

A universal non-conservative reversible elementary triangular partitioned cellular automaton that shows complex behavior

Kenichi Morita¹ 

Published online: 11 November 2017
© Springer Science+Business Media B.V., part of Springer Nature 2017

Abstract We study a simple triangular partitioned cellular automaton (TPCA), and clarify its complex behavior. It is a CA with triangular cells, each of which is divided into three parts. The next state of a cell is determined by the three adjacent parts of its neighbor cells. This framework makes it easy to design reversible triangular CAs. Among them, isotropic and eight-state (i.e., each part has only two states) TPCAs are called *elementary TPCAs* (ETPCAs). They are extremely simple, since each of their local transition functions is described by only four local rules. In this paper, we investigate a specific reversible ETPCA T_{0347} , where 0347 is its identification number in the class of 256 ETPCAs. In spite of the simplicity of the local function and the constraint of reversibility, evolutions of configurations in T_{0347} have very rich varieties. It is shown that a glider, which is a space-moving pattern, and glider guns exist in this cellular space. We also show that the trajectory and the timing of a glider can be fully controlled by appropriately placing stable patterns called blocks. Furthermore, using gliders to represent signals, we can implement universal reversible logic gates in it. By this, computational universality of T_{0347} is derived.

Keywords Reversible cellular automaton · Elementary triangular partitioned cellular automaton · Computational universality · Reversible logic gate · Glider

This paper is an extended version of the paper presented in *Cellular Automata and Discrete Complex Systems* (eds. M. Cook and T. Neary), LNCS 9664, pp. 170–184 (2016).

✉ Kenichi Morita
km@hiroshima-u.ac.jp

¹ Hiroshima University, Higashi-Hiroshima 739-8527, Japan

1 Introduction

A three-neighbor triangular cellular automaton (TCA) is one whose cell is triangular, and communicates with its three neighbor cells. Bays (1994) investigated a class of TCAs with the local functions of the type of the Game-of-Life CA (Gardner 1970, 1971), and showed their interesting behavior. On the other hand, Gajardo and Goles (2001) proposed a three-state TCA (defined on a hexagonal lattice), and proved its computational universality. Imai and Morita (2000) studied a *reversible* TCA, and showed that there is an eight-state universal reversible TCA. Morita (2016b) also showed another eight-state universal reversible TCA.

The framework of TCAs used in Imai and Morita (2000) is a triangular partitioned cellular automaton (TPCA) where each cell is divided into three parts, and each part has its own state set. Thus, TPCAs are a subclass of TCAs, where the state set of a cell is the Cartesian product of the state sets of the three parts. In a TPCA, the next state of a cell is determined depending only on the three adjacent parts of the neighbor cells (not depending on the states of the whole three neighbor cells). This framework is useful for designing reversible TCAs. This is because injectivity of the local transition function is equivalent to that of the global transition function.

An *elementary* TPCA (ETPCA) is one such that each part of a cell has only two states (i.e., the state set is $\{0, 1\}$, and hence a cell has eight states), and its local function is isotropic (i.e., rotation-symmetric). There are 256 ETPCAs in total, and there are 36 *reversible* ETPCAs (RETPCAs). Note that the reversible TPCAs given in Imai and Morita (2000) and Morita (2016b) are RETPCAs, which are also conservative, i.e., the total number of the state 1's is conserved throughout their evolution processes. ETPCAs are

extremely simple, since each of their local functions is described by only four local rules. But, they still show interesting behavior as in the case of one-dimensional elementary cellular automata (ECAs) (Wolfram 1986, 2002).

In this paper, we investigate a specific non-conservative RETPCA T_{0347} having the identification number 0347 in the class of 256 ETPCAs. It somewhat resembles the Game-of-Life CA (Gardner 1970, 1971; Berlekamp et al. 1982), and exhibits complex behavior. In particular, there exist a *glider* and *glider guns*. The glider in T_{0347} is a moving object with period 6. There are glider guns that generate gliders in three directions as well as in one direction. There is also a gun that generates gliders to the negative time direction. We can compose right-turn, left-turn, backward-turn and U-turn modules out of stable *blocks*, which can change the moving direction of a glider. It is also possible to change the direction of a glider by colliding another glider appropriately. Based on these basic operations, we can implement *gate modules* that simulate universal reversible logic gates in the cellular space of T_{0347} . By this, computational universality of T_{0347} is concluded.

2 Elementary triangular partitioned cellular automata

In this section, we give definitions on elementary triangular partitioned cellular automata (ETPCAs), their reversibility, and some related notions.

A *partitioned cellular automaton* (PCA) is a subclass of a standard CA, where a cell is divided into several parts, and each part has its own state set. Thus, the set of states of a cell is the Cartesian product of these state sets. Figure 1a illustrates the cellular space of a two-dimensional three-neighbor *triangular PCA* (TPCA). In a TPCA, the next state of a cell is determined by the states of the adjacent parts of the three neighbor cells, not by the states of the whole three cells. More precisely, it is determined by a set of local rules of the form shown in Fig. 1b. We assume there is no pair of distinct rules that have the same left-hand sides. Namely, TPCAs considered here are

deterministic. A set of local rules, thus, defines a *local function* of a TPCA. A *configuration* of a TPCA is a state of the whole (infinite) cellular space of it. Applying the local function to all the cells in a configuration in parallel, a *global function*, which gives a transition relation among configurations, is obtained.

Here, we do not give formal definitions of a TPCA, its configuration, and its global function, since their descriptions become complex (note that a triangular CA can be formally defined as a CA on a Cayley graph as in Róka 1999). However, various notions on TPCAs given below will be clearly understood without giving formal definitions.

We say a PCA is *locally reversible* if its local function is injective, and *globally reversible* if its global function is injective. Local reversibility of a given PCA is easily tested by checking if there is no pair of local rules that have the same right-hand sides. It has been shown that global reversibility and local reversibility are equivalent (Lemma 1). Therefore, such a PCA is simply called a *reversible PCA* (RPCA). Note that the lemma is given for one-dimensional PCAs in Morita and Harao (1989), but it is easy to extend it for two-dimensional PCAs.

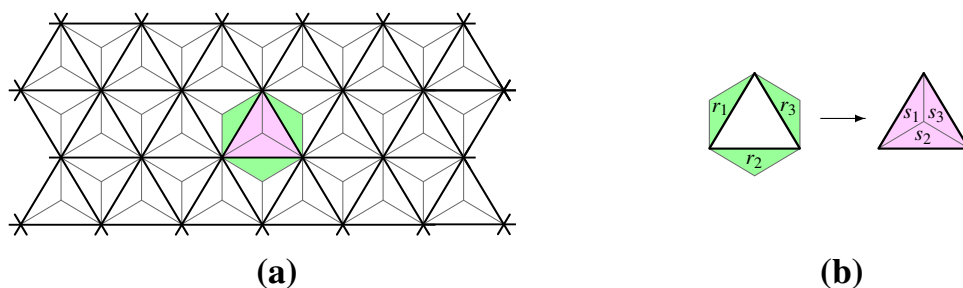
Lemma 1 (Morita and Harao 1989) *A PCA A is globally reversible iff it is locally reversible.*

By this lemma, to obtain a reversible CA, it is sufficient to give a locally reversible PCA. Thus, the framework of PCAs makes it easy to design reversible CAs.

A TPCA is called *isotropic* (or *rotation-symmetric*), if, for each local rule, the rules obtained by rotating both sides of it by a multiple of 60° exist. It should be noted that, if a TPCA is isotropic, then all three parts of a cell must have the same state set. In the following, we study only isotropic TPCAs.

An eight-state isotropic TPCA is called an *elementary TPCA* (ETPCA). Thus, each part of a cell has the state set $\{0, 1\}$ (in the following figures, the states 0 and 1 are indicated by a blank and a particle \bullet , respectively). In fact, ETPCAs are the simplest ones among two-dimensional PCAs. Yet, this class still contains many interesting PCAs as in the case of one-dimensional elementary CAs (ECAs) (Wolfram 1986, 2002).

Fig. 1 A three-neighbor triangular partitioned cellular automaton (TPCA). **a** Its cellular space, and **b** a local rule



Since an ETPCA is isotropic, and each part of a cell has two states, its local function is defined by only four local rules. Hence, an ETPCA can be specified by a four-digit number $wxyz$ where $w, z \in \{0, 7\}$ and $x, y \in \{0, 1, \dots, 7\}$ as shown in Fig. 2. Thus, there are 256 ETPCAs. Here, w and z must be 0 or 7, because ETPCAs are isotropic and deterministic. The ETPCA with the identification number $wxyz$ is denoted by T_{wxyz} . Figure 3 shows the local function of the ETPCA T_{0347} .

A reversible ETPCA is denoted by RETPCA. It is easy to see the following: An ETPCA T_{wxyz} is reversible iff

$$(w, z) \in \{(0, 7), (7, 0)\} \wedge (x, y) \in \{1, 2, 4\} \times \{3, 5, 6\} \cup \{3, 5, 6\} \times \{1, 2, 4\}.$$

Let T_{wxyz} be an ETPCA. We say T_{wxyz} is *conservative* (or *bit-conserving*), if the total number of particles (i.e., \bullet 's) is conserved in each local rule. Thus, the following holds: An ETPCA T_{wxyz} is conservative iff

$$w = 0 \wedge x \in \{1, 2, 4\} \wedge y \in \{3, 5, 6\} \wedge z = 7.$$

By above, it is easy to see that, if an ETPCA is conservative, then it is reversible. The ETPCA T_{0347} (Fig. 3) is reversible but not conservative.

In ETPCAs, there are two kinds of cells, i.e., an up-triangle cell and a down-triangle cell, as in Fig. 4. But, they differ only on their directions, and, of course, they have the same local function. Here, we denote their states shown in Fig. 4 by a triplet $(s_1, s_2, s_3) \in \{0, 1\}^3$.

For an ETPCA where $w = 0$ (i.e., T_{0xyz} for some $x, y \in \{0, \dots, 7\}$ and $z \in \{0, 7\}$), we define a *quiescent state* as the state $(0, 0, 0)$. In such an ETPCA, if all the neighbor cells are quiescent states, then the center cell becomes quiescent at the next time step. A *finite configuration* is one such that all but finite number of cells are in the quiescent state. An *infinite configuration* is one such that infinitely many cells are in non-quiescent states.

