An 8-State Simple Reversible Triangular Cellular Automaton that Exhibits Complex Behavior

Kenichi Morita(✉)

Hiroshima University, Higashi-hiroshima 739-8527, Japan
km@hiroshima-u.ac.jp

Abstract. A three-neighbor triangular partitioned cellular automaton (TPCA) is a CA whose cell is triangular-shaped and divided into three parts. The next state of a cell is determined by the three adjacent parts of its neighbor cells. The framework of TPCA makes it easy to design reversible triangular CAs. Among them, isotropic 8-state (i.e., each part has two states) TPCAs, which are called elementary TPCAs (ETPCAs), 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, and look like those in the Game-of-Life CA to some extent. In particular, a "glider" and "glider guns" exist in T_{0347}. Furthermore, using gliders to represent signals, we can implement universal reversible logic gates in it. By this, computational universality of T_{0347} is derived.

1 Introduction

A three-neighbor triangular cellular automaton (TCA) is a one whose cell is regarded as being triangular-shaped, and communicates with its three neighbor cells. Here, we use the framework of triangular partitioned cellular automata (TPCAs) [6], where each cell is divided into three parts, and each part has its own state set. TPCAs are a subclass of TCAs where the state set of a cell is the Cartesian product of the 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). Such a framework is useful for designing reversible TCAs.

We define an *elementary* TPCA (ETPCA) as a TPCA such that each part of a cell has only two states (hence a cell has eight states), and its local transition function is isotropic (i.e., rotation-symmetric). There are 256 ETPCAs in total, and there are 36 *reversible* ETPCAs (RETPCAs). 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

M. Cook and T. Neary (Eds.): AUTOMATA 2016, LNCS 9664, pp. 170–184, 2016.
DOI: 10.1007/978-3-319-39300-1_14

cellular automata (ECAs) [13,14]. In [6], it is shown that the RETPCA with the identification number 0157 (explained later) is computationally universal.

In this paper, we investigate a specific RETPCA T_{0347} having the identification number 0347. It somewhat resembles Game-of-Life CA [2,4,5], and exhibits interesting 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, U-turn, and left-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 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. The next state of a cell is determined by the states of the adjacent parts of the neighboring cells. Figure 1 shows the case of a two-dimensional three-neighbor *triangular PCA* (TPCA). A *local function* of a TPCA is specified by a set of local rules of the form shown in Fig. 1 (b). Applying it to all the cells in parallel, a *global function*, which gives a transition relation among configurations, is obtained. Hereafter, we consider only deterministic PCAs.

We say a PCA is *locally reversible* if its local function is injective, and *globally reversible* if its global function is injective. It is known that global reversibility and local reversibility are equivalent (Lemma 1). Thus, such a PCA is simply called a *reversible PCA* (RPCA). Note that, in [11], the lemma is given for one-dimensional PCAs, but it is easy to extend it for two-dimensional PCAs.

Lemma 1. [11] *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 PCA 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 the both sides of it by a multiple of 60° exist. Note 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 8-state isotropic TPCA is called an *elementary TPCA* (ETPCA). Thus, each part of a cell has the state set $\{0, 1\}$. ETPCAs are the simplest ones among two-dimensional PCAs. But, this class still contains many interesting PCAs as in the case of one-dimensional elementary CAs (ECAs) [13,14].

Since ETPCA is isotropic, and each part of a cell has only two states, its local function is defined by only four local rules. Hence, an ETPCA can be specified

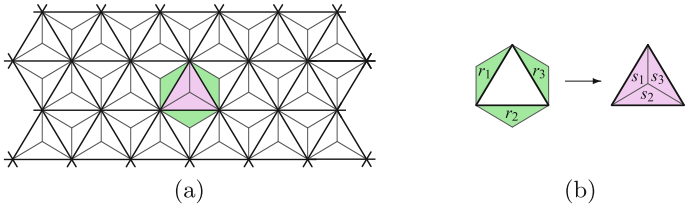

Fig. 1. A three-neighbor triangular PCA. (a) Its cellular space, and (b) a local rule.

