Algorithms and Machines Review

Meifeng Lin
Brookhaven National Laboratory

July 22-28, 2018

First and foremost…

- Thanks to the organizers for giving me the opportunity to give this review.

- Many thanks to the speakers/authors who sent me information and presentation materials.

- My apologies for any omissions due to time. All mistakes are my own.

Outline

- **Overview**
- **Algorithms**
  - HMC
  - Solvers
- **Machines/Software**
  - Multi/Many-Core Processors
  - New/Emerging Architectures
  - GPUs (to be covered in depth by Mathias Wagner)
- **Summary and Outlook**

Contributors

Algorithms

HMC

- Ting-Wai Chiu
- Xiao-Yong Jin
- David Murphy
- Carsten Urbach

Solvers

- Kate Clark
- Chulwoo Jung
- Liam Keegan
- Bob Mawhinney
- Alessandro Nada
- Jiqun Tu

Machines

Many/Multi-Core Processors

- Peter Boyle
- Stephan Durr
- Issaku Kanamori
- Jarno Rantaharju
- Daniel Richtmann

New/Emerging Architectures

- Patrick Dreher
- Ming Gong
- Piotr Korcyl
- Nils Meyer

Workflow/Job Management

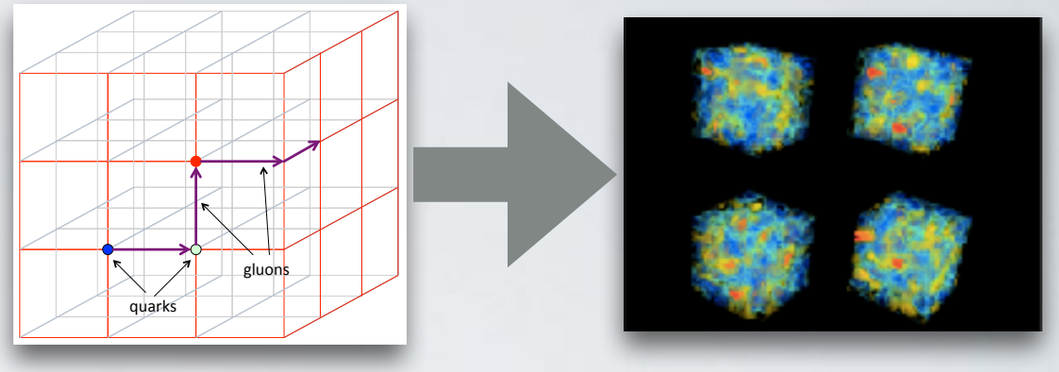
- Kenneth McElvain
- Ethan Neil
- Andre Walker-Loud

1. Overview

Computational Steps of Lattice QCD

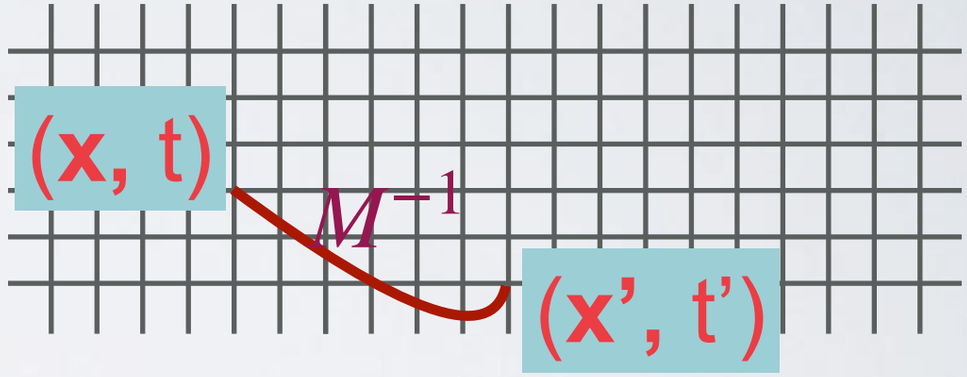
1. Gauge Ensemble Generation

- Markov Chain Monte Carlo techniques are used to generate ensembles of gauge fields according to the Euclidean QCD path integral



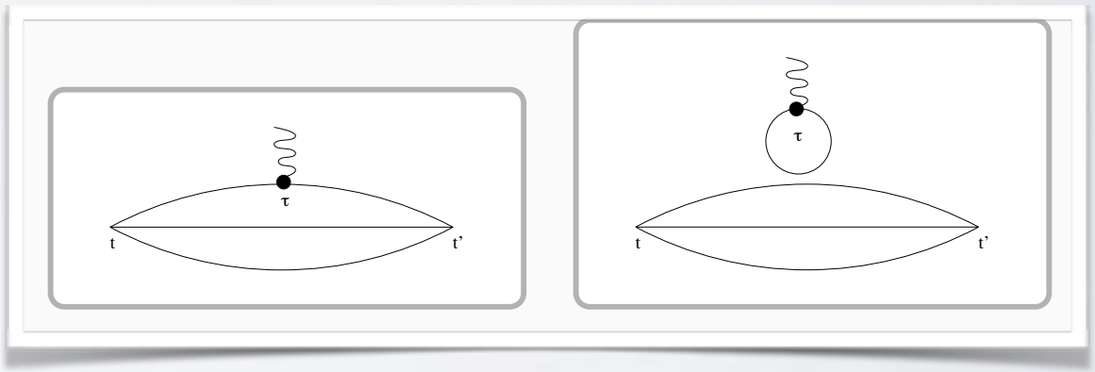
2. Quark Propagators

- M x = b is solved for each spin and color, where M is the Dirac matrix and b is the source for the quark propagator



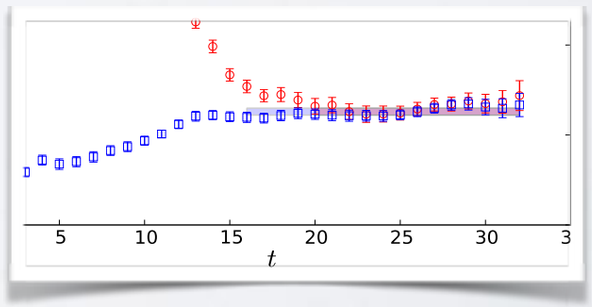
3. Contractions

- Quark propagators are contracted together with proper operators for the desired physical observables to create correlation functions.



4. Data Analysis

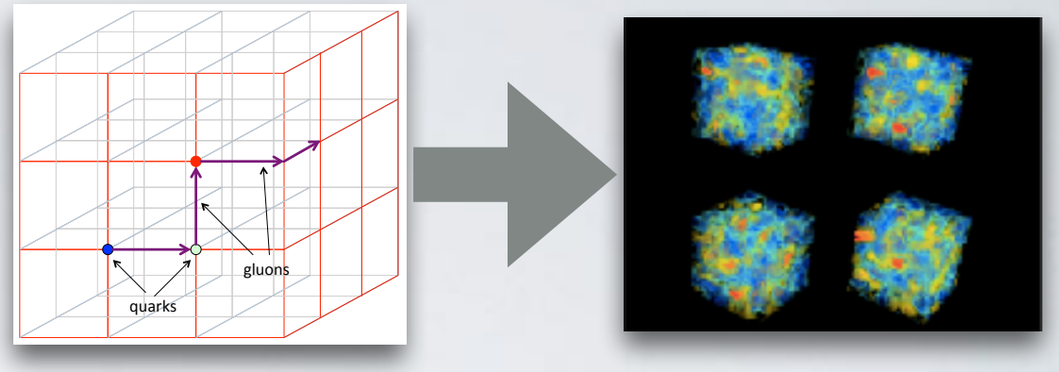
- Correlation functions are analyzed to extract physical quantities (masses, form factors, etc.)



Computational Steps of Lattice QCD

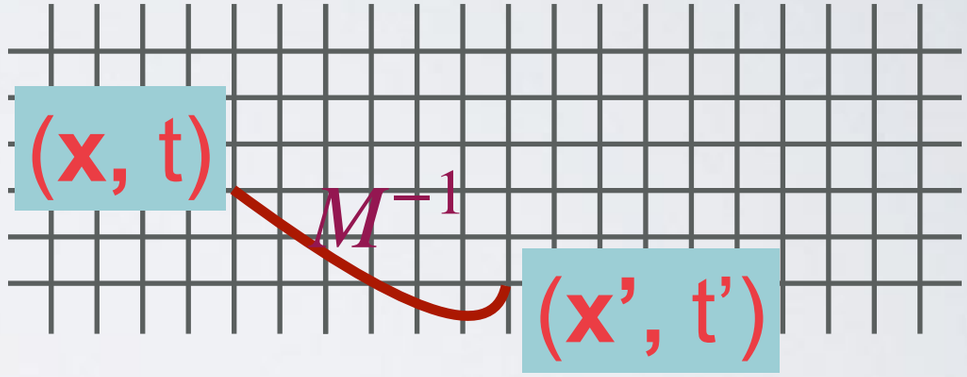
1. Gauge Ensemble Generation

- Markov Chain Monte Carlo techniques are used to generate ensembles of gauge fields according to the Euclidean QCD path integral



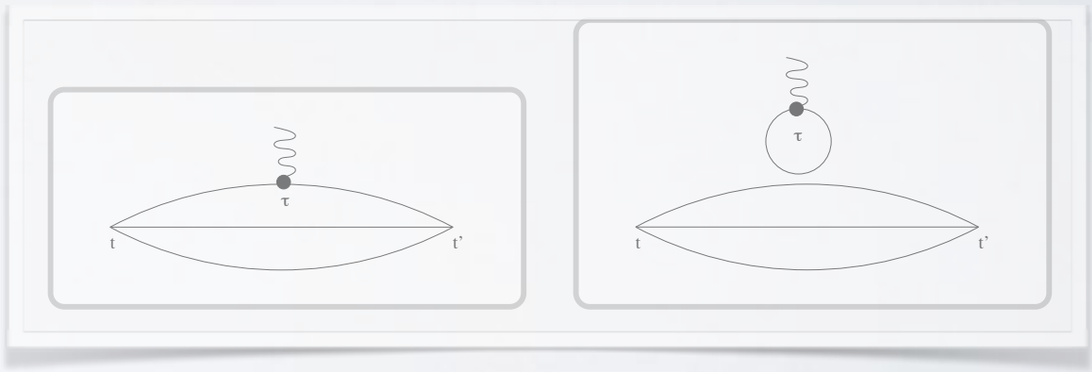
2. Quark Propagators

- M x = b is solved for each spin and color, where M is the Dirac matrix and b is the source for the quark propagator



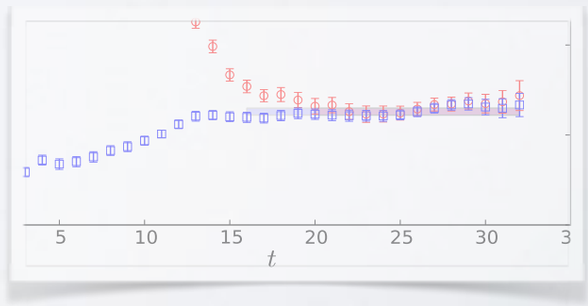
3. Contractions

- Quark propagators are contracted together with proper operators for the desired physical observables to create correlation functions.



4. Data Analysis

- Correlation functions are analyzed to extract physical quantities (masses, form factors, etc.)



Computational challenges

Lattice QCD is at a point where we can simulate directly at the physical quark masses, on fine lattices and with a large volume.

- **Going to continuum limit:**
  - The typical correlation length goes as $1/a$ ⟶ harder to generate independent gauge configurations
  - Critical slowing down ⟶ needs better HMC algorithms
- **Going to light/physical quark mass:**
  - Dirac matrix is ill-conditioned
    - affects both HMC and quark propagator calculations.
  - Need improved solvers
    - low-mode deflation, multigrid, block CG, …
  - Growing noise-to-signal $\frac{signal}{noise} \propto \sqrt{N} e^{-(M_N - \frac{3}{2} M_\pi)t}$ ⟶ Noise-reducing methods

Then there are also challenges presented by the machines available to us.

