Small Universal Reversible Counter Machines

Artiom Alhazov, Sergey Verlan, Rudolf Freund

Small Universal Reversible Counter Machines

Artiom Alhazov, Sergey Verlan, and Rudolf Freund

Abstract A k-counter machine (CM(k)) is an automaton with k counters as an auxiliary memory. It is known that CM(k) are universal for $k \geq 2$. As shown by Morita reversible CM(2) are universal. Based on results from Korec we construct four small universal reversible counter machines highlighting different trade-offs: $(10, 109, 129)$, $(11, 227, 270)$, $(9, 97, 116)$ and $(2, 1097, 1568)$, where in parentheses we indicated the number of counters, states and instructions, respectively. Since counter machines are used in many areas, our results can be the starting point for corresponding reversible universal constructions.

1 Introduction

Universality is a fundamental concept in the theory of computation. The question of finding a universal computing device in the class of Turing machines was originally proposed by A. Turing himself in [17]. A universal Turing machine would be capable of simulating any other Turing machine $\mathcal{T}$: given a description of $\mathcal{T}$ and the encoding of the input tape contents, the universal machine would halt with tape contents which would correspond to the encoding of the output of $\mathcal{T}$ for the supplied input.

In a more general setting of an arbitrary class $\mathfrak{C}$ of computing devices, the universality problem consists in finding such a *fixed* element $\mathcal{M}_0$ which would be able to simulate any other element $\mathcal{M} \in \mathfrak{C}$. More formally, if the result of running $\mathcal{M}$

Artiom Alhazov
Institute of Mathematics and Computer Science, Academy of Sciences of Moldova, Str. Academiei 5, Chişinău, MD-2028, Moldova, e-mail: artiom@math.md

Sergey Verlan
Université Paris Est, LACL (EA 4219), UPEC, F-94010, Créteil, France e-mail: verlan@u-pec.fr

Rudolf Freund
Faculty of Informatics, Vienna University of Technology, Favoritenstr. 9, 1040 Vienna, Austria, e-mail: rudi@emcc.at

with the input x is y (usually written as $\mathcal{M}(x) = y$), then $y = f(\mathcal{M}_0(g(\mathcal{M}'), h(x))$, where g is the function enumerating $\mathfrak{C}$, while f and h are the decoding and encoding functions respectively. We remark that in some cases it can be possible to have $\mathcal{M}_0 \in \mathfrak{C}$; then the input is the couple encoding of $g(\mathcal{M}')$ and x (e.g. using the Cantor pairing function). It is generally agreed that f and h should not be "too" complicated. Since it is relatively common to rely on exponential coding when working with devices computing numbers, the functions $f(x) = \log_a(x)$ and $h(x) = b^x$, for some $a, b \in \mathbb{N}$ are often used (cf. [9, 19]).

In this paper, we will adhere to the terminology established by I. Korec in [7] and call the element $\mathcal{M}_0$ defined as above *weakly* universal (or just universal). In case the functions f and h are additionally required to be identities, $\mathcal{M}_0$ will be referred to as *strongly* universal. Hence, the strong universality permits to capture the situations when the encoding does not alter the power of the device. For example, 2-register machines are weakly universal [9], but they cannot be strongly universal as they cannot compute even the square function [3, 15].

As a further development on the question of universality, C. Shannon [16] considered finding the *smallest* possible universal Turing machine, where the size is essentially given by the sizes of the alphabets of symbols and states. A series of important results concerning this direction were obtained [8, 14, 18]. For an overview of the recent results the reader is referred to [13]. Small universal devices are of considerable theoretical importance since they indicate the minimal choice ingredients sufficient for achieving computational completeness.

A k-counter machine (CM(k)) is an automaton with k counters that can hold non-negative values. In one step, the finite-state control of CM(k) can increment or decrement the contents by one or test whether it is zero or not. A related model is the register machine [9], having eventual restrictions on the form of the control and a richer set of potential instructions. However, the common variant of register machines is almost identical to CMs. Register machines and hence CMs were shown to be universal and it is known that already three registers/counters suffice for strong universality and two for the weak one [6, 9].

In [10] K. Morita studied reversible CMs and showed the universality of reversible CM(2). These machines are backward deterministic, *i.e.*, each configuration has at most one predecessor. This research lines up in the study of other reversible systems such as reversible Turing machines, reversible cellular automata and reversible logic gates, see [5, 11] for a general survey. We remark that in case of reversible machines, the notion of universality is slightly different than in the classical case [2].

In 1996, I. Korec described a number of universal register machines with considerably fewer instructions than were known to be needed for universality before [7]. Based on this result small 2- and 3-register machines were constructed [1].

In this paper we consider the construction of small universal reversible counter machines. As in [7, 12, 1] we are mainly interested in the number of instructions as well as in the trade-offs between this number an the number of counters. We construct four small universal reversible counter machines highlighting different trade-offs: $(10, 109, 129)$, $(11, 227, 270)$, $(9, 97, 116)$ and $(2, 1097, 1568)$, where in

parentheses we indicated the number of counters, the number of states and the number of instructions, respectively.

