Computing Ability of Reversible and Conservative Cellular Automata
可逆的・保存的なセル・オートマトンの計算能力

Kenichi Morita, and Katsunobu Imai
森田 憲一, 今井克暢
Faculty of Engineering, Hiroshima University
広島大学工学部
{morita, imai}@ke.sys.hiroshima-u.ac.jp
http://www.ke.sys.hiroshima-u.ac.jp/~morita/

Abstract

We introduce a new model of cellular automaton called a one-dimensional number-conserving partitioned cellular automaton (NC-PCA). An NC-PCA is a system such that a state of a cell is represented by a triple of non-negative integers, and the total (i.e., sum) of integers over the configuration is conserved throughout its evolving (computing) process. It can be thought as a kind of modelization of the physical conservation law of mass (particles) or energy. We also define a reversible version of NC-PCA, and prove that a reversible NC-PCA is computation-universal. It is proved by showing that a reversible two-counter machine, which has been known to be universal, can be simulated by a reversible NC-PCA.

1   Introduction

Recently, various kinds of interesting computing models which directly reflect laws of nature have been proposed and investigated. Among others, quantum computing, DNA computing, reversible computing, etc. have been extensively studied. A reversible computer is a system such that its transition function of the whole state is a one-to-one mapping (injection), hence, roughly speaking, it is a backward deterministic system. It is a kind of model reflecting physical reversibility, and has been known to be very important when studying inevitable power dissipation in a computing process [3, 4, 5, 13]. Besides the problem of energy consumption, such a system is also interesting from a computational viewpoint, because they have very rich ability of information processing in spite of the reversibility constraint. Until now, several versions of computation-universality of RCAs for one- and two-dimensional cases have been shown [8, 9, 10, 11, 12, 16, 17, 18, 19]. It is also possible to construct an RCA in which various objects can self-reproduce [15, 17].

Conservation of mass or energy is also an important physical law as well as reversibility. Fredkin and Toffoli [5] proposed Conservative Logic, a kind of logic circuit theory, that models both reversibility and conservation law of physics, and showed its universality. In this system, each primitive logic gate must satisfy the constraints of reversibility (i.e., its logical function is an injection), and conservation of bits (i.e., the total number of logical value "1"s is conserved between its input and output). Also for cellular automata, several universal models that are both reversible and bit-conserving have been known [8, 9, 11].

In this paper, we define a new model of cellular automaton (CA) called a one-dimensional number-conserving partitioned cellular automaton (NC-PCA), which generalize the notion of bit-conserving CA. In an NC-PCA, each cell is partitioned into three parts, i.e., left, center, and right parts, and the state of each part is represented by a non-negative integer (thus, the state of a cell is represented by a triple of non-negative integers). The next state of a cell is determined by the present states of the right part of the left-neighboring cell, the center part of this cell, and the left part of the right-neighboring cell (not depending on the whole state of the three cells). The total number is conserved during the local transition, hence the total number over a configuration is also conserved throughout the evolving process.
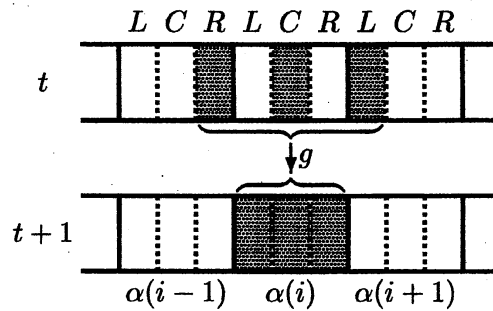
Figure 1: The local transition function g of a PCA A.

Related to this model, a few other models in which each cell state is represented by a non-negative integer have been known: a totalistic CA [1] and a sand pile model [6, 7]. In [1], a CA with a simple totalistic rule (but not necessarily number-conserving) has been shown to be universal. In [6, 7], a kind of an automata system having a specific type of number-conserving rules are studied.

Here, we investigate the computing ability of an NC-PCA, and its reversible version. We show that an NP-PCA is computation universal even if it is reversible. This strengthens the previous result that a one-dimensional reversible CA (not necessarily a number-conserving one) is computation-universal [10]. We prove it by showing that a reversible two-counter machine, which has been known to be universal [14], can be simulated by a reversible NC-PCA.

2 Number-Conserving Partitioned Cellular Automata

In order to define a one-dimensional number-conserving partitioned cellular automaton (NC-PCA), we first give a definition of a partitioned cellular automaton (PCA) that has been introduced to design a reversible cellular automaton [10].

Definition 2.1 A *deterministic one-dimensional three-neighbor partitioned cellular automaton* (PCA) is a system defined by

$$A = (\mathbf{Z}, (L, C, R), g, (\tilde{q}_L, \tilde{q}_C, \tilde{q}_R)),$$

where $\mathbf{Z}$ is the set of all integers at which cells are placed, L, C and R are non-empty finite sets of states of left, center, and right parts of a cell, $g : R \times C \times L \to L \times C \times R$ is a local function, and $(\tilde{q}_L, \tilde{q}_C, \tilde{q}_R) \in L \times C \times R$ is a quiescent state that satisfies $g(\tilde{q}_R, \tilde{q}_C, \tilde{q}_L) = (\tilde{q}_L, \tilde{q}_C, \tilde{q}_R)$.

A *configuration* over the set $Q = L \times C \times R$ is a mapping $\alpha : \mathbf{Z} \to Q$. Let $\mathrm{Conf}(Q)$ denote the set of all configurations over Q, i.e., $\mathrm{Conf}(Q) = \{\alpha \mid \alpha : \mathbf{Z} \to Q\}$. A *quiescent configuration* is the one such that all the cells are in the quiescent states $(\tilde{q}_L, \tilde{q}_C, \tilde{q}_R)$.

Let $\mathrm{pro}_L : Q \to L$ is a projection function such that $\mathrm{pro}_L(l, c, r) = l$ for all $(l, c, r) \in Q$. Projection functions $\mathrm{pro}_C : Q \to C$, and $\mathrm{pro}_R : Q \to R$ are also defined similarly. The *global function* $G : \mathrm{Conf}(Q) \to \mathrm{Conf}(Q)$ of A is defined as follows.

$$\forall x \in \mathbf{Z} : \quad G(\alpha)(x) = g(\mathrm{pro}_R(\alpha(x-1)), \mathrm{pro}_C(\alpha(x)), \mathrm{pro}_L(\alpha(x+1)))$$

Fig. 1 shows how the local function g is applied to each cell. In the following, an equation $g(r, c, l) = (l', c', r')$ is called a *rule* of A, and write it by

$$[r, c, l] \to [l', c', r'].$$

We regard the local function g as the set of such rules for convenience.

