GROMACS - Feature #2890

Feature # 2816 (New): GPU offload / optimization for update&constraits, buffer ops and multi-gpu communication

GPU Halo Exchange

03/12/2019 02:00 PM - Alan Gray

Status:	New
Priority:	High
Assignee:	
Category:	
Target version:	2020
Difficulty:	uncategorized

Description

When utilizing multiple GPUs with PP domain decomposition, halo data must be exchanged between PP tasks. Currently, this is routed through the host CPUs: D2H transfer; buffer packing on CPU; host-side MPI; and same in reverse. This is true for both the position and force buffers.

Instead, we can transfer data directly between GPU memory spaces using GPU peer-to-peer communication. Modern MPI implementations are CUDA-aware and support this, or D2D cudaMemcopies can be used directly when in threadMPI mode. The complication is that we need to pack buffers directly on the GPU. This is done by first preparing (on the CPU) a mapping of array indices involved in the exchange, which can then be used in GPU buffer packing/unpacking kernels. This mapping array only needs to populated and transferred during neighbor search steps.

Limitation: still only supports 1D data decomposition. Higher numbers of dimensions should be relatively straightforward to implement by extending the existing developments, but this still requires design and testing.

TODO: implement support for higher numbers of dimensions.
TODO: integrate call to force buffer halo exchange, when force buffer ops
patches accepted.

Subtasks:

Task # 3089: relax dlb scaling limit when that would suit GPU halo exchange	**Closed**
Task # 3092: implement better receiver ready / notify in halo exchange	**New**
Task # 3093: rework GPU direct halo-exchange related force reduction complexities	**New**
Task # 3104: implement GPU DD cycle counting	**New**
Task # 3106: Implement multiple pulses with GPU communication	**New**
Task # 3156: move ddUsesGpuDirectCommunication and related conditionals into the worklo...	**New**

Related issues:

Related to GROMACS - Feature #3052: GPU virial reduction/calculation	**New**
Related to GROMACS - Feature #3087: enable GPU peer to peer access	**New**
Related to GROMACS - Bug #3100: crash with GPU comm DD	**In Progress**
Related to GROMACS - Task #3082: move launch/synchronization points to clarif...	**New**
Related to GROMACS - Task #3171: schedule CPU H2D force contribution in separ...	**New**

Associated revisions

Revision 8f42be1e - 08/16/2019 11:29 AM - Alan Gray

GPU halo exchange

Activate with GMX_GPU_DD_COMMS environment variable.

Class to initialize and apply halo exchange functionality directly on
GPU memory space.

Fully operational for position buffer. Functionality also present for
force buffer, but not yet called (awaiting acceptance of force buffer
ops patches).

Data transfer for halo exchange is wrapped and has 2 implementations:
cuda-aware MPI (default with "real" MPI) and direct cuda memcpy
(default with thread MPI). With the latter, the P2P path will be taken

if the hardware support it, otherwise D2H,H2D.

Limitation: still only supports 1D data decomposition

TODO: implement support for higher numbers of dimensions.
TODO: integrate call to force buffer halo exchange, when force buffer ops patches accepted.

Implements part of #2890
Associated with #2915

Change-Id: I8e6473481ad4d943df78d7019681bfa821bd5798

Revision 44f607d7 - 09/16/2019 03:12 PM - Alan Gray

GPU halo exchange

Activate with GMX_GPU_DD_COMMS and GMX_USE_GPU_BUFFER_OPS environment variable.

Class to initialize and apply coordinate buffer halo exchange functionality directly on GPU memory space.

Currently only supports direct cuda memcpy, and relies on thread MPI being in use.

Updated gpucomm testing matrices to cover non-GPU case.

Limitation: still only supports thread MPI, 1D data decomposition and only coordinate halo exchange

Implements part of #2890
Associated with #2915

Change-Id: I8e6473481ad4d943df78d7019681bfa821bd5798

Revision 54c24729 - 09/18/2019 03:36 PM - Alan Gray

GPU Force Halo Exchange

Activate with GMX_GPU_DD_COMMS environment variable.