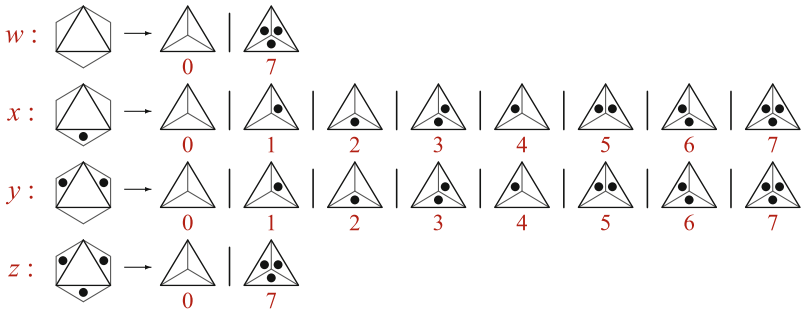

Fig. 2. Representing an ETPCA by a four-digit number $wxyz$, where $w, z \in \{0, 7\}$ and $x, y \in \{0, 1, \ldots, 7\}$. The states 0 and 1 are represented by a blank and •, respectively. Vertical bars indicate alternatives of a right-hand side of a rule.

by a four-digit number $wxyz$ such that $w, z \in \{0, 7\}$ and $x, y \in \{0, 1, \ldots, 7\}$ as shown in Fig. 2. Thus, there are 256 ETPCAs. Note that 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}.

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., •'s) is conserved in each rule. Thus, the following holds. Note that, if an ETPCA is conservative, then it is reversible.

An ETPCA T_{wxyz} is conservative iff
$w = 0 \ \wedge \ x \in \{1, 2, 4\} \ \wedge \ y \in \{3, 5, 6\} \ \wedge \ z = 7$

3 RETPCA T_{0347} and Its Properties

Here, we focus on a specific RETPCA T_{0347}, and investigate its properties. Its local function is given in Fig. 3. It is a non-conservative RETPCA. We shall see that there are many patterns (i.e., segments of configurations) that show

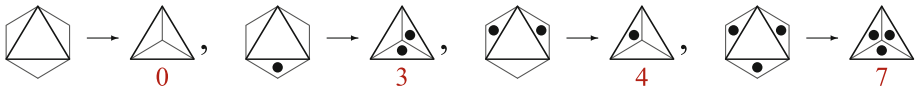

Fig. 3. Local function of the non-conservative RETPCA T_{0347}

interesting behavior as in the case of Game-of-Life CA [2,4,5]. In Sect. 4, some of them will be used to construct reversible logic gates in T_{0347}. In [10], evolving processes of various configurations of T_{0347} can be seen by movies.

3.1 Glider, Block, Fin, and Rotator in T_{0347}

The most useful object in T_{0347} is a *glider* (Fig. 4). 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. By rotating it appropriately, it can move in any of the six directions.

A *block* is an object shown in Fig. 5 (a) or (b). It does not change its pattern if no other object touches it. In this sense, it is a *stable* pattern. There are two kinds of blocks, i.e., type I (Fig. 5 (a)) and type II (Fig. 5 (b)). As explained in Subsect. 3.2, an appropriate type of a block must be used when colliding a glider with it. Combining several blocks, right-turn, U-turn, and left-turn of a glider will be implemented.

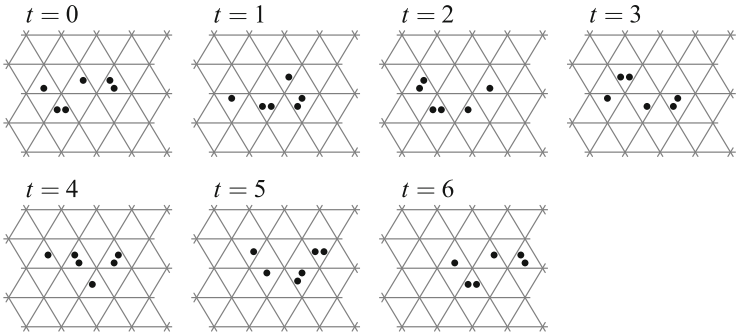

Fig. 4. Movement of a glider. The glider patterns at time $t = 0, \ldots, 5$, and 6 are said to be of phase $0, \ldots, 5$, and 0, respectively.

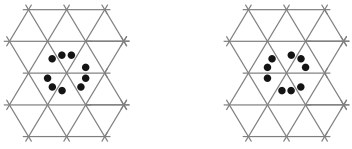

Fig. 5. Blocks of (a) type I and (b) type II. They are stable patterns.