Next, we define the notion of reversibility for PCAs.

Definition 2.2 Let $A = (\mathbf{Z}, (L, C, R), g, (\tilde{q}_L, \tilde{q}_C, \tilde{q}_R))$ be a PCA. We say A is *globally reversible* iff its global function G is one-to-one, and *locally reversible* iff its local function g is one-to-one.

It is easy to prove the following proposition on PCA, which has been shown in [10].

Proposition 2.1 Let A be a PCA. A is globally reversible iff it is locally reversible.

By Proposition 2.1, a globally or locally reversible PCA is called simply "reversible" and denoted by RPCA. By this, if we want to construct a reversible CA, it is sufficient to give a PCA whose local function g is one-to-one. This makes it easy to design a reversible CA.

When we design a one-to-one local function g, it is sufficient to define it only on a subset of $R \times C \times L$ that are needed to perform a given task. Because, we can always find a one-to-one extension from a given partial function provided that the latter function is one-to-one on the subset. This is assured by the following proposition (its proof is omitted since it is easy).

Proposition 2.2 Let A and B be finite sets such that $|A| = |B|$, and let g be a mapping $A' \to B$ for some A' ($\subset A$). If g is one-to-one, then there is a one-to-one mapping $g' : A \to B$ such that $g'(a) = g(a)$ for all $a \in A'$.

We now give a definition of a number-conserving PCA. As in the case of a reversible CA, it is also convenient to use the framework of a PCA. Because, the the notion of number-conservation can be expressed by a simple constraint on a local function of a PCA.

Definition 2.3 Let $A = (\mathbf{Z}, (\mathbf{N}_m, \mathbf{N}_m, \mathbf{N}_m), g, (0, k, 0))$ be a PCA, where $\mathbf{N}_m$ denotes the set of integers $\{0, 1, \cdots, m-1, m\}$, and $k(\leq m)$ is a non-negative integer. A is called a *one-dimensional number-conserving partitioned cellular automaton* (NC-PCA), iff it satisfies the following condition: For all (r, c, l), $(l', c', r') \in \mathbf{N}_m^3$, if $g(r, c, l) = (l', c', r')$, then $r + c + l = l' + c' + r'$.

A *reversible NC-PCA* is also defined similarly, and denoted by NC-RPCA.

3 Universality of an NC-RPCA

In this section, we show that for any reversible two-counter machine there is an NC-RPCA that simulates it. Since a reversible two-counter machines has been known to be computation-universal [14], we can conclude that an NC-RPCA is also universal.

In [14] a counter machine (CM) is defined as a kind of multi-tape Turing machine whose heads are read-only ones and whose tapes are all blank except the leftmost squares as shown in Fig. 2 (P is a blank symbol). This definition is convenient for giving the notion of reversibility on a CM.
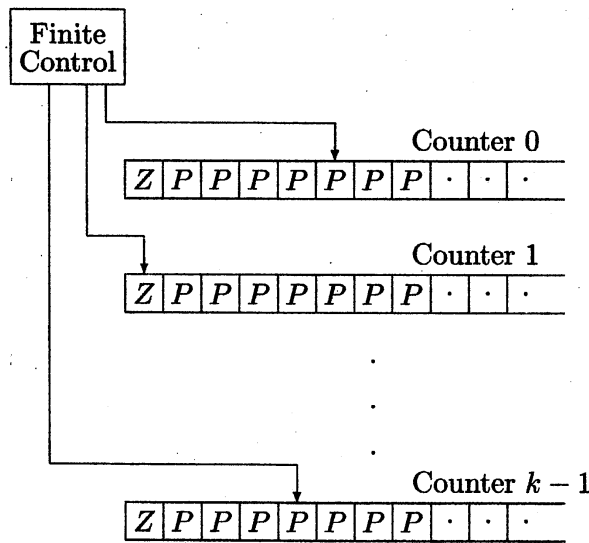
Definition 3.1 A *k-counter machine* (CM(k)) is a system

$$M = (k, Q, \delta, q_0, q_f),$$

where k is the number of tapes (or counters), Q is a nonempty finite set of internal states, $q_0 \in Q$ is an initial state, and $q_f \in Q$ is a final (halting) state. M uses $\{Z, P\}$ as a tape alphabet. δ is a move relation which is a subset of $(Q \times \{0, 1, \cdots, k-1\} \times \{Z, P\} \times Q) \cup (Q \times \{0, 1, \cdots, k-1\} \times \{-, 0, +\} \times Q)$ (where "$-$", "0", and "$+$" denote left-shift, no-shift, and right-shift of a head, respectively). Tapes are one-way (rightward) infinite. The leftmost squares of the tapes contain the symbol "Z"s, and all the other squares contain "P"s (Z and P stand for "zero" and "positive").

Each element of δ is called a *quadruple*, and is either of the form

$$[q, i, s, q'] \quad \text{or} \quad [q, i, d, q'],$$

Figure 2: A k-counter machine (CM(k)).

where $q, q' \in Q$, $i \in \{0, 1, \cdots, k-1\}$, $s \in \{Z, P\}$, $d \in \{-, 0, +\}$. The quadruple $[q, i, s, q']$ means that if M is in the state q and the i-th head is reading the symbol s then change the state into q'. It is used to test whether the contents of a counter are zero or positive. On the other hand, $[q, i, d, q']$ means that if M is in the state q then shift the i-th head to the direction d and change the state into q'. It is used to increment or decrement a counter by one (or make no change if $d = 0$).

Definition 3.2 An *instantaneous description* (ID) of a CM(k) $M = (k, Q, \delta, q_0, q_f)$ is a $(k+1)$-tuple

$$(q, n_0, n_1, \cdots, n_{k-1}) \in Q \times \mathbf{N}^k,$$

where $\mathbf{N} = \{0, 1, \cdots\}$. It represents that M is in the state q and the counter i keeps n_i (we assume the position of the leftmost square of a tape is 0). The transition relation $\vdash_{\overline{M}}$ over IDs of M is defined as follows:

$$\vdash_{\overline{M}} \begin{array}{l} (q, n_0, \cdots, n_{i-1}, n_i, n_{i+1}, \cdots, n_{k-1}) \\ (q', n_0, \cdots, n_{i-1}, n'_i, n_{i+1}, \cdots, n_{k-1}) \end{array}$$

holds iff one of the following conditions (1)–(5) is satisfied.

