Understanding Turing's Universal Machine—
Personal Style in Program Description

P. NAUR
Institute of Datalogy, Copenhagen University, Universitetsparken 1, DK-2100 Copenhagen Ø, Denmark

The study presents the results of an experiment on programmers' descriptions and understanding. The 12 participants, students of computing, have produced notes on Turing's description of his universal machine and, in a second experiment phase, have made mutual evaluations of their notes. The analysis of the resulting notes shows large individual differences of styles and evaluations with respect to all significant issues of the descriptions and understanding, including the evaluations of program descriptions having particularly full formal characteristics. The consequences of the observations for the teaching of programming and for programming methodologies are discussed. The Appendix gives an annotated version of Turing's description of the universal machine.

Received November 1991

1. INTRODUCTION

The primary purpose of the present study is to present some empirical evidence related to programmers' understanding and their use of descriptions in computer programming. Key issues are the importance of individual style differences and of formalized descriptions in programming. A secondary purpose is to demonstrate a technique of empirical investigation of issues of programming.

The programming of a computer, regarded as a human activity, is a matter of the programmer's understanding of certain matters, and dealing with and generating certain descriptions. In developing our understanding of how programming is best pursued and taught, a clear insight into the interplay between various types of descriptions and the understanding achieved by a programmer making use of them is clearly an important issue. As programming methodology is actually pursued, however, this issue seems rarely to be taken up for independent scrutiny. Instead, the typical pattern is that certain types of description or notation are developed and their formal properties ascertained, the implicit assumption being that by satisfying certain formal criteria, such as completeness and consistency in some formal sense, a notation is assured to promote the understanding of the programmer making use of it.

On this background, one relevant question is whether there is a clear correlation between the formal properties of the description forms used by a programmer and the understanding achieved by the programmer.

Another relevant question is the significance of personal factors. To what extent is it valid to claim that a form of description found by one programmer to be valuable and effective will also be found to be so by another programmer? This question posed itself to the present author as a consequence of the experience gained in studying and teaching certain formal notations strongly advocated by well known workers in the field and the finding in the following years that my intensive study of these techniques has left no noticeable trace on my own style of work. Since there is no reason to doubt that the workers who have developed the techniques find them highly effective, this experience suggests that the effectiveness or otherwise of a descriptive technique is strongly dependent on the personality of the programmer using it.

Thus the second main question of the present study is the significance of personality factors for the effectiveness of descriptive techniques in programming.

The presentation has a secondary purpose, that of demonstrating an approach to empirical evaluation of techniques in programming. In pointing this out specifically, the intent is to turn attention to what is here considered a serious defect in a large part of what is currently published as scholarly contributions to computing. The defect is that authors present descriptions of techniques that allegedly will be helpful to other workers in the field, such as programmers and program designers, but fail to present evidence that the techniques are in fact helpful. This feature of currently published papers has been displayed in detail in the case of contributions to a working conference [3], but its dominance in recent publications has been confirmed in an examination of the papers published in ACM Transactions on Programming Languages and Systems, Vol. 9, 1987, and in The Computer Journal, Vol. 31, 1988. From the present point of view it appears that to a large extent both authors and editors of these publications have inadequate notions of what must be required in the way of evidence for a claim to qualify as scientifically sound. With this situation in mind, the present study is presented as one approach to obtaining the kind of empirical evidence related to techniques of programming.
which is felt to be needed. In other words, the suggestion is that the experiment described here should be adapted so as to provide evidence related to the advantageousness claimed for dozens of other notations and techniques promoted in so many papers in recent years.

2. AN EXPERIMENT ON PROGRAM DESCRIPTION AND UNDERSTANDING

If empirical insight into the issues of program description and understanding is to be obtained, what is needed is that actual programmers are put into situations in which their grasp and use of program descriptions and their understanding of programming matters are concretely displayed. So as to achieve this an experiment was conducted as follows.

The programmers participating in the experiment were students of Datalog (Computer Science) at Copenhagen University, choosing the experiment activity as an elective part of their course work.

The experiment activity had two phases. In the first phase the participants worked independently on one and the same given program description, with the set task of developing any supplementary documentation deemed individually appropriate for its understanding, and especially so as to detect and account for errors in the given description. In addition a description of the study activity leading to the documentation had to be submitted.

In the second phase each participant was given a copy of the documentation developed by all other participants during phase 1, and was supposed to formulate a comparative characterization and evaluation of the documentation produced by all participants. In the documentation the participants were identified only by identifying numbers, so retaining mutual anonymity.

The program description given to the participants for study in the first phase was A. M. Turing's description of his universal machine [6], sections 1–7. The rationale behind this choice of study object is that Turing's description is of high intrinsic interest and quality, while it is expressed in terms that are unfamiliar to present-day programmers, and that therefore pose significant problems of understanding and of alternative description. In addition, Turing's description happens to be virtually unknown to present day students, while the present author has found it to contain inconsistencies that may reasonably be attributed to errors. A concrete measure of the understanding of it achieved by a student may thus to some degree be ascertained by the extent to which the student detects its errors. Turing's description, revised and annotated in view of the results of the present experiment, is given in the Appendix.

As a special feature of the second phase of the experiment, the documentation presented to the participants for comparative evaluation included not only that which had been developed by all other participants, but also, unknown to the participants since anonymous, a set of notes to Turing's text prepared by the present author, which includes descriptions having particularly full formal characteristics. Thus one question asked in the experiment is whether highly motivated, well informed readers will recognize the special significance of such descriptions for supporting Turing's arguments. In other words, is the degree of formal completeness of descriptions a decisive issue in the active work of programmers?

3. SURVEY OF THE ANALYSIS REPORTS

The experiment outlined in Section 2 resulted in the submission of analysis reports from 12 participants. The report prepared by the present author enters as the 13th analysis report of the experiment. Some crude facts about these 13 analysis reports are given in Table 1. The authors of the reports will be identified here, as they were during the experiment, by symbols of the form F(n), where n is an integer between 2 and 26. So as to indicate its special significance in the context, the symbol for the present author is marked here with an asterisk: F13*.

As shown in Table 1, the participants F2 and F4 failed to contribute more than the first part of the planned reports.

4. CORRECTIONS TO TURING'S TEXT VERSUS UNDERSTANDING

In the setting of the task to be done by the participants in phase 1 of the experiment, the detection of errors in Turing's text was emphasized. It is taken for granted that a reader's account of the errors in Turing's text will display at least some significant understanding of that text.

On the other hand, it will not be assumed here that there is a simple relationship between a person's understanding of a text and the errors in the text the person is capable of indicating. Indeed, neither error nor understanding can be taken to be strictly definable entities.

<table>
<thead>
<tr>
<th>TABLE 1. The analysis reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author (F number)</td>
</tr>
<tr>
<td>Phase 1</td>
</tr>
<tr>
<td>Notes to Turing's universal machine, pages</td>
</tr>
<tr>
<td>report on study activity, pages</td>
</tr>
<tr>
<td>Phase 2</td>
</tr>
<tr>
<td>report on comparative evaluation, pages</td>
</tr>
</tbody>
</table>

Even so it is assumed that the pattern of errors found by a participant, if taken as a whole, will somehow indicate a significant characteristic of that participant's understanding. With this in view, part of the task of the present analysis is to characterize the patterns of the errors indicated by the participants, so as to display the understanding underlying them.

A survey of all corrections to Turing's text indicated by the participants is given in Table 2. In Table 2 each separate correction is identified by the place in Turing's text where it applies, given by the page number and the number of millimeters measured downwards from the page heading. Elsewhere such places will be written as pairs: (page number, millimeters measured downwards).

The corrections have been grouped into (1) misprints, (2) incompleteness and slips of explanations and formulations, and (3) errors in universal machine. This grouping, which is far from obvious, is justified as follows. The seven misprints were already marked in the copies of Turing's text given to the participants, and so are of no interest as far as their understanding is concerned. They are one result of the present author's work with Turing's text. In retrospect it would have been preferable if they had been left unmarked.

The group (2) is intended to include such inconsistencies of Turing's text that do not directly influence the formulation and proper functioning of his universal machine. It includes omissions of explanations that

<table>
<thead>
<tr>
<th>Table 2. Corrections to Turing's text</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author (F Number)</strong></td>
</tr>
<tr>
<td><strong>(1) Misprints</strong></td>
</tr>
<tr>
<td>238, 104 'E' → 'E'</td>
</tr>
<tr>
<td>244, 102 'q(m)' → 'q(m)'</td>
</tr>
<tr>
<td>245, 52 'q(m)' → 'q(m)'</td>
</tr>
<tr>
<td>245, 145 's' → 's'</td>
</tr>
<tr>
<td>246, 38 'r' → 'r'</td>
</tr>
<tr>
<td><strong>(2) Incompleteness and slips of explanations and formulations</strong></td>
</tr>
<tr>
<td>231, 93 Ends of tape?</td>
</tr>
<tr>
<td>233, 57 Read: 'if and only if'</td>
</tr>
<tr>
<td>233, 62 Read: 'if and only if'</td>
</tr>
<tr>
<td>237, 194 Add: 'if there is no a → B'</td>
</tr>
<tr>
<td>238, 38 Insert comma: '→ E, B'</td>
</tr>
<tr>
<td>239, 9 'q(a, a) fails when no a is on tape'</td>
</tr>
<tr>
<td>239, 98 'e' fails if started on first 'a'</td>
</tr>
<tr>
<td>241, 98 + 118 + 131 + 145 delete rightmost 'c?' = ?'</td>
</tr>
<tr>
<td>241, 192 'r' is not a 'computing machine'</td>
</tr>
<tr>
<td>242, 165 Leftmost DDA incorrect</td>
</tr>
<tr>
<td>243, 95a Initial m-configuration?</td>
</tr>
<tr>
<td>243, 95b Initial tape position?</td>
</tr>
<tr>
<td>243, 102 'the S.D. of the machine'</td>
</tr>
<tr>
<td>244, 190 'w' is capable of printing 'c'</td>
</tr>
<tr>
<td>244, 50 'w = 8'</td>
</tr>
<tr>
<td>244, 192 'the first semi-colon', not 'last'</td>
</tr>
<tr>
<td>244, 140 Test of blank, etc.</td>
</tr>
<tr>
<td>245, 140 Test of blank, etc.</td>
</tr>
<tr>
<td>Number of correct items (2) indicated</td>
</tr>
<tr>
<td>Number of faulty items (2) indicated</td>
</tr>
</tbody>
</table>

(3) Errors in universal machine

1. Missing Initial Blank
235, 70 Blank missing in form (G) (orig. C) |
235, 70 Blank missing in form (C) |
244, 97 Entrance DDA incorrect |
244, 92 Initial blank not printed |
| 'c1' not defined |
| Add definitions of 'c1' and 'c2' |
| 'c2' not defined |
| Semicolon Before First Instruction |
| 243, 113 Read: 'preceded by semi-colons' |
| Tape must start with 'a' |
| Machine Cannot Stop |
| 244, 137 No stop when no instruction applies |
| 'a(0, x, y)' not defined |
| 'a(0, x, y) not defined |
| 'a(1, x, y) not defined |
| Configuration not 'm'-marked |
| 244, 157b + 175 Change form to anf |
| New Blank Tape Not Formed |
| 'a(1, x, y) not defined |
| Number of items 1—7 of (3) covered: |
| 0 | 3 | 4 | 1 | 7 | 4 | 6 | 4 | 3 | 3 | 0 | 3 | 4 |

Indications: d, direct; i, indirect; f, faulty.
would be needed in a complete account of the universal machine, such as the specifications of the initial m-configuration and the initial tape position. It further includes explanations that contradict formulations that are correct in the universal machine, such as the phrase ‘following the last semi-colon’ at {244, 192}. The issue at {245, 140}, concerning a test of blank, will be further discussed below.

