Namespace for isfinite in src/external/lmfit/lmmin.cpp is not specified

Description
A violation of C++11 syntax is detected in src/external/lmfit/lmmin.cpp. When compiling GROMACS with Intel Compiler the icpc compiler rises the following error:

```
/home/builder/gromacs-2018/src/external/lmfit/lmmin.cpp(395): error: more than one instance of overloaded function "isfinite"
matches the argument list:
  function "isfinite(double)"
  function "std::isfinite(double)"
argument types are: (double)
if (!isfinite(temp1)) {
  ^
```

The compiler detects multiple (two in this case) instances of isfinite. On the GNU Linux systems isfinite is supplied by both math.h and it is also part to std naming space. Providing proper name space to isfinite is recommended. See the patch attached.

Associated revisions
Revision d3d2ea00 - 02/06/2018 10:57 AM - Mark Abraham
Use isfinite unambiguously
Patch provdied by Veselin Kolev
Fixes #2400
Change-IId: 11923998a2288e20cb339e7fc59210cbf9ac5bfa8

History
#1 - 02/06/2018 08:49 AM - Mark Abraham
Thanks. We test icc/icpc 18.0.0.20170811 in our CI and observe no issue. What version were you using?

#2 - 02/06/2018 08:53 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #2400.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-IId: gromacs~release-2018~I1923998a2288e20cb339e7fc59210cbf9ac5bfa8
Gerrit URL: https://gerrit.gromacs.org/7545

#3 - 02/06/2018 11:00 AM - Mark Abraham
- Status changed from New to Resolved

Applied in changeset d3d2ea008158d3a2f2a9d3355587db3bd3d55951.

#4 - 02/06/2018 08:05 PM - Veselin Kolev
Mark Abraham wrote:

Thanks. We test icc/icpc 18.0.0.20170811 in our CI and observe no issue. What version were you using?

Hello Mark,
Thank you for your fast reply.

15.0.0.X is the version I ran to compile the code of 2018. It seems pretty old, but the current version I have installed on my build host (18.0.0.X) does not support the compilation of the code against CUDA 9.0, because 18.0.0.X is too new for that. But regardless the compiler versions and the default code syntax checking features, having more than one source of a function is against C++11. It is true that most of the compiler vendors are used to implement some workarounds to avoid detecting those kind of errors, like skipping the strict syntax checks or allowing multiple sources of the same function by default. But if you run a compilation that supports strict code syntax check, even the new versions of the compiler will detect that function source multiplicity as a violation error.

#5 - 02/08/2018 10:50 PM - Mark Abraham
- Status changed from Resolved to Closed

Veselin Kolev wrote:

Mark Abraham wrote:

Thanks. We test icc/icpc 18.0.0.20170811 in our CI and observe no issue. What version were you using?

Hello Mark,

Thank you for your fast reply.

15.0.0.X is the version I ran to compile the code of 2018. It seems pretty old, but the current version I have installed on my build host (18.0.0.X) does not support the compilation of the code against CUDA 9.0, because 18.0.0.X is too new for that. But regardless the compiler versions and the default code syntax checking features, having more than one source of a function is against C++11. It is true that most of the compiler vendors are used to implement some workarounds to avoid detecting those kind of errors, like skipping the strict syntax checks or allowing multiple sources of the same function by default. But if you run a compilation that supports strict code syntax check, even the new versions of the compiler will detect that function source multiplicity as a violation error.

Yes, we agree. But the surface area of versions that we support is enormous :-) 

#6 - 02/08/2018 11:06 PM - Veselin Kolev

Mark Abraham wrote:

Veselin Kolev wrote:

Mark Abraham wrote:

Thanks. We test icc/icpc 18.0.0.20170811 in our CI and observe no issue. What version were you using?

Hello Mark,

Thank you for your fast reply.

15.0.0.X is the version I ran to compile the code of 2018. It seems pretty old, but the current version I have installed on my build host (18.0.0.X) does not support the compilation of the code against CUDA 9.0, because 18.0.0.X is too new for that. But regardless the compiler versions and the default code syntax checking features, having more than one source of a function is against C++11. It is true that most of the compiler vendors are used to implement some workarounds to avoid detecting those kind of errors, like skipping the strict syntax checks or allowing multiple sources of the same function by default. But if you run a compilation that supports strict code syntax check, even the new versions of the compiler will detect that function source multiplicity as a violation error.

Yes, we agree. But the surface area of versions that we support is enormous :-) 

I see. Regarding the issue we experiences with the Intel Compiler versions for Linux. I installed exactly 18.0.0.20170811 and I can confirm the compilation is going smoothly to a successful end.

Thank you.

Files

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Date</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>lmmin.cpp.c++11_isfinite_namespace_ref.patch</td>
<td>1.85 KB</td>
<td>02/06/2018</td>
<td>Veselin Kolev</td>
</tr>
</tbody>
</table>