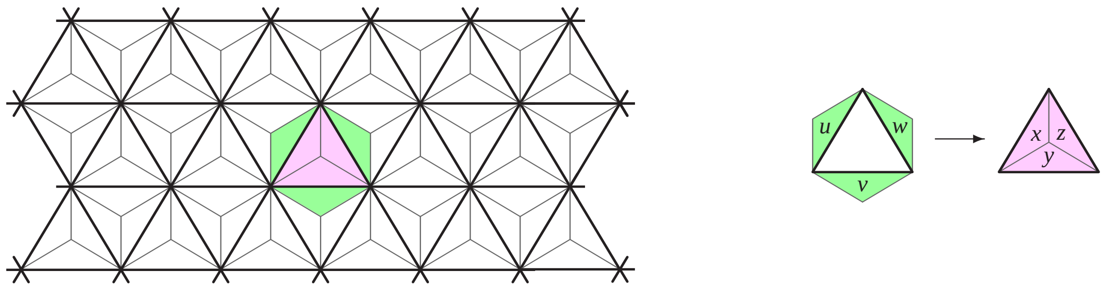
Reversible World:
A Reversible Elementary Triangular Partitioned Cellular Automaton That Exhibits Complex Behavior
(Version 2)

Kenichi Morita
morita.rcomp@gmail.com

March 2017

The rule files and the pattern files given here are for emulating the *reversible elementary triangular partitioned cellular automaton* No. 0347 (ETPCA 0347) proposed by Morita [4, 6]. The rule (i.e., local transition function) of ETPCA 0347 is quite simple, but it exhibits complex behavior, and thus presents an interesting Reversible World.

A three-neighbor triangular partitioned cellular automaton (TPCA) is a CA whose cell is triangular-shaped and divided into three parts, each of which has its own state-set (Fig. 1 (a)). The next state $[x,y,z]$ of a cell is determined by the states u, v and w of the three adjacent parts of its neighbor cells as shown in Fig. 1 (b) (not by the whole states of the three neighbor cells).



(a) (b)

Figure 1: A three-neighbor TPCA. (a) Its cellular space, and (b) a local rule $[u,v,w] \to [x,y,z]$.

The framework of TPCA makes it easy to design *reversible* triangular CAs. This is because the global transition function is injective iff the local transition function is injective. Among TPCAs, isotropic (i.e., rotation-symmetric) and 8-state (i.e., each part has only two states) TPCAs are called *elementary TPCAs* (ETPCAs) [4]. They are extremely simple, since each of their local functions is described by only four transitions.

Here, we consider a specific reversible ETPCA 0347, where 0347 is its identification number in the class of 256 ETPCAs. Its local function is described by the four isotropic transitions shown in Fig. 2. It is reversible, since there is no pair of transitions that have the same right-hand sides.

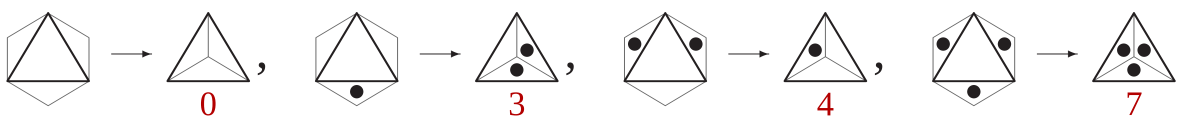


Figure 2: Local function of the reversible ETPCA 0347

Here, a pair of adjacent triangular cells (consisting of an up-triangle △ and a down-triangle ▽) of ETPCA 0347 is simulated in one square cell of Golly as in Fig. 3. A square cell has thus 64 states, and each state is indicated by a 6-bit number $a'b'c'd'e'f'$, where a' is the most significant bit. In Golly, a particle (i.e., state 1) in an up-triangle is colored in yellow, while a particle in a down-triangle is light-green, though they are the same states.

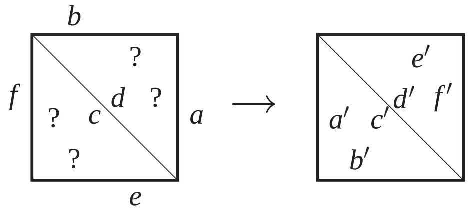


Figure 3: Emulating the transitions $[f,e,d] \rightarrow [a',b',c']$ and $[a,b,c] \rightarrow [f',e',d']$ of two triangular cells of ETPCA 0347 by one square cell of Golly. Here, "?" shows a don't-care state.

