RUTISHAUSER, H. (1961). "Interference with an ALGOL procedure" in *Annual Review in Automatic Programming*, Vol. 2, Goodman, R. (Ed.), Pergamon Press.

SHAH, B. V., BUEHLER, R. J., and KEMPTHORNE, O. (1961). "The method of parallel tangents (Partan) for finding an optimum," Office of Naval Research Report, NR-042–207 (No. 2).

SMITH, C. S. (1962). "The automatic computation of maximum likelihood estimates," N.C.B. Scientific Department Report, S.C. 846/MR/40.

Book review: ALGOL on the KDF9

ALGOL 60 *Implementation*, by B. RANDELL and L. J. RUSSELL, 1964; 418 pages. (London: *Academic Press Inc.*, 84s.)

The authors' intention in writing this book is to present a full description of their implementation of ALGOL 60 on the English Electric KDF9 computer. This aim has been most admirably fulfilled, both in the general description of the methods they have used, and in the detailed flow charts from which their programs were coded.

The general technique of implementation was based on the work of E. W. Dijkstra and J. A. Zonneveld, who wrote the first ALGOL 60 translator for the X1 computer at the Mathematical Centre, Amsterdam. The translator is built up of a number of routines, each of which processes one of the delimiters of the language. Each routine ends with a transfer of control to the basic input routine, which reads in and assembles the source text as far as the next delimiter, and passes control to the corresponding delimiter routine to process it. Many of the delimiter routines make use of a global stack for the storage of information which will be needed later by another routine. The stack mechanism is admirably suited for dealing with recursively structured languages such as ALGOL, in which expressions, and even statements, may be bracketed one inside the other to any depth. However, the method chosen for specifying the use of a stack seems rather clumsy, since all operations of pushing information down on the stack and restoring it again when required have been inserted as explicit instructions in the flow charts. A more elegant way of using a stack is to write the translator as a set of procedures which can call each other and even themselves in a recursive manner; in this case, the whole of the stack administration is incorporated behind the scenes in the procedure entry and exit mechanism. In a recursively organized translator, each procedure can be designed to process the whole of an ALGOL syntactic entity, rather than a single delimiter. This makes it possible to abolish many of the markers which otherwise have to be set, stacked, unstacked and tested in order to establish context in the source program. Since the ALGOL language recognizes the usefulness of recursion, it seems a pity that an ALGOL translator should deny itself the benefits which it makes available to others.

The implementation of an advanced programming language involves a great deal more than translating it, since a considerable amount of book-keeping must remain to be done at run-time; and the specification of control routines to perform this task is a major part of the implementation. The division of labour between the translator and the control routines is one of the most characteristic features of any system. The authors have chosen to simplify the task of the translator as much as possible, and to place a correspondingly heavy burden on the control routines. In fact, the object program produced by the translator is not even framed in the KDF9 machine code at all, but in a sort of idealized machine code, specially suited to the needs of ALGOL; and the control routines have the job of interpreting this code at run-time in order to execute the program. The main justification for this use of interpretation is that the system is designed for use in program testing. It is therefore most important that the translation process should be as fast as possible, since the programs are likely to be altered every time that they are run. In addition, interpretation makes it possible to include some extremely powerful facilities for diagnostic printout at run-time.

As far as the reader of the book is concerned, the use of the idealized machine code is of great benefit, in that the description is almost entirely computer independent, and in no way involves the particular idiosyncrasies of the KDF9 machine code. This will be of particular interest to prospective implementors of ALGOL, who may wish to use the same flow charts on a different computer, as has already been done on three other machines, Deuce, Pegasus and ACE. However, as the authors point out, the use of interpretation involves a very severe penalty in efficiency at run-time, which is likely to be tolerated only during program checkout; and it is expected that a fully tested program will be retranslated by a more complex compiler into KDF9 machine code. In the absence of such an alternative compiler, the prospective implementor would be well advised to produce object programs in the machine code of the computer on which he is working.

The presentation of the idealized machine code will also be of great interest to prospective designers of future machine codes, for it points clearly the direction in which they must orient their design. It is becoming more obvious that the power of modern computers cannot be fully exploited without the use of advanced symbolic programming languages; yet at present, the use of these languages involves a considerable expense, either in a lengthy process of optimization, or in the inefficiency of object program. It is therefore a matter of urgent practical economics that computer designers should pay close attention to the needs of implementors and users of symbolic programming languages.

An even greater contribution is the clear and detailed manner in which the authors explain the nature of the problems they encountered, and the way in which they tackled them. The book should be read with the utmost interest by all programmers who are concerned with the development and use of automatic programming languages, and, in particular, those who have not themselves had the opportunity of implementing such a language. For the benefit of this class of reader, the book includes a brief but competent survey of other published techniques, and a comprehensive bibliography.

The authors must be highly praised for the delightful clarity of their English prose. The writing of the book has obviously been a pure labour of love, and the effort and care which has been expended on it at least equals that spent on the programs which it describes. In spite of the immense wealth of detail, the main thread of the description is always kept to the fore; and as an exercise in the documentation of a complex algorithm, a standard has been set that will not readily be equalled.

C. A. R. HOARE.