A *pattern* is a finite segment of a configuration. In what follows, various useful patterns will be given. Placing such

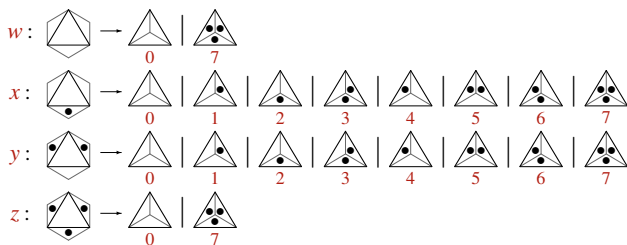


Fig. 2 Representing an ETPCA by a four-digit number $wxyz$, where $w, z \in \{0, 7\}$ and $x, y \in \{0, 1, \dots, 7\}$. The states 0 and 1 are represented by a blank and \bullet , respectively. Vertical bars indicate alternatives of the right-hand side of each local rule

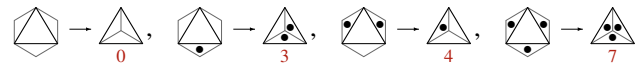


Fig. 3 Local function of the non-conservative RETPCA T_{0347} defined by four local rules



Fig. 4 **a** An up-triangle cell, and **b** a down-triangle cell in the space of ETPCA, whose states are $(s_1, s_2, s_3) \in \{0, 1\}^3$

patterns appropriately in the cellular space we can construct configurations that perform interesting tasks.

In a three-neighbor TPCA, the states in the up-triangle cells, and those in the down-triangle cells at time 0 never interact. The reason is that the next state of an up-triangle (down-triangle, respectively) cell is determined only by the neighboring three down-triangle (up-triangle) cells (Fig. 1 b). An *up-triangle configuration* (*down-triangle configuration*, respectively) is one such that all the down-triangle (up-triangle) cells are in the quiescent state. Thus, if we start from an up-triangle (down-triangle) configuration, then the next one is a down-triangle (up-triangle) configuration. From the above observation, any task performed by a TPCA can be done by giving an up-triangle (or down-triangle) configuration at time 0. However, in such configurations, “stable patterns” (Sect. 3.1.1) cannot exist. Since stable patterns are convenient for designing larger patterns and for considering their evolving processes, we use both types of cells to give stable patterns.

3 RETPCA T_{0347} and its properties

In this section, we focus on the specific non-conservative RETPCA T_{0347} (Fig. 3), and investigate its properties. We present several basic patterns, and examine how they evolve. We shall see that, in T_{0347} , many patterns show interesting behavior as in the case of the Game-of-Life CA (Berlekamp et al. 1982; Gardner 1970, 1971). In particular, a space-moving pattern called a “glider” exhibits complex behavior when it interacts with other basic patterns. Actually, it is a useful pattern for designing functional configurations. In Sect. 4, gliders and some basic patterns will be used to implement reversible logic gates in T_{0347} .

Generally, it is not easy to follow evolving processes of configurations of T_{0347} by hand. So, we made a program for simulating them. In Morita (2016a) various examples of evolving processes can be seen by movies obtained by it. In addition, we created an emulator of T_{0347} on a general

purpose CA simulator *Golly* (Trevorrow et al. 2005), whose file is available in the Rule Table Repository of *Golly* or in Morita (2017).

3.1 Patterns in T_{0347}

There are three kinds of patterns in T_{0347} . They are a periodic pattern, a space-moving pattern, and an expanding pattern. Here, several examples of them are given.

3.1.1 Periodic pattern

A pattern is called a *periodic pattern* (or a *pattern of period p*), if the following holds: Starting from the configuration consisting of one copy of it, the same pattern appears at the same position after p time steps ($p > 0$). As a special case, a pattern of period 1 is called a *stable pattern*. It should be noted that in T_{0347} there is no “eventually periodic pattern” (i.e., a pattern that becomes periodic after one or more transient steps), since T_{0347} is reversible.

A *block* is a stable pattern shown in Fig. 5a or b. There are two kinds of blocks, i.e., type I (Fig. 5a) and type II (Fig. 5b). As we shall see in Sect. 3.2, moving direction of a glider can be controlled by placing several blocks of type I and II appropriately.

The pattern at $t = 0$ in Fig. 6 is a *reflected block* (i.e., a mirror image of a block). Unlike a block, it is not stable. In this case, the same pattern first appears at the same position after 8 steps, and thus it is of period 8.

A *fin* is a periodic pattern consisting of three particles shown in Fig. 7. It simply rotates clockwise with the period 6. A fin can also go around a block as in Fig. 8. It rotates around a block by 120° in 14 steps. Thus, the whole pattern consisting of a block and a fin is of period 42. Furthermore, a fin can travel around a sequence of blocks as in Fig. 9.

A *rotator* is a periodic pattern shown in Fig. 10. Like a fin, it rotates around some point. Since it rotates by 60° in 7 steps, its period is 42.

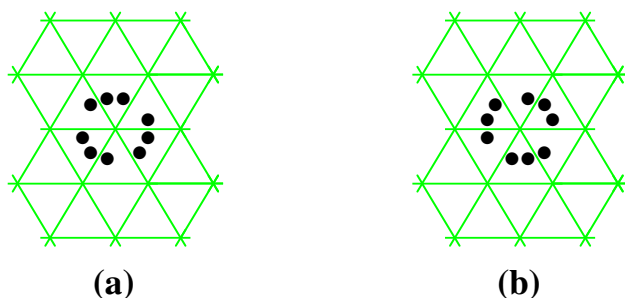


Fig. 5 Blocks of **a** type I and **b** type II. They are stable patterns

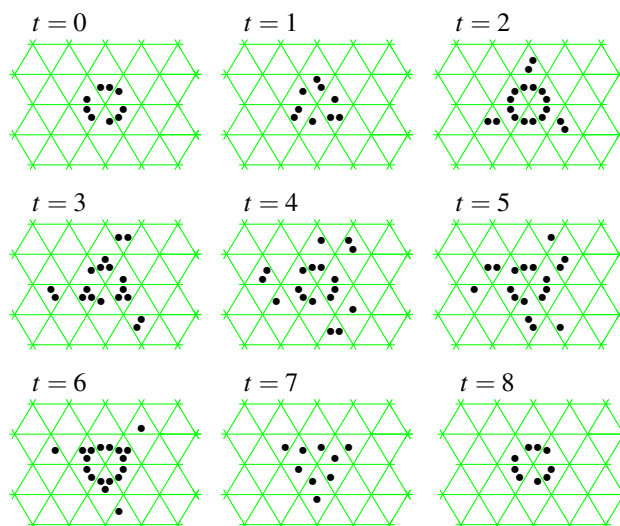


Fig. 6 The pattern at $t = 0$ is called a reflected block, which is a mirror image of a block. It is a pattern of period 8

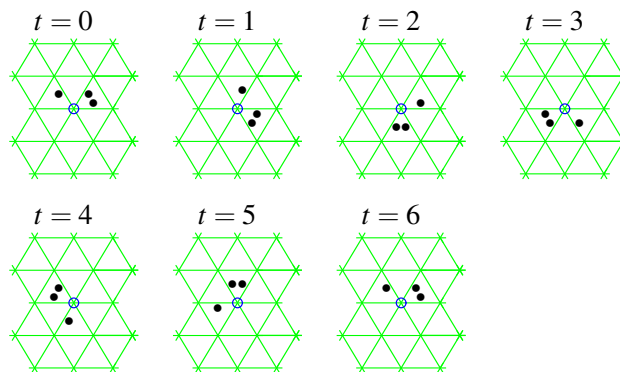


Fig. 7 A fin is a pattern of period 6. It rotates around the point indicated by \circ clockwise

3.1.2 Space-moving pattern

A *space-moving pattern* (or a *spaceship*) is one such that after some time steps p ($p > 0$) the same pattern (not rotated one) appears at a different position. Thus, it moves straight ahead in the cellular space of T_{0347} if no obstacle exists. The integer p is also called the *period* of the space-moving pattern.

A *glider* shown in Fig. 11 is a space-moving pattern. It swims in the cellular space like a fish (or an eel). It travels a unit distance, the side-length of a triangle, in 6 steps. Thus, its speed is $1/6$. The patterns in Fig. 11 at time $t = 0, \dots, 5$, and 6 are said to be of phase 0, \dots , 5, and 0, respectively. Hereafter, we regard the pattern of phase 0 as a *standard glider pattern*. By rotating it appropriately, it can move in any of the six directions. A glider will be used as a “signal” when we construct logic circuits in the cellular space of T_{0347} . So far, it is not known whether there is

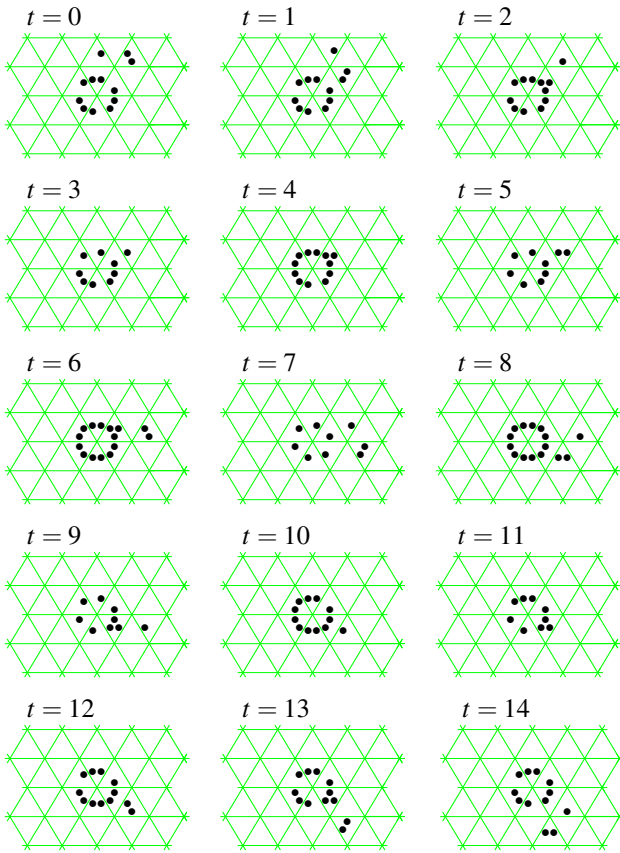


Fig. 8 A fin can travel around a block clockwise. It takes 42 steps to return to the initial position. Note that the block changes its pattern transiently, but it becomes the initial pattern again at $t = 14$

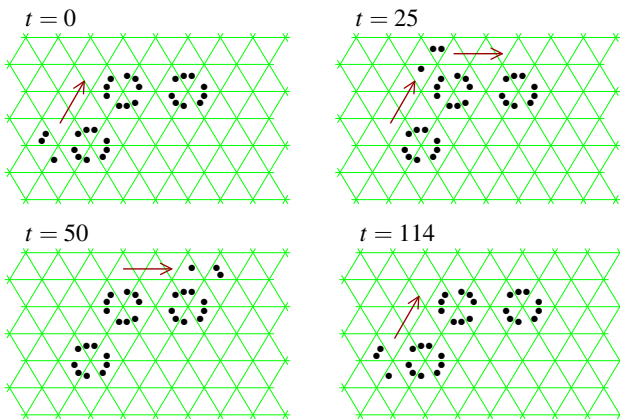


Fig. 9 A fin can also travel around a sequence of blocks

a space-moving pattern that is essentially different from a glider (here, “essentially different” means that it is not composed of two or more glider patterns).

