GROMACS - Bug #1942

maxh option and checkpoint writting do not work with REMD simulations

04/06/2016 11:08 AM - Maud Jusot

Status:	Closed		
Priority:	Normal		
Assignee:	Mark Abraham		
Category:	mdrun		
Target version:	2016		
Affected version - extra info:		**Difficulty:**	uncategorized
Affected version:	5.1.2		

Description

I don't manage to restart correctly REMD simulation (I posted it on the gmx-users mailing list here : https://www.mail-archive.com/gromacs.org_gmx-users@maillist.sys.kth.se/msg18550.html) but to summarize it :
the first simulation has no problem and finish correctly after maxh time, but when it restarts no checkpoint files are written anymore and gromacs does not stop at maxh time (even if it says it does in the output).

I tried it with 3 different versions of gromacs (4.6.5, 5.1.0 and 5.1.2) on two different clusters, so I am quite sure the problem does not come from the installation nor from the version.

Related issues:

Related to GROMACS - Bug #692: Frequency of checking for inter-simulation sig...	**Closed**

Associated revisions

Revision bc98987b - 06/23/2016 02:53 PM - Mark Abraham

Prevent use of mdrun -maxh -multi

A proper fix can probably be made in release-2016, and if so, the
content of this commit should not be merged forward.

Refs #1942

Change-Id: Ie7e6c0ca25fba09ad1794cacbe116b03e95ff0f9

Revision d5bd278b - 06/27/2016 07:31 PM - Mark Abraham

Removed unnecessary inter-simulation signalling

Generally, multi-simulation runs do not need to couple the simulations
(discussion at #692). Individual algorithms implemented with
multi-simulations might need to do so, but should take care of their
own details, and now do. Scaling should improve in the cases where
simulations are now decoupled.

It is unclear what the expected behaviour of a multi-simulation should
be if the user supplies any of the possible non-uniform distributions
of init_step and nsteps, sourced from any of .mdp, .cpt or command
line. Instead, we report on the non-uniformity and proceed. It's
always possible that the user knows what they are doing. In
particular, now that multi-simulations are no longer explicitly
coupled, any heterogeneity in the execution environment will lead to
checkpoints and -maxh acting at different time steps, unless a
user-selected algorithm requires that the simulations stay coordinated
(e.g. REMD or ensemble restraints).

In the implementation of signalling, we have stopped checking gs for
NULL as a proxy for whether we should be doing signalling at that
communication phase. Replaced with a helper object in which explicit
flags are set. Added unit tests of that functionality.

Improved documentation of check_nstglobalcomm. mdrun now reports the
number of steps between intra-simulation communication to the
log file.

Noted minor TODOs for future cleanup.

Added some trivial test cases for termination by maxh in normal-MD, multi-sim and REMD cases. Refactored multi-sim tests to make this possible without duplication. This is complicated by the way filenames get changed by mdrun -multi by the former par_fn, so cleaned up the way that is handled so it can work and be re-used better. Introduced mdrun integration-test object library to make that build system work a little better. Made some minor improvements to Doxygen setup for integration tests.

Fixes #860, #692, #1857, #1942.

Change-Id: I5f7b98f331db801b058ae2b196d79716b5912b09

History

#1 - 04/06/2016 09:43 PM - Chris Neale

I have a different system, but gromacs 5.1.2 REMD with MPI works for me when I use it like this:

```
mpirun -np 2 gmx_mpi mdrun -notunepme -deffnm MD_ -dlb yes -npme 0 -cpt 60 -maxh 0.01 -cpi MD_ -replex 500 -mu
lti 2 -ntomp 1
```

I do see strange hangs with some combinations of X and Y in

```
mpirun -np X gmx_mpi mdrun -ntomp Y -replex 500 -multi 2
```

For example, X=24 and Y=1 hangs forever on startup whereas X=2 and Y=12 runs just fine, even from a continuation.

