GROMACS - Bug #1834

Unable to access large (> 2GB) files using gromacs-5.x on 32-bit OS

10/01/2015 08:05 AM - Venkat Reddy

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:	build system		
Target version:			
Affected version - extra info:	5.0.x, 5.1.x	**Difficulty:**	uncategorized
Affected version:	5.1		

Description

Dear all,
I have a trajectory file generated by gromacs-4.5.5. Recently I tried to plot radial distribution function using 'gmx rdf' tool available in gromacs-5.1. But I am getting the following error.

```
Program:     gmx rdf, VERSION 5.2-dev
Source file: src/gromacs/utility/path.cpp (line 406)
Function:    static void gmx::File::throwOnError(const gmx::File::NotFoundInfo&)
System I/O error:
Failed to access file 'traj.xtc'.
The file could not be opened.
```

```
Reason: Value too large for defined data type
  (call to fopen() returned error code 75)
```

Few details about the software and hardware used:
1) Linux Karplus 3.11.0-12-generic #19-Ubuntu SMP i686 GNU/Linux
2) CMakeCache.txt (In the attachments)
3) src/config.h (In the attachments)
4) src/gmxpre-config.h (In the attachments)
5) I have used cmake version 2.8.11.2 for installation

Associated revisions

Revision 5d85d5af - 01/24/2016 10:44 AM - Teemu Murtola

Document limitations on 32-bit Linux

Large file support does not currently work on 32-bit Linux,
so document this in the install guide.

Related to #1834

Change-Id: I8a3dcf1c2d2423eefc44d4ac1dc366c5f8753c31

Revision acd1df19 - 05/11/2016 12:21 AM - Erik Lindahl

Fixed large file issue on 32-bit platforms

At some point gcc started to issue a warning
instead of a fatal error for the checking
code; fixed to really generate an error now.

Fixes #1834.

Change-Id: I5827f358b22de516becaac02d81229b82b02297b

History

#1 - 10/02/2015 05:53 AM - Teemu Murtola

- Category changed from testing to build system

- Assignee deleted (Teemu Murtola)

- Affected version changed from 5.1.1 to 5.1

#2 - 10/06/2015 11:43 AM - Teemu Murtola

- Subject changed from Unable to access old xtc file using gromacs-5.1 to Unable to access large (> 2GB) files using gromacs-5.1

Based on further discussion on gmx-users, the problem is that any file that is larger what 32-bit file APIs accept causes this issue. The build system tries to give a fatal error when 64-bit file support is not present, but in this case it fails to do that. According to CMake tests, off_t is apparently 64 bits and fseeko() is found, both without any additional macro definitions.

The CMake code has not changed significantly between 5.0 and 5.1, but this might have to do with C vs. C++ compilation and the standard library. Someone with access to a failing system should test with a simple application (that just does an fopen() and reports success/failure and errno) which combinations work for opening large files:

- C code.
- C++ code that uses stdlib.h and fopen().
- C++ code that uses cstdlib and fopen().
- C++ code that uses cstdlib and std::fopen().

Could you, Venkat, do that?

#3 - 10/08/2015 05:04 PM - Teemu Murtola

Teemu Murtola wrote:

> Could you, Venkat, do that?

Based on your reply on gmx-users, you volunteered, but would need more information. So you could compile this code:

```
#include <stdio.h>
// #include <cstdio>  // Replace the above with this for tests 3 and 4

#include <errno.h>
#include <string.h>

int main (int argc, char *argv[])
{
    if (argc <= 1)
        return 2;
    FILE *fp = fopen(argv[1], "r");
    // Replace the above with std::fopen() for test 4.
    if (fp == NULL) {
        printf("Failed to open %s: %s\n", argv[1], strerror(errno));
    } else {
        printf("Successfully opened %s\n", argv[1]);
        fclose(fp);
    }
    return 0;
}
```

first with the C compiler you used for Gromacs, then with the C++ compiler used, and then with the modifications indicated in the code as comments. Run each program with the path to a large xtc or other file as the only command-line parameter, and report which of these work.

#4 - 10/14/2015 08:44 PM - Teemu Murtola

- Assignee set to Venkat Reddy

I at least would need additional information, as requested in the previous comment, to try to fix this; some other developer with direct access to a system where such a failure can be reproduced might be able to do more.

#5 - 11/23/2015 05:46 PM - Mark Abraham

- Status changed from New to Feedback wanted

- Target version deleted (5.1.1)

#6 - 11/30/2015 07:49 AM - Teemu Murtola

- Subject changed from Unable to access large (> 2GB) files using gromacs-5.1 to Unable to access large (> 2GB) files using gromacs-5.x on 32-bit OS

- Affected version - extra info set to 5.0.x, 5.1.x

According to a post in gmx-users, this also seems to affect 5.0.x, and is related to using a 32-bit OS. But still not enough information to actually know why.