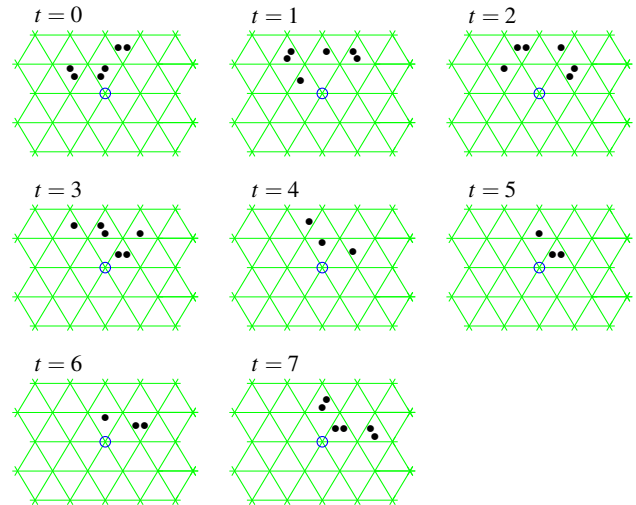


Fig. 10 A rotator is a pattern of period 42. It moves around the point indicated by \circ

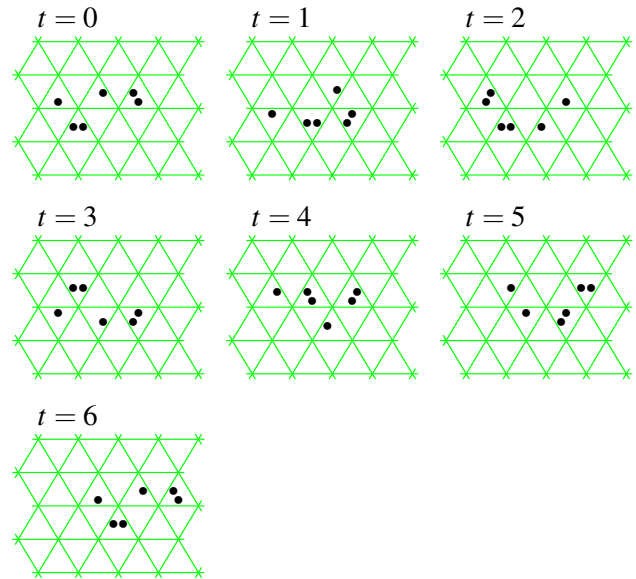


Fig. 11 A glider. It is a space-moving pattern of period 6

3.1.3 Expanding pattern

An *expanding pattern* (or more precisely, an *eventually expanding pattern*), is one such that the diameter of the pattern grows indefinitely as it evolves (though we do not give a definition of “diameter” here, it should be defined appropriately). Namely, for any integer $d_0 > 0$, there exists an integer $t_0 > 0$ such that the diameter of the pattern at time t_0 is greater than or equal to d_0 .

A pattern consisting of one particle is an example of an expanding pattern. If we start from it, a disordered pattern and gliders are generated, and the whole pattern grows bigger and bigger (Fig. 12).

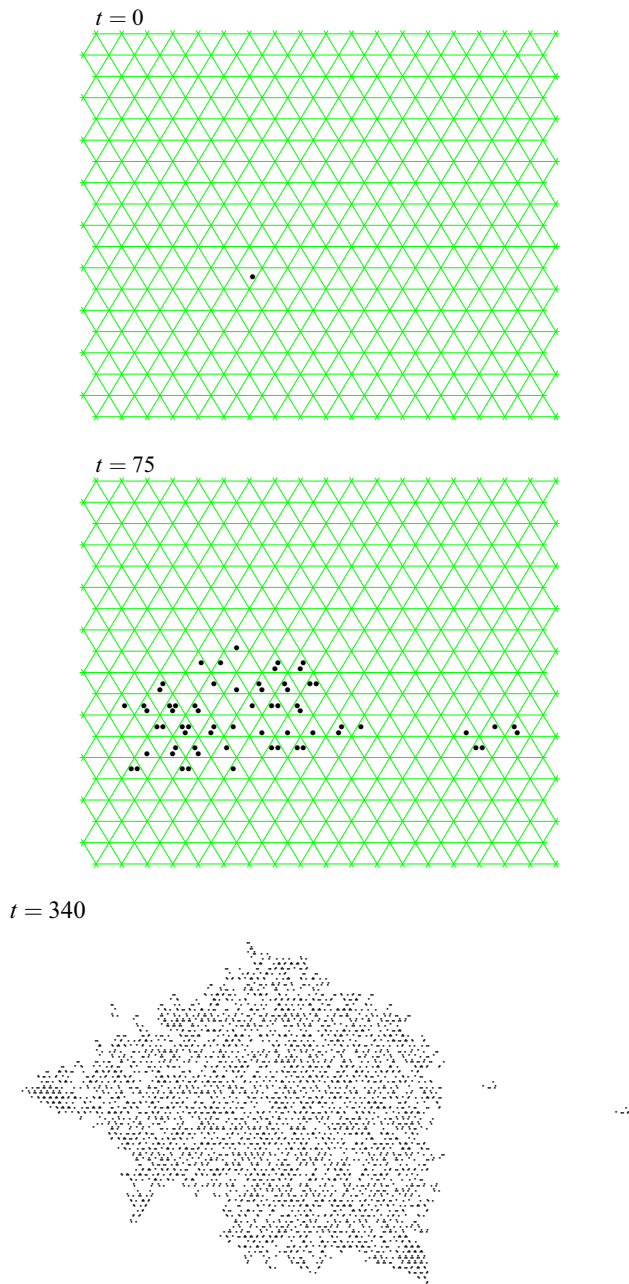


Fig. 12 Evolution process from the configuration consisting of a single particle ($t = 0$). The sizes of the patterns grow indefinitely as it evolves. Thus, every pattern in this evolution is an expanding pattern

It should be noted that any expanding pattern also expands to the negative time direction. Namely, if a pattern p at $t = 0$ is an expanding pattern, then for any $d_0 > 0$, there is a pattern p' at $t_0 < 0$ that becomes p at $t = 0$, and its diameter is greater than or equal to d_0 . This fact is explained as follows. At first, we can observe that for each pattern p_t at time t , the previous pattern p_{t-1} at $t - 1$ is easily obtained by reversely applying the local rules given in Fig. 3. Now, assume, on the contrary, there is an integer

d_{\max} such that the diameter of the pattern p_t at time t is less than d_{\max} for all $t < 0$. Then, the total number of different patterns that appear at $t < 0$ is finite. Hence, there are two integers $t_1 < t_2 < 0$ for which the patterns p_{t_1} and p_{t_2} are the same. Therefore, p_{t_1} is either a periodic pattern or a space-moving pattern, and not an expanding pattern.

Figure 13 shows that the one-particle pattern expands indefinitely to the negative time direction.

In the RETPCA T_{0347} , a disordered pattern like the one in Fig. 12 often appears, even if we compose a configuration out of periodic patterns and gliders. Therefore, when we want to give a configuration that performs some intended task, it should be designed so that it never generates a disordered pattern.

3.2 Controlling a glider by blocks

We now develop methods of controlling a glider, i.e., we investigate how we can change the move direction of a glider, and how we can adjust its timing. We first show several turn modules, such as left-turn and right-turn modules, by which the moving direction of a glider is changed. Then, we show a method of adjusting the delay time of a glider by them.

3.2.1 Backward-turn module

We can see that a single block acts as a backward-turn module. We place a glider that moves eastward, and a single block of type I as shown in Fig. 14 ($t = 0$). At $t = 12$ they collide. Then, the glider is split into a rotator and a fin. The rotator is, so to say, a “body” of the glider. Since the body has no fin, it cannot swim, and thus begins to rotate around the point indicated by \circ ($t = 16$). The fin travels around the block ($t = 38$). When it comes back to the original position, it interacts with the body ($t = 50$). By this, the rotation center of the body is shifted upward by two cells, and the fin travels around the block once more ($t = 61$). At $t = 94$, the body and the fin meet again. By this, the fin is attached to the body, and a glider is reconstructed. Finally, the glider goes westward ($t = 97$). By above, backward-turn of the glider is realized.

In the above process, if we use a type II block instead of a type I block, then the glider goes to the north-east direction, but the block cannot be re-used, since several garbage particles remain (Fig. 15). Therefore, an appropriate type of a block should be used depending on the coming direction of the glider. Figure 16 shows the allowed input positions of a glider in each type of a block.