Can you verify with your system whether the mpirun usage first listed above works or not? If it does, then I suggest the issue is not whether maxh and cpt file writing is working, but whether your simulation is hanging for some reason without either dying or producing any output ... note that I have seen these hangs last >12 hours, so its not just a hiccup.

#2 - 04/07/2016 04:48 PM - Mark Abraham

The implementation of mdrun shutdown is largely untested (because the implementation is a total mess), and the case of multi-simulation with multiple ranks is particularly problematic because one has to make sure that all ranks agree that it is time to shut down before any of them can start to shut down (else ranks are left hanging waiting for replies, which is probably what is happening here). We try to implement -maxh by having any rank noticing that time is up send a signal to all others, but it isn't acceptable to have every rank in a multi-simulation do global communication every step just to check for trivia like this. So we need to send an intra-simulation signal alongside some other message, and then have inter-simulation signal get shared, and acknowledged, and then finally everything can agree to write a checkpoint and shut down. That sounds slightly complex, and I promise that there are more problems you don't want to know about ;-) I have been working for a long time to improve this, but it is one of the more difficult things to do in the present form of the code.

With mdrun -multi, I do recommend arranging life so that the .mdp nsteps will terminate the job acceptably.

#3 - 04/14/2016 09:56 AM - Maud Jusot

Sorry for this late answer, but my message hadn't been sent :
Thanks you very much Mark for your honnest response. I will try not to use maxh for REMD simulations in this case.

To Chris : I did what you asked me to do, and the problem is still the same. And I tried on the second cluster I use which seems to have a system closer to yours and it's the same again.

I might be able to have my jobs run for more than 24h, and the extension with tpr-convert seems to work well, which "solves" the problem for me. But please let me know if I can do something else to help for that. And thank you very much for your time and your answers !

#4 - 04/15/2016 09:20 PM - Chris Neale

I presume that "gmx mdrun -nsteps $NSTEPS" is also a workable option? You'd have to predict how many steps you can reliable complete for your system in the given wallclock time, but it might be cleaner than modifying the tpr each round.

I still don't understand how modifying the .tpr solves your problem though... you seemed to have 2 problems one of which was that upon restarting no new .cpt files were written. How did you solve that?

#5 - 05/27/2016 12:35 AM - Mark Abraham

- Related to Bug #692: Frequency of checking for inter-simulation signalling is too high for large-scale parallel REMD added

#6 - 06/23/2016 03:06 PM - Mark Abraham

- Status changed from New to Fix uploaded

- Assignee set to Mark Abraham

- Target version set to 2016

The origin of this issue is likely the confused implementations of both multi-simulation and signalling, and the way that some aspects of signalling for coordinating -maxh between simulations is scheduled according to the modulus of the overall simulation step with respect to nstglobalcomm, and others according to the modulus of the relative step (ie within this simulation part) with respect to nstglobalcomm. This works fine for the first simulation, but when the -maxh checkpoint is written at a step number that is nstlist+1, those moduli often do not coincide with the subsequent nstglobalcomm.

This can't be fixed reliably in release-5-1 branch because it isn't feasible to test everything related well enough. https://gerrit.gromacs.org/#/c/5899/16 proposes a fix for release-2016 branch, and I will upload a patch for release-5-1 that issues a fatal error in the case of mdrun -multi -maxh.

#7 - 06/23/2016 03:07 PM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #1942.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: Ie7e6c0ca25fba09ad1794cacbe116b03e95ff0f9
Gerrit URL: https://gerrit.gromacs.org/5984

#8 - 06/27/2016 07:34 PM - Mark Abraham

- Status changed from Fix uploaded to Closed

#9 - 11/02/2016 11:23 PM - Chris Neale

Mark's suggestion was: "With mdrun -multi, I do recommend arranging life so that the .mdp nsteps will terminate the job acceptably."

I am currently doing this via mdrun -nsteps, though I presume it could be done directly in the .tpr file. However, there is a complication.