Top 20 of TOP 500 (June 2018)

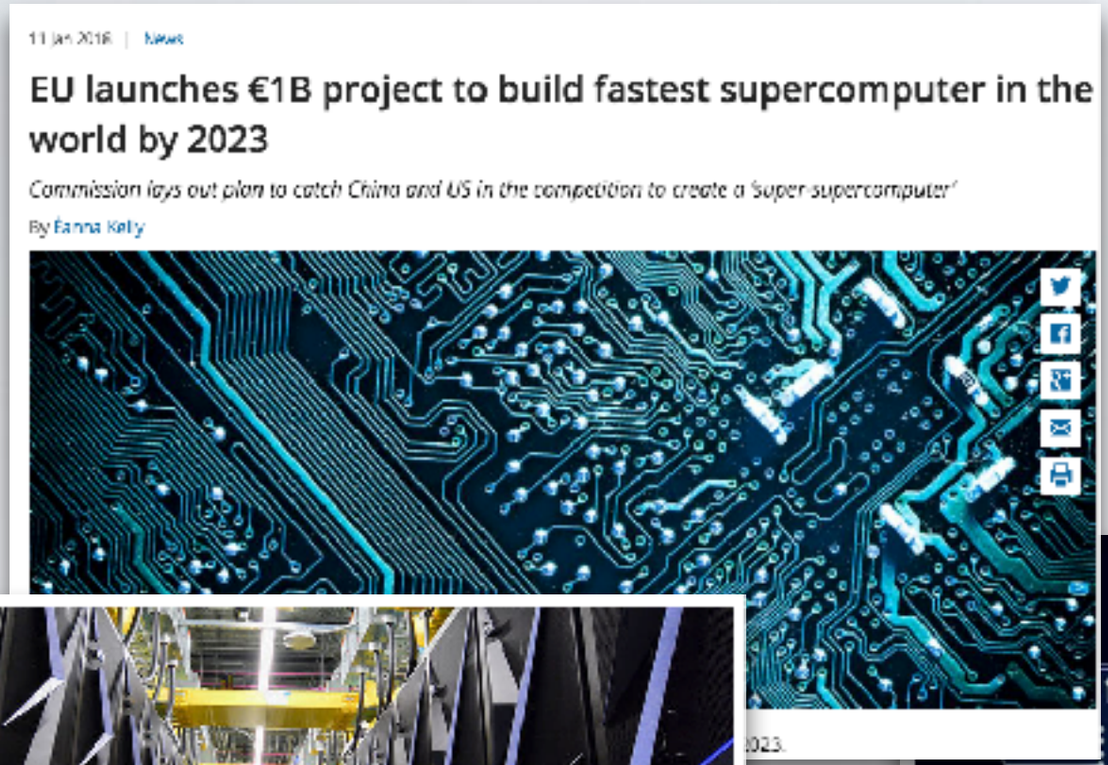
- Top 500 list has a blend of GPU-based and multi/many-core based architectures.
- In heterogeneous configurations, host CPUs are dominantly Intel Xeon CPUs.
- Several Intel Xeon Phi-based systems made it to the Top 20.

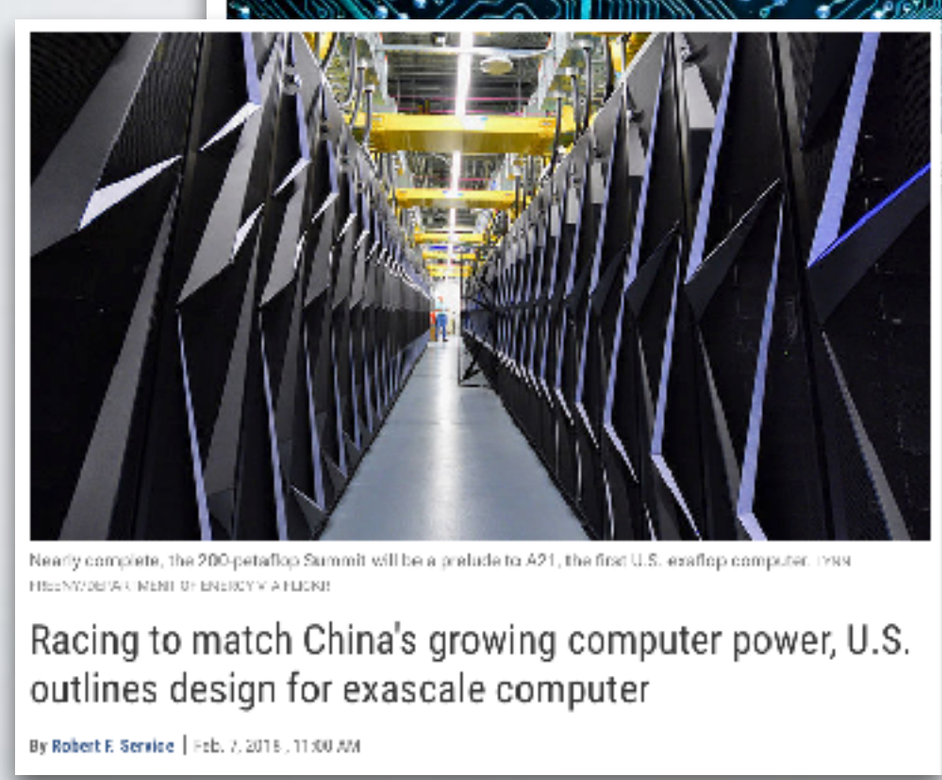
Rank	Name	Architecture	Site
1	Summit	IBM Power9 + NVIDIA GPUs	DOE/SC/Oak Ridge National Laboratory
2	Sunway TaihuLight	Sunway	National Supercomputing Center in Wuxi
3	Sierra	IBM Power9 + NVIDIA GPUs	DOE/NNSA/LLNL
4	Tianhe-2A	Intel Xeon + Matrix	National Super Computer Center in Guangzhou
5	AI Bridging Cloud Infrastruct	Intel Xeon + NVIDIA GPUs	National Institute of Advanced Industrial Science and Te
6	Piz Daint	Intel Xeon + NVIDIA GPUs	Swiss National Supercomputing Centre (CSCS)
7	Titan	AMD + NVIDIA GPUs	DOE/SC/Oak Ridge National Laboratory
8	Sequoia	BQC	DOE/NNSA/LLNL
9	Trinity	Intel Xeon Phi	DOE/NNSA/LANL/SNL
10	Cori	Intel Xeon Phi	DOE/SC/LBNL/NERSC
11	Nurion	Intel Xeon Phi	Korea Institute of Science and Technology Information
12	Oakforest-PACS	Intel Xeon Phi	Joint Center for Advanced High Performance Computin
13	HPC4	Intel Xeon + NVIDIA GPUs	Eni S.p.A.
14	Tera-1000-2	Intel Xeon Phi	Commissariat a l'Energie Atomique (CEA)
15	Stampede2	Intel Xeon Phi	Texas Advanced Computing Center/Univ. of Texas
16	K Computer	SPARC	RIKEN Advanced Institute for Computational Science (A
17	Mira	BQC	DOE/SC/Argonne National Laboratory
18	Marconi Intel Xeon Phi	Intel Xeon Phi	CINECA
19	TSUBAME3.0	Intel Xeon + NVIDIA GPUs	GSIC Center, Tokyo Institute of Technology
20		Intel Xeon	United Kingdom Meteorological Office

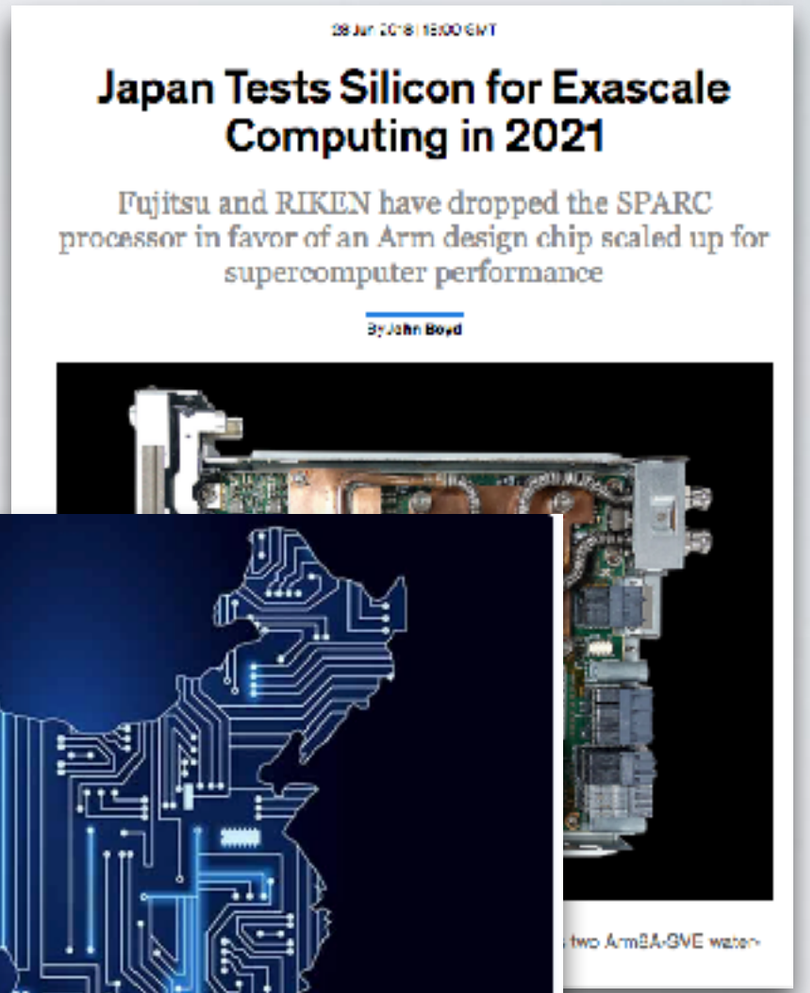
Data from top500.org

The race to exascale

- China, EU, Japan and US are all developing exascale supercomputers.







Future Pre-Exascale and Exascale Systems in the US

- Architectures on these future systems are still unknown

Relevant Pre-Exascale and Exascale Systems for ECP

Pre-Exascale Systems				Exascale Systems
2013	2016	2018	2020	2021-2022
 Argonne IBM BG/Q Open	 Argonne Intel/Cray KNL Open	 ORNL IBM/NVidia P9/Volta Open	NERSC-9 LBNL TBD Open	 Argonne Intel/Cray TBD Open
 ORNL Cray/NVidia K20 Open	 LBNL Cray/Intel Xeon/KNL Unclassified			Frontier ORNL TBD Open
 LLNL IBM BG/Q Secure	 LANL/SNL Cray/Intel Xeon/KNL Secure	 LLNL IBM/NVidia P9/Volta Secure	Crossroads LANL/SNL TBD Secure	El Capitan LLNL TBD Secure

25

Doug Kothe and Stephen Lee, ECP update @ ASCAC 2017

What we learn from existing supercomputers

- New and upcoming machines may increasingly feature "fat nodes" - a compute node that is capable of many floating point operations.
- Take for example Summit at ORNL:
    - 6 NVIDIA Volta GV100s connected with NVLink
    - Dual-socket IBM Power 9 CPUs
    - Peak Flops per node ~ **95 Tflops** (GPU, single-precision)
    - Interconnect:
        - dual-rail Mellanox EDR infiniband,
        - peak bandwidth **50 GBytes**/s bidirectional
- Lattice QCD computational intensity is ~1 flop/byte.
- Weak and strong scalings will be severely affected by the much slower growth of interconnect bandwidth.

Image from: Oak Ridge National Lab

See Mathias Wagner's talk (next) for more in-depth discussions about modern GPUs for lattice QCD.

