Four small universal Turing machines

Turlough Neary[1] and Damien Woods[2]

[1] TASS, Department of Computer Science,
National University of Ireland Maynooth, Ireland. tneary@cs.may.ie
[2] Department of Computer Science,
University College Cork, Ireland. d.woods@cs.ucc.ie

Abstract. We present small polynomial time universal Turing machines with state-symbol pairs of $(5,5)$, $(6,4)$, $(9,3)$ and $(18,2)$. These machines simulate our new variant of tag system, the bi-tag system and are the smallest known universal Turing machines with 5, 4, 3 and 2-symbols respectively. Our 5-symbol machine uses the same number of instructions (22) as the smallest known universal Turing machine by Rogozhin.

1 Introduction

Shannon [16] was the first to consider the problem of finding the smallest possible universal Turing machine. In 1962 Minsky [7] created a 7-state, 4-symbol universal Turing machine that simulates Turing machines via 2-tag systems. Minsky's technique of 2-tag simulation was extended by Rogozhin [15] to create small universal Turing machines with state-symbol pairs of $(24,2)$, $(10,3)$, $(7,4)$, $(5,5)$, $(4,6)$, $(3,10)$ and UTM$(2,18)$. Subsequently some of these machines were reduced in size to give machines with state-symbol pairs of $(3,9)$ [5], $(19,2)$ [1] and $(7,4)$ [1]. Figure 1 is a state-symbol plot where the current smallest 2-tag simulators of Rogozhin et al. are plotted as circles.

Here we present universal Turing machines with state-symbol pairs of $(5,5)$, $(6,4)$, $(9,3)$ and $(18,2)$, the later two machine having previously appeared in [9]. These machines simulate Turing machines via bi-tag systems and are plotted as triangles in Figure 1. These machines improve the state of the art in small universal Turing machines and reduce the space between the universal and non-universal curves. Our 5-symbol machine uses the same number of instructions (22) as the current smallest known universal Turing machine (Rogozhin's 6-symbol machine [15]). Also, our 5-symbol machine has less instructions than Rogozhin's 5-symbol machines. Since 2-tag systems were first used by Minsky [7] to construct his famous universal Turing machine with 7-states and 4-symbols, a number of authors [1, 14, 15] have constructed 4-symbol machines. However our 4-symbol machine is the first reduction in the number of states.

Recently the simulation overhead of Turing machines by 2-tag systems was improved from exponential [2] to polynomial [17, 18]. More precisely, if Z is a single tape deterministic Turing machine that runs in time t, then the universal Turing machines of Minsky and Rogozhin et al. now simulate Z in $O(t^8(\log t)^4)$ time. It turns out that the time overhead can be improved to $O(t^4(\log t)^2)$ (this

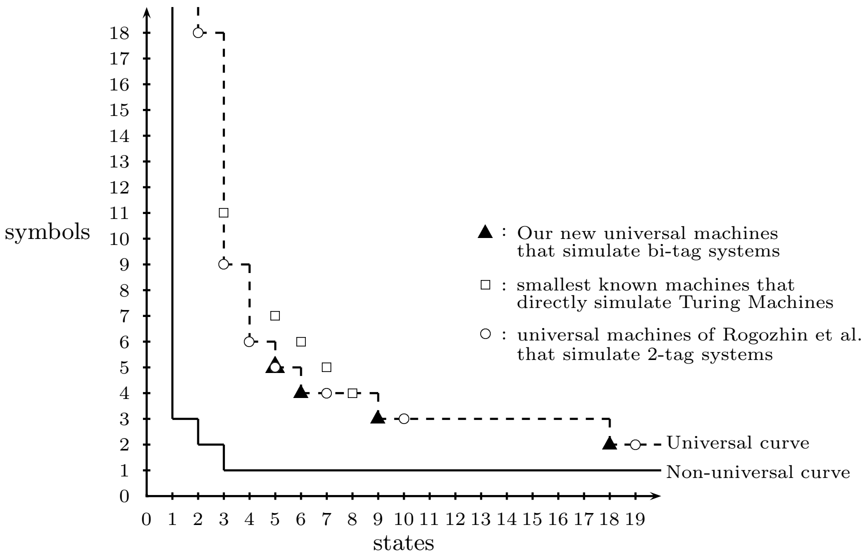

Fig. 1: Current state-symbol plot of small universal Turing machines.

result is as yet unpublished). In earlier work [11] we gave the smallest known universal Turing machines that directly simulate Turing machines. These machines run in time $O(t^2)$ and are plotted as squares in Figure 1. Assuming a single instruction is reserved for halting it is known that there are no universal Turing machine for the following state-symbol pairs: $(2,2)$ [4, 12], $(3,2)$ [13], $(2,3)$ (Pavlotskaya, unpublished), $(1,n)$ [3] and $(n,1)$ (trivial) for $n \geqslant 1$. These results induce the non-universal curve in Figure 1.

Our universal Turing machines simulate bi-tag systems with a quadratic polynomial increase in time. Hence from Theorem 1 our universal Turing machines simulate Turing machines efficiently in time $O(t^6(n))$. Information on alternative small universal Turing machines can be found in [6, 19, 20].

1.1 Preliminaries

The Turing machines considered in this paper are deterministic and have one tape. Our universal Turing machine in the class $\mathrm{UTM}(m,n)$ is denoted $U_{m,n}$. We write $c_1 \vdash c_2$ if a configuration c_2 is obtained from c_1 via a single computation step. We let $c_1 \vdash^m c_2$ denote a sequence of m computation steps and let $c_1 \vdash^* c_2$ denote 0 or more computation steps. Also, we let $\langle x \rangle$ denote the encoding of x and ϵ denote the empty word.