Note that there will be some difficulty in setting a pattern in the Golly simulator, since each cell has 64 states in this emulator. Also note that, since the emulated triangle in the square cell is not an equilateral triangle, patterns are slanted.

In spite of the extreme simplicity of the local function and the constraint of reversibility, evolutions of configurations in ETPCA 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 it. Furthermore, using gliders to represent signals, we can implement universal reversible logic gates and a 2-state reversible logic element with memory (RLEM) in it. Using the RLEM, patterns of ETPCA 0347 that simulate reversible Turing machines are constructed. Hence, reversible ETPCA 0347 is computationally universal.

The reversible ETPCA 0347 is not a *conservative* one, since the total number of particles is not conserved by the application of local rules (see Fig. 2). If otherwise, a glider gun cannot exist in this cellular space. Reversible and conservative ETPCAs and their computational universality have been studied in [2, 5, 7].

Pattern files

Sample pattern files are as below.

1. basic_objects.rle
   In this configuration, several basic objects and turn modules are shown. A *glider* is a moving object consisting of 6 particles with period 6 (Fig. 4). It is the most useful object, and will be used as a signal to implement reversible logic circuits. A *block* is a stable object consisting of 9 particles. *Right-, left-, backward-*, and *U-turn modules* are for changing the move direction of a glider, which are composed of several blocks (see [4, 6] for their details). At the bottom of this configuration, some small periodic patterns are shown. They are a *fin* (period: 6, Fig. 5), a *rotator* (period: 42), a *fan* (period: 2), a *pinwheel* (period: 8), and a *blinking block* (period: 2) (from the left to the right). Note that, if a glider collides with a turn module, it is first split into a fin and a rotator, and finally they are combined to form a glider again.

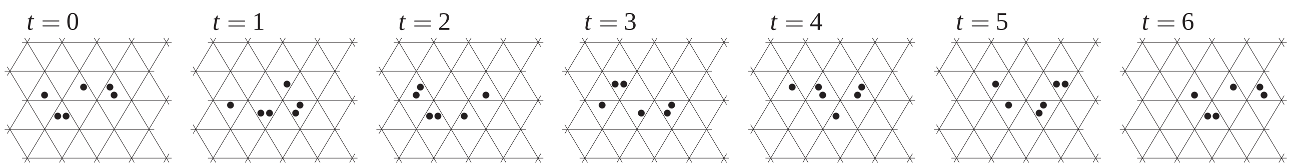


Figure 4: Glider in ETPCA 0347

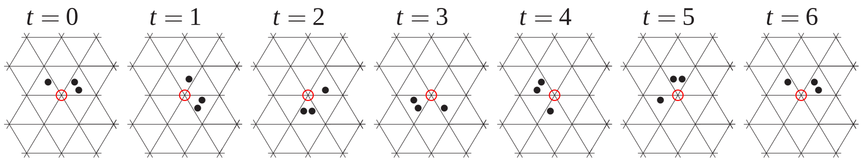


Figure 5: Fin in ETPCA 0347. It rotates around the point indicated by ○.

2. fredkin_gate.rle
   A Fredkin gate is a 3-input 3-output reversible logic gate with the logical function $(c, p, q) \mapsto (c, cp+\bar{c}q, cq+\bar{c}p)$ proposed by Fredkin and Toffoli [1]. Figure 6 shows an implementation of a Fredkin gate in ETPCA 0347. It is constructed from two switch gate modules (left) and two inverse switch gate modules (right). It is known that any reversible Turing machine can be realized by a garbage-less circuit composed of Fredkin gates and delay elements (see e.g., [6]). In this sense, it is a universal reversible gate. This file contains a pattern of a Fredkin gate module, and a module for giving gliders to the gate as inputs. In this configuration, all the 7 kinds of inputs (except (0,0,0)) are given one by one to show correctness of the pattern.