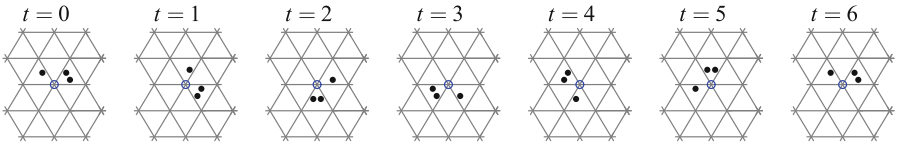

Fig. 6. A fin that rotates around the point indicated by ○

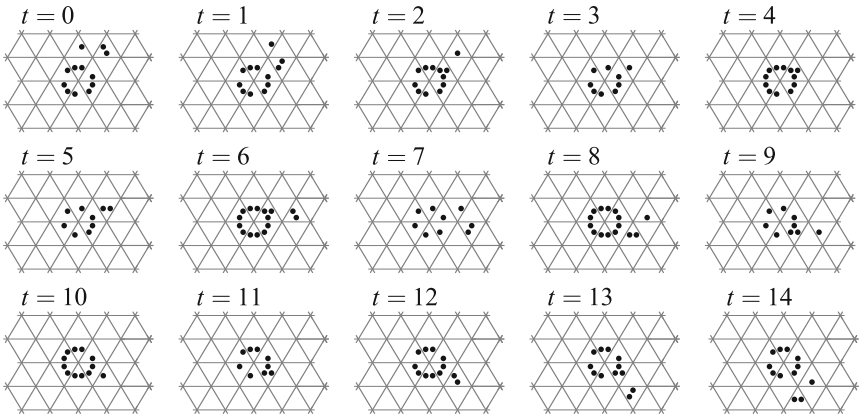

Fig. 7. A fin can travel around a block clockwise. It takes 42 steps to return to the initial position. Note that the block changes its pattern transiently, but it becomes the initial pattern again at $t = 14$.

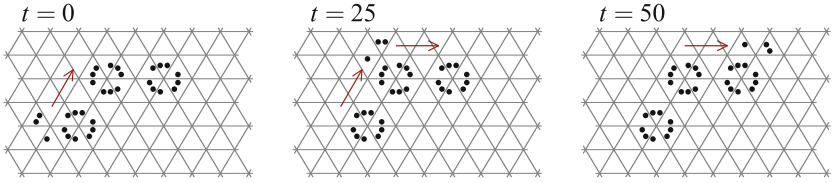

Fig. 8. A fin travelling along a sequence of blocks

A *fin* is an object that simply rotates as in Fig. 6 with period 6. It can also travel around a block (Fig. 7), or a sequence of blocks (Fig. 8) clockwise.

A *rotator* is an object shown in Fig. 9. Like a fin, it rotates around some point, and its period is 42.

3.2 Changing the Move Direction of a Glider by Blocks

We now make several experiments of colliding a glider with blocks. First, we collide a glider with a single block. We put a glider that moves eastward, and a block of type I as shown in Fig. 10 ($t = 0$). At $t = 12$ they collide. Then, the glider is split into a body (i.e., a rotator) and a fin, and the body begins to rotate around the point indicated by ○ ($t = 16$). The fin travels around the block ($t = 38$). When it comes

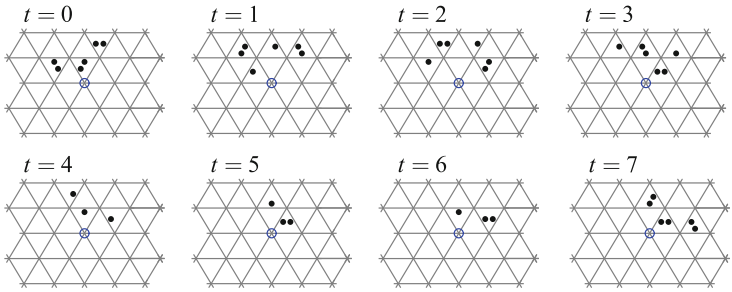

Fig. 9. A rotator that moves around the point indicated by ○. Its period is 42.

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 to reconstruct a glider. 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 some garbage remains (Fig. 11). Therefore, an appropriate type of block should be used depending on the coming direction of the glider. Figure 12 shows the allowed input positions of a glider in each type of a block.