2 Bi-tag systems

The computation of a bi-tag system is similar to that of a tag system [8]. Bi-tag systems are essentially 1-tag systems (and so they read and delete one symbol per timestep) augmented with additional context sensitive rules that read, and delete, two symbols per timestep.

Definition 1 (Bi-tag system). *A bi-tag system is a tuple* (A, E, e_h, P)*. Here* A *and* E *are disjoint finite sets of symbols and* $e_h \in E$ *is the halt symbol.* P *is the finite set of productions. Each production is of one of the following 3 forms:*

$$P(a) = a, \quad P(e, a) \in AE, \quad P(e, a) \in AAE,$$

where $a \in A$*,* $e \in E$*, and* P *is defined on all elements of* $\{A \cup ((E - \{e_h\}) \times A)\}$ *and undefined on all elements of* $\{e_h\} \times A$*. Bi-tag systems are deterministic.*

A configuration of a bi-tag system is a word of the form $s = A^*(AE \cup EA)A^*$ called the dataword. In Definition 2 we let $a \in A$ and $e \in E$.

Definition 2 (BTS computation step). *A production is applied in one of two ways:*

(i) if $s = as'$ *then* $as' \vdash s'P(a)$*,*
(ii) if $s = eas'$ *then* $eas' \vdash s'P(e, a)$*.*

Theorem 1 ([10]). *Given a deterministic single tape Turing machine* Z *that runs in time* t *then there exists a bi-tag system that simulates the computation of* Z *using space* $O(t(n))$ *and time* $O(t^3(n))$*.*

In earlier work [10] Theorem 1 is obtained by proving bi-tag systems simulate Turing machines via clockwise Turing machines. A clockwise Turing machine is a Turing machine with a tape head that moves in one direction only, on a circular tape.

3 Universal Turing machines

In this section we give the input encoding to our universal Turing machines. Following this we give each of our universal Turing machines and describe their operation by explaining how they simulate bi-tag systems. Let R denote a bi-tag system that is to be simulated. The encoding of R as a word is denoted $\langle R \rangle$. The encodings of symbols $a \in A$ and $e \in E$ are denoted $\langle a \rangle$ and $\langle e \rangle$ respectively. The encodings of productions $P(a)$ and $P(e, a)$ are denoted as $\langle P(a) \rangle$ and $\langle P(e, a) \rangle$ respectively. Let $R = (A, E, e_h, P)$ be a bi-tag system where $A = \{a_1, \ldots, a_q\}$ and $E = \{e_1, \ldots, e_h\}$.

Definition 3. *The encoding of a configuration of* R *is of the form*

$$^{\omega}c\langle R \rangle S^*(\langle A \rangle M)^*\Big(\langle A \rangle M \langle E \rangle \cup \langle E \rangle \langle A \rangle M\Big)(\langle A \rangle M)^* D c^{\omega} \tag{1}$$

where $(\langle A \rangle M)^*\Big(\langle A \rangle M \langle E \rangle \cup \langle E \rangle \langle A \rangle M\Big)(\langle A \rangle M)^* D$ *encodes* R*'s dataword via Table 2,* $c^{\omega} = ccc\ldots$*,* $^{\omega}c = \ldots ccc$*, and* $\langle R \rangle$ *is given by Equation (2) and Tables 1 and 2.*

$$\begin{aligned} \langle R \rangle = &H \langle P(e_{h-1}, a_q) \rangle V \langle P(e_{h-1}, a_{q-1}) \rangle \ldots V \langle P(e_{h-1}, a_1) \rangle \\ &\vdots \\ &V \langle P(e_1, a_q) \rangle V \langle P(e_1, a_{q-1}) \rangle \ldots V \langle P(e_1, a_1) \rangle \\ &V^2 \langle P(a_q) \rangle V^2 \langle P(a_{q-1}) \rangle \ldots V^2 \langle P(a_1) \rangle V^3 \end{aligned} \tag{2}$$

In Equation (1) *the position of the tape head is over the symbol immediately to the right of* $\langle R\rangle S^*$. *The initial state is* u_1 *and the blank symbol is* c.

UTM	$\langle P(a_i)\rangle$	$\langle P(e_j,a_i)\rangle$ $P(e_j,a_i)=a_k e_m$	$\langle P(e_j,a_i)\rangle$ $P(e_j,a_i)=a_v a_k e_m$
$U_{5,5}$	$\delta\delta d^{16i-6}$	$\delta\delta L d^{16mq}\delta d^{16k-6}$	$\delta L d^{16mq}\delta d^{16k-2}\delta d^{16v-6}$
$U_{6,4}$	$\delta^5 g^{12i-10}\delta$	$\delta^4 L g^{12mq}\delta\delta g^{12k-10}\delta$	$\delta^2 L g^{12mq}\delta\delta g^{12hq+12k-4}\delta\delta g^{12v-10}\delta$
$U_{9,3}$	$\delta\delta cc\delta c^{8i}$	$\delta cc\delta\delta c^{8mq+2}\delta c^{8k}$	$\delta\delta c^{8mq+2}\delta c^{8k}\delta c^{8v}$
$U_{18,2}$	$cb(cc)^2 cb(cc)^{4i-2}$	$(cb)^2(cc)^{4qj+2}cb(cc)^{4k-2}$	$cb(cc)^{4qj+2}cb(cc)^{4k}cb(cc)^{4v-2}$

Table 1: Encoding of P productions. Here $a_i,a_k,a_v\in A$ and $e_j,e_m\in E$. If $e_m\neq e_h$ then $L=\epsilon$. If $e_m=e_h$ then $L=g^{12q+8}$ for $U_{6,4}$ and $L=d^{10}$ for $U_{5,5}$.

