A SIMPLIFIED UNIVERSAL TURING MACHINE

By

E. F. Moore
Bell Telephone Laboratories, New Jersey

1. INTRODUCTION

In 1936 Turing[1] defined a class of logical machines (which he called a-machines, but which are now generally called Turing machines) which he used as an aid in proving certain results in mathematical logic, and which should prove of interest in connection with the theory of control and switching systems.

Given any logical operation or arithmetical computation for which complete instructions for carrying out can be supplied, it is possible to design a Turing machine which can perform this operation.

A Turing machine is defined to have only a finite number of internal states and to have an infinitely long tape on which it can read, write, and erase symbols, and which it can move one space at a time in either direction.

Turing suggested that such a machine is an abstract mathematical model of a human being assigned to performing a computation, with the states of the machine corresponding to the states of mind of the human, and the symbols on the tape corresponding to the numerical answers and to the intermediate results on scratch paper[2].

Developments since the time of Turing's paper have been in the direction of digital computers which show a much more direct resemblance to Turing machines than do human beings. In fact, several present-day digital computers do actually use magnetic or perforated paper tapes as auxiliary memories, instead of merely as input-output media. Hence, a Turing machine could also be considered a mathematical model[3] of a digital computer.

2. Multiple-tape Turing Machines

I will modify Turing's definition of the machines considered by permitting generalized Turing machines to employ more than one tape, and permitting some of the tapes to be in the form of closed loops (like a conveyor belt, or a rubber band) rather than being infinite.

I will use the term "ordinary Turing machine" to refer to what Turing called an "a-machine", and the term "multiple-tape Turing machine" to refer to a machine modified as above.

A multiple-tape Turing machine can operate more like those digital computers which have several tape drives, or which have cyclic memories (such as mercury delay lines, magnetic drums, or loops of punched paper tape). It can easily be shown that the class of calculations which can be performed on ordinary Turing machines is exactly the same as the class that multiple-tape Turing machines can handle, although the method of handling the calculations must be different, to permit all of the information to be stored on one tape. The objectives which can be attained by having more than one tape are to increase the similarity in method of operation between this abstraction and certain actual machines, and (in the example given later in this report) to simplify the internal structure and the method of action of a machine which performs some specific task.

---

(1) Turing, A. M., On computable numbers, with an application to the Entscheidungs problem, Proc. Lond. Math. Soc. Series 2, Vol. 24, pp.230-265, 1936.

(2) This, like most other mathematical models, appears to differ from the object imitated in several auxiliary properties. For instance, the human can make mistakes in computation, find some of his own mistakes, ask for a raise, ask whether the problem is really worth solving, and invent short-cut methods.

However, on closer analysis, it would seem a reasonable conjecture that sufficiently complicated mathematical models, somewhat like Turing machines could eventually be rigorously shown to have properties of the sort mentioned.

(3) The computers again differ from these abstract mathematical machines, chiefly in making mistakes (i.e., machine breakdowns and transient errors). Also, the Turing machines are unnecessarily general in their properties, since no existing physical machine can have an arbitrarily long tape. However, Turing machines can be designed which behave as if they did not have the infinite tape. If no such restriction is imposed in the Turing machine, it corresponds to a human attendant operating an actual computer by removing and replacing reels of tape whenever the machine indicates this should be done, thus removing the limitation imposed by the mechanical properties of the machine.

3. Universal Turing Machine

In addition to considering Turing machines which would perform various other mathematical operations, Turing gave a description of a universal Turing machine, which was a single ordinary Turing machine which could perform any operation which any other ordinary Turing machine could perform[4], even in case the other machine was more complicated than the universal machine.

This ability to make a relatively simple machine act like a more complicated machine is achieved by giving the simple machine complicated instructions. In particular, the universal Turing machine is given on one part of its tape a complete symbolic description of the machine it is expected to imitate. Then the universal machine stores on another part of its tape (for instance, on alternate squares) a copy of the tape that would be on the imitated machine, and makes the changes on this part of the tape which the imitated machine would make. The remaining part of the tape must be used for intermediate scratch work, for instance to record what state the imitated machine is in, what part of the tape it is scanning, etc. The internal structure of the universal Turing machine has to include instructions to use these various kinds of data, and to move back and forth between the different parts of tape.

Since the method of storing all of this information on one tape is rather complicated, the internal structure of the universal Turing machine which Turing described is also rather complicated, requiring a large number of states.

