Subject: the last line of a large file
Posted by queiny on Thu, 09 Aug 2007 19:49:04 GMT
View Forum Message <> Reply to Message

Hi,

I am processing a large file with data in the same format till the last line. In the last line, it states how many records are included in this file.

So the structure of my program is:

```
while not eof(in_unit) do begin
   readf, in_unit, input_line
    if( input_line ne 'last line) then begin
      .....
    else
      ....
    endif
endwhile
```

Do I have to use 'if/then' to test whether every input_line is the last line of the file? Since there are many data records in the file, repeat calls to 'if/then' can be time consuming. But if I don't do the test, the program will be halted when it read in the last line.

A Easy way I can think of is to delete the last line, but sometime we are not supposed to change the input files.

Subject: Re: the last line of a large file
Posted by Conor on Fri, 10 Aug 2007 15:51:17 GMT
View Forum Message <> Reply to Message

On Aug 10, 11:00 am, Carsten Lechte <c...@toppoint.de> wrote:
> Conor wrote:
>> lol! Really! What in the world is the point of putting the number of
>> lines at the end of the file?
>
> One legitimate reason would be that sometimes you only know how much
> data you have until after you have processed it all, especially if the
> data sets are so large that you only ever have a small subset in RAM.
>
> A legitimate example are zip archives, where the table of contents is
> written to the end of the file, because the the compressed sizes of
> the archive members cannot be known in advance, and it would double
> the running time to determine the compressed size beforehand, it would
> furthermore use twice the disk space to re-write the file with the

> contents in front, it would be impossible to keep the whole archive
> in RAM before writing it, and finally, one may not be able leave space
> for the contents table at the beginning of the file, to be filled in
> later, because one would have to know how long the table will be
> beforehand...
>
> Of course, this does not mean that the original poster's data has a
> legitimate reason for being organised like this.
>
> For the original poster's problem, one idea is to get the file size
> in bytes, skip to position file_size-1000, read that small chunk and
> parse it for the desired metadata. This might even be faster than
> actually counting the lines with FILE_LINES, but it is probably only
> worth it if the metadata contains more useful information that just
> the number of lines in the file.
>
> chl

As an actual suggestion, if the file_lines dosesn't take too long you can always just count the number of lines and break down the file into manageable chunks. Imagine for a moment that the following file has 1,000,000 lines and your computer can only make arrays with 10,000 rows at a time (which you would know in advanced). You might do something like this:

```
max_size = 10000
num_rows = file_lines(file)   ; 1,000,000
num_parts = num_rows/max_size ; 10 parts
num_cols = 10

data = fltarr(max_size,numcols)

for i=0,num_parts-1 do begin
readf,lun,data
; do something with data, then read the next chunk
endfor
```

There's a couple other things you can do. For starters, if you don't already know it you can calucluate the number of columns in the file by reading in the first line, using strsplit, and then rewinding the file to the beginning. Also, I haven't included it in the above code, but you'll have to keep track of the last line still. In this case what you would probably do is calculate how many lines you want to read in the last chunk of data, and worry about it then. For instance, imagine the same example but now the line has 75,000 lines and you don't want to read the last one:

```
max_size = 10000
```

```
num_rows = file_lines(file)   ; 75,000
num_parts = ceil(num_rows/max_size) ; 8 parts
num_cols = 10
last_read = max_size - (num_parts*max_size - num_rows) - 1 ; 4999

data = fltarr(max_size,numcols)

for i=0,num_parts-1 do begin
if i eq num_parts-1 then begin
    readf,lun,data
endif else begin
    data = fltarr(last_read,numcols)
    readf,lun,data
endelse
; do something with data, then read the next chunk
endfor
```

Not exactly elegant, but it should work for your problem.

Subject: Re: the last line of a large file
Posted by James Kuyper on Fri, 10 Aug 2007 17:39:51 GMT
View Forum Message <> Reply to Message

On Aug 10, 11:00 am, Carsten Lechte <c...@toppoint.de> wrote:
> Conor wrote:
>> lol! Really! What in the world is the point of putting the number of
>> lines at the end of the file?
>
> One legitimate reason would be that sometimes you only know how much
> data you have until after you have processed it all, especially if the
> data sets are so large that you only ever have a small subset in RAM.

That's easily worked around. Write a dummy length at the beginning of the file, process data until you reach the end of the file, then go back to the beginning of the file are overwrite the dummy length with the correct one. For ASCII format files, that requires being very careful to make sure that the dummy length is a sufficiently long string, and that the final length is padded with spaces (or leading 0s) to have exactly the same size.

There are obvious limitations to this technique (i.e. the output has to be seekable); but not seem applicable in this context.

Subject: Re: the last line of a large file
Posted by little davey on Fri, 10 Aug 2007 21:15:04 GMT

On Aug 9, 2:49 pm, queiny <quein...@yahoo.com> wrote:
> Hi,
>
> I am processing a large file with data in the same format till the
> last line. In the last line, it states how many records are included
> in this file.
>
> So the structure of my program is:
>
> while not eof(in_unit) do begin
> readf, in_unit, input_line
> if(input_line ne 'last line) then begin
> .....
> else
> ....
> endif
> endwhile
>
> Do I have to use 'if/then' to test whether every input_line is the
> last line of the file? Since there are many data records in the file,
> repeat calls to 'if/then' can be time consuming. But if I don't do the
> test, the program will be halted when it read in the last line.
>
> A Easy way I can think of is to delete the last line, but sometime we
> are not supposed to change the input files.

Would it be faster to use ON_IOERROR? One could process the "normal"
cases of reading the input lines. When there is an IO_ERROR, you trap
it and close the file. Naturally if there is a real IO_ERROR, you'd
have to test for a "real" input error, or the end of the file. But
that would mean IF statements executed only if there were IO_ERRORS.

-- Little Davey --

Subject: Re: the last line of a large file
Posted by queiny on Tue, 14 Aug 2007 17:22:52 GMT

Thank you all for the valuable suggestions.

I just found out that 'file_lines' and 'do loop' not always working.

IDL> Loop limit expression too large for loop variable type.
 <LONG64 (3548257)>.

And thank Connor for suggestions about trunk data to smaller pieces
and only worry about the last line problem one. I think that would be
what I will do.

Subject: Re: the last line of a large file
Posted by David Fanning on Tue, 14 Aug 2007 17:42:46 GMT
View Forum Message <> Reply to Message

queiny writes:

> Thank you all for the valuable suggestions.
>
> I just found out that 'file_lines' and 'do loop' not always working.
>
> IDL> Loop limit expression too large for loop variable type.
> <LONG64 (3548257)>.

As long as you are learning new things today, look up
COMPILE_OPT defint32. You will be glad you did. :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
