MG Proto: A Multigrid Solver for x86 multicore Systems

Balint Joo (Jefferson Lab), Thorsten Kurth (NERSC)

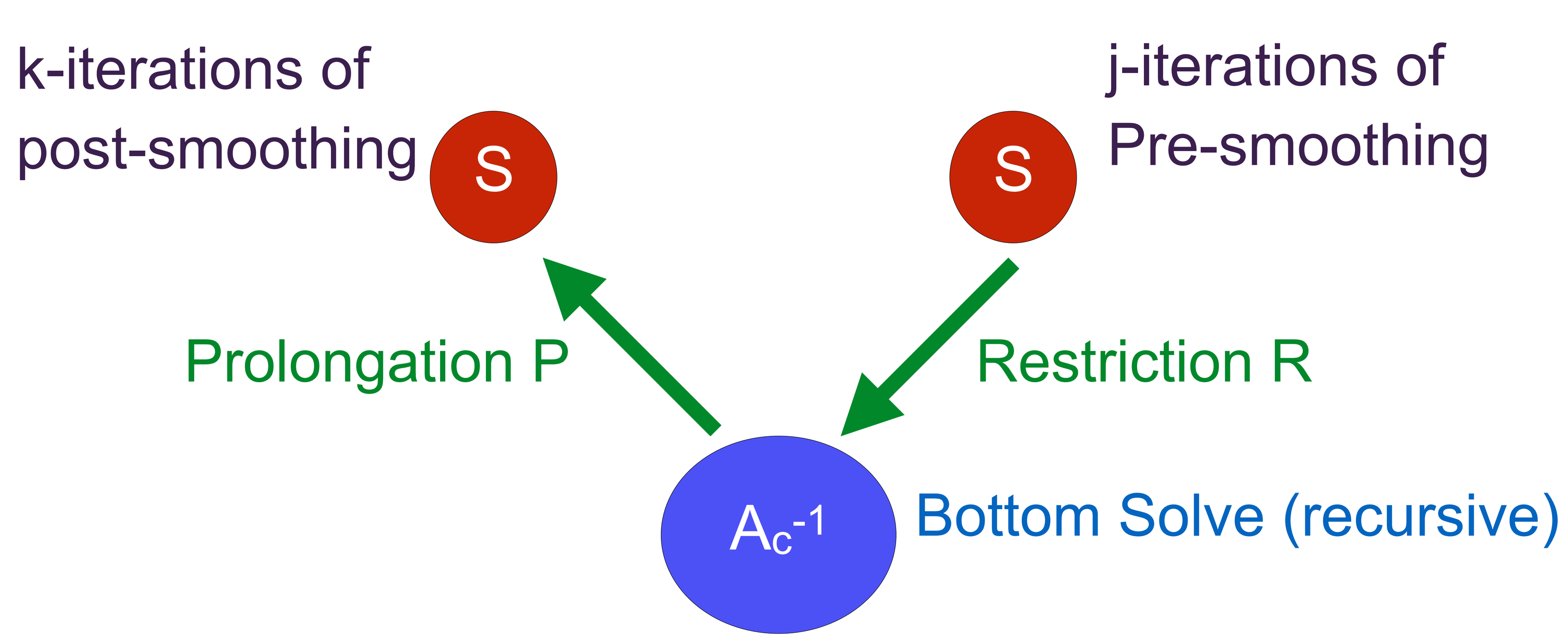
Introduction

Adaptive Aggregation based multi-grid (MG) methods [1,2,3] are becoming the standard for solvers in both the propagator calculation and recently even in the gauge generation parts of Lattice QCD calculations with Wilson Clover Fermions. The system solved is A x = b, where A is the Dirac operator

Multi-Grid Solvers

V-Cycle & K-Cycle

MG aims to reduce the short wavelength (UV) modes on a fine grid using a Smoother (S). The error due to the longer wavelength modes is solved on a coarser grid by solving with a coarsened operator (A^{-1}_c). A typical cycle is the V-cycle:



The error is reduced as (e.g. [2]):

$$e' \leftarrow (I - SA)^k (1 - PA_c^{-1}RA)(1 - SA)^j e_0$$

Near Null Space Block Aggregation

Low modes of the A are `self similar' on cubic-blocks of the lattice due to local coherence (weak approximation). Hence one way to define R is to aggregate the fine degrees of freedom over cubic blocks with near null-space vectors produced in a setup phase

Restriction: Aggregation over sites, colors, chiral spin components.