Weak Scaling on Summit

- While on single GPU, we can achieve ~20% peak using QUDA, scaling to 1024 nodes results in a factor of >10X loss in performance

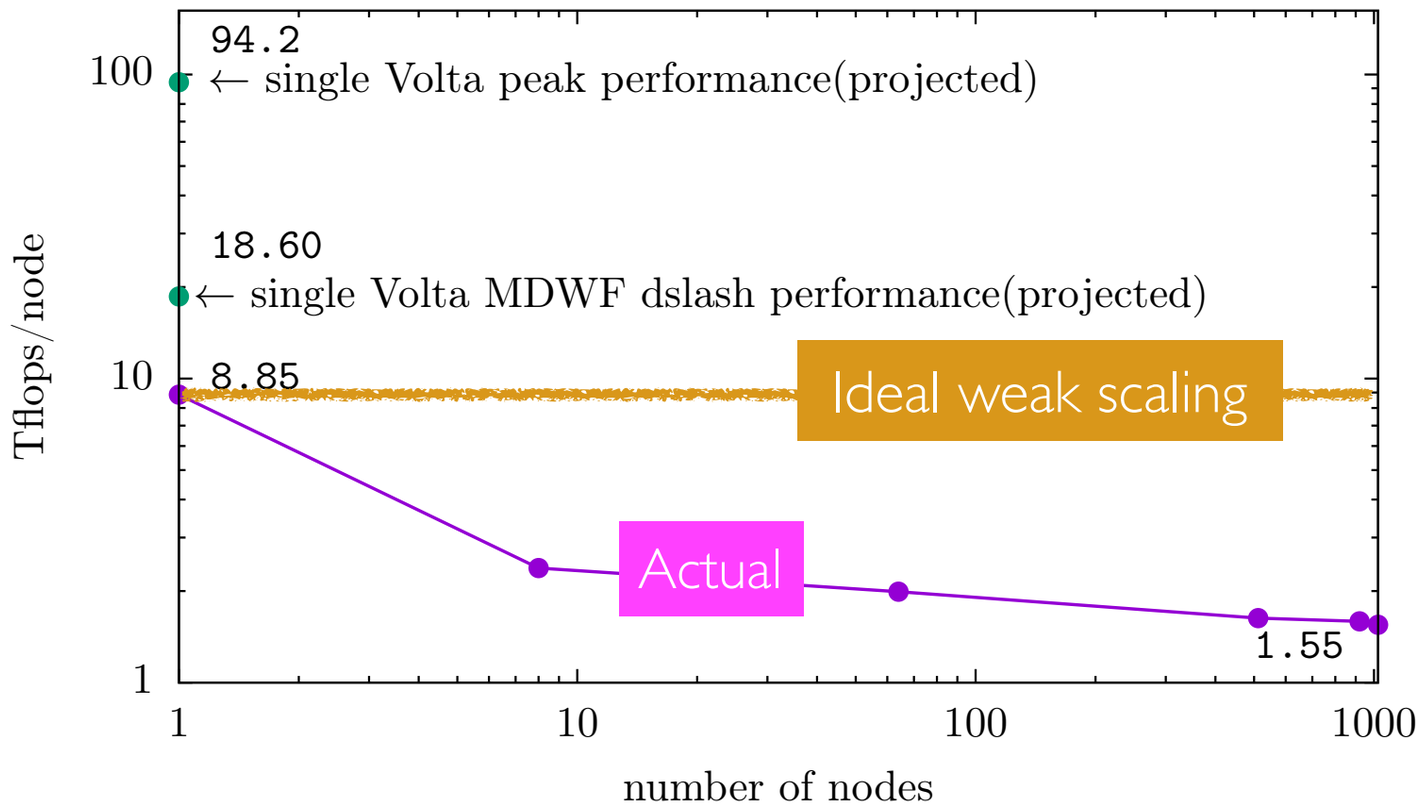


Figure 2: Half precision Möbius domain wall fermion CG weak scaling with local volume of $16 \times 12^3 \times 12$. 6 NVIDIA Volta GPUs on each compute node. Numbers provided by Chulwoo Jung.

Plot by Jiqun Tu
Data from Chulwoo Jung

To improve the scaling

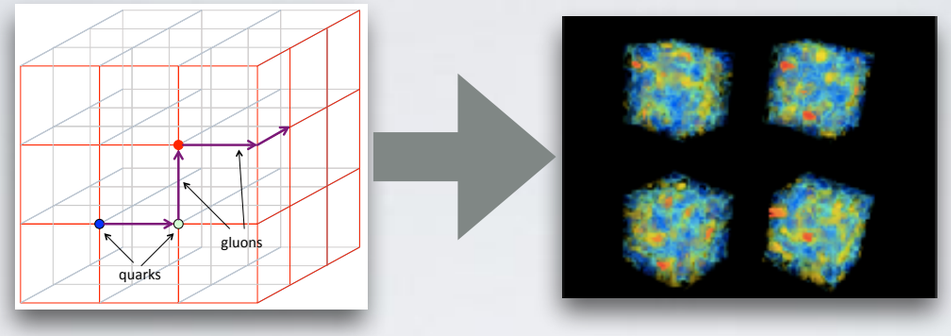
We can do two things:

1. **Reducing message size between compute nodes**
    - Reduced precision in data being transferred
    - Data compression
    - Communication-avoiding algorithms

2. **Maximizing delivered bandwidth**
    - If your application is bandwidth bound as opposed to latency bound, it is important to make sure that your achieved bandwidth is close to the line bandwidth.
    - Not a given! Software stack and environment can affect actual delivered bandwidth. Boyle, Chuvelev, Cossu, Kelly, Lehner, Meadows, arXiv:1711.04883

2. Algorithms

Gauge ensemble generation



- Recall in lattice QCD we compute ensemble average of the physical observable in the Euclidean path integral formulation,

$$\langle O \rangle = \frac{1}{Z} \int [dU] O[U] \prod_f \det(D[U] + m_f) e^{-S_g[U]}$$

$$Z = \int [dU] \prod_f \det(D[U] + m_f) e^{-S_g[U]}$$

- Calculating the fermion determinant directly is prohibitive $\longrightarrow O(V^3)$
- Pseudofermion (bosonic) fields are introduced to trade the determinant calculations with matrix inversions.

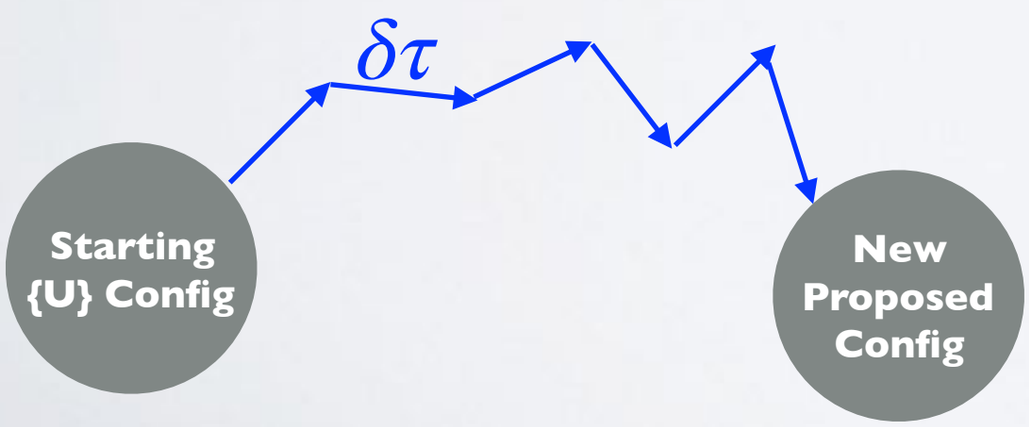
$$Z = \int [dU] \left[[d\phi^\dagger][d\phi] e^{-\phi^\dagger (MM^\dagger)^{-1} \phi - S_g[U]} \right]$$

$P[U]$

$M = D[U] + m_f$

MCMC and HMC

- To generate the gauge field configurations with the probability distribution ***P[U]***, Markov Chain Monte Carlo (MCMC) algorithms are used.
- MCMC algorithms should satisfy the following properties:
  - Detailed balance: $p[U' \leftarrow U]P[U] = p[U \leftarrow U']P[U']$
  - Ergodicity: there should be zero probability of a particular gauge configuration never appearing.
- **Hybrid Monte Carlo (HMC)**: [Duane, Kennedy, Pendleton, Roweth, 1987]
  - Couples Molecular Dynamics (MD) update, with Metropolis accept/reject step
  - **Exact algorithm**: the Metropolis step makes it free of integration errors.
  - **Keep the change in Hamiltonian small to obtain good acceptance rate:** reduce force contributions or decrease the step size (which means more force calculations).
  - Fermion force term calculations involve Dirac matrix inversions: expensive!



$$F_{\mu,pf}(x) = U_\mu(x)\left[\phi^\dagger \mathcal{D}^{-1}\frac{\partial \mathcal{D}}{\partial U_\mu(x)}\mathcal{D}^{-1}\phi\right]$$

$$\mathcal{D} = D^\dagger D$$

The Evolution of HMC

Lattice QCD has made remarkable progress in dynamical fermion simulations.

- **HMC:** [Duane, Kennedy, Pendleton, Roweth, 1987]

$$Z = \int [dU][d\phi][d\phi^\dagger] e^{-\phi^\dagger \mathcal{D}^{-\alpha} \phi - S_g}$$

$$\alpha = N_f/2$$

  - Suitable for even numbers of fermion determinants
  - Various acceleration techniques have been developed:
    - Polynomial HMC [Frezzotti & Jansen, 1997]
    - Hasenbusch mass preconditioning [Hasenbusch, 2001]
    - Domain decomposition/Schwarz preconditioning [Luscher, 2004]

- **One-flavor HMC:**
  - R algorithm (inexact) [Gottlieb et al., 1987]

$$\det \mathcal{D}^\alpha = \exp(\alpha \mathrm{Tr} \ln \mathcal{D})$$

  - Rational HMC [Clark & Kennedy, 2003]

$$Z = \int [dU][d\phi][d\phi^\dagger] e^{-\phi^\dagger r^2(\mathcal{D}) \phi - S_g}$$

- **Various improved integrators**

Quiz: What's quenching?

Exact One-flavor Fermion Algorithm (EOFA)

- For simulations with odd numbers of flavor, such as 2+1-flavor simulations to include strange quark, or 2+1+1-flavor simulations to include charm quark, RHMC can be used.
- RHMC approximates the fractional fermion determinant with rational functions of D

$$r(\mathcal{D})\phi = \left[\alpha_0 + \sum_{i=1}^{d} \frac{\alpha_i}{\mathcal{D} + \beta_i}\right] \phi$$

- Needs multi-shift CG solver and can be expensive. Also memory intensive due to additional pseudofermion vectors.
- For domain wall fermions, the Pauli-Villar fields are needed as regulators. For 2+1+1 flavor, the determinants can be rewritten as

Ting-Wai CHIU on 25 Jul 2018 at 5:10 PM

$$\frac{\det D(m_{u/d})}{\det D(m_{PV})}\frac{\det D(m_{u/d})}{\det D(m_{PV})}\frac{\det D(m_s)}{\det D(m_{PV})}\frac{\det D(m_c)}{\det D(m_{PV})}$$

$$= \left(\frac{\det D(m_{u/d})}{\det D(m_{PV})}\right)^2 \left(\frac{\det D(m_c)}{\det D(m_{PV})}\right)^2 \frac{\det D(m_s)}{\det D(m_c)}$$

- **EOFA**: Use Schur determinant identity to factorize

TWQCD, Phys. Lett.B738 (2014)

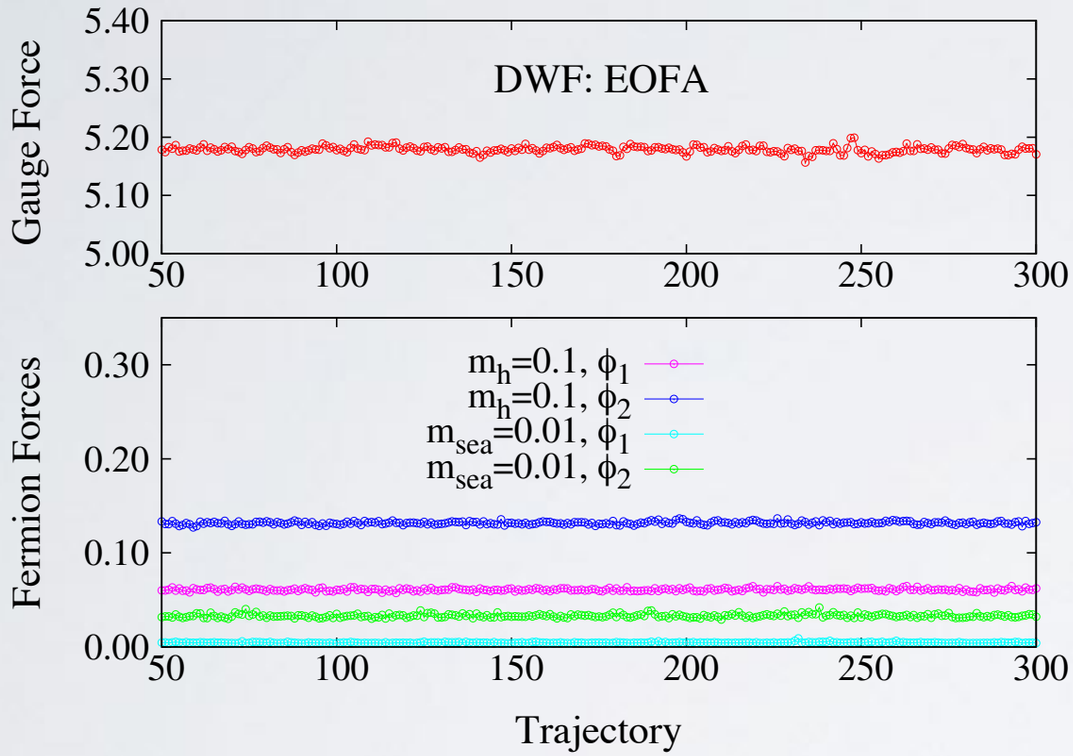
$$\det\left(\frac{D(m_1)}{D(m_2)}\right) = \frac{1}{\det(H_1)} \cdot \frac{1}{\det(H_2)}$$