The present report describes a universal Turing machine which only has fifteen states, at the expense of being a multiple-tape machine. This simplification is accomplished by putting different kinds of information on different tapes which can be moved independently, permitting all the information required at each step to be at the reading heads simultaneously.

4. Conventions for Ordinary Turing Machines

The following conventions concerning the description and operation of ordinary Turing machines are slightly simplified from those given by Turing, and are in accordance with those used by Davis[5]. The machine is assumed to have a tape which has been subdivided into squares the width of the tape, each square containing exactly one symbol. The machine is able to read what is on only one square of the tape at any given time. The machine has a finite number of internal states, $q_1, \ldots\ldots, q_n$, and has a finite number of symbols[6], S_0, S_1, ... S_m, which it can print on the scanned square of the tape.

The next move that the machine will make is assumed to be determined by the internal state of the machine and by the symbol scanned. This ordered pair (an internal state q_i and a symbol S_j scanned) will be called a determinant, and the next action taken is to be determined only by the present determinant of the machine.

There are only m + 3 different kinds of operations which the machine can perform in any one step. These are the operation R, in which the machine moves[7] to the right along the tape (i.e., it changes from scanning its present square to scanning the one immediately to the right of this), the operation L, in which the machine moves to the left along the tape, and the m + 1 operations S_k (defined for k = 0, 1, ... m) which erase the symbol on the square now scanned, and print in its place the symbol S_k.

After the machine performs one of these m + 3 operations listed above, it goes into a specified state q_ℓ, and begins the cycle again (i.e., it now performs an operation based on its new determinant).

The determination as to which operation the machine will perform and which states it will go into are given by the description of the Turing machine, which is a finite list of quadruples of the form -

$$q_i S_j X q_\ell$$

where i = 1, ... , n; ℓ = 1, ..., n; j = 0, ..., m; and X is any Turing machine operation (i.e., X = R, X = L, or X = S_k, for k = 0, ... m). The verbal interpretation of such a quadruple is that if the machine is in state q_i scanning symbol S_j, it should next perform operation X and go into state q_ℓ.

It should be noted that the first two terms of each quadruple are the determinant, and hence for the machine to be able to act consistently according to its description, we must impose the condition that no two quadruples can begin with the same determinant. This condition will be called the consistency condition, and will be taken to be part of the definition of an ordinary Turing machine.

This completes the definition of ordinary Turing machines in general, but

---

(4) It could be interpreted, loosely speaking, as a completely general-purpose digital computer.

(5) Martin Davis. Mimeographed lecture notes on recursive function theory, University of Illinois, 1951.

(6) Blank tape (i.e., tape which has not yet been printed on by the machine) is assumed to originally contain the blank symbol S_0 on each square.

(7) The somewhat confusing convention that the machine moves relative to the tape, instead of vice versa, is used here only to avoid changing the conventions established by Turing and Davis.

there is one other modification that will be made to simplify the construction of the universal Turing machine given in this report. This is to restrict m to be equal to 1, i.e., to permit only two symbols S_0, and S_1 on the ordinary Turing machine considered. To simplify the notation, we will let $S_0 = 0$, and $S_1 = 1$, and note that 0 is the symbol present on blank tape.

This restriction does not cut down the generality of the machines considered, since if any operation can be performed on a machine having m + 1 symbols, it can be performed on a machine having only two symbols, by replacing each symbol in the original machine by a block of binary digits, whose length is s, the least integer $\geq \log_2 (m + 1)$, i.e., by coding the original machine on a binary basis. In order to do this and remain within the scope of the definition given for a Turing machine, it is necessary to replace each quadruple of the original machine with a set of at most 3s - 2 quadruples (taking care not to violate the consistency condition) which scan through the block of binary digits to identify it and which step along performing a series of operations equivalent to the operation of the original machine.

Finally, before describing the universal Turing machine which can imitate any ordinary Turing machine having only two symbols, let us define precisely how the description of the ordinary Turing machine will be written on a tape, using only binary digits.

The tape description is obtained from the list of quadruples that completely defines the machine, by the use of a rule translating the quadruples into 1's and 0's in such a manner that will permit their use easily in the universal machine.

The determinant q_i0 (or q_i1) which begins each quadruple is translated into a block of $3i + 1$ (or $3i + 2$, respectively) successive 1's along the tape.

The operation which is indicated in the third term of each quadruple is translated into a string of 0's which immediately follow the above described block on the tape. The translation is as follows -

0	0
1	00
L	000
R	0000

The next state q_ℓ is translated as a block of 3ℓ successive 1's, immediately following the above string of 0's.