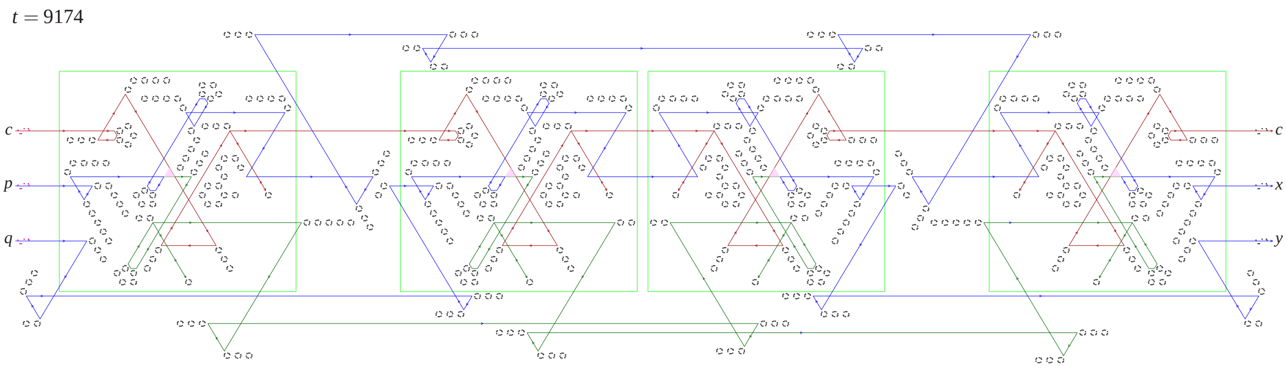


Figure 6: Fredkin gate in ETPCA 0347. Here, $x = cp+\bar{c}q$, and $y = cq+\bar{c}p$.

3. glider_gun_1w_and_absorber.rle
   A *glider gun* is a pattern that generates gliders periodically. A *glider absorber* is a pattern that absorbs (i.e., erases) gliders periodically. Note that a glider absorber can be considered as a glider gun to the "negative time direction". In this configuration (see also Fig. 7), the left-hand pattern is a 1-way glider gun that generates a glider every 1422 steps, while the right-hand one is a 1-way glider absorber that absorbs the generated gliders. Note that, since ETPCA 0347 is reversible, simple annihilation of a glider is impossible. Hence, such a "reversible erasure" mechanism is required to erase it. The design of a 1-way glider gun is based on the property that three gliders are obtained by a head-on collision of two gliders. Here, two gliders among the generated three are re-used to generate the next three, and so on. Likewise, a 1-way glider absorber is constructed based on the mechanism that two gliders are obtained by a collision of three gliders. Thus, the glider absorber has a symmetric structure to the glider gun.

4. glider_gun_3w.rle
   To give a glider gun that emits gliders in three directions is rather easy. Colliding a glider with a fin, a 3-way glider gun is obtained. It generates three gliders every 24 steps.

5. glider_gun_3w_in_both_time_directions.rle
   This pattern generates gliders to the "negative time direction" in addition to the positive time direction. In this sense, its behavior is symmetric with respect to the time axis. In other words, there are two integers t_0 and t_1 such that $t_0 < t_1$, and the pattern acts as a glider absorber when the time t satisfies $t < t_0$, and it acts as a glider gun when $t > t_1$.

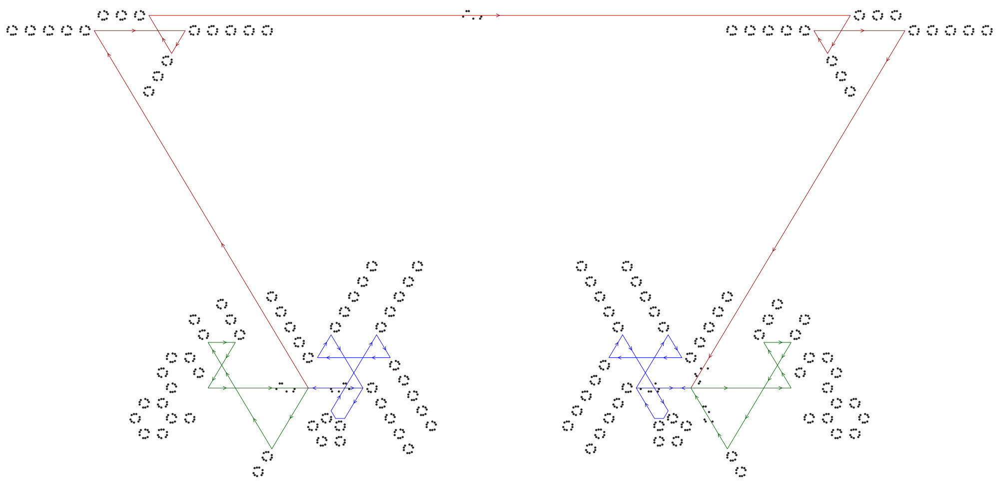


Figure 7: A 1-way glider gun (left), and a glider absorber (right)

6. glider_gun_6w_from_disordered_pattern.rle
   It is also possible to construct a 6-way glider gun. It consists of three 3-way glider guns, and one 3-way glider absorber. The given pattern in this file is a "disordered" one. But, it gradually shrinks, and finally becomes a 6-way glider gun that generates six gliders every 24 steps.

