Three Small Universal Turing Machines

Claudio Baiocchi

Dipartimento di Matematica, Università "La Sapienza" di Roma (Italy)
baiocchi@mat.uniroma1.it

Abstract. We are interested by "small" Universal Turing Machines (in short: UTMs), in the framework of $2, 3$ or 4 tape–symbols. In particular:

- 2 tape–symbols. Apart from the old 24–states machine constructed by Rogozhin in 1982, we know two recent examples requiring 22 states, one due to Rogozhin and one to the author.
- 3 tape–symbols. The best example we know, due to Rogozhin, requires 10 states. It uses a strategy quite hard to follow, in particular because even–length productions require a different treatment with respect to odd–length ones.
- 4 tape–symbols. The best known machines require 7 states. Among them, the Rogozhin's one require only 26 commands; the Robinson's one, though requiring 27 commands, fournishes an easier way to recover the output when the TM halts. In particular, Robinson asked for a 7×4 UTM with only 26 commands and an easy treatment of the output.

Here we will firstly construct a 7×4 UTM with an easy recover of the output which requires only 25 commands; then we will simulate such a machine by a (simple) 10×3 UTM and by a 19×2 UTM.

1 Tag Systems

Let us recall some notions needed later on. We are given an integer $d \geq 2$ (the *deletion number*); a "d–tag system" is nothing but a word–processor, acting on words W whose letters are the elements of a finite alphabet $\mathcal{A}$. For each letter $a \in \mathcal{A}$, a (possibly empty) word $p(a)$ is given, that is called the "production" of a; the letters whose production is empty are called *halting* letters. Words of length strictly less than d and words starting with a halting–letter are called halting–words. The action of the word–processor on the generic word W is specified by the following rules:

- if W is a halting–word, the word–processor stops;
- if not, W has the form $W = aUV$ where: a is a non–halting letter; U is a word of length $d-1$; V is another (possibly empty) word. The original word $W = aUV$ is then replaced with $W = Vp(a)$; and the work continues on the new W thus obtained.

For our purposes we can restrict ourselves to the case of deletion number $d = 2$; furthermore we can assume that all but one the halting letters are "virtual",

M. Margenstern and Y. Rogozhin (Eds.): MCU 2001, LNCS 2055, pp. 1–10, 2001.

because (due to the form of the initial word W) they will never appear as initial letters; and that the length of the word W to be processed will always remain strictly greater than 1 (in the following, when speaking of "tag systems", we will refer to such restricted case).

Following the general approach suggested by Minsky (see e.g. [Min]) we will now construct some TMs that simulate tag–systems; they will be *universal* because, in turn, TMs can be simulated by tag–systems[1].

2 Simulation of Tag–Systems

We will use the term "symbol" (or tape–symbol) to denote the elements of the TM's alphabet, and the term "string" to denote words constructed with tape–symbols; terms "letter" and "word" will instead refer to the alphabet $\mathcal{A}$ of the simulated tag–system. We will set $\mathcal{A} = \{a_0, a_1, a_2, \dots, a_n\}$, the unique actual halting letter being a_0; recall that the only case of halting corresponds to a word of length at least 2 whose first letter is a_0.

The initial tape of the simulating machine will be choosen in the form:

$$\mathcal{T} = \dots\dots \mathcal{P} \dots \mathcal{S} \dots\dots$$

where:

- dots denote zones of blanks;
- the string $\mathcal{S}$ encodes (in a form we will detail in a moment) the word W to be processed;
- the string $\mathcal{P}$ has the form:

$$\mathcal{P} = P_{n+1} P_n \cdots P_1 P_0 \tag{1}$$

  where P_j encodes, in a suitable form, the production $p(a_j)$ (if we do not specify the index of a letter $a \in \mathcal{A}$, we will denote by $P(a)$ the corresponding string in $\mathcal{P}$).

