GROMACS - Bug #1254

a likely mdrun memory corruption/race condition

05/17/2013 08:29 PM - Szilárd Páll

Status: Closed
Priority: High
Assignee: Sander Pronk
Category: mdrun
Target version: 4.6.3
Affected version - extra info:
Affected version: 4.6.2

Difficulty: uncategorized

Description

Based on some strange mdrun runtime behavior, including unexpected behavior, race condition, and segv-s it is quite likely that there is memory corruption occurring in parallel mdrun runs which might be related to affinity setting.

Symptoms:

1. The message "Pinning threads with a logical core stride of..." is often missing from the log file even if -pinstride is not set on the command line - this could only happen if the memory holding the stride gets overwritten (see gmx_thread_affinity.c:133);
2. Valgrind reports use of uninitialized values (see attached);
3. With MPI builds in some cases race conditions and segv-s have been observed. I've managed to repro on the tcbs2x.theophys.kth.se AMD compute machines with a 192k atom water system as well as a protein system Anders G. is working with (see in/nethome/anders/VSDbox/kv21_vsd-GPU_testing/*crash).

UPDATE: This bug seems to not be related to the affinity setting issue.
UPDATE2: It is not related, the deadlock is reproducible even with e5d22a35.

Related issues:

Related to GROMACS - Task #1290: OpenMPI 1.4.3 in Ubuntu 12.04 can produce si... Closed 06/25/2013

Associated revisions

Revision e4229dee - 05/24/2013 07:35 PM - Sander Pronk

Fixed a potential race condition in tMPI_Thread_create()

The pthreads version of tMPI_Thread_create contained a potential race condition where the tMPI_Thread_t structure may not be fully populated as the child thread starts.

Refs #1254

Change-Id: l8e6f6aaa3d27b88269257be8d7df66ef728d8b0d

Revision e5d22a35 - 05/30/2013 11:19 AM - Szilárd Páll

fix thread-safety issue in affinity layout detection

The pinning stride variable is shared by thread_mpi ranks and its value is changed only when its initial value is 0. This posed a thread-safety issue and in some cases the selected pinning stride was only reported in the log if rank 0 happened to arrive first to the affinity layout detection.

Refs #1254

Change-Id: ld32a8fcbeacea2205fd15c30b46320ec7dd35e5e

Revision 78569369 - 07/01/2013 08:03 PM - Sander Pronk

Comprehensive hwinfo structure concurrency fix.

The hwinfo structure and structures contained therein are inherently global to any mdrun processes/ranks. This patch makes sure that:
- The hwinfo structure is shared among all threads
- Only one thread creates a hwinfo structure
- The hwinfo structure is safe to read for all threads after they obtain it.

In addition, it fixes the detection for pthread_setaffinity in thread_mpi.

This fixes concurrency issues with thread affinity settings with or without MPI, and makes runner.c slightly easier to read because the concurrency logic is pushed to gmx_detect_hardware.c

Fixes #1270, #1254

Note that #1254 issue 3 seems to be an OpenMPI bug.

Change-Id: I236e81923324d7873f3d8633889b91c7c02a7843

History

#1 - 05/17/2013 08:31 PM - Sander Pronk
- Assignee changed from Berk Hess to Sander Pronk

That looks like it's in thread_mpi. Let me double-check.

#2 - 05/17/2013 08:35 PM - Szilárd Páll
- Description updated

Regarding symptom 3.: the last thing the MPI ranks "furthest" in the code print to the debug output is the affinity setting message.

Additionally, I've checked a few things:

  - the last version which still used sched_setaffinity() directly instead of the mechanism provided by thread_mpi does not result in valgrind warnings;
  - however, replacing pthread_setaffinity_np with sched_setaffinity does not eliminate symptom 1.

#3 - 05/20/2013 01:52 PM - Szilárd Páll
@Sander: have you found anything?

#4 - 05/20/2013 03:04 PM - Sander Pronk
I've found a potential race condition (depending on how pthread_create works) that could cause this.

Due to the nature of that condition, it would mean that this memory corruption is only seen in the context of thread_mpi and not OpenMP threads. Is this correct?