7. interaction_gate_and_its_inverse.rle
   An interaction gate is a 2-input 4-output reversible logic gate with the logical function $(x,y) \mapsto (xy, \bar{x}y, x\bar{y}, xy)$. An inverse interaction gate realizes its inverse function. It is known that a Fredkin gate, a universal reversible logic gate, can be realized by three interaction gates and three inverse interaction gates [1]. This file contains an interaction gate module and an inverse interaction gate module, which are serially connected. Hence, the whole circuit computes the 2-input 2-output identity function. To this circuit, all the 3 kinds of inputs (except (0,0)) are given one by one to verify correctness of their functions.

8. one_particle.rle
   This file contains a pattern consisting only of one particle. From this configuration, a disordered pattern emerges, and it grows bigger and bigger indefinitely. Also, many gliders appear around the disordered pattern. Such evolution processes are often observed in ETPCA 0347. Therefore, if we want to give a pattern that performs some intended task, such as logical operation or computing, we should design a pattern so that it never produces a disordered one.

9. one_particle_from_disordered_pattern.rle
   As seen in "one_particle.rle", a disordered pattern grows indefinitely from the one-particle pattern. Since ETPCA 0347 is reversible, the size (or diameter) of the pattern must grow indefinitely also to the "negative time direction" (note that its population need not grow indefinitely). In fact, from the one-particle pattern, a disordered pattern expands also to the negative time direction. This file shows such a process, i.e., the given disordered pattern first shrinks to a single particle, and then a disordered pattern appears and grows again.

10. switch_gate_and_its_inverse.rle
    A switch gate is a 2-input 3-output reversible logic gate with the logical function $(c,x) \mapsto (c, cx, \bar{c}x)$. An inverse switch gate realizes its inverse function. It is known that a Fredkin gate, a universal reversible logic gate, can be realized by two switch gates and two inverse switch gates [1] (see also "fredkin_gate.rle"). This file contains patterns of a switch gate module and an inverse switch gate module, which are serially connected. Hence, the whole circuit computes the 2-input 2-output identity function. To this circuit, all the 3 kinds of inputs (except (0,0)) are given one by one to verify correctness of their functions.

11. three_particles.rle
    This file contains a pattern with three particles. From this configuration, many gliders as well as a disordered pattern are generated, as in the case of "one_particle.rle". Since the initial pattern is symmetric under the rotation of 120 degrees, any of its descendent patterns is also so (but, there may be some difficulty to recognize it, because patterns are distorted in this emulator).

12. fin_shifting.rle
    This configuration shows how the position of a fin is shifted by colliding a glider with it. When a 2-state reversible logic element with memory (RLEM) is realized in ETPCA 0347 (see also "rlem_4-31.rle"), a fin is used to keep its state. Thus, the state of the RLEM can be changed by the shifting operations.

13. rlem_4-31.rle
    In this file, a pattern that simulates a specific reversible logic element with memory No. 4-31 (RLEM 4-31) is given. It is a useful 2-state RLEM with 4 input ports and 4 output ports, from which any reversible Turing machine can be constructed concisely [9]. The operation of RLEM 4-31 is represented in a pictorial form as shown in Fig. 8 (a). Two rectangles in the figure correspond to the two states 0 and 1. For each input symbol (output symbol, respectively) of RLEM 4-31, there is a unique input (output) port, to (from) which a signal is given (goes out). Solid and dotted lines show the input-output relation in each state. If an input signal goes through a dotted line, then the state does not change (Fig. 8 (b)). On the other hand, if a signal goes through a solid line, then the state changes (Fig. 8 (c)). Figure 9 shows the pattern of RLEM 4-31 implemented in the cellular space of ETPCA 0347.