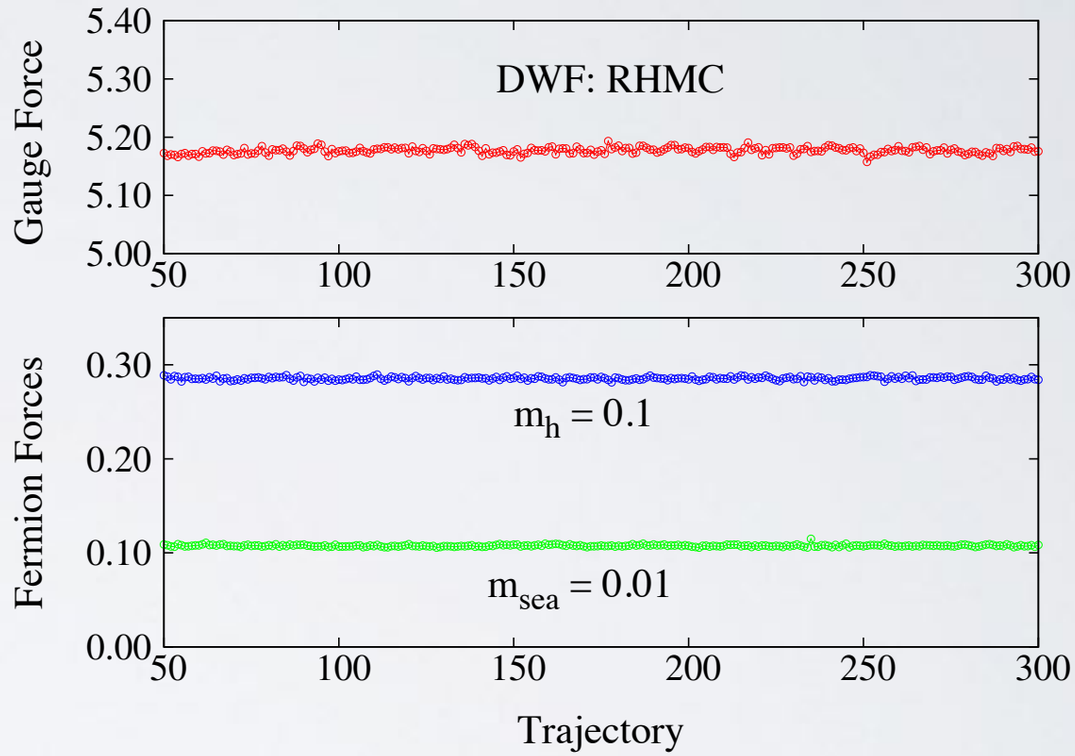
H_1 and H_2 Hermitian and positive-definite

EOFA - II

TWQCD, arXiv:1412.0819

$16^3 \times 32 \times 16$





- EOFA generates smaller fermion forces, allowing for larger step sizes.
- Overall numerical cost can be reduced.
- Being used in TWQCD's 2+1+1 flavor evolution at physical quark masses.

Ting-Wai CHIU on 25 Jul 2018 at 5:10 PM

EOFA - III

RBC-UKQCD is using EOFA for the generation of 2+1-flavor DWF with period or G-parity boundary conditions (for K to pi pi studies). [Jung, Kelly, Mawhinney & Murphy, PRD97, 054503 (2018)]

Light Quark Action	Integrator	Light Hasenbusch Masses	Δt	r_{MD}	N_{traj}	Acceptance	Efficiency
RHMC	Omelyan	0.007	0.0625	10^{-8}	850	88%	—
EOFA	FG	0.0058, 0.0149, 0.059, 0.177, 0.45	0.1667	10^{-7}	850	93%	**4.2**

Table: Fully tuned RHMC and EOFA schemes. Δt is the outermost time step, r_{MD} is the MD CG tolerance, and **efficiency is the speed-up in the total job time relative to the RHMC scheme**.

$32^3 \times 64 \times 16$

RHMC

- Omelyan integrator ($\lambda = 0.22$)
- One light quark Hasenbusch mass
- Multishift CG with single precision $\not{D}$ but accumulating solution and search vectors in double precision, coupled with reliable update to correct residual
- Even-odd preconditioning

EOFA

- Force gradient integrator
- Five light quark Hasenbusch masses
- Mixed precision defect correction CG
- Even-odd preconditioning
- Cayley preconditioning
- Force gradient forecasting
- Heatbath forecasting
- Heatbath CG tolerance tuning

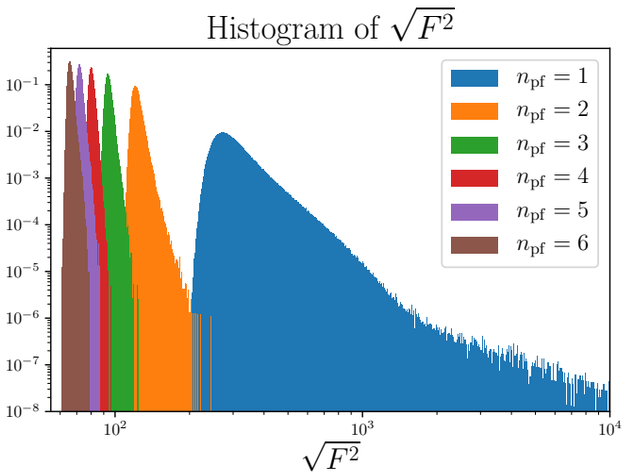
Figure: Comparison of RHMC and EOFA integration schemes.

David Murphy

RHMC with Block Solvers and Multiple Pseudofermions

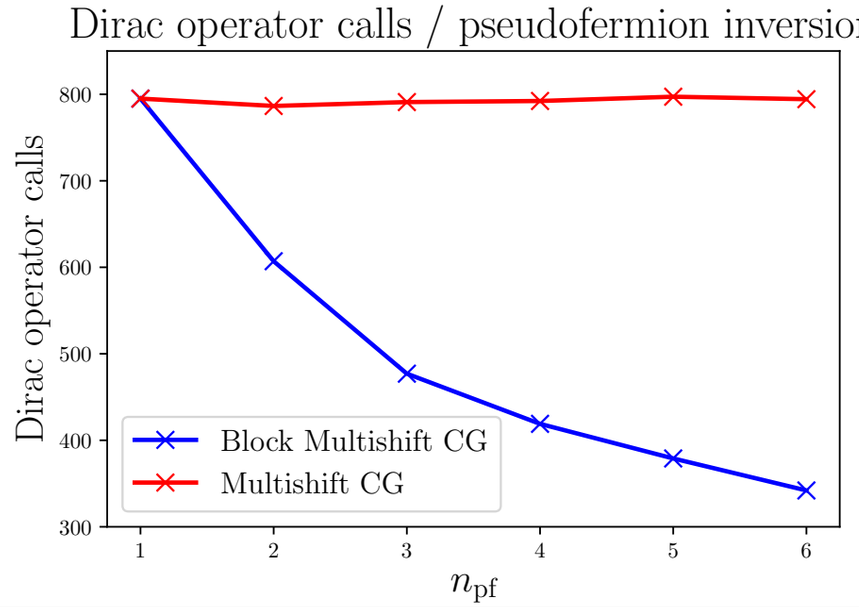
As a way to speed up regular 2-flavor HMC

- Multiple pseudofermions: $\det\left[M^\dagger M\right] = \det\left[\left(M^\dagger M\right)^{1/n_{\mathrm{pf}}}\right]^{n_{\mathrm{pf}}}$
- Dramatically reduces the mean and the variance of the RHMC pseudofermion force term:

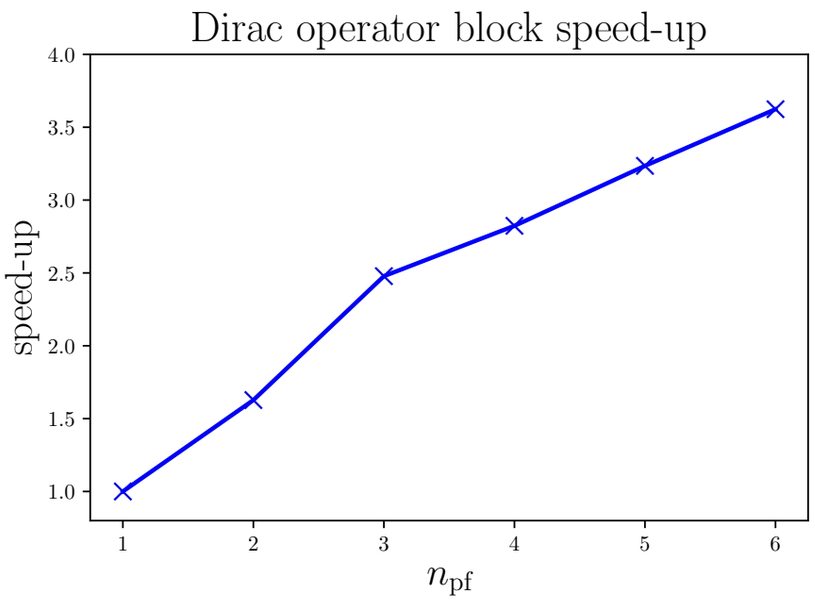


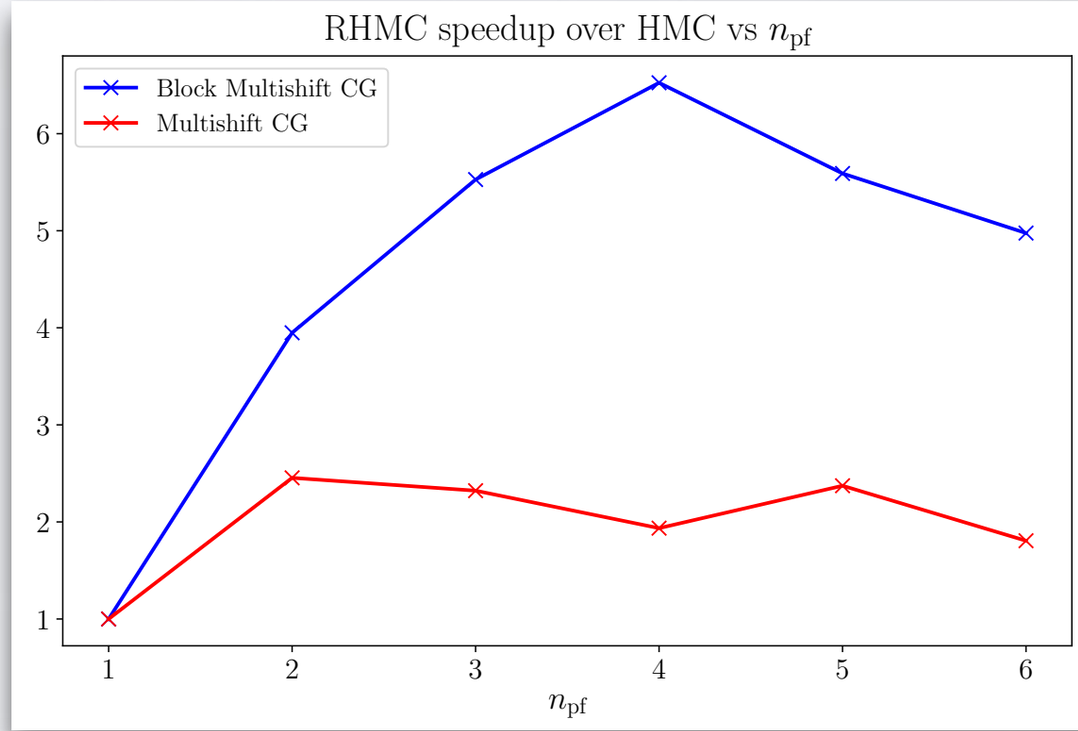
- This allows a larger integrator step size, and hence fewer Dirac operator inversions are required.

- Need to invert the Dirac operator on n_{pf} pseudofermion vectors.
- Block solvers do this simultaneously for all vectors, and can converge with significantly fewer iterations.



- Higher computational intensity (flops/bytes): only need to load gauge links once per n_{pf} vectors.





de Forcrand & Keegan, arXiv:18??.?????

Liam Keegan

Critical Slowing Down

- Ongoing efforts by the US QCD ECP team.
- Looking for ways to reduce critical slowing down in HMC.

