AB29.3.3 Rounding Control and the Algorithmic Language ALGOL 68

N.Apostolatos, H. Christ , H.Santo,
H. Wippermann

In the Draft Report on the algorithmic language ALGOL 68 (Wij 68) among the standard declarations we find the declaration of the operations on real operands in a rather ''soft'' way. It is impossible to perform error controlled numerical analysis since deviations from the exact values of unspecified direction may occur with the application of

1) operators on real operands,
2) real denotations and
3) real patterns.

Therefore we would prefer a definition, which specifies the result of the arithmetic operations in a more strict sense. Only so we can really say, what is the result of the elaboration of a phrase containing real operations or at least we can get lower and upper bounds for this result. In the remaining part of this paper we want to specify a real arithmetic in ALGOL 68 which satisfies all conditions, which might be posed now and in the future.

1. Operations on real Operands

Let $|R$ be the set of all real numbers and $|R_m$ the subset of real numbers of a considered length number in an implementation of ALGOL 68. It is assumed that zero is an element of $|R_m$ and that $|R_m$ is symmetric relative to zero, i.e. from $x \in |R_m$ it follows $-x \in |R_m$, $-x$ being the additive inverse of x in the sense of real analysis.

If $*$ stands for one of the operations $+,-,\times,:$ in the sense of real analysis, the corresponding operations $\underset{\vee}{*}$ and $\hat{*}$ with controlled rounding are defined for $a, b \in |R_m$, $(a * b) \in [\min |R_m ; \max |R_m]$:

$$a \underset{\vee}{*} b := \max \{ x \mid x \in |R_m,\ x \leq a * b \}$$
$$a \hat{*} b := \min \{ x \mid x \in |R_m,\ x \geq a * b \}$$

Taking in account the sign-rules for operations on real numbers yields the following equations [Chr 68] :

$$a \underset{\vee}{+} b = a \underset{\vee}{-} (-b)$$
$$a \hat{-} b = -(b \underset{\vee}{-} a)$$
$$a \hat{\times} b = -((-a) \underset{\vee}{\times} b)$$
$$a \hat{:} b = -((-a) \underset{\vee}{:} b)$$
$$a \hat{+} b = -((-a) \underset{\vee}{-} b)$$

Therefore only three of the eight operators need to be introduced explicitly. The remaining five can be defined within ALGOL 68 itself (Further reductions arenot possible since in general the multiplicative inverse does not exist in $|R_m$). Though one could start with another combination there are several reasons to introduce the operators $\underset{\vee}{-}$, $\underset{\vee}{\times}$, $\underset{\vee}{:}$.

All these considerations lead to an amendment-proposal concerning the standard operations on real operands ([Wij68], 10.2.3).

The following three operators of the rounding-down type are given (operations -, ×, : in the sense of real analysis):

```
op -v = (L real a,b) L real : c max{x | x ∈ |Rm, x ≤ a - b} c;
op ×v = (L real a,b) L real : c max{x | x ∈ |Rm, x ≤ a × b} c;
op :v = (L real a,b) L real : c max{x | x ∈ |Rm, x ≤ a : b} c
```

Since $|R_m$ is symmetric, the monadic operator - , yielding the additive inverse element may be defined as

```
op - = (L real a) L real : (LO -v a);
```

The addition-operator then is

```
op +v = (L real a,b) L real : (a -v - b);
```

The corresponding operators of the rounding-up type are

```
op -^ = (L real a,b) L real : (-(b -v a));
op ×^ = (L real a,b) L real : (-(-a ×v b));
op :^ = (L real a,b) L real : (-(-a :v b));
op +^ = (L real a,b) L real : (-(-a -v b));
```

Finally shortening-operators with rounding control have to be added too :

```
op downshort = (long L real a) L real : c max{ x | x ∈ |Rm, x ≤ a} c;
op upshort   = (long L real a) L real : (-downshort -a);
```

Based on these strictly defined new standard-operators the following operators are redefined:

```
op - = (L real a,b) L real : c one of the L real values 'a-vb', 'a-^b' c ;
op × = (L real a,b) L real : c one of the L real values 'a×vb', 'a×^b' c ;
op / = (L real a,b) L real : c one of the L real values 'a:vb', 'a:^b' c ;
op short = (Long L real a) L real : c one of the L real values
                                    'downshort a', 'upshort a' c ;
```

The reason for not giving a unique definition of the operators -, ×, / is to avoid restrictions on the design of floating point arithmetic.

2. Real denotations

In the Draft Report on ALGOL 68 [Wij 68] the a priori value of a real denotation is defined ([Wij 68],5.1.2.2). This a priori value is obtained by performing arithmetic operations on real operands. Since these operations are not strictly defined the a priori value is not strictly defined either. Therefore and in order to obtain a rounding controlled conversion process the following redefinitions are necessary (see [Wij 68], 5.1.2.1.a).

real denotation: rounding control option, proper real denotation.

proper real denotation: variable point numeral; floating point numeral.

rounding control:rounding down symbol; rounding up symbol.

