USING MACHINE CODE WITHIN AN ALGOL SYSTEM

PETER NAUR

Abstract

The use of machine code within programs run in the GIER ALGOL system is made possible through the provision of standard procedures which transfer the control of the machine to the instruction held as a variable of the ALGOL program. The paper describes these and some associated standard procedures.

After its first distribution in September 1962 the GIER ALGOL system (ref. 1) rapidly was adopted by the users as the principal programming language for the machine. However, with this growing use from the users to have the system extended with facilities for mixing the execution of machine coding with the ALGOL program kept increasing. At the same time the extension of some of the machines with additional peripheral equipment, such as magnetic tape units and card readers, called for an extension of the ALGOL system with facilities for communicating with these units.

The solution to this problem usually adopted by compiler builders has been to extend the source language with new statement formats specifying the machine coding to be executed. However, to us this was most unattractive for several reasons: (1) This would require a major revision of the compiler. (2) It would require some ad-hoc definitions of the form which each ALGOL quantity should take when appearing in the machine code. (3) The automatic segmentation of programs performed by the GIER ALGOL compiler would impose a heavy limitation on the freedom of the programmer, almost to the extent of cancelling the advantages of the machine coding.

The clue to the solution adopted in this situation was the following: provide a standard procedure which transfers the control of the machine to the instruction held in the variable supplied as parameter of the call. More specifically, provide a standard procedure, \textit{gier}, which, when called as follows:

\begin{equation}
\textit{gier}(A[i])
\end{equation}

transfers the control to the location where \(A[i]\) is held. Clearly, this solution is both simple and general. It requires only a slight addition to
the compiler. Also, since the machine code is held as variables, it completely eliminates the problem of the automatic segmentation of the program.

Two problems remain, however: (1) The placing of the appropriate code in the array of variables, and (2) the communication between the machine code and the surrounding ALGOL text.

The first of these problems is solved by providing a standard procedure which enables the user to place any desired bit pattern in a variable. The following is an example of a call of this procedure:

\[
\text{pack}(A[i], B, C, D, E, F, G).
\]

This will pack into the variable \(A[i]\), from bit position \(B\) to \(C\) the bit pattern \(D\), and from position \(E\) to \(F\) the bit pattern \(G\). In each case the pattern is obtained by writing the integer given as parameter in binary form. As an aside it should be mentioned that clearly this packing facility may be used more generally for packing information. To cover such uses a corresponding standard procedure for splitting the bit pattern of a variable into any desired parts is provided.

The communication between the ALGOL text and the machine code held in arrays may be handled by variables held in the same array as the machine code. However, to provide for a more flexible communication an additional standard procedure, called \textit{gierproc}, has been included. An example of a call of this might be:

\[
\text{gierproc}(A[2], B, Q).
\]

This will transfer control to the instruction held as the variable \(A[2]\), like the call \textit{gier}(\(A[2]\)). However, the machine code will have access to the full descriptions of \(B\) and \(Q\), whether these identifiers denote variables or any other ALGOL quantities, such as these descriptions are supplied in any procedure call. To make proper use of these descriptions the user will only have to know their form for each kind of ALGOL quantity.

The above facilities will give the user access to executing any sequence of machine instructions and to communicating freely with any quantity of the ALGOL text. The principal inconvenience of this is the somewhat cumbersome building-up of the actual code held in the array by means of the \textit{pack} procedure. In order to remove this inconvenience one additional standard procedure has been added, serving to read data held on paper tape in binary coded form into the machine. Since this form of tape may be produced readily by the normal service program system, the machine code needed in an ALGOL program may be written in the