The group (3) collects such erroneous formulations of Turing’s text that directly would prevent his universal machine from functioning properly. The group includes some clear cases where the actual instructions are erroneous as written by Turing. However, some further cases that are less clearly in this category are included. The uncertainty arises where Turing’s text includes contradictory formulations. These cases are further described below.

4.1. The output from the universal machine

Turing’s universal machine at {245, 140} includes a few instructions that are claimed by F8 to be erroneous, by F9 to be superfluous and that by F25 are replaced by a sequence which unconditionally will let the universal machine write the changed symbol whenever it is ‘0’ or ‘1’. F16 in this connection claims that it is nowhere stated that the symbols ‘0’ and ‘1’ must not be erased by a machine, saying that this perhaps is an error.

In order to clarify the present issue, it must be noted that Turing at {231, 148} says: “In some of the configurations in which the scanned symbol is blank (i.e. bears no symbol) the machine writes down a new symbol on the scanned square: in other configurations it erases the scanned symbol. . . . Some of the symbols written down will form the sequence of figures which is the decimal of the real number which is being computed. The others are just rough notes to assist the memory. It will only be these rough notes which will be liable to erasure”. At {232, 130} Turing makes clear that the figures spoken of here are just ‘0’ and ‘1’. Further at {235, 108} he states that in the way he will use machines, figures will only be printed on F-squares, never on E-squares, and that “the symbols on F-squares form a continuous sequence”.

From these conventions it may be deduced that, as intended by Turing, machines will only produce their results by writing the figures ‘0’ or ‘1’ on the first blank to the right of the continuous symbol sequence on the F-squares; the figures ‘0’ and ‘1’ are never written on E-squares and a ‘0’ or ‘1’ once written will never be changed.

It may be noted that these conventions with regard to the proper use of machines are decisive to the success of the universal machine. Indeed, if the machine being simulated includes instructions that would erase a figure ‘0’ or ‘1’, its execution by the universal machine will not produce the same result. This is so because the universal machine works by producing ever new updated versions of the complete configurations, never changing a complete configuration already written. In simulating the instruction of writing of ‘0’ or ‘1’ by a machine, the universal machine, in addition to writing the corresponding, complete updated configuration, writes the actual figure, ‘0’ or ‘1’, as such on the tape. Once written these figures cannot be changed by the universal machine. If a figure ‘0’ or ‘1’ is erased by the machine being simulated, in the simulation the universal machine would perform the corresponding change in writing the updated complete configuration, but would not change the additional actual figure previously written on its tape.

These facts are decisive to understanding the instructions of the universal machine at {245, 140}. The state of the universal machine’s tape when the instruction sh is carried out, apart from the indication in the figure of the symbols written after the last colon, is shown in Figure 1. The symbols written on the tape are shown along the horizontal line, the F-squares just above the line, the E-squares just below it, with the E-square carrying the mark of a symbol held on an F-square appearing immediately below it. When the instruction sh is carried out the marks u, v, x and w, have been applied to the relevant parts of the instruction being simulated and certain parts of the last complete configuration, as indicated in the figure. The task to be carried out by the instruction sh and following is to write the actual figure ‘0’ or ‘1’ on the tape to the right of the rightmost colon, if required by the instruction being simulated.

In order to decide on the correct condition for writing the actual figure, it must further be kept in mind that by the format of the S.D. instructions, every instruction writes a so-called changed symbol, which in the case that the scanned symbol should be left unchanged is the same as the scanned one. Thus the S.D. of a machine will commonly have instructions in which both the scanned and the changed symbols are either ‘0’ or ‘1’. When carrying out such an instruction no figure should be written by the universal machine. The condition for writing the actual figure is that the scanned symbol is blank and the changed symbol is one of the figures ‘0’ or ‘1’. This is precisely the condition programmed by Turing’s instructions at sh.

The fact that the present issue has given rise to faulty corrections of Turing’s text, from several of the participants, is a remarkable illustration of the relation between understanding and description in a programming context. The point is that Turing’s descriptions, his informal notes as well as the instructions of the universal machine, form an entirely coherent notion of his adopted discipline of writing figures, a discipline, moreover, which is essential to the correctness of the universal machine. However, in spite of this strong evidence, Turing’s descriptions of his discipline are claimed to be faulty by several of the participants in the experiment. The reason for this unusual reaction undoubtedly is that this particular discipline is very different from the one used with present-day computers. In other words, even if one presents a
reader with consistent informal and formal descriptions of an issue, if the issue differs radically from one to which the reader is accustomed, there is a high risk that the issue will be misunderstood. Speaking more generally, the present issue is a striking evidence of how understanding is a matter of an interplay between new evidence and the highly personal insight already formed by the person.

4.2. The missing initial blank

The first error of Turing's universal machine to be considered is that of the Missing Initial Blank. The manifestations of this error in Turing's text and in the analysis reports are indicated in Table 2. It appears as an error by the discrepancies between the several descriptions of the tape of the universal machine given by Turing. Because of it the universal machine as given by Turing will run into a never ending search for a letter 'D' on its tape very soon after start up. This 'D' is supposed to represent the blank on the square scanned in the initial configuration, and by not being there creates the present error.

The identification of the present error in the analysis reports tends to be obscured by the error at (242, 165), which in Table 2 has been placed in group (2). This error appears prominently by the difference between the leading symbols, 'DDA', at (242, 165) and the leading symbols 'DA' higher on the page at (242, 99). It is noted as a misprint by participants F4, F9 and F20, who have not noted the Missing Initial Blank.

As shown in Table 2, the manifestations of the Missing Initial Blank have been noted in eight of the analysis reports. However, only the reports by F13* and F17 indicate all three relevant points in Turing's text, and a closer inspection of the analysis reports shows a significant diversion in the manner of treatment of the error. Thus of the eight reports only those by F8, F13*, F15, F17 and F19 suggest to cure the trouble by having the instruction doing the initialization, at (244, 92), print the necessary 'D'.

For author F18 the emergence of the infinite search for a 'D' in a hand execution of the universal machine becomes a total block to the further study.

The two remaining reports, by F16 and F26, proceed in a manner which is particularly striking when related to the question of descriptions. In these reports it is noted that the never ending hunt for the missing 'D' arises in the same manner as a consequence of Error 7, New Blank Tape Not Formed. Thus they present a cure of the trouble consisting of a modification of function con so as to let this function print the missing 'D' on the tape when necessary. This manner of correction is a case of removing the symptom without removing the root of the trouble. The difficulty of this approach would be at once evident if the corresponding correction to the description of the tape of the universal machine were attempted, since it would require that description to be everywhere modified so as to admit the omitted blank as a special case needing exceptional attention. In neither of the two reports by F16 and F26 was such a modification of the description attempted, however. There can be no doubt that in these two authors' understanding the exceptional character of the omitted representation of the blank is clearly present, since they develop the
special modification of function \texttt{con} needed to handle it. The situation thus gives a clear demonstration that a programmer's understanding is only partially dependent on an explicit description of the matters at hand. Even when adopting a modification that makes existing descriptions incomplete, and thus incorrect, the programmer may well proceed without updating the descriptions, relying solely on his or her understanding of the modification.

4.3. Semi-colon as Instruction Delimiter

Error 3 of the universal machine illustrates the difficulty of identifying and counting errors. The difficulty arises because Turing at (240, 140) and (243, 113) says that the instructions are separated by semi-colons, at (241, 98 to 145) shows in four examples that instructions are followed by a semi-colon, while what is needed to make good sense in his universal machine is that each instruction is preceded by a semi-colon. Thus the removal of semi-colons following the instructions, suggested by F15, does not take care of the semi-colon required before the first instruction, while the corrections suggested by F4 and F13*:\p{"that each instruction would be preceded by a semi-colon"}, will take care of the erroneous semi-colon following instructions even without having noticed it. Again F17 cures the error by always having a semi-colon at the start of the tape, which would not remove the semi-colon after instructions.

The correction by F17 just described illustrates an important issue related to descriptions and understanding. By describing the correction in the words: "The tape must always begin with the following character sequence (semi-colon is found on an F-square): @@@@ (\text{"denotes blank\"}, the understanding becomes a strictly local issue, without clear connection to the context which makes it valid. This may be contrasted with the correction of the relevant item of Turing's description at (243, 113) suggested by F4, F9, F13*, F16 and F18: "The S.D. consists of a number of instructions, preceded by semi-colons", which directly indicates how the semi-colons are actually used in the universal machine, as leftmost indications of where on the tape the instructions may be found. It seems that it is proper to distinguish between the understanding behind these two forms of corrections to Turing's text, the difference being one of the fullness of the relations that are seen as relevant.

In other words, understanding is here seen as awareness of relevant relations of the issues of the matter. Differences of understanding may sometimes display themselves in the manner in which items are described, but in other cases they may not be thus visible.

4.4. The machine cannot stop

Error 4 of the universal machine indicated in group (3) of Table 2 is that the machine will not come to a proper state when no instruction applies. Instead the machine will move left on the tape indefinitely, looking for a semi-colon.

This error is related to the incompleteness of the explanation at (239, 9) listed in group (2) of Table 2, to the effect that the \texttt{m-function e (erase)} fails if the machine is scanning the first \texttt{a} when it is applied. The point is that Turing generally leaves it to be implied what the assumptions concerning the position of the tape are, for his \texttt{m-functions} to work properly. A natural assumption would be that whenever an \texttt{m-function} is applied the scanned square will lie to the right of the \texttt{a} pair and to the left of the first pair of blanks. This will take care of the incompleteness at (239, 9) and other similar cases.

Similarly, another natural assumption is that no machine will ever scan a square to the left of the \texttt{a} pair once this pair has been written. By violating this assumption Error 4 becomes an error.

4.5. The Understanding Achieved

Collecting the observations shown in Table 2 and in the previous subsections, the understanding achieved by the participants in the experiment may be characterized as follows.

