GROMACS - Task #1793

cleanup of integration loop

08/03/2015 10:56 PM - Mark Abraham

Status: New
Priority: Normal
Assignee: Mark Abraham
Category: mdrun
Target version: future
Difficulty: uncategorized

Description
The integrator loop has been in need of work for a long time (and it’s not just the fault of the velocity-Verlet implementation), which we have been slowly cleaning up (see https://gerrit.gromacs.org/#/q/topic:md-loop-cleanup for just some of those patches).

This should help
- find and fix bugs
- make it reasonable to implement and document old and new integration schemes well (e.g. #1137),
- modularize code (e.g. constraint, integrator, coupling algorithm, non-bonded scheme, and global-summation code should know as little about each other as possible)
- pave the road for future task parallelism at this level (currently every function call gets passed pretty much everything, which means we have no idea what the data dependencies are, and bugs could be anywhere)
- provide optimization opportunities (e.g. with leap-frog, at nsttcouple-1 steps we should be able to avoid using a blocking MPI_Allreduce to get the KE for use at the next step, but currently the necessary conditions are unclear)
- avoid pessimizations like #692
- make clear where developers of new features should add code

Subtasks:
- Task # 2423: modernize constraints code
- Task # 2492: implement force calculation via ForceProviders containing collections of I...
- Task # 2644: Replace compute_globals

Related issues:
- Related to GROMACS - Feature #1137: Proposal for integrator framework (do_md)... New
- Related to GROMACS - Bug #1858: compute globals should not have logic about w... Closed
- Related to GROMACS - Task #1868: implement mdrun -rerun better, simplifying d... Closed
- Related to GROMACS - Bug #2105: multi-domain rerun broken Closed
- Related to GROMACS - Task #2616: Model for MD state New

Associated revisions
Revision 7bc8d6d2 - 10/09/2015 12:59 PM - Mark Abraham
Remove CGLO_RERUNMD

Only one use of cglo_flags combines more than a single flag, so made cglo_flags specific to that case.

No functionality changes here

Refs #1793

Change-Id: I10dbae9efdd3f8da340b7efec08305d5de563f9c

Revision 6ead809b - 11/16/2015 03:22 PM - Mark Abraham
Remove "support" for twin-range with VV integrators

Group-scheme twin-range non-bonded interactions never worked with VV+constraints, and removing it was a to-do item. There are no plans to make it work with VV, and there are plans to remove the twin-range scheme entirely, as well as rework leap-frog more closely into the Trotter scheme.
Removed state_global from mdrun energy-summation

state_global was used only when MD_READ_EKIN was set, i.e. when restoring KE data when restarting from checkpoints. So there is no use passing it to every call to compute_globals.

Accordingly, moved use of restore_ekinstate_from_state().

Also added some docs and const correctness.

Simplified energy-summation code

We need to check that the DD has the correct number of bonded interactions being handled across all domains. The old check would happen at every call for computing energies, which was unnecessary because the number of domain-local bonds can change only at DD steps. It also required log file, global state and global topology to be passed into every call to global_stat to handle the rare error case.

This patch moves the check for correct DD of bonded interactions up to do_md, where it can be scheduled to happen only once per DD lifetime. global stat now does not need to be passed parameters that only needed for reporting in the case of a failed check.

The dispersion correction code now gets the global number of atoms from a field in forcerec. This should really be set up when a dispersion-correction object was created, but this approach will do for now (particularly because dispersion correction code is probably being called too often, so there's bigger problems to fix).

Together, mdrun energy-summation code no longer needs to be passed the global topology, reducing code complexity.

Fixed documentation of global_stat.

Fix logic for DD missing-interactions check

The logic for the active part of this check got broken in 90aa9d65e0c3 before v4.5. Historically, there was no significant effect, because all that patch changed is that we sum a double whose result won't always be read. For all integrators, there's eventually a call where CGLO_ENERGY is set and the sum is read, so the check is active and works correctly. Thus, the worst-case result was that interaction(s) were missing and not flagged until the next global-energies communication step.

However, cleanup for #1793 moved the responsibility for the check out to do_md, exposing the fact that for VV integrators (only), CGLO_ENERGY isn't set for the “global signalling + leapfrog” call to compute Globals. Thus the sum wasn't read, so the check failed when totalNumberOfBondedInteractions still had the value -1 with which it was initialized. Apparently, the regression tests don't have enough coverage of VV integrators to find this.

This also made me realise that this DD check has also been inactive for the calls to compute Globals after initial DD, DD after replica exchange, and DD during reruns, because none of those cases set
Fixes #1882. Refs #1793.
Change-id: l0b5cc448175873ec0e5cee3c3d5023654b4f1b27

Revision f3c239f6 - 02/14/2017 08:17 PM - Mark Abraham
Fix handling of previous virials

These quantities get written to checkpoint files only for the Trotter pressure-coupling integrators that need them, but they were being copied in do_md for all Trotter integrators. This meant that an appending restart of md-vv plus nose-hoover plus no pressure coupling truncated off a correct edr frame and wrote one with zero virial and wrong pressure. And in the same case, a no-append restart writes a duplicate frame that does not agree with the one written before termination.