- **Fourier-acceleration:** based on the notion that the HMC evolution step sizes should be chosen to depend on the Fourier modes. [Batrouni et al. 1985]

- Fourier acceleration, the HMC algorithm and renormalizability

  Norman CHRIST, Monday, 2:00PM

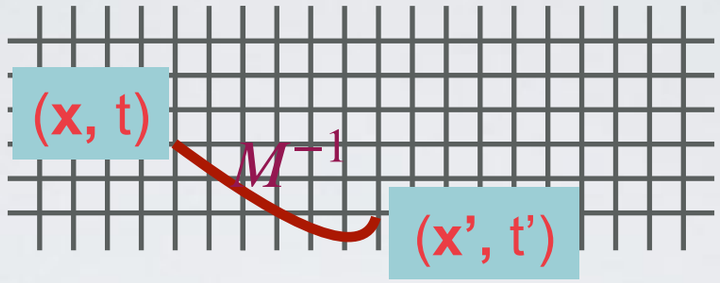
- Testing a new gauge-fixed Fourier acceleration algorithm

  Yidi ZHAO, Monday, 2:20PM

- Ensemble Quasi-Newton HMC

  Xiaoyong JIN, Monday, 2:40PM

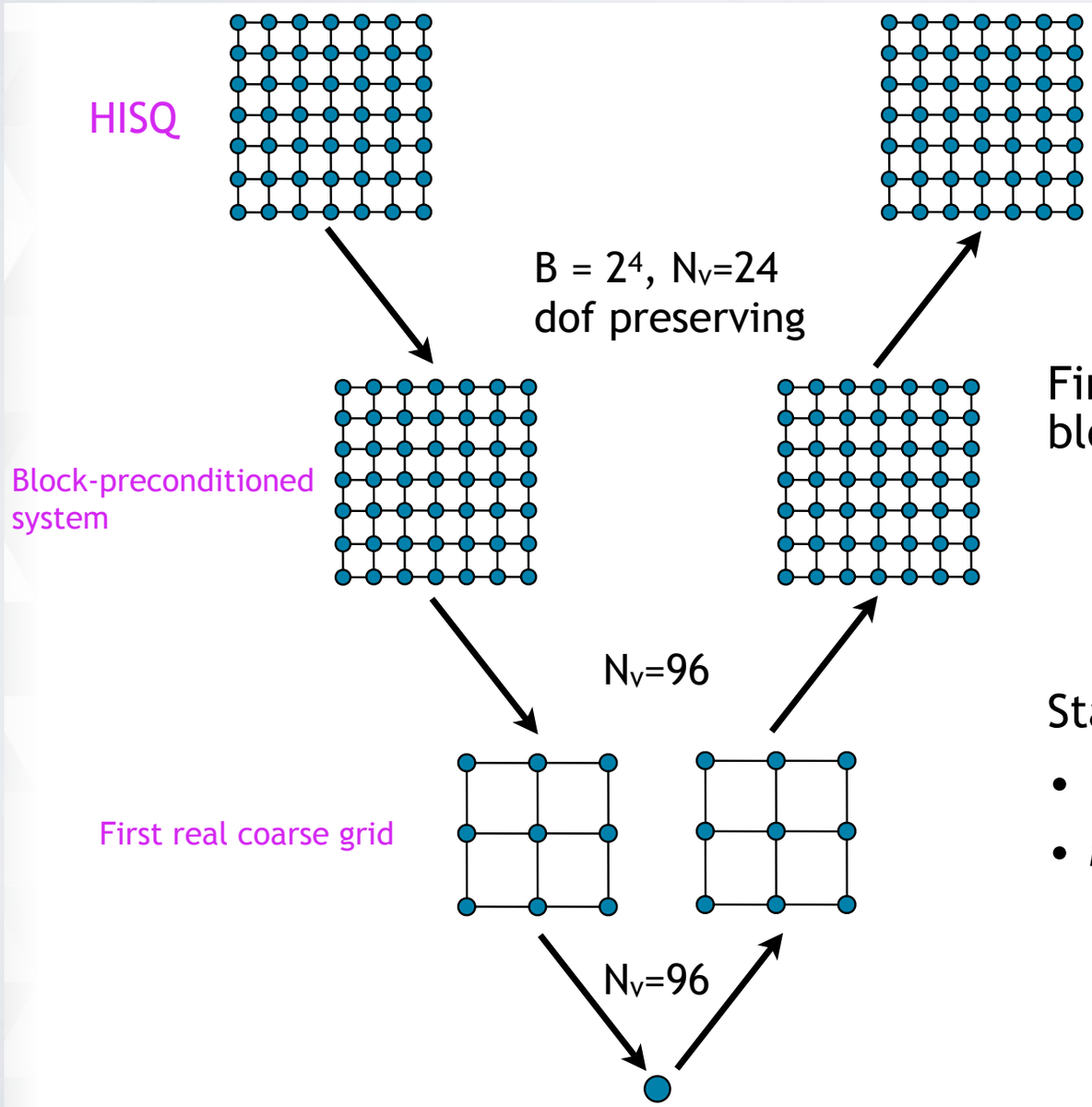
Solvers



- The computational intensive part of LQCD is to solve Ax=b.
- Needed for both fermion force calculations in HMC and in quark propagators.
- Condition number of A grows as $1/m^2$: iterative solvers such as CG converge much slower as $m \to 0$.
- **Low-mode deflation:** project out the low eigenmodes and solve in the reduced subspace → needs efficient eigenvector solvers.
- **Adaptive Multigrid**
- **Block solvers**

Staggered Multigrid

- Staggered MG has been known to be hard due to the non-Hermitian structure of the staggered Dirac operator
- New staggered MG formulation exploits staggered's 2^d block structure.
- **2D:** Brower, Clark, Strelchenko & Weinberg, arXiv:1801.07823
- **4D:** Presented by Kate CLARK on 25 Jul 2018 at 4:10 PM



HISQ MG ALGORITHM

First "coarsening" is transformation to block-preconditioned system

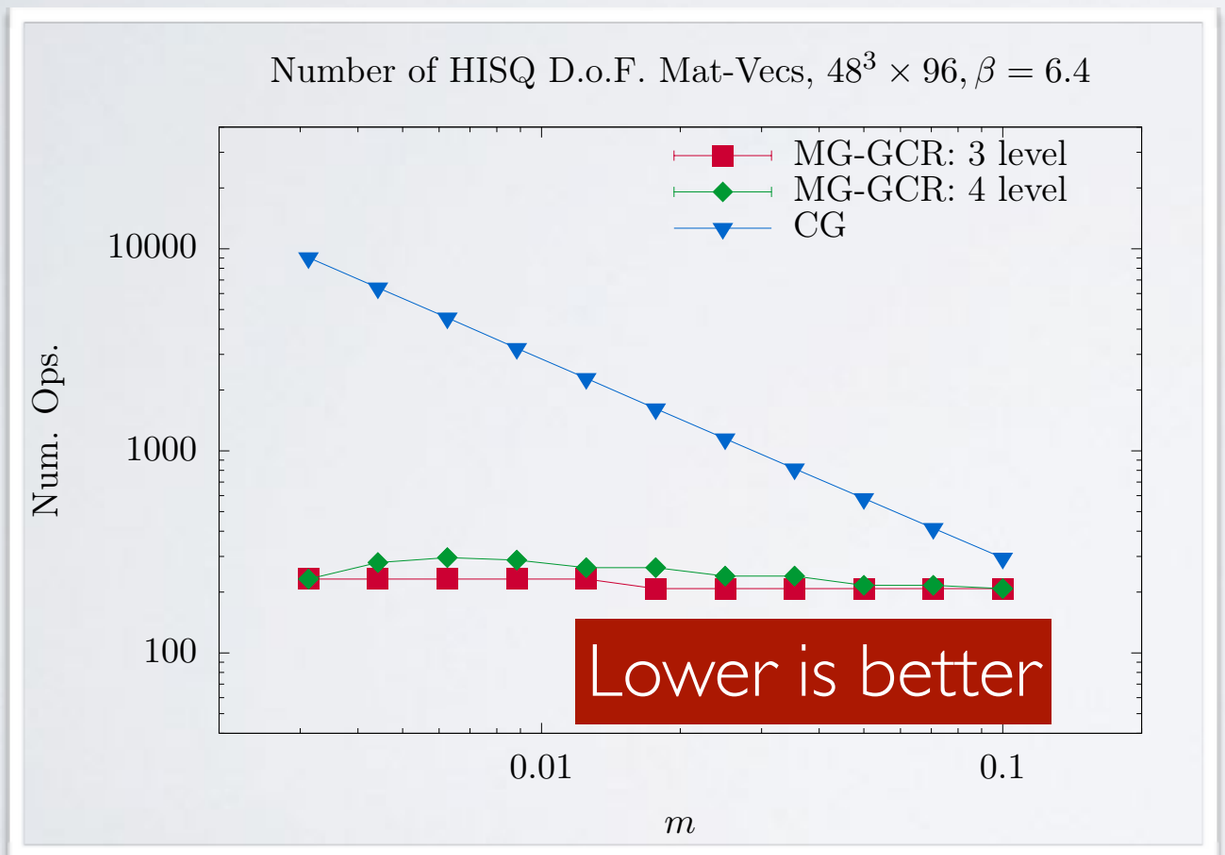
Staggered has 4-fold degeneracy

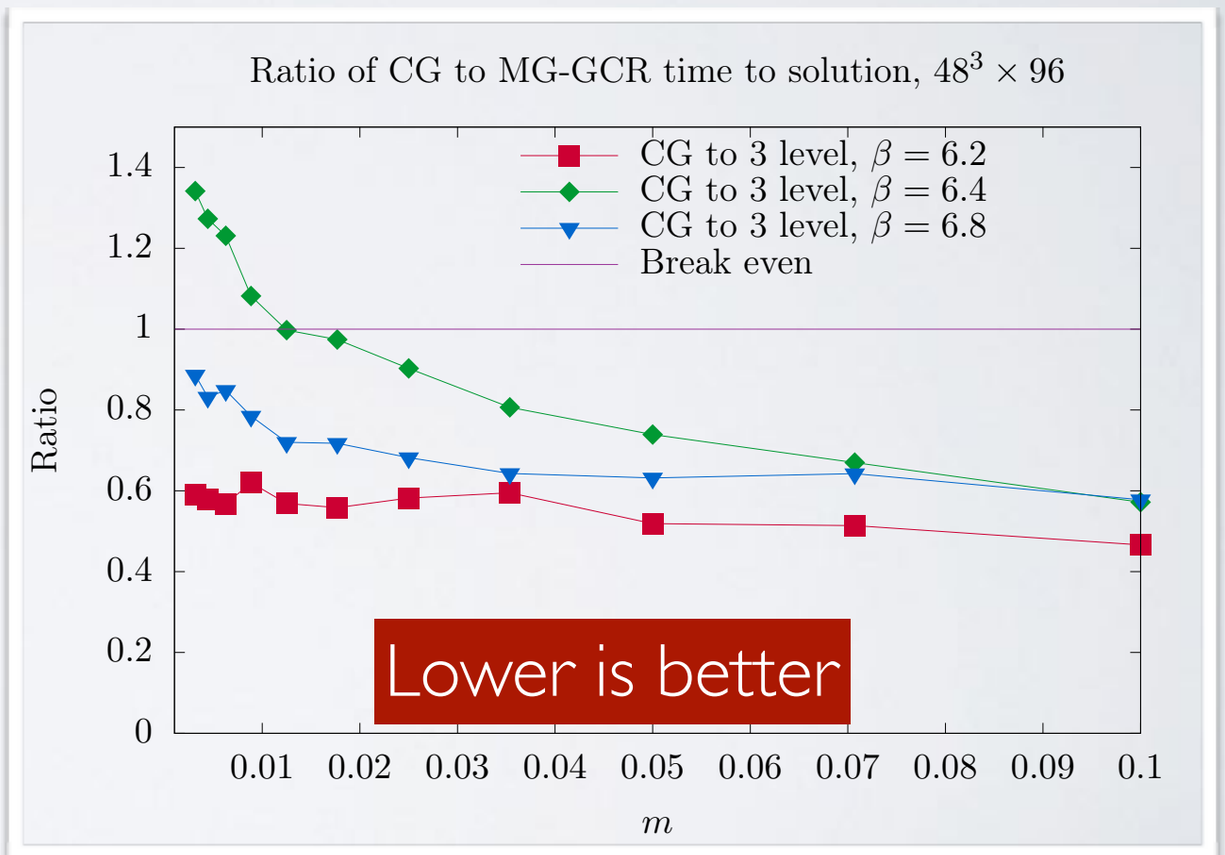
- Need 4x null space vectors (N_v=24 -> 96)
- Much more memory intensive

