GPU-accelerated Higher Representations of Wilson Fermions with `HiRep`

Erik Kjellgren Sofie Martins * Emiliano Molinaro Claudio Pica Antonio Rago

*martinss@imada.sdu.dk

Abstract

We are improving one of the available lattice Software packages `HiRep` by crucially adding GPU acceleration. This development is accompanied by an overall Software quality improvement in the build system, testing, and documentation, adding features for both CPUs and GPUs. The software is available under `https://github.com/claudiopica/HiRep` in the branch `HiRep-CUDA` and will soon be merged into `master`.

Motivation

Exploring physics beyond the standard model is limited by the computational expense of simulating strongly coupled theories on the lattice. Using Wilson Fermions constitutes to this day one of the cheapest options to explore theories with different gauge groups. In particular, simulation of theories with larger numbers of colors at sufficient precision requires a large amount of computational resources. We can improve our possibilities for exploring parameter spaces here by making use of state-of-the-art NVIDIA GPU accelerators.

1 Features

Linear Algebra Operations

- GPU-accelerated collection of linear algebra operations
- Single and double precision templates
- Reaches peak performance of the GPUs tested (NVIDIA V100, A100)

Wilson-Dirac Operator

- GPU-accelerated operator `Dphi`
- Single and double precision
- Even-odd preconditioning
- Clover improvement
- Exponential clover improvement, improving inversion performance [5]

$$D_{\mathrm{sw}} = \sum_x (4+m_0)\exp(A(x)); A(x) = -\frac{c_{\mathrm{sw}}}{4(4+m_0)}\sum_{\mu,\nu}\sigma_{\mu\nu}F_{\mu\nu}(x)$$

Inverters

Tested and available for GPUs

- CG Multishift
- BiCGstab
- γ_5-QMR
- SAP+GCR [8, 7] (in progress)

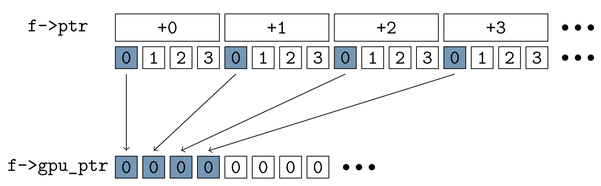
Other features in progress

- High-quality random number generation on the GPU with parameter optimized RANLUX [6]
- Support for AMD GPUs
- HMC

2 Implementation Details

2.1 GPU Geometry

Optimal memory access patterns require to reorder the way we save our field data



2.2 Communications

In strided layout sendbuffers are not coalesced. This required us to completely rewrite the geometry allocating an additional sendbuffer and synchronizing from the field before communications.

Old Geometry

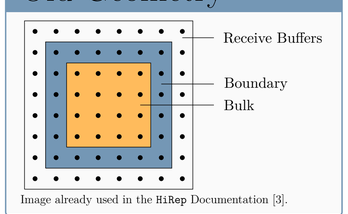
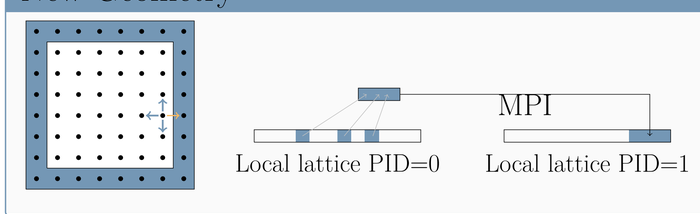


Image already used in the `HiRep` Documentation [3].

New Geometry



- GPU communications have larger message sizes due to larger local lattice sizes per PU
- Necessary to develop strategies for communication reduction
- Saved factor two in communications by partial application of Dirac operator before communications

Note that the ideas for the communication reduction by a partial application of the Dirac operator and the masking mechanism used for the new geometry are ideas that are reused from `OpenQCD`, [4].

