GROMACS planning meeting (spring 2023)

29 March 2023, 13:00 - 15:00 CEST
Zoom link (NEW)

Common themes to discuss if we have time:

- Python APIs
- Large-scale I/O (100,000 jobs, etc.), relation to file formats & time+space efficient I/O
- Portability/packaging:
    - Building multi-SIMD binaries again (how to combine with SIMD c++ code?)
    - Separate (DSO-loaded) extension library for each GPU architecture
- Scaling limitations (definitely with Plumed, unclear for colvars) due to single-threaded central and blocking I/O
-

Sebastian Wingbermühle:

- Complete plans for more versatile free energy/enhanced sampling in GROMACS:
    - https://gitlab.com/gromacs/gromacs/-/issues/4014
    - https://gitlab.com/gromacs/gromacs/-/issues/4431
    - https://gitlab.com/gromacs/gromacs/-/issues/4432
- This dev-cycle:
    - Focus on FEP/REST2 support (second half of 2023)
- Large-scale testing of RBFE with AWH
- Automated workflow for RBFE calculations with GROMACS

Berk Hess:

- Integrate constant pH, https://gitlab.com/gromacs/gromacs/-/issues/4273
    - Refactoring + test of potential calculation (Berk)
    - Lambda dynamics code (Pavel?)
    - Module + coupling work

- PDB mmCIF format support - required since PDB will stop publishing new structures in old format.

Erik Lindahl:

- Plan MDDB work (2 new postdocs):
    - File formats, compression, integrating metadata into trajectories, interfacing with the database being built in Barcelona. This might also provide the resources we need to move all our text I/O to JSON/YAML. Possibly some relation to Mark's comment about I/O threads (which is a good idea).
    - Based on discussions with Anders Ynnerman at LiU, look into B-tree format trajectories to enable very rapid loading of data after selecting the requested temporal and spatial resolution (required for pulling 50fps for domes, etc., and very large systems).
- Debug whole-cell simulations. Still not working for non-trivial systems.
- Discussion item: Using C++20 for module isolation. Despite our slight hesitation, I think it's far better that we jump in and write for the future instead of having to redo things later (which gets extra complex when related to APIs - we don't want to have to support an old one too for 5+ years).

Mark Abraham:

- SYCL tuning
    - Based on feedback on smaller projects, perhaps replace use of barriers for inter-queue synchronization with event-based sync
- I/O thread - throughput-style multi-sim use can use GPUs more efficiently with fewer simulations per GPU if the I/O for each simulation doesn't block GPU submission.
    - Need to see what status of perf for multi-task OpenMP libraries is
    - Complications with affinity handling by mdrun. Do tasks with e.g. different thread counts get a reliable set of hardware threads
    - New kind of error scenario
- Multi-sim GPU-resident runs - use the CPU as a coprocessor for pair search and I/O threads
- Severely optimistic: start working on GPU pairlist update (probably set up initially on CPU, and thereafter just move particles between sets within wider and wider radii)

Magnus Lundborg:

- AWH target distribution optimization based on AWH friction metric.
    - Almost finished.

- SD integrator updates and constraints on GPU
    - Working prototype.
    - Can hopefully be finished early autumn.

Hubert Santuz, Jérôme Hénin, Giacomo Fiorin (Colvars)

We'll be present to discuss MdModules and other issues related to Colvars integration.
Summary of the discussion:

- Will submit an MR for an ensemble temperature notifier
- will double-check a NS step callback as per a conversation with Berk
- When all the environment is in place, will submit a single MR for the Colvars library, to be placed under external/
- We are willing to be part of the conversation on the future of output files, if all the output moves to TNG, so that Colvars and other MdModules work smoothly.

Szilárd Páll

- SYCL: [1]
    - Performance: improve hipSYCL/AMD GPU performance / scaling
    - Portability: broader implementation support
- PME decomposition follow-up #3884 and related [2]
- DD / PP parallelization: GPU DLB, NVSHMEM, consider kernel fusion [3]
- Continue improving the md loop / schedule modularity (more use of workload flags, refactor/move search/DD) [4]
- Graph scheduling: enable CPU tasks and MPI communication (related to [4]) [5]

Alan Gray (NVIDIA):

- Strong overlap with Szilárd (above).
- NVSHMEM GPU Communications, as a faster alternative to MPI on multi-node
- Single-node multi-GPU enhancements with thread-MPI
- Energy/Power investigation and optimization
- CUDA Graph enhancements - extend to support CPU forces and libMPI/NVSHMEM (depends on MD loop refactoring)

Andrey Alekseenko

- Improving SYCL performance
    - LUMI/Dardel, primarily
    - working with runtime developers on overhead

- Refactoring DeviceContext to have the same meaning on all platforms
- Other refactoring / clean-up of GPU glue code now that most features are in place

Eric Irrgang

I don't expect to attend the meeting. I don't have any feature/functional goals, so my contributions for the remainder of the year will depend in large part on specific project-level goals. If the team can agree that something I can contribute to is a priority and can identify reviewers (or otherwise express that effort is allocated to help contributions to be merged), then please let me know how important the following things are (relative priority), and I can follow up with interested stakeholders.

Things I can offer to do:
* Update gmxapi to require Python 3.9, and remove workarounds for a bunch of deprecations.
* Add automated testing for Python 3.11, 3.12
* Add cross-version API compatibility testing.
* Migrate to the CPython stable ABI
* Add management of the Python Global Interpreter Lock to allow threaded concurrency for gmxapi tasks.
* Clean up the Python metaprogramming that makes gmxapi implementation code hard to read.
* Express public GROMACS data exchange (in terms of DLPack) so we can normalize language-agnostic multidimensional binary data.
* Re-share the demonstration of a more API-friendly Trajectory Analysis Framework Runner, with Python bindings.
* Re-share updates to the sample_restraint that reduce boilerplate with much more concise C++ templates for expressing inputs/outputs, which may also serve as an example of how we could update the gmx::Options frameworks

Peter Kasson (as estimated by Eric Irrgang)

May not have received the invite in time to attend, but is still interested in helping to evolve extensible MDModules and public ForceProvider API.

Alessandra Villa

- Workshop code in GROMACS
- Developing training materials and maintaining them
- Webpage "news" section
- Looking a community - planning for user survey.

Farzaneh Jalalypour

- New tutorials (e.g. How to restrain ion channel pore, how to develop a tool and exploit GROMACS cpp template)- with Alessandra
- code in GMX (implementing Perturbation method as analysis tool)
- Release membrane protein simulation tutorial and create a tag in GROMACS forum to answer users questions

Vedran Miletić

- Complete variable unshadowing effort
- Add two more stop conditions, possibly expression-based
    - “Distance”
    - “Location”