Computers and Commerce: 1

by A. S. Douglas

Summary: This series of four articles covers in outline the application of computers to business operations, including accounting, stock recording, production control and management procedures. In the first article below previous experience on scientific work is analysed and the conclusions are discussed in a commercial context. It is suggested that a radical approach to deciding the field of computer application is essential and that work should be allocated between programmers on the basis of "one man, one job." The use of autocodes and subroutines in business applications is discussed and the difficulties of introducing these techniques are outlined.

INTRODUCTION

For the purpose of these articles I shall divide the applications of computers in commercial undertakings into four sections. These are (1) Scientific and engineering work, (2) Accounting, statistics and sales analyses, (3) Stock recording and control, and (4) Production and movement control. Of these, the principal applications hitherto have been, of course, to scientific and engineering problems. Whilst these are of only indirect concern to a large proportion of commercial firms, nevertheless I shall begin by outlining the methods employed in the solution of such problems, in the belief that there is something of fundamental value to all users to be gained by a study of these methods.

The procedure for solving any given scientific problem can be briefly stated in terms of the following sequence of operations:

(*a*) define the problem qualitatively,
(*b*) reduce the problem to mathematical notation,
(*c*) select the numerical method of solution, carrying out any necessary mathematical manipulation; and
(*d*) draw up the computer program, including (i) the specification of techniques for enumeration where necessary, (ii) the internal organization of the storage, input and output, and (iii) the external organization of data preparation and the subsequent handling of results.

It is important to appreciate that these stages are strongly interdependent. The way in which the problem is formulated mathematically and the amount of manipulation of equations carried out affects fundamentally the choice of numerical method, and vice-versa. Furthermore the numerical methods suitable for a particular computer must be selected with the construction of that computer in mind, since the choice made may be directly influenced by speed and storage considerations. Finally, the presentation selected for results obviously affects all preceding choices. Thus at stage (*b*) a knowledge of the difficulties likely to be encountered in later stages is essential, whilst stage (*c*) cannot be carried out efficiently unless both the requirements of stage (*b*) and also the outlines of stage (*d*) are foreseen. Stages (*d*) (i) and (ii) can, in some circumstances, be made quite trivial by the use of mechanical techniques of coding and assembly and by ignoring considerations of speed.

It would appear that the steps required in the solution of commercial problems would not differ greatly from these, although formal mathematical notation might not be involved. This analysis, therefore, leads me to draw my first moral from scientific experience. It is that efficient programming for a computer can be done only by someone who understands both the fundamental operation of the commercial system into which the computer work is to be fitted and also the techniques used in programming the chosen computer. This leads me to advocate strongly a "vertical" division of labour among programmers—one man to one job—rather than a "horizontal" division, with one man drawing up the flow diagram and another coding for the computer from it. However, the closest attention must be paid to the co-ordination of the work as a whole.

In the present state of development of the subject there are still several fields, particularly in engineering, where mathematical manipulation and even physical approximation has been carried out with a view to simplifying hand computation and the analysis has not yet been revised for computer operation.

In such cases when applying a computer the standard formulae for calculating the answers required should be regarded with reserve. It is our experience that it is often more rapid and accurate to solve the problem with a computer in a more primitive form.

This brings me to my second moral, which is that all methods in previous use for punched-card or hand systems should be treated with suspicion when examining a job to see if it is suitable for a computer. It is of great importance that a fundamental approach to the problem should be, at the least, considered although, of course, there may be many good reasons against its adoption in a particular case.

Turning now to a slightly more technical aspect, there are one or two programming concepts of great use in scientific work on which over-emphasis has, in my view, been placed in connection with commercial applications. The first of these is the development of autocodes, and the second the use of general subroutines (I do not refer to subroutines for multiplication, division, square-rooting, etc., which are little more than simple extensions of the basic code of the computer, but to more generalized routines utilizing several parameters).

The autocode in scientific work depends for its usefulness upon the assumption that all scientific users comprehend mathematical notation, and would be willing to accept a few restrictions on the use of it and one or two extensions to it, rather than have to learn computer codes directly. In the commercial field such an assumption is obviously invalid and, furthermore, most of the work required involves not so much mathematical as logical operations, the appropriate language for which is not widely known. It follows that any business autocode would most suitably translate plain English and it is notable that many such codes developed in the United States have taken steps in this direction of recent years.

If English words are not to be used, it is probable that any alternative code will be at least as difficult to learn as direct machine code, and the advantages of such a code over the direct code will be largely illusory. The translation of English is, of course, possible but elaborate, and it would probably be necessary to restrict drastically the words permitted in order to avoid difficulties and ambiguities. For practical purposes at present such an approach may be ignored or, at least, left to the universities and manufacturers to develop.

The usefulness of the generalized subroutine in scientific work is based on the fact that the solving of many problems in theoretical physics gives rise to similar equations (indeed, if they did not involve one of a limited number of situations we would be hard put to solve the problems quantitatively at all) and hence the only difference between one situation and another can frequently be defined by quite a small set of parameters. Furthermore, we do not often have to solve again any problem once completed, so that, pride apart, running time considerations are not of major importance.

In commercial applications the necessary analysis to codify the work satisfactorily has yet to be done, even for so apparently well understood a task as payroll. Moreover, almost all operations are repeated frequently, sometimes daily. Even as little as 1 minute of unnecessary work each day may imply 50 hours' lost time over the life of a program. If it requires one man-month of programming time to eliminate the waste, the saving may still be worth while. With the use of off-line printing systems, output time in itself is no longer a dominant consideration in programming. Thus careful examination must be made of the internal organization of the computer storage. It is usually found that routines arranged to deal with a wide generality of cases use the storage inefficiently in particular instances. This seems to indicate that a large number of routines taking advantage of certain special circumstances would be preferable to a single general routine in commercial work, even if such a routine could be constructed easily. Taken to its logical conclusion, in a computer with a large number of stores, all of equally fast access, loops should be written out in full and any routines used should be open. In order to avoid excessive preparatory work one would naturally turn to the use of generator techniques such as those developed by Grace Hopper.* However, in all computers suitable for commercial applications, more than one level of storage is used and these techniques themselves introduce inefficiencies.

Concluding our study of scientific applications, we may classify the types of work carried out into four main categories. These are: (1) mechanization of existing hand or punched-card applications; (2) mechanization of work known to be susceptible to ordinary numerical attack, but not carried out previously due to the prohibitive time or cost likely to be involved; (3) use of trial and error techniques; and (4) the use of simulation techniques. Naturally the study of all these techniques has given rise to new research and the techniques themselves have already found employment in many new fields. Many of the most profitable scientific applications of computers have largely been under (3) and (4), and we may expect that these will play a great part in an industrial context in the future. Some applications to organizational problems will be discussed in more detail in a later article.

* Compiling Routines. *Computers and Automation*, Vol. 2, No. 4, p. 1, May 1953.

British Computer Society Conference: June 1959

The Council of the Society has announced that the First British Computer Society Conference will be held in Cambridge from 22 to 25 June 1959. Details will be announced later.