2 Definitions

We now recall the formal definition of CM given in [10]. We denote by $\mathbb{N}$ the set of all non-negative integers.

Definition 1. A k-counter machine (CM(k)) is the 5-tuple $M = (k, Q, \delta, q_0, q_f)$, where k is the number of counters, Q is a nonempty finite set of states, $q_0 \in Q$ is the initial state, $q_f \in Q$ is the final (halting) state and δ is the move relation, which is a subset of $Q \times \{1, \ldots, k\} \times \{Z, P\} \times Q \cup Q \times \{1, \ldots, k\} \times \{-, 0, +\} \times Q$.

We will use the notation *Ri* to denote the counter i.

Definition 2. An instantaneous description of a CM(k) $M = (k, Q, \delta, q_0, q_f)$ is a $k+1$-tuple $(q, n_1, \ldots, n_k) \in Q \times \mathbb{N}^k$.

The transition relation $\vdash$ is defined as follows:

$$(q, n_1, \ldots, n_i, \ldots, n_k) \vdash (q, n_1, \ldots, n_i', \ldots, n_k)$$

iff one of the following conditions is satisfied:

1. $[q, i, Z, q'] \in \delta$ and $n_i = n_i' = 0$ (the zero test instruction).
2. $[q, i, P, q'] \in \delta$ and $n_i = n_i' > 0$ (the non-zero instruction).
3. $[q, i, -, q'] \in \delta$ and $n_i - 1 = n_i'$ (the minus instruction).
4. $[q, i, 0, q'] \in \delta$ and $n_i = n_i'$ (the jump instruction).
5. $[q, i, +, q'] \in \delta$ and $n_i + 1 = n_i'$ (the plus instruction).

In order to define the computation of the counter machine we need to consider input and output counters. Without losing the generality, we may assume that the input counters are numbered from 1 to i and the output ones are numbered from j to l. Then the result of the computation of M on the vector $(n_1, \ldots, n_i)$ can be defined as follows:

$$M(n_1, \ldots, n_i) = \{(n_j', \ldots, n_l') \mid (q_0, n_1, \ldots, n_i, 0, \ldots, 0) \vdash^* (q_f, n_1', \ldots, n_k')\}.$$

According to Korec [7], a machine $\mathcal{M}$ is (weakly) universal if there exist recursive functions h and g such that for any machine M we have

$$M(x) = f(\mathcal{M}(\#_M, h(x))), \text{ where } \#_M \text{ is the number of } M \text{ in some enumeration.}$$

A machine is said to be strongly universal if f and h are identities.

In [7] it was shown that there exist a strongly universal register machine U_{32} with 32 instructions and a weakly universal register machine with 29 instructions. While

the model of register machine used in [7] is slightly different from Definition 1, there is no difficulty in translating it to/from the form used in this paper. Moreover, this translation keeps the same number of states and basically the same number of instructions (with the remark that zero-test instructions from register machines correspond to two instructions in the counter machine). We give below the corresponding list of instructions. Note that this translation adds two additional states that are not considered in [7] for technical reasons: q_0 and q_f corresponding to the start and the end state respectively. Hence, the resulting machine U_{34} has 46 instructions. Notice also that U_{32} is numbering registers from 0 to 7. We recall that the code of the simulated machine is initially stored in $R1$, the initial value in $R2$ and the result is obtained in $R0$. At the end of the computation all values of counters $R3$–$R7$ are bounded.

$[q_1, 1, P, q_2]$	$[q_1, 1, Z, q_6]$	$[q_2, 1, -, q_3]$	$[q_3, 7, +, q_1]$
$[q_4, 5, P, q_5]$	$[q_4, 5, Z, q_7]$	$[q_5, 5, -, q_6]$	$[q_6, 6, +, q_4]$
$[q_7, 6, P, q_8]$	$[q_7, 6, Z, q_4]$	$[q_8, 6, -, q_9]$	$[q_9, 5, +, q_{10}]$
$[q_{10}, 7, P, q_{11}]$	$[q_{10}, 7, Z, q_{13}]$	$[q_{11}, 7, -, q_{12}]$	$[q_{12}, 1, +, q_7]$
$[q_{13}, 6, P, q_{14}]$	$[q_{13}, 6, Z, q_1]$	$[q_{14}, 4, P, q_{15}]$	$[q_{14}, 4, Z, q_{16}]$
$[q_{15}, 4, -, q_1]$	$[q_{16}, 5, P, q_{17}]$	$[q_{16}, 5, Z, q_{23}]$	$[q_{17}, 5, -, q_{18}]$
$[q_{18}, 5, P, q_{19}]$	$[q_{18}, 5, Z, q_{27}]$	$[q_{19}, 5, -, q_{20}]$	$[q_0, 1, 0, q_1]$
$[q_{20}, 5, P, q_{21}]$	$[q_{20}, 5, Z, q_{30}]$	$[q_{21}, 5, -, q_{22}]$	$[q_{22}, 4, +, q_{16}]$
$[q_{23}, 2, P, q_{24}]$	$[q_{23}, 2, Z, q_{25}]$	$[q_{24}, 2, -, q_{32}]$	
$[q_{25}, 0, P, q_{26}]$	$[q_{25}, 0, Z, q_{32}]$	$[q_{26}, 0, -, q_1]$	
$[q_{27}, 3, P, q_{28}]$	$[q_{27}, 3, Z, q_{29}]$	$[q_{28}, 3, -, q_{32}]$	$[q_{29}, 0, +, q_1]$
$[q_{30}, 2, +, q_{31}]$	$[q_{31}, 3, +, q_{32}]$	$[q_{32}, 4, P, q_{15}]$	$[q_{32}, 4, Z, q_f]$