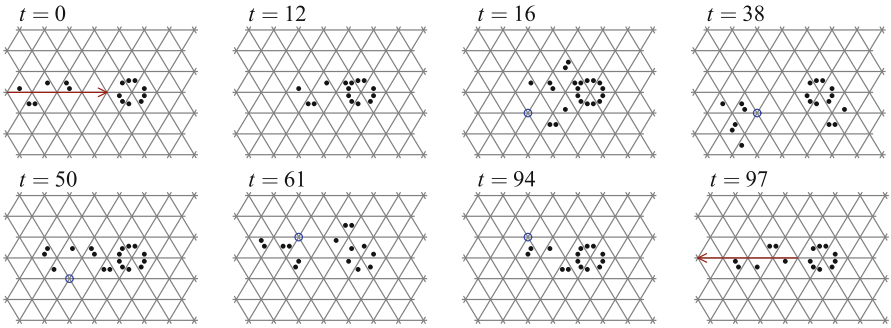

Fig. 10. Colliding a glider with a type I block. It works as a backward-turn module.

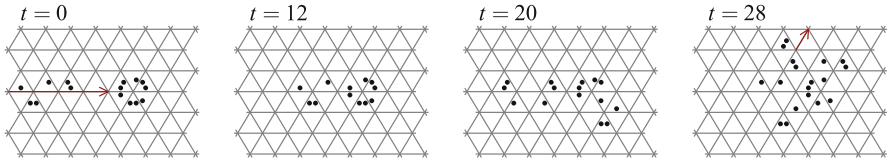

Fig. 11. If we collide a glider moving eastward with a type II block, garbage remains

Next, we collide a glider with two blocks (Fig. 13). As in the case of one block (Fig. 10), the glider is 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 form a glider, which goes to the south-west direction ($t = 334$). Hence, two blocks act as a right-turn module.

Figures 14 and 15 show that three blocks and five blocks also act as right-turn modules. They have shorter delays than the case of two blocks. Note that, as in Fig. 15, blocks need not be placed in a straight line. Namely, the sequence of

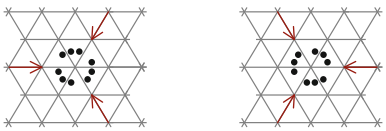

Fig. 12. Allowed input positions for (a) the type I block, and (b) the type II block

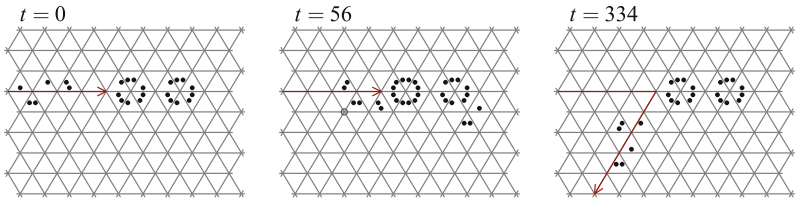

Fig. 13. 120°-right-turn module composed of two blocks

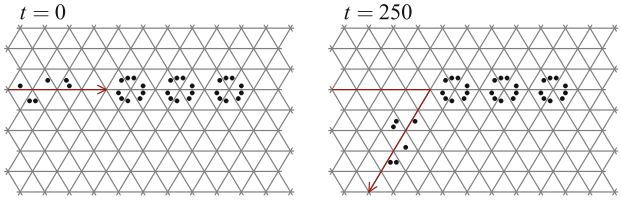

Fig. 14. Right-turn module composed of three blocks

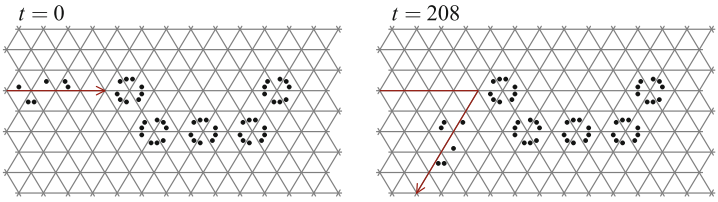

Fig. 15. Right-turn module composed of five blocks, which are not linearly placed

blocks can be bent slightly. But, if we do so, an appropriate type of a block must be used at each position. Otherwise, blocks will be destroyed.

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

A U-turn module is given in Fig. 16. Also in this case, the glider is first split into a rotator and a fin ($t = 36$). But, 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. 10).

Figure 17 shows a 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. 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 ($t = 366$).