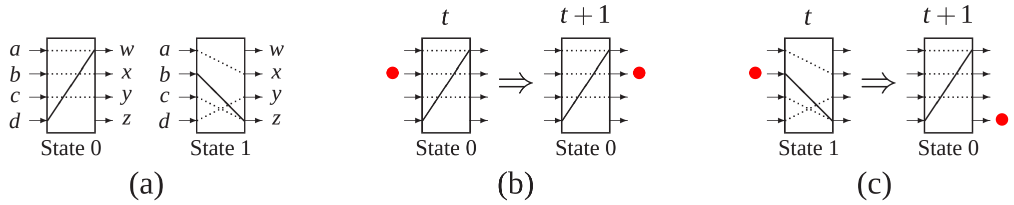


Figure 8: RLEM 4-31, and its operation examples. (a) Two states of RLEM 4-31. (b) The case that the state does not change, and (c) the case that the state changes.

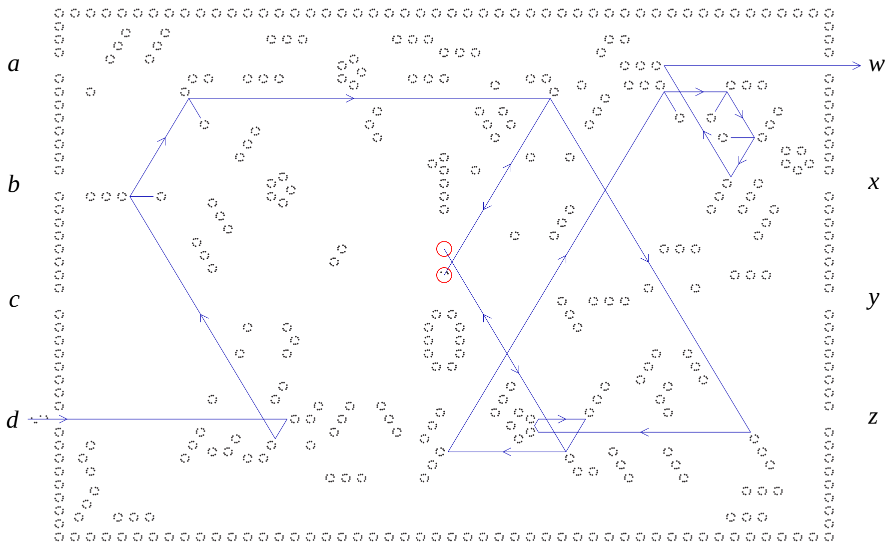


Figure 9: Realization of RLEM 4-31 in ETPCA 0347. Two circles in the middle of the pattern show possible positions of a fin. In this case, it is at the lower position, which indicates the state of the RLEM 4-31 is 0. Here, a glider is given at the input port d. The path from d to the output port w shows the trajectory of the glider. By this, the state changes from 0 to 1.

14. rtm_parity_n.rle ($n=2,3$)
They are configurations that simulate a reversible Turing machine (RTM) T_{parity} that has the set of quintuples $\{[q_0,0,1,R,q_1],\ [q_1,0,1,L,q_{\text{acc}}],\ [q_1,1,0,R,q_2],\ [q_2,0,1,L,q_{\text{rej}}],\ [q_2,1,0,R,q_1]\}$. For example, $[q_0,0,1,R,q_1]$ means that if T_{parity} reads the symbol 0 in the state q_0, then rewrite the symbol to 1, shift the head to the right, and go to the state q_1. Assume a symbol string $0\,1^n\,0$ $(n=0,1,\ldots)$ is given as an input. Then, T_{parity} halts in the accepting state q_{acc} iff n is even, and all the read symbols are complemented. Figure 10 shows the computing process for the input string 0110.

It has been shown that any RTM can be constructed out of RLEM 4-31 concisely [9]. Figure 11 gives the whole circuit for simulating T_{parity} for the input 1. The circuit consists of two components. They are a finite control unit (left), and a tape unit (right). The tape unit is composed of an infinite copies of a memory cell module, which is a vertical array of nine RLEMs. Each memory cell simulates one square of the tape. The top RLEM of a memory cell keeps a tape symbol. The remaining eight RLEMs execute read/write and head-shift commands sent from the finite control. They also keep the head position. If the states of the eight RLEMs are all 1, then the head position is at this memory cell. If they are all 0, the head is not here. If a particle is given to the "Begin" port, it starts to compute. Its answer will be obtained at "Accept" or "Reject" port.

