B The problem of a common language, especially for scientific numeral work (motives, restrictions, aims and results of the Zurich Conference on ALGOL)

By F. L. Bauer and K. Samelson, University of Mainz (Fed. Rep. of Germany)

The need for a common language for easy and precise intercommunication has been felt for a long time. This is shown by the existence of users' associations (SHARE, USE) which, however, have solved the communication problem only on the basis of their special computer language. For practical reason, it is obvious that such a common language should not be chosen to suit the order codes of one or more existing computers at the expense of others with different codes. There is no doubt that a universal computer-oriented language (UNCOL) is important for a number of technical problems of intercommunication, including that of translating from a common language to the special language for each computer; but it was felt by the Zurich Conference that a common language for numerical analysis and for scientific computation ought to be as close as possible to normal mathematical notation, which is already largely universal. A common language should be, like FORTRAN, an operational, constructive language (that is to say it should define constructively a sequence of operations in real time) such that a computer may either perform the orders as given or translate them mechanically into a computer language. This excludes all implicit mathematical definitions, but special attention was given by the Conference to the possibility of using free constructive definitions of numerical procedures to serve instead of sub-routines and library routines. The language required thus appears as an algorithmic language (ALGOL) in the sense of Rutishauser's early idea, and therefore as a problem-oriented language. The task of the Conference was to standardise the arithmetic notation and to enlarge it to make it fully operational.

Le problème d'un langage commun notamment pour les travaux scientifiques numériques (raisons d'être, limites, buts et résultats des travaux de la Conférence de Zurich sur l'ALGOL). Le besoin d'un langage commun qui permettrait aux hommes de science de communiquer aisément et avec précision se fait sentir depuis longtemps déjà, comme en témoigne l'existence de plusieurs associations d'usagers (SHARE, USE). Pourtant celles-ci n'ont résolu le problème qu'à partir du langage propre à leur calculatrice. Or il semblait évident, pour des raisons d'ordre pratique, que le langage commun recherché ne pouvait pas être un langage spécialement adapté à l'une ou à plusiers des calculatrices existantes, car elles utilisent des langages profondément différents, et toutes les autres seraient ipso facto défavorisées. Il n'est pas douteux qu'un langage adaptable à toutes les calculatrices (UNCOL) contribuerait puissamment à résoudre un certain nombre de problèmes techniques de communication, et faciliterait notamment le passage du langage commun au langage particulier de chaque machine; mais il a semblé aux spécialistes participant à la Conférence de Zurich, qu'un langage commun conforme aux besoins de l'analyse numérique et du calcul scientifique automatique, devrait s'écarter le moins possible de la notation mathématique usuelle, ce qui serait d'ailleurs, une première garantie d'universalité. Un langage comme le FORTRAN devrait constituer un langage constructif opérationnel (désignant une série d'opérations en temps réel définies de manière constructive) qui permettrait à une machine d'exécuter diversement les ordres reçus ou de les traduire automatiquement dans son langage particulier. Cela exclut tous définitions implicites de la mathématique, mais la Conférence s'est particulièrement attachée aux possibilités d'utilisation de définitions constructives libres des procédés numériques, notion qui s'apparente à celle de sous-programme et de programme pré-enregistré. Ainsi, le langage désiré se présente comme un langage algorithmique (ALGOL) conforme à l'idée primitivement émise par Rutishauser, c'est-à-dire un langage conçu en fonction du problème à résoudre. Il appartenait à la Conférence de normaliser la notation arithmétique et de la compléter en vue de lui donner un caractère parfaitement opérationnel.

Das Problem einer einheitlichen Sprache, insbesondere für wissenschaftlich-numerische Arbeiten. Schon seit langer Zeit besteht das Bedürfnis nach einer einheitlichen Sprache zwecks Vereinfachung und Präzisierung des Austausches von Informationen. Die bestehenden Benutzungsgemeinschaften von Rechenautomaten zeigen dies deutlich. Sie haben jedoch das Problem nur auf der Grundlage der Maschinensprache eines speziellen Rechners gelöst. Eine einheitliche Sprache kann aus praktischen Gründen natürlich nicht eine spezielle auf Rechenmaschinen gerichtete Sprache sein, welche dann einige der existierenden Rechenmaschinen begünstigte, die in ihren Maschinensprachen stark voneinander abweichen. Ohne Zweifel hat eine allgemeine Sprache vom Typ der Rechenmaschinensprachen (Universal Computer Oriented Language UNCOL) Bedeutung für einige Probleme, z. B. für das Problem der Übersetzung einer einheitlichen Sprache in eine Rechenmaschinensprache. Aber in der Konferenz in Zürich war man der Meinung, daß eine einheitliche Sprache zum Gebrauch für wissenschaftliche Aufgaben und insbesondere für Aufgaben der numerischen Mathematik sich so weit wie möglich an die übliche mathematische Schreibweise anschließen sollte, die ohnehin schon weitgehend universell ist.