For instance, the following are translations of a few quadruples -

q_101q_2 111100111111

q_21Lq_1 11111111000111

To write the tape description of an entire ordinary Turing machine on a tape, we write the translations of the quadruples in any order, separating each from the next with any number of 0's.

It should be noted that the above tape descriptions have the property that a block of N 1's represents a determinant if, and only if, N is not divisible by 3.

5. A Multiple-tape Universal Turing Machine

Let us consider a Turing machine having three tapes, which will be called T_1, T_2, and T_3, on each of which only the symbols 0 or 1 can occur. The description of this machine is given by a list of sextuples below, with the first four items in each representing the determinant, which in this case must consist of the internal state and the symbol scanned on each of the three tapes. For instance q_81101 is the determinant which indicates that the sextuple in which it occurs is to have effect only if the machine is in internal state q_8, scans 1 on T_1, scans 0 on T_2, and scans 1 on T_3.

As an added convention θ is used in the determinant to indicate that the sextuple is to have effect regardless of whether the corresponding tape shows a 0 or a 1.

The fifth symbol in each sextuple is the operation which is to be performed next. These operations are given as in the case of the ordinary Turing machine, except that a subscript indicates which tape it is to apply to. Thus R_1 indicates a step to the right along T_1, and 0_3 indicates the printing of a 0 on T_3.

Finally the sixth symbol in each sextuple gives the next state of the Turing machine.

The complete list of sextuples of the machine is given below, with explanations of the operations. The list is broken up into parts A, B, and C to permit over-all explanation of the parts. The sequence of steps which this machine goes through in part A, part B, and then part C will perform one step of the actions of the Turing machine being imitated.

In order to use this machine, the tape T_1 should be a loop of tape, containing the description of the machine being imitated, T_2 should be an infinite tape which is blank except for containing the determinant of the machine being imitated (written as a string of 1's, as described above), and T_3 should be a copy of the infinite tape which would be on the machine being imitated.

The three tapes are used for the purposes indicated above at each step, except that T_2 will contain the sequence of all the past determinants of the machine being imitated, only the most recent of which is used at any step.

As an aid in following the written description below, Figure 1 shows most of the same information in diagrammatic form.

6. Description of the Machine

Part A

In states q_1 through q_6 the multiple-tape Turing machine is search-

ing along T_1 to find the quadruple that will be pertinent to the next operation of the machine it is imitating.

It is done by searching along T_1 to find a block of 1's having the same length as the block on T_2.

$q_1 1 1 \theta R_2 q_2$ The machine steps along to the right on T_1 and T_2
$q_2 1 1 \theta R_1 q_1$ alternately, starting in state q_1 at the left end of the block of 1's on each tape.

$q_1 1 0 \theta L_2 q_3$ If the machine reaches 0 on T_2 while within the block of 1's on T_1, the block on T_1 was shorter, and the machine begins preparation for comparing the next block on T_1 with the same block on T_2 by moving back one space to return to the block on T_2.

$q_3 1 1 \theta R_1 q_3$ The machine moves along toward the right end of the block of 1's on T_1.

$q_3 0 1 \theta L_2 q_4$ When the machine has just passed the right end of the block of 1's on T_1, it moves back toward the beginning of the block on T_2.

$q_1 0 1 \theta L_2 q_4$ If the machine reaches a 0 on T_1 while still within the block of 1's on T_2, the block on T_2 was shorter, and the machine moves back toward the beginning of the block on T_2.

$q_4 0 1 \theta L_2 q_4$ The machine continues moving to the left on T_2, to return to the beginning of the block of 1's on T_2.

$q_4 0 0 \theta R_1 q_5$ As soon as the machine has passed the beginning of the block of 1's on T_2, it begins moving along toward the next block.

$q_5 0 0 \theta R_1 q_5$ The machine continues past the 0's toward the next block on T_1.

$q_5 1 0 \theta R_2 q_1$ As soon as the machine has reached the beginning of the next block on T_1 it moves back to the beginning of the original block on T_2 to begin making the next comparison of blocks.

$q_1 0 0 \theta R_2 q_6$ If the comparison reaches the end of both blocks of 1's simultaneously, the blocks were of equal lengths, and hence the proper quadruple has been found. T_2 is moved to the right to put a space between the block which has just been used and the next block that will be printed.