As for register machines, counter machines can be represented in a graphical manner as a graph whose nodes are labeled by elements from Q and having a directed edge going from q to q' labeled by iX if $[q, i, X, q'] \in \delta$, see e.g. Fig. 1.

A CM M is said to be deterministic if for any pair of instructions $[p, i, X, p']$ and $[q, j, Y, q']$ from δ it holds

$$p \neq q \ \vee \ (i = j \ \wedge \ X \neq Y \ \wedge \ X, Y \notin \{-, 0, +\}) .$$

A CM M is said to be reversible if if for any pair of instructions $[p, i, X, p']$ and $[q, j, X, q']$ from δ it holds

$$p' \neq q' \ \vee \ (i = j \ \wedge \ X \neq Y \ \wedge \ X, Y \notin \{-, 0, +\}).$$

In the graphical form the deterministic property implies that each node has at most two outgoing arcs. In the case of two arcs, both of them should correspond to the zero and non-zero test of the same counter. Since U_{32} is deterministic, it is not surprising that U_{34} is deterministic too.

Similarly, the reversible property implies that each node has at most two incoming arcs. As in the deterministic case, when there are two incoming arcs then both of them should correspond to the zero and non-zero test of the same counter. We will call *non-reversible* a node (state) that is not fulfilling this property.

We also recall the following results from [10]:

Theorem 1 ([10], Theorem 3.1). *For any deterministic CM(k) $M = (k, Q, \delta, q_0, q_f)$, there is a deterministic reversible CM(k + 2) $M' = (k + 2, Q', \delta', q_0, q_f)$ such that*

$$\begin{aligned} (q_0, m_1, \ldots, m_k) \vdash_M^* (q_f, n_1, \ldots, n_k) \quad \textit{iff} \\ \exists h \in \mathbb{N} \quad (q_0, m_1, \ldots, m_k, 0, 0) \vdash_{M'}^* (q_f, n_1, \ldots, n_k, h, 0) \end{aligned}$$

holds for all $m_1, \ldots, m_k, n_1, \ldots, n_k \in \mathbb{N}$.

The proof of above theorem produces the history of the computation, which is recorded in the value h using a method from [4]. Obviously, this value is not known in advance and it is not bounded. The next theorem shows that using k additional counters it is possible to bound it, hence obtaining each time a "clean" computation where only the value of the output is not bounded in advance. We should call such machines *garbage-less*.

Theorem 2 ([10], Theorem 3.2). *For any deterministic CM(k) $M = (k, Q, \delta, q_0, q_f)$, there is a deterministic reversible CM(2k + 2) $M' = (2k + 2, Q', \delta', q_0, q'_f)$ such that*

$$\begin{aligned} (q_0, m_1, \ldots, m_k) \vdash_M^* (q_f, n_1, \ldots, n_k) \quad \textit{iff} \\ (q_0, m_1, \ldots, m_k, 0, \ldots, 0) \vdash_{M'}^* (q'_f, m_k, \ldots, m_k, 0, 0, n_1, \ldots, n_k) \end{aligned}$$

holds for all $m_1, \ldots, m_k, n_1, \ldots, n_k \in \mathbb{N}$.

Next theorem shows that as in [9] any number of counters can be packed into two in a reversible manner.

Theorem 3 ([10], Theorem 4.1). *For any deterministic CM(k) $M = (k, Q, \delta, q_0, q_f)$, there is a deterministic reversible CM(2) $M' = (2, Q', \delta', q_0, q_f)$ such that*

$$\begin{aligned} (q_0, m_1, \ldots, m_k) \vdash_M^* (q_f, n_1, \ldots, n_k) \quad \textit{iff} \\ (q_0, p_1^{m_1} \ldots p_k^{m_k}, 0) \vdash_{M'}^* (q_f, p_1^{n_1} \ldots p_k^{n_k}, 0) \end{aligned}$$

holds for all $m_1, \ldots, m_k, n_1, \ldots, n_k \in \mathbb{N}$, where p_i denotes the ith prime number.

3 Strong Universality

In this section we will construct several small universal reversible counter machines. We start by analyzing the machine U_{34}. It can be easily seen that U_{34} has the following non-reversible states:

$$q_1, q_4, q_6, q_7, q_{15}, q_{16}, q_{32}.$$

States q_1 and q_{32} are non-reversible because of multiple incoming arcs (6 and 4 respectively). The other states are non-reversible because the incoming arcs do not correspond to opposite checks of the same counter.