Kate Clark

Staggered Multigrid

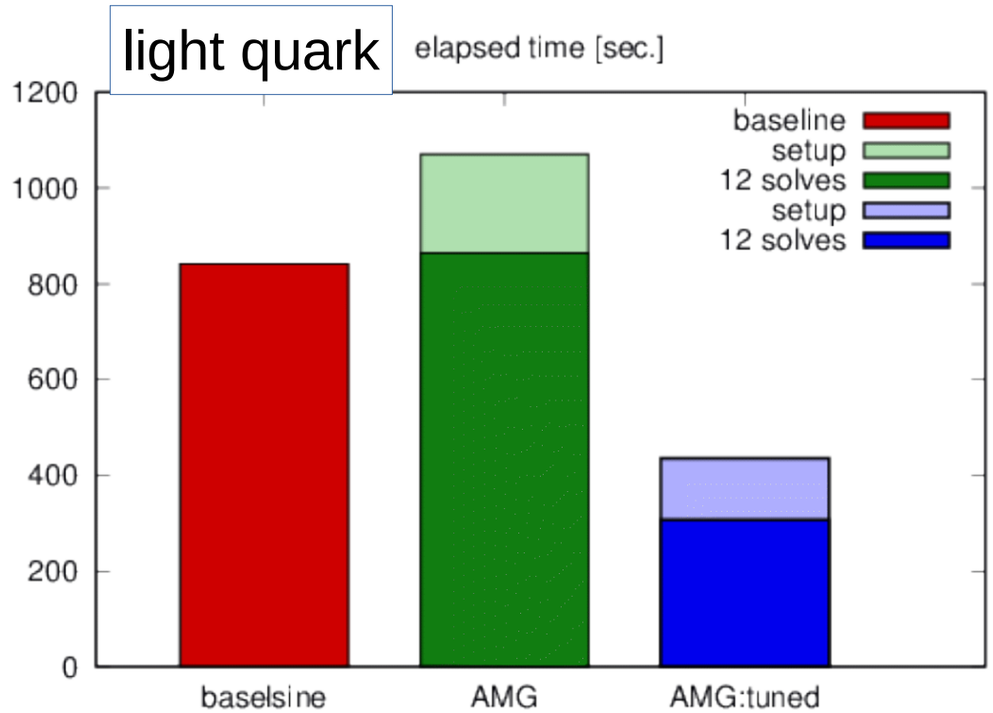
- 4D HISQ MG algorithm is implemented in QUDA.
- # of MG-GCR iterations independent of quark masses.
- Initial speedup seen for small quark masses.
- Further speedup in time-to-solution will need reduced setup cost and better coarse-grid solvers.

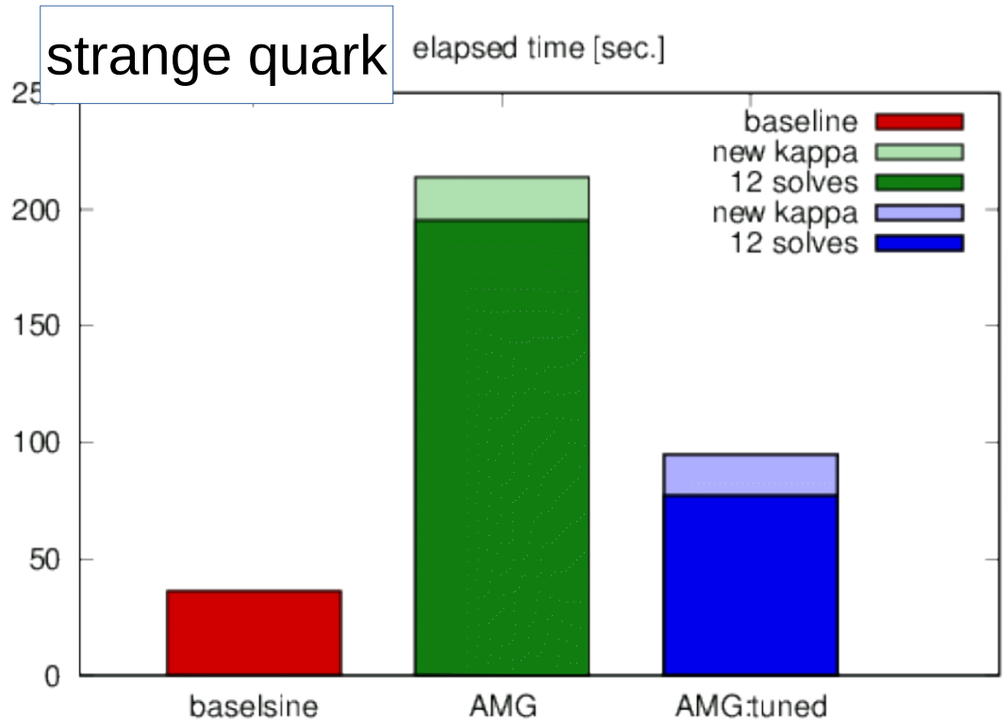




Kate Clark

DDalphaAMG on K Computer





Lower is better

baseline: well tuned solver for K [efficiency: 22%]

mixed precision FBiCGstab sovler, where the single precision solver uses Domain decomposition. The solver inside the domain is an even odd preconditioned SSOR.

cf. K.-I.Ishikawa et al [PACS collaboration]., PoS LATTICE2015(2016) 075

AMG(before tuning) [efficiency 3.0%]

cf. A.Frommer et al., SIAM J.Sci. Comput. 36 (2014) A1581;https://github.com/DDalphaAMG

AMG: tuned [efficency 5.3%] THIS WORK

cf. https://github.com/i-kanamori/DDalphaAMG/tree/K/

Configuration: n_f=2+1 clover, 96^4 lattice, m_pi=146 MeV, 1/a=2.33 GeV [PACS]

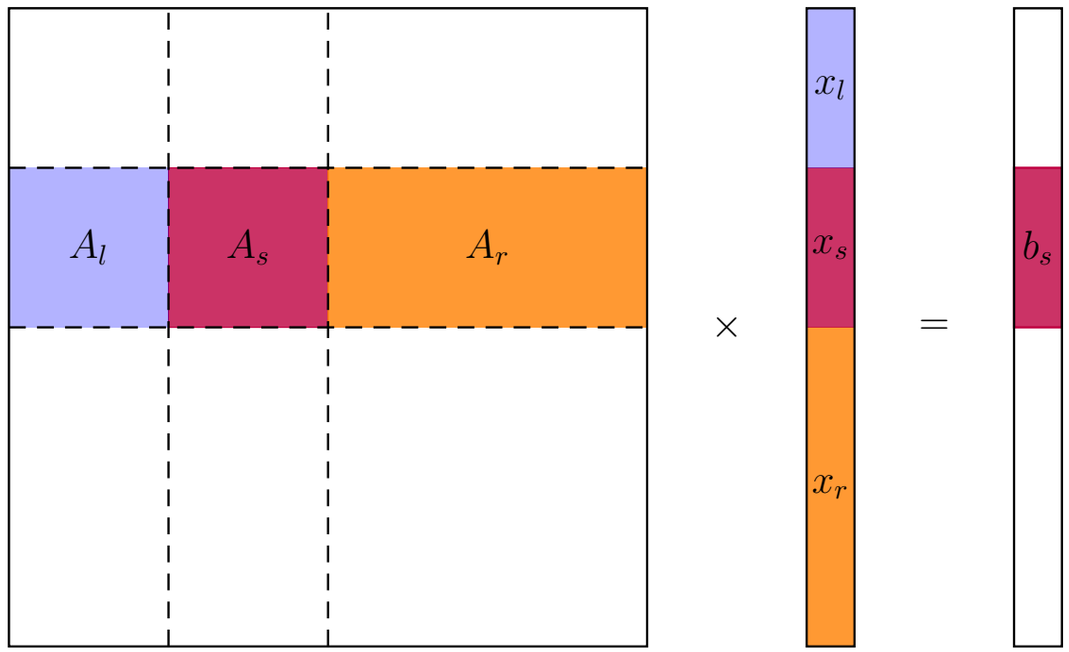
Poster by Issaku KANAMORI on 24 Jul 2018 at 6:45 PM

Multisplitting Preconditioned CG for Mobius DWF

Guo, Mawhinney & Tu, arXiv:1804.08593

- On machines like Summit, off-node communications are expensive compared to local arithmetics.
- Trade communication with local flops

$$Ax = b : A_l x_l + A_s x_s + A_r x_r = b_s$$



[O'Leary, 1985]

Jiqun TU on 23 Jul 2018 at 4:10 PM

1. Solve the block-diagonal term iteratively:

$$\begin{aligned} A_s x_s^{(k+1)} &= b_s - A_l x_l^{(k)} - A_r x_r^{(k)} \\ &= b_s - \left(A x^{(k)} - A_s x_s^{(k)} \right) \\ &= r^{(k)} + A_s x_s^{(k)} \equiv \hat{b}_s^{(k)} \end{aligned}$$

Local

2. Use as a preconditioner for outer CG

$r_0 = b - Ax_0$
$z_0 = M^{-1} r_0$
$p_0 = z_0$
$k = 0$
while have not converged **do**
  $\alpha_k = \langle r_k, z_k \rangle / \langle p_k, A p_k \rangle$
  $x_{k+1} = x_k + \alpha_k p_k$
  $r_{k+1} = r_k - \alpha_k A p_k$
  $z_{k+1} = M^{-1} r_{k+1} \leftarrow A_s x_s^{(k+1)} = r^{(k)} + A_s x_s^{(k)}$
  only first cycle, zero initial guess, iterate a fixed number of times
  $\beta_k = \langle z_{k+1}, r_{k+1} \rangle / \langle z_k, r_k \rangle$
  $p_{k+1} = z_{k+1} + \beta_k p_k$
  $k = k + 1$
end while

Multisplitting Preconditioned CG for Mobius DWF

Guo, Mawhinney & Tu, arXiv:1804.08593

- Solving inner CG sloppily with 3-6 iterations can already reduce the number of outer CG iterations by 2-3x.
- If local flops are fast enough for the inner CG to be faster than off-node communications, we gain overall.
- Could benefit strong scaling greatly which is needed for gauge evolution.

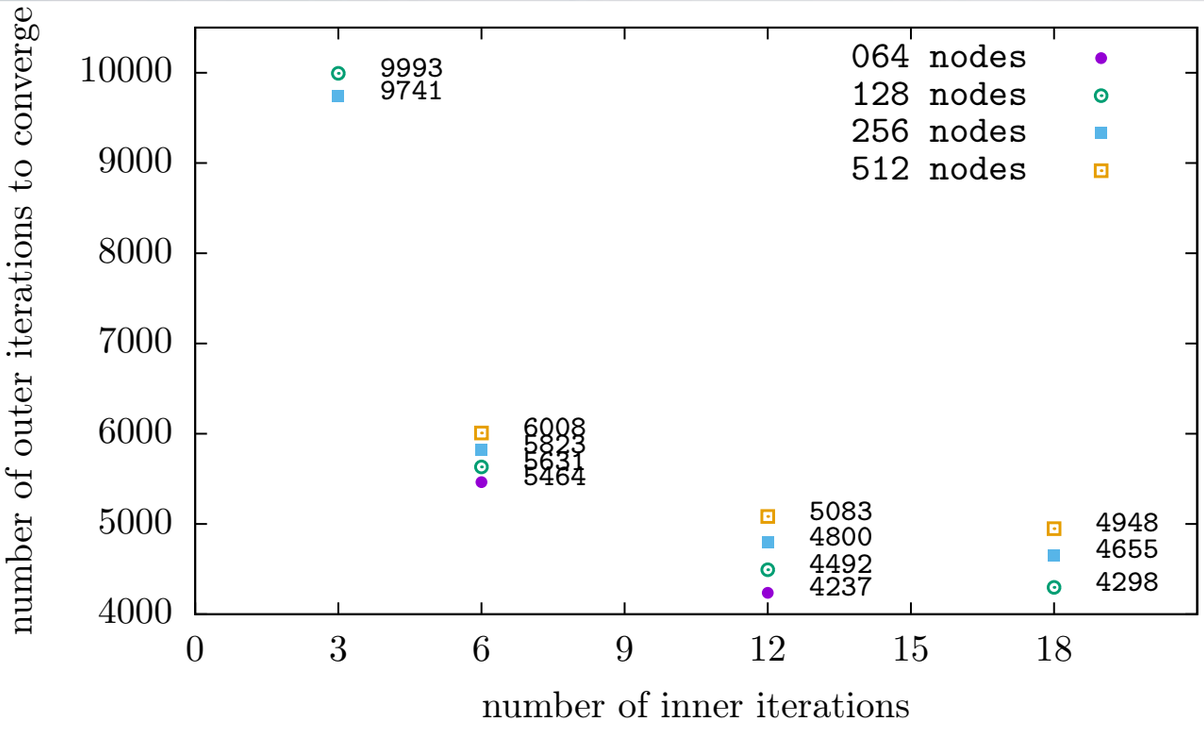


Figure 5: MSPCG solve on a $64^3 \times 128$ lattice ($a^{-1} = 2.36$ GeV) with physical pion mass. Plain CG takes 18092 iterations to converge to the same precision(10^{-10}). KNL at CORI.