UTM	$\langle a_i\rangle$	$\langle e_j\rangle$	$\langle e_h\rangle$	S	M	D	V	H
$U_{5,5}$	b^{4i-1}	b^{4jq}	$b^{4hq+2}\delta$	d^2	δ	ϵ	δ	cd
$U_{6,4}$	b^{8i-5}	b^{8jq}	$b^{8q(h+1)+5}\delta$	g^2	δ	b	δ	H
$U_{9,3}$	b^{4i-1}	b^{4jq}	b^{4hq}	c^2	δ	ϵ	δcc	$bccbc$
$U_{18,2}$	$(bc)^{4i-1}$	$(bc)^{4jq}$	$(bc)^{4hq}$	c^4	b^2	$(bc)^2$	cb	cb

Table 2: Symbol values for Equations (1) and (2). If $e_m\neq e_h$ then $M'=\epsilon$ for $U_{5,5}$ and $U_{6,4}$. The value of H for $U_{6,4}$ is given in Equation (3) in Section 3.4.

3.1 Universal Turing machine algorithm overview

Each of our universal Turing machines use the same basic algorithm. Here we give a brief description of the simulation algorithm by explaining how our machines locate and simulate a production. The encoded production to be simulated is located using a unary indexing method. The encoded production $\langle P(a_i)\rangle$ or $\langle P(e_j,a_i)\rangle$ in Equation (2) is indexed (pointed to) by the number of symbols contained in the leftmost encoded symbol or pair of symbols in the encoded dataword (Equation (1)). For illustration purposes we will assume that we are using $U_{9,3}$. If the leftmost encoded symbol is $\langle a_i\rangle=b^{4i-1}$ (Table 2) then the value $4i-1$ is used to index $\langle P(a_i)\rangle$. If the leftmost encoded symbol is $\langle e_j\rangle=b^{4jq}$, and $\langle a_i\rangle=b^{4i-1}$ is adjacent, then the value $4jq{+}4i{-}1$ is used to index $\langle P(e_j,a_i)\rangle$. The number of b symbols in the encoded symbol, or pair of encoded symbols, is equal to the number of δc^* words between the leftmost encoded symbol and the encoded production to be simulated. To locate this production, $U_{9,3}$ simply changes each δc^* to δb^*, for each b in the leftmost encoded symbol or pair of encoded symbols. This process continues until the δ that separates two encoded symbols in the dataword is read. Note from Equation (1) that there is no δ marker between each $\langle e_j\rangle$ and the $\langle a_i\rangle$ to its right, thus allowing $\langle e_j\rangle\langle a_i\rangle$ to be read together during indexing. After indexing, our machines print the indexed production immediately to the right of the encoded dataword. After the indexed production has been read, then $\langle R\rangle$, the encoding of R, is restored to its original value. This completes the simulation of the production.

3.2 $U_{9,3}$

$U_{9,3}$	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9
c	bRu_1	cLu_3	cLu_3	bLu_9	cRu_6	bLu_4	δLu_4	cRu_7	bLu_5
b	cLu_2	cLu_2	bLu_4	bLu_4		bRu_6	bRu_7	cRu_9	cRu_8
δ	δRu_3	δLu_2	δRu_1	δLu_4	δLu_8	δRu_6	δRu_7	δRu_8	cRu_1

Table 3: Table of behaviour for $U_{9,3}$.

Example 1 ($U_{9,3}$ simulating the execution of the production $P(a_1)$). This example is presented using three Cycles. The tape head of $U_{9,3}$ is given by an underline. The current state of $U_{9,3}$ is given to the left in bold. The dataword $a_1 e_j a_i$ is encoded via Equation (1) and Table 2 as $bbb\delta b^{4jq}b^{4i-1}\delta$ and $P(a_1)$ is encoded via Table 1 as $\langle P(a_1)\rangle = \delta\delta cc\delta c^8$. From Equation (1) we get the initial configuration:

$$\boldsymbol{u_1},\ \ldots\langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^8\delta cc\delta cc\delta cc\underline{b}bb\delta b^{4jq}b^{4i-1}\delta ccc\ldots$$

Cycle 1 (index next production). In Cycle 1 (Table 4), $U_{9,3}$ reads the leftmost encoded symbol and locates the next encoded production to execute. $U_{9,3}$ scans right until it reads b in state u_1. Then $U_{9,3}$ scans left in states u_2 and u_3 until it reads the subword δc^*. This subword is changed to δb^* as $U_{9,3}$ scans right in states u_1 and u_3. The process is repeated until $U_{9,3}$ reads b in state u_3. This indicates that we have finished reading the leftmost encoded symbol, or pair of encoded symbols, and that the encoded production to be executed has been indexed. This signals the end of Cycle 1 and the beginning of Cycle 2.

$U_{9,3}$	u_1	u_2	u_3
c	bRu_1	cLu_3	cLu_3
b	cLu_2	cLu_2	bLu_4
δ	δRu_3	δLu_2	δRu_1

Table 4: Cycle 1 of $U_{9,3}$.