Of the 12 participants (omitting F13*, the present author), the following nine: F4, F8, F15, F16, F17, F18, F19, F25 and F26, appear to have arrived at a generally adequate understanding of Turing's universal machine. However, this understanding displays itself in an individually unique manner in the reports from the participants, each participant indicating a distinct pattern of errors. This is presented in Table 3, which shows some of the information of Table 2 in a different arrangement. Here the participants are ordered according to the number of errors of the universal machine indicated in their report. The errors themselves are arranged according to the number of reports indicating them. A glance at Table 3 shows that only one pair of analysis reports, that of F8 and F15, indicates the same set of errors in the universal machine. However, these two reports differ entirely in the number of kind (2) corrections given, F15 being remarkable by giving 12 such corrections, five of which are indicated by no other participant.

Thus as the conclusion of this discussion of the errors

\begin{table}
\centering
\caption{Patterns of error indications}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
Participant (F number) & Number of corrections of kind (2) & 2 & 1 & 5 & 7 & 3 & 6 & 4 \\
\hline
16 & d & d & d & d & d & d & . \\
8 & d & d & d & d & . & . & . \\
15 & d & d & d & d & . & . & . \\
17 & d & d & d & . & . & . & . \\
26 & d & d & . & . & d & . & . \\
41 & d & d & . & . & . & . & . \\
18 & d & d & . & . & . & . & . \\
19 & d & d & . & . & . & . & . \\
25 & d & d & . & . & . & . & . \\
\hline
\end{tabular}
\end{table}
found by the participants, even participants who appear to have achieved adequate understanding of Turing's presentation give evidence of a wide range of individual differences of how that understanding is constituted.

5. DESCRIPTION OF THE UNIVERSAL MACHINE IN THE ANALYSIS REPORTS

Having characterized the understanding achieved by the participants, we shall turn to the question, how was this understanding achieved? In particular, in the present section we shall consider the forms of description employed by the participants in the experiment. For the present discussion the forms used for describing the symbols held on the tape of the universal machine are of special interest, since it is in the way these symbols are processed by the instructions of the machine that the most significant errors of Turing's presentation show themselves.

Of the 13 analysis reports, eight make use of some specialized notation for describing the tape of the universal machine. Some of these descriptions are shown by examples in Figures 1 and 2. The specialized notations are described in Table 4 in terms of four important formal characteristics. These four characteristics relate to the manner in which a notation will support an explanation of every detailed step of the universal machine, on the one hand, and will support invariably valid statements related to the universal machine, on the other.

By F17, valid for m-configuration inst of the universal machine processing a specific example:

\[ \ldots \ldots : DVCvCvDxExD A D DwCwDwDwCw:0: \]

By F19, valid for m-configuration inst of the universal machine processing a specific example (including an incorrect 'z', a final colon missing):

\[ \text{inst} \ldots : zD A A D DuLuDyAyAyAy:0 : DxC D A A D \]

By F25, generally valid for m-configuration inst of the universal machine:

\[ \begin{align*}
&\text{(DC...DC... ...)} \\
&\uparrow \text{'v'} \\
&\text{(DC...)} \\
&\uparrow \text{'x'} \\
&\text{(DA...DC...)} \\
&\uparrow \text{'y'} \\
&\text{(DC...DC... ...)} \\
&\uparrow \text{'w'}
\end{align*} \]

**FIGURE 2.** Examples of formal tape descriptions.

<table>
<thead>
<tr>
<th>TABLE 4. Description of the universal machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author (F number)</td>
</tr>
<tr>
<td>--------------------</td>
</tr>
<tr>
<td>Description includes correctness argument, yes/</td>
</tr>
<tr>
<td>Form of tape descriptions:</td>
</tr>
<tr>
<td>production/string figure</td>
</tr>
<tr>
<td>Each tape description is specific to an m-configuration, yes/no</td>
</tr>
<tr>
<td>The tape descriptions include marks, yes/no</td>
</tr>
<tr>
<td>Tape descriptions include the complete tape, not merely a part, yes/no</td>
</tr>
<tr>
<td>Tape description is general, not specific to an execution example, yes/no</td>
</tr>
<tr>
<td>Number of 'yes'</td>
</tr>
</tbody>
</table>

The first characteristic is whether each tape description is specific to an m-configuration. Only if this is affirmed will the tape description be directly related to a particular instruction of the universal machine. The second and third characteristics are whether the description is complete by including the marks and all parts of the tape. Unless complete in both of these respects a description is insufficient as support of the explanation of the instructions. The fourth characteristic is that the description is generally valid, not tied to a particular execution example. Only when this is satisfied will the description give direct support to generally valid statements about the universal machine.

As shown in Table 4 only the analysis report by F13*, the present author, uses tape descriptions having all four characteristics. By including this analysis report side by side with the ones prepared by the participants, the significance or otherwise to the programmers' understanding, of notations that satisfy certain criteria of completeness, is certain to be displayed in the experiment.

Additional information given in Table 4 permits certain relations to be investigated. First, Table 4 includes an indication whether the description of each analysis report includes a correctness argument related to the universal machine. It may be seen that apart from the description by the author, F13*, only the report by F15 includes an explicit correctness argument. The argument given by F15 is not based on a formalized description,
indeed, as indicated in the table, F15 does not present any such. However, F15 does present a formalized description of the tape of example II in Turing's Section 3, as does also F13*.

5.1. Error Detection Versus Description Form

Comparing now the extent of the error detection shown in Table 2 and the use of formalized tape descriptions in the analysis reports given in Table 4, it may be noted that the participants reporting the most errors (apart from F13*), F16 indicating six, and F8, F15, F17 and F26 each indicating four errors, differ entirely in their use or non-use of formalized descriptions. Again, those participants who present formalized tape descriptions satisfying three of the four formal characteristics, F17, F18, F19 and F25, are only moderately successful in indicating errors. Thus it is not that case that success in indicating the errors of Turing's description is related to the use of formalized descriptions.

6. SAMPLE HAND EXECUTIONS AND UNDERSTANDING

As another aspect of how understanding is achieved, in the present section the use of sample hand executions in the analysis reports will be reviewed. Such use is directly suggested by Turing's section 3: Examples of computing machines, which presents not only two specific examples of machines, but also a number of important general conventions of the way they will be used, in terms of illustrative examples. Of the 13 analysis reports, 7 refer to the use of sample hand executions, as quoted below.

F4 in notes to the reader of Turing's report says: "§ 3 shows a couple of simple examples of machines. ... It is well worth the trouble to work them through so as to get used to the way Turing lets machines use their tape".

F16 in notes to the reader of Turing's report says: "Notes to chapter 3: Examples of computing machines. ... It is highly recommended that the two examples are worked through thoroughly...". Describing his own study activity, F16 says: "It is clear from chapter 2 of the present report that Turing's chapters 6 and 7 are difficult to grasp. In my opinion a small example at the beginning of chapter 7 would have been a great help to the understanding. The problem was not so much the notation or the contents of the m-functions, it was the manner of operation of the universal machine which gave trouble. I actually had to go the 'hard' way, trying a small example, so as to achieve clarity".

F17 in notes to the reader of section 7 of Turing's report says: "In what follows I shall use examples to illustrate the way of the operation of the many m-functions". The explanation of the m-functions is in fact supported by eight sample hand executions of the symbols held on the tape of the machine. In a later general comment on the universal machine he says: "It is reasonably easy to form a rough picture of how the machine works, and then make this probable by execution by hand of a single cycle of the main loop of \textbf{U}". In notes on his own study activity he says: "Introduction and § 1–§ 4. By far most of the time was spent on hand execution of all example machines of § 4, so as to grasp their precise manners of work...". "The universal machine, as defined in the text, does not work. ... If one tries to execute \textbf{U} by hand, taking 'DN' as the initial configuration, the machine does not work. After having worked intensely with the article, I cannot understand why it took so long to reveal this problem".

F18 tries to apply the universal machine to a machine having the table 'q_1, S_0, S_7, R_9', which will write the sequence '000000...'. Executing the universal machine by hand he soon correctly arrives at an instruction with no defined action. As a result of the work with this example his work on understanding Turing's article is blocked, since "it is difficult to achieve an understanding of \textbf{U} if one cannot do a hand execution of \textbf{U} on some examples".

In his notes on the study activity he says: "I believe the only way one may understand the arrangement of the machine is by 'executing it by hand'. In a sense it is easier—but not enough—to grasp the abstract idea behind it... One may of course also profit from Turing's own commentaries, but they do not provide the detailed understanding".

F19 explains each of 19 of Turing's m-functions by means of one or more sample hand executions. The operation of the universal machine is illustrated by a sample hand execution covering five full cycles of the machine, explained by 52 different pictures of the tape. In notes on his study activity he several times mentions the need to do hand executions of Turing's machines in order to understand them. He further explains that he was able to understand Turing's section 6: The universal computing machine only after having understood Turing's section 7: Detailed description of the universal machine. It may be noted, however, that even with this stress on hand executions, it is not the case that F19 performs these executions purely mechanically. This is revealed by the fact that the tape contents allegedly produced by Turing's universal machine have a systematic error, in that the 'z' used to mark the semi-colon before the current instruction is not erased at the proper stage, as may be seen by comparing Figures 1 and 2. Further, a final colon is missing in several pictures.

F20 in notes to Turing's report repeatedly recommends the reader to do hand executions of the machines so as to obtain understanding. In notes on his study activity he says: "I have used relatively long time on working with the examples of the article. Working with an example in my opinion gives an understanding of the issues behind the example".

F25 in notes on his study activity says: "What I have used the time for is to do hand executions and to try to understand the concrete program pieces in chapters 4 and 7".

The use of sample hand executions of machines for
achieving understanding, as explicitly displayed in the analysis reports, may be summarized as follows:

Author (F number)
2 4 8 9 13* 15 16 17 18 19 20 25 26
Use of sample hand executions:
(0, none; 3, extensive)
0 1 0 0 0 0 1 3 2 3 3 3 0

In other words one may say that the use of sample hand executions is quite prominent in many, although by no means all, of the analysis reports. Further differences between the participants are visible in the manner in which the sample hand executions are done. In one case the execution is done by purely formal manipulation, as by a machine, while in several other cases the execution seems to be guided by a mixture of formal manipulation and insight into the purpose of the formal process.

How is the use of sample hand executions related to other characteristics of each analysis study? Comparing the figures on the use of sample hand executions given above with (1) the number of errors in the universal machine found in each study, and (2) the number of formal characteristics satisfied by the formalized tape descriptions used in each study, no systematic relation between these characteristics is apparent. In other words, the analysis reports differ among each other in these respects, but suggest no interplay between them.

7. MUTUAL ATTITUDES OF THE EXPERIMENT PARTICIPANTS

In phase 2 of the present experiment the participants were invited to express their opinion on the understandability of the analysis reports produced by all other participants, and to relate that opinion to the forms of description used and the degree of completeness of error detection achieved in the analysis reports. In the present section the findings of participants in these matters will be reviewed.

