GROMACS - Feature #3029

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

Feature # 2817 (In Progress): GPU X/F buffer ops

GPU force buffer ops + reduction

07/05/2019 07:38 PM - Szilárd Páll

Status:	In Progress
Priority:	High
Assignee:	
Category:	mdrun
Target version:	2020-beta3
Difficulty:	uncategorized

Description

Implement the force buffer layout transform on GPU (from nbmxn to native layout); as in case of the CPU, this tasks could in some cases be combined with the force reduction to produce the final force-buffer that the integration can take as input.

Use-cases (TODO list all use-cases/subcases):

1. buffer ops transform-only on GPU * optional D2H transfer of force buffer (by default transfer)
2. reduce with other on-GPU forces (currently PME) * needs efficient on-device sync * needs to make sure that the reduced forces are not D2H trasferred * needs to consider virial calculation (to be done on the bonded + nonbonded forces)
3. reduction with CPU-side forces * ensure that CPU-side forces H2D is issued as early as possible * also consider virial contribution * evaluate whether this is more efficient to do when DD halo exchange is anyway doing staged copy and forces are already in cache (and need to be in CPU cache for MPI);

General considerations:

- launch the transform kernel back-to-back after the nonbonded rather than later, next to the CPU buffer ops/reduction
- evaluate what is #atoms threshold under which it is not worth taking the 10-15 us overhead of kernel launch (especially for non-local buffer ops)
- implement on-GPU event sync between PME and reduction task in the nonbonded queue (see fully functional prototype for the coordianate dependency shared earlier here: https://gerrit.gromacs.org/c/gromacs/+/8588)

Subtasks:

Feature # 3052: GPU virial reduction/calculation	**New**
Task # 3128: do not fall back to CPU path on energy-only steps	**Closed**
Feature # 3142: centralize and clarify GPU force buffer clearing	**In Progress**
Task # 3170: investigate GPU f buffer ops use cases	**New**

Associated revisions

Revision 0a4ca2c4 - 08/15/2019 04:22 PM - Alan Gray

PME reduction for CUDA F buffer operations

Enable with GMX_USE_GPU_BUFFER_OPS env variable.

Provides functionality to perform reduction of PME forces in F buffer ops kernel. Currently active when single GPU performs both PME and PP (multi-GPU support will follow in patch which perfoms PME/PP comms direct between GPUs). When active, Device->Host copy of PME force and CPU-side reduction is disabled.

Implements part of #3029, refs #2817

Change-Id: I3e66b6919c1e86bf0bed42b74136f8694626910b

Revision c5bcd713 - 09/12/2019 05:08 PM - Szilárd Páll

Reorganize on-GPU PME force reduction flag handling

Instead of passing around a flag everywhere that tells PME whether forces are reduced on GPU or CPU (and whether transfer needs to happen

for the latter), we pass the flag once when configuring PME for the next step and store it internally.

Refs #3029

Change-Id: I81fa2dc93dd979e2b85b4d7fe8cf266a3fde9b8f

Revision 889b6f9a - 09/30/2019 09:18 AM - Szilárd Páll

Make the wait on PME GPU results conditional

When the PME forces are reduced on-GPU and no energy/virial output is produced, we can avoid blocking waiting on the CPU for the PME GPU taks to complete.

This however would break the timing accounting which needs to happen after PME tasks completed. Hence the accounting is moved to the PME output clearing.

Refs #3029, #2817

Change-Id: I4e7f3aa43754a187fe5d6b584803444967516958

History

#1 - 07/05/2019 10:57 PM - Szilárd Páll

Szilárd Páll wrote:

- evaluate what is #atoms threshold under which it is not worth taking the 10-15 us overhead of kernel launch (especially for non-local buffer ops)

Quick test on a Xeon E5 2620v4 gives ~25k atoms as crossover point for F buffer ops (12-14 us), i.e. ~3000 atoms/core (for rather slow cores). This means that at least in the strong-scaling regime, if we don't use direct GPU communication, at least the nonlocal buffer ops would be better off on the CPU.

#2 - 08/12/2019 06:44 PM - Szilárd Páll

- Description updated

#3 - 09/24/2019 06:52 PM - Mark Abraham

- Target version changed from 2020-beta1 to 2020-beta2

#4 - 11/01/2019 03:23 PM - Paul Bauer

- Target version changed from 2020-beta2 to 2020-beta3

bump