which all subsequent variable calls refer is set; for example, the statement:

A = TEMPERATURE + 460

after the above line setting means:

A = TEMPERATURE (5) + 460

It may be necessary to refer to a particular component in a particular stream, this could be done using the variable STREAM COMPOSITION (I, J) or better still:

HYDROGEN (I)

which refers to the hydrogen in stream I or

HYDROGEN

if previously the statement LINE I has been used. Thus, if a particular line has been previously set, the composition of the stream represented by that line could be set:

HYDROGEN = 15
METHANE = 65
CARBON DIOXIDE = 19
CARBON MONOXIDE = 1

Conclusions

At present, no full-scale simulation has been completed but work is progressing on the writing of unit routines for a particular problem. The compiler will be used by final year undergraduate students as an aid to their design problem. No doubt the operation of integrated design programs will suggest valuable improvements to the compiler. It is already evident that much work needs to be done on a more standard approach to the design of particular units before the full-scale design of chemical plant can be achieved using a set of subroutines obtained from a library collection; it is hoped, however, that this compiler will help to promote further efforts in this direction.

It is proposed at a later date to raise the level of the language to that of ALGOL.

Acknowledgements

The author would like to thank Mr. F. R. A. Hopgood of the Atlas Computer Laboratory and Dr. R. B. E. Napper of the University of Manchester for their assistance in the use of the Compiler–Compiler.

References

Steward, D. G. (1967). *A survey of computer aided chemical process design*, Report ESL-R-304. M.I.T. Project DSR 74994.

Shannon, P. T. *et al.* (1966). Computer simulation of a sulphuric acid plant, *Chem. Eng. Progress*, Vol. 62 (6), p. 49.

Sargent, R. W. H. ,Westerberg, A. W. (1964). SPEED-UP in chemical engineering design, *Trans. Inst. Chem. Eng.*, Vol. 42, p. T190.

Westerberg, A. W. (1964). D.I.C. Thesis, Imperial College, London.

Andrew, S. M. (1967). Computer flowsheeting using Network 67: an example, *Trans. Inst. Chem. Eng.*, Vol. 46, p. T123.

Sargent, R. W. H. (1967). Integrated design and optimisation of processes, *Chem. Eng. Progress*, Vol. 63 (9), p. 71.

Cockerham, R. G. *et al.* (1965). The catalytic gasification of petroleum feedstocks, *J. Inst. Gas. Eng.*, Vol. 5 (2), p. 109.

Hutchison, H. P., and Forder, G. J. (1967). 'Chemical plant network specification using a visual display', Inst. Chem. Eng. symposium on 'Efficient computer methods for the practising chemical engineer', University of Nottingham, April, 1967.

Niemann, R. E. (1967). A generalised material and energy balance computer program, M.S. thesis, Faculty of Engineering Department of Chemical and Petroleum Engineering, University of Alberta, Edmonton, Canada.

Brooker, R. A. *et al.* (1963). The compiler-compiler, *Annual Review in Automatic Programming*, Vol. III, Pergamon Press.

Book Review

Theory of Recursive Functions and Effective Computability, by H. Rogers, Jr., 1967; 482 pages. (McGraw-Hill Publishing Co. Ltd., Maidenhead, £6 17s. 6d.)

Most of this book describes various current areas of research into recursive function theory which is essentially, as the author puts it, a mathematical theory concerned with questions of existence or non-existence of computational methods when limitations on time and memory are removed. The first three chapters provide a general introduction to the subject covering briefly, but extremely lucidly, the idea of an algorithm, of diagonalisation arguments, Turing machines, Church's thesis and the concept of recursive unsolvability. Topics covered in the remaining thirteen chapters are mainly concerned with unsolvability structures, i.e. with problems of classification of unsolvable problems. Applications to logic and the foundation of mathematics are made throughout but the book does not deal with application of recursive function theory to other areas. An example of the kind of problem dealt with is whether the proof of the undecidability of some task can be proved by using the well-known result that the halting problem is undecidable. Are all unsolvable problems really only the halting problem in another guise? How does one classify unsolvable problems?

The subject matter is presented in a semi-formal way; for example, Church's Thesis is often used rather informally (if it is 'obvious' how to compute some functions then that function is recursive). The book is extremely well written, and the proofs are models of conciseness, without being loose. There are numerous thought-provoking examples. A consistent notation (with its own index) is used throughout.

This book is not a gentle introduction to the subject. However, the first three chapters are required reading for anyone concerned with the theoretical side of programming, especially for the informal notion of an algorithm, for the remarks on possible usefulness of the theory and for the remarks on goals of the theory. Any worker in the area will need ready access to the book; much of the material exists only in mathematical journals and here we have a readable unified treatment. Any potential author of a book on a mathematical subject should look at this book to see how to present a difficult theory in a convincing, rigorous manner without confusing the presentation with the minute detail of formal proofs.

D. C. Cooper (Swansea)