(1) $[q, i, Z, q'] \in \delta$ and $n_i = n'_i = 0$.
(2) $[q, i, P, q'] \in \delta$ and $n_i = n'_i > 0$.
(3) $[q, i, -, q'] \in \delta$ and $n_i - 1 = n'_i$.
(4) $[q, i, 0, q'] \in \delta$ and $n_i = n'_i$.
(5) $[q, i, +, q'] \in \delta$ and $n_i + 1 = n'_i$.

We denote reflexive and transitive closure of $\vdash_{\overline{M}}$ by $\vdash_{\overline{M}}^{*}$, and n-step transition by $\vdash_{\overline{M}}^{n}$ ($n = 0, 1, \cdots$).

Definition 3.3 Let $M = (k, Q, \delta, q_0, q_f)$ be a CM(k), and

$$\alpha_1 = [p_1, i_1, x_1, p'_1] \quad \text{and} \quad \alpha_2 = [p_2, i_2, x_2, p'_2]$$

be two distinct quadruples in δ. We say α_1 and α_2 *overlap in domain* iff the following holds, where $D = \{-, 0, +\}$.

$$p_1 = p_2 \ \wedge \ [i_1 \neq i_2 \ \vee \ x_1 = x_2 \ \vee \ x_1 \in D \ \vee \ x_2 \in D]$$

We say α_1 and α_2 *overlap in range* iff the following holds.

$$p_1' = p_2' \wedge [i_1 \neq i_2 \vee x_1 = x_2 \vee x_1 \in D \vee x_2 \in D]$$

A quadruple α is called *deterministic* (*reversible*, respectively) iff there is no other quadruple in δ which overlaps in domain (range) with α. M is called *deterministic* (*reversible*, respectively) iff every quadruple in δ is deterministic (reversible). A reversible CM(k) is denoted by RCM(k).

For example, the following pair

$$[q_1, 2, P, q_3] \text{ and } [q_4, 2, +, q_3]$$

overlaps in range, while the pair

$$[q_1, 2, Z, q_3] \text{ and } [q_4, 2, P, q_3]$$

does not. As seen from this definition, every ID of a deterministic (reversible, respectively) CM(k) has at most one ID that immediately follows (precedes) it. Hereafter, we consider only deterministic reversible and deterministic irreversible CM(k)s.

It has been known that an RCM(2) is computation-universal [14].

Proposition 3.1 [14] For any Turing machine T, there is a deterministic RCM(2) M that simulates T.

We need the following Lemma to prove Theorem 3.1

Lemma 3.1 For any deterministic CM(2) $M = (2, Q, \delta, q_0, q_f)$, there is a deterministic CM(2) $M' = (2, Q', \delta, q_0', q_f')$ that simulates M satisfying the following conditions: (i) The initial state q_0' never appears as the fourth element of a quadruple in δ' (hence it appears only at time 0). (ii) If M is reversible then M' is also reversible.

Proof. In the case M is irreversible, it is very easy to construct such M' by adding a new initial state to Q. So, we consider the reversible case. In [14], a construction method of a reversible CM(2) M_2 that simulates a given CM(k) M_1 ($k = 1, 2, \cdots$) (that is not necessarily reversible) has been shown. By checking the construction method shown in [14], we can verify that M_2 satisfies the above condition (i), provided that M_1 also satisfies it. Hence the Lemma holds. $\square$

Theorem 3.1 For any deterministic CM(2) M, there is a deterministic NC-PCA A that simulates M satisfying the following condition: If M is reversible then A is also reversible.

Proof. Without loss of generality, we assume that the state set of M is $Q = \{q_0, q_1, \cdots, q_{m-1}\}$, and the initial and final states are q_0, and q_{m-1}, respectively. Hence,

$$M = (2, \{q_0, q_1, \cdots, q_{m-1}\}, \delta, q_0, q_{m-1}).$$

Further assume that q_0 never appears as the fourth element of a quadruple in δ (by Lemma 3.1). Let Inc_j, Dec_j, Nop, and Test_j be the sets of states defined as follows ($j \in \{0, 1\}$).

$$
\begin{aligned}
\text{Inc}_j &= \{q_i \mid [q_i, j, +, q_k] \in \delta \text{ for some } q_k \in Q\} \\
\text{Dec}_j &= \{q_i \mid [q_i, j, -, q_k] \in \delta \text{ for some } q_k \in Q\} \\
\text{Nop} &= \{q_i \mid [q_i, j, 0, q_k] \in \delta \text{ for some } j \in \{0, 1\}, \text{ and } q_k \in Q\} \\
\text{Test}_j &= \{q_i \mid [q_i, j, s, q_k] \in \delta \text{ for some } s \in \{Z, P\}, \text{ and } q_k \in Q\}
\end{aligned}
$$

These sets stand for instructions of "increment the counter j", "decrement the counter j", "no-operation", and "test if the counter j is zero or positive". It is easy to see that Inc_j, Dec_j, Nop, and Test_j are pairwise disjoint (for example, $\text{Inc}_0 \cap \text{Inc}_1 = \emptyset$, $\text{Dec}_1 \cap \text{Nop} = \emptyset$, etc.), since M is deterministic.

We now construct an NC-PCA A that simulates M. Each part of a cell of A keeps a number at most $m + 18$, and the quiescent state is $(0,0,0)$. Thus,

$$A = (\mathbf{Z}, \mathbf{N}^3_{m+18}, g, (0,0,0)).$$

Before defining the local function g, we fix a coding method of an ID of M by a configuration of A. First, we assign an "operation code" to each state of M by the following function $\gamma : Q \to \{0,2,4,6,7\}$.

$$\gamma(q_i) = \begin{cases} 2 & \text{if } q_i \in \text{Inc}_0 \\ 4 & \text{if } q_i \in \text{Dec}_0 \\ 6 & \text{if } q_i \in \text{Inc}_1 \\ 7 & \text{if } q_i \in \text{Dec}_1 \\ 0 & \text{otherwise} \end{cases}$$

Note that the states in $\text{Nop} \cup \text{Test}_0 \cup \text{Test}_1$ and the halting states have the same operation code 0. We then define a coding function $\varphi : Q \times \mathbf{N}^2 \to \text{Conf}(\mathbf{N}^3_{m+18})$, which maps each ID of M to a configuration of A. Let $I = (q_i, n_0, n_1)$ be an ID of M. A configuration $\varphi(I)$ is computed by the following procedure.

begin
  $\alpha :=$ the quiescent configuration;
  $\text{pro}_L(\alpha(0)) := i + 10$;
  $\text{pro}_C(\alpha(0)) := (m + 16) - (i + 10) - \gamma(q_i)$;
  $\text{pro}_R(\alpha(0)) := \gamma(q_i)$;
  for each $j \in \{0,1\}$ **do**
    if $n_j = 0$ and $q_i \in \text{Inc}_j$ **then** $\text{pro}_R(\alpha(0)) := \text{pro}_R(\alpha(0)) + 2^j$
    else $\text{pro}_C(\alpha(n_j)) := \text{pro}_C(\alpha(n_j)) + 2^j$;
  $\varphi(I) := \alpha$
end.

