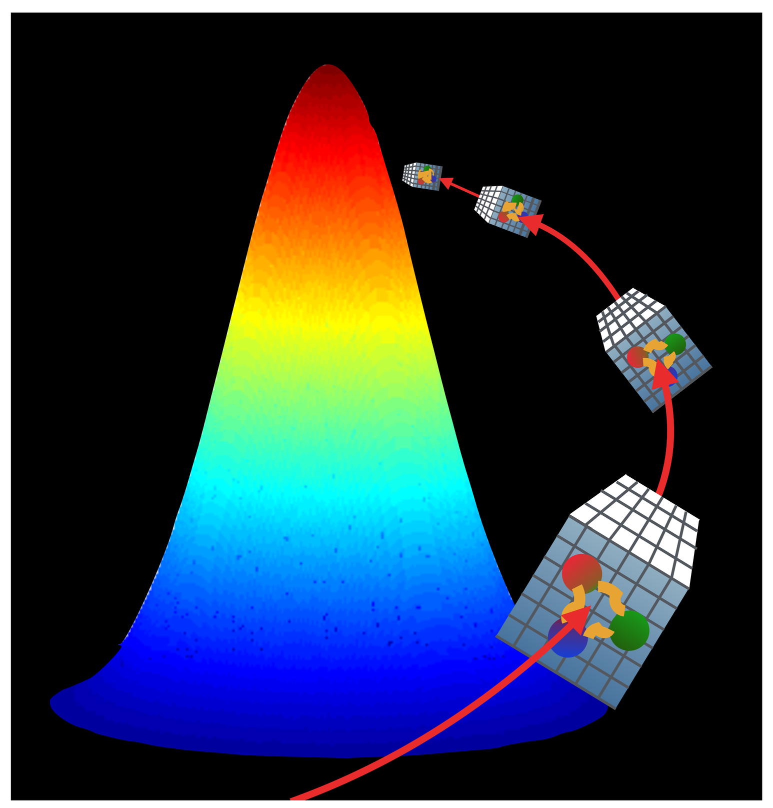
Accelerating Gauge Generation for Lattice QCD on Summit

Kate Clark (NVIDIA), Balint Joo (Jefferson Lab), Mathias Wagner (NVIDIA), Evan Weinberg (Boston University), Boram Yoon (Los Alamos National Laboratory)

mclark@nvidia.com, bjoo@jlab.org, mathiasw@nvidia.com, weinbe2@bu.edu, boram@lanl.gov

Introduction

The generation of gauge field configurations is a necessary first step to any Nuclear or High Energy Physics Calculation using Lattice Gauge Theory methods. The gauge configurations sample the strong force fields in the vacuum.



*Gauge fields must sample the QCD Equilibrium dictated by the Action (**S**) of the theory.*

$$P_{eq}(U) \propto e^{-S(U)}$$

Hybrid Molecular Dynamics Monte Carlo Methods generate new configurations by suggesting new trial states from old ones using Molecular Dynamics (MD). The trial states are subject to a Metropolis accept/reject test.

The MD defines canonically conjugate momenta (π) and integrates the (fictitious) time evolution of a Hamiltonian system with $H = T(p) + S(U)$. The integration scheme must be reversible and area-preserving to satisfy detailed balance.

Breaking the Determinant

The effect of dynamical sea quarks is to add determinant weights to $P_{eq}(U)$. These are simulated using pseudofermion terms in the action. A variety of terms can be combined to give the final action. The MD can be tuned by appropriate combinations of these terms to express the true flavor structure being simulated.

Two Flavor Terms:

$$\det\left[M_l^\dagger(U)\ M_l(U)\right] \longrightarrow e^{-\phi^\dagger\left[M_l^\dagger(U)\ M_l(U)\right]^{-1}\phi}$$

Single Flavor Terms by Rational Approximation:

$$\det\left[M_s(U)\right] \longrightarrow e^{-\phi^\dagger\left[M_s^\dagger(U)\ M_s(U)\right]^{-1/2}\phi}$$

$$\longrightarrow e^{-\phi^\dagger\ A\sum p_i\left[M_s^\dagger(U)\ M_s(U)+q_i\right]^{-1}\ \phi}$$

Preconditioning Ratio Terms:

$$\det\left[M_l^\dagger M_l\right] = \frac{\det\left[M_l^\dagger M_l\right]}{\det\left[M_1^\dagger M_1\right]}\frac{\det\left[M_1^\dagger M_2\right]}{\det\left[M_2^\dagger M_2\right]}\ldots\det\left[M_n^\dagger M_n\right]$$

Giving rise to: $$\frac{\det\left[M^\dagger M\right]}{\det\left[M_2^\dagger M_2\right]} \longrightarrow e^{-\phi^\dagger\ M_2\left[M^\dagger M\right]^{-1}M_2^\dagger\ \phi}$$

Force Terms and Solvers

Each pseudofermion component generates an MD force term. Evaluating these requires solving sparse linear systems with M.

$$\left(M^\dagger M\right)X = \phi \qquad \left(M^\dagger M + q_i\right)X_i = \phi$$

Two flavor solve typically done in two steps:

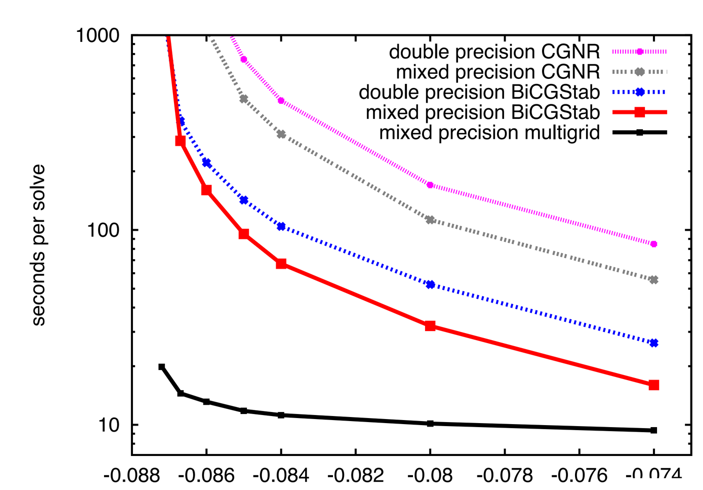
$$M^\dagger Y = \phi \quad \text{then} \quad MX = Y$$

using e.g. DD+GCR or prior to that BiCGStab. Further, solutions can be predicted from the previous solves along a trajectory

These solves are typically done with a Multi-shift Conjugate Gradients (MCG) solver. Single flavor terms have smaller forces than 2 flavor terms at the same mass. Hence previously we used 1+1 flavor to cancel the last term in a chain or ratios.

For light quarks, the system becomes ill-conditioned and the forces become large. We can tame the large forces by using ratio terms with longer time-steps. MCG gets all solutions in a single solve but does not scale very well on current inter-connect fabrics. Two flavor solves allow scalable & multi-grid preconditioners. Our code uses solvers which have been highly optimized for GPUs, implemented in the QUDA library.

Adaptive Aggregation Multi-Grid



Adaptive Aggregation Multi-Grid Solvers reduce critical slowing down with quark mass from J.C. Osborn et. al. arXiv: 1011.2775

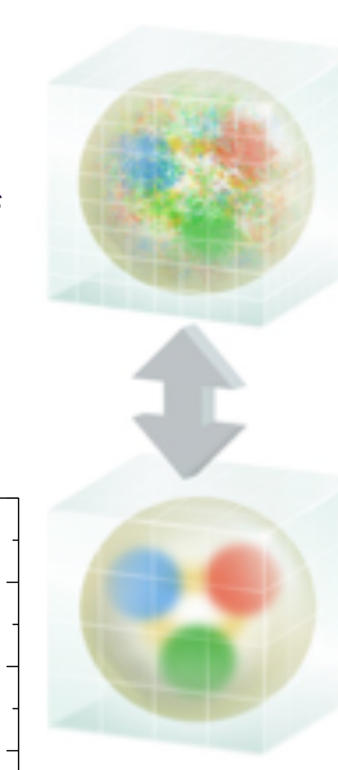
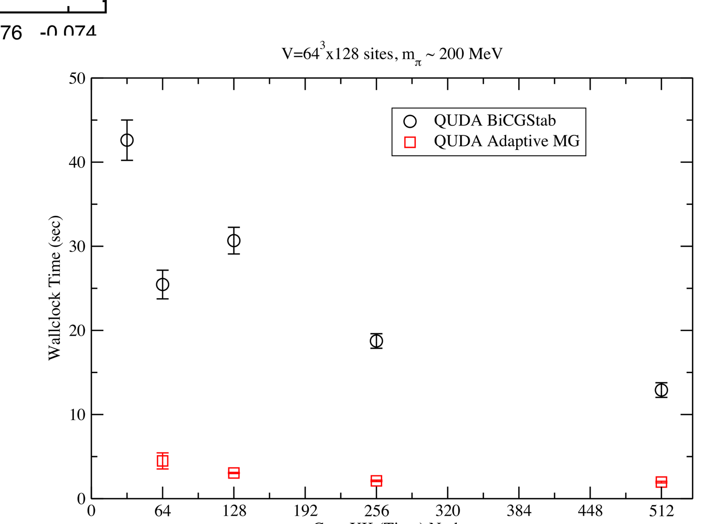


Image Credit: Joanna Griffin, Jefferson Lab Public Affairs

Adaptive Aggregation Multi Grid in the QUDA Library, data from Clark et. al. SC'16. Solver outperforms BiCGStab in QUDA by 7x-10x