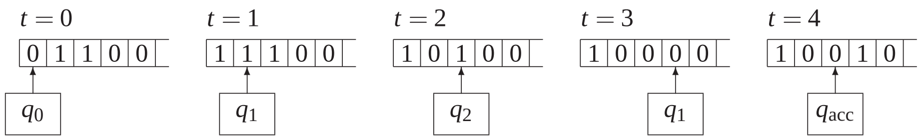


Figure 10: A computing process of the RTM T_{parity} for the given unary number 2

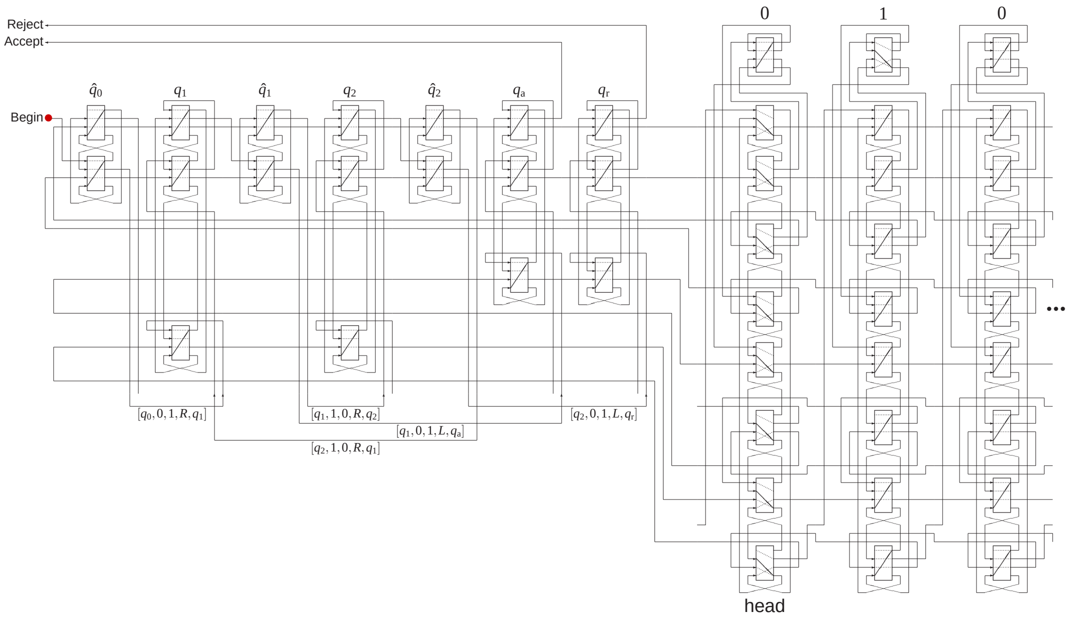


Figure 11: A circuit composed of RLEM 4-31 that simulates the RTM T_{parity} [9]

Figure 12: A configuration of ETPCA 0347 in Golly that simulates the circuit of Fig. 11

The configuration of ETPCA 0347 that simulates T_{parity} is obtained by putting copies of the pattern of Fig. 9 at the positions of RLEM 4-31 in the Fig. 11. The resulting configuration is shown in Fig. 12. The file "rtm_parity_n.rle" ($n = 2, 3$) gives the configuration of T_{parity} for the input n constructed in this way. Giving a glider to "Begin" port, its computation starts. Finally, a glider will appear at "Accept" or "Reject" port.

Since it takes a large number of steps to have an answer as shown in Table 1, the simulation speed of Golly should be increased by the key "+" or by the hyper-speed mode. However, if we do so, the glider moves very fast, and is almost invisible. Hence, it is very hard to distinguish which port "Accept" or "Reject" the glider passed. To magnify the answer, two small "bomb patterns" are put to the left of the finite control unit. If the glider passes the Accept port, then it hits the bomb for acceptance, and an explosion occurs. Likewise, if the glider passes the Reject port, then it hits the bomb for rejection. Note that the number of steps shown in Table 1 gives the time when the glider reaches a bomb.