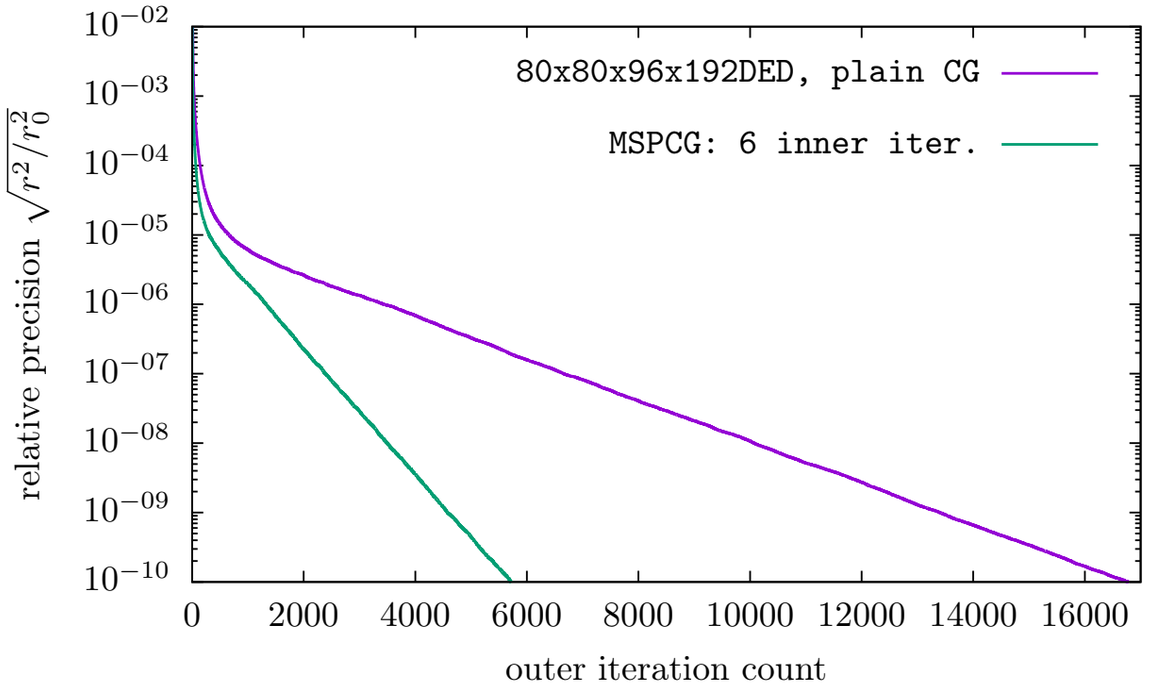
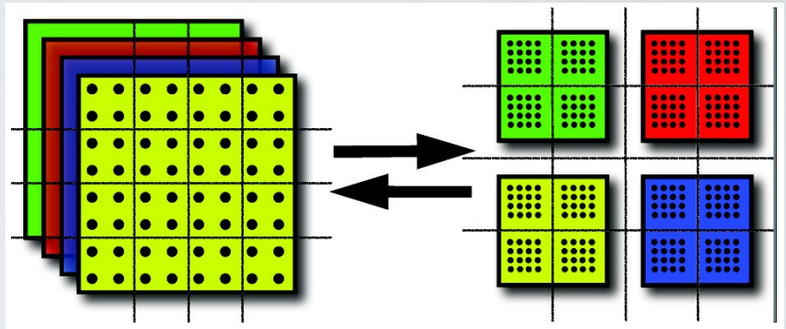


Figure 6: MSPCG solve on a $80^2 \times 96 \times 192$ lattice ($a^{-1} = 3.00$ GeV) with physical pion mass. Test performed on CORI at NERSC with 1024 KNL nodes.

Jiqun TU on 23 Jul 2018 at 4:10 PM

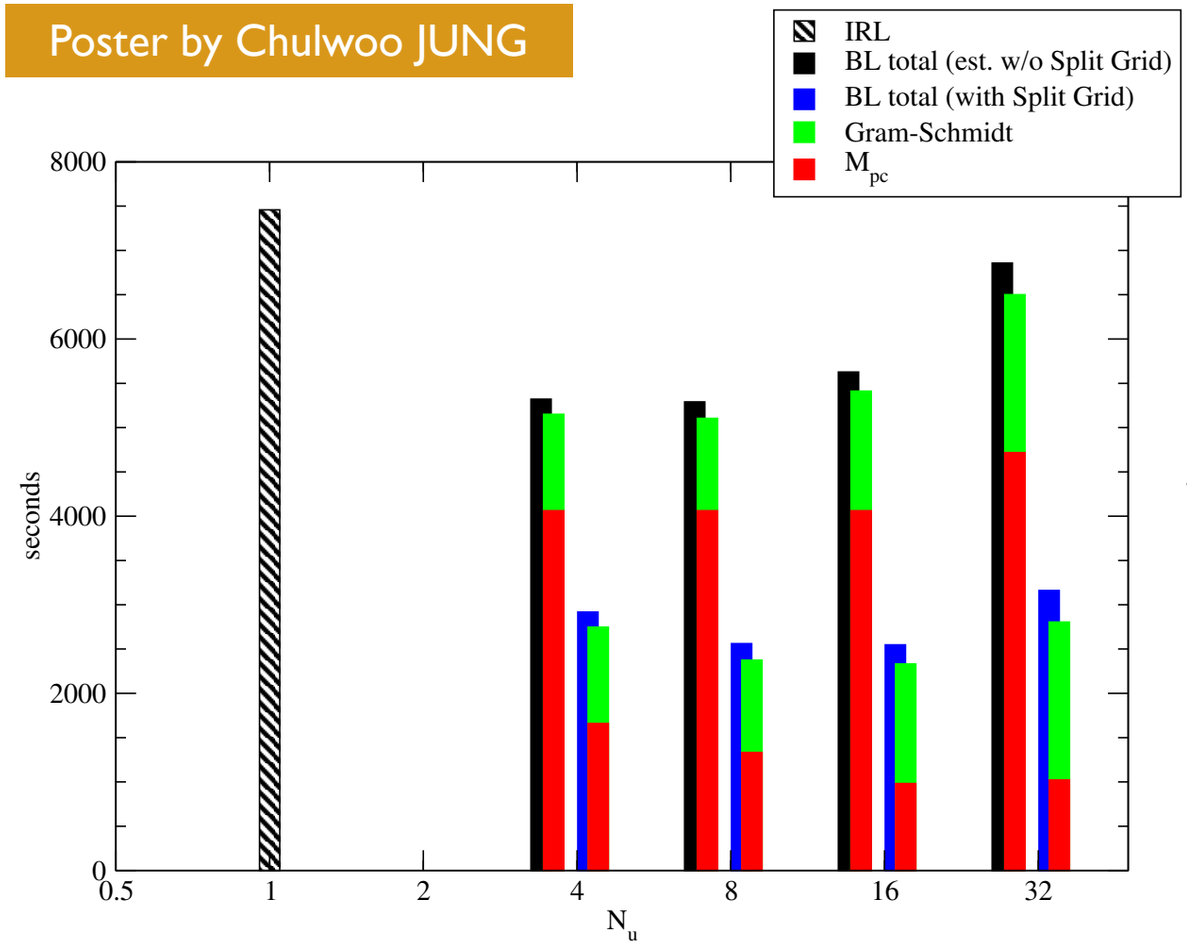
Split Grid and Block Lanczos for DWF

Jang and Jung combine split-grid and block lanczos to generate eigen pairs for DWF efficiently.

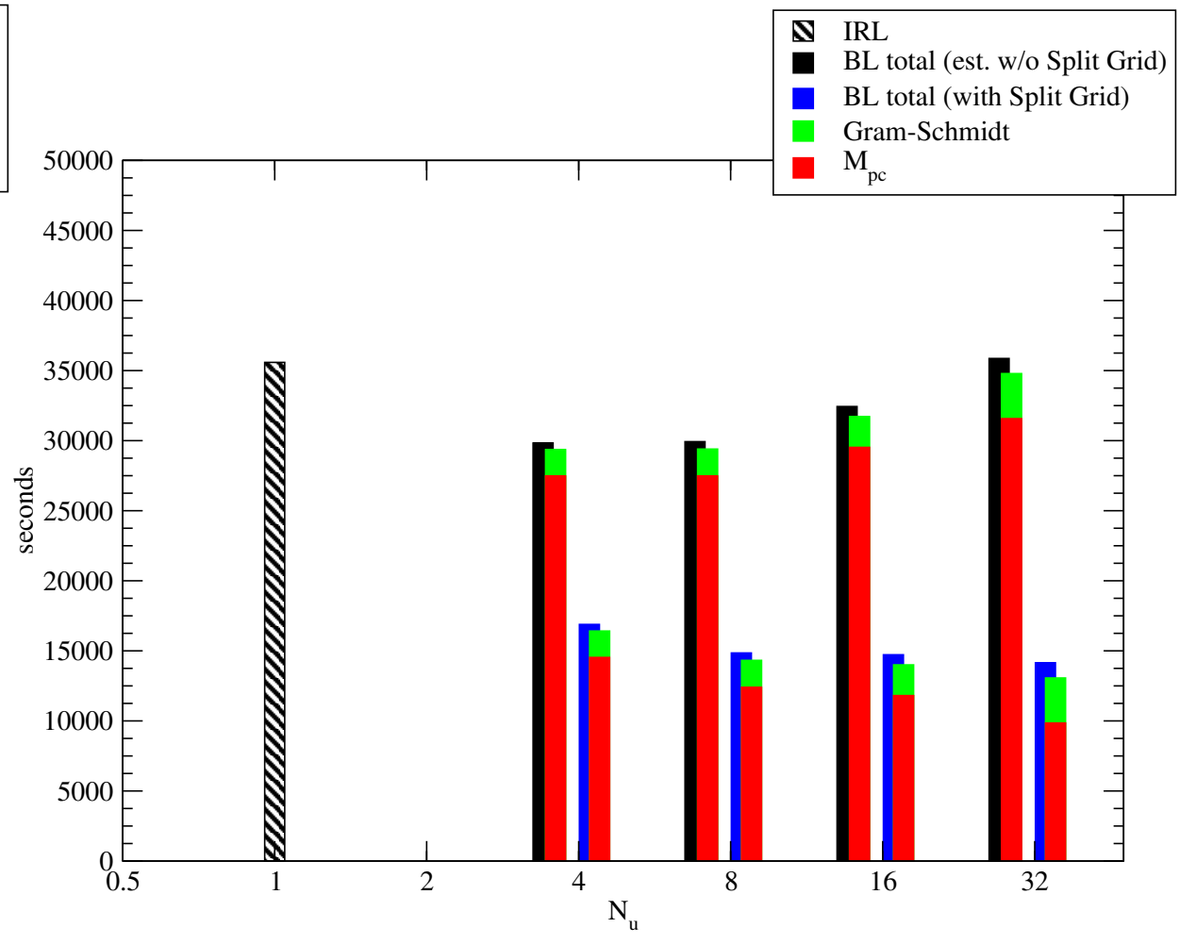


ALCF Theta (KNL 7230, 64 cores/node +Cray XC40)

Poster by Chulwoo JUNG



24ID ensemble (128 nodes)



48ID ensemble (512 nodes)

Factor of ~ 3 improvement seen. Further optimization in GS can give another factor of 2.

Other Topics

- **Studies of reversibility violations in HMC**
  - SU(2), Urbach, arxiv:1710.07526
  - SU(3), Urbach, in preparation
- **Multi-level integration for meson propagators**
  - Presented by Dr. Alessandro NADA on 23 Jul 2018 at 3:00 PM
- **Multi-level integration for meson propagators: disconnected contributions**
  - Presented by Dr. Tim HARRIS on 23 Jul 2018 at 3:20 PM

3. Machines/Software

Software Considerations

Given the diversity of current and upcoming HPC architectures, you may want to design your software with following considerations:

- **Performance Portability**
  - How much tradeoff do you want to make between performance and portability?
  - Is it possible to design your software to be portable and at the same time reasonably performant?
- **Programming Models**
  - What programming models do you want to use c.f. performance portability?
  - OpenMP, OpenACC, OpenCL, CUDA, Kokkos, etc.
- **Programming Languages**
  - Probably a very personal choice, but may affect usability, extensibility and performance.
- **Data Layout**
  - Important since LQCD computations are bandwidth bound.
  - Need to maximize data reuse and coalescence (GPU).

