Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by steinhh on Mon, 01 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <36b57934.0@news.nwl.ac.uk> wmc@bas.ac.uk writes:
[...]
> But since this has come up, & its one of my pet peeves: why cannot where
> return a null array to indicate no-elements-match. And then array[null] would
> match to nothing. This would allow one to say
>
> array[where(wurble)]='stoat'
>
> instead of the ugly
>
> i=where(wurble,count)
> if (count gt 0) then array[i]='stoat'
>
> I use Perl a lot, and the contrast is very striking there: perl handles null
> values quite happily and it simplifies a lot of things.

I agree, it would be a nice feature... However, I have problems
seeing how to implement this, without altering e.g. the way IDL
allows you to index arrays with an array that is out of
bounds.. (try help,(findgen(10))(findgen(20)-5))

Maybe using (float/double) NaN values? Hmm. That could work!
Let's see,

```
array[NaN] = 5        ; Would be allowed, but does nothing
array[ [NaN,0] ] = 5  ; Assigns 5 to element 0 only.

data = array[NaN]       ; data is set to NaN !
data = array[ [NaN,0] ] ; Data is set to array[0]
```

The only problem I can see is that the indices will sometimes
have to be converted to floats/doubles in order to store the NaN
values, thus some overhead will occur...

> While I'm here: would RSI please put a decent regexp package into IDL?

Take a look at http://www.uio.no/~steinhh/IDL/additions.html for an
example of how to add it using dynamically loadable modules.

It requires a Unix flavor with regex.h, or alternatively the POSIX
compliant inteface for the GNU regular expression library (not
tested, though) - see
http://sunland.gsfc.nasa.gov/info/regex/Top.html, especially the
section "Programming with Regex".

Regards,

Stein Vidar