Part B

In states q_6 through q_{11} the multiple-tape Turing machine examines the number of 0's immediately following the block of 1's just located on T_1 which indicate what operation the ordinary Turing machine being imitated is expected to perform on its tape. The corresponding operation is then performed on T_3.

$q_6 0 0 \theta R_1 q_7$ The multiple-tape Turing machine is scanning the first 0 following the determinant block on T_2. In order to begin counting how many successive 0's there are, the machine moves to the right on T_1.

$q_7 1 0 \theta 0_3 q_8$ Since the multiple-tape Turing machine has reached a 1 on T_1, there was only a single 0. This indicates that the operation the ordinary Turing machine should perform next is to print a 0, so T_3 takes this action.

$q_7 0 0 \theta R_1 q_9$ There were at least two successive 0's on T_1, so the machine moves to the right along T_1, to continue counting the 0's.

$q_9 1 0 \theta 1_3 q_8$ There were exactly two successive 0's on T_1, so the multiple-tape machine prints a 1 on T_3.

$q_9 0 0 \theta R_1 q_{10}$ The machine continues counting 0's on T_1.

$q_{10} 1 0 \theta L_3 q_8$ There were exactly three successive 0's on T_1, so the multiple-tape machine moves to the left on T_3.

$q_{10} 0 0 \theta R_1 q_{11}$ There were exactly four successive 0's on T_1, so the multiple-
$q_{11} 1 0 \theta R_3 q_8$ tape machine moves to the right on T_3, after having moved along T_1, to the beginning of the next block.

Part C

In states q_8 and q_{12} through q_{15}, the multiple-tape Turing machine prints on T_2 the determinant for the next operation, obtaining it from the symbol on T_3 and the state of the ordinary Turing machine described on T_1. If the length of the block of 1's on T_1 is $3l$, then the number of 1's which must be copied on T_2 is $3l + 1$ or $3l + 2$, depending on whether T_3 is scanning a 0 or a 1.

$q_8 1 0 0 1_2 q_{13}$ If T_3 is scanning a 0, print a single 1 on T_2 before going into state q_{13} to begin copying from T_1.

$q_8 1 0 1 1_2 q_{12}$ If T_3 is scanning a 1, print two 1's on T_2 before going into state q_{13} to begin copying from T_1.

$q_{12} 1 1 1 R_2 q_{15}$

$q_{15} 0 0 0 1_2 q_{13}$

$q_{13} 1 1 0 R_2 q_{14}$ Each time this cycle of three steps is gone through, the machine copies a single 1 from T_1 to T_2, and steps along both tapes to the next position.

$q_{14} 1 0 0 R_1 q_{15}$

$q_{15} 0 0 0 1_2 q_{13}$

$q_{13} 0 1 0 0 L_2 q_4$ When the Turing machine reaches the end of the block of 1's on T_1, the copying is completed. The tapes must now be prepared for the operations in Part A, by moving back toward the beginning of the block on T_2, and entering the cycle of Part A at the appropriate point.

7. Physical Realizability

If it were desired to build a working model of the machine just described, standard teletype tape equipment could not be used, since it does not have the properties assumed. A punch which would be able to operate on tape moving in either direction, and to read from the same line that it is in a position to punch in would be required.

But since holes in punched tape cannot be erased once they are punched, in order to make a machine using punched tape capable of imitating the behavior of an ordinary Turing machine which has this erasing property the coding of the description of the machine would have to be in a more complicated fashion. This has been shown by Shannon (8) to be always possible, although his method of doing it is somewhat complicated.

It should be mentioned, however, that the properties of the tapes assumed in Turing machines are very much like the properties attained by magnetic tapes, which have erasability, reversibility, and the ability to use the same reading head for either reading or writing.

If a tape mechanism were available which had the properties assumed and which could be connected directly to relay circuits, it would be possible to build a working model of this machine using perhaps twenty or twenty-five relays. These figures are mentioned since the number of relays is frequently used as a measure of the complexity of a logical machine, and would enable comparisons to be made between this and actual digital computers.

This estimated number of relays is intended to be sufficiently generous to permit the relay realization of this machine to include the solution of several circuit problems. First, the complexity of the relay sequencing required will probably prohibit many of the possible relay combinations from being useable. Second, the digits being read on each tape must be used in many parts of the circuit, and hence extra relays to follow these digits may be required, and finally, since while the tape is in motion the information as to the present determinant is unreadable, it may be necessary to insert extra states (not in the logical description given) to record each determinant.