It is also possible to make a 60°-right-turn module as in Fig. 18. First, the input glider makes 120°-right-turn by the three blocks ($t = 250$). Next, the glider is reflected by the left-side block. Finally, it makes 120°-right-turn again by the

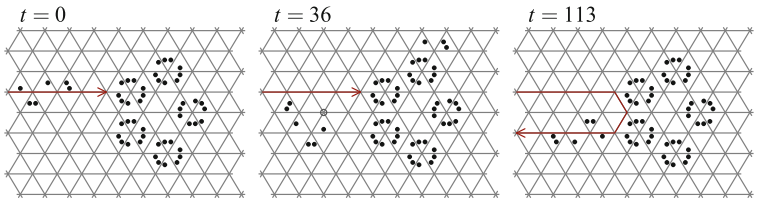

Fig. 16. U-turn module

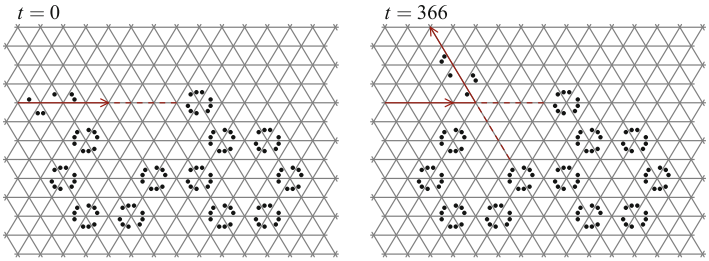

Fig. 17. 120°-left-turn module

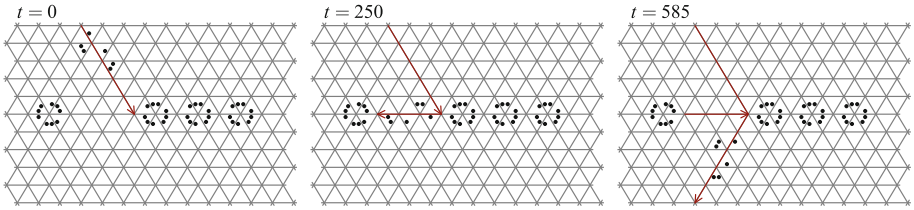

Fig. 18. 60°-right-turn module composed of four blocks

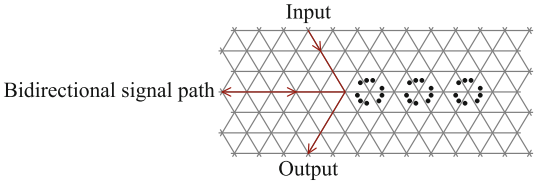

Fig. 19. Interface between a bidirectional signal path, and unidirectional signal paths

three blocks ($t = 585$). This mechanism is also used as an interface between a bidirectional signal path and unidirectional signal paths as shown in Fig. 19.

Table 1 shows the net delay d (i.e., the additional delay caused by the module) and the phase shift s of each turn module. Here, the relation $s = (-d) \bmod 6$ holds. In the case of Fig. 13, we can regard the travelling distance of the glider from $t = 0$ to 334 is 5. Since the speed of a glider is 1/6, $d = 334 - 5/(1/6) = 304$. Combining these turn modules, we can control the moving direction of a glider freely. It is also possible to shift the phase of a glider. For example, by making right-turn (120°) and then left-turn (120°), its phase is shifted by 2.

Table 1. Net delay and phase shift of the seven turn modules

Module	Net delay d	Phase shift s
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
Right-turn (60°) by 4 blocks	513	+3

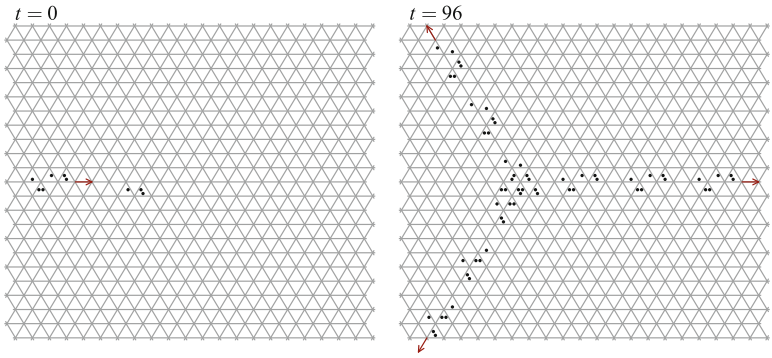

Fig. 20. Three-way glider gun. It generates three gliders every 24 steps.