$U_{9,3}$	u_4	u_5	u_6	u_7	u_8	u_9
c	bLu_9	cRu_6	bLu_4	δLu_4	cRu_7	bLu_5
b	bLu_4		bRu_6	bRu_7		
δ	δLu_4	δLu_8	δRu_6	δRu_7	δRu_8	

Table 5: Cycle 2 of $U_{9,3}$.

$$\vdash \quad \boldsymbol{u_2},\ \ldots\langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^8\delta cc\delta cc\delta c\underline{c}cbb\delta b^{4jq}b^{4i-1}\delta ccc\ldots$$
$$\vdash^2 \quad \boldsymbol{u_3},\ \ldots\langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^8\delta cc\delta cc\underline{\delta}cccbb\delta b^{4jq}b^{4i-1}\delta ccc\ldots$$
$$\vdash^4 \quad \boldsymbol{u_1},\ \ldots\langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^8\delta cc\delta cc\delta bbb\underline{b}b\delta b^{4jq}b^{4i-1}\delta ccc\ldots$$
$$\vdash^{44} \quad \boldsymbol{u_1},\ \ldots\langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^8\delta bb\delta bb\delta bbbbb\underline{\delta}b^{4jq}b^{4i-1}\delta ccc\ldots$$
$$\vdash^2 \quad \boldsymbol{u_4},\ \ldots\langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^8\delta bb\delta bb\delta bbbbb\underline{\delta}b^{4jq}b^{4i-1}\delta ccc\ldots$$

In the configuration immediately above the encoded production $\langle P(a_1)\rangle$ has been indexed and we have entered Cycle 2.

Cycle 2 (Print production). Cycle 2 (Table 5) prints the encoded production, that was indexed in Cycle 1, immediately to the right of the encoded dataword. $U_{9,3}$ scans left in state u_4 and records the next symbol of the encoded production to be printed. If $U_{9,3}$ reads the subword ccc it enters state u_6, scans right, and prints b at the right end of the encoded dataword. A single b is printed for each cc pair that does not have δ immediately to its left. If $U_{9,3}$ reads the subword $c\delta cc$ it scans right in state u_7 and prints δ at the right end of the encoded dataword. This process is repeated until the end of the encoded production is detected by reading the subword $\delta\delta cc$ which causes $U_{9,3}$ to enter Cycle 3.

$$\begin{array}{ll}
\vdash^{13} & \boldsymbol{u_4}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^6 c\underline{c}(\delta bb)^3 bbb\delta b^{4jq}b^{4i-1}\delta ccc\ldots \\
\vdash^{3} & \boldsymbol{u_6}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^6 \underline{b}b(\delta bb)^3 bbb\delta b^{4jq}b^{4i-1}\delta ccc\ldots \\
\vdash^{4(jq+i)+14} & \boldsymbol{u_6}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^6 bb(\delta bb)^3 bbb\delta b^{4jq}b^{4i-1}\delta \underline{c}cc\ldots \\
\vdash & \boldsymbol{u_4}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta cc\delta c^6 bb(\delta bb)^3 bbb\delta b^{4jq}b^{4i-1}\underline{\delta} bcc\ldots
\end{array}$$

In the configuration immediately above the first symbol of the encoded production $\langle P(a_1)\rangle$ has been printed. Following the printing of the last symbol of the encoded production we get:

$$\begin{array}{ll}
\vdash^{*} & \boldsymbol{u_4}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta c\underline{c}\delta b^8(\delta bb)^3 bbb\delta b^{4jq}b^{4i-1}\delta b^3\delta cc\ldots \\
\vdash^{3} & \boldsymbol{u_8}, \ldots \langle P(a_2)\rangle(\delta cc)^2\underline{\delta}\delta bb\delta b^8(\delta bb)^3 bbb\delta b^{4jq}b^{4i-1}\delta b^3\delta cc\ldots
\end{array}$$

In the configuration immediately above we have finished printing the encoded production $\langle P(a_1)\rangle$ to the right of the dataword and we have entered Cycle 3.

Cycle 3 (restore tape). Cycle 3 (Table 6) restores $\langle R\rangle$ to its original value. The tape head of $U_{9,3}$ scans right switching between states u_8 and u_9 changing b symbols to c symbols. This continues until $U_{9,3}$ reads the δ marking the leftmost end of the dataword in u_9. Note from Table 1 and Equation (2) that there is an even number of b symbols between each pair of δ symbols in $\langle R\rangle$ hence each δ symbol in $\langle R\rangle$ will be read in state u_8. Each a_i symbol in the dataword is encoded by an odd number of b symbols ($\langle a_i\rangle = b^{4i-1}$) and hence the first δ symbol in the dataword will be read in state u_9. This δ symbol marks the left end of the new dataword and causes $U_{9,3}$ to enter state u_1 thus completing Cycle 3 and the production simulation.

$U_{9,3}$	u_8	u_9
b	cRu_9	cRu_8
δ	δRu_8	cRu_1

Table 6: Cycle 2 of $U_{9,3}$.

$$\begin{array}{ll}
\vdash^{25} & \boldsymbol{u_9}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta c\delta c^8(\delta cc)^3 ccc\underline{\delta} b^{4jq}b^{4i-1}\delta b^3\delta cc\ldots \\
\vdash & \boldsymbol{u_1}, \ldots \langle P(a_2)\rangle(\delta cc)^2\delta\delta c\delta c^8(\delta cc)^3 cccc\underline{b} b^{4jq-1}b^{4i-1}\delta b^3\delta cc\ldots
\end{array}$$

In the configuration immediately above our example simulation of production $P(a_1)$ is complete.