Considering first the mutual judgement of the understandability of the analysis reports, several of the participants expressed uncertainty with respect to the very notion of understandability, and the actual opinions are phrased so as to indicate that uncertainty. Reducing these opinions to a crude linear scale, from 0 (very difficult to understand) to 4 (easy to understand), as done in Table 5, is therefore a precarious procedure. It seems justified in the present context because only negative conclusions will be drawn from the data thus obtained.

Table 5 gives all the mutual judgements of understandability of the analysis reports. For each report the average score of understandability of that report has been formed and the reports have been ordered in Table 5 along decreasing average understandability. The first thing to note about the understandability scores given in Table 5 is the wide range of the scores given to most of the reports. Even the reports having the lowest average readability scores are judged to be easily readable by some of the participants. In other words, there is hardly any general agreement about the readability of the reports.

Comparing the average readability score with other characteristics of the analysis reports shown in Table 5, taken from Table 2, indicates no significant relation to these indicators of the completeness of the error detection.

The opinions of the participants on the merits of special forms of description range widely. One participant is generally positive: F26: "There is a clear tendency that I find the reports that have short, concise notes to be best understandable. There are 3 reports that use figures, properly speaking: those by F9, F13, and F16. These three are the most easily understood reports".

Two participants qualify their opinion: F8: "Of the special forms of presentation I have been impressed by the runs of F19, even though they take up too much space. The tape pictures of F16 are also good. Both contribute

<table>
<thead>
<tr>
<th>TABLE 5.</th>
<th>The mutual judgements of understandability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Understandability by evaluator (F number)</td>
</tr>
<tr>
<td></td>
<td>(0, very difficult; 4, easy to understand)</td>
</tr>
<tr>
<td></td>
<td>Average</td>
</tr>
<tr>
<td>Report by (F number)</td>
<td>8</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>15</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>19</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>*</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>13*</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>?</td>
</tr>
</tbody>
</table>

to making the report readable. Conversely I find that the
diagrams of F9 confuse more than they explain. I also
find the figures of F13 difficult to read, even though this
in part seems to have been the fault of the typography".
F18: "The reports of best understandability are those that
make use of illustrations and examples and a certain
measure of formal notation, but that deal with these
means in a striving towards what is simple and unpreten-
tentious, and use them sparingly".

Five participants are predominantly negative to the
special notations used: F9: "Understandability is rather
a question of how many details are thrown at the reader,
whether relevant or not (many details tire the reader, and
thus hamper the understanding)"). F15: "In my opinion a
concise language helps the understanding ... there are
several examples of tables/figures/grammars that do not
help the understanding, even though the form of description
as such has good uses". F16: "I don't believe there is any
relation between the forms used in the reports and their
understandability". F17: "The reports having the best
understandability are written in the best language. Clearly
thought out, clearly written; as Turing's article". F25: "In
the reports generally there is relation between their under-
standability and their use of description forms. The more
special notations and drawings are used, the more poorly
understandable the report ... The most easily understand-
able reports are those that are written in pure Danish.
However, one does have to sketch the complete configura-
tions in some way, which is done by everybody".

Considering more specifically the issue of rigorous
argumentation in the analysis reports, as noted in Table 4
the analysis reports F13* and F15 include explicit argu-
ments for the correctness of the universal machine. In
the report by F13* the argument builds on the generally
valid, formal tape descriptions discussed in Section 5,
while in that of F15 the argument is expressed informally.
In both cases the arguments include an explicit statement
of the general form of the contents of the tape of the
universal machine that would justify Turing's formulatio
of his universal machine by an induction proof.

The question of how these general correctness argu-
ments are viewed by the remaining participants in the
experiment can be answered simply: they are not noticed
at all. In none of the nine or 10 reports of mutual
evaluation is there any mention of these particular
features of the analysis reports by F13* and F15. Even
F15, when evaluating the report by F13*, does not pay
any attention to it. In other words, the presentation of
the elements of a rigorous argument in support of
Turing's universal machine is ignored in the mutual
evaluation of the analysis reports.

Turning to another manifestation of formality in the
analysis reports, consider the reactions to the form of
tape description used in the report by F13*, the present
author, shown by an example in Figure 1. As already
noted in the discussion of Table 4 in Section 5, this form
of tape description has properties of formal completeness
that are unique in the context. It gives concrete expres-
sion to virtually all of the conventions introduced by
Turing in his sections 3, 6 and 7, and by displaying
general snapshots of the tape of the universal machine
supports correctness arguments of general validity. At
the same time it is so detailed that it allows direct,
symbol-by-symbol exhibition of the effects of the instruc-
tions of the universal machine, as demonstrated in the
discussion in Section 4.1 above.

How do the participants in the experiment react to
the formally complete tape descriptions of the report by
F13*? The reactions are mixed. Five participants, F9,
F15, F17, F19 and F26, express no particular opinion
on the issue. There are negative reactions by three
participants: F8: "The snapshots just make it more difficult
to follow what happens"; F18: "The diagrams of the
snapshots are exceedingly difficult to understand in detail".
F25: "The more special notations and drawings are used,
the more difficult the report becomes. The most notable
example is F13, which in spite of being among the most
complete reports is made incomprehensible by an unneces-
sary and poor notation".

On the other hand, two participants are positive: F16:
"The contents of the report are brilliant. Especially the
idea of introducing snapshots is good". F20: "...it is a
notation entirely different from Turing's ... once the
notation is understood the notes can help the reader
enormously..."

In this context it is further noteworthy that to some
of the participants minor formal matters of the presenta-
tions, such as the forms used to indicate places in
Turing's text, either line count or linear distance, assume
great importance, even to the extent of provoking expres-
sions of strong dislike.

In summary, on what makes an understandable
explanation of Turing's presentation there is very little
general agreement. Whatever slight tendency for agree-
ment on understandability there is, has no relation,
neither to the success of the report in pointing out
Turing's errors, nor to the extent of use of formalized
descriptions.

The reactions of the participants to a formally com-
plete description of the core of Turing's universal
machine range from eager appreciation to complete
rejection. It most emphatically is not so that a description
having outstanding formal qualities will be universally
accepted and appreciated for its merits, even within a
group of motivated and well informed readers.

8. CONCLUDING DISCUSSION

Even on the basis of the limited material of the present
investigation, the issues taken up for study can be
answered rather firmly. First, the effectiveness of a
particular technique in programming appears to be
overwhelmingly dependent on the personality of the
programmer using it. This seems to hold for manners of
analysis, such as the use of examples, and for the use
of particular forms and techniques of description.
Commonly a technique that has been found by one
programmer to be highly effective, when presented to another programmer has been rejected as worthless.

Second, the properties of a descriptive technique with respect to its support of rigorous proof or its formal completeness have no generally agreed value to programmers. The appreciation of such properties are as subject to the personality of the programmer as anything else.

Third, with respect to the manners of work that are adopted and the techniques of description that are found useful, programmers differ among one another to an extent that defies any grouping into definable styles. Among the 12 participants, one approached Turing’s text by a purely formal interpretation, which because of an error in Turing’s universal machine led to a blocking of the study. Seven of the remaining participants, while also making use of formal interpretation, combined this with a development of the insight into the informal aspects of Turing’s presentation and thereby became able to correct the formal error.

The present observations, if typical of programmers’ attitudes and reactions, have important consequences for the evaluation of programming methodologies and for the teaching of programming. They suggest that any methodology which imposes a particular style or manner of work on the programmers, at best may be taken to be useful to a part of the population of programmers, while any claim to general usefulness or applicability of a methodology is likely to be false. Consequently, presentations of techniques claimed to be helpful to programmers which do not include investigations of their effectiveness in actual use must be considered defective. The observations further suggest that teaching of programming at best can achieve awareness of the students to certain manners of description and procedure, while any attempt to make a mixed population of programmers actually use a particular methodology in their work is likely to fail. More particularly, the observations indicate that formal properties of the techniques of a methodology, such as the use of a formally complete notation, are as subject to the programmers’ idiosyncrasies as anything else.

Acknowledgements

The discussion of the present paper is based on the work of the following participants in the experiment on program description and understanding: Morten V. Christiansen, Karl-Christian Hansen, Kristoffer Høgsbro Holm, Morten Due Jørgensen, Niels Jørgen Kruse, Thomas Nikolajsen, Lars Mosgaard Pedersen, Egil Rosager Poulsen, Jakob Rehof, Mikael Spliid, Jan Steffensen and Magnus Vejlstrup.

REFERENCES


APPENDIX: ANNOTATED VERSION OF TURING’S DESCRIPTION OF THE UNIVERSAL MACHINE

The Significance of Turing’s Description

The present annotated version of Turing’s original description of this universal machine will support the discussion of the experiment on program description. However, there are several other reasons for publishing such a description. For one thing, although Turing’s contributions are widely recognized and referred to, there are many different indications that, even among academic professionals in the field of computing, first hand familiarity with Turing’s description of his universal machine is exceptional. For example, when presenting the present notes at a colloquium at the Department of Computer Science, Brown University, Rhode Island, in 1990, I asked those present who had studied Turing’s original paper to raise their hand and not a single one came up.

Typically the knowledge about Turing’s universal machine among students of computing has been obtained at second hand, in the context of automata theory, as presented, for example, by Hopcroft and Ullman [2]. A presentation in such a context tends to promote two different misunderstandings, however. The first is that Turing’s concept depends on the notions of automata theory and thus that computing, as it is said, has such theory as its foundation. One approach to realizing the fallaciousness of this view has been presented by the present author [4]. Another approach is to study Turing’s original presentation. Such a study will show that Turing erects his construction of a universal machine upon simple, intuitive ideas, independently of any further concepts of automata.

The second misunderstanding promoted by presentations such as the one given by Hopcroft and Ullman is the wedding of the notion of a universal machine in the sense of Turing to that of Turing machines. This misunderstanding is displayed directly by Hopcroft and Ullman’s use of the phrase “universal Turing machine”. The flaw is the suggestion that the universal machine has to be a Turing machine. This, however, is besides the point. A universal machine in the sense of Turing can be written in any programming language. That Turing himself employs what we now call a Turing machine is explained simply by the fact that he had no other notations for describing discrete-state
processes to choose from. However, to a student of a later generation, with an ordinary background of programming, a universal machine in the sense of Turing is best expressed in whatever programming language is familiar. If the student is made to construct, in that programming language, a simulator of another programming language, he or she will have all that is needed to grasp Turing's result, the equipotency of computers. A rehash of Turing's own construction of a universal machine, such as the one given by Hopcroft and Ullman, serves no particular purpose in this connection.

As the second reason for republishing Turing's original presentation, in the heated debates concerning the origin of the concept of a computing device controlled by a program held in its store (see, e.g. Aspray [1]), it is generally ignored that, as expressed by Randell [5], "Turing's famous 1936 paper, in which he introduced the idea of a universal machine, can perhaps be regarded as implying the stored program concept". It appears that there is a need for a wider appreciation of the fact that what Turing calls a machine is what nowadays is called a special-purpose computer, or a computer with a fixed program, while Turing's universal machine, if described in the terms of a later generation, is an interpreter of programs held on the tape of the machine, i.e. in its one and only store, and thus what we would call a program-controlled computer.

