Subject: Re: Read Total lines in an ASCII file
Posted by Mark Hadfield on Tue, 17 Dec 2002 23:22:19 GMT
View Forum Message <> Reply to Message

<wmc@bas.ac.uk> wrote in message news:3dff99e5@news.nwl.ac.uk...
> Mark Hadfield <m.hadfield@niwa.co.nz> wrote:
>> Method 1, the two-pass method, is
>> surprisingly quick here, but suffers if the file is on a slow network
>> drive.
>
> If the read and re-read are close in time (as they should be) the file
> will still be in whatever cache the o/s uses, and the second read (as
> far as the o/s is concerned) will be much faster.

Yes. Actually, my test penalises the two-pass method relative to all the other one-pass methods, because the file is created just before it is read, so is already in cache.

> This may well survive being on a network drive

Depends on the network protocol and parameters. My network connection doesn't seem to do read caching very well.

> ps: did you try method 5, ie spawn wc -l, then use method 0?

I think you mean my method 1 (ie two passes: read the file once to count the lines, create the result array, then read the file again to get the data into the array).

So I revisited method 1, comparing three ways of counting the lines in the file:

1a - Count lines with IDL readf statements in a while loop
1b - Count lines by spawning "wc -l"
1c - Count lines with IDL 5.6 FILE_LINES function

and here are the times taken to read the same 20,000-line, uncompressed file on my hard drive

1a 0.24 s
1b 0.32 s
1c 0.09 s

FILE_LINES is the clear winner. (Isn't it a pity it doesn't accept a COMPRESS keyword!)

Spawning "wc -l" is the slowest. Note that this is on Windows 2000 with the Cygwin "wc" command. Unix is much faster at spawning subprocesses than

Windows, so method 1b may be competitive there.

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