Eine einheitliche Sprache sollte — genauso wie FORTAN — operativ und konstruktiv sein (d. h., daß in dieser Sprache Folgen von konstruktiv definierten Operationen ausgedrückt werden, die in der Zeit ablaufen sollen). Dies ist notwendig, wenn die Sprache von einer Rechenmaschine interpretiert werden soll, oder wenn sie maschinell in die Sprache einer Rechenmaschine übersetzbar sein soll. Dies schließt alle diejenigen Teile der mathematischen Schreibweise aus, in denen implizite Definitionen verwendet werden. Großen Wert hingegen legte die Konferenz darauf, daß die Sprache Möglichkeiten zur konstruktiven Definition von numerischen Prozessen enthält an Stelle von Unterprogrammen und Bibliotheksprogrammen. Die angestrebte Sprache ist also eine algorithmische Sprache (ALGOL) im Sinne von Rutishauser's erster Idee, und sie ist eine spezielle „Problem-orientierte" Sprache. Es war die Aufgabe der Konferenz, die arithmetische Schreibweise zu standardisieren und diejenigen Ergänzungen hinzuzufügen, die sie vollständig arbeitsfähig machen.

Проблема универсального языка специально для численного решения научных задач (мотивы, ограничения, цели и результаты Цюрихской конференции по АЛГОЛ). Значение такого универсального языка для легкой и точной связи ощущается уже в течение некоторого времени. Это подтверждается существованием ассоциации потребителей(SHARE, USE), которые, однако, разрешили проблему связи только на основе специального языка для их вычислительных машин. Если исходить из практических соображений, то кажется очевидным, что универсальный язык не может быть даже специальным языком, ориентированным на вычислительную машину, если отдать предпочтение одному или нескольким из существующих языков для столь различных вычислительных машин. Несомненно, язык, ориентированный на универсальную вычислительную машину (UNGOL) имеет значение для ряда технических задач связи, включающих процесс преобразования общего языка в языки вычислительных машин; однако Цюрихская конференция показала, что общий язык для численного анализа и для использования вычислительных машин для научных вычислений должен быть как можно ближе к общей математической системе обозначений, тем более что она уже является унифицирующим элементом. Вопрос стоит о том, чтобы язык, подобный FORTRAN'у, был операционным конструктивным языком (основной смысл которого состоит в последовательности операций в реальном масштабе времени, определяемой конструктивным путем), допускающим использование языка вычислительной машиной или механическое преобразование его в специальный язык для вычислительных машин. Это исключает, конечно, все подразумеваемые части (использование подразумеваемых операций) математической системы обозначений. Однако на конференции особое внимание было уделено возможности использования свободных конструктивных определений для процессов исчисления, как дубликата идеи подпрограмм и библиотечных программ. Таким образом желаемый язык представляется как алгорифмический язык (АЛГОЛ), в смысле ранних идей Рутисхаузера, т. е. ялвеятся специальным языком, ориентированным на задачу.

В задачи конференции входила стандартизация арифметической системы обозначений и дополнение ее, чтобы она была полностью операционной.