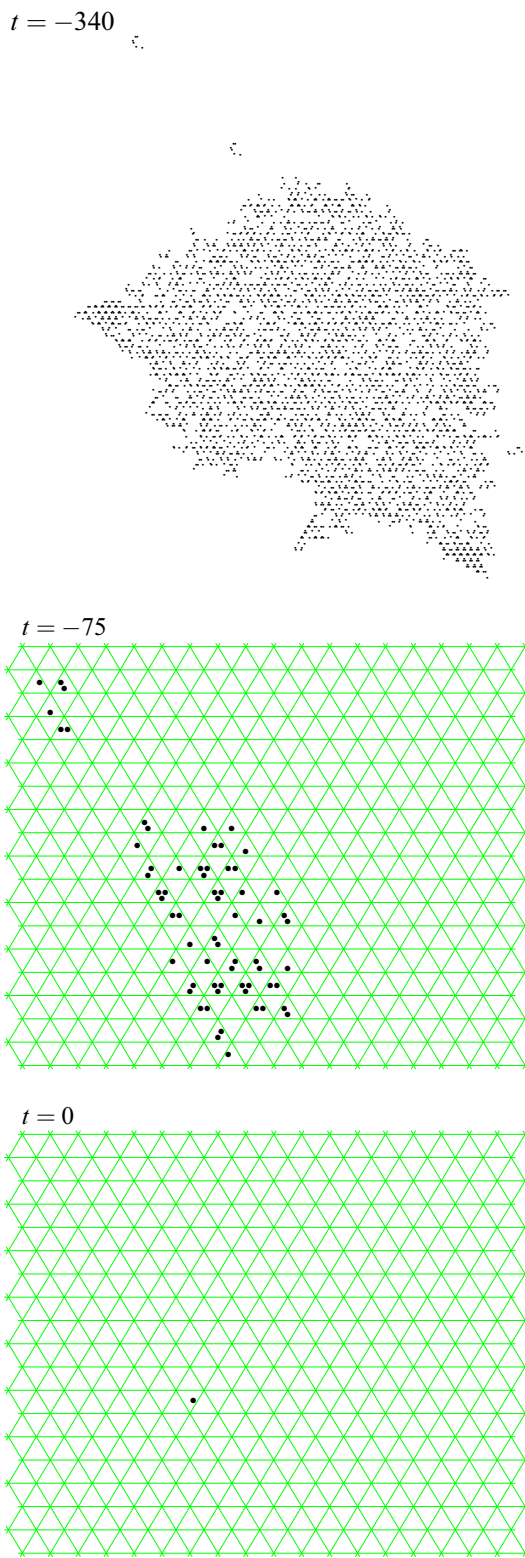


Fig. 13 The pattern at $t = -340$ shrinks to the single-particle pattern at $t = 0$. After that, it evolves as shown in Fig. 12. Hence, every pattern that appears at $t < 0$ is also an eventually expanding pattern, though it first shrinks to a single particle

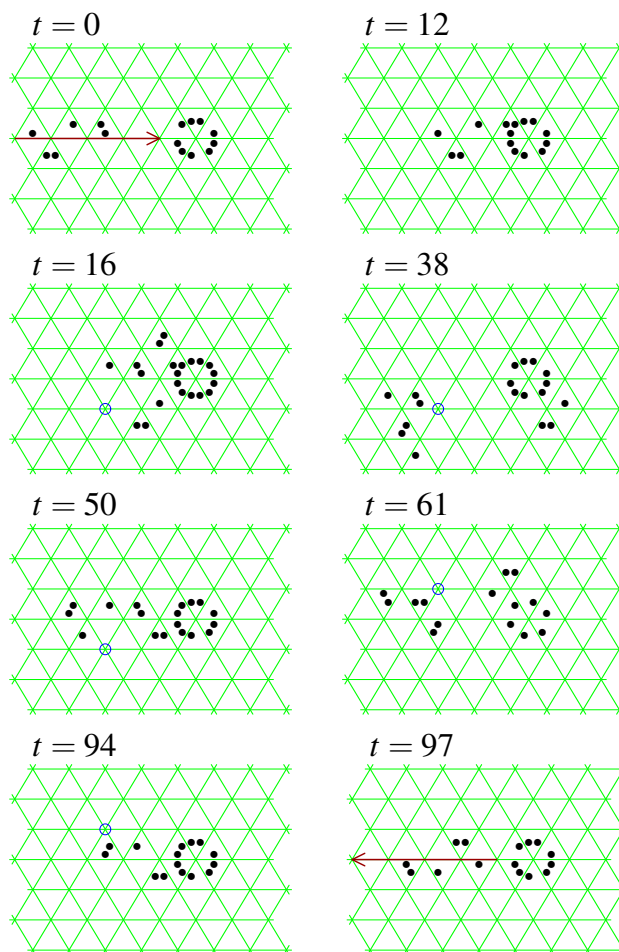


Fig. 14 Colliding a glider with a type I block. It works as a backward-turn module

3.2.2 Right-turn modules

Right-turn of a glider by 120° is realized by colliding a glider with a sequence of two blocks (Fig. 17). As in the case of one block (Fig. 14), the glider is first split into a rotator and a fin ($t = 56$). The fin travels around the blocks three times without interacting with the rotator. At the end of the fourth round, they meet to reconstruct a glider. Then, it goes to the south-west direction ($t = 334$). Hence, two blocks act as a 120° -right-turn module.

Figures 18 and 19 show that sequences of three blocks and five blocks also act as 120° -right-turn modules. They have shorter delays than the case of two blocks. Note that, as in Fig. 19, blocks need not be placed in a straight line. More precisely, the sequence of blocks can be bent by $\pm 60^\circ$ at each block. But, if we do so, the other type of a block must be used at the next position. Otherwise, blocks will be destroyed.

If we collide a glider with a sequence of four, six, or seven blocks, they will be destroyed. On the other hand, a

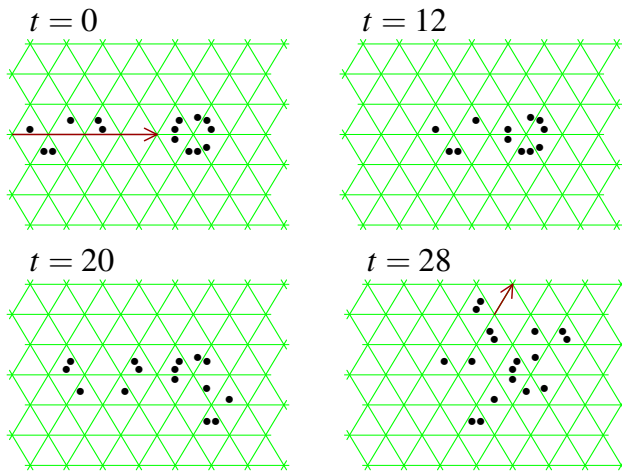


Fig. 15 If we collide a glider moving eastward with a type II block, garbage particles remain. Thus, it cannot be re-used as a turn module

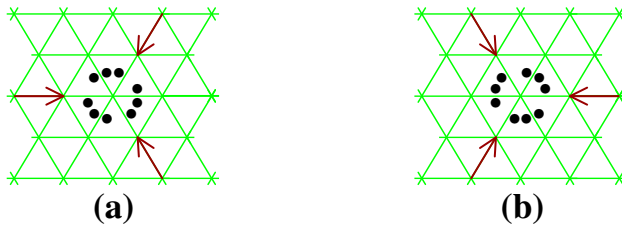


Fig. 16 Allowed input positions for **a** the type I block, and **b** the type II block

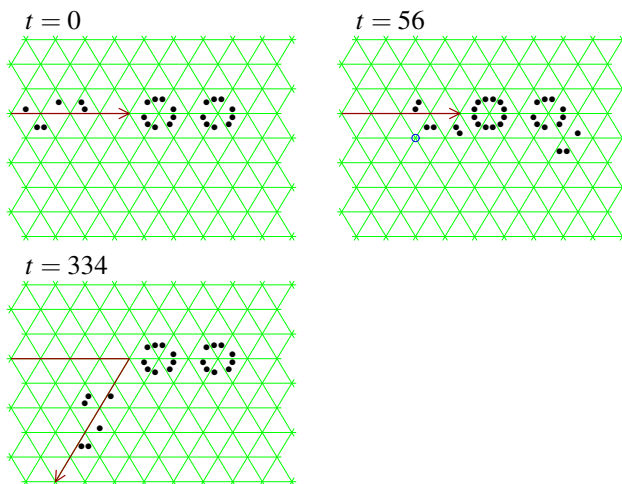


Fig. 17 120°-right-turn module composed of two blocks

sequence of eight blocks acts as a backward-turn module like one block, and that of nine blocks acts as a right-turn module like two blocks. Generally, a sequence of $n + 7$ blocks ($n > 0$) shows a similar behavior as that of n blocks though the total delay is longer. The reason is as follows. The period that the fin goes around the $n + 7$ blocks is $36(n + 7) + 6$. Thus, when the fin comes back to the initial

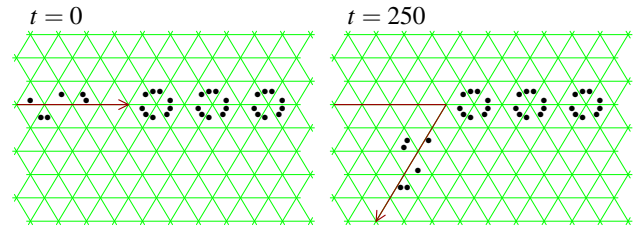


Fig. 18 120°-right-turn module composed of three blocks

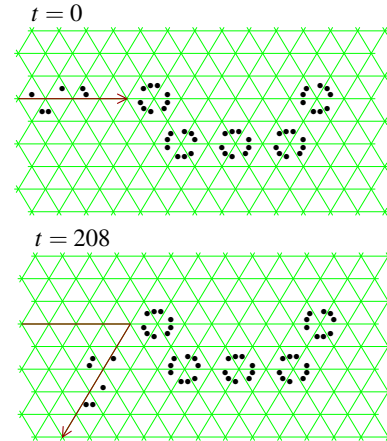


Fig. 19 120°-right-turn module composed of five blocks, which are not linearly placed

position, the phase of the rotator becomes the same as in the case of n blocks, since the period of a rotator is 42, and 36×7 is a multiple of 42.

It is also possible to make a 60°-right-turn module as in Fig. 20. It is composed of a 120°-right-turn module with three blocks and a backward-turn module consisting of one block. First, the input glider makes 120°-right-turn by the three blocks ($t = 250$). Next, the glider is reflected by the backward-turn module ($t = 341$). Then, it makes 120°-right-turn by the three blocks again ($t = 585$). Thus, the whole pattern acts as a 60°-right-turn module. Note that, by replacing the three blocks by two or five blocks, we obtain a 60°-right-turn module having a different delay time. We can also adjust the total delay time by changing the distance between the backward-turn module and the 120°-right-turn module. By this, the travelling distance of a glider is changed.

