STICHTING
MATHEMATISCH CENTRUM
2e BOERHAAVESTRAAT 49
AMSTERDAM

MR 47

Operating Experience with ALGOL 60.

E.W. Dijkstra

[1962]

Reprinted from The Computer Journal, Vol. 5, No. 2

Operating experience with ALGOL 60*

By E. W. Dijkstra

This paper describes the circumstances under which the ALGOL 60 translator of the Mathematical Centre, Amsterdam, has been constructed. It describes its main features, virtues and shortcomings. Furthermore, it tells how the translator has shown itself to be an inspiring tool for a varied group of users.

For the Computation Department of the Mathematical Centre, Amsterdam, ALGOL 60 came exactly at the right moment, i.e. some months before our new machine, an X1 produced by N.V. Electrologica, The Hague, came into operation. And before going on I should like to mention some of the circumstances which greatly eased the development of our ALGOL 60 translator.

(*a*) As our older machine, the ARMAC, was still in operation and taking care of the service computations as in the preceding years, we could give the development of our translator the highest priority as far as X1 machine time was concerned.

(*b*) Faced with a new machine we were relatively free from preconceived notions as to how this machine should be used. No traditions with regard to its use had to be broken, because such traditions had not yet grown.

(*c*) As the speed of the X1 was considerably higher than that of the ARMAC, we had the joyous feeling that the speed of the object program did not matter too much. The resulting frivolousness saved us a great amount of trouble and pain.

(*d*) The fact that the X1 is a fixed-point binary computer enabled us to include suitable red-tape operations in the floating-point subroutines at little or no expense as far as run-time speed was concerned.

(*e*) Thanks to the fact that our previous machinery was highly unsuitable for any form of automatic programming, we did not have the slightest experience in language translation. We thought that this was a great drawback; it turned out to be one of our greatest advantages. Again there were no obsolete traditions to get rid of.

(*f*) In contrast to the great speed of the X1 its store was small: 4,096 words of ferrite-core store and no backing store. As a result all "strategy questions" where "space" and "time" had to be weighted against one another were settled almost automatically.

(*g*) As some of us, in particular Prof. A. van Wijngaarden, had been heavily involved in the creation of ALGOL 60, our group, as a whole, was probably more tolerant with respect to its less lucky features than would otherwise have been the case. As a result we did not waste our time on discussions as to subsets or modified versions, but took the challenge as it stood.

On the other hand, the limited size of the store has had two undesirable effects. Firstly, we did not dare to try a "load-and-go translator." We therefore aimed at a single-pass sequential translation, simultaneously reading the source text and punching out the object program. (It turned out that this pass had to be preceded by a rapid so-called "prescan" in which identifiers of procedures and labels are collected.)

* Report MR47 of the Computation Department of the Mathematical Centre, 2de Boerhaavestraat 49, Amsterdam, Netherlands.

Secondly, not being sure whether it could be done at all with a 4K store, we restricted ourselves to the bare minimum and omitted practically all syntax checking.

Not being too sure of the reliability of the paper-tape equipment we did include, however, a great variety of parity (and other) checks on the corresponding input and output operations. Without these and other precautions it is very doubtful whether we would have succeeded in getting the translator into operation at a date as early as August 1960.

Main Features of our Amsterdam ALGOL 60 System

Input of source program is via seven-hole paper tape, produced on a slightly adapted Flexowriter. (The code of this Flexowriter was inspired by the Flexowriter of Regnecentralen, Copenhagen; in particular we copied a non-escaping key with underlining as its lower-case symbol and a vertical stroke as its upper-case symbol.) The Mathematical Centre has now four of these; another ten are scattered over various places in Holland and Western Germany.*

Furthermore the Flexowriter tape-feed has been changed to blank tape, and the translator permits pieces of blank tape to be inserted between different "paragraphs" of the source-language program. This facility, when wisely used, considerably decreases the amount of paper-tape handling required for corrections.

We have only a few restrictions on the language. The main restrictions are a prescribed order for the declarations at the beginning of a block and, more serious, no own arrays with dynamic bounds.

Standard functions and library procedures are at the disposal of the programmer in the form of procedures which he may use without explicit declaration. It is as if our system embeds each given program in a surrounding block, at the beginning of which the library procedures are declared. This growing library also contains the procedures for input and output; in this way input and output have been implemented in a flexible, expandable way without violation of the syntactical rules of ALGOL 60, and without the introduction of new syntactical elements.

The translation process is fast: during translation the X1 performs an average of 1,000 machine instructions to generate one instruction of the object program. But roughly 50% of the instructions in the object program are subroutine calls—they activate a "macro"—therefore translation time should be negligible. However, due to the slow speed of the output punch, the translation process is output-limited by a factor of three, and our operator told me that in some unfavourable cases translation time became nearly equal to the time taken by the computation proper.

A further possible decrease in overall efficiency of the translation process may be caused by the amount of paper-tape handling required from the operator. I think that we have been insufficiently aware of the speed of the machine, and that we have not paid enough attention to the reduction of the amount of tape handling. The translation of a series of short programs keeps a trained operator continuously busy, inserting, winding and identifying tapes.