Fixed by copying them only when they are useful to copy, so that in the problem case, the correctly computed post-restart virial is not wiped with zeroes from state fields that were never read from the checkpoint. Cases that use the previous-virial values are not affected.

Refs #1793
Change-id: l84908122aefbe8658f423eaf4e5bd4ae25a93d24

Revision 280ba5a3 - 03/28/2018 01:49 PM - Mark Abraham
Free more memory in grompp and mdrun

This will be increasingly useful as we start adding more GoogleTest cases that call grompp and mdrun repeatedly.

Refs #1793, #1868
Change-id: lc0faa4c3dec2891d0dc6166561f8ccd4391f44

Revision ad79f778 - 04/05/2018 07:28 PM - Mark Abraham
Create gromacs/mdrun module and move code there

Improving how we dispatch the work for do_md and friends requires that they all compile as part of libgromacs (or all outside of it). The code currently in src/programs/mdrun is there simply because it hasn't been moved from the old src/kernel layout, rather than through any clear design, so we should move it now that we have a reason.

Other code from src/programs/mdrun now needs to move into libgromacs also, but we currently don't have a module for miscellaneous mdrun features, so we're adding to the abuse of mdrun for now.

Refs #1793
Refs #2423
Change-id: l4f9ad5d0bf658675472b49b2d565f588b7214e

Revision 61f895f - 04/05/2018 07:41 PM - Mark Abraham
Simplified handling of simulation runners

Arguments to runners are now passed via struct members. That struct is expected to change over the medium term, as most of its contents should either be in a container of modules, or data with such a module. As such, it does not make sense to do a lot of coding to manage its contents in a sound way. The practical effects of using this struct are almost the same as the old approach of passing a forest of arguments - some variables are declared in a scope, initialized with values, then used once.

The form is chosen so that

- the do_md() etc. functions do not need a large number of useless
textual changes now

- future changes to names and types of the former parameters do not need to be edited in matching way in multiple places of code and their doxygen
- we don't need to mark things gmx_unused anywhere

Useless unchecked return values have been removed.

Some excessive includes in integrator.h were removed, which generates a few minor fixes elsewhere.

Refs #1793
Change-Id: i678598175192c9c88113fd79fcee17f8e5c504e

Revision 1ab524ff - 04/12/2018 09:29 AM - Mark Abraham
Reorganize energy evaluation for EM

Used aggregate initialized structs to simplify future refactoring.

Refs #1793, #2423
Change-Id: led8bf3b4cb2d7f238c4b674ce4f9564a37f6a20

Revision acc3940a - 04/17/2018 10:03 AM - Mark Abraham
Break apart update_constraints

There are four distinct kinds of work being done, and never was any call to update_constraints doing all of them, so it's better to have a group of functions, each of which do one thing, and the relevant ones called. This also makes it simpler to express by returning fast that when we don't have constraints, we do nothing.

Made the logic for whether this is a log or energy step match that of the main MD loop. The old implementation may not have prepared for the last step correctly when it was triggered by something other than the nsteps inputrec value.

Removed a comment mentioning iteration, which is a feature that was removed a while ago.

Removed some ancient debug dump output.

Refs #2423, #1793
Change-Id: l21c10826721ddc9a79a33b1dc79971a20d0855d9

Revision e9a53455 - 04/17/2018 10:03 AM - Mark Abraham
Shift per-step control logic to do_md

There are multiple reasons why do_md can decide this is the last step, so we should be consistent about it.

Refs #1793
Change-Id: l46f377630fa12be482cbb532c05d96efe0537523

Revision 5ae5bf42 - 04/25/2018 02:12 PM - Mark Abraham
Refactor SD update

The former use of multiple boolean control variables made the logic hard to follow. Without constraints, the two parts of the integrator are fused, which is now expressed explicitly.

This means it is now clear that the second half of the sd update does not compute anything from the forces.

Removed some of the vestiges of the way we once had two SD integrators.

Refs #2423, #1793
Move IMD initialization and fuse DD setup

init_IMD does not use the local state, and only the global state positions on master rank, so we can move it before the local state initialization.

This permits us to fuse some DD setup stages

Removed inaccurate comment about DD partitioning.

Refs #1793

Fix assertions for SD update

Some of the assertion logic in the refactoring of 5ae5bf42159cf was wrong, but we apparently have no tests of the SD integrator before now.

Refs #1793

Introduce reset handler

This change introduces a reset handler which encapsulates the setting and handling of resetting signals, and the actual counter resetting. This cleans up the do_md loop, exchanging several lines of code by a single command. It also makes a step towards a task-based design by only calling the internal functions when necessary (if counter resetting was requested, signal setting only on master).

Refs #1793

Introduce checkpoint handler

This change introduces a checkpoint handler which encapsulates the setting and handling of checkpoint signals, and owns the bCPT boolean. This cleans up the do_md loop, exchanging several lines of code by a single command. It also makes a step towards a task-based design by only calling the internal functions when necessary (if checkpointing is enabled, signal setting only on master), and by owning as much data as possible.