El problema de un lenguaje común, especialmente para las tareas científicas numéricas (motivos, restricciones, objetivos y resultados de la Conferencia de Zurich sobre el ALGOL). Desde hace algún tiempo se ha comprendido el valor de un lenguaje común para una intercomunicación científica fácil y precisa. La prueba de ello es que existen organizaciones de usuarios (SHARE, USE) las cuales, sin embargo, sólo han resuelto el problema de la comunicación sobre la base de su lenguaje particular de cálculo. Por razones prácticas parece obvio que un lenguaje común no podría ser un lenguaje que favoreciera más o menos a una o a algunas de las calculadoras existentes, diferentes en una medida considerable en su lenguaje de cálculo. Sin duda un lenguaje universal orientado al cálculo (UNCOL) tiene importancia para cierto número de problemas de intercomunicación técnica que incluyen los procesos de traducción de un lenguaje común al lenguage particuliar a cada calculadora; pero se pensó en la Conferencia de Zurich que un lenguaje común para el análisis numérico y para el empleo de calculadoras en el cálculo científico habría de ser lo más cercano posible a la notación matemática común, tanto más cuanto que ésta constituye ya un elemento unificador común. No se hizo cuestión del hecho de que un lenguaje como FORTRAN debiera ser un lenguaje constructivo operacional (es decir una secuencia de operaciones en el tiempo real definido de un modo constructivo) con el fin de que una calculadora pudiese ejecutar los adenes dados o trasladarlo mecánicamente a un lenguaje especial de las calculadoras. Esto excluye todas las definiciones implícitas de la matemática. Pero la Conferencia prestó especial atención a la posibilidad de emplear definiciones constructivas libres para los procedimientos numéricos en lugar de rutinas y subrutinas de bibliotecas. Así el lenguaje buscado se presenta como lenguaje algorítmico (ALGOL) en el sentido de la idea primitiva de Rutishauser, siendo un lenguaje orientado a problemas especiales. La tarea de la Conferencia consistió en la standardización de la notación aritmética y en completarla con el fin de hacerla enteramente operacional.

1. Motives for ALGOL

In May 1958, a conference between representatives of the ACM ad hoc committee on language and the GAMM Programmierungsausschuss took place at Zurich, Switzerland. The conference succeeded in working out an algorithmic language (ALGOL) for scientific numerical work, which was intended to serve as a common language for numerical analysis and to be used in scientific computations. A conference report has been published simultaneously in the journals "Numerische Mathematik" and "Communications of the ACM". It is a preliminary record of ALGOL and is subject to possible corrections and improvements, both formal and essential. It is based on thorough, independent preparatory work of several groups, representing both designers, manufacturers and users of computers and also mathematicians concerned with numerical analysis.

The fact that a compromise was possible between groups with quite different philosophies and approaches, means a remarkable progress. In this way the very real danger was eliminated that these groups would each develop their own language, whereas it seemed to be far preferable that competition should be directed toward a common goal. Since the need for easy and precise intercommunication in the computer field has been felt for some time, particularly with respect to numerical analysis and scientific computations, it is hoped that ALGOL will be widely accepted.

2. Operational-constructive languages

Since common languages in the computer field are not only means of intercommunication, but also of giving instructions to computers, they are necessarily operational-constructive languages. It will be worthwhile to analyse first this concept in order to see how ALGOL fits into the scheme. An operational-constructive language (OPCOL) deals with constructive operations and their objects, both of them identified by certain symbols. It is a formal system with syntactical rules of formation of permissible patterns of symbols, representing constructive schemes of given operations on given objects.

Any computer language is operational-constructive, the objects being coded contents of storage locations and registers, the operations being standard transformations of these data. Existing operational-constructive languages differ in the domain of objects and of operations. In the simplest cases, we have numerical objects (represented by digits, or symbolically by variable names), and arithmetical operations; or logical objects (truth values) and operations of propositional logic. In a more advanced case, the objects may be formal expressions of, say, algebra, set theory or formal logic, with operations of reduction, check for identity or expansion. There are also examples of operational-constructive languages whose objects are functional expressions, the operations being formal differentiation and integration.

The simplest and most natural way of defining syntactical rules for an OPCOL follows from its operational structure. But the same operational structure may be expressed in different notations, as is well known from mathematics and formal logic. To give an example, let us consider the expression in usual arithmetic notation:

$$((a + b) \times (a - b) + c) \times d$$

In a functional notation, where

Σ (x, y)	is the sum	$y + x$
Δ (x, y)	is the difference	$y - x$
Π (x, y)	is the product	$y \times x$

this reads

$$\Pi (d, \Sigma (c, \Pi (\Delta (b, a), \Sigma (b, a))))$$

In operator notation, as used in group theory, define

A_a the addition of a
S_a the subtraction of a
M_a the multiplication of a

Then we get

$$M_d\, A_c\, M_{S_b a}\, A_b\, a$$

In formal logic, the Warszawa parenthesis free notation

$$\Pi d \Sigma c\ \Pi \Delta b\ a \Sigma b a$$

is used, which is called a simple language [1].