We start by reducing the number of incoming arcs to each state to at most four. This is performed by adding two additional states as in [10], Lemma 3.1. We remark that added states are non-reversible. This yields the following machine U_{36}, see also Fig. 1 (we emphasized in bold the differences with respect to the U_{34} machine):

$[q_1, 1, P, q_2]$	$[q_1, 1, Z, q_6]$	$[q_2, 1, -, q_3]$	$[q_3, 7, +, \mathbf{q_{34}}]$
$[q_4, 5, P, q_5]$	$[q_4, 5, Z, q_7]$	$[q_5, 5, -, q_6]$	$[q_6, 6, +, q_4]$
$[q_7, 6, P, q_8]$	$[q_7, 6, Z, q_4]$	$[q_8, 6, -, q_9]$	$[q_9, 5, +, q_{10}]$
$[q_{10}, 7, P, q_{11}]$	$[q_{10}, 7, Z, q_{13}]$	$[q_{11}, 7, -, q_{12}]$	$[q_{12}, 1, +, q_7]$
$[q_{13}, 6, P, q_{14}]$	$[q_{13}, 6, Z, \mathbf{q_{33}}]$	$[q_{14}, 4, P, q_{15}]$	$[q_{14}, 4, Z, q_{16}]$
$[q_{15}, 4, -, \mathbf{q_{33}}]$	$[q_{16}, 5, P, q_{17}]$	$[q_{16}, 5, Z, q_{23}]$	$[q_{17}, 5, -, q_{18}]$
$[q_{18}, 5, P, q_{19}]$	$[q_{18}, 5, Z, q_{27}]$	$[q_{19}, 5, -, q_{20}]$	$[q_0, 1, 0, q_1]$
$[q_{20}, 5, P, q_{21}]$	$[q_{20}, 5, Z, q_{30}]$	$[q_{21}, 5, -, q_{22}]$	$[q_{22}, 4, +, q_{16}]$
$[q_{23}, 2, P, q_{24}]$	$[q_{23}, 2, Z, q_{25}]$	$[q_{24}, 2, -, q_{32}]$	$[\mathbf{q_{34}}, \mathbf{1}, \mathbf{0}, \mathbf{q_1}]$
$[q_{25}, 0, P, q_{26}]$	$[q_{25}, 0, Z, q_{32}]$	$[q_{26}, 0, -, \mathbf{q_{33}}]$	$[\mathbf{q_{33}}, \mathbf{1}, \mathbf{0}, \mathbf{q_{34}}]$
$[q_{27}, 3, P, q_{28}]$	$[q_{27}, 3, Z, q_{29}]$	$[q_{28}, 3, -, q_{32}]$	$[q_{29}, 0, +, \mathbf{q_{33}}]$
$[q_{30}, 2, +, q_{31}]$	$[q_{31}, 3, +, q_{32}]$	$[q_{32}, 4, P, q_{15}]$	$[q_{32}, 4, Z, q_f]$

Note that it was possible to reduce in the same manner the number of incoming arcs to two. Then it is possible to use the construction from Theorem 1 in order to construct an equivalent reversible machine. In this case each pair of arcs (instructions) leading to a non-reversible state are replaced by 21 instructions and 16 additional states. A quick computation shows that using this method a strongly universal reversible machine with 273 instructions and 235 states is obtained.

We show below that for U_{36} a more efficient construction can be used. We will use a modified version of the technique from Theorem 1. We recall that the core of this proof is that in order to make non-reversible states reversible an additional counter is used to keep track of the computation history. More precisely, this counter stores a number whose bits keep track of which of the two incoming arcs was used to reach a non-reversible state. Another additional counter is needed for technical reasons.