Table 1: The number of steps for T_{parity} in ETPCA 0347 to give an answer

	Number of steps	Answer
rtm_parity_2.rle	5,342,077	Accept
rtm_parity_3.rle	6,876,940	Reject

15. rtm_power_of_two_n.rle ($n = 2, 3, 4, 6, 8$)
They are configurations that simulate an RTM T_{power} that has the following set of quintuples.

$$\begin{array}{llll} \{\ [q_0,0,0,R,q_1], & [q_1,0,0,R,q_2], & [q_2,0,0,L,q_6], & [q_2,1,0,R,q_3], \\ \ \ [q_3,0,1,L,q_4], & [q_3,1,1,R,q_3], & [q_4,0,0,L,q_7], & [q_4,1,0,L,q_5], \\ \ \ [q_5,0,1,R,q_2], & [q_5,1,1,L,q_5], & [q_6,0,0,L,q_{\text{rej}}], & [q_6,1,1,R,q_1], \\ \ \ [q_7,0,0,L,q_{\text{acc}}], & [q_7,1,1,L,q_{\text{rej}}]\ \} & & \end{array}$$

Assume a symbol string $0\,0\,1^n\,0$ ($n = 0, 1, \ldots$) is given as an input. Then, T_{power} halts in the accepting state q_{acc} iff $n = 2^k$ holds for some $k \in \{0, 1, 2, \ldots\}$. Figure 13 shows the computing process for the input string $0\,0\,1^8\,0$. Table 2 shows the time when T_{power} in ETPCA 0347 gives an answer. The file "rtm_power_of_two_n.rle" ($n = 2, 3, 4, 6, 8$) gives the configuration of T_{power} for the input n.

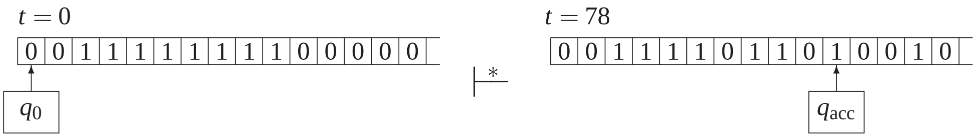


Figure 13: A computing process of the RTM T_{power} for the given unary number 8

Table 2: The number of steps for T_{power} in ETPCA 0347 to give an answer

	Number of steps	Answer
rtm_power_of_two_2.rle	23,391,049	Accept
rtm_power_of_two_3.rle	21,703,216	Reject
rtm_power_of_two_4.rle	55,607,635	Accept
rtm_power_of_two_6.rle	79,288,612	Reject
rtm_power_of_two_8.rle	154,678,141	Accept

Open problems

Some problems on ETPCA 0347 that have not been solved are listed below.

1. Are there moving objects other than the standard glider?
   So far, it is not known whether there exist moving objects that are essentially different from the standard glider. Here, “essentially different” means that the objects are not composed only of two or more standard gliders.
2. Are there patterns other than those in “basic_objects.rle” that are stable or of short period, and show interesting behavior?
3. Is there a 1-way glider gun with a shorter period?
   The gun in “glider_gun_1w_and_absorber.rle” is of period 1422. It is not known whether there is a simpler 1-way glider gun.
4. Is it possible to create two gliders from one glider and a stable or periodic pattern?
   Three gliders can be created by a head-on collision of two gliders. This mechanism is used to compose a 1-way glider gun in “glider_gun_1w_and_absorber.rle”. However, it is not known whether two gliders are obtained by colliding a glider with another (relatively simple) pattern.
5. Is there a Fredkin gate module with a shorter input-output delay?
   The Fredkin gate module given in “fredkin_gate.rle” is rather complex, and the delay between the input and the output is more than 9000 steps. How can we design much simpler one?
6. Can a reversible sequential machine be implemented directly and simply?
   A sequential machine (SM) is a kind of a finite automaton with an output port as well as an input port. It is known that *every* two-state reversible SMs with three or more input/output symbols are universal, i.e., any reversible SM and any reversible Turing machine can be constructed out of it rather simply [8, 9]. Although any reversible SM can be implemented as a garbage-less circuit composed of Fredkin gates and delay elements, the resulting circuit will become huge. Thus, it will be very useful if some two-state reversible SM is realized simply.