#5 - 05/20/2013 03:53 PM - Szilárd Páll
Sander Pronk wrote:

  I've found a potential race condition (depending on how pthread_create works) that could cause this.

  Due to the nature of that condition, it would mean that this memory corruption is only seen in the context of thread_mpi and not OpenMP threads. Is this correct?

As it seems to only reproduce with -ntmpi > 1, symptom 1. could be explained by a thread_mpi-specific race condition. However the valgrind warnings (symptom 2) are also present with -ntmpi = 1 and symptom 3 is MPI-specific.

#6 - 05/20/2013 11:54 PM - Szilárd Páll
- Description updated

Note that Symptom 3. seems to be caused by another bug (in many cases mdrun hangs or crashes in unrelated parts of the code)

@Sander: do you think symptoms 1. and 2. are/could be both related to the potential race condition? If you have a fix, could you try if either or both get resolved?

#7 - 05/21/2013 01:46 AM - Szilárd Páll

Szilárd Páll wrote:

  Note that Symptom 3. seems to be caused by another bug (in many cases mdrun hangs or crashes in unrelated parts of the code)
Not so sure anymore. I've found cases which suggest that symptom 3 could be related to affinity setting.

In several cases most ranks (except 1 or 2) are deadlocked at the following spot which is rather close to the `gmx_set_thread_affinity()` call:

```
(gdb) bt
#0 0x0000007f8d750602d8 in poll () from /lib/x86_64-linux-gnu/libc.so.6
#1 0x0000007f8d746d60b0 in ?? () from /usr/lib/libopen-pal.so.0
#2 0x0000007f8d746d6ff0 in ?? () from /usr/lib/libopen-pal.so.0
#3 0x0000007f8d746d2221 in opal_progress () from /usr/lib/libopen-pal.so.0
#4 0x0000007f8d75d9d655 in ?? () from /usr/lib/libmpi.so.0
#5 0x0000007f8d6eea9afa in ?? () from /usr/lib/libopen-pal.so.0
#6 0x0000007f8d6dd8ff0 in ?? () from /usr/lib/libopen-pal.so.0
#7 0x0000007f8d6d2221 in opal_progress () from /usr/lib/libopen-pal.so.0
```

while 1-2 ranks tend to deadlock in the MPI_Allreduce() in `gmx_set_thread_affinity()`:

```
(gdb) bt
#0 0x0000007f85f7a902d8 in poll () from /lib/x86_64-linux-gnu/libc.so.6
#1 0x0000007f85f710eab0 in ?? () from /usr/lib/libopen-pal.so.0
#2 0x0000007f85f710d8ff0 in ?? () from /usr/lib/libopen-pal.so.0
#3 0x0000007f85f7102221 in opal_progress () from /usr/lib/libopen-pal.so.0
#4 0x0000007f85f7c10221 in opal_progress () from /usr/lib/libopen-pal.so.0
#5 0x0000007f85f87c10221 in opal_progress () from /usr/lib/libopen-pal.so.0
#6 0x0000007f85f87c6c655 in ?? () from /usr/lib/libopen-pal.so.0
#7 0x0000007f85f87c6c655 in ?? () from /usr/lib/libopen-pal.so.0
#8 0x0000007f85f7c10221 in opal_progress () from /usr/lib/libopen-pal.so.0
#9 0x0000007f85f7c10221 in opal_progress () from /usr/lib/libopen-pal.so.0
#10 0x0000007f85f7c10221 in opal_progress () from /usr/lib/libopen-pal.so.0
```

05/03/2020
the valgrind issues appear to be caused by valgrind interacting with get_thread_affinity_layout() and then gmx_cpuid_topology(), which then appears to return erroneous data.

I'll check whether gmx_cpuid.c could have issues.

#11 - 05/24/2013 04:55 AM - Szilárd Páll
I might have found the source of symptom 1. It looks like hw_opt is shred between tMPI ranks, but get_thread_affinity_layout() will actually set hw_opt->core_pinning_stride on every rank depending on the variables initial value. I think this causes the anomaly as some ranks will see the hw_opt->core_pinning_stride already updated by some other rank and not its original value and if rank 0 (which has fplog!=NULL happens to be late, it will see the updated value and not print the message to the log.

However, the valgrind error is still there and so is the other crash.