For example, the configuration $\varphi(q_i, 2, 0)$ such that $q_i \in \text{Inc}_1$ is shown in Fig. 3.

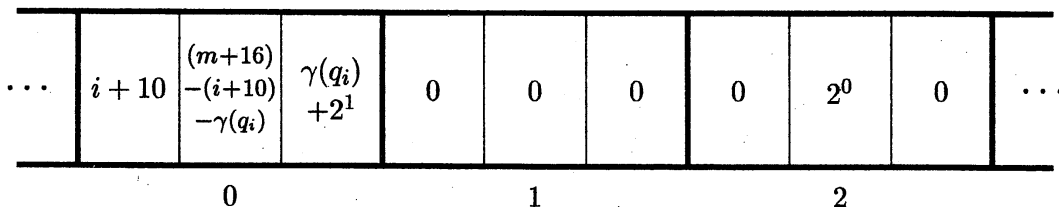| $\cdots$	$i + 10$	$(m+16)$ $-(i+10)$ $-\gamma(q_i)$	$\gamma(q_i)$ $+2^1$	0	0	0	0	2^0	0	$\cdots$
		0			1			2		

Figure 3: The configuration $\varphi(q_i, 2, 0)$ such that $q_i \in \text{Inc}_1$.

Each configuration $\varphi(I)$ has the number $m + 19$ in total. The number n_j kept by the counter j ($j \in \{0,1\}$) is recorded by putting the number 2^j at the center part of the cell n_j (at the right part of the cell 0, if $n_j = 0$ and the current state is in Inc_j). The number 2^j is called a *counter marker*. The remaining $m + 16$ particles are used to record the state of M, and to execute operations on counters.

In what follows, each state in $\{1, 2, \cdots, m + 9\}$ appearing in the left or the right part (not in the center part) of a cell is called a *signal*. Signals in $\{10, 11, \cdots, m + 9\}$ are called *state signals*, and are used to record the current state of M. State signals are kept by the cells 0 and -1 (they go back and forth between these cells). Signals in $\{2, 4, 6, 7\}$ are called *operation signals*, which are used to execute increment/decrement operations. Each of the four operation signals sometimes carry a counter marker to move it to the right- or left-neighboring cell. At that time, these signals "2", "4", "6", and "7" temporarily become "3", "5", "8", and "9", respectively. The signal "1" is a special one called an *initial/final signal* (it will be explained later).

We now define the local function g of A as follows.

1. Rules for the cases where no signal exists:
For each $x \in \mathbf{N}_{m+18}$, include the following rule in g.

$$[\ \ 0, \ \ x, \ \ 0 \ \ ] \ \rightarrow \ [\ \ 0, \ \ x, \ \ 0 \ \ ] \tag{1}$$

2. Rules for state signals:
For each $x \in \{10, 11, \cdots, m + 9\}$, include the following rule in g.

$$[\ \ 0, \ \ 0, \ \ x \ \ ] \ \rightarrow \ [\ \ 0, \ \ 0, \ \ x \ \ ] \tag{2}$$

3. Rules for the increment operation:
For each $j \in \{0, 1\}$ and $c \in \{0, 1\}$, include the following rules in g ($\oplus$ denotes the addition in mod 2).

$$[\ \ 2 + 4j, \ \ c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \ \rightarrow \ [\ \ 0, \ \ c \cdot 2^{j \oplus 1}, \ \ 2 + 4j \ \ ] \tag{3.1}$$
$$[\ \ 2 + 4j, \ \ 2^j + c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \ \rightarrow \ [\ \ 0, \ \ c \cdot 2^{j \oplus 1}, \ \ 2 + 4j + 2^j \ \ ] \tag{3.2}$$
$$[\ \ 2 + 4j + 2^j, \ \ c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \ \rightarrow \ [\ \ 2 + 4j, \ \ 2^j + c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \tag{3.3}$$
$$[\ \ 0, \ \ c \cdot 2^{j \oplus 1}, \ \ 2 + 4j \ \ ] \ \rightarrow \ [\ \ 2 + 4j, \ \ c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \tag{3.4}$$

4. Rules for the decrement operation:
For each $j \in \{0, 1\}$ and $c \in \{0, 1\}$, include the following rules in g.

$$[\ \ 4 + 3j, \ \ c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \ \rightarrow \ [\ \ 0, \ \ c \cdot 2^{j \oplus 1}, \ \ 4 + 3j \ \ ] \tag{4.1}$$
$$[\ \ 4 + 3j, \ \ 2^j + c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \ \rightarrow \ [\ \ 4 + 3j + 2^j, \ \ c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \tag{4.2}$$
$$[\ \ 0, \ \ c \cdot 2^{j \oplus 1}, \ \ 4 + 3j + 2^j \ \ ] \ \rightarrow \ [\ \ 4 + 3j, \ \ 2^j + c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \tag{4.3}$$
$$[\ \ 0, \ \ c \cdot 2^{j \oplus 1}, \ \ 4 + 3j \ \ ] \ \rightarrow \ [\ \ 4 + 3j, \ \ c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \tag{4.4}$$

5. Rules for waiting for the completion of the increment/decrement operation:
For each $[q_i, j, d, q_k] \in \delta$ such that $d \in \{+, -\}$, and for each $c \in \{0, 1\}$, include the following rule in g.

$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \ \rightarrow$$
$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c \cdot 2^{j \oplus 1}, \ \ 0 \ \ ] \tag{5}$$

6. Rules for simulating M's state transition from a state in Inc_j:
For each $[q_i, j, +, q_k] \in \delta$ and $c \in \{0, 1\}$, if $q_k \notin \text{Inc}_{j \oplus 1}$, then include the following rule in g.