3 Software Quality

Github CI

- Unit tests for large amout of compilation variables
- Code coverage reporting with `codecov`
- Automated documentation generation for github pages
- Code formatting check

Github pages

- User Manual
- Developer Handbook
- `Doxygen` function reference

Usability

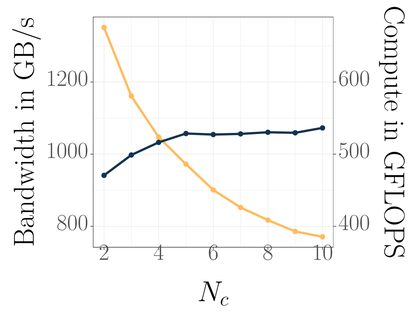
- Simple build setup, helptext with information on all compilation variables
- Very fast compilation with `ninja`, independent of gauge group
- Largely self-contained, no dependencies aside from `gcc`, `ninja`, `CUDA`, MPI, `perl`
- Automatic consideration of hardware topology using `hwloc`
- New benchmarking code available

3.1 Benchmarks

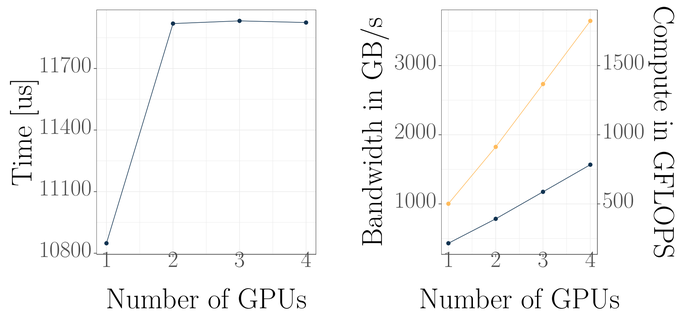
Large-N_c Scaling

These Large-N_c tests have been performed on a 32^4 lattice on a single NVIDIA V100 GPU for different N_c in the fundamental representation.

- Hopping term kernel execution at larger-N_c should be compute bound
- Large-N_c behavior is not ideal yet
- This is already a large-N_c-improved experimental kernel.



Inter-node benchmarks



- Hopping term execution time approx. 9% overhead going from a single GPU to multiple GPUs
- Multi-GPU software scales ideally here

These tests have been performed on a 32×48^3 local lattice on a single node with up to 4 NVIDIA V100 GPUs connected via NVLink for $N_c = 3$ with fermions in the fundamental representation. Compute performance is given everywhere in terms of double-precision floating point operations.

4 Summary and Outlook

The ported code is performant and scales well. Substantial improvements in overall software quality have been made. More features are on the way.

References

[1] John Cheng, Max Grossman, Ty McKercher, and Barbara Chapman. *Professional CUDA C programming*. Wrox, Indianapolis, Indiana, 2014.

[2] Luigi Del Debbio, Agostino Patella, and Claudio Pica. Higher representations on the lattice: Numerical simulations. SU(2) with adjoint fermions. *Phys. Rev. D*, 81:094503, 2010.

[3] Claudio Pica et. al. Hirep. `https://github.com/claudiopica/HiRep`.

[4] Martin Lüscher et. al. Openqcd. `https://luscher.web.cern.ch/luscher/openQCD/`.

[5] Anthony Francis, Patrick Fritzsch, Martin Lüscher, and Antonio Rago. Master-field simulations of O(a)-improved lattice QCD: Algorithms, stability and exactness. *Comput. Phys. Commun.*, 255:107355, 2020.

[6] Martin Luscher. A Portable high quality random number generator for lattice field theory simulations. *Comput. Phys. Commun.*, 79:100–110, 1994.

[7] Martin Luscher. Lattice QCD and the Schwarz alternating procedure. *JHEP*, 05:052, 2003.

[8] Martin Luscher. Solution of the Dirac equation in lattice QCD using a domain decomposition method. *Comput. Phys. Commun.*, 156:209–220, 2004.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement № 813942. GPU development was possible using GPUs supplied by the UCloud interactive HPC system, which is managed by the eScience Center at the University of Southern Denmark