Subject: Re: Fast editing of text file?
Posted by Michael Wallace on Tue, 14 Jun 2005 23:26:38 GMT

View Forum Message <> Reply to Message

You can divide your file into two files: file with everything except the 12th line, file with only the 12th line. When you need to update the 12th line, you won't have to recreate the original file. You'll only need to update the little file with the 12th line in it. That should be quite a bit faster than having to recreate the original file each time.

-Mike

t_314159@yahoo.com wrote:

> Hi,
> I have a text file that I must access many times during program
> execution (1000's of times, really!) and I *always* only need to change
> the the text on the 12th line of the file. My question, how to
> efficiently access and alter the text of ONLY the 12th line without
> altering even a single space or comma on any of the rest of the lines?
> And did I mention fast since i've got to loop through this many times?
>
> For example:
>
> --BEGIN FILE---
> test file
> A
> 0   1 0
> 4,5,1
> [some text lines I'm not showing here, I'll show my 12th line next]
> 500, 0.02, 1.00587
> [more text lines to the end of file]
> --END FILE---
>
> What I need to do is keep the file exactly as is (there's some weird
> formatting of spaces, tabs, commas on different lines that a follow-on
> old fortran program expects) EXCEPT for the 12th line where I need to
> change those 3 #'s every time (I know that line is expected to be
> comma+space delimeted).
>
> The only thing I can think of is to:
> openr, infile, inFileName, /get_lun;
> create a new outfile for write;
> loop through infile line by line writing directly to outfile up to, but
> not including, the 12th line;
> do a newline = strtrim(a,2) + ", " + strtrim(b,2) + ", " + strtrim(c,2)
> where a, b, and c are int,float,or double values I need to write on
> line 12;

> writeu, outfile, newline
> continue looping through infile writing lines 13 to EOF to outfile;
> free_lun both infile and outfile;
> delete the infile;
> rename the outfile to the infile's old name;
> call the model that works on that text file;
> repeat 40,000 times with new values for a, b, c each time;
>
> Is there a better way? Note that significant digits of a, b, c may
> change each time so the formatting and # of places each takes will vary
> each time. Next time the 3 numbers may be .3, 1000.00003, 0.
>
> Thanks you for the help!!!
>

Subject: Re: Fast editing of text file?
Posted by t_314159 on Wed, 15 Jun 2005 00:15:34 GMT
View Forum Message <> Reply to Message

Would that save time? Actually, I think this method would add even more
steps. I still would have to merge them back together each time with
something along my example because the fortran program that'll *use*
the text file (executable, no access to the fortran code so can't alter
in there) requires this single text file. Maybe I'm not following your
logic...

Subject: Re: Fast editing of text file?
Posted by Michael Wallace on Wed, 15 Jun 2005 04:22:58 GMT
View Forum Message <> Reply to Message

t_314159@yahoo.com wrote:
> Would that save time? Actually, I think this method would add even more
> steps. I still would have to merge them back together each time with
> something along my example because the fortran program that'll *use*
> the text file (executable, no access to the fortran code so can't alter
> in there) requires this single text file. Maybe I'm not following your
> logic...

You're following my logic, but until now I don't believe you had said
that you couldn't alter the code that'd process the file. I had assumed
that you could edit that code and it'd just be a quick change to read
from two files instead of one.

-Mike

Subject: Re: Fast editing of text file?
Posted by marc schellens[1] on Wed, 15 Jun 2005 07:55:31 GMT

View Forum Message <> Reply to Message

As you describe it you always use the same file for your fortran
program.

Find out the file position of the 12th line (read or write it till
then,
then use GET_LUN)

When you open it again you use after opening POINT_LUN.
Then you write your 12th line (and all following lines - see below for
options).

You don't need to write the following lines, if your fortran program
can handle trailing (leading) spaces. In that case write the file the
first time with
the maximum possible line length.

Then use POINT_LUN and write a string (preceded) padded with as many '
' as
needed to fill up to the maximum line length.

Or if your fortran program cannot handle spaces format all numbers to
the maximum possible amount of digits.
(ie. write 1.100000 instead of 1.1, etc (use the FORMAT keyword to
write))

If this isn't possible either read the trailing strings once into a
string array and
write this out.
WRITE,lun,strarr

Cheers,
marc

Subject: Re: Fast editing of text file?
Posted by Andrew Cool on Thu, 16 Jun 2005 02:55:51 GMT

View Forum Message <> Reply to Message

t_314159@yahoo.com wrote:
> Hi,
> I have a text file that I must access many times during program
> execution (1000's of times, really!) and I *always* only need to change
> the the text on the 12th line of the file. My question, how to
> efficiently access and alter the text of ONLY the 12th line without

> altering even a single space or comma on any of the rest of the lines?
> And did I mention fast since i've got to loop through this many times?

If lines[0:10] and lines[12:*] don't change, why not hold those
permanently
in memory as two string arrays, array_top and array_bottom.

Concoct Line 12 with your 3 new numbers,
and then printf to a new version of the file :-

```
printf,outlun, array_top      ; lines 1..11
printf,outlun, line12         ; line 12
printf,outlun, array_bottom   ; lines 13..end of file
```

This way you've only got 1 read, at the very start of your program, and
then 1 write per change of numbers.

Sounds easy.

Andrew C.

