Reversibility of Turing Machine Computations

Zvi M. Kedem*

NYU CS Technical Report TR-2013-956
May 13, 2013

Abstract

Since Bennett's 1973 seminal paper, there has been a growing interest in general-purpose, reversible computations and they have been studied using both mathematical and physical models. Following Bennett, given a terminating computation of a deterministic Turing Machine, one may be interested in constructing a new Turing Machine, whose computation consists of two stages. The first stage emulates the original Turing Machine computation on its working tape, while also producing the trace of the computation on a new history tape. The second stage reverses the first stage using the trace information. Ideally, one would want the second stage to traverse whole-machine states in the reverse order from that traversed in the first stage. But this is impossible other than for trivial computations. Bennett constructs the second phase by using additional controller states, beyond those used during the first stage. In this report, a construction of the new machine is presented in which the second stage uses the same and only those controller states that the first stage used and they are traversed in the reverse order. The sole element that is not fully reversed is the position of the head on the history tape, where it is out of phase by one square compared to the first stage.

1 Introduction

In 1973, Bennett produced a seminal paper [1] in the field of reversible computations. Informally stated, Bennett's goal is to construct a machine performing the same computation as the original machine and then undoing it, while preserving a copy of the useful output of the computation. In this report, the preservation of the output will not be considered, as the focus is on reversibility only.

A terminating computation of a deterministic Turing Machine can be described by a sequence of *whole-machine states*, between the sequential steps of the computation, in the form of

$$s_1, s_2, \ldots, s_{f-2}, s_{f-1}, s_f. \tag{1}$$

One may want to extend the computation to obtain

$$s_1, s_2, \ldots, s_{f-2}, s_{f-1}, s_f, s_{f+1}, s_{f+2}, \ldots, s_{F-1}, s_F, \tag{2}$$

which consists of (1) and its "reverse," in the sense that $F = 2f - 1$ and $s_{1+i} = s_{F-i}$, for $i = 0, 1, \ldots, f-2$, so that (2) is

$$s_1, s_2, \ldots, s_{f-2}, s_{f-1}, s_f, s_{f-1}, s_{f-2}, \ldots, s_2, s_1. \tag{3}$$

However, as the computation is deterministic, this is only possible if $s_{f-2} = s_f$, implying $f = 1$ or $f = 2$. That is, either the system is stationary, or it alternates between two states. But it may be possible to "reverse the arrow of time" and in this way reverse the computation, just as in a classical mechanical system.

*New York University, zvi.kedem@nyu.edu

2 Bennett's Construction

It is assumed that the reader is completely familiar with [1], but as needed various points of that paper will be briefly summarized in this section.

Bennett starts with a 1-tape Turing Machine whose controller's operation is defined by *quintuples* of the form

$$A_j\ T \rightarrow T'\ \sigma\ A_k, \tag{4}$$

where A_j is the "old" state, A_k the "new" state, T is a symbol to be read on the tape, T' is the symbol to be written, and σ is one of $-$, 0, and $+$, respectively indicating left, null, and right shift.

Bennett first modifies the original 1-tape machine by replacing a quintuple with a pair of *quadruples*. Consider a "generic" quintuple, one that does not deal with the initial or the final state. So let the quintuple in (4) be the mth quintuple in some ordering of the quintuples. It is replaced by a pair of quadruples

$$\begin{aligned} A_j\ T &\rightarrow T'\ A'_m \\ A'_m\ / &\rightarrow \sigma\ A_k, \end{aligned} \tag{5}$$

where A'_m is an additional state, and the first quadruple instructs the head to overwrite T with T' and not to shift, and the second quadruple instructs the head to only shift as specified by σ and not to write. Let the machine be $\mathcal{M}$.

Then, Bennett constructs a new machine $\mathcal{M}'$, which has three tapes, *working*, *history*, and *output*, and which operates in three stages. In stage 1, $\mathcal{M}'$ behaves just like $\mathcal{M}$ and performs the computation using the working tape while also writing a trace of the computation on the history tape. In stage 2, $\mathcal{M}'$ copies the working tape, which now contains the output of the computation, onto the output tape. In Stage 3, $\mathcal{M}'$ restores the working tape and the history tape. The quadruples defining $\mathcal{M}'$ are listed in Table 1 of [1].