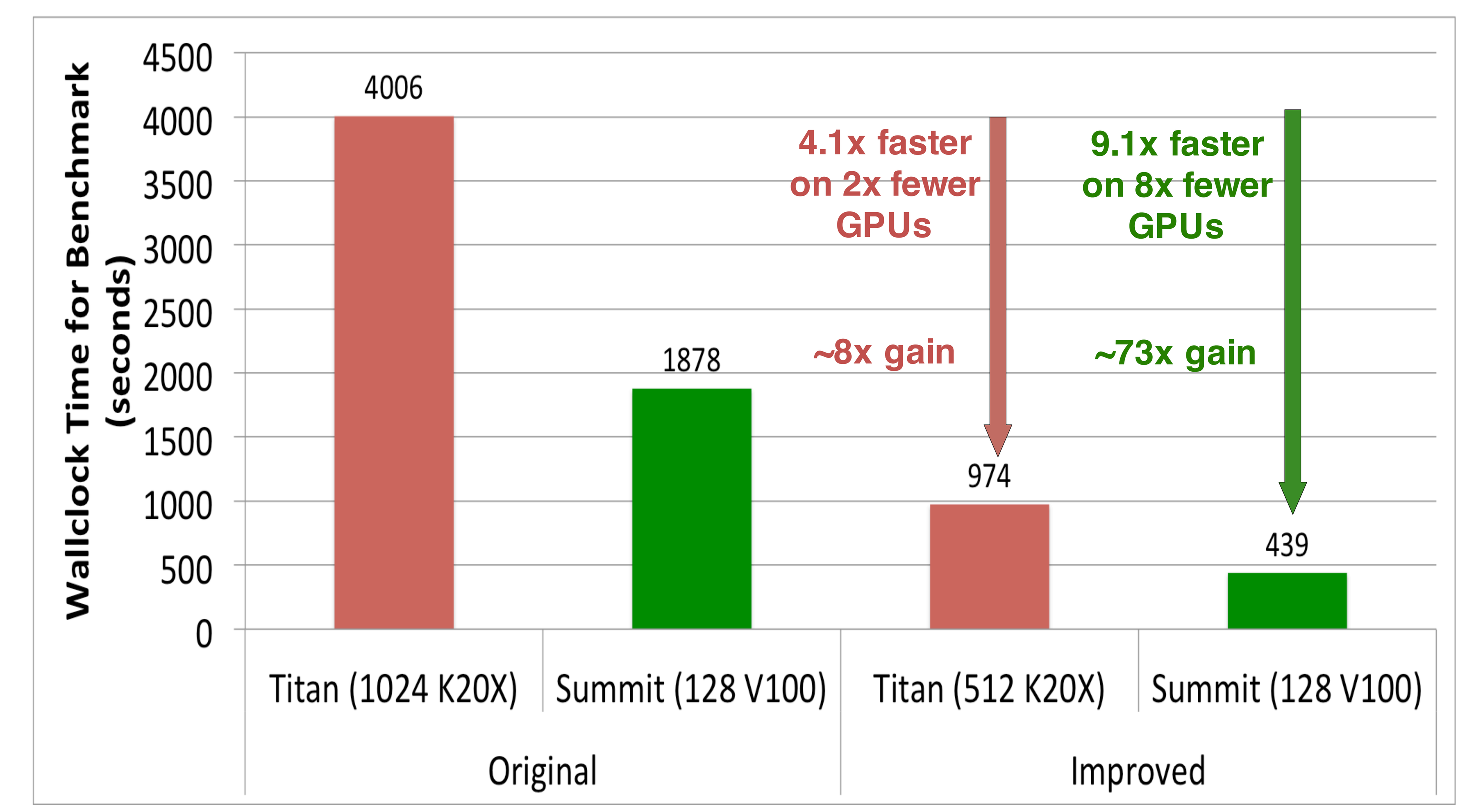


Adaptive Aggregation Multi-rid (AAMG) based solvers overcome critical slowing down of solvers with light quark masses, taming the cost of light ratio terms. However, AAMG requires a setup phase including *computing near null-space vectors*, and *constructing coarse operators*. For the method to be viable for gauge generation, these steps needed to be *moved to and optimized* on GPU. As the MD proceeds, gauge fields change and the preconditioner degrades. Rather then recomputing fully, we polish the null vectors with a fixed iteration count when an iteration threshold is exceeded

Other Optimizations

Apart from the MG solver, we further reduced cost by implementing a *force gradient integrator*. We integrated the *chronological predictor* from QUDA into Chroma. In our solvers we optimized the *pipeline-length*. We implemented *reduced (16-bit) precision* for halo exchanges and are testing a more aggressive reduction (8-bit) to *reduce bandwidth* needs on the network. We upgraded the QDP-JIT software to use LLVM-6.0 (trunk) to enable Volta and POWER9 optimizations.

Results



Trajectory times for a benchmark on Summit and Titan showing overall gains. Due to the power of 2 problem size, only 4 out of 6 GPUs were used on Summit nodes.

Conclusions & Outlook

The joint effect of algorithmic and architectural optimizations, retuning of the MD structure and raw architectural performance optimizations improved the GPU hour cost of a benchmark trajectory by 73x on Summit through a 9.1x wall clock time speedup on 8x fewer devices. The improvements, when fed back onto Titan reduced the previous cost by 8x. This is a *game changer for gauge generation.*

Acknowledgement

This work is supported by the US Department Of Energy, Office of Science, Offices of Nuclear Physics and Advanced Scientific Computing Research, through the SciDAC program under contract DE-AC05-06OR23177 under which JSA LLC operates and manages Jefferson Lab, and under the 17-SC-20-SC Exascale Computing Project.