GROMACS - Bug #2473

mdrun sometimes stalls due to large coordinates with no constraints

03/29/2018 10:31 PM - Szilárd Páll

Status:	New		
Priority:	Normal		
Assignee:			
Category:	mdrun		
Target version:			
Affected version - extra info:	2018.1-dev-20180329-72f2827	**Difficulty:**	uncategorized
Affected version:	2018.1		

Description

A user report indicates that some runs can stall:
https://mailman-1.sys.kth.se/pipermail/gromacs.org_gmx-users/2018-March/119513.html

My testing showed that in some cases, with the provided input, runs would stall occasionally without finishing.

So far the backtrace I could obtain is:

```
#0  0x00007fcf456a0d22 in ?? () from /usr/lib/x86_64-linux-gnu/libgomp.so.1
#1  0x00007fcf4569f889 in ?? () from /usr/lib/x86_64-linux-gnu/libgomp.so.1
#2  0x000000000094848b in do_force_cutsVERLET(_IO_FILE*, t_commrec*, t_inputrec*, long, t_nrnb*, g
mx_wallcycle*, gmx_localtop_t*, gmx_groups_t*, float (*) [3], gmx::ArrayRef<gmx::BasicVector<float
> >, history_t*, gmx::ArrayRef<gmx::BasicVector<float> >, float (*) [3], t_mdatoms*, gmx_enerdata_
t*, t_fcdata*, float*, t_graph*, t_forcerec*, interaction_const_t*, gmx_vsite_t*, float*, double,
gmx_edsam*, int, int, DdOpenBalanceRegionBeforeForceComputation, DdCloseBalanceRegionAfterForceCom
putation) [clone .isra.39] ()
#3  0x0000000000949e62 in do_force(_IO_FILE*, t_commrec*, t_inputrec*, long, t_nrnb*, gmx_wallcycl
e*, gmx_localtop_t*, gmx_groups_t*, float (*) [3], gmx::ArrayRef<gmx::BasicVector<float> >, histor
y_t*, gmx::ArrayRef<gmx::BasicVector<float> >, float (*) [3], t_mdatoms*, gmx_enerdata_t*, t_fcdat
a*, gmx::ArrayRef<float>, t_graph*, t_forcerec*, gmx_vsite_t*, float*, double, gmx_edsam*, int, in
t, DdOpenBalanceRegionBeforeForceComputation, DdCloseBalanceRegionAfterForceComputation) ()
#4  0x000000000041785d in gmx::do_md(_IO_FILE*, t_commrec*, gmx::MDLogger const&, int, t_filenm co
nst*, gmx_output_env_t const*, MdrunOptions const&, gmx_vsite_t*, gmx_constr*, gmx::IMDOutputProvi
der*, t_inputrec*, gmx_mtop_t*, t_fcdata*, t_state*, ObservablesHistory*, gmx::MDAtoms*, t_nrnb*,
gmx_wallcycle*, t_forcerec*, ReplicaExchangeParameters const&, gmx_membed_t*, gmx_walltime_account
ing*) ()
#5  0x0000000000434af0 in gmx::Mdrunner::mdrunner() ()
#6  0x000000000041da43 in gmx::Mdrunner::mainFunction(int, char**) ()
#7  0x000000000041e3b3 in gmx_mdrun(int, char**) ()
#8  0x0000000000441c63 in gmx::CommandLineModuleManager::run(int, char**) ()
#9  0x000000000040ea6c in main ()
```