Subject: Re: Fast editing of text file?
Posted by Mark Hadfield on Thu, 16 Jun 2005 03:28:04 GMT

View Forum Message <> Reply to Message

andrew.cool@dsto.defence.gov.au wrote:
>
> If lines[0:10] and lines[12:*] don't change, why not hold those
> permanently
> in memory as two string arrays, array_top and array_bottom.
>
> Concoct Line 12 with your 3 new numbers,
> and then printf to a new version of the file :-
>
> printf,outlun, array_top   ; lines 1..11
> printf,outlun, line12   ; line 12
> printf,outlun, array_bottom   ; lines 13..end of file
>
> This way you've only got 1 read, at the very start of your program, and
> then 1 write per change of numbers.

It might be that the concatenation of these three components could be
done more quickly by a system command like Unix "cat". In other words
you would have file_top and file_bottom containing the unchanging bits;
IDl would write a new version of file_middle each time, then

```
   spawn, "cat file_top file_middle file_bottom > file"
```

On the other hand it could be slower...

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

Subject: Re: Fast editing of text file?

Posted by Michael Wallace on Thu, 16 Jun 2005 08:05:40 GMT

View Forum Message <> Reply to Message

> It might be that the concatenation of these three components could be
> done more quickly by a system command like Unix "cat". In other words
> you would have file_top and file_bottom containing the unchanging bits;
> IDl would write a new version of file_middle each time, then
>
>   spawn, "cat file_top file_middle file_bottom > file"
>
> On the other hand it could be slower...

This seems to be pretty fast. I created a three files: file of first 11 lines, file of the 12th line and another file for all the remaining lines. I made the last file 150,000 lines long just for the fun of it.

```
$ wc -l upper.txt middle.txt lower.txt
      11 upper.txt
       1 middle.txt
  150000 lower.txt
  150012 total
$ date && cat upper.txt middle.txt lower.txt > test.txt; date
Thu Jun 16 02:56:29 CDT 2005
Thu Jun 16 02:56:29 CDT 2005
$ wc -l test.txt
  150012 test.txt
```

So, doing the concatenation took under a second. My machine is good, but not blazingly fast.

-Mike

Subject: Re: Fast editing of text file?

Posted by Craig Markwardt on Thu, 16 Jun 2005 14:51:41 GMT

View Forum Message <> Reply to Message

t_314159@yahoo.com writes:
> Hi,
> I have a text file that I must access many times during program
> execution (1000's of times, really!) and I *always* only need to change
> the the text on the 12th line of the file. My question, how to
> efficiently access and alter the text of ONLY the 12th line without
> altering even a single space or comma on any of the rest of the lines?
> And did I mention fast since i've got to loop through this many times?

If you can keep each line a fixed record size, then it might be easiest to write lines 1-11, use POINT_LUN to determine the file pointer position, and then write the rest of the file.

After that, you can open the file in update mode, (OPENU), and POINT_LUN directly to the position of line 12, rewrite it, and proceed. However, this procedure only works if line 12 has a fixed record size.

Good luck,
Craig

--
 -- --------------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@REMOVEcow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 -- --------------

Subject: Re: Fast editing of text file?
Posted by R.Bauer on Sun, 19 Jun 2005 08:12:48 GMT

View Forum Message <> Reply to Message

t_314159@yahoo.com wrote:

Hi, nameless

does it really save time to proceed this way. On which timescale we are talking?

If I understand right I would read in the whole file at once in a string array. You could determine the file line length by file_lines(). Then I would keep this array in memory. I would alter index 11 and then I would write the whole file at once.

This needs only seconds.

```
PRO example

  file='test.dat'
  txt=MAKE_ARRAY(file_lines(file),/STRING)
  OPENR,lun,file,/get_lun
  READF,lun,txt
  FREE_LUN,lun

  a=systime(1)
  FOR I=0L,40000L DO BEGIN
    txt[11]=STRTRIM(STRING(i),2)

    OPENW,lun,file,/GET_LUN,width=200
    PRINTF,lun,txt
    FREE_LUN,lun

  ENDFOR
  print,systime(1)-a
END
```

> Hi,
> I have a text file that I must access many times during program
> execution (1000's of times, really!) and I *always* only need to change
> the the text on the 12th line of the file. My question, how to
> efficiently access and alter the text of ONLY the 12th line without
> altering even a single space or comma on any of the rest of the lines?
> And did I mention fast since i've got to loop through this many times?
>
> For example:
>
> --BEGIN FILE---
> test file
> A
> 0   1 0
> 4,5,1
> [some text lines I'm not showing here, I'll show my 12th line next]
> 500, 0.02, 1.00587
> [more text lines to the end of file]
> --END FILE---
>
> What I need to do is keep the file exactly as is (there's some weird
> formatting of spaces, tabs, commas on different lines that a follow-on
> old fortran program expects) EXCEPT for the 12th line where I need to
> change those 3 #'s every time (I know that line is expected to be
> comma+space delimeted).
>
> The only thing I can think of is to:

> openr, infile, inFileName, /get_lun;
> create a new outfile for write;
> loop through infile line by line writing directly to outfile up to, but
> not including, the 12th line;
> do a newline = strtrim(a,2) + ", " + strtrim(b,2) + ", " + strtrim(c,2)
> where a, b, and c are int,float,or double values I need to write on
> line 12;
> writeu, outfile, newline
> continue looping through infile writing lines 13 to EOF to outfile;
> free_lun both infile and outfile;
> delete the infile;
> rename the outfile to the infile's old name;
> call the model that works on that text file;
> repeat 40,000 times with new values for a, b, c each time;
>
> Is there a better way? Note that significant digits of a, b, c may
> change each time so the formatting and # of places each takes will vary
> each time. Next time the 3 numbers may be .3, 1000.00003, 0.
>
> Thanks you for the help!!!

--
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de
http://www.fz-juelich.de/icg/icg-i/
=== ======
a IDL library at ForschungsZentrum Juelich
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro. html