Fine Grid d.o.f : V_f x N_{spin} x N_{color}

Coarse Grid d.o.f : V_c x N_{chiral} x N_{null}

Coarse Operator

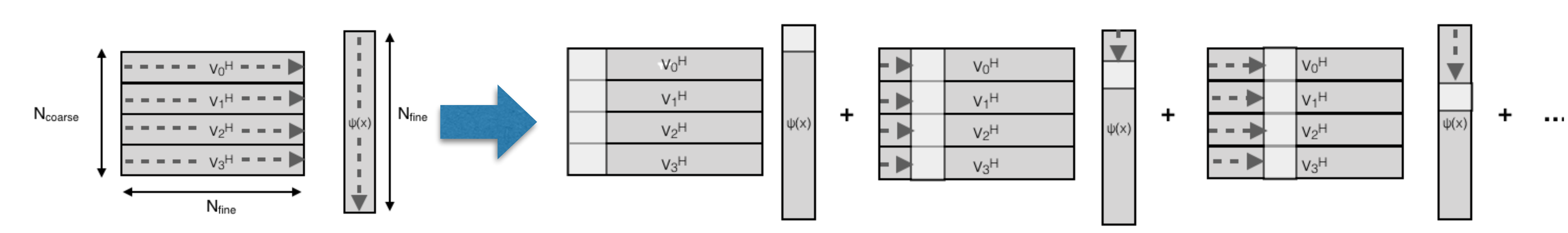
The resulting Coarse operator is a nearest-neighbor operator similar in structure to the fine operator:

$$A_c(x) = X_0(x) + \sum_{\mu=1..8} X_\mu(x)\ \delta_{x,x+\hat{\mu}}$$

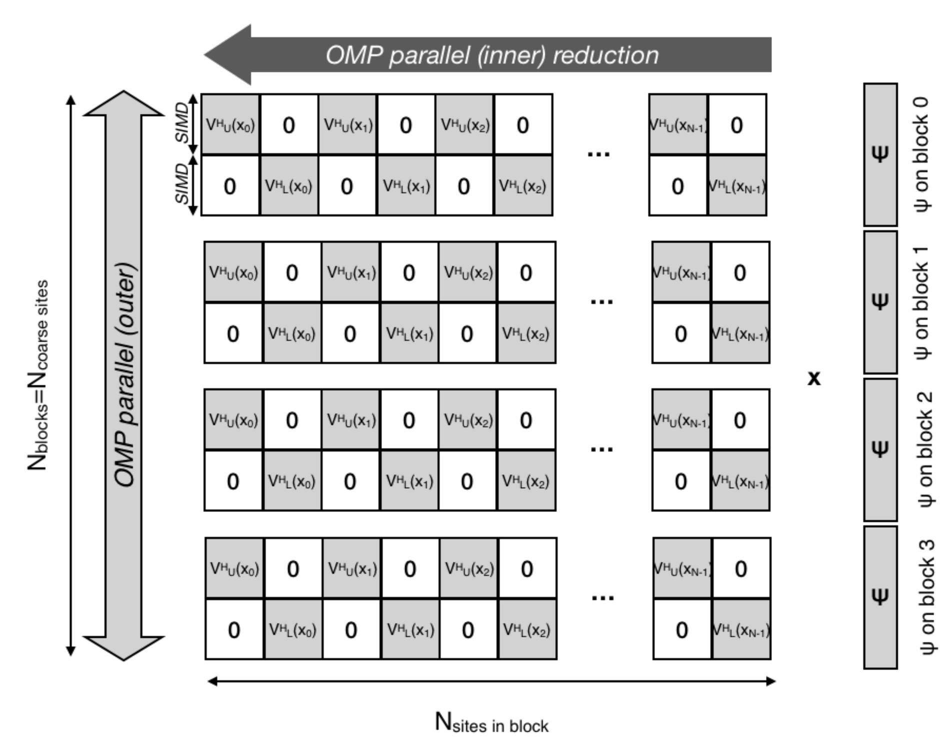
Where $X_0(x)$ and $X_\mu(x)$ are matrices of dimension N_{null}xN_{chiral}.