Third, Turing's construction presents an original contribution to the technique of programming in the form of skeleton tables, later known as macros, which seems to not have been appreciated before.

Fourth, Turing's original description presents difficulties to readers of a later generation, for various reasons. It describes the universal machine, not as an end in itself, but as a construction required in establishing a strictly defined concept of computable numbers for use in proofs of theorems of mathematical logic. In this way Turing's contributions to the ideas of computing become less accessible. Moreover, both the terminology and the machine organization employed by Turing are rather different from what has later become the norm. As a further difficulty, Turing's description has several errors that may deter the reader's understanding.

With these circumstances in mind the present Appendix presents an extract of Turing's original text [6], concentrating on the construction of the universal machine, corrected for known errors, and augmented with footnotes aiming at relating Turing's idea to those of a later generation and at amplifying his argument. References to such footnotes are given as superscript numbers. The beginning of each page of the original edition is indicated by the page number within curly brackets, and in addition any point in the text to which reference is made from the main part of the present paper is indicated as follows: {original page number, millimeters measured downwards on the page}.

The Universal Machine, by A. M. Turing [6]

1. Computing machines (231)

We may compare a man in the process of computing a real number to a machine which is only capable of a finite number of conditions $q_1, q_2, \ldots, q_k$ which will be called 'm-configurations'. The machine is supplied with a 'tape' (the analogue of paper) running through it and divided into sections (called 'squares') each capable of bearing a 'symbol' $(231, 93)$. At any moment there is just one square, say the $r$th, bearing the symbol $S(r)$ which is 'in the machine'. We may call this square the 'scanned square'. The symbol on the scanned square may be called the 'scanned symbol'. The 'scanned symbol' is only one of which the machine is, so to speak, 'directly aware'. However, by altering its $m$-configuration the machine can effectively remember some of the symbols which it has 'seen' (scanned) previously. The possible behaviour of the machine at any moment is determined by the $m$-configuration $q_m$ and the scanned symbol $S(r)$. This pair $q_m, S(r)$ will be called the 'configuration': thus the configuration determines the possible behaviour of the machine $(231, 148)$. In some of the configurations in which the scanned symbol is blank (i.e. bears no symbol) the machine writes down a new symbol on the scanned square: in other configurations it erases the scanned symbol. The machine may also change the square which is being scanned, but only by shifting it one place to right or left. In addition to any of these operations the $m$-configuration may be changed. Some of the symbols written down $(232)$ will form the sequence of figures which is the decimal of the real number which is being computed. The others are just rough notes to 'assist the memory'. It will only be these rough notes which will be liable to erasure.

It is my contention that these operations include all those which are used in the computation of a number...

2. Definitions

Automatic machines. If at each stage the motion of a machine (in the sense of § 1) is completely determined by the configuration, we shall call the machine an 'automatic machine' (or $a$-machine). ... In this paper I deal only with automatic machines, and will therefore omit the prefix $a$.

Computing machines. If an $a$-machine prints two kinds of symbols, of which the first kind (called figures) consists entirely of 0 and 1 (the others being called symbols of the second kind), then the machine will be called a computing

---

1. The tape is assumed to extend infinitely in one direction from a definite starting point.
machine (232, 130). If the machine is supplied with a blank tape and set in motion, starting from the correct initial m-configuration, the subsequence of the symbols printed by it which are of the first kind will be called the sequence computed by the machine. The real number whose expression as a binary decimal is obtained by prefacing this sequence by a decimal point is called the number computed by the machine.

At any stage of the motion of the machine, the number of the scanned square, the complete sequence of all symbols on the tape, and the m-configuration will be said to describe the complete configuration at that stage. The changes of the machine and tape between successive complete configurations will be called the moves of the machine.

{233} ...

3. Examples of computing machines

I. A machine can be constructed to compute the sequence 010101 ... . The machine is to have the four m-configurations 'b', 'c', 't', 'e' and is capable of printing '0' and '1'. The behaviour of the machine is described in the following table in which 'R' means 'the machine moves so that it scans the square immediately on the right of the one it was scanning previously'. Similarly for 'L'. 'E' means 'the scanned symbol is erased' and 'P' stands for 'prints'. This table (and all succeeding tables of the same kind) is to be understood to mean that for a configuration described in the first two columns the operations in the third column are carried out successively, and the machine then goes over into the m-configuration described in the last column. When the second column is left blank, it is understood that the behaviour of the third and fourth columns applies for any symbol and for no symbol.  

The machine starts in the m-configuration b with a blank tape.

<table>
<thead>
<tr>
<th>m-configuration</th>
<th>symbol</th>
<th>operations</th>
<th>final m-configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>None</td>
<td>P0, R</td>
<td>c</td>
</tr>
<tr>
<td>c</td>
<td>None</td>
<td>R</td>
<td>e</td>
</tr>
<tr>
<td>e</td>
<td>None</td>
<td>P1, R</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>None</td>
<td>R</td>
<td>b</td>
</tr>
</tbody>
</table>

{234} If (contrary to the description in § 1) we allow the letters L, R to appear more than once in the operations column we can simplify the table considerably.

<table>
<thead>
<tr>
<th>m-configuration</th>
<th>symbol</th>
<th>operations</th>
<th>final m-configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>None</td>
<td>P0</td>
<td>b</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>R, R, P1</td>
<td>b</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>R, R, P0</td>
<td>b</td>
</tr>
</tbody>
</table>

II. As a slightly more difficult example we can construct a machine to compute the sequence 0010110110111101111111111111111111111 ... . The machine is to be capable of five m-configurations, viz. 'o', 'q', 'p', 't', 'b' and of printing '@', 'x', '0', '1'. The first three symbols on the tape will be '@@0'; the other figures follow on alternate squares. On the intermediate squares we never print anything but 'x'. These letters serve to 'keep the place' for us and are erased when we have finished with them. We also arrange that in the sequence of figures on alternate squares there shall be no blanks.

<table>
<thead>
<tr>
<th>m-configuration</th>
<th>symbol</th>
<th>operations</th>
<th>final m-configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>@</td>
<td>P @, R, P @, R, P 0, R, R, P 0, L, L</td>
<td>o</td>
</tr>
<tr>
<td>o</td>
<td>0</td>
<td>R, R, P1, L, L</td>
<td>q</td>
</tr>
<tr>
<td>q</td>
<td>Any (0 or 1)</td>
<td>R, R</td>
<td>q</td>
</tr>
<tr>
<td>p</td>
<td>@</td>
<td>P 1, L</td>
<td>p</td>
</tr>
<tr>
<td>p</td>
<td>None</td>
<td>R</td>
<td>p</td>
</tr>
<tr>
<td>f</td>
<td>Any</td>
<td>R, R</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>None</td>
<td>P 0, L, L</td>
<td>o</td>
</tr>
</tbody>
</table>

2 As a symbol on a square of the tape, 'blank' and 'no symbol' are the same. In the table of a machine, 'None' in the symbol column indicates a configuration having 'blank' as the scanned symbol. Other conventions related to the symbol column can be derived from Turing's practice and expressed as rules of abbreviation as follows:

- Symbol column: The instruction line stands for the totality of lines obtained by replacing the symbol column by:
  - (empty) None and every symbol
  - Any Every symbol but not None
  - not x None and every symbol except x

---

THE COMPUTER JOURNAL, Vol. 36, No. 4, 1993
To illustrate the working of this machine a table is given below of the first few complete configurations. These complete configurations are described by writing down the sequence of symbols which are on the tape, (235) with the m-configuration written below the scanned symbol. The successive complete configurations are separated by colons.

\[
\begin{array}{cccccccc}
@ & \alpha & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
@ & \alpha & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
\end{array}
\]

This table could also be written in the form

\[
b : @ @ 0 0 0 @ @ 0 0 0 0 0 \ldots \ldots \quad \text{(235, 78)} \quad \text{(G)}
\]

in which a space has been made on the left of the scanned symbol and the m-configuration written in this space. This form is less easy to follow, but we shall make use of it later for theoretical purposes.

The convention of writing the figures only on alternate squares is very useful; I shall always make use of it (235, 108). I shall call the one sequence of alternate squares F-squares and the other sequence E-squares. The symbols of the E-squares will be liable to erasure. The symbols on F-squares form a continuous sequence. There are no blanks until the end is reached. There is no need to have more than one E-square between each pair of F-squares: an apparent need of more E-squares can be satisfied by having a sufficiently rich variety of symbols capable of being printed on E-squares. If a symbol β is on an F-square S and a symbol α is on the E-square next on the right of S, then S and β will be said to be marked with α. The process of printing this α will be called marking β (or S) with α.\(^4\)

4. Abbreviated tables

There are certain types of process used by nearly all machines, and these, in some machines, are used in many connections. These processes include copying down sequences of symbols, comparing sequences, erasing all symbols of a given form, etc. Where such processes are concerned we can abbreviate the tables for the m-configurations considerably by the use of 'skeleton tables'. In skeleton tables there appear capital German letters and small Greek letters. These are of the nature of 'variables'. By replacing each capital German letter throughout by an m-configuration (236) and each small Greek letter by a symbol, we obtain the table for an m-configuration.

The skeleton tables are to be regarded as nothing but abbreviations: they are not essential. So long as the reader

\(^3\) Replacing Turing's inverted 'ε' everywhere by 'α', the original has:

\[
b : @ @ 0 0 0 \ldots \ldots \quad \text{(C)}
\]

The erroneous omission of the blank following b is found similarly later. In order to prevent confusion with other uses of 'C', Turing's uses of references to (C) are changed to (G) here and later.

\(^4\) The machine of II and the general conventions for the use of the tape can be described by means of the following example of a picture of the tape of the machine:

<table>
<thead>
<tr>
<th>Numbers of 'l's:</th>
<th>m</th>
<th>m+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>p None</td>
<td>\ldots \ldots p</td>
<td>p \ldots \ldots p</td>
</tr>
</tbody>
</table>

Only provided \(r=0\) \(r\) letters α, \(0 \leq r \leq m\)

This form of picture is a general snapshot of the contents of the tape, valid whenever the machine is in configuration p, None, as indicated to the left. The symbols written on the squares of the tape are shown along the horizontal line, with the F-squares above and the E-squares below the line, the mark on a symbol on an F-square shown directly below it. The relevant m-configuration is shown above the scanned symbol if this is on an F-square, below it, as in the present picture, if it is on an E-square. The picture may also show several positions of the m-configuration, and may indicate the direction of transitions induced by the carrying out of the instructions of the machine by means of arrows. In the present example the arrows indicate the transitions induced by carrying out the instruction 'p None L, L p'. Additional generally valid descriptions are shown, separately for the symbols on the F-squares and E-squares.

