Subject: Re: IDL's handling of LOGICAL quantities (WHERE)
Posted by Liam Gumley on Tue, 12 Oct 1999 07:00:00 GMT
View Forum Message <> Reply to Message

James Tappin wrote:

```
>
>  \begin{rant}
>  I've finally decided to have a public moan about one of the weaknesses of IDL's
>  handling of logical operations: to boot -- that the WHERE function  follows
>  a C-like interpretation while most other things are  Fortran-like.
>
>  for example suppose we have an array (m)  some of whose values are NaN  then the
>  (inefficient) loop:
>  for j=0, n_elements(m) do if not finite(m(j)) then m(j)=0
>  will set all non-finite elements of m to 0.
>  However:
>  m(where(not finite(m))) = 0
>  will zero out the whole array since where sees (not 1) as a Yes.
>  [The correct solution is of course:
>  m(where(finite(m) ne 1)) = 0
>  ]
>
>  Or a simpler example:
>  IDL> a = [0, 1, 0, 1]
>  IDL> print, where(a eq 0)
>            0           2
>  IDL> print, where(not (a ne 0))
>            0           1           2           3
>
>  I guess the proper answer isto have  aproper  logical or boolean type and
>  functions like FINITE and logical operations should return it, and of course
>  WHERE should accept it.
```

I think it's useful to look at the output of NOT to understand what it's doing. For example,

```
IDL> print, not 0
      -1
IDL> print, not 1
      -2
```

This shows that NOT is a bitwise operator for integer operands, which sets each bit in the operand to it's complement. Funnily enough, if you use a float operand, the results are what you'd expect of a logical (rather than bitwise) operator, e.g.

```
IDL> print, not 0.0
      1.00000
```

```
IDL> print, not 1.0
      0.00000
```

To filter out non-finite values in an array, I'd use a function:

```
FUNCTION CHECK_FINITE, DATA, VALUE=VALUE

;- Check arguments
if n_params() ne 1 then message, 'Usage: RESULT = CHECK_FINITE(DATA)'
if n_elements(data) eq 0 then message, 'DATA is undefined'
if n_elements(value) eq 0 then value = 0.0

;- Set any non-finite elements to VALUE
index = where(finite(data) eq 0, count)
if count gt 0 then data[index] = value

;- Return the result
return, data

END
```

Example:

```
IDL> a = findgen(5)
IDL> a[0:1] = 1.0/0.0
% Program caused arithmetic error: Floating divide by 0
IDL> print, a
          Inf          Inf      2.00000      3.00000      4.00000
IDL> a = check_finite(a)
IDL> print, a
      0.00000      0.00000      2.00000      3.00000      4.00000
```

Cheers,
Liam.

--
Liam E. Gumley
Space Science and Engineering Center, UW-Madison
http://cimss.ssec.wisc.edu/~gumley