SIMD for matrix-vector operation

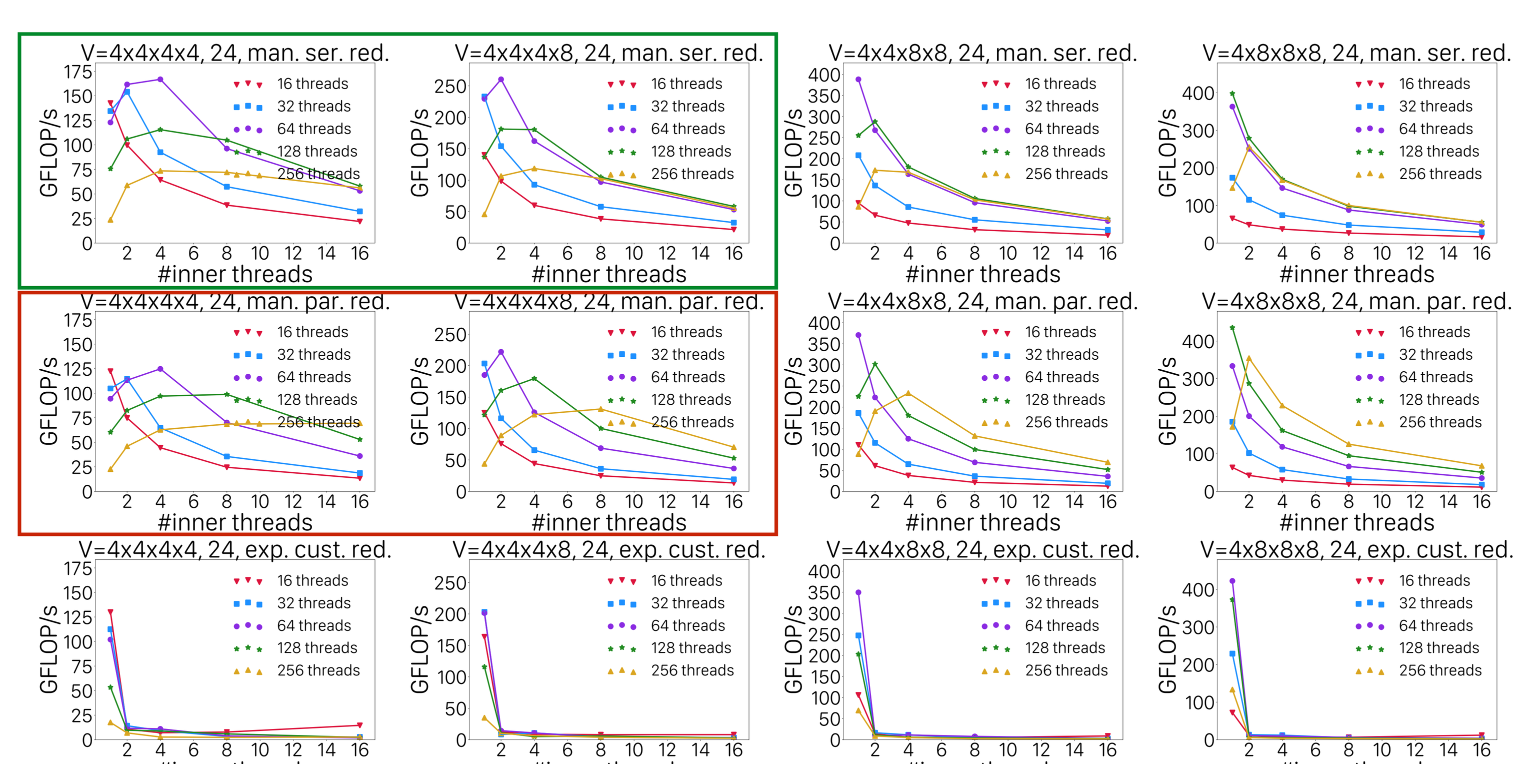
Applying Ac consists of 9 matrix vector multiplications. We restrict to N_{null} being a multiple of 8 and vectorize using AVX512 intrinsics:



Nested Parallelism for Aggregation



Aggregations for restriction and prolongation permit nested parallelism through a) parallelism over blocks and b) within blocks. Parallelism within blocks may be desirable if there are very few coarse sites (e.g. a coarse level with 16 sites, on a KNL system)

