Lattice QCD on Blue Waters

Justin Foley

The MILC Collaboration

Blue Waters NEIS-P2 Symposium
A more descriptive title might be

“Toward improved linear solvers for large-scale multi-GPU calculations in Lattice QCD”
Fundamental interactions in nature:
- Gravity
- Electromagnetism
- Weak Nuclear Interaction
- Strong Nuclear Interaction
Quantum Chromodynamics (QCD) is the theory of the Strong Interaction.

QCD is a Relativistic Quantum Field Theory - Quantum Mechanics + Special Relativity.

It describes the interaction of fundamental matter particles called quarks and force carriers called gluons.

Analogy with electromagnetism: quark ⇔ electron, gluon ⇔ photon (but behavior is very different).

Quarks bind together to form protons and neutrons ⇒ atomic nuclei.
Why QCD is interesting

- Applications in Nuclear Physics
- Heavy-ion physics / physics of the early universe
- Search for new physics beyond the Standard Model
- Similarities with condensed matter systems - graphene, cold atoms
In general, QCD is not amenable to analytic methods.

Wilson 1974 - solve QCD on a computer.

Space and time is approximated by a 4D lattice, quarks are associated with lattice sites and gluons reside on the links between sites.

Multiple discretization schemes in use. MILC uses the Highly-Improved Staggered Quark (HISQ) formalism.
Most ($\gtrsim 70\%$) time in a lattice calculation is spent solving the linear system

$$A\phi = \eta$$

(1)

- $\phi, \eta$ are quark fields, and $A$ is a sparse matrix
- In state-of-the-art calculations, $\text{rank}(A) \approx 10^9$
- In the HISQ formalism, $A = Q^\dagger Q$, where $Q$ is the HISQ matrix, with stencil

(17 points in 4 dimensions)
Solve $A\phi = \eta$ using iterative Krylov-subspace methods.

In the HISQ formalism, $A$ is Hermitian positive-definite and Conjugate Gradient is the Krylov method of choice.

In fact, for this particular system, the residual decreases monotonically.
- **QUDA**: An opensource library for QCD on Nvidia GPUs
  - [Lattice.github.com/quda](https://lattice.github.com/quda)
- Written in C++ and CUDA
- Linear-solver support for multiple lattice formulations
- QUDA linear-solve performance on a $36^4$ lattice on a single K20X is 160 Gflops for single- and mixed-precision CG and 80 Gflops for double-precision CG
- Mixed double/single-precision solver uses reliable updating (Sleijpen and van der Vorst)
This performance is not sustained on large numbers of GPUs.

The lattice is decomposed into regular subdomains, which are assigned to different GPUs.

Each application of $A$ involves the exchange of data between GPUs ($Q$ involves communication of quark field in a boundary region three-lattice sites wide).

In practice, linear solves are communication bound.
Reducing inter-processor communication

- **Domain decomposition:**
  Solve the preconditioned linear system

\[ \mathbf{M} \mathbf{A} \phi = \mathbf{M} \eta, \]

where \( \mathbf{M} \approx \mathbf{A}^{-1} \), but involves less or no inter-processor communication [Additive Schwarz method, Schwarz alternating procedure].

- Reduce number of applications of \( \mathbf{A} \) and hence inter-GPU communication
- Only ever need to evaluate matrix-vector products \( \mathbf{M} \rho \)
To compute $M\rho$, use iterative solver to evaluate $A^{-1}\rho$ on each lattice subdomain ignoring interprocessor communication

$$M\rho = \sum_{i=1}^{N_D} A_i^{-1} \rho_i$$

Dirichlet boundary conditions on each subdomain $\Rightarrow \kappa(A_i) < \kappa(A)$

Since $M$ is a preconditioner, implement approximately (half-precision data types, small number of inner-solver iterations, use only a subset of points in the HISQ stencil)

$$M\rho \approx \sum_{i=1}^{N_D} A_i^{-1} \rho_i$$

Use MR or steepest-descent algorithm in the preconditioning step.
Non-overlapped domain-decomposition reduces inter-processor communication by 40 to 50 percent.

![Graph showing the number of outer-solver iterations/Standard CG iterations for different numbers of GPUs and quark iterations. The graph includes lines for light quark (5 and 10 MR iterations) and strange quark (5 and 10 MR iterations).]
...which translates into a 30% reduction in solve times on large numbers of GPUs ($\gtrsim 1024$)

Linear solves at the strange-quark mass on a $96^3 \times 192$ lattice

- Non-overlapped domain-decomposed CG
- Unpreconditioned CG
However, better value at lower numbers of GPUs, where simple Additive Schwarz wins you little

Can we improve on this?
In the preconditioner, overlap domains to mitigate boundary effects.

However, the (restricted) preconditioning operator is not Hermitian \( \Rightarrow \) cannot be used with CG.
The General Conjugate Residual (GCR) algorithm supports non-Hermitian matrices
but an iteration of GCR is more expensive ($\geq \times 2$) than an iteration of CG
Tests on $96^3 \times 192$ lattice on 256 ($1 \times 4 \times 4 \times 16$) GPUs
Domain overlap widths of 0, 2, and 4 lattice sites
$N_{krylov} = 60$ in GCR solver
Overlapping subdomains reduced number of outer-solver iterations (which involve communication) by a factor of 3.7
However, preconditioned GCR still lags behind CG
Have implemented Additive Schwarz preconditioning for the HISQ formalism in the QUDA QCD library.

Non-overlapped domains reduce inter-processor communication by about 30% on large (above-optimal) numbers of GPUs.

Overlapping domains further reduces inter-processor communication, but this improvement is offset by a large increase in arithmetic workload.
Ongoing and future work

- Optimal approach may involve limited inter-processor communication in preconditioning step
- An improved preconditioning scheme, which further lowers the condition number of the system, could facilitate half-precision data types in the outer-solver iteration
Performance of domain-decomposed CG solver vs. unpreconditioned CG solver

Linear solves at the strange-quark mass on a $96^3 \times 192$ lattice

- Compare to solve times on slide 14