This mechanism can be used as an interface between a bidirectional signal path and unidirectional signal paths as shown in Fig. 21. A *bidirectional signal path* is one on which a glider, which is regarded as a signal, moves in both directions. Sometimes we have to use such a signal path. For example, as shown in Fig. 28, three gliders are generated by a head-on collision of two gliders, and one of the three gliders goes backward on one of the input paths.

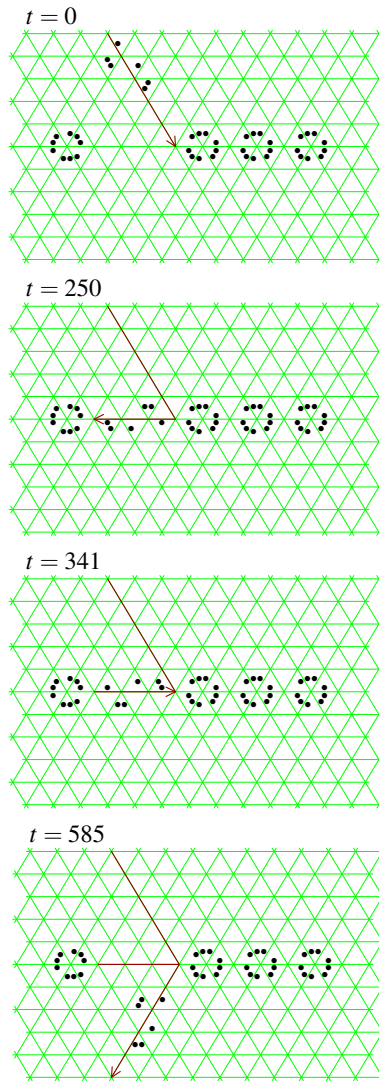


Fig. 20 60°-right-turn module composed of a 120°-right-turn module and a backward-turn module

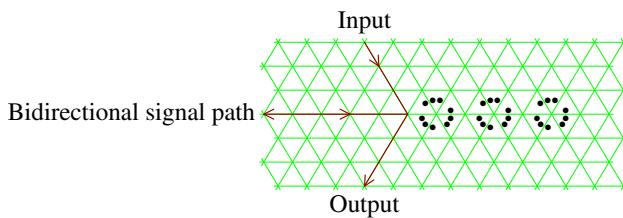


Fig. 21 Interface between a bidirectional signal path, and unidirectional signal paths

Using the structure shown in Fig. 21, the input and the output to/from a bidirectional path are split.

3.2.3 U-turn module

A U-turn module is given in Fig. 22. Also in this case, the glider is first split into a rotator and a fin ($t = 36$). But,

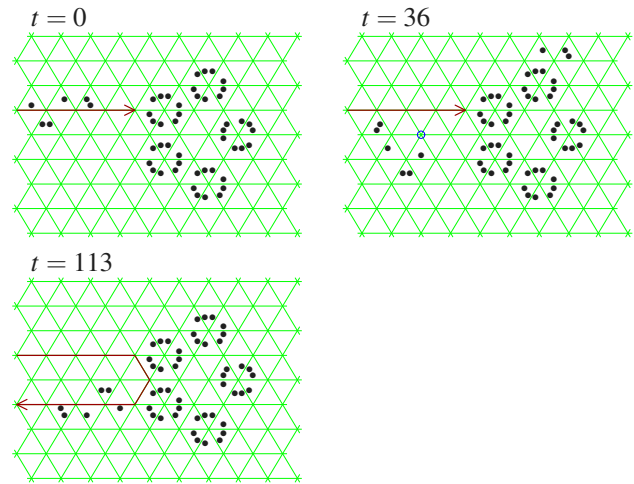


Fig. 22 U-turn module

slightly before the fin comes back to the start position, it meets the rotator, and a glider is reconstructed, which moves westward ($t = 113$). Note that, here the output path is different from the input path, while in the backward-turn module they are the same (Fig. 14).

3.2.4 Left-turn module

Figure 23 shows a 120°-left-turn module. It is more sophisticated than the right-turn and U-turn modules. The glider is split into a rotator and a fin as before ($t = 78$). The fin first travels outside of the module, and then inside. But, around the middle of the module it meets the rotator. A glider is reconstructed from them, and it moves to the north-west direction ($t = 366$).

3.2.5 Adjustment of the delay time of a glider

Using the turn modules given above, we can also adjust the delay of a glider. We first calculate the delay and the phase shift caused by each turn module.

The *net delay* d of a turn module is the additional delay caused by the module. For example, consider the case of 120°-right-turn module with five blocks (Fig. 19). We can regard the travelling distance of the glider (along the arrow line) from $t = 0$ to 208 is 5 [note that the glider patterns at $t = 0$ and 208 are both standard ones (i.e., of phase 0)]. If a glider travels straight the distance of 5, then it takes $5/(1/6) = 30$ steps, since the speed of a glider is $1/6$. Therefore, the net delay for this module is $d = 208 - 30 = 178$.

The *phase shift* s of a turn module is the shift value of the phase of a glider by the module, and calculated by the relation $s = (-d) \bmod 6$. In the case of 120°-right-turn module with five blocks, the phase shift s is $(-178) \bmod 6 = 2$.

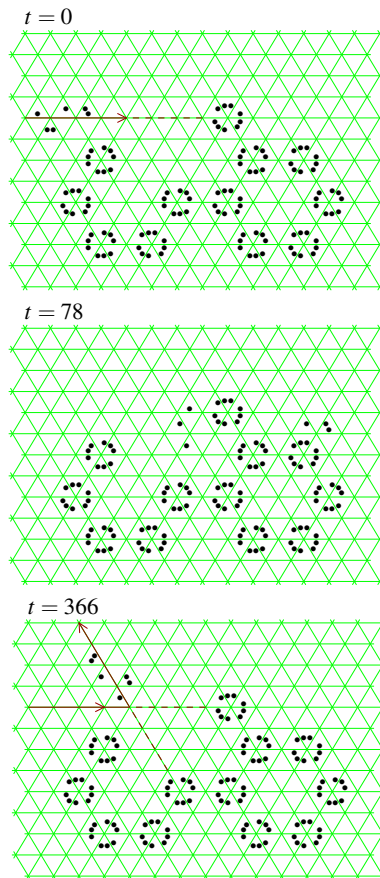


Fig. 23 120°-left-turn module

Table 1 shows the net delay and the phase shift of each turn module. Assume a pattern composed of turn modules is given. Then, its total delay is easily calculated by Table 1. Consider the 60°-right-turn module in Fig. 20. It is composed of a 120°-right-turn module with three blocks and a backward-turn module. Therefore, its net delay is $220 \times 2 + 73 = 513$, and its phase shift is $(-513) \bmod 6 = 3$ (note that the 120°-right-turn module in this pattern is used twice).

We now explain a method of adjusting the delay of a glider. Since the direction of a glider is freely controlled by

Table 1 Net delay and phase shift of the six turn modules

Module	Net delay	Phase shift
Backward-turn	73	+ 5
Right-turn (120°) by 2 blocks	304	+ 2
Right-turn (120°) by 3 blocks	220	+ 2
Right-turn (120°) by 5 blocks	178	+ 2
U-turn	77	+ 1
Left-turn (120°)	342	0

the turn modules given above, we hereafter consider the timing problem only for east-moving gliders.

At first, we observe that an extra delay of 6 steps can be realized by modifying the signal path using 120°-right-turn modules appropriately. For example, consider the signal path shown in Fig. 24a. Note that, timing adjustment of a glider is required when interacting it with another glider or with a periodic pattern. Therefore, we can assume the given pattern for a signal path has several 120°-right-turn modules (if otherwise, insert three turn modules as in Fig. 24a). An extra delay of 6 steps is achieved by extending the length of the signal path by 15, and replacing two occurrences of three-block sequences by five-block sequences as in Fig. 24b. By these modifications, we have an extra delay of $15/(1/6) - (220 - 178) \times 2 = 6$ steps. A delay of $6n$ steps ($n = 2, 3, \dots$) is also realized in a similar manner.

It is also possible to shift the phase of a glider (i.e., to adjust the timing by a few steps that is less than 6). Note that the phase shift of an odd number is impossible if a glider moves in the same direction (say, eastward). This is because the standard glider pattern of east-moving one always occupies up-triangle cells. Therefore, it is sufficient to show that the phase shift of 2 steps is possible (the phase shift of 4 steps is realized by repeating the procedure). Consider the signal paths shown in Fig. 25. There are three 120°-right-turn modules in Fig. 25a. Therefore, the total phase shift value is $(2 + 2 + 2) \bmod 6 = 0$. In Fig. 25b, the total phase shift is $2 + 0 = 2$, since there are one 120°-right-turn module and one 120°-left-turn module. Thus, phase shift of + 2 relative to (a) is realized in (b).

By above, it is guaranteed that adjustment of timing of gliders is always possible. But, in the following

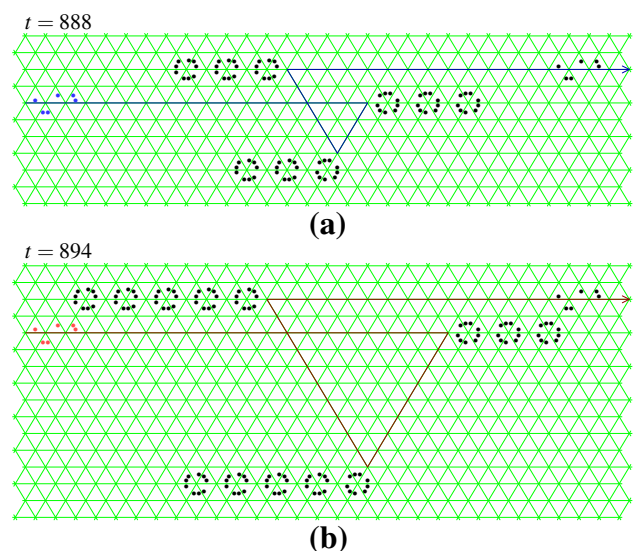


Fig. 24 Adjustment of the delay. In **b** the delay of the glider is larger than that in **a** by 6 steps

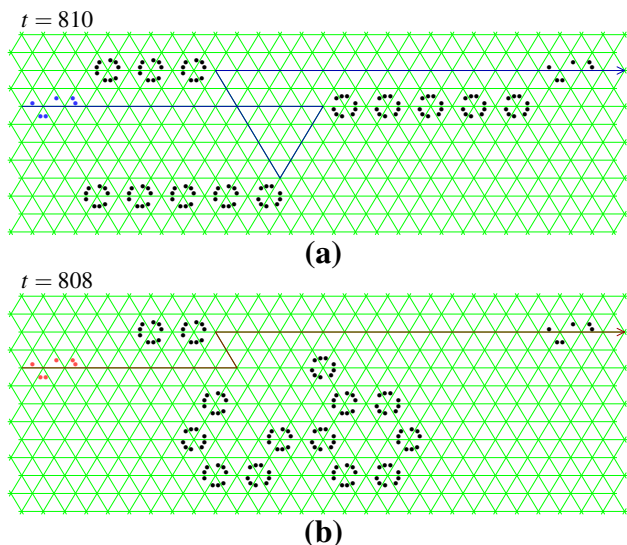


Fig. 25 Adjustment of the phase. In **b** the phase of a glider is shifted by 2 relative to that in **a**

constructions of larger patterns, ad hoc methods for reducing the sizes of the patterns are employed.