The rounding control option indicates what kind of rounding must be applied to determine the value of a L - real - denotation. The value of a L - real denotation is obtained in the following steps (for L see [Wij 68],9.c.)

Step 1: The L - real-denotation is considered. The proper - real - - denotation contained in the considered L - real - denotation is called '' arithmetic - denotation'' and is considered. It is supposed, that all values have a length number such, that all operations may be performed without being undefined and at least corresponding the number of long - symbols contained in the considered L - real - denotation. If an an integral value is caused to become a real value, then the length number of the real value is thought to be such, that the application of the operator round ([Wij 68], 10,2,3.p) on this real value delivers an integral value equivalent to the original one. The value zero is called "exponent". Step 2 is taken.

Step 2: If the considered arithmetic - denotation contains no exponent - part then Step 3 is taken. The a priori value of the integral - denotation of the exponent - part is called "exponent". If the exponent - part contains a minus - symbol then "exponent" is subtracted from zero and the result is called "exponent". The exponent - part is discarded. Step 3 is taken.

Step 3: If the considered arithmetic-denotation contains a point-symbol, then the integral value, which is equivalent to the number of digit-tokens following the point-symbol is subtracted from "exponent" and called "exponent" and the point-symbol is discarded. If the integral-denotation contained in the arithmetic-denotation is preceded by a digit-zero-sequence, then this sequence is discarded. Step 4 is taken.

Step 4: The arithmetic-denotation now is an integral-denotation. Its a priori value is called "mantissa". Step 5 is taken.

Step 5: An integral value is constructed by multiplying the integral value one by ten as many times, as the absolute value ([Wij 68], 10.2.2.k) of "exponent" specifies. This integral value is then caused to become a real value, which is called "evaluated exponent". The integral value "mantissa" is caused to become a real value hence forth called "mantissa". Step 6 is taken.

Step 6: If the considered L-real-donotation contains no rounding-control (a rounding-down-symbol, a rounding-up-symbol) then the multiplying operator × (×↓ , ×↑), the dividing operator / (/↓ , /↑) and the shortening operator short (downshort, upshort)(see section 1) are considered. If "exponent" is not less zero ([Wij 68], 10.2.2.e) then a real value is obtained by performing the multiplication of "mantissa" and "evaluated exponent" applying the considered multiplying operator, otherwise a real value is obtained by performing the division of "mantissa" by "evaluated exponent" applying the considered dividing operator. The considered shortening operator is applied on this real value as many times as the difference of the length number of this real value and the considered L-real-denotation specifies.

3. Real patterns

For the same reasons as stated in the previous section the notion "real pattern" used in transput operations has to be redefined. (see [Wij 68], 5.5.1.2.a)

real pattern: direction control option, proper real pattern.

proper real pattern: sign mould option, real mould; floating point mould.

direction control: direct down; direct up.

direct down: letter w.

direct up: letter u.

The direction-control-option indicates what kind of rounding must be applied in transput processes.

A proper-real-pattern defines a set of "representations", which is considered; each representation is a string and possesses an a priori value. If the real-pattern contains a letter w (letter u) then on output a real value is edited into that representation of the considered set with the largest (smallest) a priori value not greater (not less) than this real value, on input a representation of the considered set is indited into a real value, which is the largest (smallest) real value not greater (not less) than the a priori value of this representation.

4. Interval structures and associated operations
(An example for the application of the proposed standard operators.)

Recently Interval Analysis (see e.g. [Mo 66]) has become more important. The efforts taken in this field at the University of Karlsruhe, Germany, lead to new theoretical knowledge, a lot of interval algorithms, and a new programming language for handling intervals (Ap 68). In ALGOL 68 according to the Draft Report [Wij 68] it is not possible to formulate interval algorithms since there are no

operators defined on interval structures. These operators will be described with the aid of the proposed standard operators.

A mode named "interval" may be created by the mode-declaration:

```
struct interval = (real low, real up) ;
```

The fieldselectors "low" and "up" stand mnemonically for "lower bound of interval" and "upper bound of interval".

Relational operators for intervals may be defined with the following operator declarations:

```
op <= (interval a, b) bool : up of a < low of b) c Interval
        a is said to be less than interval b, if the upper
        bound of a is less than the lower bound of b c;

op >= (interval a, b) bool : (low of a > up of b) ;
```

The monadic minus may be defined by:

```
op - = (interval a) interval : (- up of a, - low of a) ;
```

According to Moore [Mo 66] the diadic interval operations may be defined in the following way:

Be a an interval with lower bound a_1 and upper bound a_2, b an interval with lower bound b_1 and upper bound b_2, $*$ one of the symbols $+, -, \times, /$ then the arithmetic operations on intervals are defined by:

$$[a_1, a_2] * [b_1, b_2] = \{ x * y \mid a_1 \leq x \leq a_2,\ b_1 \leq y \leq b_2 \}$$

(in case of division, $0 \notin [b_1, b_2]$).