Theorem 2. *Given a bi-tag system $\langle R\rangle$ the computation of $\langle R\rangle$ is simulated by $U_{9,3}$.*

Proof. In order to prove the correctness of $U_{9,3}$ we must prove that $U_{9,3}$ simulates any possible $P(a)$ or $P(e,a)$ production of an arbitrary bi-tag system and, that $U_{9,3}$ also simulates halting when the encoded halt symbol $\langle e_h \rangle$ is encountered. In Example 1 $U_{9,3}$ simulates $P(a_1)$ for an arbitrary bi-tag system where a_1 is the leftmost symbol in a fixed dataword. This example easily generalises to any production $P(a_i)$ where a_i is the leftmost symbol in an arbitrary dataword. When some $e \in E$ is the leftmost symbol in the dataword then some production $P(e,a)$ must be executed. The simulation of $P(a_1)$ in Example 1 is also used to verify the simulation of $P(e,a)$. Note from Equation (1) that there is no δ marker between each $\langle e_j \rangle$ and the adjacent $\langle a_i \rangle$ to its right, thus allowing $\langle e_j \rangle$ and $\langle a_i \rangle$ to be read together during Cycle 1. Using the encoding in Defintion 3, the number of b symbols in $\langle e_j \rangle\langle a_i \rangle$ indexes $\langle P(e,a) \rangle$. Thus, the indexing of $\langle P(e,a) \rangle$ is carried out in the same manner as the indexing of $\langle P(a) \rangle$. The printing of production $\langle P(e,a) \rangle$ during Cycle 2 and the subsequent restoring of $\langle R \rangle$ during Cycle 3 proceed in the same manner as with $P(a_1)$.

If the encoded halt symbol $\langle e_h \rangle = b^{4hq}$ is the leftmost symbol in the encoded dataword, and $\langle a_i \rangle$ is adjacent, this is encoded via Definition 3 as follows:

$$\boldsymbol{u_1},\ bccbc\langle P(e_{h-1},a_q)\rangle\delta cc\ldots\langle P(a_1)\rangle(\delta cc)^3(cc)^*\underline{b}b^{4hq-1}b^{4i-1}\delta(\langle A\rangle\delta)^*c\ldots$$

During Cycle 1, immediately after reading the $(4hq+3)^{th}$ b symbol in the dataword, $U_{9,3}$ scans left in u_2 and we get the following:

$$\vdash^* \ \boldsymbol{u_2},\ bccb\underline{c}\langle P(e_{h-1},a_q)\rangle\delta cc\ldots\langle P(a_1)\rangle(\delta cc)^3(cc)^*c^{4hq+3}b^{4i-4}\delta(\langle A\rangle\delta)^*c\ldots$$
$$\vdash^4 \ \boldsymbol{u_5},\ \underline{b}bbbc\langle P(e_{h-1},a_q)\rangle\delta cc\ldots\langle P(a_1)\rangle(\delta cc)^3(cc)^*c^{4hq+3}b^{4i-4}\delta(\langle A\rangle\delta)^*c\ldots$$

There is no transition rule in Table 3 for the case when in u_5 read b hence the computation halts. □

The proof of correctness given for $U_{9,3}$ can be applied to the remaining machines in a straightforward way, so we do not restate it.

3.3 $U_{5,5}$

$U_{5,5}$	u_1	u_2	u_3	u_4	u_5
g	bLu_1	gRu_1	bLu_3		
b	gLu_1	gRu_2	dRu_5	gRu_4	dRu_3
δ	cRu_2	cRu_2	δRu_3	cRu_4	dRu_1
c	δLu_1	bLu_3	δLu_3	δLu_3	
d	bLu_1	gRu_2	bLu_5	bLu_2	bLu_4

Table 7: Table of behaviour for $U_{5,5}$.

The dataword $a_1e_ja_i$ is encoded via Equation (1) and Table 2 as $bbb\delta b^{4jq}b^{4i-1}\delta$ and $P(a_1)$ is encoded via Table 1 as $\langle P(a_1)\rangle = \delta\delta d^{10}$. From Equation (1) we get the initial configuration:

$$\boldsymbol{u_1},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^{10}\delta\delta\delta\underline{b}bb\delta b^{4jq}b^{4i-1}\delta ccc\ldots$$

Cycle 1 (index next production). In Cycle 1 (Table 8) when $U_{5,5}$ reads b in state u_1, it changes it to g and scans left until it reads δ. This δ is changed to c and $U_{5,5}$ then enters state u_2 and scans right until it reads g which causes it to re-enter state u_1. This process is repeated until $U_{5,5}$ reads the δ that separates a pair of encoded symbols in the encoded dataword. This signals the end of Cycle 1 and the beginning of Cycle 2.

$U_{5,5}$	u_1	u_2
g	bLu_1	gRu_1
b	gLu_1	gRu_2
δ	cRu_2	cRu_2
c	δLu_1	
d	bLu_1	

Table 8: Cycle 1 of $U_{5,5}$.

$U_{5,5}$	u_2	u_3	u_4	u_5
g		bLu_3		
b	gRu_2		gRu_4	
δ	cRu_2	δRu_3	cRu_4	
c	bLu_3	δLu_3	δLu_3	
d	gRu_2	bLu_5	bLu_2	bLu_4

Table 9: Cycle 2 of $U_{5,5}$.

$U_{5,5}$	u_3	u_5
b	dRu_5	dRu_3
δ	δRu_3	dRu_1

Table 10: Cycle 3 of $U_{5,5}$.

$$\begin{array}{ll}
\vdash^3 & \boldsymbol{u_1},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^{10}\delta\delta cg\underline{b}b\delta b^{4jq}b^{4i-1}\delta ccc\ldots \\
\vdash^{18} & \boldsymbol{u_1},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^{10}cccggg\underline{\delta}b^{4jq}b^{4i-1}\delta ccc\ldots \\
\vdash & \boldsymbol{u_2},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^{10}cccgggc\underline{b}b^{4jq-1}b^{4i-1}\delta ccc\ldots
\end{array}$$