3.3 Glider guns in T_{0347}

A *glider gun* is a pattern that generates gliders periodically. It is well known that there is a glider gun in the Game-of-Life CA (Gardner 1971). We show there are two types of glider guns in the RETPCA T_{0347} . They are three-way and one-way glider guns.

In T_{0347} , it is easy to create a *three-way glider gun*. As shown in Fig. 26, it is obtained by colliding a glider with a fin. Interestingly, there is a *three-way glider absorber* in T_{0347} (Fig. 27). It absorbs three gliders every 24 steps, and if the input gliders run out, it finally produces a fin and a glider. Thus, it is considered as a “backward glider gun” that generates gliders to the negative time direction. Combining the gun and the absorber, it is also possible to create a three-way glider gun that generates gliders in both positive and negative time directions (Morita 2016a).

Next, we construct a *one-way glider gun*. We first observe that three gliders are generated by the head-on collision of two gliders as in Fig. 28. Based on this mechanism, we can design a one-way glider gun. Namely, two of the three generated gliders are circulated and reused to generate the next three. Figure 29 shows such a glider gun, where a glider is generated every 1422 steps

It is also possible to compose a *one-way glider absorber*. Symmetrically to the process of Fig. 28, two gliders are obtained by the collision of three gliders, i.e., one of the three gliders is reversibly erased as shown in Fig. 30. Based on this, we can construct a one-way glider absorber

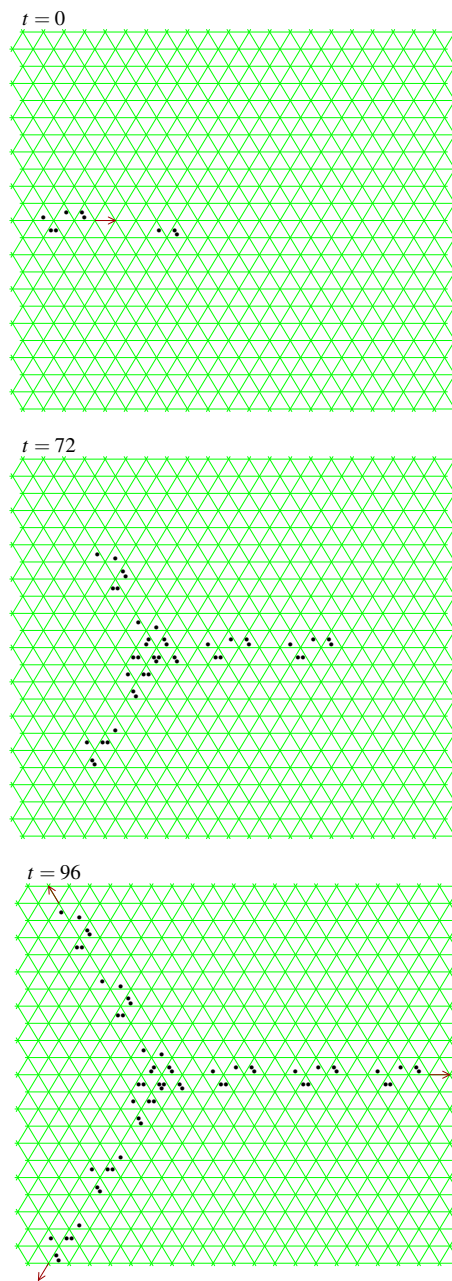


Fig. 26 Three-way glider gun. It generates three gliders every 24 steps

given in Fig. 31. This pattern is a “quasi-mirror-image” of the one-way glider gun pattern (except the gliders). The positions of the blocks are just the mirror images of those in the glider gun. But, each block is replaced by the other type of the corresponding block (note that the mirror image of a block is not stable as shown in Fig. 6). To this one-way glider absorber, an infinite number of gliders must be given at the right timing (if the input gliders run out, this pattern will be destroyed, and a disordered pattern will expand).

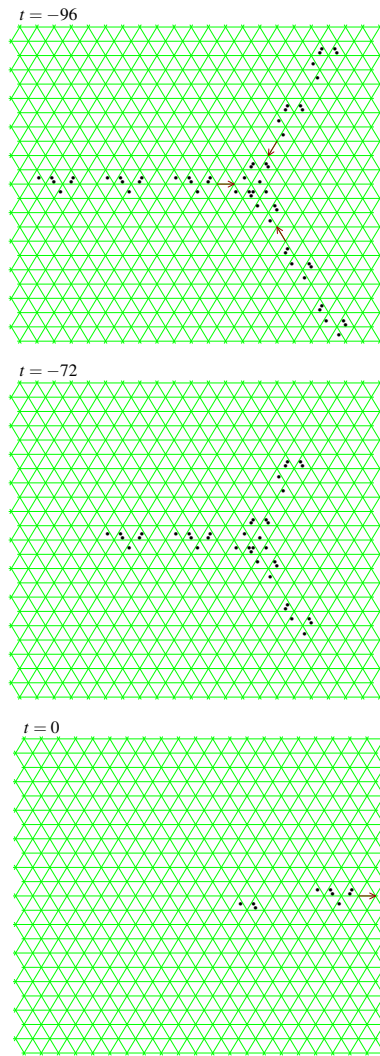


Fig. 27 Glider absorber. It is considered as a glider gun to the negative time direction

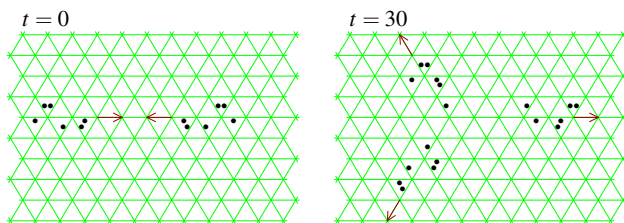


Fig. 28 Generating three gliders by the head-on collision of two gliders

Connecting a one-way glider gun and an absorber appropriately, we have a pattern where an infinite stream of gliders generated by the gun is absorbed by the absorber. Figure 32 is such a pattern of period 1422.

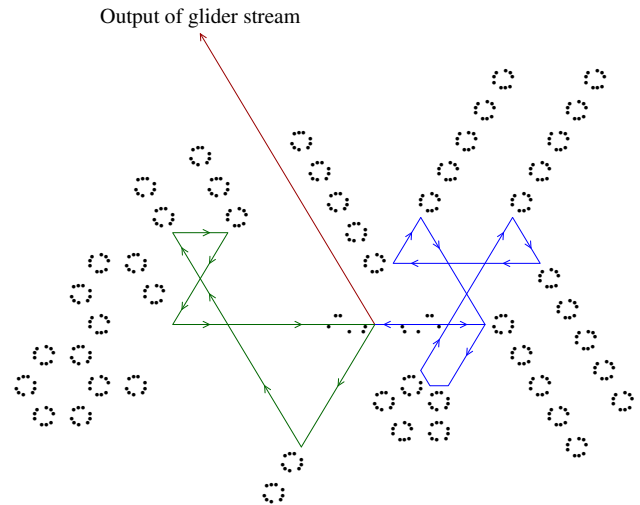


Fig. 29 One-way glider gun. It generates a glider every 1422 steps

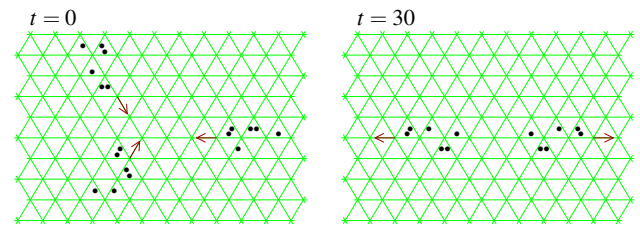


Fig. 30 Generating two gliders by the collision of three gliders

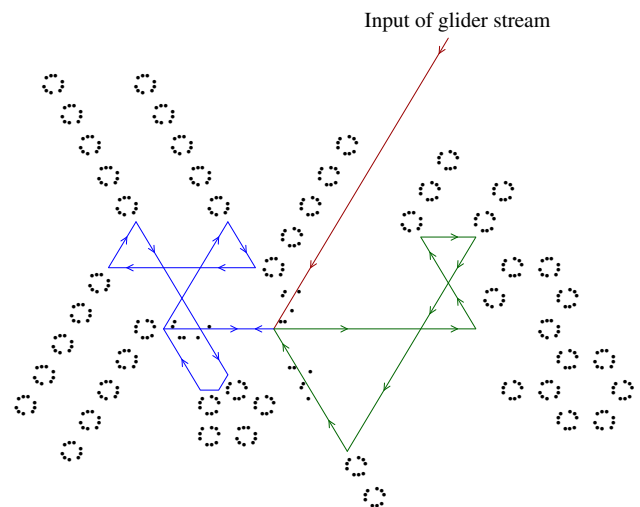


Fig. 31 One-way glider absorber. It absorbs a glider every 1422 steps

4 RETPCA T_{0347} is computationally universal

In this section, we show Turing universality of T_{0347} , i.e., any Turing machine can be simulated in it.