Consider a trivial modification of $\mathcal{M}'$, still to be called $\mathcal{M}'$, that is just a restriction of Bennett's construction to two tapes and reversibility (without copying the output onto the third tape). The quadruples of $\mathcal{M}'$ are listed in Table 1. The two symbols in brackets refer to the two tapes.

The machine will proceed in two stages:

1. In stage 1, the "compute" stage, $\mathcal{M}'$ executes the computation as $\mathcal{M}$ did using the working tape, while writing a trace of the computation on the history tape. This stage corresponds to Bennett's first stage.
2. In stage 2, the "restore" stage, $\mathcal{M}'$ uses the information on the history tape to restore the original values to the working tape, while erasing (and thus restoring to the original blank values) the history tape. This stage corresponds to Bennett's third stage.

To enable the computing and the restoring, the two quadruples of (5) are replaced by four quadruples:

1. For stage 1:

$$\begin{aligned} A_j[T\ \ /\ ] &\rightarrow [T'\ \ +\ ]A'_m \\ A'_m[/\ \ b\ ] &\rightarrow [\sigma\ \ m\ ]A_k. \end{aligned}$$

2. For stage 2:

$$\begin{aligned} C_k[/\ \ m\ ] &\rightarrow [-\sigma\ \ b\ ]C'_m \\ C'_m[T'\ \ /\ ] &\rightarrow [T\ \ -\ ]C_j. \end{aligned}$$

2.1 Example

Here and subsequently, when a reference to the quintuples labeled with m is made, it is assumed that $2 < m < N - 1$, so that a "generic" pair of instructions can be discussed.

Consider an example of a small part of the computation. Let, then, $\mathcal{M}$ contain quadruples

$$\begin{aligned} A_j\ T &\rightarrow T'\ A'_m \\ A'_m\ / &\rightarrow -\ A_k. \end{aligned}$$

Note the choice of σ as the left shift in this example. Then, in $\mathcal{M}'$, there will be quadruples:

Table 1: Table 1 of [1] restricted to two tapes and slightly reformatted. Note also that in the Nth pair of quadruples in the Compute Stage, A_f is replaced by C_f.

Stage		*Quadruples*	*Contents of tape*	
			Working tape	*History tape*
			_INPUT	_
Compute				
	1)	$\begin{cases} A_1\,[b \quad /\,] \to [b \quad +\,]\,A'_1 \\ A'_1\,[/ \quad b\,] \to [+ \quad 1\,]\,A_2 \end{cases}$		
		$\vdots$		
	m)	$\begin{cases} A_j\,[T \quad /\,] \to [T' \quad +\,]\,A'_m \\ A'_m\,[/ \quad b\,] \to [\sigma \quad m\,]\,A_k \end{cases}$		
		$\vdots$		
	N)	$\begin{cases} A_{f-1}\,[b \quad /\,] \to [b \quad +\,]\,A'_N \\ A'_N\,[/ \quad b\,] \to [0 \quad N\,]\,C_f \end{cases}$		
			_OUTPUT	HISTORY
Retrace				
	N)	$\begin{cases} C_f\,[/ \quad N\,] \to [0 \quad b\,]\,C'_N \\ C'_N\,[b \quad /\,] \to [b \quad -\,]\,C_{f-1} \end{cases}$		
		$\vdots$		
	m)	$\begin{cases} C_k\,[/ \quad m\,] \to [-\sigma \quad b]\,C'_m \\ C'_m\,[T' \quad /\,] \to [T \quad -\,]\,C_j \end{cases}$		
		$\vdots$		
	1)	$\begin{cases} C_2\,[/ \quad 1\,] \to [- \quad b\,]\,C'_1 \\ C'_1\,[b \quad /\,] \to [b \quad -\,]\,C_1 \end{cases}$		
			_INPUT	_

1. For stage 1:

$$A_j[T\ /] \to [T'\ +]A'_m$$
$$A'_m[/\ b] \to [-\ m]A_k.$$

2. For stage 2:

$$C_k[/\ m] \to [+\ b]C'_m$$
$$C'_m[T'\ /] \to [T\ -]C_j.$$

In Fig. 1, there are three consecutive states of the computation from stage 1 and the corresponding three consecutive states from stage 2. Only the relevant squares, two in number, are depicted for each of the two tapes.