In this case, then if the segment terminates on a number of steps that is a multiple of the argument to -replex, then no exchanges will be attempted on the final step of the simulation, nor the fist step of the continuation. I presume that this will cause substantial problems in demuxing. because there is a gap in which no exchanges were attempted (and therefore not recorded in the .log file). Very likely the issue can be fixed by also using mdrun -noappend and inserting an extra line for exchanges with complete rejection at the interface of the two runs. Just thought this would be a useful place to point this out.

#10 - 11/03/2016 05:15 PM - Mark Abraham

Chris Neale wrote:

> Mark's suggestion was: "With mdrun -multi, I do recommend arranging life so that the .mdp nsteps will terminate the job acceptably."
>
> I am currently doing this via mdrun -nsteps, though I presume it could be done directly in the .tpr file. However, there is a complication.
>
> In this case, then if the segment terminates on a number of steps that is a multiple of the argument to -replex, then no exchanges will be attempted on the final step of the simulation, nor the fist step of the continuation. I presume that this will cause substantial problems in demuxing. because there is a gap in which no exchanges were attempted (and therefore not recorded in the .log file). Very likely the issue can be fixed by also using mdrun -noappend and inserting an extra line for exchanges with complete rejection at the interface of the two runs. Just thought this would be a useful place to point this out.

Hmm. Thanks. :-(. Certainly a replica-exchange implemention must take care not to exchange twice at the same step. Currently there is explicit code that tries to prevent replica exchange at the very first step, and at the last step of every simulation part. But with our current forest of ways to restart simulations and mangle nsteps, I can well believe that something is setting the effective init_step .mdp field to zero and falling unexpectedly foul of the first check. #1768 is already open to note that having more than one way to do things is a recipe for buggy behaviour. One workaround would be to expand that first check to also check the simulation part - if how the .tpr + .cpt is working isn't **also** mangling the simulation part.

Chris, can you give some more details on how you discovered this, ie how the two mdrun were called, and anything that happened in the meantime, e.g. grompp?

#11 - 11/04/2016 02:50 AM - Chris Neale

I have been running like this:

ibrun -np $NP gmx_mpi mdrun -notunepme -deffnm ${NAME} -cpi ${NAME} -dlb yes -npme $NPME -cpt 60 -multi $NREP -replex 1000 -hrex -plumed plumed.dat -nsteps $NSTEPS -noappend

Note that this version of gromacs is patched with plumed (so that I can do replica exchange with solute tempering), and I just realized that this may have affected things. I'll test without plumed and report back if there is any difference.

Perhaps this is not the best way to do it? I just figured that it was easier to use mdrun -nsteps than remaking the .tpr files each time.

Also, just to be clear, as far as I can tell there is no problem with multiple exchange attempts, but the opposite, namely that at the interface between two runs there is no exchange attempt. As far as REMD is concerned, I believe that either multiple exchange attempts or a missed exchange attempt do not cause problems with the run itself, they just complicate demuxing. I actually don't use demuxing for much except making pretty movies for presentations (which I can accept to lose) and to compute the replica mobility (which I really don't want to lose). I think one should be able to get the later with the existing demux script, but that one will need to do something special with scripting at the interface between runs before going to

demux.pl (and that may be complicated if there were any crashes since one probably does not want to insert anything there.

Thanks for your help Mark,
Chris.

#12 - 11/04/2016 02:58 AM - Chris Neale

Note that I am explicitly running a number of steps that is a multiple of the exchange interval (in the previous example, -replex 1000 and -nsteps 40000000. I did consider doing something like exchanging every 1000 steps and running with mdrun -nsteps 40000001 , but I figured this would be even harder to deconvolute since even without crashes I would have to figure out the run interfaces that ended with nsteps as a multiple of -replex and those that did not.

Files

tpr_files.tar.gz	68.4 KB	04/06/2016	Maud Jusot
output_files.tar.gz	5.05 MB	04/06/2016	Maud Jusot