Cycle 2 (Print production). Cycle 2 (Table 9) begins with $U_{5,5}$ scanning right and printing b to the right of the encoded dataword. Following this $U_{5,5}$ scans left in state u_3 and records the next symbol of the encoded production to be printed. If $U_{5,5}$ reads the subword $dddd$ it enters state u_2, scans right, and prints b at the right end of the encoded dataword. If $U_{5,5}$ reads the subword δdd it scans right in state u_4 and prints δ at the right end of the encoded dataword. This process is repeated until the end of the encoded production is detected by reading δ in state u_3 causing $U_{5,5}$ to enter Cycle 3.

$$\begin{array}{ll}
\vdash^* & \boldsymbol{u_3},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^6ddd\underline{d}\delta\delta\delta bbb\delta b^{4jq}b^{4i-1}\delta bcc\ldots \\
\vdash^3 & \boldsymbol{u_2},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^6\underline{d}bbb\delta\delta\delta bbb\delta b^{4jq}b^{4i-1}\delta bcc\ldots \\
\vdash^* & \boldsymbol{u_3},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d\underline{d}b^8\delta\delta\delta bbb\delta b^{4jq}b^{4i-1}\delta bbbcc\ldots \\
\vdash^2 & \boldsymbol{u_4},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\underline{\delta}bbb^8\delta\delta\delta bbb\delta b^{4jq}b^{4i-1}\delta bbbcc\ldots \\
\vdash^* & \boldsymbol{u_3},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\underline{\delta}\delta bbb^8\delta\delta\delta bbb\delta b^{4jq}b^{4i-1}\delta bbb\delta cc\ldots
\end{array}$$

Cycle 3 (restore tape). In Cycle 3 (Table 10) the tape head of $U_{5,5}$ scans right switching between states u_3 and u_5 changing b symbols to d symbols. This continues until $U_{5,5}$ reads the δ marking the leftmost end of the encoded dataword in u_5. Note from Table 1 and Equation (2) that there is an even number of d symbols between each pair of δ symbols in $\langle R\rangle$ hence each δ symbol in $\langle R\rangle$ will be read in state u_3. Each a_i symbol in the dataword is encoded by an odd number of symbols ($\langle a_i\rangle = b^{4i-1}$) and hence the first δ symbol in the dataword will be read in in state u_5. This causes $U_{5,5}$ to enter state u_1 thus completing Cycle 3 and the production simulation.

$$\vdash^{19} \quad \boldsymbol{u_1},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\delta\delta d^{10}\delta\delta\delta dddd\underline{b}b^{4jq-1}b^{4i-1}\delta bbb\delta cc\ldots$$

Halting for $\mathbf{U_{5,5}}$. If the encoded halt symbol $\langle e_h \rangle = b^{4hq+2}\delta$ is the leftmost symbol in the encoded dataword then this is encoded via Definition 3 as follows:

$$\boldsymbol{u_1},\ cd\langle P(e_{h-1}, a_q)\rangle\delta \ldots \delta^2\langle P(a_1)\rangle\delta^3(dd)^*\underline{b}b^{4hq+1}\delta(\langle A\rangle\delta)^*c\ldots$$

The computation continues as before until $U_{5,5}$ enters Cycle 2 and scans left in u_3. Immediately after $U_{5,5}$ reads the leftmost d during this leftward scan we get:

$$\vdash \qquad \boldsymbol{u_5},\ \underline{c}b\langle P'(e_{h-1}, a_q)\rangle\delta \ldots \delta^2\langle P'(a_1)\rangle\delta^3(dd)^*b^{4hq+2}\delta(\langle A\rangle\delta)^*bc\ldots$$

In the configuration above, $\langle P'\rangle$ denotes the word in which all the d symbols in $\langle P\rangle$ are changed to b symbols. There is no transition rule in Table 7 for the case 'when in u_5 read c' hence the computation halts.

3.4 $\boldsymbol{U_{6,4}}$

$U_{6,4}$	u_1	u_2	u_3	u_4	u_5	u_6
g	bLu_1	gRu_1	bLu_3	bRu_2	bLu_6	bLu_4
b	gLu_1	gRu_2	bLu_5	gRu_4	gRu_6	gRu_5
δ	cRu_2	cRu_2	δLu_5	cRu_4	δRu_5	gRu_1
c	δLu_1	gRu_5	δLu_3	cRu_5	bLu_3	

Table 11: Table of behaviour for $U_{6,4}$.

The dataword $a_1e_ja_i$ is encoded via Equation (1) and Table (2) as $bbb\delta b^{8jq}b^{8i-5}\delta b$. From Equation (1) we get the initial configuration:

$$\boldsymbol{u_1},\ \ldots\delta^2\langle P(a_2)\rangle\delta^2\langle P(a_1)\rangle\delta\delta\delta\underline{b}bb\delta b^{8jq}b^{8i-5}\delta bcc\ldots$$

Cycle 1 (index next production). In Cycle 1 (Table 12) when $U_{6,4}$ reads b in state u_1 it scans left until it reads a δ. This δ is changed to c and $U_{6,4}$ then enters state u_2 and scans right until it reads g which causes it to re-enter state u_1. This process is repeated until $U_{6,4}$ reads the δ that separates a pair of encoded symbols in the encoded dataword. This signals the end of Cycle 1 and the beginning of Cycle 2.

$U_{6,4}$	u_1	u_2
g	bLu_1	gRu_1
b	gLu_1	gRu_2
δ	cRu_2	cRu_2
c	δLu_1	

Table 12: Cycle 1 of $U_{6,4}$.

$U_{6,4}$	u_2	u_3	u_4	u_5	u_6
g		bLu_3	bRu_2	bLu_6	bLu_4
b	gRu_2	bLu_5	gRu_4		
δ	cRu_2	δLu_5	cRu_4	δRu_5	
c	gRu_5	δLu_3	cRu_5	bLu_3	

Table 13: Cycle 2 of $U_{6,4}$.