Looking at the upper part of the figure, illustrating a fragment of stage 1: the controller is in state A_j; H_1, the head assigned to the working tape, is on a square containing symbol T; and H_2, the head assigned to the history tape, is on a square whose contents is not specified, though it is known that this is the last square written on the history tape, and the square to the right of it contains a blank, b. Then, see the evolution during two steps is shown. Note that H_1 shifted one square to the left, as instructed by a quadruple, and the square immediately to the right of H_2, and not indicated on the figure, contains a b.

Looking at the lower part of the figure, illustrating the corresponding fragment of stage 2, all the squares to the right of H_2 have been restored to the original values of b and the machine will now undo a fragment of the computation

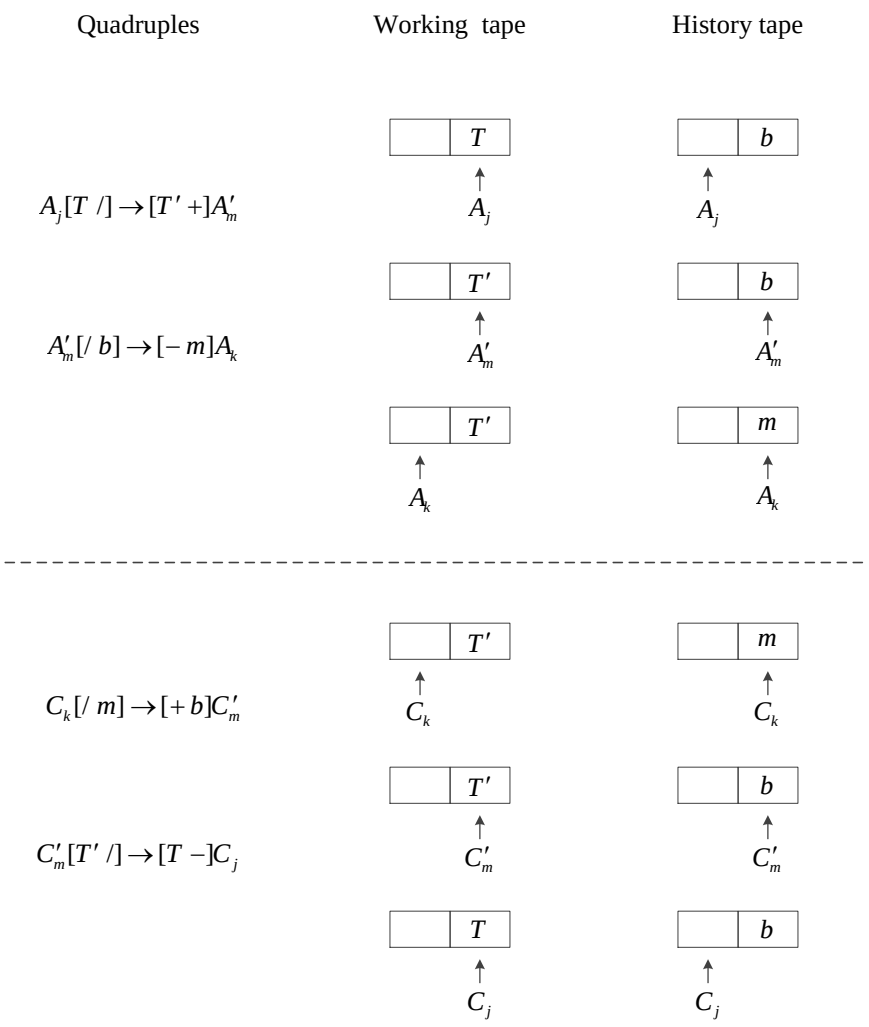

Figure 1: A fragment of stage 1 and the corresponding fragment of stage 2 for $\mathcal{M}'$.

