Subject: Very tricky NetCDF "bug"
Posted by Fabzi on Fri, 14 Dec 2012 18:08:53 GMT
View Forum Message <> Reply to Message

Hi IDLers,

I've lost (again) a bunch of hours and also got older faster than usual
because of exeptionally high stress level this week.

In our group, we are handling/creating a lot of large NCDF files. A lot.
Of very large files. The NetCDF4 format was a very welcome improvement
thanks to the compression options and the possibility to create large
files (> 2Gb).

However, after I made some apparently harmless changes to some routines,
our post-processing tool just made IDL VERY slow. So slow, that you
could think it froze (even if the process still used 100% processor). It
took me days to find out where the error was, and I suspected IDL much
later than other factors (broken hard drive, failure in the network,
corrupt files...).

Anyway. To make things short, the problem is coming from the "STRIDE"
keyword, which is said to be set by default to "[1,1,...]" etc. My NCDF
object, for some reason, changed lately with STRIDE set to "[1,1,...]"
by default, too (so kind of replacing the IDL default). And this was the
reason of non-failing but extremely slow NCDF_VARGET calls...

The programm below shows what I mean. (be carefull, it creates a test.nc
file of about ~ 600 Mb in your working directory).

Three interesting things are shown:
- NETCDF4 is about twice faster then NETCDF3 to read a variable (which
is good)
- NETCDF3 is about 20% slower when you set STRIDE to its default value
by yourself instead of letting IDL do it alone
- NETCDF4 is about 9 times slower with the same test!

I spare you the results with much larger AND compressed files.

I one of you could be so nice and reproduce my results? If you see the
same I'll make a bug report...

Thanks!

Fab

```
pro test_ncdf_strides, NETCDF4_FORMAT=netcdf4_format
```

```
    ; This creates a ~ 600 MB NCDF file
    fid = NCDF_CREATE('test.nc', /CLOBBER, NETCDF4_FORMAT=netcdf4_format)
    dim1_id = NCDF_DIMDEF(fid, 'dim1', 200)
    dim2_id = NCDF_DIMDEF(fid, 'dim2', 200)
    dim3_id = NCDF_DIMDEF(fid, 'dim3', 27)
    dim4_id = NCDF_DIMDEF(fid, 'dim4', 24*3)
    v1_id = NCDF_VARDEF(fid, 'var1', [dim1_id, dim2_id, dim3_id,
dim4_id], /FLOAT)
    v2_id = NCDF_VARDEF(fid, 'var2', [dim1_id, dim2_id, dim3_id,
dim4_id], /FLOAT)
    dummy = FLTARR(200, 200, 27, 24*3)
    NCDF_VARPUT, fid, v1_id, dummy
    NCDF_VARPUT, fid, v2_id, dummy
    NCDF_CLOSE, fid

    ; open and read
    fid = NCDF_OPEN('test.nc', /NOWRITE)
    ; Read
    logt0 = SYSTIME(/SECONDS)
    NCDF_VARGET, fid, 'var1', var
    print, 'Get var1         :' + STRING(SYSTIME(/SECONDS)-logt0)
    var = 0. ; free memory
    logt0 = SYSTIME(/SECONDS)
    NCDF_VARGET, fid, 'var2', var, STRIDE=[1,1,1,1]
    print, 'Get var2 (STRIDE):' + STRING(SYSTIME(/SECONDS)-logt0)
    var = 0. ; free memory

    ; Just to be sure it's not some kind of IDL cache stuff
    logt0 = SYSTIME(/SECONDS)
    NCDF_VARGET, fid, 'var1', var, STRIDE=[1,1,1,1]
    print, 'Get var1 (STRIDE):' + STRING(SYSTIME(/SECONDS)-logt0)
    var = 0. ; free memory
    logt0 = SYSTIME(/SECONDS)
    NCDF_VARGET, fid, 'var2', var
    print, 'Get var2         :' + STRING(SYSTIME(/SECONDS)-logt0)
    var = 0. ; free memory
    NCDF_CLOSE, fid

end
```