The diadic plus and minus in ALGOL 68 now are defined by:

```
op + = (interval a, b) interval : (low of a ⩛ low of b,
       up of a ⩓ up of b) ;

op - = (interval a, b) interval : (a + (-b));
```

The multiplication is defined by:

```
op × = (interval a, b) interval :
       (interval aa (a < 0 | - a | a), bb (b < 0 | - b | b) ;
        real uu (up of a ⩔ up of b), ll ;
        if low of aa ⩾ 0
           thef low of bb ⩾ 0
              then ll := low of aa ⩛ low of bb
              else ll := up of aa ⩛ low of bb
           elsf low of bb ⩾ 0
              then ll := low of aa ⩛ up of bb
              else
```

```
                (real hl (up of aa ⩛ low of bb),
                     hu (low of aa ⩓ low of bb);
                 ll := low of aa ⩛ up of bb ;
                 if hl < ll then ll := hl fi ;
                 if hu > uu then uu := hu fi )
        fi ;
        if (up of a < 0) = (up of b <  0)
            then interval (ll, uu)
            else interval (- uu, - ll)
        fi ) ;
```

The division may be defined as follows. (The label "interval overflow" refers to a part of program handling the case $0 \in [b_1;b_2]$.)

```
op / = (interval a, b) interval :
       (interval aa (a <  0 |  - a | a ),
                 bb (b < 0 |  interval overflow | b) ;
        real uu ( up of aa ⩓ low of bb ),
             ll ( low of aa  < 0 | low of aa ⩛ low of bb
                                 | low of aa ⩛ up of bb ) ;
        if (up of a < 0) = (up of b < 0)
           then interval (ll, uu)
           else interval (-uu, - ll)
        fi ) ;
```

Here is not the place to define further interval operators, for instance the very useful sign-and power-operator. It has been shown, that an interval-package may be easily incorporated in ALGOL 68, if however ALGOL 68 is extended to contain the proposed standard operators.

5. Examples

Suppose $|R_{m1}$ is the following set of floating-point numbers of length number 1

$$|R_{m1} = \{ m_{10}exp \mid m=-0.99(0.01)0.99, exp= 0, \mp1, \mp2, \ldots \}$$

$|R_{m2}$ one of length number 2

$$|R_{m2} = \{ m_{10}exp \mid m = -0.9999(0.0001)0.9999, exp = 0, \mp1, \mp2, \ldots \} ,$$

some special examples are given:

a) Standard operators

$$.89_{10}{}^{1} \text{ ⩛ } .55_{10}{}^{-5} = .88_{10}{}^{1}$$
$$.89_{10}{}^{1} \text{ ⩓ } .55_{10}{}^{-5} = .89_{10}{}^{1}$$
$$.89_{10}{}^{1} - .55_{10}{}^{-5} = .89_{10}{}^{1}$$

$$-.89_{10}{}^{1} \text{ ⩛ } -.55_{10}{}^{-5} = -.89_{10}{}^{1}$$
$$-.89_{10}{}^{1} \text{ ⩓ } -.55_{10}{}^{-5} = -.88_{10}{}^{1}$$
$$-.89_{10}{}^{1} - -.55_{10}{}^{-5} = -.89_{10}{}^{1}$$

downshort	.8425	=	.84
upshort	.8425	=	.85
short	.8425	=	.84

b) Real denotations

The $|R_{m1}$ a priori values of the real denotations

.8425, w .8425, u .8425

are resp ..84, .84, .85,

their $|R_{m2}$ a priori value is .8425 in all cases.

c) Real patterns

The $|R_{m2}$ value -.8425 is edited on output under control of

fw + .4df	as	-.8425
fu + .4df	as	-.8425
f + .4df	as	-.8425
fw + .2df	as	-.85
fu + .2df	as	-.84
f + .2df	as	-.84

The representation .842575 is indited under control of

fw + .6df	into	.8425
fu + .6df	into	.8426
f + .6df	into	.8426

which is a value in $|R_{m2}$.

d) Interval operators

$[.89_{10}1,.12_{10}2\,] + [.55_{10}-5,.75_{10}-4\,] = [\,.89_{10}1,.13_{10}2\,]$

$[\quad " \quad] - [\quad " \quad] = [\,.88_{10}1,.12_{10}2\,]$

$[\quad " \quad] \quad [\quad " \quad] = [\,.48_{10}-4,.90_{10}-3\,]$

$[\quad " \quad] / [\quad " \quad] = [\,.11_{10}6,.22_{10}7\,]$

Literature

[Ap68] Apostolatos, N. et al.:
The Algorithmic Language Triplex-ALGOL 60
Numerische Mathematik 11,pp.175-180(1968)

[Mo66] Moore, R.E.:
Interval Analysis
Englewood Cliffs, N.J. Prentice-Hall Inc.(1966)

[Wij 68] Wijngaarden, A.van et al.:
Draft Report on the Algorithmic Language ALGOL 68
Supplement to ALGOL Bulletin 26 (1968)

[Chr68] Christ,H.:
Realisierung einer Maschinenintervallarithmetik
auf beliebigen ALGOL 60-Compilern
Ber. d. Inst. f. Angew. Math., Rechenzentrum
Univ. Karlsruhe Jan. 1968.

Rechenzentrum, Universität Karlsruhe
Received July 1968