The picture displays the general conventions for the use of the tape adopted by Turing. The tape is used from a left end marked by the pair @@, to a right end distinguished by having a blank on an F-square. This blank will never be marked, hence the right limit of the current use of the tape can be detected by the presence of two consecutive blank squares. Writing on an F-square is only done on the first blank square at the right end.

The present picture may be taken to be the hypothesis of a proof of Turing's example II by induction. From the picture it may readily be derived that if \(r=0\) there are \(m+1\) figures 'l' in the rightmost sequence. When this situation has been reached the machine will continue to make the left moves until the configuration p, α is reached. This will bring the machine into m-configuration \(\alpha\) and subsequently, via \(\alpha\) and \(\beta\), back to \(p\). At this stage the situation described by the present picture applies once more, the value of \(m\) having been increased by 1 and \(r=m\). Thus the validity of the induction hypothesis and of Turing's machine II is established.
understands how to obtain the complete tables from the skeleton tables, there is no need to give any exact definitions in this connection.\(^5\)

Let us consider an example:\(^6\)

\[
\begin{array}{|c|c|c|c|}
\hline
m\text{-configuration} & Symbol & Behaviour & Final m\text{-configuration} \\
\hline
\def\mymathcal{\mathcal{C}} \at \def\mymathcal{\mathcal{B}} \at \def\mymathcal{\mathcal{Z}} & \at & L & \def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \ \text{From the } m\text{-configuration} \\
\not\at & L & \def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \text{ the machine finds the symbol of the form } \at \\
\hline
\def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) & \not \def\mymathcal{\mathcal{Z}} \text { nor None} & R & \def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \\
None & R & \def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \text{ if there is no } \def\mymathcal{\mathcal{Z}} \text{ then the } m\text{-configuration becomes } \def\mymathcal{\mathcal{B}}. \\
\hline
\def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) & \not \def\mymathcal{\mathcal{Z}} \text { nor None} & R & \def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \\
None & R & \def\mymathcal{\mathcal{C}} (\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \\
\hline
\end{array}
\]

If we were to replace \(\def\mymathcal{\mathcal{C}}\) throughout by \(q\) (say), \(\def\mymathcal{\mathcal{B}}\) by \(\tau\), and \(\def\mymathcal{\mathcal{Z}}\) by \(x\), we should have a complete table for the \(m\text{-configuration}\) \(f(q, \tau, x)\). \(f\) is called an '\(m\text{-function}'\).

The only expressions which are admissible for substitution in an \(m\text{-function}\) are the \(m\text{-configurations}\) and symbols of the machine. These have to be enumerated more or less explicitly; they may include expressions such as \(\def\mymathcal{\mathcal{E}}(e, x)\); indeed they must if there are any \(m\text{-functions}\) used at all. If we did not insist on this explicit enumeration, but simply stated that the machine had certain \(m\text{-configurations}\) (enumerated) and all \(m\text{-configurations}\) obtainable by substitution of \(m\text{-configurations}\) in certain \(m\text{-functions}\), we should usually get an infinity of \(m\text{-configurations}\); e.g. we might say that the machine was to have the \(m\text{-configuration}\) \(q\) and all \(m\text{-configurations}\) obtainable by substituting an \(m\text{-configuration}\) \(\def\mymathcal{\mathcal{C}}\) in \(\def\mymathcal{\mathcal{P}}(\def\mymathcal{\mathcal{C}})\). Then it would have \(q, \def\mymathcal{\mathcal{P}}(q), \def\mymathcal{\mathcal{P}}(\def\mymathcal{\mathcal{P}}(q)), \ldots\) as \(m\text{-configurations}\).

Our interpretation rule then is this. We are given the names of the \(m\text{-configurations}\) of the machine, mostly expressed in terms of \(m\text{-functions}\). We are also given skeleton tables. All we want is the complete table for the \(m\text{-configurations}\) of the machine. This is obtained by repeated substitution in the skeleton tables.

\[\text{(237) Further examples. (In the explanations the symbol } '\rightarrow' \text{ is used to signify 'the machine goes into the } m\text{-configuration ...')}
\]

\[
e(\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \at f(e, \def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}) \ \text{the first } \def\mymathcal{\mathcal{Z}} \text{ is erased and } \rightarrow \def\mymathcal{\mathcal{C}}. \ \text{If there is no } \def\mymathcal{\mathcal{Z}} \rightarrow \def\mymathcal{\mathcal{B}}.
\]

\[
e(\def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}) \at e(\def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}) \ \text{From } \def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}} \text{ all letters } \def\mymathcal{\mathcal{Z}} \text{ are erased and } \rightarrow \def\mymathcal{\mathcal{B}}.
\]

The last example seems somewhat more difficult to interpret than most. Let us suppose that in the list of \(m\text{-configurations}\) of some machine there appears \(e(b, x) = q\), say). The table is

\[
e(b, x) \at e(\def\mymathcal{\mathcal{B}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}, \def\mymathcal{\mathcal{Z}}).
\]

or

\[
q \at e(q, b, x).
\]

Or, in greater detail:

\[
e(q, b, x) \at e(q, b, x) \at e(q, b, x) \at e(q, b, x) \at e(q, b, x)
\]

In this we could replace \(e(q, b, x)\) by \(q\)’ and then give the table for \(f\) (with the right substitutions) and eventually reach a table in which no \(m\text{-functions}\) appeared.

\[
\begin{array}{|c|c|c|c|}
\hline
\text{pe}(\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}) & \text{Any} & R, R & \text{pe}(\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}, \@) \\
\text{pe}_1(\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}) & \text{None} & P, \beta & \text{pe}_1(\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}) \ \text{From } \text{pe}(\def\mymathcal{\mathcal{C}}, \def\mymathcal{\mathcal{B}}) \text{ the machine prints } \beta \text{ at the end of the sequence of symbols and } \rightarrow \def\mymathcal{\mathcal{C}}. \\
\hline
l(\def\mymathcal{\mathcal{C}}) & L & \def\mymathcal{\mathcal{C}} \\
\tau(\def\mymathcal{\mathcal{C}}) & R & \def\mymathcal{\mathcal{C}} \\
\hline
\end{array}
\]

\(^5\)The skeleton tables are the same kinds of construction as have later been called macros. For Turing's explanations of the skeleton tables to be valid, it is generally necessary to assume that when an \(m\text{-function}\) is applied the scanned square lies within an active section of the tape, to the right of the \(\@\@\) pair and to the left of the first blank on an F-square.

\(^6\)In the original the clauses 'nor None' in \(f\), and \(f\) are missing.
\[ f'(\mathcal{E}, \mathcal{B}, \alpha) \]
\[ f'(\mathcal{E}, \mathcal{B}, \beta) \]
\[ \text{From } f'(\mathcal{E}, \mathcal{B}, \alpha) \text{ it does the same as for } f(\mathcal{E}, \mathcal{B}, \alpha) \text{ but moves to the left before } \to \mathcal{C}. \]

\[ c(\mathcal{E}, \mathcal{B}, \alpha) \]
\[ c_1(\mathcal{E}) \]
\[ \beta \]
\[ f'(c_1(\mathcal{E}), \mathcal{B}, \alpha) \]
\[ c(\mathcal{E}, \mathcal{B}, \beta). \text{ The machine writes at the end the first symbol marked } \alpha \text{ and } \to \mathcal{C}. \text{ If there is no } \alpha \to \mathcal{B} \text{ (237, 194).} \]

(238) The last line stands for the totality of lines obtainable from it by replacing \( \beta \) by any symbol which may occur on the tape of the machine concerned.

\[ c(e(\mathcal{E}, \mathcal{B}, \alpha), \mathcal{B}, \alpha) \]
\[ c(e(\mathcal{E}, \mathcal{B}, \alpha), \mathcal{B}, \beta) \]
\[ c(\mathcal{E}, \mathcal{B}, \alpha). \text{ The machine copies down in order at the end all symbols marked } \alpha \text{ and erases the letters } \alpha; \to \mathcal{B}. \]

\[ r(e(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ r(e(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ E, P \beta \]
\[ f(r(e(\mathcal{E}, \mathcal{B}, \alpha, \beta), \alpha, \beta)) \]
\[ r(e(\mathcal{E}, \mathcal{B}, \alpha, \beta), \alpha, \beta). \text{ The machine replaces the first } \alpha \text{ by } \beta \text{ and } \to \mathcal{E}, \to \mathcal{B} \text{ if there is no } \alpha \text{ (238, 58).} \]

\[ r(e(\mathcal{B}, \alpha, \beta)) \]
\[ r(e(\mathcal{B}, \alpha, \beta), \alpha, \beta) \]
\[ r(e(\mathcal{B}, \alpha, \beta), \alpha, \beta). \text{ The machine replaces all letters } \alpha \text{ by } \beta; \to \mathcal{B}. \]

\[ c(p(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ c(p(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ \gamma \]
\[ f(c(p(\mathcal{E}, \mathcal{B}, \gamma), \alpha, \beta)) \]
\[ f(c(p(\mathcal{E}, \mathcal{B}, \gamma), \alpha, \beta)) \]
\[ \mathcal{E} \]
\[ \mathcal{A} \]
\[ c(p(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ c(p(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ \gamma \]
\[ c(p(\mathcal{E}, \mathcal{B}, \gamma, \alpha, \beta)) \]
\[ c(p(\mathcal{E}, \mathcal{B}, \gamma, \alpha, \beta)) \]
\[ \mathcal{E} \]
\[ \mathcal{A} \]
\[ \text{The first symbol marked } \alpha \text{ and the first marked } \beta \text{ are compared. If there is neither } \alpha \text{ nor } \beta, \to \mathcal{E}. \text{ Otherwise } \to \mathcal{A}. \]

\[ c(p(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ c(p(\mathcal{E}, \mathcal{B}, \alpha, \beta)) \]
\[ \gamma \]
\[ c(p(\mathcal{E}, \mathcal{B}, \gamma, \alpha, \beta)) \]
\[ c(p(\mathcal{E}, \mathcal{B}, \gamma, \alpha, \beta)) \]
\[ \mathcal{E} \]
\[ \mathcal{A} \]
\[ \text{The sequence of symbols marked } \alpha \text{ is compared with the sequence marked } \beta. \to \mathcal{E} \text{ if they are similar. Otherwise } \to \mathcal{A}. \text{ Some of the symbols } \alpha \text{ and } \beta \text{ are erased.} \]

\[ 239 \]
\[ q(\mathcal{E}) \]
\[ \{ \text{Any} \}
\[ R \]
\[ q(\mathcal{E}) \]
\[ \{ \text{None} \}
\[ R \]
\[ q(\mathcal{E}) \]
\[ q(\mathcal{E}, \alpha) \]
\[ q_1(\mathcal{E}) \]
\[ q(\mathcal{E}, \alpha) \]
\[ q_1(\mathcal{E}) \]
\[ \alpha \]
\[ \{ \text{not } \alpha \}
\[ L \]
\[ q_1(\mathcal{E}, \alpha) \]
\[ \text{The machine finds the last symbol of form } \alpha \text{ to } \to \mathcal{E}. \]

\[ p(\mathcal{E}, \alpha, \beta) \]
\[ p(\mathcal{E}, \alpha, \beta) \]
\[ \text{The machine prints } \alpha \beta \text{ at the end.} \]

\[ c(\mathcal{E}, \mathcal{B}, \alpha, \beta) \]
\[ c(\mathcal{E}, \mathcal{B}, \alpha, \beta) \]
\[ \text{The machine copies down at the end first the symbols marked } \alpha, \text{ then those marked } \beta, \text{ and finally those marked } \gamma; \text{ it erases the symbols } \alpha, \beta, \gamma. \]

---

7 The last clause has been added to the original.
8 The last comma has been added to the original.
9 The original has \( f'(c_1(\mathcal{E}, \ldots) \). A definition of \( c(\mathcal{B}, \alpha) \), which is not used elsewhere, has been omitted.
5. Enumeration of computable sequences

A computable sequence $\gamma$ is determined by a description of a machine which computes $\gamma$. Thus the sequence 001010101101111 ... is determined by the table on p. (234), and, in fact, any computable sequence is capable of being described in terms of such a table.