just described above. The controller is in state C_k. H_1 is on a square whose contents is unspecified, and to its right is a square with T', which needs to be replaced by T. The information to do that is in the square on which H_2 is. The first quadruple instructs H_1 to shift to the right and H_2 to write a b. The second quadruple instructs H_1 to overwrite T' with T and instructs H_2 to shift to the left.

3 Bennett's Construction Modified

So far, Bennett's construction was described. Now, by building on Bennett's construction, it will be shown that it is possible to time-reverse a Turing Machine computation so that there are no new states in stage 2, and the traversal of states in stage 2 is the reversal of their traversal in stage 1.

Before describing the modified construction, it is useful reviewing Bennett's reason for the replacement of quintuple (4) by a pair of quadruples (5). The sequence of operations in a quintuple during a single step involves in general two squares:

1. Read the symbol on the current square.
2. Write a symbol on that square.

Table 2: Table 1 modified, with quadruples replaced by quintuples. Letter x stands for any symbol that could be on a tape, so that any quintuple with x stands for a family of quintuples, one for each possible value of x on the tape. Note for future reference that H_2 does not shift in instructions N of stage 1 but shifts to the left in instructions N of stage 2.

Stage	*Quintuples*	*Contents of tape*	
		Working tape	*History tape*
		_INPUT	_
Compute			
1)	$\begin{cases} A_1\,[b \quad b\,] \rightarrow [b\,0 \quad b\,0\,]\,A'_1 \\ A'_1\,[b \quad b\,] \rightarrow [b+ \quad 1+\,]\,A_2 \end{cases}$		
	$\vdots$		
m)	$\begin{cases} A_j\,[T \quad b\,] \rightarrow [T'\,0 \quad b\,0\,]\,A'_m \\ A'_m\,[T' \quad b\,] \rightarrow [T'\,\sigma \quad m+\,]\,A_k \end{cases}$		
	$\vdots$		
N)	$\begin{cases} A_{f-1}\,[b \quad b\,] \rightarrow [b\,0 \quad b\,0\,]\,A'_N \\ A'_N\,[b \quad b\,] \rightarrow [b\,0 \quad N\,0\,]\,A_f \end{cases}$		
		_OUTPUT	HISTORY_
Retrace			
N)	$\begin{cases} A_f\,[b \quad N\,] \rightarrow [b\,0 \quad b-\,]\,A'_N \\ A'_N\,[b \quad x\,] \rightarrow [b\,0 \quad x\,0\,]\,A_{f-1} \end{cases}$		
	$\vdots$		
m)	$\begin{cases} A_k\,[x \quad m\,] \rightarrow [x\,{-\sigma} \quad b-\,]\,A'_m \\ A'_m\,[T' \quad x\,] \rightarrow [T\,0 \quad x\,0\,]\,A_j \end{cases}$		
	$\vdots$		
1)	$\begin{cases} A_2\,[x \quad 1\,] \rightarrow [x- \quad b-\,]\,A'_1 \\ A'_1\,[b \quad b\,] \rightarrow [b\,0 \quad b\,0\,]\,A_1 \end{cases}$		
		_INPUT	_

3. Shift.

As stated in [1], the reversal of these actions is

1. Shift.
2. Read a symbol on the current square.
3. Write a symbol on that square.

But this is not allowed under the original definition of quintuples. Bennett thus "decomposes" the quintuple into two quadruples. The first performs

1. Read the symbol on the current square.
2. Write a symbol on that square.

The second performs

1. Shift.

During stage 1, H_1, on the working tape, in general, changes the direction of its shifts multiple times. During stage 2, these shifts need to be reversed, and a shift occurs before a symbol can be read and overwritten based on the specifications of the original quintuple.