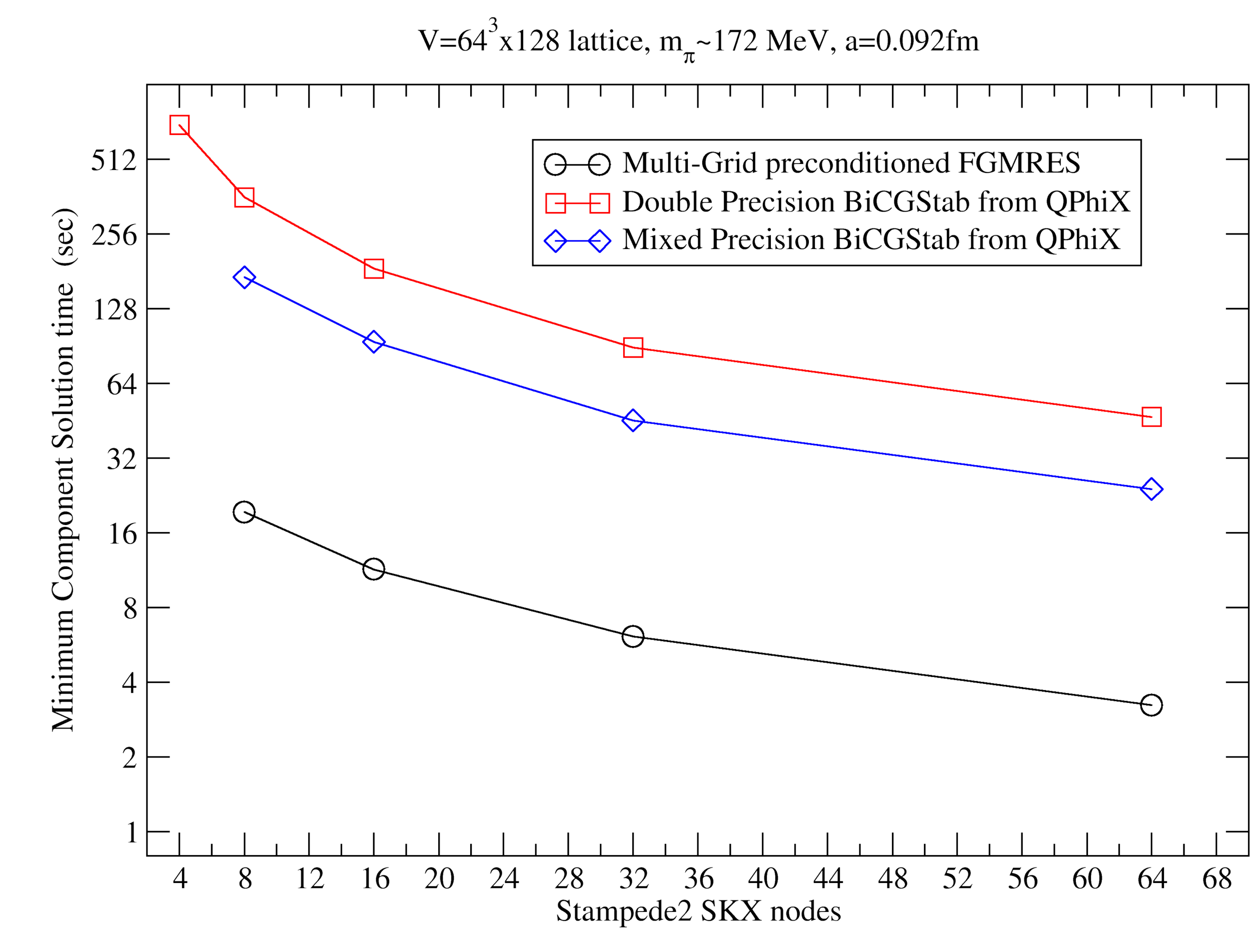


Utilizing threading within the site was only really beneficial when there was not enough parallelism via sites ($V=4^4$ and $V=4^3$x8 cases). In this instance benefit was visible when manual (man) implementation of nested parallelism was employed, rather than through explicit OpenMP nested (exp) parallelism. In these (man) cases serial reductions (ser. red.) in the blocks proved more efficient than parallel ones. [4]

Other optimizations

Our MGProto implementation uses the QPhiX library for threading and vectorization on the fine grid. In addition we have implemented Schur Decomposition based even-odd preconditioning In all the solvers used on all MG levels.

Performance Results



Performance results and strong scaling on Stampede 2 using Skylake (SKX) nodes Multigrid provides approximately an 8x reduction in solve time than the fastest available, mxsed precision BiCGStab solver from the QPhiX library. Similar performance improvements are also visible on KNL systems, e.g. Cori and Theta

Conclusions & Outlook

Our implementation delivers a roughly 8x improvement over our best previous solver for KNL and Skylake systems. Coincidentally, 64 nodes of Stampede performs similarly to 64 nodes of Titan in 2016 using QUDA-MG. This work opens Cori and Theta for propagator calculations and for gauge generation using multi-grid in the future. It also serves as a basis for performance portability explorations.

References

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.