Results:

```
IDL> .FULL_RESET_SESSION
IDL> print, !VERSION
{ x86_64 linux unix linux 8.2.1 Aug 20 2012      64      64}
IDL> test_ncdf_strides
```

```
% Compiled module: TEST_NCDF_STRIDES.
% Loaded DLM: NCDF.
Get var1         :      0.97150087
Get var2 (STRIDE):       1.1921642
Get var1 (STRIDE):       1.1618340
Get var2         :      0.93627405
IDL> test_ncdf_strides, /NETCDF4_FORMAT
Get var1         :      0.54904699
Get var2 (STRIDE):       4.4865382
Get var1 (STRIDE):       4.4508131
Get var2         :      0.53942299
IDL> test_ncdf_strides, /NETCDF4_FORMAT
Get var1         :      0.54891205
Get var2 (STRIDE):       4.4209800
Get var1 (STRIDE):       4.3824911
Get var2         :      0.53269601
IDL> test_ncdf_strides
Get var1         :      0.96190310
Get var2 (STRIDE):       1.1909840
Get var1 (STRIDE):       1.1693609
Get var2         :      0.93884301
```

Subject: Re: Very tricky NetCDF "bug"
Posted by chris_torrence@NOSPAM on Sat, 15 Dec 2012 05:22:48 GMT

View Forum Message <> Reply to Message

On Friday, December 14, 2012 11:08:53 AM UTC-7, Fab wrote:
> Hi IDLers,
>
>
>
> I've lost (again) a bunch of hours and also got older faster than usual
>
> because of exeptionally high stress level this week.
>
>
>
> In our group, we are handling/creating a lot of large NCDF files. A lot.
>
> Of very large files. The NetCDF4 format was a very welcome improvement
>
> thanks to the compression options and the possibility to create large
>
> files (> 2Gb).
>
>
>

> However, after I made some apparently harmless changes to some routines,
>
> our post-processing tool just made IDL VERY slow. So slow, that you
>
> could think it froze (even if the process still used 100% processor). It
>
> took me days to find out where the error was, and I suspected IDL much
>
> later than other factors (broken hard drive, failure in the network,
>
> corrupt files...).
>
>
>
> Anyway. To make things short, the problem is coming from the "STRIDE"
>
> keyword, which is said to be set by default to "[1,1,...]" etc. My NCDF
>
> object, for some reason, changed lately with STRIDE set to "[1,1,...]"
>
> by default, too (so kind of replacing the IDL default). And this was the
>
> reason of non-failing but extremely slow NCDF_VARGET calls...
>
>
>
> The programm below shows what I mean. (be carefull, it creates a test.nc
>
> file of about ~ 600 Mb in your working directory).
>
>
>
> Three interesting things are shown:
>
> - NETCDF4 is about twice faster then NETCDF3 to read a variable (which
>
> is good)
>
> - NETCDF3 is about 20% slower when you set STRIDE to its default value
>
> by yourself instead of letting IDL do it alone
>
> - NETCDF4 is about 9 times slower with the same test!
>
>
>
> I spare you the results with much larger AND compressed files.
>