History

#1 - 03/29/2018 10:31 PM - Szilárd Páll

- Subject changed from mdrun sometimes stalling to mdrun stalls

#2 - 03/29/2018 10:31 PM - Szilárd Páll

- Subject changed from mdrun stalls to mdrun sometimes stalls

#3 - 03/29/2018 10:50 PM - Szilárd Páll

Got a backtrace with symbols, but this seems to indicate that mdrun is stalled in put_atoms_in_box() which does not make sense for now -- might need to sleep on it.

```
(gdb) i threads
  Id   Target Id         Frame
* 1    Thread 0x7f9f10ceb7c0 (LWP 15535) "gmx" put_atoms_in_box (ePBC=<optimized out>, box=0x37b6ee4, x=...) a
t /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1463
  2    Thread 0x7f9f004cf700 (LWP 15539) "gmx" 0x00007f9f076a78c8 in accept4 (fd=12, addr=..., addr_len=0x7f9f
```

```
004ced98, flags=524288) at ../sysdeps/unix/sysv/linux/accept4.c:40
  3    Thread 0x7f9effcce700 (LWP 15544) "gmx" 0x00007f9f0769a74d in poll () at ../sysdeps/unix/syscall-templa
te.S:84
  4    Thread 0x7f9eff44c700 (LWP 15545) "gmx" pthread_cond_wait@@GLIBC_2.3.2 () at ../sysdeps/unix/sysv/linux
/x86_64/pthread_cond_wait.S:185
  5    Thread 0x7f9efec4b700 (LWP 15546) "gmx" put_atoms_in_box (ePBC=<optimized out>, box=0x37b6ee4, x=...) a
t /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1467
  6    Thread 0x7f9efe44a700 (LWP 15547) "gmx" 0x00000000004b0730 in put_atoms_in_box (ePBC=<optimized out>, b
ox=0x37b6ee4, x=...) at /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1467
  7    Thread 0x7f9efdc49700 (LWP 15548) "gmx" put_atoms_in_box (ePBC=<optimized out>, box=0x37b6ee4, x=...) a
t /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1465
  8    Thread 0x7f9efd448700 (LWP 15549) "gmx" 0x00000000004b0730 in put_atoms_in_box (ePBC=<optimized out>, b
ox=0x37b6ee4, x=...) at /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1467
  9    Thread 0x7f9efcc47700 (LWP 15550) "gmx" put_atoms_in_box (ePBC=<optimized out>, box=0x37b6ee4, x=...) a
t /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1465
  10   Thread 0x7f9ef7fff700 (LWP 15551) "gmx" put_atoms_in_box (ePBC=<optimized out>, box=0x37b6ee4, x=...) a
t /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1465
  11   Thread 0x7f9ef77fe700 (LWP 15552) "gmx" put_atoms_in_box (ePBC=<optimized out>, box=0x37b6ee4, x=...) a
t /home/pszilard/projects/gromacs/gromacs-18/src/gromacs/pbcutil/pbc.cpp:1465
```

#4 - 04/03/2018 09:27 AM - Berk Hess

This is likely because of very large, or possibly non-finite, coordinates, which lead to extremely high while loop counts.

#5 - 04/03/2018 07:05 PM - Szilárd Páll

Berk Hess wrote:

> This is likely because of very large, or possibly non-finite, coordinates, which lead to extremely high while loop counts.

That's likely, though I thought a run would crash before this can happen.

I gave up as I didn't have non-optimized build, but I can have a closer look.

#6 - 04/03/2018 07:30 PM - Szilárd Páll

Just noticed: the setup does no constraints, that's why the issue propagates and the run does not crash.

I assume the large coordinates mean the system is not stable, right?

Can we have some low-cost runtime check to avoid this?

#7 - 04/03/2018 08:39 PM - Szilárd Páll

- Subject changed from mdrun sometimes stalls to mdrun sometimes stalls due to large coordinates with no constraints

Confirmed -- for some reason in debug mode the system took far more tries to get to this stage:

```
(gdb) up
#1  0x00000000004b2666 in put_atoms_in_box (ePBC=0, box=0x492bef4, x=...) at /home/pszilard/projects/gromacs/g
romacs-18/src/gromacs/pbcutil/pbc.cpp:1465
1465                        x[i][d] += box[d][d];
(gdb) p x[i]
$6 = (gmx::BasicVector<float> &) @0x495c988: {x_ = {-1.47941133e+09, -2.80910336e+09, 1.6737783e+09}}
```

I assume we can consider this instability -- the question is if we can / is it worth trying to catch?

#8 - 04/03/2018 09:25 PM - Berk Hess

It would be nice to catch large forces, also to give a nicer warning than e.g. the particles moved more than ... in PME. But the question is what the cost of performing a check every step is.

#9 - 04/04/2018 04:44 PM - Mark Abraham

We'd want a way to implement this that might be cheap enough to run every step. Watching the energy doesn't help, because particles a long way away contribute less potential energy.

Sum all the positions and if the total changes by more than a factor of 10 then we have a problem?

#10 - 04/04/2018 10:15 PM - Berk Hess

Summing is not more efficient than a per atom check and not better either.
We can check if displacement in a step is not more than e.g. the cut-off length. This can be done using f*invmass or v. But the catch all issues we would need to do this every step, which might lead to measurable overhead.

Files

topol.tpr	1.59 MB	03/29/2018	Szilárd Páll