Note, that in contrast, the movements of H_2 on the history tape follow the same pattern for every machine. Looking at Table 1, note that during stage 1, H_2 shifts only to the right, and during stage 2, it shifts only to the left. Therefore, the behavior of H_2 can be specified using the original-form quintuples.

This provides significant advantage: the direction of the shift can depend not only on the state of the controller, but also on the symbol read by H_2. This makes it natural to produce a specification of the machine so that it behaves differently depending on the stage the machine is in.

Table 1 is modified to obtain Table 2. This modification of $\mathcal{M}'$ will be called $\mathcal{M}''$. In its definition, quadruples are replaced by quintuples of the form (using Bennett's notation)

$$A_\alpha[\,\beta \quad \gamma\,] \to [\,\delta\,\epsilon \quad \zeta\,\eta\,]A_\theta,$$

which means: *if* the machine is in state A_α, H_1 reads β, and H_2 reads γ, *then* H_1 writes δ and shifts by ϵ, and H_2 writes ζ and shifts by η, and the new state is A_θ.

Examining Table 2, note that the quintuples do not add anything new to the behavior of H_1. They are just the quadruples of Table 1 written more cumbersomely. However, the situation is quite different with H_2. Looking at the two quintuples of m in stage 1, H_2 does nothing in the first quintuple and both writes and shifts in the second quintuple. In stage 2, it both writes and shifts in the first quintuple an does nothing in the second quintuple.

The key observation is, that as a consequence of the construction, in stage 1, H_2 reads a blank, b, and in stage 2, it reads a non-blank m, $m \neq b$. Therefore the behaviors of the machines, including the direction of shifts, are different in the two stages, even though the controller states are the same in stage 2 as in stage 1.

3.1 Example

Consider now an example, analogous to Example 2.1. So again $\mathcal{M}$ contains quadruples

$$A_j\ T \to T'\ A'_m$$
$$A'_m\ / \to -\ A_k.$$

Then, in $\mathcal{M}''$, there will be quintuples

1. For stage 1:

$$A_j[\,T \quad b\,] \to [\,T'\,0 \quad b\ \ 0\,]A'_m$$
$$A'_m[\,T' \quad b\,] \to [\,T' - \quad m\,+\,]A_k.$$

2. For stage 2:

$$A_k[\,x \quad m\,] \to [\,x\,+ \quad b\,-\,]A'_m$$
$$A'_m[\,T' \quad x\,] \to [\,T\ \ 0 \quad x\,0\,]A_j.$$

In Fig. 2, there are three consecutive states of the computation, from stage 1 and the corresponding three consecutive states from stage 2. Only the relevant squares, three in number, are depicted for each of the two tapes.

Looking at the upper part of the figure, illustrating a fragment of stage 1, the controller is in state A_j, H_1 is on a square containing symbol T, and H_2 is on a square whose contents is a blank, b. Contrast this with Example 2.1, in which it was known that H_2 *was on a square not containing* b. Then, observe the evolution during two steps. Note that H_1 shifted one square to the left, as instructed by a quintuple, and the square under H_2, which is not indicated in the figure, contains a b.

Looking at the lower part of the figure, illustrating the corresponding fragment of stage 2, all the squares to the right of H_2 have been restored to the original values of b and the machine will now undo a fragment of the computation just described above. The controller is in state A_k. H_1 is on a square whose contents is unspecified, and to its the right is a square with T', which needs to be replaced by T. The information to do that is in the square on which H_2 is. The first quintuple instructs H_1 to shift to the right and for H_2 to write a b and to shift to the left. The second quintuple instructs H_1 to overwrite T' with T and instructs H_2 not to do anything.