Somewhat to our surprise the working system proved to be a highly efficient, convenient and inspiring tool, efficient in particular for non-trivial problems. Here I can mention one of the things which contributed greatly to the overall efficiency: in the run-time system the extensive facilities of the X1 for parallel programming are much more thoroughly exploited than in nearly all machine-code programs.

After its completion our translator was very well received. In order to avoid possible misunderstanding I should like to point out that there are some marked differences between the computer fields in the Netherlands and in Great Britain.

First of all, "Autocodes" had hardly been used in Holland and, secondly, there is less stress on matrix operations. So ALGOL 60 had machine code as its main competitor and one of the objections frequently raised against ALGOL 60 in the United Kingdom, viz. the absence of matrix operations, was hardly heard.

The absence of autocodes in Holland was compensated for by rather nice machine order codes, and people were quite used to programming in machine language. I must say that before the creation of ALGOL 60 the need for an autocode was sometimes felt, but we hardly felt inclined to develop them. As I told you before, our previous machines were very ill-suited for automatic programming, and we thought the autocodes then existing insufficiently attractive as languages to invest much effort in their implementation. When ALGOL 60 came, we were very glad that the past had allowed us to skip the stage of the autocode.

With matrix operations it is another matter. In the U.K. you had one or more groups that were very active in this field, and the result was a number of powerful coding schemes for dealing with matrices. The presence of these schemes tends to attract computations suitable for them, and is partly responsible for your stress on the importance of dealing with matrices. Here is a considerable amount of feed-back. This must be borne in mind when I tell you how happy we are with ALGOL 60. If we have a feeling that it satisfies most of our needs, then this is partly due to the qualities of this language, but undoubtedly also to the fact that we had adjusted, unconsciously, our needs to the possibilities of this language.

Experience with the System

I should like to split the description of our operating experience into two parts, viz. the educational activities and the actual experience at the machine.

Our programming course on ALGOL 60 is given on four consecutive days. Each day consists of lectures in the morning and the afternoon, and exercises and demonstrations in the evening.

* And there is at least one in the U.K.—at the Cambridge University Mathematical Laboratory.—ED.

In these courses we cover the whole of ALGOL 60, including a thorough discussion of "Jensen's Device" and of recursiveness (or, more generally, nested activations of the same procedure). I mention this, because it is sometimes pointed out that these "advanced features of ALGOL 60" will frighten and repel potential users. Our experience quite definitely points in the opposite direction: the audience was thrilled by them every time the course was given. In practical computations these features are not too frequently used, but the bare fact that the programmers could use them if they wanted to made the language very appealing.

We have given the course four times, with a total number of about 240 participants, and one may ask how many machine users have been created in this way. For our own installation it is about thirty people, fifteen of whom are to be considered as regular machine users; the other fifteen turn up at less frequent intervals. I consider this a very good result, because these thirty people had to be recruited from those attendants that did not have their own machine at their disposal. At least one half of the participants were from other computing centres.

Actual use of our ALGOL 60 system is steadily increasing. I cannot give figures for other X1 installations, which have received copies of our translator. At the installation at the Mathematical Centre we started with 20% machine time spent on ALGOL programs; in the meantime this has been increased to 50%.

Omission of syntactical checking in translation has proved to have been a grave error. Every user finds that his first program contains a number of silly, clerical errors. This number of errors per program decreases very fast as the programmer gets more experience, and it is therefore my impression that it is hardly worth the trouble to let the translator look for the next error after the first one has been found. The omission of syntactical checking is the more regretful as it could have been incorporated at so little expense.

Furthermore, we find that the program for a particular problem is often processed in a couple of successive versions. Roughly: the first version is just plainly wrong, because it contains some logical errors, neglect of some exceptional cases, etc. The second version works, but the programmer is not satisfied with its performance. In the third version the programmer, who in the meantime understands his problem better, improves his strategy, and in a fourth version he improves on the programming. This is more or less the background of the fact that our "irregular users" suddenly turn up four times within a period of, say, two weeks; then we don't see them for quite a long time, but usually they return sooner or later . . . with their next problem. This experience is very encouraging.

The run-time system has no additional diagnostic facilities. We could include them, but from the fact that we have not done so one can deduce that the need for them is regarded as insufficiently urgent. In the case of longer programs it is quite usual to insert some conditional output statements in the earlier versions of the program. If they are enclosed between two pieces of blank tape on the input tape it is a trivial operation to remove them in the final version.

Furthermore, there is no possibility of a "post-mortem dump." There is no point in just printing out the contents of the store: as storage allocation is fully dynamic these data would be too hard to interpret. If a post-mortem dump were to be of any value it would have to produce the values of variables in store together with their identifiers in source language. This, however, would imply the availability of the complete "identifier table," but this is nowhere available in its entirety, not even during translation (this is only account of storage limitations).

The translator gives no print-out of the object program, again because there is no point in it. The structure of the object program has so little in common with that of handwritten programs that with a thorough knowledge of just ALGOL 60 on the one hand and just the X1 on the other, the print-out of the object program still won't be very helpful. As a consequence, all modifications and corrections must be made in the source-language program: we have made it virtually impossible to correct or to modify the object program, and we have done so on purpose. Some people like to have this possibility in order to avoid retranslation, but we regard this as an obsolete technique, which is not to be encouraged. Quite the contrary.

UNWIN BROTHERS LIMITED, WOKING AND LONDON