$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c \cdot 2^{j \oplus 1}, \ \ \gamma(q_i) \ \ ] \ \rightarrow$$
$$[\ \ k + 10, \ \ (m + 16) - (k + 10) - \gamma(q_k) + c \cdot 2^{j \oplus 1}, \ \ \gamma(q_k) \ \ ] \tag{6.1}$$

For each $[q_i, j, +, q_k] \in \delta$ and $c \in \{0, 1\}$, if $q_k \in \text{Inc}_{j \oplus 1}$, then include the following rule in g.

$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c \cdot 2^{j \oplus 1}, \ \ \gamma(q_i) \ \ ] \ \rightarrow$$
$$[\ \ k + 10, \ \ (m + 16) - (k + 10) - \gamma(q_k), \ \ \gamma(q_k) + c \cdot 2^{j \oplus 1} \ \ ] \tag{6.2}$$

7. Rules for simulating M's state transition from a state in Dec_j:
For each $[q_i, j, -, q_k] \in \delta$ and $c, c' \in \{0, 1\}$, if $q_k \notin (\text{Inc}_j \cup \text{Inc}_{j \oplus 1})$, then include the following rule in g.

$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c' \cdot 2^{j \oplus 1}, \ \ \gamma(q_i) + c \cdot 2^j \ \ ] \ \rightarrow$$
$$[\ \ k + 10, \ \ (m + 16) - (k + 10) - \gamma(q_k) + c \cdot 2^j + c' \cdot 2^{j \oplus 1}, \ \ \gamma(q_k) \ \ ] \tag{7.1}$$

For each $[q_i, j, -, q_k] \in \delta$ and $c, c' \in \{0, 1\}$, if $q_k \in \text{Inc}_j$, then include the following rule in g.

$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c' \cdot 2^{j \oplus 1}, \ \ \gamma(q_i) + c \cdot 2^j \ \ ] \ \rightarrow$$
$$[\ \ k + 10, \ \ (m + 16) - (k + 10) - \gamma(q_k) + c' \cdot 2^{j \oplus 1}, \ \ \gamma(q_k) + c \cdot 2^j \ \ ] \tag{7.2}$$

For each $[q_i, j, -, q_k] \in \delta$ and $c, c' \in \{0, 1\}$, if $q_k \in \text{Inc}_{j \oplus 1}$, then include the following rule in g.

$$[\ \ i + 10, \ \ (m + 16) - (i + 10) - \gamma(q_i) + c' \cdot 2^{j \oplus 1}, \ \ \gamma(q_i) + c \cdot 2^j \ \ ] \ \rightarrow$$
$$[\ \ k + 10, \ \ (m + 16) - (k + 10) - \gamma(q_k) + c \cdot 2^j, \ \ \gamma(q_k) + c' \cdot 2^{j \oplus 1} \ \ ] \tag{7.3}$$

8. Rules for simulating M's state transition from a state in Nop:
For each $[q_i, j, 0, q_k] \in \delta$ and $c, c' \in \{0,1\}$, if $q_k \notin (\text{Inc}_0 \cup \text{Inc}_1)$, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + c \cdot 2^0 + c' \cdot 2^1, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) - \gamma(q_k) + c \cdot 2^0 + c' \cdot 2^1, \quad \gamma(q_k) \;] \quad (8.1)$$

For each $[q_i, j, 0, q_k] \in \delta$ and $c, c' \in \{0,1\}$, if $q_k \in \text{Inc}_j$ for some j, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + c \cdot 2^j + c' \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) + \gamma(q_k) + c' \cdot 2^{j \oplus 1}, \quad \gamma(q_k) + c \cdot 2^j \;] \quad (8.2)$$

9. Rules for simulating M's state transition from a state in Test$_j$:
For each $[q_i, j, Z, q_k] \in \delta$ and $c \in \{0,1\}$, if $q_k \notin (\text{Inc}_j \cup \text{Inc}_{j \oplus 1})$, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + 2^j + c \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) - \gamma(q_k) + 2^j + c \cdot 2^{j \oplus 1}, \quad \gamma(q_k) \;] \quad (9.1)$$

For each $[q_i, j, Z, q_k] \in \delta$ such that and $c \in \{0,1\}$, if $q_k \in \text{Inc}_j$, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + 2^j + c \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) - \gamma(q_k) + c \cdot 2^{j \oplus 1}, \quad \gamma(q_k) + 2^j \;] \quad (9.2)$$

For each $[q_i, j, Z, q_k] \in \delta$ and $c \in \{0,1\}$, if $q_k \in \text{Inc}_{j \oplus 1}$, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + 2^j + c \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) - \gamma(q_k) + 2^j, \quad \gamma(q_k) + c \cdot 2^{j \oplus 1} \;] \quad (9.3)$$

For each $[q_i, j, P, q_k] \in \delta$ and $c \in \{0,1\}$, if $q_k \notin \text{Inc}_{j \oplus 1}$, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + c \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) - \gamma(q_k) + c \cdot 2^{j \oplus 1}, \quad \gamma(q_k) \;] \quad (9.4)$$

For each $[q_i, j, P, q_k] \in \delta$ and $c \in \{0,1\}$, if $q_k \in \text{Inc}_{j \oplus 1}$, then include the following rule in g.

$$[\; i+10, \quad (m+16) - (i+10) + c \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; k+10, \quad (m+16) - (k+10) - \gamma(q_k), \quad \gamma(q_k) + c \cdot 2^{j \oplus 1} \;] \quad (9.5)$$

10. Rules for the initial/final signal:
Include the following rules in g.

$$[\; 1, \quad 0, \quad 0 \;] \; \rightarrow \; [\; 0, \quad 0, \quad 1 \;] \quad (10.1)$$
$$[\; 0, \quad 0, \quad 1 \;] \; \rightarrow \; [\; 1, \quad 0, \quad 0 \;] \quad (10.2)$$

For each $c, c' \in \{0,1\}$, if $q_0 \notin (\text{Inc}_0 \cup \text{Inc}_1)$, then include the following rule in g.

$$[\; 1, \quad (m+15) + c \cdot 2^0 + c' \cdot 2^1, \quad 0 \;] \; \rightarrow$$
$$[\; 10, \quad (m+16) - 10 - \gamma(q_0) + c \cdot 2^0 + c' \cdot 2^1, \quad \gamma(q_0) \;] \quad (10.3)$$

