Subject: Re: Read Total lines in an ASCII file
Posted by Mark Hadfield on Sun, 15 Dec 2002 21:09:52 GMT
View Forum Message <> Reply to Message

"Med Bennett" <no.spam@this.address.please> wrote in message news:3DFA296F.65A8E1A0@this.address.please...

[IDL code for counting lines in code omitted]

> That should work fine, but I would think that it would be slow for
> large files because of the loop. I do this instead - also somewhat
> crude but maybe faster:
>
> junk=strarr(1000000L)
> on_ioerror,done
> openr,lun,'mytextfile.txt'
> readf,lun,junk
> stop,'need to increase array size'
> done:
> close,lun
> junk = junk[where(strlen(junk) gt 0)]
> end

Hmmm. This doesn't actually determine the number of lines in the file. It reads & returns all the non-empty lines.

Of course, when people ask for a way to count the lines in a file, usually what they want to do next is to read the file contents. Addressing the problem, "how do I read all the data from an ASCII file with an unknown number of lines?". The first thing to do is to go to David's site and see what he says. I found this

http://www.dfanning.com/tips/unknown_rows.html

It points to some useful routines, but doesn't really discuss the general approaches. I can think of three:

1 - Pre-allocate an array big enough to hold the maximum expected amount of data, read the data into it, then trim the array.

2 - Read the file once to count lines, allocate a data array of exactly the right size, then read the file again to store the data.

3 - Read the file once, storing the data in an extensible data structure, then (optionally) copy the data out of the extensible structure into an array.

No. 3 is the most flexible and arguably the most aesthetically

pleasing, but unfortunately you will have to write the "extensible data structure" yourself (or use someone else's), since IDL doesn't have anything suitable. IDL arrays *look* like they can be extended, but in fact every time you extend an IDL array you create a new one.

No. 2 seems very inefficient, but with disk caching it often turns out that reading a file twice doesn't take much longer than reading it once.

No. 1 (as Med proposes) is probably the fastest, but has the disadvantage of a built-in hard limit that will bite you when you are reading *really* big files. To some extent the choice between them will depend on what it is that you want to read out of each line. Will you want to skip any lines?

I have done some comparisons in the past and will dig them out if I can.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)