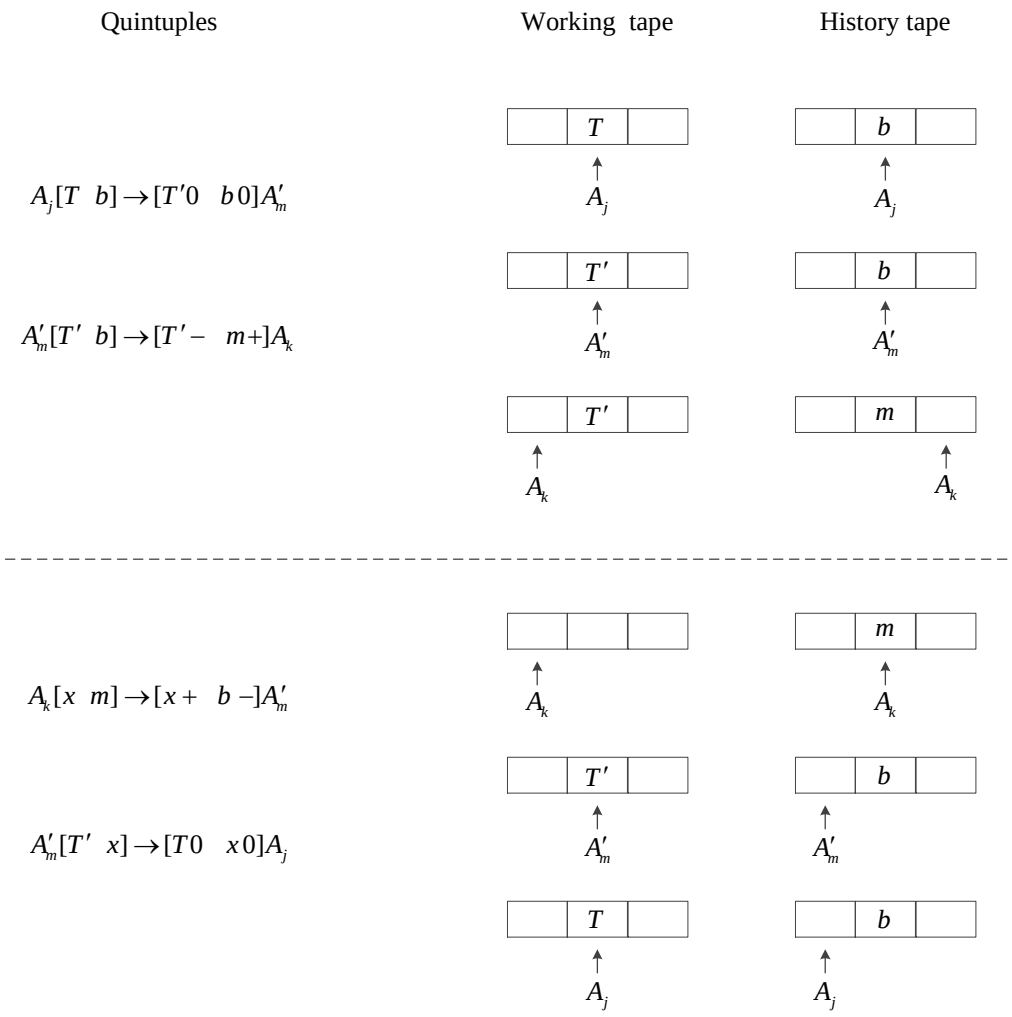

Figure 2: A fragment of stage 1 and the corresponding fragment of stage 2 for $\mathcal{M}''$.

4 Discussion

As stated in Section 1 it is impossible to obtain a computation of the form (3). However, it is possible to obtain something of very close form to it.

A whole-machine state (cf. [1]) is the state of the controller, the contents of the tapes, and the positions of the heads on the tapes. For a 2-tape Turing Machine, this could be a quintuple of the form

$$s = (\alpha, \beta^1, \beta^2, \gamma^1, \gamma^2),$$

where

1. α is the state of the controller.
2. β^1 (respectively β^2) is the description of tape 1 (respectively tape 2). This will be a minimal finite sequence of squares on the tape that includes all the non-blank squares. Note that as the origin of a tape is not specified as part of the definition of a Turing Machine (a tape is homogeneous), it is not meaningful to specify the absolute position of the sequence of squares on the tape.
3. γ^1 (respectively γ^2) is the position of the H_1 (respectively H_2) with respect to the leftmost square of β^1 (respectively β^2). If β^1 (respectively β^2) is empty, then γ^1 (respectively γ^2) is not meaningful and is set to 0.