#12 - 05/31/2013 10:46 AM - Mark Abraham
- Target version changed from 4.6.2 to 4.6.3

#13 - 05/31/2013 08:54 PM - Szilárd Páll
- Description updated

#14 - 05/31/2013 08:58 PM - Szilárd Páll
- File repro.tpr added

Szilárd Páll wrote:

1. With MPI builds in some cases race conditions and segv-s have been observed. I’ve managed to repro on the tcbs2x.theophys.kth.se AMD compute machines with a 192k atom water system as well as a protein system Anders G. is working with (see in/nethome/anders/VSDbox/kv21_vsd-GPU_testing/crash).

UPDATE: This bug seems to not be related to the affinity setting issue.

UPDATE2: It is not related, the deadlock is reproducible even with e5d22a35.

Reproducible using the attached tpr on any of the tcbs[2-7] machines with the following command line deadlocks:

mpirun -np 4 $mdrun $opts -nb cpu -g test_4x8 -nsteps 10000 -npme 1 -pin on

while with -pin off it does not.

#15 - 06/03/2013 03:11 PM - Sander Pronk
- Status changed from New to In Progress
- Affected version changed from 4.6.1 to 4.6.2

After speaking to Szilárd, I just realized another source of deadlocks for problem #2. Will come with a fix shortly.

#16 - 06/04/2013 08:25 PM - Szilárd Páll
- Subject changed from a likely mdrun memory corruption to a likely mdrun memory corruption/race condition

Jenkins builds are deadlocking (e.g. this). Apparently this only happens to the MPI builds which may suggest that symptom 3 may have resurfaced.

#17 - 06/21/2013 11:00 AM - Mark Abraham

Assuming this is the same repro.tpr Szilard mentioned in https://gerrit.gromacs.org/#/c/2433/, then I have observed hangs in gmx_setup_nodecomm all the way back to 46bc0de pre-4.6. Will try some parallel memory debugging with DDT.

#18 - 06/21/2013 03:44 PM - Mark Abraham

Update: can reproduce comm_split failure on tcbs21 with GMX_GPU=off. Worked fine on povel, no memory errors found by DDT.

#19 - 06/24/2013 05:09 PM - Mark Abraham

Update: I could observe irreproducible crashes on a similarly-sized waterbox right back to 4.5, on OpenMPI 1.4.3 on our local machines. We tried MPICH-1.4.1 on tcbproject03, and observed both 4.5.5 and https://gerrit.gromacs.org/#/c/2433/ to work reproducibly. povel found no memory issue on https://gerrit.gromacs.org/#/c/2433/. Sander's experience of MPI valgrind was that OpenMPI provokes lots of "clearly not GROMACS problem" warnings.

So we suspect a bug in OpenMPI 1.4.3. Will try some other versions tomorrow.
I updated tcbs21 to the OpenMPI 1.5.4 package in the precise repo, and see no problems. I don't think there are any occurrences of problem category 3 that Szilard and I have seen that can't be written off to a bug there. [http://svn.open-mpi.org/svn/ompi/branches/v1.4/NEWS](http://svn.open-mpi.org/svn/ompi/branches/v1.4/NEWS) does mention a couple of plausible suspects, but nothing leaps out at me.

So I think the patch in gerrit is good to go once we've updated the commit message accordingly.

---

**#21 - 06/26/2013 12:39 AM - Mark Abraham**
- Status changed from In Progress to Fix uploaded

**#22 - 07/05/2013 06:28 PM - Mark Abraham**
- Status changed from Fix uploaded to Resolved

**#23 - 08/13/2013 08:55 PM - Sander Pronk**
- % Done changed from 0 to 100

Applied in changeset 78569369348e07a0300a03f00e667e6187858025.

**#24 - 12/16/2013 05:41 PM - Rossen Apostolov**
- Status changed from Resolved to Closed

---

**Files**

<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>valgrind.out</td>
<td>5.86 KB</td>
<td>05/17/2013</td>
<td>Szilárd Pál</td>
</tr>
<tr>
<td>repro.tpr</td>
<td>897 KB</td>
<td>05/31/2013</td>
<td>Szilárd Pál</td>
</tr>
</tbody>
</table>