We will associate to any $a_j \in \mathcal{A}$ a positive integer N_j (again, if we do not specify the index of an $a \in \mathcal{A}$ we will write $N(a)$ instead of N_j). The definition of the function N will be given later on; for the moment let us say only that it must be injective. Once N has been choosen, we fix two tape–symbols μ, ν with $\mu \neq \nu$ and we set[2]:

$$\begin{cases} \text{if } W = b_1 b_2 \dots b_k \text{ is the initial word,} \\ \text{we choose } \mathcal{S} = \nu^{N(b_1)} \mu \nu^{N(b_2)} \mu \dots \mu \nu^{N(b_k)} \end{cases} \tag{2}$$

so that the exponent of ν will specify (the function N being injective) which letter we are dealing with.

[1] the details can be found e.g. in the self–contained and very clear paper [Rob]

[2] ν stay for "normal", μ for "marker"

In particular let W start with a non–halting letter a, whose production is $p(a) = a_{j_1} a_{j_2} \dots a_{j_p}$; in order to "elaborate" such a W we must append to the right end of $\mathcal{S}$ the string $\Sigma(a) = \mu\nu^{N_{j_1}} \mu\nu^{N_{j_2}} \mu \dots \nu^{N_{j_p}}$; and then kill the leftmost part of $\mathcal{S}$ (up to, and included, the second marker). In order to append $\Sigma(a)$ it could be useful to have, somewhere on the left of the scanned zone, a "mirror image" $\Sigma'(a)$ of $\Sigma(a)$; thus it will be sufficient to copy each symbol of $\Sigma'(a)$ at the end of $\mathcal{S}$[3].

The needed $\Sigma'(a)$ will in fact be the rightmost part of $P(a)$ (see formula (4) later on); the leftmost part of $P(a)$ being a "super–marker", signalling the end of $\Sigma'(a)$. Of course, before copying the needed string, we must "reach" it; in other words, by using the initial part $\nu^{N(a)}$ of $\mathcal{S}$, we must "neutralize" the part of $\mathcal{P}$ which is on the right of $P(a)$. This will be realized through a clever choice of the function N, which will be choosen in such a way that:

$$\left\{ \begin{array}{l} \text{if the initial letter of the word } W \text{ is } a_\ell, \\ \text{the corresponding initial string } \nu^{N_\ell} \text{ in } \mathcal{S} \text{ will force} \\ \text{the TM to "neutralize" } P_0, P_1, \dots P_{\ell-1}. \end{array} \right. \tag{3}$$

Such a neutralization will be the first part of the whole TM's job; the second part will copy the string; the third phase will recover the neutralized part of $\mathcal{P}$ and will annihilate the part of $\mathcal{S}$ corresponding to the two initial letters of the starting word W.

3 A 7×4 UTM with 25 Commands

We will choose as tape–symbols 0 (the blank) and α, β, γ. Concerning $\mathcal{S}$, we will choose α as the normal symbol and γ as the marker; concerning $\mathcal{P}$, the normal symbol will be 0 and the marker will be β; more precisely, a string like $\dots 00\beta$ will act as marker; a string like $\dots \beta 0 \beta$ will act as super–marker. We will fix the form of $\mathcal{P}$ by setting:

$$\left\{ \begin{array}{l} P_{n+1} = \beta;\ P(a) = 0\beta \text{ for any halting letter } a; \\ \text{for non halting letters, if } p(a) = b_1 b_2 \dots b_k, \text{ we choose:} \\ P(a) = 0\beta 0^{N(b_k)} 00 \beta 0^{N(b_{k-1})} 00 \beta \dots \beta 0^{N(b_2)} 00 \beta 0^{N(b_1)} 00 \beta. \end{array} \right. \tag{4}$$

Remark 1. In particular halting letters will all be encoded in $\mathcal{P}$ through the same string. This is not a problem, because virtual halting letters will never appear in first position in $\mathcal{S}$, thus their representations in $\mathcal{P}$ do not need to differ each one other (but their codes in $\mathcal{S}$ do need to differ...). On the other hand, if we sort $\mathcal{A}$ by putting all virtual halting letters at the end, we could suppress their representation in $\mathcal{P}$[4].

[3] $\Sigma'(a)$ is read from right to left, thus we ask for the mirror image. Of course we do not need a true copy: e.g. both markers and normal symbols used in $\mathcal{P}$ could differ from the ones (μ, ν) used in $\mathcal{S}$...