#7 - 11/30/2015 12:03 PM - Venkat Reddy

Hi,
I am extremely sorry for delay in reply.
I have executed the code as Teemu suggested:

Trial-I: with C-compiler
$_gcc-4.8 test.c_
$_./a.out full-12us.xtc_

Trial-II: with C++ compiler
$_g++ test.c_
$_./a.out full-12us.xtc_

Trial-III: by replacing stdio.h with cstdio
$_g++ test.c_
$_./a.out full-12us.xtc_

Trial-IV by replacing stdio.h with cstdio and 'fopen()' with 'std::fopen()'
$_g++ test.c_
$_./a.out full-12us.xtc_

But every time it ended up with the error "Value too large for defined data type"

However, I got the message "Successfully opened" with small file as input in all the above conditions.

Please suggest me how to proceed further.

#8 - 11/30/2015 12:48 PM - Teemu Murtola

Interesting. Could you add

```
#define _FILE_OFFSET_BITS 64
```

as the first line in the test program, and try again? It is sufficient to test with the C compiler only and with the first variant.

#9 - 11/30/2015 01:02 PM - Venkat Reddy

Hi Teemu,
I have modified the code accordingly. Now it is able to open large trajectory files successfully. Thank you for your help.

#10 - 11/30/2015 01:12 PM - Teemu Murtola

Thanks for testing, at least we now know something. But it's still not clear why the build system does not realize that it should define this macro. It tests whether it is necessary, but for some reason seems to think that even without this macro, everything is fine.

To provide a fix in Gromacs, we would still need to figure out the reason for this.

#11 - 01/06/2016 08:52 PM - Teemu Murtola

- Status changed from Feedback wanted to Accepted

- Assignee deleted (Venkat Reddy)

Moving the issue to accepted state, since there are now at least three independent reports on gmx-users of this same issue: large file support does not work in Gromacs on 32-bit Linux, even though we claim that it should not be possible to compile Gromacs without such support. It looks like that the issue is in the CMake detection, and my current guess would be that the problem has always been there. So probably also earlier versions are affected, if compiled using CMake. But until someone with access to a failing 32-bit system can pinpoint what is exactly wrong with the build system, the issue is likely to stay in the current, unresolved state.

#12 - 01/18/2016 05:30 PM - Gerrit Code Review Bot

Gerrit received a related DRAFT patchset '1' for Issue #1834.
Uploader: Teemu Murtola (teemu.murtola@gmail.com)
Change-Id: I8a3dcf1c2d2423eefc44d4ac1dc366c5f8753c31
Gerrit URL: https://gerrit.gromacs.org/5558

#13 - 01/18/2016 08:14 PM - Szilárd Páll

I was able to reproduce the exact same behavior as the OP's tests on our 32-bit ARM dev box. However, I've no clue how to proceed. I could help with testing though (or arrange access if it's quicker for you that way!).

#14 - 01/18/2016 08:38 PM - Teemu Murtola

Szilárd Páll wrote:

> I was able to reproduce the exact same behavior as the OP's tests on our 32-bit ARM dev box. However, I've no clue how to proceed. I could help with testing though (or arrange access if it's quicker for you that way!).

As I wrote earlier, I think the issue is somewhere in the CMake code (in cmake/gmxTestLargeFiles.cmake). What would be useful is to trace exactly which of those try_compile() calls there get run, and with what results, including the compiler output. And a confirmation that if the detection is hacked to set _FILE_OFFSET_BITS, Gromacs works correctly with large files afterwards. Depending on the results, it might then be easier or harder trying to figure how cmake/TestFileOffsetBits.c should be changed to produce the necessary detection results. I think Erik has originally written all of that, so he might remember why some of the stuff is like it is.

#15 - 05/10/2016 06:40 PM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #1834.
Uploader: Erik Lindahl (erik.lindahl@gmail.com)
Change-Id: I3522816e0e012c9141c5a27dee0c82a8ff6bb1ab
Gerrit URL: https://gerrit.gromacs.org/5854

#16 - 05/10/2016 06:40 PM - Erik Lindahl

- Status changed from Accepted to Fix uploaded

#17 - 05/11/2016 12:22 AM - Gerrit Code Review Bot

Gerrit received a related patchset '2' for Issue #1834.
Uploader: Erik Lindahl (erik.lindahl@gmail.com)
Change-Id: I5827f358b22de516becaac02d81229b82b02297b
Gerrit URL: https://gerrit.gromacs.org/5857

#18 - 05/11/2016 01:30 AM - Erik Lindahl

- Status changed from Fix uploaded to Resolved

Applied in changeset acd1df190381b1fe53f534480beac4b175f32b45.

#19 - 05/14/2016 02:37 PM - Erik Lindahl

- Status changed from Resolved to Closed

Files

CMakeCache.txt	56 KB	10/01/2015	Venkat Reddy
config.h	10.6 KB	10/01/2015	Venkat Reddy
gmxpre-config.h	2.82 KB	10/01/2015	Venkat Reddy