8. Conclusion

This machine indicates that very complicated logical processes can be done using a fairly small number of mechanical or electrical components, provided large amounts of a memory medium are available. With present speeds and costs of components, it would not be economically feasible to use such a machine to perform complicated operations because of the extreme slowness and fairly large amount of memory required.

However, with present trends toward the development of high-speed, electronic logical components, this result suggests that it may be possible to reduce the number of components required for logical control purposes, particularly if any cheap memory devices are developed. It would be impossible to draw any precise conclusions from this extreme example as to the extent to which memory and speed can be traded for complexity since no precise quantitative relations between them have been obtained.

---

(8) Unpublished work.

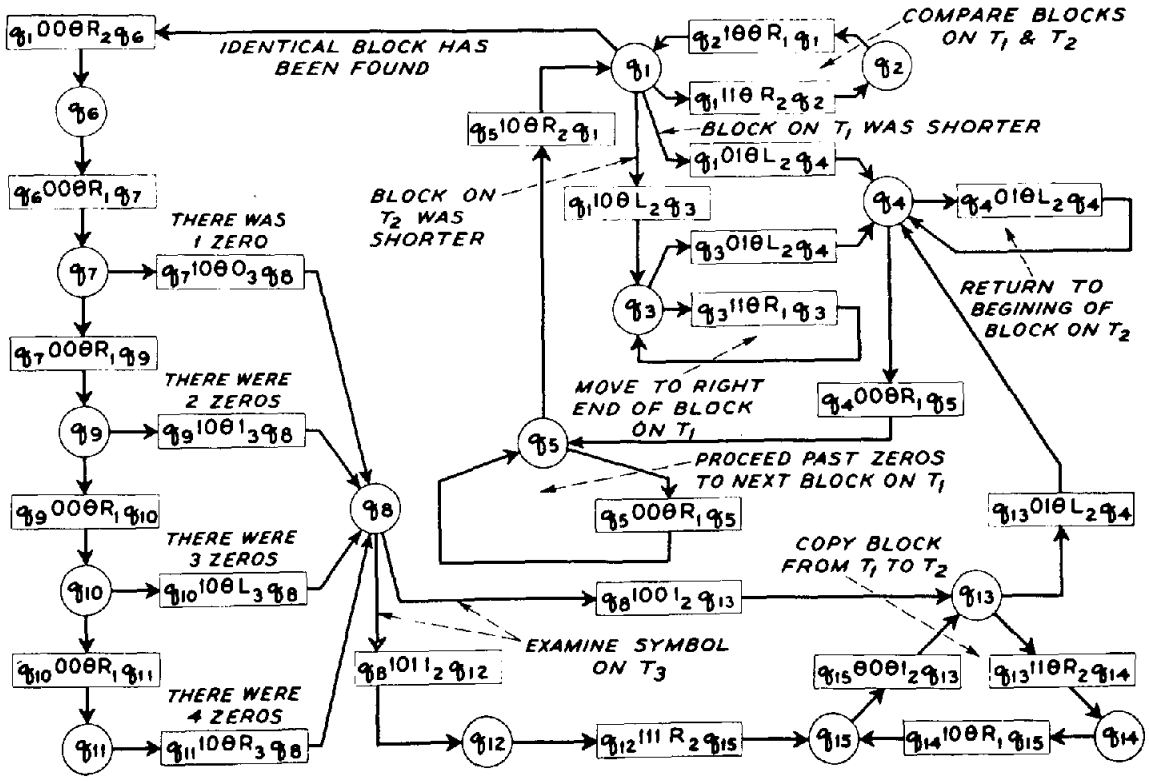

Figure 1

SIMPLE LEARNING BY A DIGITAL COMPUTER

By

The Computation Laboratory
Harvard University

1. Introduction

Digital computers can readily be programmed to exhibit modes of behavior which are usually associated only with the nervous systems of living organisms. This paper describes a concrete example of one practical technique by which the Electronic Delay Storage Automatic Calculator (EDSAC) of the University Mathematical Laboratory, Cambridge, was made capable of modifying its behavior on the basis of experience acquired in the course of operation.

Techniques of this type may have some value for those who, like psychologists and neurophysiologists, are interested in the potentialities of existing digital computers as models of the structure and of the functions of animal nervous systems. The description will be given in two stages. In the first stage (Section 2) the behavior of the EDSAC under the control of a response learning program will be presented from the point of view of an experimenter who can control the input of the machine