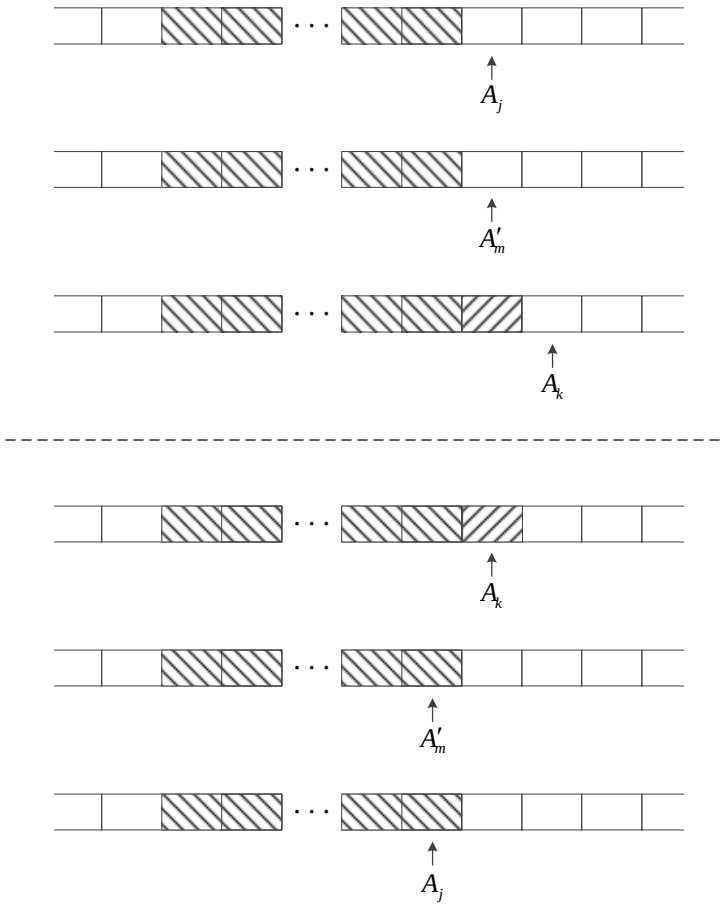

Figure 3: Whole-machine configurations corresponding to Fig. 2. Only tape 2 is shown. Empty squares denote blank squares and shaded squares denote non-blank squares. Squares that are not shown to the left and to the right of the shown squares all contain blanks. The squares indicated by ellipses are shaded and do not contains blanks. The square shaded from lower left to upper right corners contains m. Symbols in other shaded squares are not specified.

In Fig. 3, the values of α, β^2, and γ^2, corresponding to Fig. 2 are shown schematically. Let the *top* machine state in the upper part of the figure take place in step t_1 and the *bottom* machine state in the lower part of the figure take place in step t_2. Then observe that for $i = 0$, 1, or 2.

1. $\alpha_{t_1+i} = \alpha_{t_2-i}$,
2. $\beta^2_{t_1+i} = \beta^2_{t_2-i}$,
3. $\gamma^2_{t_1+i} = \gamma^2_{t_2-i-3}$.

Note that the positions of H_2 are not the same for the corresponding whole-machine states.

The reason for the "out of synchronization" of H_2 is the difference between instructions corresponding to N in stage 1 and stage 2 of Table 2. In stage 1 during the two instructions in N, H_2 does not move, and in stage 2 it moves one square to the left. Thus, during stage 2, it will be "ahead" of the other elements of the whole-machine state.

Note that after $\mathcal{M}''$ reaches the end of stage 2, H_2 is one square to the left of the square it was on at the beginning of stage 1. But as tape 2 is empty and there is no origin, in both cases $\gamma^2 = 0$.

Reference

[1] Bennett, C. H. Logical reversibility of computation. *IBM Journal of Research and Development* **17**, 525–532 (1973).