Refs #1793

Introduce stop handler

This change introduces a stop handler which encapsulates the setting and handling of stop signals, and owns the bLastStep boolean (which is, however, currently mirrored as a local variable in do_md for convenience). This cleans up the do_md loop, exchanging several lines of code by a single command. To set the signal, the StopHandler object loops over a vector of stop condition functions registered previously with a helper object of type StopHandlerHelper. This allows to formulate different conditions, and only build them if the current setup requires them (e.g. only have a stop criterion based on time if a maximal run time was set, only build stop conditions on master rank, ...).
The stop handler itself is created in do_md by a helper object owned by the calling code. The helper object offers an interface to register stop conditions via std::function objects. This allows higher-level code to inject stop conditions.

Refs #1793

Change-Id: ia9077841d96a9bc6d6f000eef4093a9fbd9f5363

Revision 1b1abdba - 12/10/2018 04:50 AM - Pascal Merz

Failproof signal conversion

When introducing the signal handlers, we decided to use scoped enums to define the different simulation signals (changes Ia90778, I05ca7a, I5dec05). In the SimulationSignal objects which actually get reduced, these are stored as signed char. When the signals get handled, these signed char are converted back to the scoped enums. Currently, this is done using a static_cast.

Although in the current code, the signals get handled immediately after being reduced, and signals are only set by master, there is no formal guarantee that this is always true. If the signals get reduced multiple times, or by different ranks, the signed char stored in the SimulationSignal object could get larger than +1 / -1. In the old code, this would not lead to problems, as it was just checked that the unsigned char in the SimulationSignal was != 0 (or <0 / >0, for the stop signal). In the new code, the signal handling would fail in this case, without proper error message - the static_cast will not fail, but the following comparison to the enum values will always fail. This commit introduces a conversion function for each of the enums and explicit enum values for the StopSignal to return to the failproof behavior of the old code.

Refs #1793

Change-Id: lac8ea3946effba1797e16d8c918c7d49ce4dc828

Revision efa13a69 - 08/27/2019 06:51 PM - Mark Abraham

Add integration tests for exact restarts

These tests demonstrates the extent to which mdrun checkpoint restarts reproduce the same run that would have taken place without the restart.

I've been working on these, and the bugs they exposed, for a few years, but the code has been fixed for a few years now.

The tests don't run with OpenCL because they have caused driver out of memory issues.

Refs #1137, #1793, #1882, #1883

Change-Id: l8bc441d945f13158bbe10f097e772ea87cc6a559

History

#1 - 08/03/2015 11:08 PM - Mark Abraham

- Related to Feature #1137: Proposal for integrator framework (do_md) in future GROMACS added

#2 - 08/03/2015 11:11 PM - Mark Abraham

- Description updated

#3 - 10/07/2015 08:34 PM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: f10dbaebef1dd3f88da340b7flec08305d56e5639c
Gerrit URL: https://gerrit.gromacs.org/5165

#4 - 10/07/2015 09:33 PM - Gerrit Code Review Bot

Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: I937f397602c761c92cd8e48dd6041a0966e4a094
Gerrit URL: https://gerrit.gromacs.org/5188

#7 - 11/24/2015 11:58 PM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: I195f5ad7a679f397754931602ca643aaf85d47ba1d
Gerrit URL: https://gerrit.gromacs.org/5379

#8 - 11/25/2015 12:08 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: I195f5ad7a679f397754931602ca643aaf85d47ba1d
Gerrit URL: https://gerrit.gromacs.org/5380

#9 - 11/25/2015 12:31 AM - Mark Abraham
- Related to Bug #1858: compute globals should not have logic about which integrator is in use added

#10 - 01/05/2016 02:19 PM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: I0b5cc448175873ed05e5cee3c35023654b2f1b27
Gerrit URL: https://gerrit.gromacs.org/5525

#11 - 01/31/2017 03:02 PM - Mark Abraham
- Related to Task #1868: implement mdrun -rerun better, simplifying do_md added

#12 - 01/31/2017 03:03 PM - Mark Abraham
- Related to Bug #2105: multi-domain rerun broken added

#13 - 02/09/2017 06:02 PM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: b3f695d06585675472b49b2d5654f588b7214e
Gerrit URL: https://gerrit.gromacs.org/6467

#14 - 03/02/2018 11:34 PM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: d1d8f5c706b6585675472b49b2d5654f588b7214e
Gerrit URL: https://gerrit.gromacs.org/7648

#15 - 03/03/2018 09:45 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Change-Id: af9ad5d06585675472b49b2d5654f588b7214e
Gerrit URL: https://gerrit.gromacs.org/7649

#16 - 03/13/2018 01:24 PM - Gerrit Code Review Bot
Gerrit received a related patchset '2' for Issue #1793.
Uploader: Mark Abraham (mark.j.abraham@gmail.com)
Gerrit URL: https://gerrit.gromacs.org/8578

#28 - 10/31/2018 10:49 PM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1793.
Uploader: Pascal Merz (pascal.merz@colorado.edu)
Change-IID: gromacs~release-2019~Iac8ea3946effba1797e16d8c918c7d49ce4dc828
Gerrit URL: https://gerrit.gromacs.org/8636