Gromacs - Bug #1334
concurrency-related bug with thread-MPI
09/13/2013 10:36 PM - Sziárd Páll

**Status:** Closed
**Priority:** High
**Assignee:** Berk Hess
**Category:** mdrun
**Target version:** 4.6.4

**Description**

78569, while fixing some thread-safety related aspects of hardware detection, introduced other concurrency issue(s) which manifest(s) in:

- hardware detection report not printed to log/console when ntmpi>1;
- GPU oversubscription check not working: e.g -ntmpi 2 -gpu_id 00 starts to run, but due to concurrency issues with GPU context sharing between tMPI ranks, it throws an error or segfaults before exiting (which is the reason why GPU oversubscription has not been allowed with tMPI).

The issue has also been reported on the user's list:

**Related issues:**
Related to Gromacs - Bug #1270: affinity setting broken with MPI

**Associated revisions**
Revision 95d10d39 - 10/31/2013 11:56 PM - Berk Hess
reorganized GPU detection and selection

The GPU selection has been separated from the GPU detection and now happens after the thread-MPI threads are started.
The GPU user/auto-selected options have been removed from gmx_hw_info_t, such that it only contains hardware info and can be passed around as const.
As both the CPU and GPU options structs are now tMPI rank local, tMPI thread concurrency issues are avoided.
Fixes #1334 #1359

The GPU detection is now skipped with mdrun -nb cpu
CPU acceleration binary/hardware mismatch is now only printed once to stderr (instead of #MPI-rank times to stdout).
Removed the master_inf_t struct.

Change-Id: If497f611b911808f6d01ca83f41ae288061dd361

**History**

11/25/2015
Here's what happens: because of mutex-based implementation, now the first thread to arrive grabs the mutex and does the consistency checks, but all messages and warnings are printed using md_print_warn()/md_print_info() which only print on rank 0. Hence, if rank 0 is not the first to arrive and grab the mutex, the warnings/errors as well as detection information will not be issued.

I don't have a suggestion for a good solution. Making sure that tMPI rank 0 executes the critical region would defeat the purpose of the elegant "proper" threading-style mutex-based implementation. To me it still seems that this issue represents yet another reason for not treating the thread-MPI parallelization on the top level as a "native" multi-threading implementation, but more like an MPI implementation which in some cases requires special measures to ensure thread safety.

For the full discussion see my comments on gerrit #2433 PS5.

Here is a possible workaround: in the beginning of gmx_check_hw_runconf_consistency() do the following:

t_commrec *cr_hack;

#ifdef GMX_THREAD_MPI
    cr_hack = NULL;
#else
    cr_hack = cr;
#endif

and replace cr with cr_hack in all md_print_warn() and mn_print_info() calls.

This is rather hack-ish workaround, but to me it seems the least invasive solution - unless we want to strip the mutex-based implementation (which I'd be in favor of).

PS: I set the target version because the problem has been well defined and there is a suggestion for the solution.
#5 - 10/16/2013 03:21 PM - Mark Abraham
- Status changed from New to Fix uploaded

https://gerrit.gromacs.org/#q,lf497611b911809f6d01ca83f41aa288061dd361.n.z proposes a fix

#6 - 11/05/2013 04:10 PM - Berk Hess
- Status changed from Fix uploaded to Resolved
- % Done changed from 0 to 100

Applied in changeset 95d10d9903aed9c700009f6e9feda9c6f1d60517.

#7 - 11/19/2013 04:02 PM - Szilárd Páll
- Status changed from Resolved to Closed