Functional and operator notation are obviously isomorphic and may be shortened immediately to a simple language. Usual arithmetic notation, on the other hand, has rules of precedence, which complicate its operational structure. Like most existing notations, these rules are sequential, despite the fact that complicated structures become more clear in two dimensional notation. Unfortunately, higher dimensions are normally ruled out for technical reasons. In all cases the syntactical rules can most effectively be described in a recursive way. The formal rules of composition are expressed easily by means of insertion in the case of functional notation, or by a process of agglomeration of strings of symbols in the Warszawa notation. Both show how much non-essential ambiguity is possible in developing formal languages.

3. Limitation of operational-constructive languages

Consider the static equilibrium expressed by

$$q_1' = q_1 + e_1$$
$$q_1' e_1' = e_1 q_2$$
$$e_1' + q_2' = q_2 + e_2$$
$$q_2' e_2' = e_2 q_3$$
$$e_2' + q_3' = q_3$$

This system of equations has no operational meaning. It may be solved for the dashed quantities, giving the construction

$$q_1' := q_1 + e_1$$
$$e_1' := (e_1 q_2)/q_1'$$
$$q_2' := q_2 + e_2 - e_1'$$
$$e_2' := (e_2 q_3)/q_2'$$
$$q_3' := q_3 - e_2'$$

where the $:=$ sign means "defined by" or "results from" and has a dynamic, directional meaning. But it may be solved also for the undashed quantities, giving

$$q_3 := e_2' + q_3'$$
$$e_2 := (q_2' e_2')/q_3$$
$$q_2 := e_1' + q_2' - e_2$$
$$e_1 := (q_1' e_1')/q_2$$
$$q_1 := q_1' - e_1$$

In some cases, as here, we may succeed in deciding from the context what explicit method of calculation is meant. But generally, implicit definitions such as

"x such that $x^3 + a_1 x^2 + a_2 x + a_3 = 0$"

do not define the constructive procedure within the given domain of numerical objects and arithmetic operations. Indeed, questions of existence and uniqueness are to be solved first, and even a uniquely existent result may be obtained by different ways of approximation. Therefore, we exclude from OPCOL even trivial cases of implicit definition. In the domain of numerical operations, the equality sign $=$ is meaningful only for stating a condition between already constructed objects.

4. The purpose of an operational-constructive language

An OPCOL may contain free definitions which state that some new special sequences of letters are to be taken as abbreviations of other sequences already defined within the OPCOL. They may be used for the practical purpose of avoiding complicated repetitions. From a theoretical point of view, however, it is of interest to determine whether complicated instructions in an OPCOL can be expressed by free definitions using simpler operations. Thus it is possible to consider the problem of finding a corresponding primitive OPCOL, in which no operations can be expressed in terms of other operations of the OPCOL. This is important for the study of the theory of translating one OPCOL into another (say a computer language) by means of a learning device. However, we should not for the time being expect such a learning device to simulate intuition. It seems that the translater resulting from the learning device will be as intelligent or unintelligent as the teaching of its instructor.

This last remark leads to the necessity to take into account the purpose of an OPCOL. While an OPCOL for theoretical studies may be developed quite independently, an OPCOL for practical use is a means of communication and instruction within a certain circle of application, and it will be considered to be a common language within this area. Therefore, its structure, notation and symbol representation will be influenced by these circumstances.

Machine languages (ML), from their very meaning, are operational-constructive languages, in which the operands are the coded contents of storage locations and registers and the operators are standard transformations of these data. They differ in as much as existing computers differ. Therefore, a universal computer oriented language (UNCOL) may be of real value in permitting communication on a level close to computers. Recent studies [2] on a hypothetical UNCOL as a basic language show a variety of useful applications even in dealing with translation of an OPCOL to a special ML. It turns out that the UNCOL idea does not compete with other, more problem-oriented languages.

These problem-oriented common languages (POL) can have only a limited universality, since the structure, notation and symbolism of customary ways of stating problems are quite different in different fields, being governed by internal structural peculiarities and by long traditions. The situation is extremely rigid in all fields close to mathematical treatment, where universality extends even over two hemispheres. In particular, there is the wide class of problems arising from scientific and other fields which lead to numerical computation, problems whose customary formulation is strongly influenced by mathematical structure, notation and symbolism. Thus, a POL for scientific numerical computation is, apart from some lack in standardization, already predetermined to a high degree. Other POL may be useful in economic data processing, in handling mathematical structures, in stating translation processes for natural languages or in stating translations between OPCOL's. Obviously, a universal problem oriented language may not be expected to exist.