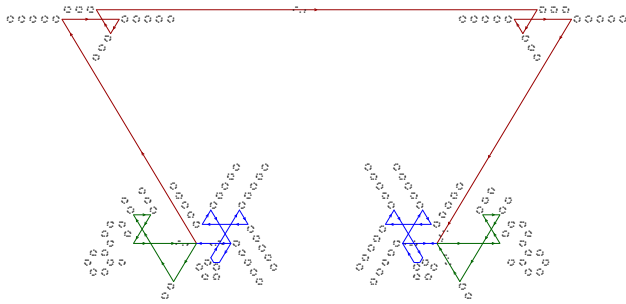


Fig. 32 One-way glider gun and absorber. An infinite glider stream generated by the gun (left) is absorbed by the absorber (right)

4.1 Showing Turing universality of a reversible CA

To prove Turing universality of a reversible CA, it is sufficient to show that any reversible logic circuit composed of switch gates (Fig. 33a), inverse switch gates (Fig. 33b), and delay elements can be simulated in it (Lemma 6).

Lemma 6 can be derived, e.g., in the following way. First, a Fredkin gate (Fig. 34) can be constructed out of two switch gates and two inverse switch gates (Lemma 2). Second, any reversible sequential machine (RSM), in particular, a rotary element (RE), which is a 2-state 4-symbol RSM, is composed only of Fredkin gates and delay elements (Lemma 3). Note that an RE is a special type of a reversible logic element with memory (RLEM) (Morita et al. 2012). Third, any reversible Turing machine is constructed out of REs (Lemma 4). Finally, any (irreversible) Turing machine is simulated by reversible one (Lemma 5). Thus, Lemma 6 follows. Note that the circuit that realizes a reversible Turing machine constructed by this method becomes an infinite (but ultimately periodic) circuit.

Lemma 2 (Fredkin and Toffoli 1982) *A Fredkin gate can be simulated by a circuit composed of switch gates and inverse switch gates, which produces no garbage signals.*

Lemma 3 (Morita 1990) *Any RSM (in particular RE) can be simulated by a circuit composed of Fredkin gates and delay elements, which produces no garbage signals.*

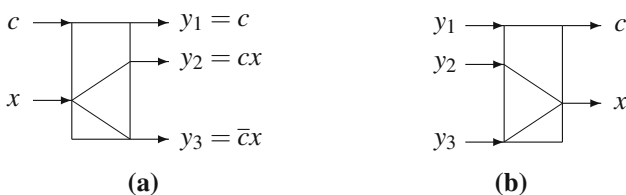


Fig. 33 **a** Switch gate. **b** Inverse switch gate, where $c = y_1$ and $x = y_2 + y_3$ under the assumption $(y_2 \rightarrow y_1) \wedge (y_3 \rightarrow \bar{y}_1)$

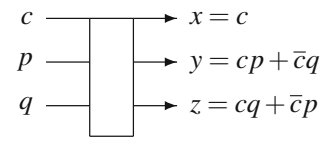


Fig. 34 Fredkin gate

Lemma 4 (Morita 2001) *Any reversible Turing machine can be simulated by a garbage-less circuit composed only of REs.*

Lemma 5 (Bennett 1973) *Any (generally irreversible) Turing machine can be simulated by a garbage-less reversible Turing machine.*

Lemma 6 *A reversible CA is Turing universal, if any circuit composed of switch gates, inverse switch gates, and delay elements is simulated in it.*

So far, Turing universality of several kinds of reversible two-dimensional CAs has been shown in this way. They are the 2-state reversible block CA model by Margolus (1984), the two models of 16-state reversible PCAs on square grid by Morita and Ueno (1992), the conservative RETPCA T_{0157} by Imai and Morita (2000), and the conservative RETPCA T_{0137} by Morita (2016b).

4.2 Making switch gate and inverse switch gate in T_{0347}

We show a switch gate and an inverse switch gate can be implemented in T_{0347} using gliders as signals. The operation of a switch gate is realized by colliding two gliders as shown in Fig. 35. It is important that, in this collision, the glider from the input port c travels to the south-east direction with no delay even though it interacts with the glider from x (hence its phase is not shifted also).

Here, we implement a switch gate as a “gate module” in the standard form. Otherwise, adjustment of signal timing, in particular, adjustment of the phase of a glider becomes very cumbersome when designing a larger circuit. A gate module is a pattern embedded in a rectangular-like region in the cellular space that satisfies the following (Fig. 36): (1) It realizes a reversible logic gate. (2) Input ports are at the left end. (3) Output ports are at the right end. (4) Delay between input and output is constant and a multiple of 6.

Figure 37 shows a switch gate module. A switch gate operation (Fig. 35) is performed around the center of this pattern. The delay between input and output of this module is 2232 steps, a multiple of 6, and thus satisfies the condition (4) above.

Figure 38 shows the inverse switch gate operation in T_{0347} . The process of collision of two gliders is just the same as in Fig. 35. Only the positions of the input/output

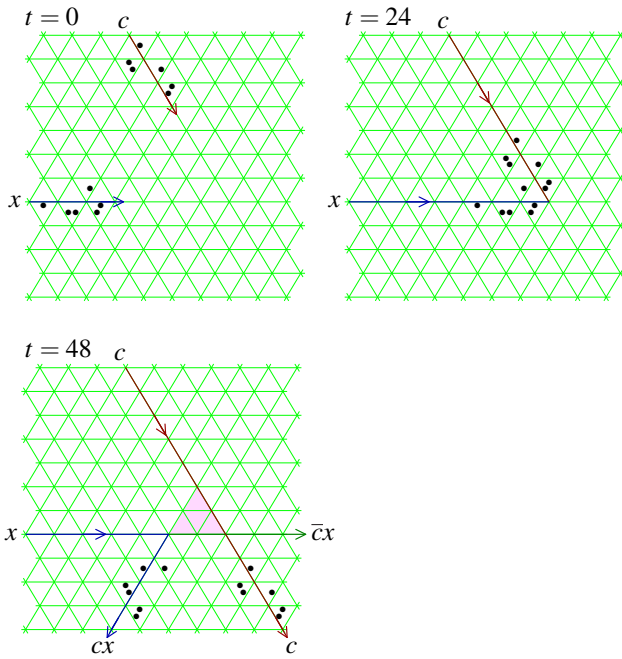


Fig. 35 Switch gate operation realized by collision of two gliders

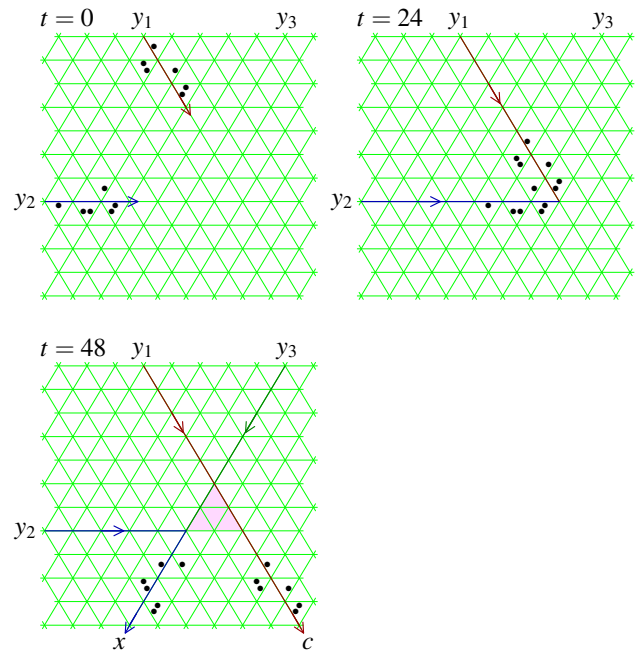


Fig. 38 Inverse switch gate operation realized by collision of two gliders

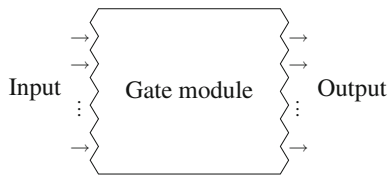


Fig. 36 Gate module in the standard form

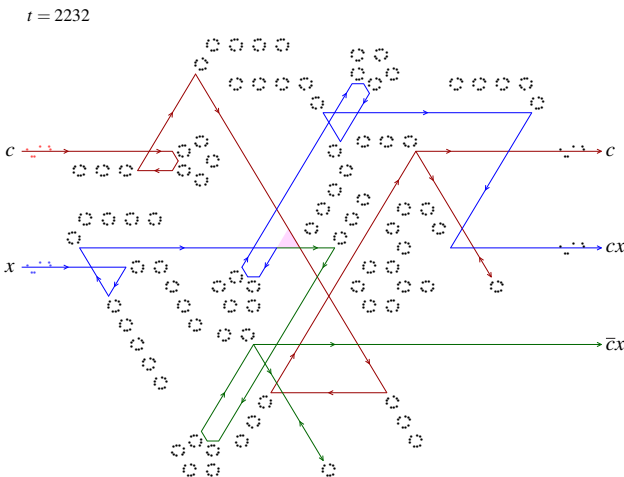


Fig. 37 Switch gate module implemented in T_{0347}

ports are different. An inverse switch gate module is given in Fig. 39. It is a quasi-mirror-image of the switch gate module (as in the case of a one-way glider gun and an absorber). The delay between input and output is also 2232 steps.

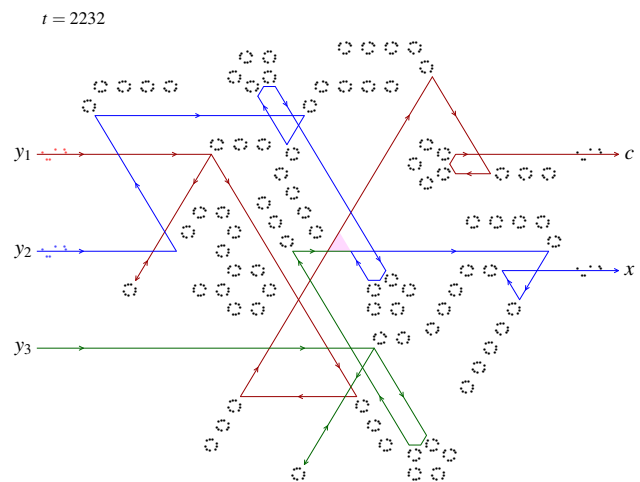


Fig. 39 Inverse switch gate module implemented in T_{0347} . Here, $c = y_1$ and $x = y_2 + y_3$ under the assumption $(y_2 \rightarrow y_1) \wedge (y_3 \rightarrow \bar{y}_1)$

4.3 Making larger circuits in T_{0347}

If we use only 120° -right-turn modules to connect gate modules, then there is no need of adjusting the phases of gliders. This is because the total phase shift becomes 0, if we make 120° -right-turns three times (see Table 1). Therefore, we have to adjust the delay by $6n$ ($n = 1, 2, \dots$) steps, and this can be done by a similar manner as in Fig. 24. In this way, we can construct a larger circuit easily.