$U_{6,4}$	u_5	u_6
b	gRu_6	gRu_5
δ	δRu_5	gRu_1

Table 14: Cycle 3 of $U_{6,4}$.

Cycle 2 (Print production). Cycle 2 (Table 13) begins with $U_{6,4}$ scanning right and printing bb to the right of the encoded dataword. Following this $U_{6,4}$ scans left in state u_3 and records the next symbol of the encoded production to be printed. If $U_{6,4}$ reads the subword $ggg\delta$ or $gggb$ it enters state u_2 scans right and prints bb at the right end of the encoded dataword. If $U_{6,4}$ reads the subword δggb it scans right in state u_4 and prints δb at the right end of the encoded dataword. This process is repeated until the end of the encoded production is detected by reading δ in state u_5 causing $U_{6,4}$ to enter Cycle 3.

Cycle 3 (restore tape). In Cycle 3 (Table 14) the tape head of $U_{6,4}$ scans right switching between states u_5 and u_6 changing b symbols to g symbols. This continues until $U_{6,4}$ reads the δ marking the leftmost end of the encoded dataword in u_6. Note from Table 1 and Equation (2) that there is an even number of g symbols between each pair of δ symbols in $\langle R\rangle$ hence each δ symbol in $\langle R\rangle$ will be read in state u_5. Each a_i symbol in the dataword is encoded by an odd number of symbols ($\langle a_i\rangle = b^{8i-5}$) and hence the first δ symbol in the dataword will be read in state u_6. This causes $U_{6,4}$ to enter state u_1 thus completing Cycle 3 and the production simulation.

Special case for $U_{6,4}$. If we are simulating a production of the form $P(e,a) = a_v a_k e_m$ we have a special case. Note from Table 2 and Cycle 2 that the simulation of $P(e,a) = a_v a_k e_m$ for $U_{6,4}$ will result in the word $b^{8v-5}\delta b^{8hq+8k-3}\delta b^{8mq} b$ being printed to to the right of the dataword. It is clear from Table 2 that a_k is not encoded in this word. However when $U_{6,4}$ reads the subword $b^{8hq+8k-3}\delta$ it will index $\langle P(a_k)\rangle$ in H which results in $\langle a_k\rangle$ being printed to the dataword. To see this, note that the value of H from Equation (2) for $U_{6,4}$ is as follows:

$$H = cgb\delta\delta\langle P(a_q)\rangle V^2\langle P(a_{q-1})\rangle \ldots V^2\langle P(a_1)\rangle V^3 \tag{3}$$

The halting condition for $U_{6,4}$ occurs in a similar manner to that of $U_{5,5}$. It occurs during the first scan left in Cycle 2 when $U_{6,4}$ reads c in state u_6 at the left end of $\langle R\rangle$.

3.5 $U_{18,2}$

$U_{18,2}$	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9
c	bRu_2	cRu_1	cLu_5	cLu_5	cLu_4	bRu_2	cLu_8	bRu_{12}	bLu_{10}
b	bRu_3	bRu_1	bLu_9	bLu_6	cLu_4	cLu_4	bLu_9	bLu_7	bLu_7
$U_{18,2}$	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}	u_{16}	u_{17}	u_{18}
c	cRu_{13}	bLu_7	cRu_{11}	cLu_{15}	cRu_{13}	bLu_9	cRu_{17}		cRu_{15}
b	bRu_{15}	bRu_{12}	bRu_{11}	bRu_{14}	bRu_{13}	cRu_{16}	bRu_{15}	cRu_{18}	cRu_1

Table 15: Table of behaviour for $U_{18,2}$.

The dataword $a_1 e_j a_i$ is encoded via Equation (1) and Table (2) as $bcbcbcbb(bc)^{4jq}(bc)^{4i-1}bb(bc)^2$. From Equation (1) we get the initial configuration:

$$\boldsymbol{u_1}, \ldots\langle P(a_2)\rangle(cb)^2\langle P(a_1)\rangle cbcbcb\underline{b}cbcbcbb(bc)^{4jq}(bc)^{4i-1}bb(bc)^2cc\ldots$$

Cycle 1 (index next production). In Cycle 1 (Table 16) $U_{18,2}$ scans right in states u_1, u_2 and u_3 until it reads the subword bc. Following this, it scans left in states u_4, u_5 and u_6 until it reads the subword cb. This cb is changed to bb

and $U_{18,2}$ re-enters state u_1 and scans right. This process is repeated until $U_{18,2}$ reads the *bb* that separates a pair of encoded symbols in the encoded dataword during a scan right. This signals the end of Cycle 1 and the beginning of Cycle 2.

$U_{18,2}$	u_1	u_2	u_3	u_4	u_5	u_6
c	bRu_2	cRu_1	cLu_5	cLu_5	cLu_4	bRu_2
b	bRu_3	bRu_1	bLu_9	bLu_6	cLu_4	cLu_4

Table 16: Cycle 1 of $U_{18,2}$.

Cycle 2 (Print production). In Cycle 2 (Table 17) $U_{18,2}$ scans left in states u_7, u_8 and u_9 and records the next symbol of the encoded production to be printed. If $U_{18,2}$ reads the subword *cc* then it scans right in states $u_{11}$and u_{12} and changes the *cc* immediately to the right of the encoded dataword to *bc*. If $U_{18,2}$ reads the subword *ccb* it scans right in states u_{13} and u_{14} and changes the rightmost *bc* in the encoded dataword to *bb*. This process is repeated until the end of the encoded production is detected by reading the subword *bcb* during the scan left. This causes $U_{18,2}$ to enter Cycle 3.

