000011.0\bar{1}011110000000000000$	(0.367188, 0.750000)
57: $00000000000000000001.10\bar{1}01111000000000000$	(0.683594, 0.500000)
58: $00000000000000000010.0\bar{1}011110000000000000$	(0.367188, 0.250000)
66: $00000000000000000101.\bar{1}0111100000000000000$	(0.734375, 0.625000)
64: $0000000000000000101\bar{0}.01111000000000000000$	(0.468750, 0.312500)
65: $00000000000000000101.\bar{0}0111100000000000000$	(0.234375, 0.625000)
58: $0000000000000000101\bar{1}.01111000000000000000$	(0.468750, 0.812500)
66: $000000000000000101\bar{1}1.11110000000000000000$	(0.937500, 0.906250)
64: $00000000000000101\bar{1}10.11100000000000000000$	(0.875000, 0.453125)
65: $000000000000000101\bar{1}1.01110000000000000000$	(0.437500, 0.906250)
58: $00000000000000101\bar{1}11.11100000000000000000$	(0.875000, 0.953125)
59: $0000000000000101\bar{1}110.11000000000000000000$	(0.750000, 0.476562)
64: $000000000000101\bar{1}1100.10000000000000000000$	(0.500000, 0.238281)
65: $0000000000000101\bar{1}110.01000000000000000000$	(0.250000, 0.476562)
58: $000000000000101\bar{1}1101.10000000000000000000$	(0.500000, 0.738281)
59: $00000000000101\bar{1}11010.00000000000000000000$	(0.000000, 0.369141)
60: $0000000000101\bar{1}110101.00000000000000000000$	(0.000000, 0.684570)
61: $00000000000101\bar{1}11010.10000000000000000000$	(0.500000, 0.369141)
62: $000000000000101\bar{1}1101.01000000000000000000$	(0.250000, 0.738281)
63: $00000000000101\bar{1}11011.10000000000000000000$	(0.500000, 0.869141)
58: $0000000000101\bar{1}110110.00000000000000000000$	(0.000000, 0.434570)
66: $000000000101\bar{1}1101101.00000000000000000000$	(0.000000, 0.717285)
67: $0000000000101\bar{1}110110.10000000000000000000$	(0.500000, 0.434570)
68: $00000000000101\bar{1}11011.01000000000000000000$	(0.250000, 0.869141)
31: $000000000000101\bar{1}1101.10100000000000000000$	(0.625000, 0.738281)
32: $00000000000101\bar{1}11010.01000000000000000000$	(0.250000, 0.369141)
33: $000000000000101\bar{1}1101.00100000000000000000$	(0.125000, 0.738281)
34: $00000000000101\bar{1}11011.01000000000000000000$	(0.250000, 0.869141)
29: $000000000000101\bar{1}1101.10100000000000000000$	(0.625000, 0.738281)
30: $0000000000000101\bar{1}110.11010000000000000000$	(0.812500, 0.476562)

$$\vdots$$