GROMACS - Bug #2989
(thread-) MPI setup hanging on bs_jetson_tk1

06/20/2019 02:41 PM - Paul Bauer

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>core library</td>
</tr>
<tr>
<td>Category:</td>
<td>core library</td>
</tr>
<tr>
<td>Target version:</td>
<td>2020-infrastructure-stable</td>
</tr>
<tr>
<td>Affected version - extra info:</td>
<td>git master</td>
</tr>
<tr>
<td>Affected version:</td>
<td>git master</td>
</tr>
<tr>
<td>Difficulty:</td>
<td>uncategorized</td>
</tr>
</tbody>
</table>

**Description**
Running gmx mdrun leads to a hang during the thread-MPI setup phase since commit https://gerrit.gromacs.org/c/gromacs/+11197

**Related issues:**
- Related to GROMACS - Task #2986: Post submit failing in two configurations
- Related to GROMACS - Task #2992: Split hw_opt in const user options and dynam...

**Associated revisions**
- Revision 5c15e1c2 - 07/01/2019 06:19 PM - Berk Hess
  - Fix MPI deadlock in affinity setting
- Change-Id: I454b38deb9ff11c90cdf4a19aaa80e79e8898df5

**History**

**#1 - 06/20/2019 02:42 PM - Paul Bauer**
- Related to Task #2986: Post submit failing in two configurations added

**#2 - 06/20/2019 03:44 PM - Szilárd Páll**
Reproduced, the issue seems to be that some tMPI ranks are stuck in analyzeThreadsOnThisNode() MPI_Scan(), while the other is already in the next bcast collective:

Here's the backtrace.

```
$ gdb --args gmx mdrun -ntmpi 2 -ntomp 1 -nb cpu -notunepme -s topol ...
[...]
(gdb) bt
#0  tMPI_Event_wait (ev=0x398c8) at /nethome/pszilard/gromacs-master/src/external/thread_mpi/src/event.cpp:71
#1  0xb6cc9bdc in tMPI_Wait_for_others (cev=0x36fd4, myrank=0) at /nethome/pszilard/gromacs-master/src/external/thread_mpi/src/collective.cpp:522
#2  0xb6cc8ebc in tMPI_Bcast (buffer=0xbeffe9a0, count=8, datatype=0xb6fcbb24 <tmpi_byte>, root=0, comm=0x396b8) at /nethome/pszilard/gromacs-master/src/external/thread_mpi/src/bcast.cpp:98
#3  0xb6bfcdb2 in gmx::Mdrunner::mdrunner (this=0xbeffede8) at /nethome/pszilard/gromacs-master/src/gromacs/mdrun/runner.cpp:1225
#4  0xb6ea3b5a in gmx_bcast_sim (nbytes=8, b=0xbefe9a0, cr=0x5ad90) at /nethome/pszilard/gromacs-master/src/gromacs/gmx11b/network.cpp:286
#5  0xb6bfcdb2 in gmx::Mdrunner::mdrun (this=0xbeffede8) at /nethome/pszilard/gromacs-master/src/gromacs/mdrun/runner.cpp:1225
#6  0xb671bd98 in gmx::main (argc=12, argv=0xbefe608) at /nethome/pszilard/gromacs-master/src/gromacs/cmdline/cmdlinemodulemanager.cpp:133
#7  0xb71d034 in gmx::CommandLineModuleManager::run (this=0xbefe48c, argc=12, argv=0xbefe608) at /nethome/pszilard/gromacs-master/src/gromacs/cmdline/cmdlinemodulemanager.cpp:589
#8  0x00012b7c in main (argc=13, argv=0xbefe604) at /nethome/pszilard/gromacs-master/src/gromacs/cmdline/gmx.cpp:60
```

04/01/2020 1/2
Simple reason could be that the refactoring broke things and `hw_opt.thread_affinity != threadaffOFF` check evaluates differently on the two ranks (affinities get turned off as for some reason the mask on this slave is 0x1 by default).

Can't dig deeper ATM, but if anybody wants to debug here's a binary you can grab on dev-jetson01:
/home/pszilard/gromacs-master/build_gcc8/bin/gmx

### #3 - 06/20/2019 04:09 PM - Szilárd Páll

A few more tests:

```
$ gmx mdrun -ntmpi 2 -ntomp 1 -nb cpu -notunepme -s topol -pin off
$ gmx mdrun -ntmpi 2 -ntomp 1 -nb cpu -notunepme -s topol -pin on
```

both work to some extent -- crashes due to unrelated issue).

The latter actually fails to set affinities, but this may be a peculiarity on the ARM board.

This seems to support my above hypothesis about the thread affinity check not evaluating to the same value on all ranks

### #4 - 06/20/2019 04:41 PM - Mark Abraham

ok I'll look over the code again on the flight

### #5 - 06/21/2019 12:34 AM - Mark Abraham

I didn't spot anything that suggested the code is a problem :-( I suggest we revert my change (on Monday!)

### #6 - 06/24/2019 12:36 PM - Szilárd Páll

Mark Abraham wrote:

> I didn't spot anything that suggested the code is a problem :-( I suggest we revert my change (on Monday!)

The issue was indeed that conditional on line 1194 that I pointed at; after refactoring that conditional communication became a path that was triggered with tMPI and as on the Jetson TK1 the kernel seems to change the system-wide affinity mask based on load, different ranks detected different masks and therefore different values of `hw_opt.thread_affinity`.

### #7 - 06/24/2019 12:42 PM - Berk Hess

- Related to Task #2992: Split `hw_opt` in const user options and dynamic settings added

### #8 - 07/01/2019 06:30 PM - Berk Hess

- Status changed from New to Resolved

Applied in changeset 5c15e1c2eacc840b977f3fd683edab453f0e5ae.

### #9 - 07/01/2019 11:36 PM - Mark Abraham

- Status changed from Resolved to Closed