5. General requirements for a mathematical common language

Since a fully universal language may be unattainable, the Zurich Conference restricted its aim strictly to a practical problem oriented common language for scientific numerical calculation, under the following main objectives:

1) "The new language should be as close as possible to standard mathematical notation and be readable with little further explanation".
2) "It should be possible to use it for the description of computing processes in publications."
3) "The new language should be mechanically translatable into machine programs."

Most of these requirements are met already, at least partially, by previous attempts, such as FORTRAN and MATH-MATIC. It is interesting to compare this with a list of aims of a group working in the Soviet Union [3] on automatic programming:

a) "The representation of the source information has to be close to the mathematical formulation of the problem."
b) "The size of the auxiliary and technical work, not connected with the mathematical formulation, has to be reduced to minimum."
c) "The source information must give the full information about the structure of the object program."
d) "The source information has to be maximally compactable and lookable."

With this approach, it may be expected that the next step in the "programming programs" of the Moscow Academy would come very close to ALGOL. Indeed, there is essentially coincidence between the two groups of requirements. Correspondingly, the Moscow Academy PPS is structurally equivalent to basic ALGOL, but includes a level of more computer oriented language similar to customary compiler techniques. No effort was made in this direction by the Zurich Conference, since such questions were considered to have their proper place in the discussion of a universal computer oriented language (UNCOL).

6. Detailed requirements to be met by ALGOL

6.1 *Structural postulates*

For a problem oriented computational language, the primary objects obviously are numbers, to be connected by arithmetic operations. Since decisions depending on binary relations are unavoidable operations, the operations of Boolean algebra, with truth values as objects, are a consequent, although not strictly necessary, addition. Whence a certain number of different operations and certain classes of objects including the respective symbolic representations (such as variables, relations, Boolean variables) are necessary. Rules of composition for these operations are recursive in the well known way used, for example in the foundation of number theory.

The concept of sentences in natural languages has an ALGOL counterpart in self-contained structures, such as arithmetical formulae, called statements. Compound statements, corresponding to sections, paragraphs, chapters, volumes may be formed by means of a non-overlapping parenthesis structure, which provides a unique way for labeling parts of an ALGOL program.

The possibility of adding new components to the language by free definitions was considered to be essential. These definitions may be taken to be abbreviations for certain sequences of operations expressed either in ALGOL or in any other OPCOL. They have meaning only if they are supported by the defining statements, and are therefore not equivalent in validity to the basic symbols of the language. No attempt was made by the Conference to create a metalanguage providing facilities for introducing new basic symbols into the language. However, this does not mean a disapproval by the Conference of a study of such advanced languages.

6.2 *Formal postulates*

A problem oriented common language should conform to universally accepted conventions not only in structure but also in form. This means that the characters used in ALGOL to represent the abstract symbols of the language should be those commonly used in mathematics, as far as necessary and possible, provided that this does not lead to ambiguity. Furthermore the language should, whenever possible, be self-explanatory in the cases where no standard conventional notation parallel to the symbols of the language exists. This means that such symbols should be words from a natural language which indicate the meaning of the symbols.

6.3 *Technical postulates*

Finally, there are, for a language such as ALGOL, certain very important technical conditions to be considered. One of the reasons, indeed the main reason, for the development of ALGOL was the wish to have an easily understandable, and mechanically translatable programming language. This last purpose can be truly attained only when the writing of programs made in the language can be coupled mechanically to coding, and no intermediate human recording is necessary. Therefore, characters used in the reprensentation of the language should be available on commercial coding devices such as tape or card punches, which means that the total number of characters is severely bounded.

On the other hand, the number of characters available varies with the type of machinery, and may increase considerably for new hardware appearing in the near future. For this reason, a set of characters was used for the documentation of ALGOL which is, in effect, a compromise between what is currently available on coding machinery, and what is desirable (and in common use in printed textbooks). To indicate expressly that this language serves primarily as a reference point and should not prejudice choice of characters for use on hardware, the language was called the "reference language".

However, the characters of the reference language were so chosen that the most important of them are, or can be made, available on existing hardware. They are therefore recommended for practical use in so far as this is feasible. This implies at the same time the desirability of forming "hardware groups"; that is groups of institutions using identical hardware for coding, which should employ the algorithmic language in a form identical down to the set of characters used. This would allow free exchange of programs without any human interference between all members of the group. A start was made by those members of the ZMMD group in Europe using teletype CCIT 2 coding equipment. Furthermore existing hardware demands that a strictly linear notation, possibly broken down into lines, should be chosen. Since it is not practicable with this limitation to deal with the subscripts and exponents which appear in publications, the characters of the reference language were so chosen that a direct transliteration to genuine subscript and superscript notation in the publication language is possible.

