JTC1/SC22/WG5/N1253

Date: January 21, 1997
To: X3J3
From: Dick Hendrickson, R. Baker Kearfott
Subject: Interval Intrinsic Functions Operations
References: X3J3/96-065

General guidelines used are:

* Fortran intrinsic functions that accept REAL data should also accept INTERVAL data.

* All interval functions shall return enclosures of the range of values as real functions. That is, for a function f(x), the returned enclosure for f(X), where X = [a,b] is an interval, shall contain the set of values:

  {f(x} | a <= x <= b}

  this definition generalizes in a straightforward way to functions of n-variables.

* Those generic intrinsics that are REAL elemental functions should also operate as elemental functions with INTERVAL vector data.

  Note: The sharpness of the enclosures is not specified, but an ideal enclosure should be the smallest-width interval with machine numbers as endpoints that contains the actual range. Thus, the only accuracy requirement for interval versions of the intrinsics mandates that they contain the range of the corresponding mathematical function over the set of interval arguments.

* Array operations, such as "reshape," work just as they would on any other array.

Dick systematically went through F95 intrinsics, categorizing each intrinsic as follows:

def?	possibly does not apply
dna	does not apply to intervals
easy	straightforward, almost obvious, definition and computation
medium	harder than above to define and compute, but nothing really surprising
hard	hard to define and to compute
nc	no change from how F95 applies to structures

Baker, with analysis and other advice from the "interval" experts (cf. X3J3/95-065), modified Dick's original classifications. In such cases, comments appear in the right column. In particular, continuous monotonic functions are not classified as "hard," since they can be computed from appropriately rounded endpoint values; in turn, endpoint values can be obtained from existing floating point libraries if the accuracy of those libraries is known. An additional classification of:

tricky lots of edge and corner cases, otherwise easy

to replace Dick's "hard" has been added.

There is an issue to be resolved concerning what should be returned if part of the argument is in the domain of the function and part is not, e.g. SQRT((<-1,1>)). This can be disallowed, the range over the meaningful part of the argument can be returned, or some facility can perhaps be provided for the user to choose. Based on the discussion, my first recommendation is that the user somehow be given control over which of the above options is executed. If such user control is not possible, it is possibly slightly more desirable to merely return bounds on the range over the meaningful part of the domain; this scheme involves less runtime overhead because of consistency considerations, and possible surprise of the user. Furthermore, I also recommend that an exception be raised and NAN be returned if the argument lies entirely outside the domain of the function. However, in some cases, it would be more nearly correct. For example the range of SQRT over (<-1,1>) IS defined, but in the complex plane; an exception raised here would thus be analogous to an overflow.

For many of the functions listed below, examples and more extensive email discussion are available. There is a fair amount of consensus within the interval discussion group, (cf. X3J3/96-065), but I expect some additional discussion of several points, as indicated.