[4] but the leftmost part, $P_{n+1} = \beta$, is still needed

Remark 2. The neutralization of a P_j will simply consist in replacing each 0 by an α. Let us assume that the needed neutralization has been done; and let $P(a)$ be the rightmost non–neutralized part of $\mathcal{P}$. The string $P(a)$ ends with a substring like $\beta 0^x 00\beta$; its final part 00β will act as a first marker. Meeting such a marker, the TM will change it into $\alpha\alpha\beta$ and will append a marker γ on the right of $\mathcal{S}$. Coming back, the TM will encounter $\beta 0^x \alpha\alpha\beta$ and each 0 if any[5] will be changed into an α and that will force the TM to append an α. The work continues with the next (on the left) string like $\beta 0^y 00\beta$; and so on, until we reach (instead of a $\beta 0^z 00\beta$) the super–marker $\beta 0\beta$ (the leftmost β being the rightmost symbol of the closer P on the left, possibly P_{n+1}); this ends the second step of the work, and the restore phase will start.

Now we need an explicit formula for the function N; we will choose:

$$N_0 := 0; N_k := N_{k-1} + 1 + |p(a_k)| \qquad (k = 1, \ldots, n) \tag{5}$$

where $|X|$ denotes the length of the word X; remark that the "+1" ensures the injectivity of the function N.

Remark 3. In particular, see (2), the actual halting letter a_0 will be represented in $\mathcal{S}$ by an empty–string; its presence in the middle of W will correspond to a substring $\gamma\gamma$ in $\mathcal{S}$; the presence of a_0 in first (resp. last) position will correspond to the fact that the string $\mathcal{S}$ starts (resp. ends) with a γ. Halting–words in the tag–system thus correspond to $\mathcal{S}$ starting with a γ; in such a case the TM will simply halt.

With the choices (4) and (5) the reader will have no difficulties to check that property (3) holds if, in the first phase of the TM's work, each α of the initial part $\alpha^{N(a)}$ in $\mathcal{S}$ neutralizes up to and included the first substring 0β on its left; thus the initial TM's can be regulated by the following piece of table where, in first column, we indicate by arrows the direction the head is moving to:

Table 1. The first part of the TM's work.

Direction	State	0	α	β	γ	Description
$\rightarrow$	A	$\square$	$0\mathcal{L}B$	$\square$	Halt	The whole TM's work starts here
$\leftarrow$	B	$0\mathcal{L}B$	$0\mathcal{L}B$	$\beta\mathcal{L}C$	$\square$	Neutralization in $\mathcal{P}$
$\leftarrow$	C	$\alpha\mathcal{R}D$	$0\mathcal{L}B$	$\square$	$\square$	Doubt
$\rightarrow$	D	$\alpha\mathcal{R}D$	$0\mathcal{L}B$	$\beta\mathcal{R}D$	$\alpha\mathcal{L}E$	Neutralization in $\mathcal{S}$

[5] the letter a_0 corresponds to $x = 0$; see the following formula (5)

Remark 4. We want to point out that cells marked as $\boxed{\ }$ refer to cases we will never encounter if the initial tape corresponds to the simulation of a tag–system; in particular we could fill them with any command without affecting the work (and the universality) of our TM. On the contrary, the cell marked "Halt" *must* remain empty: the machine must halt if the string $\mathcal{S}$ starts with the marker γ (see Rem.3).

Concerning the second step of the TM's work almost any detail has been given in Rem.2; the corresponding piece of table is given by:

Table 2. The second part of the TM's work.

Direction	State	0	α	β	γ	Description
$\leftarrow$	E	$\alpha\mathcal{R}I$	$\alpha\mathcal{L}E$	$\beta\mathcal{L}F$	$\gamma\mathcal{L}E$	searches leftward
$\leftarrow$	F	$\alpha\mathcal{L}G$	$\alpha\mathcal{L}E$	$\boxed{\ }$	$\boxed{\ }$	doubt
$\leftarrow$	G	$\alpha\mathcal{R}H$	$\boxed{\ }$	$\beta\mathcal{R}J$	$\boxed{\ }$	decided
$\rightarrow$	H	$\gamma\mathcal{L}E$	$\alpha\mathcal{R}H$	$\beta\mathcal{R}H$	$\gamma\mathcal{R}H$	appends γ
$\rightarrow$	I	$\alpha\mathcal{L}E$	$\alpha\mathcal{R}I$	$\beta\mathcal{R}I$	$\gamma\mathcal{R}I$	appends α

and the third step requires just one line:

Table 3. The third part of the TM's work.