It will be useful to put these tables into a kind of standard form. In the first place let us suppose that the table is given in the same form as the first table, for example, I on p. (233). That is to say, that the entry in the operations column is always one of the forms $E:E,R; E,L; P_x: P_x R; P_x L; R:L$; or no entry at all. The table can always be put into this form by introducing more $m$-configurations. Now let us give numbers to the $m$-configurations, calling them $q_1, ..., q_r$, as in §1. The initial $m$-configuration is always to be called $q_1$. We also give numbers to the symbols $S_0, ..., S_m$ (240) and, in particular, blank $= S_0$, $0 = S_1$, $1 = S_2$. The lines of the tables are now of the form

<table>
<thead>
<tr>
<th>$m$-configuration</th>
<th>Symbol</th>
<th>Operations</th>
<th>Final-$m$ configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_i$</td>
<td>$S_j$</td>
<td>$PS_{j-k} L$</td>
<td>$q_m$ ($N_1$)</td>
</tr>
<tr>
<td>$q_i$</td>
<td>$S_j$</td>
<td>$PS_{j-k} R$</td>
<td>$q_m$ ($N_2$)</td>
</tr>
<tr>
<td>$q_i$</td>
<td>$S_j$</td>
<td>$PS_k$</td>
<td>$q_m$ ($N_3$)</td>
</tr>
</tbody>
</table>

Lines such as

$q_i$ $S_j$ $E, R$ $q_m$

are to be written as

and lines such as

$q_i$ $S_j$ $R$ $q_m$

to be written as

$q_i$ $S_j$ $PS_{j-k}$ $R$ $q_m$

In this way we reduce each line of the table to a line of one of the forms $(N_1)$, $(N_2)$, $(N_3)$.

From each line of form $(N_1)$ let us form an expression $q_i S_j S_k R q_m$; from each line of form $(N_2)$ we form an expression $q_i S_j S_k R q_m$; and from each line of form $(N_3)$ we form an expression $q_i S_j S_k R q_m$.

Let us write down all expressions so formed from the table for the machine and precede each of them by a semi-colon (240, 140). In this way we obtain a complete description of the machine. In this description we shall replace $q_i$ by the letter 'D' followed by the letter 'A' repeated $i$ times, and $S_j$ by 'D' followed by 'C' repeated $j$ times. This new description of the machine may be called the "standard description" (S.D.). It is made up entirely from the letters 'A', 'C', 'D', 'L', 'R', 'N' and from 'r'.

6. The universal computing machine (241)

It is possible to invent a single machine which can be used to compute any computable sequence (241, 192). If this machine $U$ is supplied with a tape on the beginning of which is written the S.D. of some computing machine $M$, (242) then $U$ will compute the same sequence as $M$. In this section I explain in outline the behaviour of the machine. The next section is devoted to giving the complete table for $U$.

Let us first suppose that we have a machine $M$ which will write down on the $F$-squares the successive complete configurations of $M$. These might be expressed in the same form as on p. (235), using the second description, (G), with all symbols on one line. Or, better, we could transform this description (as in §5) by replacing each $m$-configuration by 'D' followed by 'A' repeated the appropriate number of times, and by replacing each symbol by 'D' followed by 'C' repeated the appropriate number of times. The numbers of letters 'A' and 'C' are to agree with the numbers chosen in §5, so that, in particular, 'O' is replaced by 'DCC', '1' by 'DCCC', and the blanks by 'D'. These substitutions are to be made after the complete configurations have been put together, as in (G). Difficulties arise if we do the substitution first. In each complete configuration the blanks would all have to be replaced by 'D', so that the complete configuration would not be expressed as a finite sequence of symbols.

If in the description of the machine $M$ of §3 we replace '0' by 'DAA', '1' by 'DCCC', 'q' by 'DAAA', then the sequence

\[ce_3(\mathcal{B}, \alpha, \beta, \gamma, \delta, e)\]

\[e_1(\mathcal{C})\]

\[\begin{array}{c}
@ \\
\text{Not } @ \\
\text{Any } \\
\text{None}
\end{array}
\]

\[R, L\]

\[\mathcal{C},\mathcal{E}\]

\[\mathcal{G}\]

\[e_1(\mathcal{C})\]

\[\text{From } e(\mathcal{C}) \text{ the marks are } \\
\text{erased from all marked } \]

\[\text{symbols. } \rightarrow \mathcal{C}. \text{ (239, 95 and 98).}\]

\[ce_3(\mathcal{B}, \delta, e), \alpha, \beta, \gamma)^{10}\]

\[\text{The definition of } ce_3(\mathcal{B}, \alpha, \beta, \gamma, \delta, e) \text{ has been added.}\]

\[\text{In the original the expressions were said to be separated, not preceded, by semi-colons. A following section, from (240, 175) to (241, 165), introducing description numbers, is omitted.}\]
(G) becomes$^{12}$

\[
\text{DAD : DCCDCCCDAACDDC : DCCDCCCDAADCDCC} : \ldots \quad [242, 99] \quad (G_1)
\]

(This is the sequence of symbols on \( F \)-squares.)

It is not difficult to see that if \( M \) can be constructed, then so can \( M' \). The manner of operation of \( M' \) could be made to depend on having the rules of operation (i.e. the S.D) of \( M \) written somewhere within itself (i.e. within \( M' \)); each step could be carried out by referring to these rules. We have only to regard the rules as being capable of being taken out and exchanged for others and we have something very akin to the universal machine.

One thing is lacking: at present the machine \( M' \) prints no figures. We may correct this by printing between each successive pair of complete configurations the figures which appear in the new configuration but not in the old. The \((G_1)\) becomes$^{13}$

\[
\text{DAD : 0 : 0 : DCCDCCCDAACDDC : DCCC} : \ldots \quad [242, 165] \quad (G_2)
\]

It is not altogether obvious that the \( E \)-squares leave enough room for the necessary 'rough work', but this is, in fact, the case.

The sequences of letters between the colons in expressions such as \((G_1)\) may be used as standard descriptions of the complete configurations.$^{14}$

7. Detailed description of the universal machine \((243)\)

A table is given below of the behaviour of this universal machine. The \( m \)-configurations of which the machine is capable are all those occurring in the first and last columns of the table, together with all those which occur when we write out the unabbreviated tables of those which appear in the table in the form of \( m \)-functions, e.g. \( e(\text{anf}) \) appears in the table and is an \( m \)-function. Its unabbreviated table is (see p. \( 239\))

\[
e(\text{anf}) \quad \begin{cases} 
@ & R \\
\text{Not } @ & L \\
\end{cases} \\
e_1(\text{anf}) \quad \begin{cases} 
\text{Any} & R, E, R \\
\text{None} & \text{anf} \\
\end{cases}
\]

Consequently \( e_1(\text{anf}) \) is an \( m \)-configuration of \( M \).

When \( M \) is ready to work the tape running through it bears on it the symbol \( @ \) on an \( F \)-square and again \( @ \) on the next \( E \)-square; after this, on \( F \)-squares only, comes the S.D of the machine followed by a double colon \( : \) (a single symbol, on an \( F \)-square) \((243, 95 \text{ and } 102)\). The S.D consists of a number of instructions, preceded by semi-colons \((243, 113)\).$^{15}$

\[\text{DA : DCCDCCCDAACDDC : DCCDCCCDAADCDCC} : \ldots \quad (C_1)\]

\[\text{DDA : 0 : 0 : DCCDCCCDAACDDC : DCCC} : \ldots \quad (C_2)\]

\[12\text{ The original has } (\text{cf. footnote } 3 \text{ above}): \]

\[\text{DA : DCCDCCCDAACDDC : DCCDCCCDAADCDCC} : \ldots \quad (C_1)\]

\[13\text{ The original has } (\text{cf. footnotes } 3 \text{ and } 13 \text{ above}): \]

\[\text{DDA : 0 : 0 : DCCDCCCDAACDDC : DCCC} : \ldots \quad (C_2)\]

\[14\text{ A paragraph on description numbers is omitted.}\]

\[15\text{ The original has } "\ldots \text{ separated by semi-colons". Initially the } m \text{-configuration is } b \text{ with the universal machine scanning a square to the right of the } \@ \text{ pair.}\]
Each instruction consists of five consecutive parts\textsuperscript{16}

(i) 'D' followed by a sequence of letters 'A'. This describes the relevant \( m \)-configuration.

(ii) 'D' followed by a sequence of letters 'C'. This describes the scanned symbol.

(iii) 'D' followed by another sequence of letters 'C'. This describes the symbol into which the scanned symbol is to be changed.

(iv) 'L', 'R', or 'N', describing whether the machine is to move to left, right, or not at all.

(v) 'D' followed by a sequence of letters 'A'. This describes the final \( m \)-configuration.

