AB31.3.3 ALGOL 68 - Comments and Recommendations

Henry J. Bowlden

Westinghouse Electric Corporation
Research & Development Center
Churchill Boro.
Pittsburgh, Pa. 15235
Computer Sciences 401

April 1, 1969
Received 10.4.69 addressed
to Editor AB.

To: The Architects of Algol 68

Gentlemen:

Please consider the following comments and recommendations with respect to Algol 68.

My base reference is MR99, with the exception of the revisions to section 8.4.1. These are based on a copy of version contained in MR100. I am indebted to Professor S.W. Marshall for this copy and for discussions on the monadic operator problem.

Monadic Operators

I am especially concerned about the problems of incompatibility which arise as a result of the unusual handling of monadic operators in MR99. This leads to special difficulties in the implied result that $(0-a\uparrow 2) \neq (-a\uparrow 2)$, contrary to all algebraic conventions. The argument that "↑" is not the conventional exponentiation operator is an evasion of the real issue, which lies with the improper handling of the monadic operators.

I understand, from miscellaneous correspondence, that a major reason for the present handling of this problem was the difficulty of presenting the meaning of intermediate-priority monadic operators in a simple way. The proposal below demonstrates that a simple two-step algorithm suffices to give a complete parse of a formula, and also provides the syntactic alterations required to implement it.

Non-Local Quantities in Routines

A careful reading of section 8.6.2.2, the "copy rule" of Algol 68, leads to the conclusion that non-local quantities in a routine are to be interpreted in the context of the call, rather than that of the denotation. This is a departure from Algol 60 which I suspect is unintentional. Furthermore, although it is bad enough for identifiers, the possibility of redefining operators and made declarators makes it impossible even to begin to parse a routine denotation in situ.

Recursive Operator Definitions

Consider the definition

```
op fact = (int n) real:  (n=0 | 1 | :n > 0 | n × fact (n-1))
```

This appears to be allowed by the report, but would seem to cause an infinite recursive loop in the compiler when it is elaborated. Since no provisions exist for terminating such a recursive loop, some safeguard must be incorporated into the specifications.

Formal Parameters "Called by Value"

The Algol 60 "call by value" parameter which can be used as a local variable is missing for no apparent reason. It represents a significant fraction of uses of the parameter mechanism. The following modification to Step 2 of 5.4.2 will restore this facility.

> Step 2: If the routine denotation does not contain a formal-parameters-pack, then Step 3 is taken; otherwise, each of the formal-parameters of the copy is modified in the following manner. If the MODE is a reference mode, an equals-symbol followed by a skip-symbol is inserted following the identifier; otherwise, the sequence 'formal MODE declarer, MODE identifier' is replaced by 'reference symbol, formal MODE declarer, MODE identifier, equals-symbol, MODE local generator, becomes symbol, skip symbol'. After all formal-parameters of the formal-parameters-pack have been altered in this manner, its open-symbol is deleted and its close-symbol is replaced by a go-on-symbol.

Regarding Integer Division

There are two internally consistent pairs of definitions for the quotient and mod operators; but the important point is that they must be a pair. It must, in particular, be possible to reconstruct the dividend from values of the divisor, quotient, and remainder. The definitions in MR99 lead to an ambiguity in this process. One may define a÷b either as entier (a/b) or as sign (a/b) entier abs (a/b); but in either case, a÷:b must be defined as a -a÷b×b. The first definition above gives the number-theoretic interpretation of the remainder; the second is more commonly available in hardware. A hybrid set is not acceptable.

Structure Extensions

Both SIMULA 67 and GPL contain objects very similar to the structure of Algol 68. The major difference in each case is the possession of a routine. This is not just a field of the structure, but rather is "the operating rule", with the fields of the structure being treated as parameters to which it has direct access. SIMULA 67 also allows for the sub-class concept which improves the capability of the language in several ways. I think it would be worth the effort to design an extension (probably optional) to Algol 68 which would allow similar features.

Default Declarations

A prime criticism raised by devotees of other languages is the requirement of declarations for every identifier. It is only after extensive experience with the improved diagnostic power so provided that one comes to concede the value of this requirement. However, this requirement becomes even more of a nuisance in a conversational computing mode. There follows a proposal for providing a default facility at the option of an implementation (or even of a programmer).