7. Additional features of ALGOL

Arithmetic operations, together with conditions depending on Boolean expressions and a forwarding operation to describe multiple connectivity of the flow graph in linear form, already form a primitive system sufficient for the immediate purpose of ALGOL. In addition ALGOL contains elements called "declarations" which give only additional information about properties of objects, such as the character of numbers (integer, double precision, complex, Boolean) or the dimensionality of arrays of numbers. There are also definitions of functions and of more complicated processes, called procedures, which correspond to the library programs of orthodox programming.

From practical considerations it was felt desirable to add certain redundant operational elements to the language which are abbreviations of processes expressible in the primitive language. These elements are

1) a symbol describing recursions in a given variable,
2) symbols describing a multiple alternative and a forwarding label dependent on arithmetic calculations,
3) a symbol interrupting the linear flow of sentences of the language which effectively causes the insertion of a certain sequence of sentences written down separately, allowing substitutions at the same time.

This concludes our remarks on ALGOL, detailed survey of its syntactical structure is given by J. W. Backus (these proceedings, II C).

8. References

[1] Rosenbloom, P. C.: *The elements of mathematical logic.* New York: 1950.

[2] Strong, J., et al.: *The problem of programming communication with changing machines.* Comm. A.C.M. 1, 1958.

[3] Ershov, A. P.: *The work of the computing centre of the Academy of Sciences of the USSR in the field of automatic programming.* Lecture, Symposium on the Mechanization of Thought Processes, Teddington (England), November 1958.

9. Discussion

C. Strachey (UK) disagreed with the basis of ALGOL.

a) It is undesirable to attempt to confine numerical analysis to a rigid and strictly limited system of notation. The ALGOL reference language is such a language and as such is bound to be inadequate. The general acceptance of any rigid language would fetter the development of new ideas but unless a language is accepted totally its formalities become worse than useless.

b) There are two distinct problems which are confused in ALGOL. The first is that of devising a system for describing algorithms, since the ordinary notation of mathematics is not able to describe the dynamic processes involved at all adequately. The other problem, which by comparison is of trivial difficulty, is how to write the resulting description of an algorithm in a way which can easily be fed into a computer. The reference language and thence the publication language of ALGOL are dominated by the second of these problems, which is enough to disqualify ALGOL as a publication language for general use.

c) The reference language is not itself fully determined without some extra statements. For example, the details of the arithmetic operations such as word-length, method of rounding etc. require specification.

K. Samelson (and *F. L. Bauer*) in answer:

a) The ALGOL language was never intended to be used in theoretical mathematical research, though it may be used to present the results of this research to other people. It is considered to be adequate to do this at present.
b) The ALGOL publication language is only a transliteration which has certain visual advantages in printed texts. Common mathematical notation will always be available in addition, for example, in text books.
c) This will be considered in future work.

J. E. Bartlett (USA): Discussions in the USA with reference to ALGOL etc. produce enthusiasms and dissents similar to those shown here. Some computer users began to consider an external language as much as 10 years ago, and for some years there have been attempts by individual users to extend the vocabularies of such languages. What seems necessary now is to examine the extent to which engineers can assist in the evolution of common languages.
Many active users are trying to bring the computer language nearer to ALGOL and to minimise the transliteration from the reference language to the publication language. Several such attempts have been reported, and one example appears in our collaboration with the Los Alamos Scientific Laboratory, University of California. This involves the use of an automatic typewriter with 132 characters, and with superscript and subscript control, as the basic coding instrument. Other coding equipment allows this information to be coded on cards with 12 bits per column and 80 columns per card, or on 6 hole paper tape for the Los Alamos Maniac II. It can also list information both from the Los Alamos computer and from its own output to assist in the maintenance of the decks of cards. It has recently been decided to mark each binary column in a way which will aid manual retrieval and filing of cards.
Finally there are two questions. It is stated in the paper on ALGOL — "complicated structures become clear in two dimensions. Unfortunately, higher dimensions are normally ruled out for technical reasons". Is it to be inferred that these technical reasons are due to hardware, and that the lack of suitable hardware has hampered the achievement of a widely used programming language? Secondly, does the conventional use of superscripts and subscripts give an adequate multi-dimensional notation? If hardware problems are imposing compromises, then many engineers would profit from early collaboration with users.