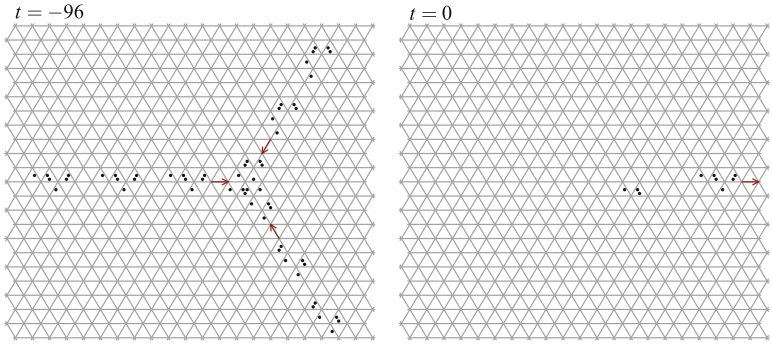

Fig. 21. Glider absorber. It is considered as a glider gun to the negative time direction.

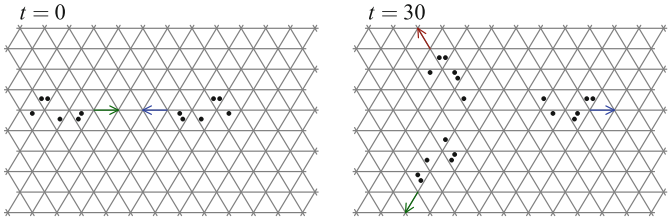

Fig. 22. Generating three gliders by the head-on collision of two gliders

3.3 Glider Guns in T_{0347}

A *glider gun* is a pattern that generates gliders periodically as the one in Game-of-Life [5]. In the RETPCA T_{0347}, it is very easy to create a three-way glider gun. As shown in Fig. 20, it is obtained by colliding a glider with a fin.

Interestingly, there is a *glider absorber* in T_{0347} (Fig. 21). It absorbs three gliders every 24 steps, and finally produces a fin and a glider. It is considered as a "backward glider gun" that generates gliders to the negative time direction.

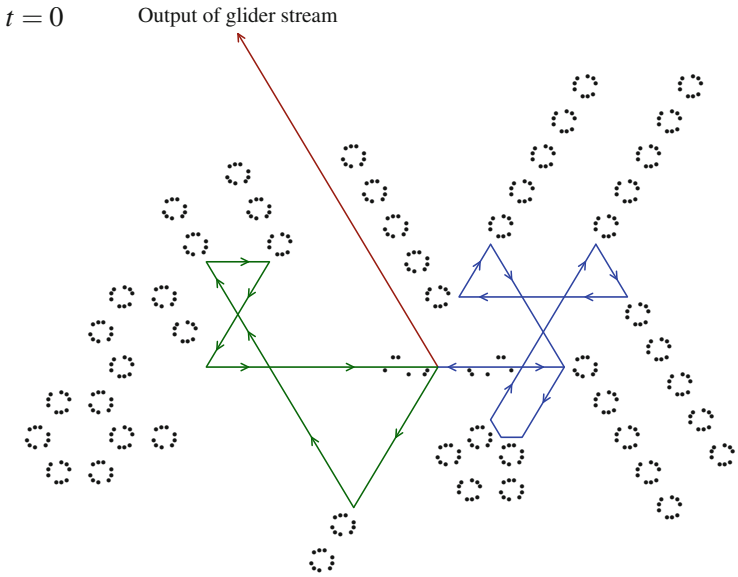

Fig. 23. One-way glider gun. It generates a glider every 1422 steps.

There is another way of composing a glider gun. Figure 22 shows that three gliders are generated by the head-on collision of two gliders. Based on this mechanism, we can design a one-way glider gun as shown in Fig. 23, where two of the three generated gliders are re-used to generate the next three.

4 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.

4.1 Showing Turing Universality of 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. 24 (a)), inverse switch gates (Fig. 24 (b)), and delay elements can be simulated in it (Lemma 6).

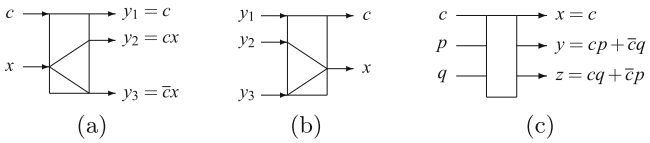

Fig. 24. (a) Switch gate. (b) Inverse switch gate, where $c = y_1$ and $x = y_2 + y_3$ under the assumption $(y_2 \to y_1) \wedge (y_3 \to \overline{y_1})$. (c) Fredkin gate.