For each $c, c' \in \{0,1\}$, if $q_0 \in \text{Inc}_j$ for some j, then include the following rule in g.

$$[\; 1, \quad (m+15) + c \cdot 2^j + c' \cdot 2^{j \oplus 1}, \quad 0 \;] \; \rightarrow$$
$$[\; 10, \quad (m+16) - 10 - \gamma(q_0) + c' \cdot 2^{j \oplus 1}, \quad \gamma(q_0) + c \cdot 2^j \;] \quad (10.4)$$

For each $c, c' \in \{0,1\}$, include the following rule in g.

$$[\; (m+9), \quad (m+16) - (m+9) + c \cdot 2^0 + c' \cdot 2^1, \quad 0 \;] \; \rightarrow$$
$$[\; 1, \quad (m+15) + c \cdot 2^0 + c' \cdot 2^1, \quad 0 \;] \quad (10.5)$$

We can see that A correctly simulates M step by step. It is easy to verify that each rule conserves the total number between left- and right-hand sides, and hence A is an NC-PCA.

Now, we show that the following statement holds: If M is reversible, so is A. Assume M is reversible. It suffices to show that each of the above rules has a different right-hand side from those of the other rules. First, we can easily verify that rules (1), (2), (3.x), (4.x), (10.1), (10.2), and (10.5) satisfy this constraint by simply comparing their right-hand sides with other rules. The rules (10.3), and (10.4) are also so. Because the state q_0 does not appear as the forth element of a quadruple in δ, hence the right-hand sides of these rules never matches those of (6.x), (7.x), (8.x), and (9.x), as well as the others.

Next, we consider the rules (5), which correspond to the quadruple $[q_i, j, d, q_k] \in \delta$ such that $d \in \{+, -\}$. The state q_i may appear as the fourth element of the other quadruples. But, since $\gamma(q_i) > 0$ (because $q_i \in (\mathrm{Inc}_0 \cup \mathrm{Inc}_1 \cup \mathrm{Dec}_0 \cup \mathrm{Dec}_1)$), the right-hand sides of the rules (5) never match those of (6.x), (7.x), (8.x), and (9.x).

We finally verify that rules (6.x), (7.x), (8.x), and (9.x) satisfy the reversibility constraint. Let Inc', Dec', Nop', and Test' be the sets of states of M defined as follows.

$$\mathrm{Inc}' = \{q_k \mid [q_i, j, +, q_k] \in \delta \text{ for some } j \in \{0, 1\}, \text{ and } q_i \in Q\}$$
$$\mathrm{Dec}' = \{q_k \mid [q_i, j, -, q_k] \in \delta \text{ for some } j \in \{0, 1\}, \text{ and } q_i \in Q\}$$
$$\mathrm{Nop}' = \{q_k \mid [q_i, j, 0, q_k] \in \delta \text{ for some } j \in \{0, 1\}, \text{ and } q_i \in Q\}$$
$$\mathrm{Test}' = \{q_k \mid [q_i, j, s, q_k] \in \delta \text{ for some } j \in \{0, 1\}, s \in \{Z, P\}, \text{ and } q_i \in Q\}$$

For each $q_k \in (\mathrm{Inc}' \cup \mathrm{Dec}' \cup \mathrm{Nop}')$ there is exactly one quadruple containing q_k as the fourth element, since M is reversible (thus the quadruple is of the form $[q_i, j, d, q_k]$ ($q_i \in Q, j \in \{0, 1\}$, $d \in \{-, 0, +\}$)). Hence, the rules in (6.x), (7.x), and (8.x) corresponding to this quadruple have different right-hand sides from the others. Next, for each $q_k \in \mathrm{Test}'$, there are at most two quadruples containing q_k as the fourth element since M is reversible. They are of the form $[q_i, j, s, q_k]$ ($q_i \in Q, j \in \{0, 1\}, s \in \{Z, P\}$). If there is only one, the rules in (9.x) corresponding to this quadruple satisfy the reversibility constraint as above. In the case there are two, they must be of the forms $[q_i, j, Z, q_k]$ and $[q_i', j', P, q_k]$, because M is reversible. We can see the rules in (9.1)–(9.3), and (9.4)–(9.5) corresponding to the two rules have mutually different right-hand sides, because the center part of the cell 0 should be different between two cases of the contents of the counter j. They also differs from the other rules.

By above, each rule in g has different right-hand side from the others, and thus we can conclude that if M is reversible, A is also reversible. $\square$

Example 3.1 Consider a deterministic RCM(2) $M_2 = (2, Q, \{Z, P\}, \delta, q_0, q_6)$ having the following quadruples as δ.

$$
\begin{array}{llll}
[q_0, & 1, & Z, & q_1] \\
[q_1, & 0, & Z, & q_6] \\
[q_1, & 0, & P, & q_2] \\
[q_2, & 0, & -, & q_3]
\end{array}
\qquad
\begin{array}{llll}
[q_3, & 1, & +, & q_4] \\
[q_4, & 1, & +, & q_5] \\
[q_5, & 1, & P, & q_1]
\end{array}
$$

M_2 performs the computation $(q_0, n, 0) \vdash^*_{M_2} (q_6, 0, 2n)$ for any $n(= 0, 1, \cdots)$. An NC-RPCA A_2 constructed from M_2 by the method given in Theorem 3.1 is as follows.

$$A_2 = (\mathbf{Z}, \mathbf{N}_{25}^3, g_2, (0, 0, 0))$$

The local function g_2 is defined by the following set of rules, and a simulation process of $(q_0, 2, 0) \vdash^{12}_{M_2} (q_6, 0, 4)$ is shown in Fig. 4.