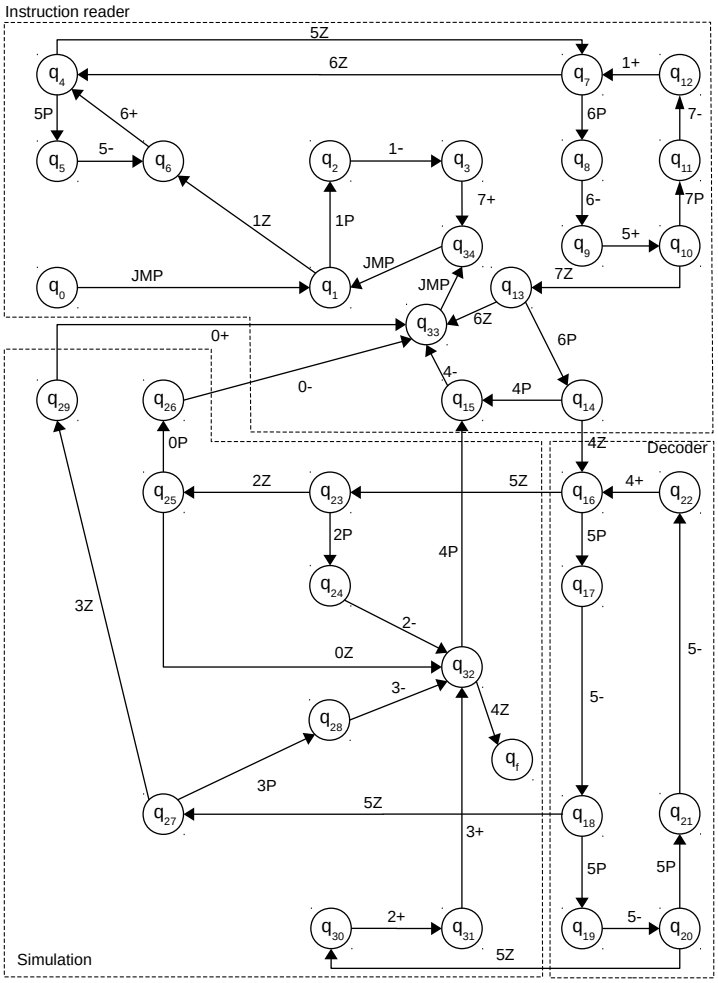

Fig. 1 Strongly universal counter machine U_{36}.

With the goal of minimization of the number of instructions and states we allow up to four incoming arcs to a node. Hence, our history will keep a base-4 representation of the used choice.

We start by the observation that in the case of state q_4 the incoming arcs are labeled by $6Z$ and $6+$. We introduce a new state q_4' and we replace the instruction $[q_6, 6, +, q_4]$ by two instructions:

$$[q_6, 6, +, q_4'],\ [q_4', 6, P, q_4]$$

A similar transformation is done for the state q_{16} (using counter $R4$).

Next, we observe that the two jump instructions at state q_1 can be replaced by the test of counter $R8$ that is supposed to store the history of the computation (recall that the construction from Theorem 1 adds two additional counters $R8$ and $R9$). Since at the beginning of the computation the history is empty (equal to zero) and after the first cycle returning to q_1 it is not empty, it is possible to correctly discriminate both cases:

$$[q_0, 8, Z, q_1],\ [q_{34}, 8, P, q_1]$$

Now we concentrate on the state q_{34}. On one branch the counter $R7$ is positive (because of the R7+ instruction). On the other branch counter $R7$ is always zero, because the last operation on $R7$ that is performed in order to reach q_{34} is the instruction $[q_{10}, 7, Z, q_{13}]$ that ensures that $R7$ is empty. Hence, it is possible to use the following instructions to make q_{34} reversible:

$$[q_3, 7, +, q_3'],\ [q_3', 7, P, q_{34}],\ [q_{33}, 7, Z, q_{34}]$$

Consider now the state q_{15}. Using the flowchart depicted at Fig. 1 it can be easily verified that the value of counter $R5$ can discriminate the two branches. Indeed, the instruction $[q_9, 5, +, q_{10}]$ ensures that $R5$ is positive when going to q_{15} from q_{14}. On the other hand, in order to reach q_{32} one has to pass through q_{23}, q_{27} or q_{30}. But this implies a zero test on $R5$. Hence, we can use following instructions to obtain the reversible behavior of q_{15}:

$$[q_{32}, 4, P, q_{32}'],\ [q_{32}', 5, Z, q_{15}],\ [q_{14}, 4, P, q_{14}'],\ [q_{14}', 5, P, q_{15}]$$

Thus, only 4 states (q_6, q_7, q_{32}, q_{33}) need to be made reversible. We describe below the procedure that allows to replace any node with at most four incoming edges by an equivalent reversible construction.

Consider a state q_t that has 4 incoming arcs. Let the corresponding instructions be $[q_{s_j}, i_{s_j}, X_j, q_t]$, $1 \leq j \leq 4$. Then consider the following rules (depicted also on Fig. 2):

$$[q_{s_j}, i_{s_j}, X_j, q_{s_{j1}}], \quad 1 \leq j \leq 4$$

$$\begin{array}{llll}
[q_{s_{11}}, 9, Z, q_{t_{10}}] & [q_{s_{21}}, 9, Z, q_{t_8}] & [q_{s_{31}}, 9, Z, q_{t_6}] & [q_{s_{41}}, 9, Z, q_{t_4}] \\
[q_{t_1}, 8, -, q_{t_2}] & [q_{t_2}, 9, +, q_{t_3}] & [q_{t_3}, 9, P, q_{t_4}] & [q_{t_4}, 9, +, q_{t_5}] \\
[q_{t_5}, 9, P, q_{t_6}] & [q_{t_6}, 9, +, q_{t_7}] & [q_{t_7}, 9, P, q_{t_8}] & [q_{t_8}, 9, +, q_{t_9}] \\
[q_{t_9}, 9, P, q_{t_{10}}] & [q_{t_{10}}, 8, P, q_{t_1}] & [q_{t_{10}}, 8, Z, q_{t_{11}}] & [q_{t_{11}}, 9, -, q_{t_{12}}] \\
[q_{t_{12}}, 8, +, q_{t_{13}}] & [q_{t_{13}}, 9, P, q_{t_{14}}] & [q_{t_{13}}, 9, Z, q_t] & [q_{t_{14}}, 8, P, q_{t_{11}}]
\end{array}$$

Clearly, the above rules allow to reach q_t in a reversible manner. We also observe that if a node has less than four incoming arcs, then for reversibility it is sufficient to delete unused q_{sj1} nodes.