Section name	Category	Comments
13.11.1 present	nc	
13.11.2 abs	easy	Range of absolute value, rather than magnitude
13.11.2 aimag	dna	
13.11.2 aint	def?	Some questions: May the result be an interval of integers? Discussion and pictures are available.
13.11.2 anint	def?	Some questions, as with AINT.
13.11.2 ceiling	easy	
13.11.2 cmplx	easy	Use CMPLX(MID(X))
13.11.2 conjg	dna	
13.11.2 dble	easy	Use DBLE(MID(X))
13.11.2 dim	easy	
13.11.2 dprod	dna	
13.11.2 floor	easy	
13.11.2 int	easy	Use INT(MID(X))
13.11.2 max	easy	
13.11.2 min	easy	
13.11.2 mod	tricky	There are lots of cases. It turns out that they are like those needed for ATAN2. Nevertheless, it is perfectly well defined.
13.11.2 modulo	tricky	(see MOD)
13.11.2 nint	def?	Some questions, as with AINT.
13.11.2 real	easy	Use REAL(MID(X)).
13.11.2 sign	medium	If B>0, return ABS(A); if B<0, return -ABS(A); if 0.IN.B, return -ABS(A).CH.ABS(A). This is consistent with the definition that the result contain the range.
13.11.3 acos	medium	ACOS is monotonic.
13.11.3 asin	medium	ASIN is monotonic.
13.11.3 atan	medium	ATAN is monotonic.
13.11.3 atan2	tricky	There are a number of edge cases to be considered. But, in principle, it is not difficult. There is a question concerning what to do if the intervals cross a branch point. For example, if X=(<-1>) and Y=(<-.5,.5>), ATAN2(Y,X) could simply contain the interval [-pi,pi], since the arguments contain the branch point; alternately, may the range be extended to include values greater than pi or less than -pi? The latter leads to a more precise description of the range and hence to more meaningful computations in many instances.
13.11.3 cos	medium	The logic for handling inflection points of this function is well-known, compact, and publicly available free of charge.

```
13.11.3 cosh    medium
13.11.3 exp     medium
13.11.3 log     medium
13.11.3 log10   medium
13.11.3 sin     medium        See the comment for "cos"
13.11.3 sinh    medium
13.11.3 sqrt    medium
13.11.3 tan     medium        The main problem here is
                              representation of infinite values.
                              This could be handled the same
                              as the floating-point version,
                              that is, by returning an error.

                             Implementations should be *permited*
                             to use the IEEE -inf and +inf
                             values to correctly depict the
                             range.  For implementations that do not
                             have these values, there are a number of
                             processor dependent possabilities,
                             including:

                             a) Assign some values, such as the
                                largest and smallest real values
                                to used in place of -inf and +inf.

                             b) Return NaN or abort when intervals
                                cannot be returned that contain
                                the correct result.

                             In no case should a standard conforming
                             interval program be permitted to
                             return an interval result that does not
                             contain the correct answer.

13.11.3 tanh    medium

13.11.4 achar   dna
13.11.4 adjustl dna
13.11.4 adjustr dna
13.11.4 char    dna
13.11.4 iachar  dna
13.11.4 ichar   dna
13.11.4 index   dna
13.11.4 len_trim        dna
13.11.4 lge     dna
13.11.4 lgt     dna
13.11.4 lle     dna
13.11.4 llt     dna
13.11.4 repeat  dna
13.11.4 scan    dna
13.11.4 trim    dna
13.11.4 verify  dna

13.11.5 len_trim        dna

13.11.6 kind    nc
13.11.6 selected_int_kind       dna
13.11.6 selected_real_kind      nc

13.11.7 logical dna

13.11.8 digits  nc
13.11.8 epsilon nc
13.11.8 huge    nc
13.11.8 maxexponent nc
13.11.8 minexponent nc
13.11.8 precision   nc
13.11.8 radix   nc
13.11.8 range   nc
13.11.8 tiny    nc

13.11.9 bit_size dna

13.11.10 btest  dna
13.11.10 iand   dna
```

```
13.11.10 ibclr  dna
13.11.10 ibits  dna
13.11.10 ibset  dna
13.11.10 ieor   dna
13.11.10 ior    dna
13.11.10 ishft  dna
13.11.10 ishftc dna
13.11.10 not    dna

13.11.11 transfer     nc

13.11.12 exponent     dna
13.11.12 fraction     dna
13.11.12 nearest dna
13.11.12 rrspacing    dna
13.11.12 scale   dna
13.11.12 set_exponent dna
13.11.12 spacing dna

13.11.13 dot_product  nc
13.11.13 matmul  nc

13.11.14 all     dna
13.11.14 any     dna
13.11.14 count   dna
13.11.14 maxval  nc
13.11.14 minval  nc
13.11.14 product nc
13.11.14 sum     nc

13.11.15 allocated   nc
13.11.15 lbound  nc
13.11.15 shape   nc
13.11.15 size    nc
13.11.15 ubound  nc

13.11.16 merge   nc
13.11.16 pack    nc
13.11.16 spread  nc
13.11.16 unpack  nc

13.11.17 reshape nc

13.11.18 cshift  nc
13.11.18 eoshift nc
13.11.18 transpose   nc

13.11.19 maxloc  nc
13.11.19 minloc  nc

13.11.20 associated  nc
13.11.20 null    nc

13.11.21 cpu_time    dna
13.11.21 date_and_time dna
13.11.21 mvbits  dna
13.11.21 random_number def?     A possible definition is: When X is an
        interval, RANDOM_NUMBER(X) shall return an interval
        X = (<A, B>); where A and B are psudorandom numbers from a
        jointly uniform distributions in the intervals: [0, 1.0], with
        the restriction that A .LE. B . Both the conditional
        distribution of A|B and B|A are uniform.

        Discussion is still ongoing concerning what is
        meaningful to statisticians.

13.11.21 random_seed   nc
13.11.21 system_clock  dna
```