   Note: This problem was solved affirmatively in March 2017 (i.e., in Version 2), since RLEM 4-31 is realized directly without using reversible gates (see the pattern “rlem_4-31.rle”).
7. Is ETPCA 0347 construction-universal?
   Is it possible to give a universal constructor that creates *any* pattern in some specified class of patterns (e.g., the class of patterns consisting of blocks), if a description of the pattern is given.
8. Is it possible to simulate universal systems in finite configurations?
   If we construct a reversible Turing machine out of Fredkin gates and delay elements, then the circuit becomes infinite (but ultimately periodic). Can we design finite configurations that simulate universal systems as in [3, 10]?
9. Is there a periodic pattern whose ratio of the maximum number and the minimum number of particles in one cycle is larger than 3:1?
   This problem is to see what extent a periodic pattern can expand and shrink in a reversible space. The ratio of the pattern *pinwheel* given in “basic_objects.rle” (see also Fig. 14) is 3:1. So far, it is not known whether there is a periodic pattern with the ratio larger than this.

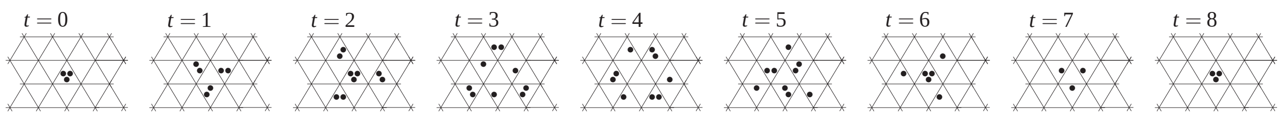


Figure 14: The maximum and the minimum numbers of particles in a *pinwheel* are 9 and 3.

Version history

- Reversible_World (Version 1) June 2016
  A rule file for ETPCA 0347 and eleven pattern files Nos. 1–11 are given. Nine open problems are presented.
- Reversible_World_v2 (Version 2) March 2017
  The rule file and the eleven pattern files Nos. 1–11 are exactly the same as in Version 1. Pattern files Nos. 12–15 are newly added. The open problem No. 6 is affirmatively solved here.

Acknowledgements: I express my great thanks to the developing and support teams of Golly.

References

[1] Fredkin, E., Toffoli, T.: Conservative logic. Int. J. Theoret. Phys. **21**, 219–253 (1982). DOI 10.1007/BF01857727

[2] Imai, K., Morita, K.: A computation-universal two-dimensional 8-state triangular reversible cellular automaton. Theoret. Comput. Sci. **231**, 181–191 (2000). DOI 10.1016/S0304-3975(99)00099-7

[3] Morita, K.: Universal reversible cellular automata in which counter machines are concisely embedded. Hiroshima University Institutional Repository, http://ir.lib.hiroshima-u.ac.jp/00031367 (2011)

[4] Morita, K.: An 8-state simple reversible triangular cellular automaton that exhibits complex behavior. In: Cellular Automata and Discrete Complex Systems, (eds. M. Cook, T. Neary) LNCS 9664, pp. 170–184 (2016). DOI 10.1007/978-3-319-39300-1_14

[5] Morita, K.: Reversible and conservative elementary triangular partitioned cellular automata (slides with simulation movies). Hiroshima University Institutional Repository, http://ir.lib.hiroshima-u.ac.jp/00039997 (2016)

[6] Morita, K.: A reversible elementary triangular partitioned cellular automaton that exhibits complex behavior: Glider, glider gun, and universalty (slides with simulation movies). Hiroshima University Institutional Repository, http://ir.lib.hiroshima-u.ac.jp/00039321 (2016)

[7] Morita, K.: Universality of 8-state reversible and conservative triangular partitioned cellular automaton. In: ACRI 2016 (eds. S. El Yacoubi, et al.) LNCS 9863, pp. 45–54 (2016). DOI 10.1007/978-3-319-44365-2_5

[8] Morita, K., Ogiro, T., Alhazov, A., Tanizawa, T.: Non-degenerate 2-state reversible logic elements with three or more symbols are all universal. J. Multiple-Valued Logic and Soft Computing **18**, 37–54 (2012)

[9] Morita, K., Suyama, R.: Compact realization of reversible Turing machines by 2-state reversible logic elements. In: Proc. UCNC 2014 (eds. O. H. Ibarra, L. Kari, S. Kopecki), LNCS 8553, pp. 280–292 (2014). DOI 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

[10] Morita, K., Tojima, Y., Imai, K., Ogiro, T.: Universal computing in reversible and number-conserving two-dimensional cellular spaces. In: Collision-based Computing (ed. A. Adamatzky), pp. 161–199. Springer (2002). DOI 10.1007/978-1-4471-0129-1_7