>
>
> I one of you could be so nice and reproduce my results? If you see the
>
> same I'll make a bug report...
>
>
>
> Thanks!
>
>
>
> Fab
>
>
>
>
>
> pro test_ncdf_strides, NETCDF4_FORMAT=netcdf4_format
>
>
>
>   ; This creates a ~ 600 MB NCDF file
>
>   fid = NCDF_CREATE('test.nc', /CLOBBER, NETCDF4_FORMAT=netcdf4_format)
>
>   dim1_id = NCDF_DIMDEF(fid, 'dim1', 200)
>
>   dim2_id = NCDF_DIMDEF(fid, 'dim2', 200)
>
>   dim3_id = NCDF_DIMDEF(fid, 'dim3', 27)
>
>   dim4_id = NCDF_DIMDEF(fid, 'dim4', 24*3)
>
>   v1_id = NCDF_VARDEF(fid, 'var1', [dim1_id, dim2_id, dim3_id,
>
> dim4_id], /FLOAT)
>
>   v2_id = NCDF_VARDEF(fid, 'var2', [dim1_id, dim2_id, dim3_id,
>
> dim4_id], /FLOAT)
>
>   dummy = FLTARR(200, 200, 27, 24*3)
>
>   NCDF_VARPUT, fid, v1_id, dummy
>
>   NCDF_VARPUT, fid, v2_id, dummy
>

```
>	NCDF_CLOSE, fid
>
>
>
>	; open and read
>
>	fid = NCDF_OPEN('test.nc', /NOWRITE)
>
>	; Read
>
>	logt0 = SYSTIME(/SECONDS)
>
>	NCDF_VARGET, fid, 'var1', var
>
>	print, 'Get var1          :' + STRING(SYSTIME(/SECONDS)-logt0)
>
>	var = 0. ; free memory
>
>	logt0 = SYSTIME(/SECONDS)
>
>	NCDF_VARGET, fid, 'var2', var, STRIDE=[1,1,1,1]
>
>	print, 'Get var2 (STRIDE):' + STRING(SYSTIME(/SECONDS)-logt0)
>
>	var = 0. ; free memory
>
>
>
>	; Just to be sure it's not some kind of IDL cache stuff
>
>	logt0 = SYSTIME(/SECONDS)
>
>	NCDF_VARGET, fid, 'var1', var, STRIDE=[1,1,1,1]
>
>	print, 'Get var1 (STRIDE):' + STRING(SYSTIME(/SECONDS)-logt0)
>
>	var = 0. ; free memory
>
>	logt0 = SYSTIME(/SECONDS)
>
>	NCDF_VARGET, fid, 'var2', var
>
>	print, 'Get var2          :' + STRING(SYSTIME(/SECONDS)-logt0)
>
>	var = 0. ; free memory
>
>	NCDF_CLOSE, fid
>
```

```
>
>
>  end
>
>
>
>
>
>  Results:
>
>
>
>  IDL> .FULL_RESET_SESSION
>
>  IDL> print, !VERSION
>
>  { x86_64 linux unix linux 8.2.1 Aug 20 2012      64      64}
>
>  IDL> test_ncdf_strides
>
>  % Compiled module: TEST_NCDF_STRIDES.
>
>  % Loaded DLM: NCDF.
>
>  Get var1          :      0.97150087
>
>  Get var2 (STRIDE):       1.1921642
>
>  Get var1 (STRIDE):       1.1618340
>
>  Get var2          :      0.93627405
>
>  IDL> test_ncdf_strides, /NETCDF4_FORMAT
>
>  Get var1          :      0.54904699
>
>  Get var2 (STRIDE):       4.4865382
>
>  Get var1 (STRIDE):       4.4508131
>
>  Get var2          :      0.53942299
>
>  IDL> test_ncdf_strides, /NETCDF4_FORMAT
>
>  Get var1          :      0.54891205
>
>  Get var2 (STRIDE):       4.4209800
>
```

```
>  Get var1 (STRIDE):      4.3824911
>
>  Get var2        :      0.53269601
>
>  IDL> test_ncdf_strides
>
>  Get var1        :      0.96190310
>
>  Get var2 (STRIDE):      1.1909840
>
>  Get var1 (STRIDE):      1.1693609
>
>  Get var2        :      0.93884301
```

Hi Fab,

Sorry to hear that you are having troubles with NetCDF4. Our wrapper code is just passing the STRIDE values down to the NetCDF4 library. If the user doesn't pass in a STRIDE, then we don't pass it into NetCDF4.

Presumably, within the NetCDF4 library, there are two different code paths, one for "no stride" (i.e. stride=1) and the other for stride=N. It's probably not checking to see if you pass in stride=1, but is just going straight to the "slower" stride code.

So, in short, I would recommend putting a check into your pro code, and only pass in the stride when you absolutely have to.

Hope this helps.
Cheers,
Chris
ExelisVIS

Subject: Re: Very tricky NetCDF "bug"
Posted by Fabzi on Sat, 15 Dec 2012 17:29:12 GMT

View Forum Message <> Reply to Message

Hi Chris,

On 12/15/2012 06:22 AM, Chris Torrence wrote:
> Presumably, within the NetCDF4 library, there are two different code
>paths, one for "no stride" (i.e. stride=1) and the other for stride=N.
>It's probably not checking to see if you pass in stride=1, but is just
> going straight to the "slower" stride code.

Yes, that's what I was thinking. It's not really "a bug", but it's just
that my process completely froze and it was really hard to find the
bottleneck...

> So, in short, I would recommend putting a check into your pro code,
> and only pass in the stride when you absolutely have to.
Yes, that's what I did.

Thanks,

Fab