Lemma 6 can be derived, e.g., in the following way. First, a Fredkin gate (Fig. 24 (c)) can be constructed out of switch gates and 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). Third, any *reversible Turing machine* is constructed out of REs (Lemma 4). Finally, any (irreversible) Turing machine is simulated by a 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. [3] *A Fredkin gate can be simulated by a circuit composed of switch gates and inverse switch gates, which produces no garbage signals.*

Lemma 3. [8] *Any RSM (in particular RE) can be simulated by a circuit composed of Fredkin gates and delay elements, which produces no garbage signals.*

Lemma 4. [9] *Any reversible Turing machine can be simulated by a garbage-less circuit composed only of REs.*

Lemma 5. [1] *Any (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 reversible two-dimensional CAs has been shown in this way. They are the 2-state reversible block CA model by Margolus [7], the two models of 16-state reversible PCAs on square grid by Morita and Ueno [12], and the conservative RETPCA T_{0157} by Imai and Morita [6].

4.2 Making Switch Gate and Inverse Switch Gate Modules 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. 25. 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. 26): (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 27 shows a switch gate module. The delay in this module is 2232 steps.

An inverse switch gate module is implemented in a similar manner as shown in Fig. 28. It is a "quasi-mirror-image" of the switch gate module. The positions

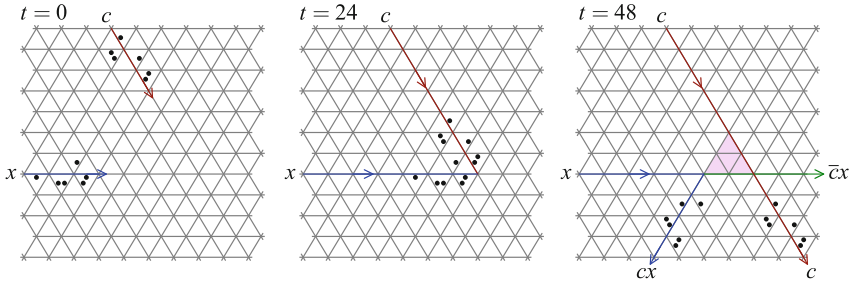

Fig. 25. Switch gate operation realized by collision of two gliders

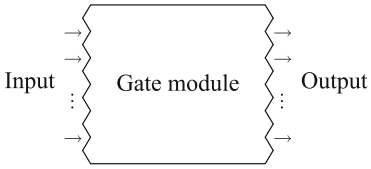

Fig. 26. Gate module in the standard form

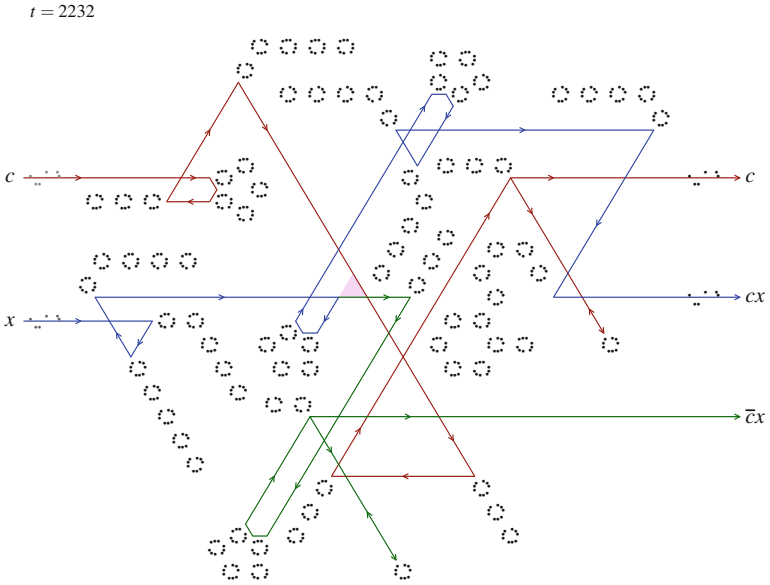

Fig. 27. Switch gate module implemented in T_{0347}

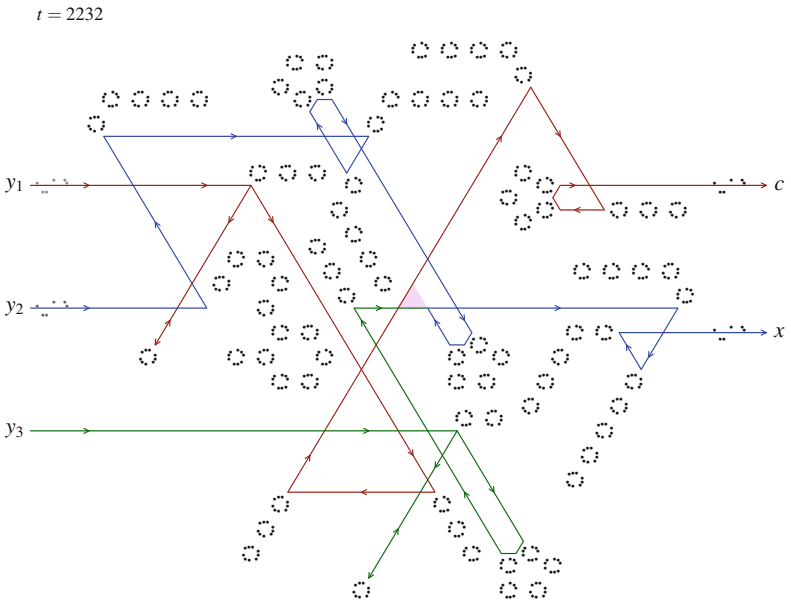

Fig. 28. Inverse switch gate module implemented in T_{0347}

of the blocks are just the mirror images of those in the switch gate. But, each block is replaced by the other type of the corresponding block (not by the mirror image of the block). The delay between input and output is also 2232 steps.

If we use only 120°-right-turn modules (Table 1) to connect gate modules, then there is no need of adjusting the phases of gliders. This is because the phase shift becomes 0, if we make 120°-right-turns three times. In this way, we can construct a larger circuit easily. 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 studied a specific reversible ETPCA T_{0347}. 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 phase of a glider can be completely controlled. Logical operation is also performed by interacting gliders. In this way, Turing universality of T_{0347} with infinite configurations was proved.