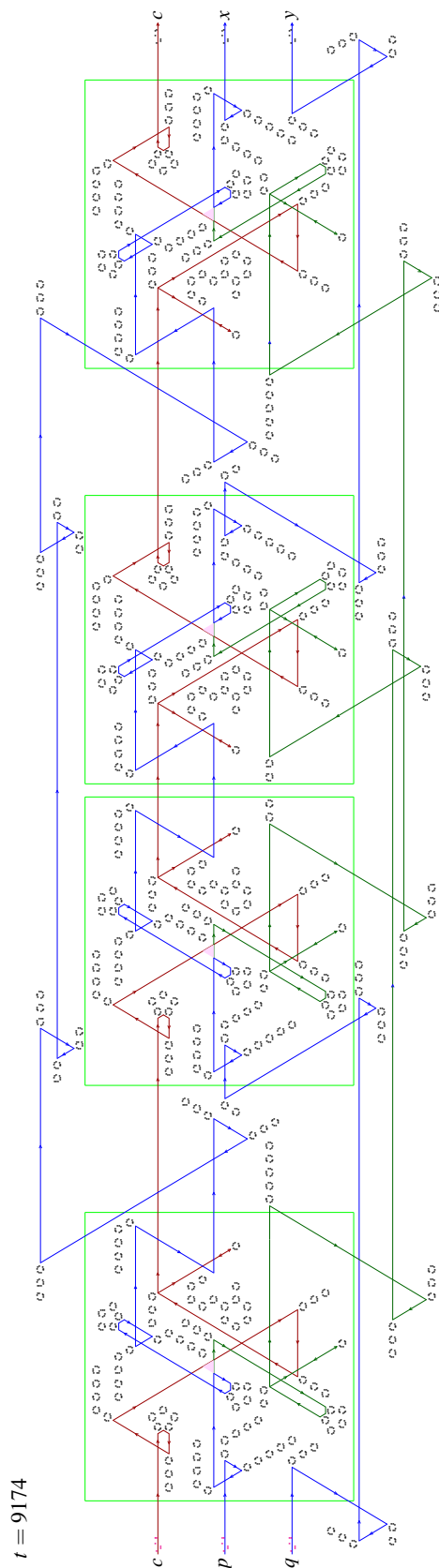


Fig. 40 Fredkin gate module implemented in T_{0347} . Here, $x = cp + \bar{c}q$ and $y = cq + \bar{c}p$

Figure 40 shows a Fredkin gate module in T_{0347} , which is composed of two switch gate modules (left) and two inverse switch gate modules (right). Here, the switch gate and inverse switch gate modules are connected using only 120° -right-turn modules. The total delay between input and output of the Fredkin gate module is 9174, which is again a multiple of 6.

In a similar manner, we can construct any circuit composed of Fredkin gates in the cellular space of T_{0347} . Thus, we have the following theorem.

Theorem 1 The RETPCA T_{0347} with infinite (but ultimately periodic) configurations is Turing universal.

5 Concluding remarks

Among 256 ETPCAs, we investigated a specific ETPCA T_{0347} , which is non-conservative and reversible. In spite of its extreme simplicity of the local function and the constraint of reversibility, T_{0347} shows interesting behavior. Here, a glider plays the key role in T_{0347} . By placing blocks appropriately, trajectory and the timing of a glider can be completely controlled. Logical operation is also performed by interacting gliders. Using such properties and operations, the Fredkin gate is realized in this cellular space. In this way, Turing universality of T_{0347} with infinite configurations was proved.

There is another method of constructing reversible Turing machines (RTMs) in the space of T_{0347} . This method is to use a specific reversible logic element with memory (RLEM) No. 4-31 as a basic element (Morita and Suyama 2014). It is known that RLEM 4-31 can be implemented in T_{0347} directly (i.e., without using reversible logic gates). By this, we can design a configuration of T_{0347} that simulates a given RTM rather simply. Files of such configurations are available in Morita (2017), by which we can see the whole computing processes of example RTMs on Golly.

On the other hand, it is not known whether universal systems are simulated in the finite configurations of T_{0347} . In Morita et al. (2002), it has been shown that any reversible two-counter machine, a Turing universal model, can be realized as a finite configuration of the 3^4 -state reversible 4-neighbor PCA on the square grid. There, a counter is implemented by a “position marker” whose position can be shifted (i.e., pushed or pulled) by colliding a signal with it. However, in T_{0347} such phenomena have not yet been found.

It is also not known whether there is a universal constructor in T_{0347} , which can build any pattern in some specified class of patterns (e.g., the class of all patterns

consisting of blocks) as in the case of the Game-of-Life CA (Berlekamp et al. 1982).

Besides T_{0347} , it has already been shown that the conservative RETPCAs T_{0157} and T_{0137} with infinite configurations are Turing universal (Imai and Morita 2000; Morita 2016b), where a single particle rather than a glider is used to represent a signal. It is left for the future study to find other ETPCAs that are universal.

Acknowledgements This work was supported by JSPS KAKENHI Grant Number JP15K00019.

References

- Bays C (1994) Cellular automata in the triangular tessellation. *Complex Syst* 8:127–150
- Bennett CH (1973) Logical reversibility of computation. *IBM J Res Dev* 17:525–532. <https://doi.org/10.1147/rd.176.0525>
- Berlekamp E, Conway J, Guy R (1982) *Winning ways for your mathematical plays*, vol 2. Academic Press, New York
- Fredkin E, Toffoli T (1982) Conservative logic. *Int J Theoret Phys* 21:219–253. <https://doi.org/10.1007/BF01857727>
- Gajardo A, Goles E (2001) Universal cellular automaton over a hexagonal tiling with 3 states. *Int J Algebra Comput* 11:335–354. <https://doi.org/10.1142/S0218196701000486>
- Gardner M (1970) Mathematical games: the fantastic combinations of John Conway's new solitaire game "life". *Sci Am* 223(4): 120–123. <https://doi.org/10.1038/scientificamerican1070-120>
- Gardner M (1971) Mathematical games: on cellular automata, self-reproduction, the Garden of Eden and the game "life". *Sci Am* 224(2):112–117. <https://doi.org/10.1038/scientificamerican0271-112>
- Imai K, Morita K (2000) A computation-universal two-dimensional 8-state triangular reversible cellular automaton. *Theoret Comput Sci* 231:181–191. [https://doi.org/10.1016/S0304-3975\(99\)00099-7](https://doi.org/10.1016/S0304-3975(99)00099-7)
- Margolus N (1984) Physics-like model of computation. *Physica D* 10:81–95. [https://doi.org/10.1016/0167-2789\(84\)90252-5](https://doi.org/10.1016/0167-2789(84)90252-5)
- Morita K (1990) A simple construction method of a reversible finite automaton out of Fredkin gates, and its related problem. *Trans IEICE Jpn E-73*:978–984
- Morita K (2001) A simple reversible logic element and cellular automata for reversible computing. In: Margenstern M, Rogozhin Y (eds) *MCU 2001*, LNCS 2055, pp 102–113. https://doi.org/10.1007/3-540-45132-3_6
- Morita K (2016a) A reversible elementary triangular partitioned cellular automaton that exhibits complex behavior (slides with simulation movies). Hiroshima University Institutional Repository. <http://ir.lib.hiroshima-u.ac.jp/00039321>
- Morita K (2016b) Universality of 8-state reversible and conservative triangular partitioned cellular automata. In: El Yacoubi S et al (eds) *ACRI 2016*, LNCS 9863, pp 45–54. https://doi.org/10.1007/978-3-319-44365-2_5
- Morita K (2017) Reversible world: data set for simulating a reversible elementary triangular partitioned cellular automaton on Golly. Hiroshima University Institutional Repository. <http://ir.lib.hiroshima-u.ac.jp/00042655>
- Morita K, Harao M (1989) Computation universality of one-dimensional reversible (injective) cellular automata. *Trans IEICE Jpn E72*:758–762
- Morita K, Suyama R (2014) Compact realization of reversible Turing machines by 2-state reversible logic elements. In: Ibarra OH, Kari L, Kopecki S (eds) *Proc. UCNC 2014*, LNCS 8553, pp 280–292. https://doi.org/10.1007/978-3-319-08123-6_23, Slides with figures of computer simulation: Hiroshima University Institutional Repository. <http://ir.lib.hiroshima-u.ac.jp/00036076>
- Morita K, Ueno S (1992) Computation-universal models of two-dimensional 16-state reversible cellular automata. *IEICE Trans Inf Syst E75-D*:141–147
- Morita K, Tojima Y, Imai K, Ogiro T (2002) Universal computing in reversible and number-conserving two-dimensional cellular spaces. In: Adamatzky A (ed) *Collision-based computing*, Springer, pp 161–199. https://doi.org/10.1007/978-1-4471-0129-1_7
- Morita K, Ogiro T, Alhazov A, Tanizawa T (2012) Non-degenerate 2-state reversible logic elements with three or more symbols are all universal. *J Multiple-Valued Logic Soft Comput* 18:37–54
- Róka Z (1999) Simulations between cellular automata on Cayley graphs. *Theoret Comput Sci* 225:81–111. [https://doi.org/10.1016/S0304-3975\(97\)00213-2](https://doi.org/10.1016/S0304-3975(97)00213-2)
- Trevorrow A, Rokicki T, et al (2005) Golly: an open source, cross-platform application for exploring Conway's Game of Life and other cellular automata. <http://golly.sourceforge.net/>. A file for emulating T_{0347} (*Reversible_World_2.zip*) on Golly is available at the Rule Table Repository. <http://github.com/GollyGang/ruletablerepository>
- Wolfram S (1986) *Theory and applications of cellular automata*. World Scientific Publishing, Singapore
- Wolfram S (2002) *A new kind of science*. Wolfram Media Inc, Champaign