Rules by (1) ($x \in \{0, 1, \cdots, 25\}$) :
$[\,0,\ x,\ 0\,] \rightarrow [\,0,\ x,\ 0\,]$
Rules by (2) ($y \in \{10, 11, \cdots, 16\}$) :
$[\,0,\ 0,\ y\,] \rightarrow [\,0,\ 0,\ y\,]$
Rules by (3.1):
$[\,2,\ 0,\ 0\,] \rightarrow [\,0,\ 0,\ 2\,]$
$[\,2,\ 2,\ 0\,] \rightarrow [\,0,\ 2,\ 2\,]$
$[\,6,\ 0,\ 0\,] \rightarrow [\,0,\ 0,\ 6\,]$
$[\,6,\ 1,\ 0\,] \rightarrow [\,0,\ 1,\ 6\,]$
Rules by (3.2):
$[\,2,\ 1,\ 0\,] \rightarrow [\,0,\ 0,\ 3\,]$
$[\,2,\ 3,\ 0\,] \rightarrow [\,0,\ 2,\ 3\,]$
$[\,6,\ 2,\ 0\,] \rightarrow [\,0,\ 0,\ 8\,]$
$[\,6,\ 3,\ 0\,] \rightarrow [\,0,\ 1,\ 8\,]$
Rules by (3.3):
$[\,3,\ 0,\ 0\,] \rightarrow [\,2,\ 1,\ 0\,]$
$[\,3,\ 2,\ 0\,] \rightarrow [\,2,\ 3,\ 0\,]$
$[\,8,\ 0,\ 0\,] \rightarrow [\,6,\ 2,\ 0\,]$
$[\,8,\ 1,\ 0\,] \rightarrow [\,6,\ 3,\ 0\,]$
Rules by (3.4):
$[\,0,\ 0,\ 2\,] \rightarrow [\,2,\ 0,\ 0\,]$
$[\,0,\ 2,\ 2\,] \rightarrow [\,2,\ 2,\ 0\,]$
$[\,0,\ 0,\ 6\,] \rightarrow [\,6,\ 0,\ 0\,]$
$[\,0,\ 1,\ 6\,] \rightarrow [\,6,\ 1,\ 0\,]$
Rules by (4.1):
$[\,4,\ 0,\ 0\,] \rightarrow [\,0,\ 0,\ 4\,]$
$[\,4,\ 2,\ 0\,] \rightarrow [\,0,\ 2,\ 4\,]$
$[\,7,\ 0,\ 0\,] \rightarrow [\,0,\ 0,\ 7\,]$
$[\,7,\ 1,\ 0\,] \rightarrow [\,0,\ 1,\ 7\,]$
Rules by (4.2):
$[\,4,\ 1,\ 0\,] \rightarrow [\,5,\ 0,\ 0\,]$
$[\,4,\ 3,\ 0\,] \rightarrow [\,5,\ 2,\ 0\,]$
$[\,7,\ 2,\ 0\,] \rightarrow [\,9,\ 0,\ 0\,]$
$[\,7,\ 3,\ 0\,] \rightarrow [\,9,\ 1,\ 0\,]$
Rules by (4.3):
$[\,0,\ 0,\ 5\,] \rightarrow [\,4,\ 1,\ 0\,]$
$[\,0,\ 2,\ 5\,] \rightarrow [\,4,\ 3,\ 0\,]$
$[\,0,\ 0,\ 9\,] \rightarrow [\,7,\ 2,\ 0\,]$
$[\,0,\ 1,\ 9\,] \rightarrow [\,7,\ 3,\ 0\,]$
Rules by (4.4):
$[\,0,\ 0,\ 4\,] \rightarrow [\,4,\ 0,\ 0\,]$
$[\,0,\ 2,\ 4\,] \rightarrow [\,4,\ 2,\ 0\,]$
$[\,0,\ 0,\ 7\,] \rightarrow [\,7,\ 0,\ 0\,]$
$[\,0,\ 1,\ 7\,] \rightarrow [\,7,\ 1,\ 0\,]$

Rules by (5) for $[q2, 0, -, q3]$:
$[\,12,\ 7,\ 0\,] \rightarrow [\,12,\ 7,\ 0\,]$
$[\,12,\ 9,\ 0\,] \rightarrow [\,12,\ 9,\ 0\,]$
Rules by (5) for $[q3, 1, +, q4]$:
$[\,13,\ 4,\ 0\,] \rightarrow [\,13,\ 4,\ 0\,]$
$[\,13,\ 5,\ 0\,] \rightarrow [\,13,\ 5,\ 0\,]$
Rules by (5) for $[q4, 1, +, q5]$:
$[\,14,\ 3,\ 0\,] \rightarrow [\,14,\ 3,\ 0\,]$
$[\,14,\ 4,\ 0\,] \rightarrow [\,14,\ 4,\ 0\,]$
Rules by (6.1) for $[q3, 1, +, q4]$:
$[\,13,\ 4,\ 6\,] \rightarrow [\,14,\ 3,\ 6\,]$
$[\,13,\ 5,\ 6\,] \rightarrow [\,14,\ 4,\ 6\,]$
Rules by (6.1) for $[q4, 1, +, q5]$:
$[\,14,\ 3,\ 6\,] \rightarrow [\,15,\ 8,\ 0\,]$
$[\,14,\ 4,\ 6\,] \rightarrow [\,15,\ 9,\ 0\,]$
Rules by (7.3) for $[q2, 0, -, q3]$:
$[\,12,\ 7,\ 4\,] \rightarrow [\,13,\ 4,\ 6\,]$
$[\,12,\ 7,\ 5\,] \rightarrow [\,13,\ 5,\ 6\,]$
$[\,12,\ 9,\ 4\,] \rightarrow [\,13,\ 4,\ 8\,]$
$[\,12,\ 9,\ 5\,] \rightarrow [\,13,\ 5,\ 8\,]$
Rules by (9.1) for $[q0, 1, Z, q1]$:
$[\,10,\ 15,\ 0\,] \rightarrow [\,11,\ 14,\ 0\,]$
$[\,10,\ 16,\ 0\,] \rightarrow [\,11,\ 15,\ 0\,]$
Rules by (9.1) for $[q1, 0, Z, q6]$:
$[\,11,\ 13,\ 0\,] \rightarrow [\,16,\ 8,\ 0\,]$
$[\,11,\ 15,\ 0\,] \rightarrow [\,16,\ 10,\ 0\,]$
Rules by (9.4) for $[q1, 0, P, q2]$:
$[\,11,\ 12,\ 0\,] \rightarrow [\,12,\ 7,\ 4\,]$
$[\,11,\ 14,\ 0\,] \rightarrow [\,12,\ 9,\ 4\,]$
Rules by (9.4) for $[q5, 1, P, q1]$:
$[\,15,\ 8,\ 0\,] \rightarrow [\,11,\ 12,\ 0\,]$
$[\,15,\ 9,\ 0\,] \rightarrow [\,11,\ 13,\ 0\,]$
Rules by (10.1) and (10.2):
$[\,1,\ 0,\ 0\,] \rightarrow [\,0,\ 0,\ 1\,]$
$[\,0,\ 0,\ 1\,] \rightarrow [\,1,\ 0,\ 0\,]$
Rules by (10.3):
$[\,1,\ 22,\ 0\,] \rightarrow [\,10,\ 13,\ 0\,]$
$[\,1,\ 23,\ 0\,] \rightarrow [\,10,\ 14,\ 0\,]$
$[\,1,\ 24,\ 0\,] \rightarrow [\,10,\ 15,\ 0\,]$
$[\,1,\ 25,\ 0\,] \rightarrow [\,10,\ 16,\ 0\,]$
Rules by (10.4):
$[\,16,\ 7,\ 0\,] \rightarrow [\,1,\ 22,\ 0\,]$
$[\,16,\ 8,\ 0\,] \rightarrow [\,1,\ 23,\ 0\,]$
$[\,16,\ 9,\ 0\,] \rightarrow [\,1,\ 24,\ 0\,]$
$[\,16,\ 10,\ 0\,] \rightarrow [\,1,\ 25,\ 0\,]$