Now considering all above constructions together it is possible to construct a strong universal reversible counter machine. Such a machine has 109 states (36

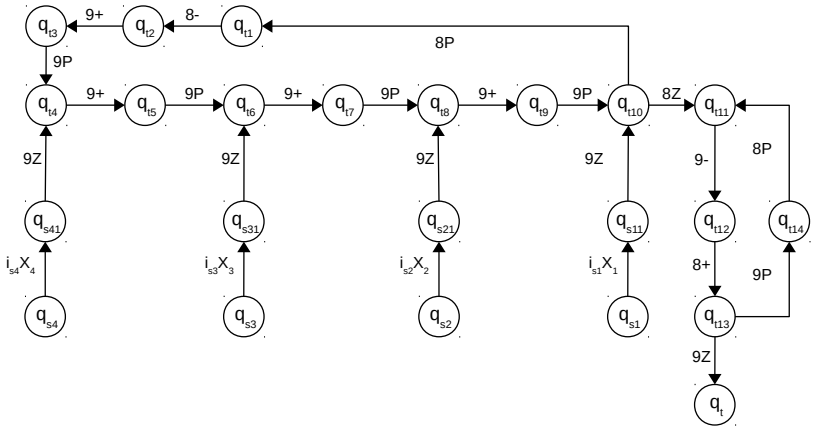

Fig. 2 Reversible junction of 4 instructions in q_t.

states from U_{36}, plus 5 additional states used for the reversibility of q_1, q_4, q_{15}, q_{16} and q_{34}, plus 2×18 states used for the reversibility of q_{32} and q_{33}, plus 2×16 states used for the reversibility of q_6 and q_7) and 129 instructions ($48 + 5 + 2 \times 20 + 2 \times 18$).

Theorem 4. *There exists a strongly universal reversible counter machine U_{109} with 10 counters, 109 states and 129 instructions.*

Now we will show how to bound the final value of non-output counters, *i.e.* obtain a garbage-less CM. We will use the construction from Theorem 2. This construction works as follows. Machine U_{109} is run leaving the history in $R8$. Next the following copy procedure is executed transferring the resulting value from $R0$ to $R10$. This is done by copying the value of $R0$ to $R9$ and then copying it back from $R9$ to $R0$ and $R10$.

$$
\begin{array}{llll}
[q_f, 9, Z, q_{c_1}] & [q_{c_1}, 0, Z, q_{c_5}] & [q_{c_1}, 0, P, q_{c_2}] & [q_{c_2}, 0, -, q_{c_3}] \\
[q_{c_3}, 9, +, q_{c_4}] & [q_{c_4}, 9, P, q_{c_1}] & [q_{c_5}, 9, Z, p_f] & [q_{c_5}, 9, P, q_{c_6}] \\
[q_{c_6}, 9, -, q_{c_7}] & [q_{c_7}, 0, +, q_{c_8}] & [q_{c_8}, 10, +, q_{c_9}] & [q_{c_9}, 0, P, q_{c_5}]
\end{array}
$$

Finally, the machine is run in a reverse manner (technically a copy of all rules with reverse operations should be provided, working on states where q replaced by p). This gives a total number of $109 \times 2 + 9 = 227$ states and $129 \times 2 + 12 = 270$ instructions.

Theorem 5. *There exists a strongly universal garbage-less reversible counter machine U_{227} with 11 counters, 227 states and 270 instructions.*

4 Weak universality

Now consider the weakly universal machine U_{31} (based on U_{29} from [7]). Below we give the list of rules of this machine. In fact, the only modification in this machine with respect to U_{31} concerns the simulation block, which is also depicted on Fig. 3. We remark that states q_{25}, q_{26} and q_{29} are absent. Below we bold emphasized corresponding changes.