Dr. Henry J. Bowlden
Advisory Physicist
Computer Sciences R & D

ALGOL 68

Proposed Revision for Monadic Operators

8.4.1 Syntax

a)* SORTETY formula: SORTETY MOID ADIC formula.

b) MOID PRIORITY formula firm LMODE-PRIORITY left operand, procedure with LMODE parameter and RMODE parameter MOID PRIORITY operator, firm RMODE PRIORITY plus one right operand; procedure with RMODE parameter MOID PRIORITY operator, firm RMODE PRIORITY plus one right operand.

c)* operand: firm MODE ADIC OPERAND.

d) firm MODE PRIORITY OPERAND: firm MODE PRIORITY formula; firm MODE PRIORITY plus one OPERAND.

e) firm MODE priority NINE plus one right operand: firm MODE monadic operand.

f) firm MODE priority NINE plus one left operand: firm MODE secondary.

g) firm MODE monadic operand: firm MODE monadic formula; firm MODE secondary.

h) MOID monadic formula: procedure with RMODE parameter MOID PRIORITY operator, firm RMODE PRIORITY plus one right operand; procedure with RMODE parameter MOID priority NINE plus one operator, firm RMODE monadic operand.

In 1.2.4, add

o) OPERAND: left operand; right operand.

Restatement of section 7.3 heading;

{Priority-declarations provide the indication-defining occurrences of priority-indications, e.g. σ in priority σ = 6, which may then be used in the declaration of operations. Priorities from 1 to 9 are available. If an operator-defining occurrence of a monadic operator does not identify an indication-defining occurrence (see 4.2.2.b) then the operator is a monadic operator of priority 10.}

Rule for Elaboration of a Formula

In the rule, μ, β, and σ stand for monadic operators, dyadic operators and secondaries, respectively. An operator is classified as monadic by the absence of a secondary to its left.

Step 1: If the formula has the form μσ or σβσ, then it is elaborated using the applicable definition of its operator, and the elaboration is complete. (If no definition of the operator is applicable, the formula is syntactically incorrect.) Otherwise, proceed to step 2.

Step 2: The formula is scanned, from the left, to find a combination of the form μσβσ or $\sigma\beta_1\sigma\beta_2\sigma$ in which the priority of the first operator (μ or β_1) is not less than the priority of the second operator (β or β_2). The first operator with its operand(s) is elaborated as described in step 1, and the resulting value is substituted as a σ. Repeat from step 1.

Proposal for Default Declarations

I. Syntactic Additions

Add to 3.1.1 d:

default symbol	...
global symbol	global
local symbol	local

Alter 7.4.1:

a) identity declaration: local symbol option, formal MODE declarer, MODE classifier, equals symbol, actual MODE parameter.

c) MABEL classifier: MABEL identifier;
MABEL prefix, default symbol.

d) global declaration: global symbol, MODE classifier.

Add to 4.1.1:

f)* prefix: MODE prefix.

g) MODE prefix: TAG; empty.

II. Semantic Modifications

Replace 4.1.2, Step 3 by:

Step 3: If the home contains a defining occurrence of terminal production of 'MABEL classifier' which matches the given occurrence by the criteria in 4.1.2C, and which is not in a global declaration, then Step 4 is taken; otherwise, Step 2 is taken.

Step 4: If the defining occurrence is a terminal production of 'MABEL identifier' then the given occurrence identifies it; otherwise, a defining occurrence of the same terminal production of 'MABEL identifier' is generated and the given occurrence identifies it. All such generated defining occurrences of any given terminal production of 'MABEL identifier' in a given reach are to be considered one single defining occurrence.

4.1.2C: A terminal production of 'MABEL classifier' is said to "match" a terminal production of 'MABEL identifier' if and only if:

i) The number of characters in the prefix is not greater than the number of characters in the identifier

ii) The characters in the prefix are identical to the initial characters of the identifier.

iii) If the terminal production of 'MABEL classifier' does not contain a default symbol, then it matches if and only if it is the same terminal production of 'MABEL identifier'.

iv) If more than one defining occurrence is found in Step 3 which matches the given occurrence, the following order of priorities is understood. If one of these defining occurrences is a terminal production of 'MABEL identifier', then this one is chosen; otherwise, the matching defining occurrence whose prefix contains the greatest number of characters is chosen.

7.4.3 Local Declarations

An identity-declaration commencing with a local-symbol is considered to be copied into any reach contained within the reach of the given identity-declaration. This copying process continues in a recursive fashion until the limit of nested reaches is encountered. If, as a result of this process, any reach contains more than one defining occurrence of a given terminal production of 'MABEL classifier' (of which, by the context conditions, no more than one may occur in a declaration which was not copied by this process from an outer reach) then two cases may exist. If the declaration which is actually within this reach also contains a local-symbol, then the copying process is discontinued for this reach and all reaches internal to it; otherwise, the copy within this reach is suppressed, but the copying process is continued within reaches internal to this reach.

Westinghouse Electric Corporation,
Churchill Boro,
Pittsburgh, Pa. 15235
(Received April 1969)