Direction	State	0	α	β	γ	Description
$\rightarrow$	J	$\boxed{\ }$	$0\mathcal{R}J$	$\beta\mathcal{R}J$	$\alpha\mathcal{R}A$	Restore

We promised a 7×4 UTM and we wrote a 10×4 table; however, according with Rem.4, nothing forbids to suppress state C, by replacing anywhere a switch to state C with a switch to state D; two other similar groupments give raise to a 7×4 UTM; see Table 4. Let us point out the major differences with respect to the 7×4 UTMs of Minsky [Min], Rogozhin [Ro2] and Robinson [Rob]. In these three TMs:

- the super–marker is given by a couple $\beta\beta$ of markers;
- almost each run of the machine swaps the markers ($\beta \longleftrightarrow \gamma$).

We choosed as the super–marker the string $\beta 0 \beta$; this choice, and the fact that we never swap the markers, allows to gain one empty cell in the table (column γ of row B; the swap of the markers would require to use such a cell for other purposes). On the other hand, our UTM halts exactly where the simulated tag–system would stop: when the machine halts, the output (to be interpreted in terms of tag–systems) starts from the γ scanned by the Head and ends on the first blank on the right.

Table 4. A 7×4 UTM with 25 commands and easy recover of the output.

Old names	New names	0	α	β	γ
A, B	T	$0\mathcal{L}T$	$0\mathcal{L}T$	$\beta\mathcal{L}U$	Halt
C, D	U	$\alpha\mathcal{R}U$	$0\mathcal{L}T$	$\beta\mathcal{R}U$	$\alpha\mathcal{L}V$
E	V	$\alpha\mathcal{R}Z$	$\alpha\mathcal{L}V$	$\beta\mathcal{L}W$	$\gamma\mathcal{L}V$
F	W	$\alpha\mathcal{L}X$	$\alpha\mathcal{L}V$	$\square$	$\square$
G, J	X	$\alpha\mathcal{R}Y$	$0\mathcal{R}X$	$\beta\mathcal{R}X$	$0\mathcal{R}T$
H	Y	$\gamma\mathcal{L}V$	$\alpha\mathcal{R}Y$	$\beta\mathcal{R}Y$	$\gamma\mathcal{R}Y$
I	Z	$\alpha\mathcal{L}V$	$\alpha\mathcal{R}Z$	$\beta\mathcal{R}Z$	$\gamma\mathcal{R}Z$

4 A 10×3 UTM

We come back to the original 10 states of Tables 1, 2 and 3 because, with respect to Table 4, some different groupment will be needed. In order to work with only 3 tape–symbols we will now replace the marker γ by the string $\beta\beta$; this choice (say "the marker in $\mathcal{S}$ is twice the marker in $\mathcal{P}$") was already used by Rogozhin in its 10×3 UTM[6]. Concerning row G in Table 2 it is sufficient to suppress the (empty) column γ; concerning row A in Table 1 we just need to (suppress the column γ and) put the Halt in column β; also for row I in Table 2 we confined ourselves to suppress the column γ (thus row I will, as needed, leave unchanged the strings $\beta\beta$); concerning rows E, F of Table 2 we just did a little change: we suppressed of course column γ in both rows, but the (empty) column β in row F needs now to be filled with $\beta\mathcal{R}E$ (thus accomplishing the work of column γ in the old row E). All remaining rows require many steps (and possibly many states; e.g. state H splits into H, H_1, H_2) for the simulation.

[6] see [Ro2], where our symbols $0, \alpha, \beta$ are denoted by $0, 1, b$ respectively

Table 5. States without horizontal separating line can be grouped together, thus getting a 10×3 UTM.