On the other hand, it is an open problem 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). It is also not known

whether universal systems (such as reversible Turing machines, reversible counter machines, or some others) are simulated in the finite configurations of T_{0347}.

Besides T_{0347}, it has already been shown that the conservative RETPCA T_{0157} with infinite configurations is Turing universal [6], 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.

Acknowledgement. This work was supported by JSPS KAKENHI Grant Number 15K00019.

References

1. Bennett, C.H.: Logical reversibility of computation. IBM J. Res. Dev. **17**, 525–532 (1973)
2. Berlekamp, E., Conway, J., Guy, R.: Winning Ways for Your Mathematical Plays, vol. 2. Academic Press, New York (1982)
3. Fredkin, E., Toffoli, T.: Conservative logic. Int. J. Theoret. Phys. **21**, 219–253 (1982)
4. Gardner, M.: Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". Sci. Am. **223**(4), 120–123 (1970)
5. Gardner, M.: Mathematical games: On cellular automata, self-reproduction, the Garden of Eden and the game "life". Sci. Am. **224**(2), 112–117 (1971)
6. Imai, K., Morita, K.: A computation-universal two-dimensional 8-state triangular reversible cellular automaton. Theoret. Comput. Sci. **231**, 181–191 (2000)
7. Margolus, N.: Physics-like model of computation. Physica D **10**, 81–95 (1984)
8. Morita, K.: A simple construction method of a reversible finite automaton out of Fredkin gates, and its related problem. Trans. IEICE Japan **E–73**, 978–984 (1990)
9. Morita, K.: A simple universal logic element and cellular automata for reversible computing. In: Margenstern, M., Rogozhin, Y. (eds.) MCU 2001. LNCS, vol. 2055, pp. 102–113. Springer, Heidelberg (2001)
10. Morita, K.: A reversible elementary triangular partitioned cellular automaton that exhibits complex behavior (slides with simulation movies). Hiroshima University Institutional Repository (2016). http://ir.lib.hiroshima-u.ac.jp/00039321
11. Morita, K., Harao, M.: Computation universality of one-dimensional reversible (injective) cellular automata. Trans. IEICE Japan **E72**, 758–762 (1989)
12. Morita, K., Ueno, S.: Computation-universal models of two-dimensional 16-state reversible cellular automata. IEICE Trans. Inf. Syst. **E75–D**, 141–147 (1992)
13. Wolfram, S.: Theory and Applications of Cellular Automata. World Scientific Publishing, Singapore (1986)
14. Wolfram, S.: A New Kind of Science. Wolfram Media Inc, Champaign (2002)