$U_{18,2}$	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}	u_{13}	u_{14}	u_{15}
c	cLu_8	bRu_{12}	bLu_{10}	cRu_{13}	bLu_7	cRu_{11}	cLu_{15}	cRu_{13}	bLu_9
b	bLu_9	bLu_7	bLu_7	bRu_{15}	bRu_{12}	bRu_{11}	bRu_{14}	bRu_{13}	

Table 17: Cycle 2 of $U_{18,2}$.

Cycle 3 (restore tape). In Cycle 3 (Table 18) the tape head of $U_{18,2}$ scans right in states u_{15}, u_{16}, u_{17} and u_{18} changing each *bc* to *cc* and each *bb* to *cb*. This continues until $U_{18,2}$ reads the *bb* marking the leftmost end of the dataword in u_{17} and u_{18}. Note from Table 1 and Equation (2) that the number of *cc* subwords between each pair of δ symbols in $\langle R\rangle$ is even hence each *bb* pair will be read in states u_{15} and u_{16} and restored to *cb*. Each a_i symbol in the dataword is encoded by an odd number of *bc* subwords ($\langle a_i\rangle = (bc)^{4i-1}$) and hence the first *bb* pair in the dataword will be read in in state u_{17} and u_{18} causing $U_{18,2}$ to enter state u_1 thus completing Cycle 3 and the production simulation.

$U_{18,2}$	u_{15}	u_{16}	u_{17}	u_{18}
c		cRu_{17}		cRu_{15}
b	cRu_{16}	bRu_{15}	cRu_{18}	cRu_1

Table 18: Cycle 3 of $U_{18,2}$.

There is no halting condition for $U_{18,2}$ and as such $U_{18,2}$ simulates bi-tag systems that have no halting symbol e_h. Such bi-tag systems simulate halting by entering a simple repeating sequence of configurations.

References

1. C. Baiocchi. Three small universal Turing machines. In M. Margenstern and Y. Rogozhin, editors, *Machines, Computations, and Universality*, volume 2055 of *LNCS*, pages 1–10, Chişinău, Moldova, May 2001. MCU, Springer.

2. J. Cocke and M. Minsky. Universality of tag systems with $P = 2$. *Journal of the ACM*, 11(1):15–20, Jan. 1964.
3. G. Hermann. The uniform halting problem for generalized one state Turing machines. In *Proceedings, Ninth Annual Symposium on Switching and Automata Theory*, pages 368–372, New York, Oct. 1968. IEEE.
4. M. Kudlek. Small deterministic Turing machines. *TCS*, 168(2):241–255, Nov. 1996.
5. M. Kudlek and Y. Rogozhin. A universal Turing machine with 3 states and 9 symbols. In W. Kuich, G. Rozenberg, and A. Salomaa, editors, *Developments in Language Theory*, volume 2295 of *LNCS*, pages 311–318. Springer, May 2002.
6. M. Margenstern and L. Pavlotskaya. On the optimal number of instructions for universality of Turing machines connected with a finite automaton. *International Journal of Algebra and Computation*, 13(2):133–202, Apr. 2003.
7. M. Minsky. Size and structure of universal Turing machines using tag systems. In *Recursive Function Theory, Symposium in Pure Mathematics*, volume 5, pages 229–238, Provelence, 1962. AMS.
8. M. Minsky. *Computation, finite and infinite machines*. Prentice-Hall, 1967.
9. T. Neary. Small polynomial time universal Turing machines. In T. Hurley, A. Seda, et al., editors, 4^{th} *Irish Conference on the Mathematical Foundations of Computer Science and Information Technology*, pages 325–329, Cork, Ireland, Aug. 2006.
10. T. Neary and D. Woods. A small fast universal Turing machine. Technical Report NUIM-CS-TR-2005-12, National university of Ireland, Maynooth, 2005.
11. T. Neary and D. Woods. Small fast universal Turing machines. *TCS*, 362(1–3):171–195, Oct. 2006.
12. L. Pavlotskaya. Solvability of the halting problem for certain classes of Turing machines. *Mathematical Notes (Springer)*, 13(6):537–541, June 1973.
13. L. Pavlotskaya. Dostatochnye uslovija razreshimosti problemy ostanovki dlja mashin T'juring. *Problemi kibernetiki*, pages 91–118, 1978. (Sufficient conditions for the halting problem decidability of Turing machines) (in Russian).
14. R. Robinson. Minsky's small universal Turing machine. *International Journal of Mathematics*, 2(5):551–562, 1991.
15. Y. Rogozhin. Small universal Turing machines. *TCS*, 168(2):215–240, Nov. 1996.
16. C. E. Shannon. A universal Turing machine with two internal states. *Automata Studies, Annals of Mathematics Studies*, 34:157–165, 1956.
17. D. Woods and T. Neary. On the time complexity of 2-tag systems and small universal Turing machines. In *In* 47^{th} *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 132–143, Berkeley, California, Oct. 2006. IEEE.
18. D. Woods and T. Neary. Remarks on the computational complexity of small universal Turing machines. In T. Hurley, A. Seda, et al., editors, *Fourth Irish Conference on the Mathematical Foundations of Computer Science and Information Technology*, pages 334–338, Cork, Ireland, Aug. 2006. MFCSIT.
19. D. Woods and T. Neary. The complexity of small universal Turing machines. In S. B. Cooper, B. Lowe, and A. Sorbi, editors, *Computability in Europe 2007*, volume 4497 of *LNCS*, pages 791–798, Sienna,Italy, June 2007. CIE, Springer.
20. D. Woods and T. Neary. Small semi-weakly universal Turing machines. In J. Durand-Lose and M. Margenstern, editors, *Machines, Computations, and Universality*, LNCS, Orelans, France, Sept. 2007. MCU, Springer. This Volume.