State	0	α	β
A	$\square$	$0\mathcal{L}B$	Halt
H_1	$\beta\mathcal{L}H_2$	$\square$	$\square$
B	$0\mathcal{L}B$	$0\mathcal{L}B$	$\beta\mathcal{L}C$
C	$\alpha\mathcal{R}D$	$0\mathcal{L}B$	$\square$
D	$\alpha\mathcal{R}D$	$0\mathcal{L}B$	$\beta\mathcal{R}D_1$
D_1	$\alpha\mathcal{R}D$	$\square$	$\alpha\mathcal{L}D_2$
D_2	$\square$	$\square$	$\alpha\mathcal{L}D_3$
D_3	$\square$	$\alpha\mathcal{L}E$	$\square$

State	0	α	β
E	$\alpha\mathcal{R}I$	$\alpha\mathcal{L}E$	$\beta\mathcal{L}F$
F	$\alpha\mathcal{L}G$	$\alpha\mathcal{L}E$	$\beta\mathcal{L}E$
H_2	$\square$	$\square$	$\beta\mathcal{L}E$
G	$\alpha\mathcal{R}H$	$\square$	$0\mathcal{R}J_1$
J	$\square$	$0\mathcal{R}J$	$0\mathcal{R}J_1$
J_1	$\square$	$\alpha\mathcal{L}J_2$	$0\mathcal{R}A$
J_2	$\beta\mathcal{R}J$	$\square$	$\square$
H	$\beta\mathcal{R}H_1$	$\alpha\mathcal{R}H$	$\beta\mathcal{R}H$
I	$\alpha\mathcal{L}E$	$\alpha\mathcal{R}I$	$\beta\mathcal{R}I$

Remark 5. Row H_1 could be grouped both with J or with H_2, but it is more convenient to group it with row A, and to absorbe row H_2 into (the new form of) row F. The most tricky simulations concern column β in row J, where we wrote a 0; the choice is not very natural, but allows to treat any case by just one more state (the union of J_1 and J_2). Concerning row G, as already said, it could be treated by simply suppressing the column γ; however nothing forbids of apply to it the same treatment used for row J: in column β we can write $0\mathcal{R}J_1$. The choice, already not so natural in the treatment of row J, can here seem absurd. However it works: the head, in state J_1, will meet an α that forces the TM to come back and write the needed β. In such a way row G can be joined to row J (because we did not join J and H_1); and row H_1 can be joined to row A thus giving raise to the 10×3 UTM of Table 5.

Remark 6. Concerning the structure of the whole machine, we find our TM somewhat simpler than the Rogozhin's one: we do not need to differentiate the treatment for even–length and odd–length productions; and the couple $\beta\beta$ acts simply as a marker[7].

[7] in the Rogozhin's UTM the second β of the couple acts as "first normal symbol α of the following block"; this operation being compensated by the fact that, when writing the marker $\beta\beta$ on the right of $\mathcal{S}$, the leftmost β of the couple overwrites the last α already written...

5 A 19×2 UTM

A fundamental result of Shannon [Sha] states that any Turing Machine with m states and n symbols can be simulated by a 2–symbols TM with $m \cdot \phi(n)$ states, where the function ϕ is given by:

$$\text{let } l \text{ be minimal such } n \leq 2^l; \text{ then } \phi(n) = 3 \cdot 2^l + 2l - 7.$$

Remark 7. The Shannon result is the first step for the construction of small UTMs; the second step being the fact that any 2–symbols TM can be simulated by a tag system, and the third one being that (as we shortly explained) tag systems can be simulated by a small Turing Machine. Details can e.g. found in [Rob], where is also given an easier proof (at the expenses of a worst value for ϕ) of the Shannon's result.
The first step in the Shannon's technique transforms the original tape by replacing each old symbol with its l–digits binary description[8]. If applied to 7×4 TMs the Shannon's formula gives the existence of a simulating 63–states TM; perhaps the factor $\phi(4) = 9$ in the Shannon's formula[9], one can expect that better values should be obtained by using some peculiarity of the table (empty cells, repeated commands,...); in fact, applied to the Robinson's UTM, it can be lowered to a (quite astonishing) value close to 3: we proved in [Bai] that such a UTM can be simulated by a 22×2 TM[10].