The machine \( \mathcal{U} \) is to be capable of printing 'A', 'C', 'D', 'O', 'I', 'u', 'v', 'w', 'x', 'y', 'z'.\textsuperscript{17} \cite{243, 190}. The S.D. is formed from 'I', 'A', 'C', 'D', 'L', 'R', 'N', and is followed by \('.\textsuperscript{17} \)

**Subsidiary skeleton table** \cite{244}

\[
\begin{align*}
\text{con}(\mathcal{E}, x) & \quad \begin{cases}
\text{Not } A & R, R & \text{con}(\mathcal{E}, x) \\
A & L, P \ x, R & \text{con}_1(\mathcal{E}, x) \\
\text{Not } C & R, R & \text{con}_2(\mathcal{E}, x)
\end{cases} \\
\text{con}_1(\mathcal{E}, x) & \quad \begin{cases}
A & R, P, x, R & \text{con}_1(\mathcal{E}, x) \\
D & R, P, x, R & \text{con}_2(\mathcal{E}, x)
\end{cases} \\
\text{con}_2(\mathcal{E}, x) & \quad \begin{cases}
C & R, P, x, R & \text{con}_2(\mathcal{E}, x) \\
\text{Not } C & R, R & \mathcal{E}
\end{cases}
\end{align*}
\]

\text{Starting from an } \mathcal{F} \text{-square, } \mathcal{S} \text{ say, the sequence } Q \text{ of symbols describing a configuration closest on the right of } \mathcal{S} \text{ is marked out with letters } x. \rightarrow \mathcal{E}. \text{ In the final configuration the machine is scanning the square which is four squares to the right of the last square of } Q. \text{ The machine prints } \text{DAD on the } \mathcal{F} \text{-squares after } \rightarrow \text{anf} \text{.} \text{ The machine marks the configuration in the last complete configuration with } y. \rightarrow \text{tom}.

The table for \( \mathcal{U} \)

\[
\begin{array}{c|c}
\text{b} & \text{b} = \begin{cases}
(b_1, b_1, \ldots) & \text{b. The machine prints } \text{DAD on the } \mathcal{F} \text{-squares after } \rightarrow \text{anf} \\
\text{anf} & \text{the } \mathcal{F} \text{-squares after } \rightarrow \text{anf} \\
\text{anf}_1 & \text{anf} \text{ } \rightarrow \text{tom, } y
\end{cases} \\
\end{array}
\]

\text{The initial state of the universal machine may be represented by the following picture of the tape:}

\[
\begin{array}{cccccccccc}
\text{Standard description, S.D.} & \text{Instruction} & \text{Scanned} & \text{Changed} & \text{Move } \mathcal{E} (L, R, N) & \text{m-conf.} & \text{symbol} & \text{symbol} & \text{Final m-conf.} \\hline
\text{b} & 0 & 1 & A & C & D & \text{DA} & \text{DC} & \text{CC} & \text{CMDA}
\end{array}
\]

\text{For the tape of the universal machine it holds invariably:}

11. On the \( \mathcal{F} \)-squares only the following symbols will be written:

\[
\begin{array}{cccccccccc}
\text{...} & 0 & 1 & A & C & D & \text{DA} & \text{DC} & \text{CC} & \text{CMDA}
\end{array}
\]

and only in the contexts described below.

12. \( A, C \) and \( D \), are only used as follows:

\[
\begin{array}{cccccccccc}
\text{Description part} & \text{Use of } A, C, D & \text{Corresponding machine symbol} \\hline
\text{m-configuration} & \text{DA, initial configuration} & q_1 \\
\text{DA } \ldots A (i \text{ letters } A) & \text{DA } \ldots A (i \text{ letters } A) & q_i \\
\text{Symbol} & \text{D} & \text{blank } = S_0 \\
& \text{DC} & 0 = S_1 \\
& \text{DC} & 1 = S_2 \\
& \text{DC } \ldots C (i \text{ letters } C) & S_j
\end{array}
\]

13. Writing on \( \mathcal{F} \)-squares is only done on the next blank to the right.

14. The symbols on the \( \mathcal{F} \)-squares to the right of \( \ldots \) describe the configurations generated by the machine described to the left of \( \ldots \), in sequence from left to right, and in addition include the figures \( \mathcal{O} \) and \( \mathcal{I} \) that have been printed by that machine. Each complete configuration and figure is preceded by \( \ldots \).

15. The symbols used as marks on the \( E \)-squares are:

\[
\begin{array}{cccccccccc}
\text{a, w, x, y, z}
\end{array}
\]

\text{The original omits } \mathcal{O} \text{ and } \mathcal{I}.

\text{The original has } C \text{ instead of } Q \text{ and } \text{con}(\mathcal{E}, y) \text{ appears before the sentence 'In the final ...'.}

\text{The original lacks the final operations } R, R, P \text{ and says that 'DAD' is printed. Thus the initial complete configuration lacks the representation of the blank on the scanned square, cf. footnotes 12 and 13.}

\text{The original has } (\text{anf}_1, y) \text{ referring to an undefined } m \text{-function.}

\textbf{The Computer Journal, Vol. 36, No. 4, 1993}
\[ \begin{align*} 
\text{fom} & \quad \{ 
\begin{array}{l}
z \\
\text{not } z \text{ nor } @ \\
@ 
\end{array} 
\} \\
R, P_z, L & \quad \text{con}(\text{imp}, x) \\
L, L & \quad \text{fom} \\
\text{stop} & \quad \text{fom} \\
\text{imp} & \quad \{ 
\begin{array}{l}
\text{cpe}(e(\text{anf}, x), y), \text{sim}, x, y) 
\end{array} 
\} 
\end{align*} \]

\text{anf}. Taking the long view, the last instruction relevant to the last configuration is found. It can be recognized afterwards as the instruction following the first semi-colon marked \( z \) (244, 192). \(^{23} \rightarrow \text{sim} \)

\(^{21}\) The original has no special treatment of \( @ \), causing the universal machine to scan squares indefinitely to the left, beyond the \( @ @ \) pair, when no instruction applies.

\(^{22}\) The original gives the instruction as follows:

\[ \text{imp} \quad \text{cpe}(\text{fom}, x, y), \text{sim}, x, y) \]

This makes use of an undefined \( m \)-function, \( e(\text{fom}, x, y) \), and erroneously continues at \( \text{fom} \), which would fail to reset the \( y \)-marks which have been partly erased by the \( \text{cpe} \) function. At this point the state of the tape is:

\[ \begin{align*} 
\text{Standard description, S.D.} \\
\text{Instruction} \\
\text{mov} -\text{conf} & \quad \text{Scanned} \quad \text{Changed} \quad \text{Move} \quad \text{m} \quad (L, R, N) \\
\text{guration} & \quad \text{symbol} \quad \text{symbol} \quad \downarrow \quad \text{Final } m \quad -\text{conf}. \\
\text{kmp} & \quad x \\
\text{First } z \text{-marked semi-colon} & \quad \text{No matching configuration has been found} \\
\text{Complete configurations and printed figures, of form } (G_2) & \\
\text{Last complete configuration} \\
\text{Symbols} & \quad \text{mov} -\text{conf} \quad \text{Scanned} \quad \text{Symbols} \\
\text{or empty} & \quad \text{guration} \quad \text{symbol} \quad \text{or empty} \\
\text{\text{:DAD} } & \quad \text{DC} \quad \text{DCA} \quad \text{DAD} \quad \text{DC} \quad \text{DC} \quad \text{DC} \\
\text{Last colon} \\
\text{In relation to the local loop this general snapshot displays the induction hypothesis of the process for finding the instruction having an } m \text{-configuration and a scanned symbol matching those of the last complete configuration. The loop has been applied to all instructions to the right of all } z \text{-marked semi-colons, except the first one, no match having been found. In the present cycle of the loop the instruction to the right of the first } z \text{-marked semi-colon is tried.} \\
\text{In relation to the overall loop of the universal machine the induction hypothesis is that the series of complete configurations and printed figures shown in the picture, taken from left to right up to and including the last one, describe the results of the series of computational steps carried out by the machine given in the standard description up to a certain stage.} \\
\text{The original has } \text{the last semi-colon}. \\
\end{align*} \]
sim. The machine marks out the instructions. That part of the instructions which refers to operations to be carried out is marked with $u$, and the final m-configuration with $y$ \{245, 31\}.\textsuperscript{24}

The letters $z$ are erased.

$m_1\{245, 52\}$.\textsuperscript{25} The last complete configuration is marked out into four sections. The configuration is left unmarked. The symbol directly preceding it is marked with $x$. The remainder of the complete configuration is divided into two parts, of which the first part is marked with $v$ and the last with $w$. A colon is printed after the whole. $\rightarrow s_B$.

$s_B$.\textsuperscript{26} The instructions (marked $w$) are examined \{245, 140\}. It is found that they involve 'Print 0' or 'Print 1', then 0: or 1: is printed at the end.

\[\begin{array}{ll}
\text{Standard description, S.D.} & \\
\hline
\text{Instruction} & \text{Scanned} \quad \text{Changed} \quad \text{Move} \quad \text{Final m-conf.} \\
\hline
\text{duration} & \text{symbol} \quad \text{symbol} \quad \text{symbol} \\
\hline
\text{inst} & \text{DA} \quad \text{DCC} \quad \text{CDC} \quad \text{CMDA} \quad \text{M} \quad \text{N} \\
\hline
\text{Symbols} & \text{written} \\
\hline
\text{Written after last colon.} & \text{DDCC} \quad \text{blank 0} \rightarrow 0: \\
\text{depending on scanned} & \text{DDCC} \quad \text{blank 1} \rightarrow 1: \\
\text{and changed symbols:} & \text{Otherwise} \rightarrow \text{empty} \\
\hline
\text{Complete configurations and printed figures, of form (G2).} & \\
\hline
\text{Last complete configuration} & \\
\hline
\text{Symbols} & \text{m-conf.-} \quad \text{Scanned} \quad \text{Symbols} \\
\text{or empty} & \text{duration} \quad \text{symbol} \quad \text{symbol} \quad \text{or empty} \\
\hline
\text{DAD:} & \text{CC} \quad \text{CDC} \quad \text{CMDA} \quad \text{C} \quad \text{V} \quad \text{V} \\
\text{The symbol preceding the scanned one} & \\
\end{array}\]

As indicated by this picture, Turing's instructions correctly make the printing of the figures depend on the scanned symbol being blank. The case where a figure '0' or '1' is written on itself, leaving it unchanged, must not give rise to printing.
Turing's Contribution

To the question of the origin of the concept of a computing device controlled by a program held in its internal store, it is clear that, regarded as a matter of principle, Turing's universal machine displays this concept in a fully developed form. Thus the credit for this idea must be granted to Turing.

On the other hand, as said by M. V. Wilkes [1], the stored program concept cannot properly be regarded just as a matter of principle. Rather, it is 10% a matter of principle and 90% a matter of difficult engineering. On such grounds the credit for the invention must be shared between Turing and the inventors who solved the engineering problems around 1945.

Quite apart from the stored program concept, Turing's work makes an original and significant contribution to the technique of programming, through the introduction and use of the skeleton tables. By his virtuoso handling of this technique, Turing manages to build up the complete formulation of the universal machine from the extremely primitive facilities of Turing machines, in terms of merely three pages of definitions of auxiliary m-functions and a two-page program. Thus Turing demonstrates the significance of his techniques by using them to make a lucid construction of a formal device never before seen or even just conceived of, the universal machine.

---

27 In the original the first line is misprinted as 'q((inst1), u)', the m-function \( c_3 \) is undefined, and the line for \( \text{inst}_2(0) \) has \( c_3 \) instead of \( c_2 \). The original further fails to print the necessary additional blank to the right on the tape.