$[q_1, 1, P, q_2]$	$[q_1, 1, Z, q_6]$	$[q_2, 1, -, q_3]$	$[q_3, 7, +, q_{38}]$
$[q_4, 5, P, q_5]$	$[q_4, 5, Z, q_7]$	$[q_5, 5, -, q_6]$	$[q_6, 6, +, q_4]$
$[q_7, 6, P, q_8]$	$[q_7, 6, Z, q_4]$	$[q_8, 6, -, q_9]$	$[q_9, 5, +, q_{10}]$
$[q_{10}, 7, P, q_{11}]$	$[q_{10}, 7, Z, q_{13}]$	$[q_{11}, 7, -, q_{12}]$	$[q_{12}, 1, +, q_7]$
$[q_{13}, 6, P, q_{14}]$	$[q_{13}, 6, Z, q_{37}]$	$[q_{14}, 4, P, q_{15}]$	$[q_{14}, 4, Z, q_{16}]$
$[q_{15}, 4, -, q_{36}]$	$[q_{16}, 5, P, q_{17}]$	$[q_{16}, 5, Z, q_{23}]$	$[q_{17}, 5, -, q_{18}]$
$[q_{18}, 5, P, q_{19}]$	$[q_{18}, 5, Z, q_{27}]$	$[q_{19}, 5, -, q_{20}]$	$[q_0, 1, 0, q_1]$
$[q_{20}, 5, P, q_{21}]$	$[q_{20}, 5, Z, q_{30}]$	$[q_{21}, 5, -, q_{22}]$	$[q_{22}, 4, +, q_{16}]$
$[q_{23}, \mathbf{0}, P, \mathbf{q_{24}}]$	$[q_{23}, \mathbf{0}, Z, \mathbf{q_1}]$	$[q_{24}, \mathbf{0}, -, q_{32}]$	
$[q_{27}, \mathbf{2}, P, q_{28}]$	$[q_{27}, \mathbf{2}, Z, \mathbf{q_1}]$	$[q_{28}, \mathbf{2}, -, q_{32}]$	
$[q_{30}, \mathbf{0}, +, q_{31}]$	$[q_{31}, \mathbf{2}, +, q_{32}]$	$[q_{32}, 4, P, q_{15}]$	$[q_{32}, 4, Z, q_f]$

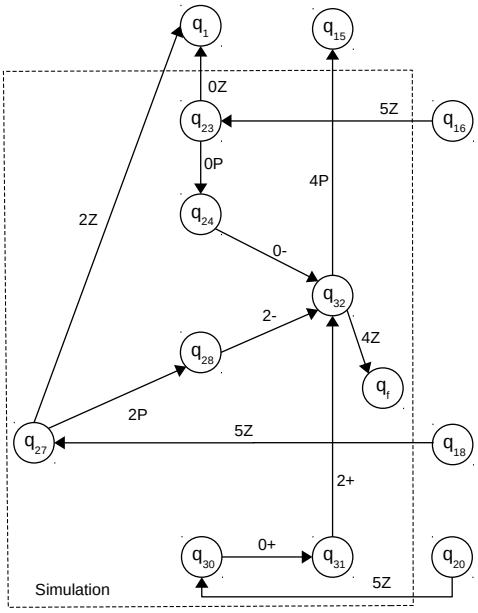

Fig. 3 Weakly universal counter machine U_{31}.

We observe that there are only 3 incoming arcs to q_{32}. In order to minimize the number of nodes and arcs we will construct a machine with at most 3 incoming arcs to each node. First add states q_{33} and q_{34} like in the case of U_{36}, see Fig. 1. Next, consider a new state q_{35} and add following rules:

$$[q_{33}, 6, P, q_{35}] \qquad [q_{13}, 6, Z, q_{35}] \qquad [q_{35}, 1, 0, q_{34}]$$

We remark that on the branch yielding to q_{33} the value of $R6$ is positive, because of the rule $[q_{13}, 6, P, q_{14}]$. Hence, q_{35} is reversible.

Now we remark that in order to make reversible a state with at most three incoming arcs we can use a slightly modified version of the flowchart from Fig. 2. In fact, states $q_{s_{41}}$, q_{t_3} and q_{t_4} should be removed and the arrow leading to q_{t_3} should lead now to q_{t_5}. This allows to store the incoming arc number in the history, interpreted as a base-3 number.

Hence we showed how it is possible to construct a weak universal reversible counter machine. This machine has 97 states ($33 + 6 + 2 \times 15 + 2 \times 14$) and 116 instructions ($44 + 6 + 2 \times 17 + 2 \times 16$).

Theorem 6. *There exists a weakly universal reversible counter machine U_{97} with* 9 *counters,* 97 *states and* 116 *instructions.*

Now we can use the construction from Theorem 3 to obtain a weakly universal reversible counter machine U with 2 counters. We refer to [10] for more details. In order to compute the parameters of U we recall that each jump instruction from U_{97} is performed by one instruction in U, each plus or minus instruction of Ri is performed by $p_i + 10$ instructions and $p_i + 7$ new states (p_i being the ith prime number). Each couple of zero and non-zero check on Ri instructions is performed by $3p_i + 3$ instructions using $2p_i + 2$ new states. A single zero check instruction is simulated using $3p_i + 2$ instructions and $2p_i + 1$ states, while a single non-zero check instruction is simulated using $2p_i + 4$ instructions and $p_i + 2$ states. To minimize the number of instructions we will use a different sequence of prime numbers associated to counters (trying to use smaller primes for more frequently used counters).