Also our 7×4 UTM can be simulated by a 22×2 TM; however, in order to prove a better result, we will follow a different strategy, using variable length codes: the marker β will be replaced by a one–digit symbol, while α, γ and 0 will be replaced by two–digits symbols.

With respect to the original 10 states (see Tables 1, 2 and 3) we will "duplicate" the old symbol α, by replacing it somewhere with an $\overrightarrow{\alpha}$ and somewhere with an $\overleftarrow{\alpha}$, thus getting a 10×5 table; the replacement being done in such a way that arrrows over the α denote the direction along which the Head moves when it arrives to scan such a symbol. It is not a difficult task: in the initial tape the symbol α appears only in $\mathcal{S}$ (thus it will be scanned "coming from the left") and we replace it by $\overrightarrow{\alpha}$; in the table commands like $\alpha\mathcal{R}K$ must be replaced by $\overleftarrow{\alpha}\mathcal{R}K$[11] while commands like $\alpha\mathcal{L}K$ must be replaced by $\overrightarrow{\alpha}\mathcal{L}K$. The result is given in Table 6.

Remark 8. In row J column γ of Table 6 we wrote "$\overleftarrow{\alpha}\mathcal{R}A$" instead of "$0\mathcal{R}A$", as in Table 3; it is quite obvious that this does not affect the behaviour of the UTM. In fact, concerning the next symbol to scan, we have two cases: it can be a γ, or a $\overrightarrow{\alpha}$. In the first case the machine will halt (thus the $\overleftarrow{\alpha}$ we wrote has no relevance); in the second case the machine will come back in state B, where symbols $\overleftarrow{\alpha}$ and 0 are treated in the same way.

[8] of course initial zeros can not be suppressed!
[9] factor which is sharp in the framework of the simulation of *any* 4–symbols TM
[10] a 22×2 UTM was also obtained, by direct construction, in Rogozhin [Ro3]
[11] when the symbol we are writing will be read again, we will arrive to it moving left!

Table 6. An intermediate 10×5 table.

State	0	$\overrightarrow{\alpha}$	$\overleftarrow{\alpha}$	β	γ
A	$\square$	$0\mathcal{L}B$	$\square$	$\square$	Halt
B	$0\mathcal{L}B$	$\square$	$0\mathcal{L}B$	$\beta\mathcal{L}C$	$\square$
C	$\overleftarrow{\alpha}\mathcal{R}D$	$\square$	$0\mathcal{L}B$	$\square$	$\square$
D	$\overleftarrow{\alpha}\mathcal{R}D$	$0\mathcal{L}B$	$\square$	$\beta\mathcal{R}D$	$\overrightarrow{\alpha}\mathcal{L}E$
E	$\overleftarrow{\alpha}\mathcal{R}I$	$\square$	$\overrightarrow{\alpha}\mathcal{L}E$	$\beta\mathcal{L}F$	$\gamma\mathcal{L}E$
F	$\overrightarrow{\alpha}\mathcal{L}G$	$\square$	$\overrightarrow{\alpha}\mathcal{L}E$	$\square$	$\square$
G	$\overleftarrow{\alpha}\mathcal{R}H$	$\square$	$\square$	$\beta\mathcal{R}J$	$\square$
H	$\gamma\mathcal{L}E$	$\overleftarrow{\alpha}\mathcal{R}H$	$\square$	$\beta\mathcal{R}H$	$\gamma\mathcal{R}H$
I	$\overrightarrow{\alpha}\mathcal{L}E$	$\overleftarrow{\alpha}\mathcal{R}I$	$\square$	$\beta\mathcal{R}I$	$\gamma\mathcal{R}I$
J	$\square$	$0\mathcal{R}J$	$\square$	$\beta\mathcal{R}J$	$\overleftarrow{\alpha}\mathcal{R}A$

With respect to Table 6, let us now rename the five symbols 0, $\overrightarrow{\alpha}$, $\overleftarrow{\alpha}$, β and γ by $''00''$, $''01''$, $''10''$, $''1''$ and $''11''$ respectively; the next step consists in interpreting such new symbols as a single–digit symbol (the $''1''$) or as 2–digit symbols (the remaining ones[12]). Let us point out the main reason for our choice of variable–length codes: independently from the direction we are walking in, when we scan a 0 we are sure it is the starting of a normal symbol; conversely, when scanning a 1, it must be a marker (or possibly the beginning of a marker). With such a remark in mind, it is nothing but a tedious work to check that Table 7 simulates Table 6.

