Subject: Re: reading long files
Posted by rdc on Sat, 06 Nov 1993 09:51:08 GMT
View Forum Message <> Reply to Message

A previous post mentions a two pass method of reading the file first
determining the necessary size of the array and then reading into it. For
people running on unix systems spawning a 'wc' will accomplish the first
pass faster than reading it with idl. Our wc returns # lines, # words, and
characters in the file so the array dimensions can be calculated by your
proceedure. (its more efficient to use a 1d array and REFORM it to 2d
after reading it) One drawback (other than portability) is when some
entries completely fill the available width of their column. This causes
two entries to run together and a free format read will lose its place.
But this situation can be flagged when #cols = #words / #lines doesn't
come out integer. This situation has to be handled as a special case,
otherwise a proceedure to do this can be kludged up to be pretty general.

Another aproach I have used a few times is to grow the array a record at a
time into a stattic array. This avoids both the wasteful use of memory and
also greatly speeds things up since the array isn't repeatedly being
recopied. Of course you have to predimension the array to be at least as
large as the file being read and you can have nightmares thinking about
ways you can overflowing it. Don't know if the options I used still work
with version 3.x, it was about 2.2 when I tried this. I dropped it in
favor of the safer two pass method.

--
Richard Clark
rclark@lpl.arizona.edu (or the address in the header)
For the 10,000th time: it's in Astronomical Algorithms by Jean Meeus!

Subject: Re: reading long files
Posted by knipp on Mon, 08 Nov 1993 10:51:21 GMT
View Forum Message <> Reply to Message

In article 9nn@organpipe.uug.arizona.edu, rdc@helium.gas.uug.arizona.edu (richard d clark)
writes:
>
> A previous post mentions a two pass method of reading the file first
> determining the necessary size of the array and then reading into it. For
> people running on unix systems spawning a 'wc' will accomplish the first
> pass faster than reading it with idl. Our wc returns # lines, # words, and
> # characters in the file so the array dimensions can be calculated by your
...
> Richard Clark
> rclark@lpl.arizona.edu (or the address in the header)
> For the 10,000th time: it's in Astronomical Algorithms by Jean Meeus!

Another way to count just the number of lines in a file (!UNDER UNIX)
is:

 awk "END {print NR}" >filename<

Use SPAWN and capture the output in a variable.

A more general way is to count the number of carr.returns in the file:

CR_BYTE = 10b   ; (p.e. for UNIX-Systems)

OPENR, UNIT, FILENAME, /GET_LUN
INFO = FSTAT(UNIT)
I = ASSOC(UNIT, BYTARR(INFO.SIZE))
POS = WHERE(I(0) EQ CR_BYTE)
FREE_LUN, UNIT

NOF_LINES = N_ELEMENTS(POS)

Karl

_____
_____
  __ ____ __
  / // _ \ / / Karlheinz Knipp   phone: +49 511 - 762 4922
 / // /_/ // /   University of Hannover   fax: +49 511 - 762 2483
 / // ____// /   Institute for Photogrammetry
/ // /   //   Nienburger Str.1
/_//_/   /_/   FRG 30167 Hannover 1   email: knipp@ipi.uni-hannover.de

---

Subject: Re: reading long files
Posted by knipp on Wed, 10 Nov 1993 08:54:53 GMT
View Forum Message <> Reply to Message

In article 12269@newsserver.rrzn.uni-hannover.de, knipp@ipi.uni-hannover.de (K Knipp) writes:
>
> A more general way is to count the number of carr.returns in the file:
>

```
> CR_BYTE = 10b    ; (p.e. for UNIX-Systems)
>
> OPENR, UNIT, FILENAME, /GET_LUN
> INFO = FSTAT(UNIT)
> I = ASSOC(UNIT, BYTARR(INFO.SIZE))
> POS = WHERE(I(0) EQ CR_BYTE)
> FREE_LUN, UNIT
>
> NOF_LINES = N_ELEMENTS(POS)
>
```

Sorry, but I mixed up LINEFEED and CARRIAGE RETURN. Nevertheless the code above is correct with ascii 10, which off course is LINEFEED under UNIX.

Karl

_____
_____

   __ ____ __
  / // _ \ / / Karlheinz Knipp   phone: +49 511 - 762 4922
 / // /_/ // /   University of Hannover   fax: +49 511 - 762 2483
 / // ____// /   Institute for Photogrammetry
/ // /   / /   Nienburger Str.1
/_//_/   /_/   FRG 30167 Hannover 1   email: knipp@ipi.uni-hannover.de