The table below gives the number of instructions of each type in U_{97}, as well as the chosen prime number.

i	+	−	P&Z	Z	P	p_i
0	1	1	1	0	0	23
1	1	1	1	0	0	19
2	1	1	1	0	0	17
4	1	1	2	0	1	11
5	1	4	4	1	1	5
6	1	1	2	0	2	7
7	1	1	1	1	1	13
8	4	4	4	1	5	3
9	10	4	4	12	12	2

Theorem 7. *There exists a weakly universal reversible counter machine* U_{1097} *with* 2 *counters,* 1097 *states and* 1568 *instructions.*

5 Conclusions

The table below summarizes the parameters of universal reversible counter machines constructed in this paper.

Counters	states	instructions	universality	garbage-less
10	109	129	strong	no
11	227	270	strong	yes
9	97	116	weak	no
2	1097	1568	weak	no

We remark that as noted by [2] there is no universal reversible counter machine in the strict sense, because any reversible machine computes an injective function and, obviously, the universal machine as defined above is not injective. In [2, 4] it is highlighted that corresponding constructions provide a machine for reversibly simulating any irreversible machine. Another solution provided in the paper above is to define the universal machine $\mathcal{M}$ as follows $\mathcal{M}(\#_M, x) = (\#_M, M(x))$, *i.e.*, the result of the computation of the universal machine on the pair consisting of the code of the simulated machine M and its input x is equal to the pair of the code of the simulated machine M and the result of its computation on the given input x. Since U_{32} keeps a copy of the code of the simulated machine in $R1$, we can deduce that the simulations that we provide are universal in the above sense.

There are several possible directions for further research. One of them is the investigation of different trade-offs between the number of counters, states and instructions. It can be particularly interesting to see if as in [7] the increase of the number of counters can lead to the decrease of the number of instructions. Another interesting direction is the investigation of the universality type and independency for different number of counters.

References

1. Alhazov, A., Ivanov, S., Pelz, E., Verlan, S.: Small universal deterministic Petri nets with inhibitor arcs. Journal of Automata, Languages and Combinatorics **21**(1-2), 7–26 (2016)
2. Axelsen, H.B., Glück, R.: What do reversible programs compute? In: M. Hofmann (ed.) Foundations of Software Science and Computational Structures - 14th International Conference, FOSSACS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings, *Lecture Notes in Computer Science*, vol. 6604, pp. 42–56. Springer (2011)

3. Barzdin, I.M.: Ob odnom klasse machin Turinga (machiny Minskogo), russian. Algebra i Logika **1**, 42–51 (1963)
4. Bennett, C.: Logical reversibility of computation. IBM Journal of Research and Development **17**, 525–532 (1973)
5. Bennett, C.H.: Notes on the history of reversible computation. IBM Journal of Research and Development **44**(1), 270–278 (2000)
6. Ivanov, S., Pelz, E., Verlan, S.: Small universal non-deterministic Petri nets with inhibitor arcs. In: H. Jürgensen, J. Karhumäki, A. Okhotin (eds.) Descriptional Complexity of Formal Systems - 16th International Workshop, DCFS 2014, Turku, Finland, August 5-8, 2014. Proceedings, *Lecture Notes in Computer Science*, vol. 8614, pp. 186–197. Springer (2014)
7. Korec, I.: Small universal register machines. Theoretical Computer Science **168**(2), 267–301 (1996)
8. Minsky, M.: Size and structure of universal Turing machines using tag systems. In: Recursive Function Theory: Proceedings, Symposium in Pure Mathematics, Provelence, vol. 5, pp. 229–238 (1962)
9. Minsky, M.: Computations: Finite and Infinite Machines. Prentice Hall, Englewood Cliffts, NJ (1967)
10. Morita, K.: Universality of a reversible two-counter machine. Theoretical Computer Science **168**(2), 303–320 (1996)
11. Morita, K.: Reversible cellular automata. In: G. Rozenberg, T. Bäck, J.N. Kok (eds.) Handbook of Natural Computing, pp. 231–257. Springer (2012)
12. Morita, K.: Universal reversible Turing machines with a small number of tape symbols. Fundam. Inform. **138**(1-2), 17–29 (2015)
13. Neary, T., Woods, D.: The complexity of small universal Turing machines: A survey. In: M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, G. Turán (eds.) SOFSEM 2012: 38th Conference on Current Trends in Theory and Practice of Computer Science, *Lecture Notes in Computer Science*, vol. 7147, pp. 385–405. Springer (2012)
14. Rogozhin, Y.: Small universal Turing machines. Theoretical Computer Science **168**(2), 215–240 (1996)
15. Schroeppel, R.: A two counter machine cannot calculate 2N. In: AI Memos. MIT AI Lab (1972)
16. Shannon, C.E.: A universal Turing machine with two internal states. Automata Studies, Annals of Mathematics Studies **34**, 157–165 (1956)
17. Turing, A.M.: On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society **42**(2), 230–265 (1936)
18. Watanabe, S.: 5-symbol 8-state and 5-symbol 6-state universal Turing machines. Journal of the ACM **8**(4), 476–483 (1961)
19. Woods, D., Neary, T.: The complexity of small universal Turing machines: A survey. Theoretical Computer Science **410**(4-5), 443–450 (2009)