Let us just add a few words about the treatment of state G. In such a state we can meet only a 1 or a 00; in the first case we must (and we do, with the command $1\mathcal{R}J$) leave the 1 unchanged, going right in state J; in the second case we should replace 00 by 10, going right in state H; the command $1\mathcal{L}H$ we wrote, after two more TM's steps, has exactly this effect.

References

Bai. Baiocchi, C.: $3N + 1$, UTM e Tag–Systems. Dipartimento di Matematica dell'Università "La Sapienza" di Roma **98/38** (1998).

Min. Minsky, M.: Computation: Finite and Infinite Machines. Prentice Hall, Englewood Cliffs, NJ, 1967.

[12] in particular: γ is again replaced by $\beta\beta\ldots$

Table 7. States in the rightmost part can be replaced as indicated; joining states C and J_1 we get a 19×2 UTM.

State	0	1	State	0	1	State	0	1
A	$1\mathcal{R}A_1$	Halt				$A_1 \to D_1$	$\square$	$0\mathcal{L}B_1$
B	$0\mathcal{L}B_1$	$1\mathcal{L}C$	B_1	$0\mathcal{L}B$	$0\mathcal{L}B$			
C	$0\mathcal{L}C_1$	$\square$	C_1	$1\mathcal{R}C_2$	$0\mathcal{L}B$	$C_2 \to D_1$	$0\mathcal{R}D$	$\square$
D	$1\mathcal{R}D_1$	$1\mathcal{R}D_2$	D_1	$0\mathcal{R}D$	$0\mathcal{L}B_1$			
			D_2	$1\mathcal{R}D_1$	$1\mathcal{L}D_3$	$D_3 \to E_1$	$\square$	$0\mathcal{L}E$
E	$1\mathcal{L}E_1$	$1\mathcal{L}E_3$	E_1	$1\mathcal{R}E_2$	$0\mathcal{L}E$	$E_2 \to I_1$	$\square$	$0\mathcal{R}I$
			E_3	$1\mathcal{L}F_1$	$1\mathcal{L}E$			
			F_1	$0\mathcal{L}G$	$0\mathcal{L}E$	$F \to E_3$	$1\mathcal{L}F_1$	$\square$
G	$1\mathcal{L}H$	$1\mathcal{R}J$						
H	$1\mathcal{R}H_1$	$1\mathcal{R}H$	H_1	$1\mathcal{L}H_2$	$0\mathcal{R}H$	$H_2 \to E_3$	$\square$	$1\mathcal{L}E$
I	$1\mathcal{R}I_1$	$1\mathcal{R}I$	I_1	$1\mathcal{L}I_2$	$0\mathcal{R}I$	$I_2 \to E_1$	$\square$	$0\mathcal{L}E$
J	$0\mathcal{R}J_1$	$1\mathcal{R}J_2$	J_1	$\square$	$0\mathcal{R}J$			
			J_2	$0\mathcal{R}J_1$	$0\mathcal{R}A$			

Rob. Robinson, R. M.: Minsky's small Turing machine. International Journal of Mathematics **2 (5)** (1991) 551–562.

Ro1. Rogozhin, Y.: Seven universal Turing machine. Systems and Theoretical Programming, Mat.Issled **69** Academiya Nauk Moldavskoi SSR, Kishinev (1982) 76–90 (Russian).

Ro2. Rogozhin, Y.: Small universal Turing machines. Theoretical Computer Science **168-2** (1996) 215–240.

Ro3. Rogozhin, Y.: A universal Turing machine with 22 states and 2 symbols. Romanian Journal of Information Science and Technology **1** (3) (1998) 259–265.

Sha. Shannon, C. E.: A Universal Turing Machine With Two Internal States. Automata Studies (*Shannon & McCarthy Editors*) Princeton University Press (1956) 157–165.