Extends GPU Halo exchange feature to provide GPU Force halo exchange functionality. Does not yet support virial steps, which require an extra shift force reduction - these are currently performed on the non-buffer ops / non direct-comm path. Also has same limitations as coordinate halo exchange.

Performs part of #2890. Future work to improve synchronization towards a more one-sided scheme (#3092) and to make depenencies more explicit (#3093)

Change-Id: Ifc23cc8db2655f7258e68b34e7cdc7b71994e1e8

Revision 4bc2605d - 10/17/2019 03:57 PM - Szilárd Páll

Eliminate spurious GPU->CPU copy

When GPU direct communication-based halo exchange is used, non-local forces should not be copied back to the CPU prior to halo exchange.
This commit removes the incorrectly unconditional leftover copy.

Fixes/Improves #2890
Refs #3156
Change-Id: I25e521204f30da1e257232e9117c3fe4f0a83b08

History

#1 - 03/12/2019 02:37 PM - Alan Gray

- Target version set to 2020

#2 - 03/12/2019 02:39 PM - Gerrit Code Review Bot

Gerrit received a related patchset '8' for Issue #2890.
Uploader: Alan Gray (alang@nvidia.com)

Change-Id: gromacs~master~I8e6473481ad4d943df78d7019681bfa821bd5798
Gerrit URL: https://gerrit.gromacs.org/9225

#3 - 03/12/2019 02:41 PM - Alan Gray

- Description updated

#4 - 03/12/2019 02:46 PM - Alan Gray

- Description updated

#5 - 03/13/2019 11:55 AM - Alan Gray

Position halo exchange patch awaiting review.

#6 - 05/28/2019 12:22 PM - Alan Gray

- Description updated

#7 - 05/28/2019 12:23 PM - Alan Gray

- Description updated

#8 - 08/01/2019 05:30 PM - Szilárd Páll

- Related to Feature #3052: GPU virial reduction/calculation added

#9 - 08/01/2019 05:36 PM - Szilárd Páll

Note that the f exchange needs to consider #3052. Short-term solution could be to have the direct communication disabled on virial steps, but a final solution should ideally avoid such complexities in control code.

#10 - 09/12/2019 04:24 PM - Szilárd Páll

- Related to Feature #3087: enable GPU peer to peer access added

#11 - 09/24/2019 03:16 PM - Szilárd Páll

- Related to Bug #3100: crash with GPU comm DD added

#12 - 10/11/2019 05:26 PM - Szilárd Páll

- Related to Task #3082: move launch/synchronization points to clarify task dependencies added

#13 - 10/11/2019 05:57 PM - Szilárd Páll

@Alan, can you please update the status of the outstanding TODOs and subtasks, in particular #3093 and #3082 and likely #2965 are quite important and have not received updates since filed. Additionally, extensive testing for robustness & performance are the high-prio tasks -- will file a subtask for the latter too. I think these are necessary ingredients to be able to promote this feature from dev-feature to stable user-facing feature.
Have you had further thoughts on whether there are any use-cases (e.g. old CPU/chipset or complex PCI-E subsystem) where a possible automation may have to switch off direct GPU comm?
I realize that the thorough testing will have to happen close to and post-beta2 considering that plumbing is still ongoing.

Additionally, single pulse communication limit, as we realized, can result in fragile behavior and requires a lot of support-scaffolding to make it possible (and safe) to use the GPU DD feature in its current form. We think that we have the necessary restrictions in place (we might want to add an extra check that might abort the run) but this aspect will require thorough testing, i.e. running multi-rank highly imbalanced runs with DLB on.
Ideally we would like this limitation addressed, but I realize this is unlikely to happen before a final realease as we have no code yet (I assume?) and other tasks have higher priority. I also suspect such an addition would require a change set that is just too large a risk for breakage that post-beta2. If so, we should however consider it (together with the 1D DD limitation) among the first candidate improvements post-release.

#14 - 10/22/2019 12:20 AM - Szilárd Páll

- Related to Task #3171: schedule CPU H2D force contribution in separate stream added