t \ cell	-1	0	1	2	3	4	
-1	1	24		1			
0		10 15		1			$(q_0,2,0)$
1	10	15		1			
2		11 14		1			$(q_1,2,0)$
3	11	14		1			
4		12 9 4		1			$(q_2,2,0)$
5	12	9	4	1			
6		12 9		5			
7	12	9	4 1				
8		13 4 8	1				$(q_3,1,0)$
9	13	4	6 3				
10		14 3 6	3				$(q_4,1,1)$
11	14	3	1 8				
12		14 3	1	6 2			
13	14	3	6 1	2			
14		15 8	1	2			$(q_5,1,2)$
15	15	8	1	2			
16		11 12	1	2			$(q_1,1,2)$
17	11	12	1	2			
18		12 7 4	1	2			$(q_2,1,2)$
19	12	7	5	2			
20		13 5 6		2			$(q_3,0,2)$
21	13	5	6	2			
22		13 5			8		
23	13	5			6 2		
24		13 5		6	2		
25	13	5	6		2		
26		14 4 6			2		$(q_4,0,3)$
27	14	4	6		2		
28		14 4		6	2		
29	14	4			8		
30		14 4				6 2	
31	14	4			6	2	
32		14 4		6		2	
33	14	4	6			2	
34		15 9				2	$(q_5,0,4)$
35	15	9				2	
36		11 13				2	$(q_1,0,4)$
37	11	13				2	
38		16 8				2	$(q_6,0,4)$
39	16	8				2	
40		1 23				2	

Figure 4: A simulation process of an RCM(2) M_2 by an NC-RPCA A_2.

From Lemma 3.1 and Proposition 3.1, universality of an NC-RPCA is concluded.

Corollary 3.1 An NC-RPCA is computation-universal.

4 Concluding Remarks

In this paper, we proved that an NC-RPCA can simulate a reversible two-counter machine, hence it is computation-universal. In [10], universality of an RCA (not necessarily number-conserving) has been shown by simulating a one-tape reversible Turing machine by an RCA. It is also possible to simulate a one-tape reversible Turing machine by an NC-RPCA. But, if we employ a simulation method in which the contents of each tape square are stored in each cell, then the quiescent state of the NC-RPCA should be $(0, m, 0)$ for some $m > 0$ rather than $(0, 0, 0)$.

References

[1] Albert, J., and Culik II, K., A simple universal cellular automaton and its one-way and totalistic version, *Complex Systems*, **1**, 1–16 (1987).

[2] Bennett, C.H., Logical reversibility of computation, *IBM J. Res. Dev.*, **17**, 525–532 (1973).

[3] Bennett, C.H., Notes on the history of reversible computation, *IBM J. Res. Dev.*, **32**, 16–23 (1988).

[4] Feynman, R.P., *Feynman Lectures on Computation* (eds. A.J.G.Hey, and R.W.Allen), Perseus Books, Reading, Mass. (1996). (邦訳：「ファインマン計算機科学」, 岩波書店 (1999).)

[5] Fredkin, E., and Toffoli, T., Conservative logic, *Int. J. Theoret. Phys.*, **21**, 219–253 (1982).

[6] Goles, E., Sand pile automata, *Ann. Inst. Henri Poincaré*, **56**, 75–90 (1992).

[7] Goles, E., and Margenstern, M., Sand pile as a universal computer, *Int. J. Modern Physics C*, **7**, 113–122 (1996).

[8] Imai, K., and Morita, K., A computation-universal two-dimensional 8-state triangular reversible cellular automaton, *Proc. of the Second Colloquium on Universal Machines and Computations*, Volume II (Metz), 90–99 (1998). (to appear in *Theoret. Comput. Sci.*)

[9] Margolus, N., Physics-like model of computation, *Physica*, **10D**, 81–95 (1984).

[10] Morita, K., and Harao, M., Computation universality of one-dimensional reversible (injective) cellular automata, *Trans. IEICE Japan*, **E-72**, 758–762 (1989).

[11] Morita, K., and Ueno, S., Computation-universal models of two-dimensional 16-state reversible cellular automata, *IEICE Trans. Inf. & Syst.*, **E75-D**, 141–147 (1992).

[12] Morita, K., Computation-universality of one-dimensional one-way reversible cellular automata, *Inform. Process. Lett.*, **42**, 325–329 (1992).

[13] Morita, K., Reversible cellular automata (in Japanese), *J. Information Processing Society of Japan*, **35**, 315–321 (1994).

[14] Morita, K., Universality of a reversible two-counter machine, *Theoret. Comput. Sci.*, **168**, 303–320 (1996).

[15] Morita, K., and Imai, K., Self-reproduction in a reversible cellular space, *Theoret. Comput. Sci.*, **168**, 337–366 (1996).

[16] Morita, K., Margenstern, M., and Imai, K., Universality of reversible hexagonal cellular automata, MFCS'98 Workshop on Frontiers between Decidability and Undecidability, Brno (1998).

[17] Morita, K., Cellular automata and artificial life – Computation and life in reversible cellular automata –, *Proc. of the 6th Summer School on Complex Systems*, Santiago, 1–40 (1998).

[18] Toffoli, J., Computation and construction universality of reversible cellular automata, *J. Comput. Syst. Sci.*, **15**, 213–231 (1977).

[19] Toffoli, T., and Margolus, N., Invertible cellular automata: a review, *Physica D*, **45**, 229–253 (1990).