Intel Many/Multi-Core Processors

- Surprising news at SC17: Intel discontinued the successor to Intel "Knights Landing".

- In Q3, 2017, Intel released the new Xeon Scalable Processors (formerly code-named "Skylake").
- Support Intel Advanced Vector Extension 512 (AVX-512)
- 2 hyper threads per core
- No MCDRAM. Has fast L3 cache.

Edition	Number of Cores	AVX512 FMA units	Maximum number of sockets
Platinum	Up to 28	2	8
Gold	Up to 22	1-2	4
Silver/Bronze	Up to 12	1	2

LQCD Software for KNL/Skylake

- **Four ways to vectorization:**
  - Leave it to the compilers
  - Use compiler directives: #pragma omp simd
  - Use compiler intrinsics
  - Write assembly
- **LQCD software that supports AVX512:**
  - **OpenQCD**: intrinsics
  - **Bridge++**: intrinsics
  - **Stephan Durr's Fortran code**: OpenMP
  - **Grid**: intrinsics/assembly
  - **QPhiX**: intrinsics (https://github.com/JeffersonLab/qphix)
  - There may be others…

KNL code should also work on Skylake with the core-avx512 instruction set.

AVX512 Extension to OpenQCD

http://github.com/sa2c/OpenQCD-AVX512

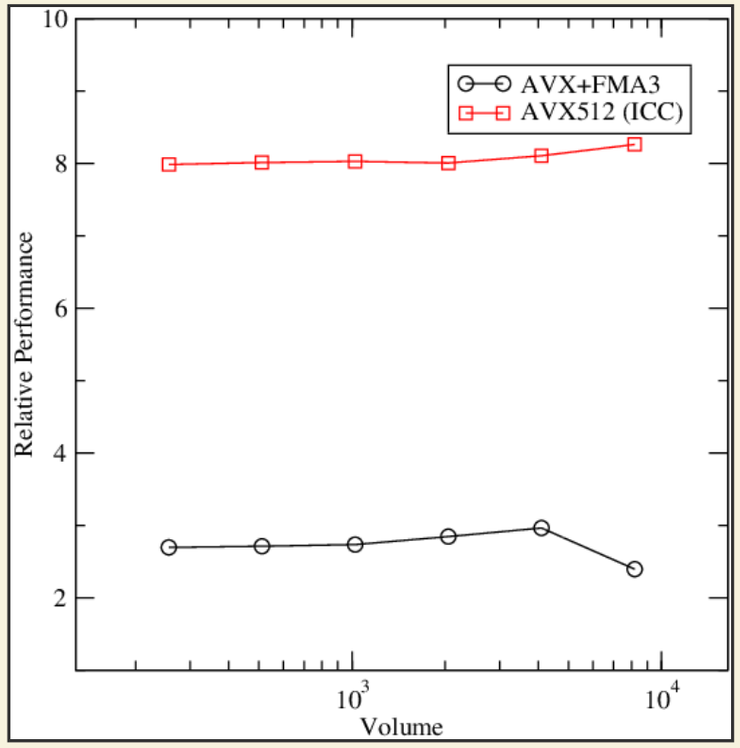
Jarno RANTAHARJU on **25 Jul 2018** at **4:50 PM**

- OpenQCD (http://luscher.web.cern.ch/luscher/openQCD/) mainly supports O(a)-improved Wilson fermion.
- V1.6 has support for SSE and AVX2
- Authors added support for AVX512 by using intrinsics [Bennett, Dawson, Mesiti & Rantaharju, arXiv:1806.06043]
- Paying particular attention to register memory use and cache use

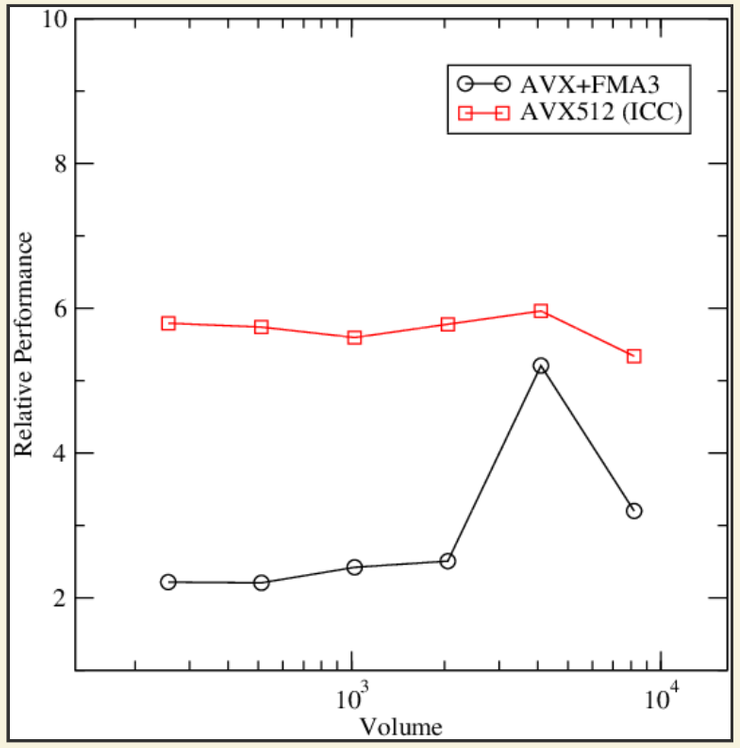
Single Core Performance

Against vanilla baseline

Skylake



single Precision



Double Precision

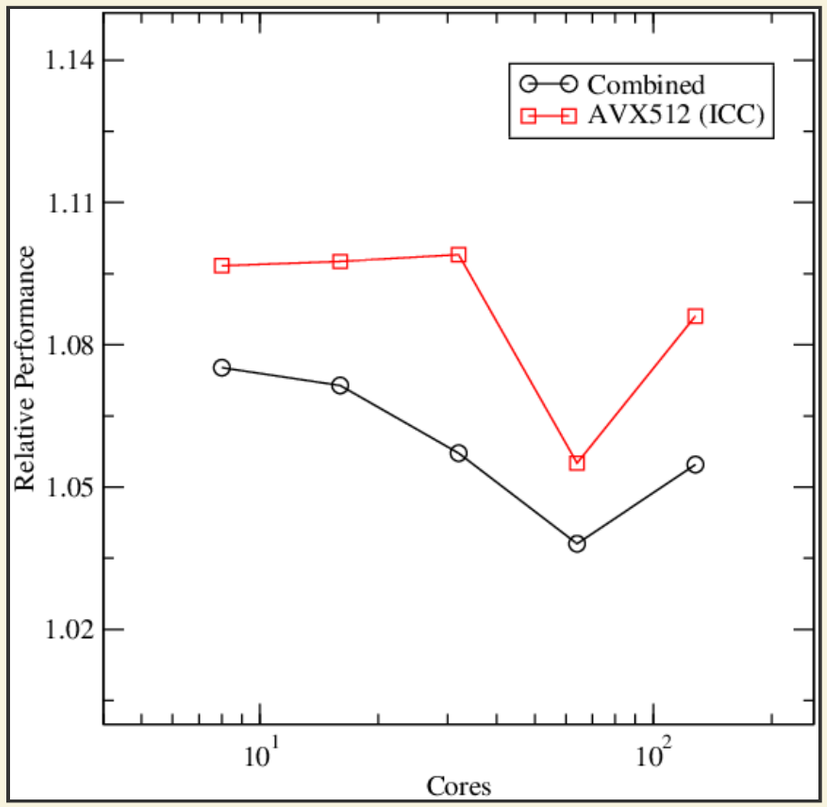
AVX512 Extension to OpenQCD

http://github.com/sa2c/OpenQCD-AVX512

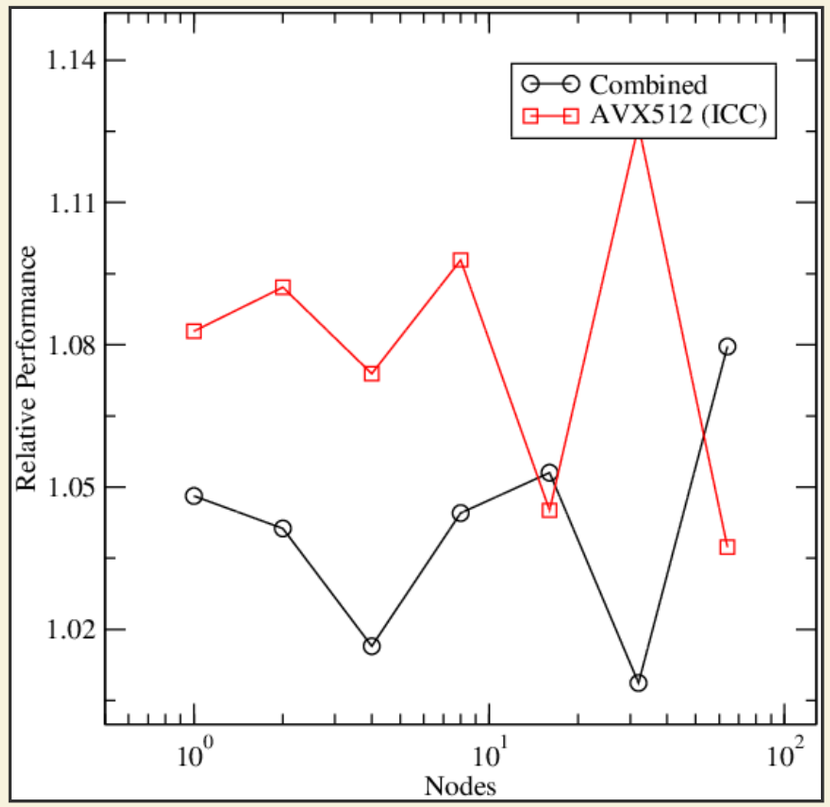
Jarno RANTAHARJU on **25 Jul 2018** at **4:50 PM**

- Going to multi-core/multi-node suffers from memory bandwidth bottleneck
- Overall gain over vanilla code:
  - 5%-10% on Skylake
  - 20%-40% on KNL

Strong Scaling on a Skylake Cluster



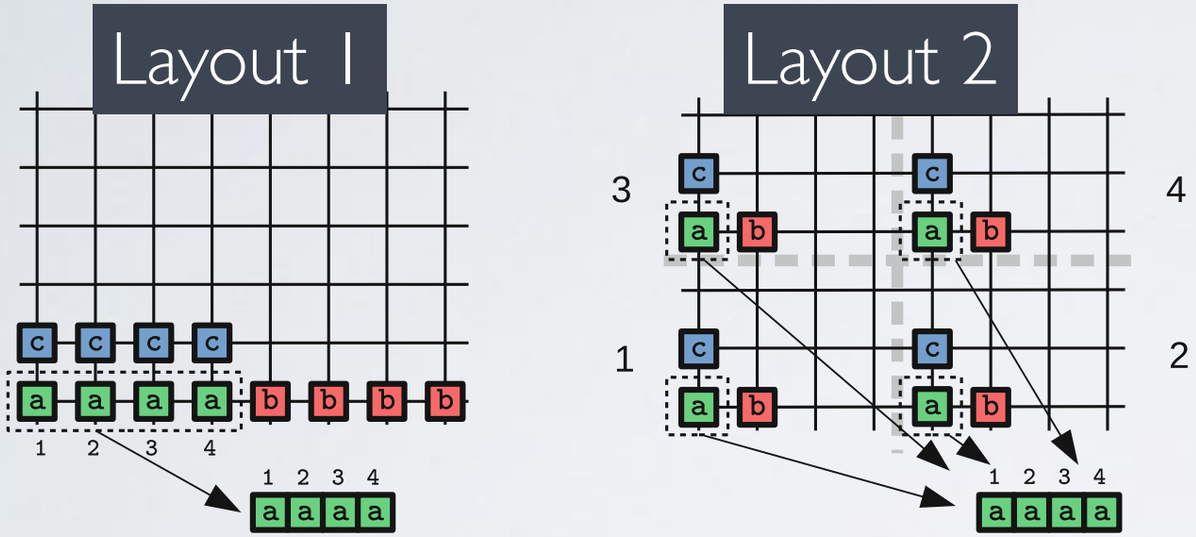
V=24^3x48



V=24^3x48

Bridge++

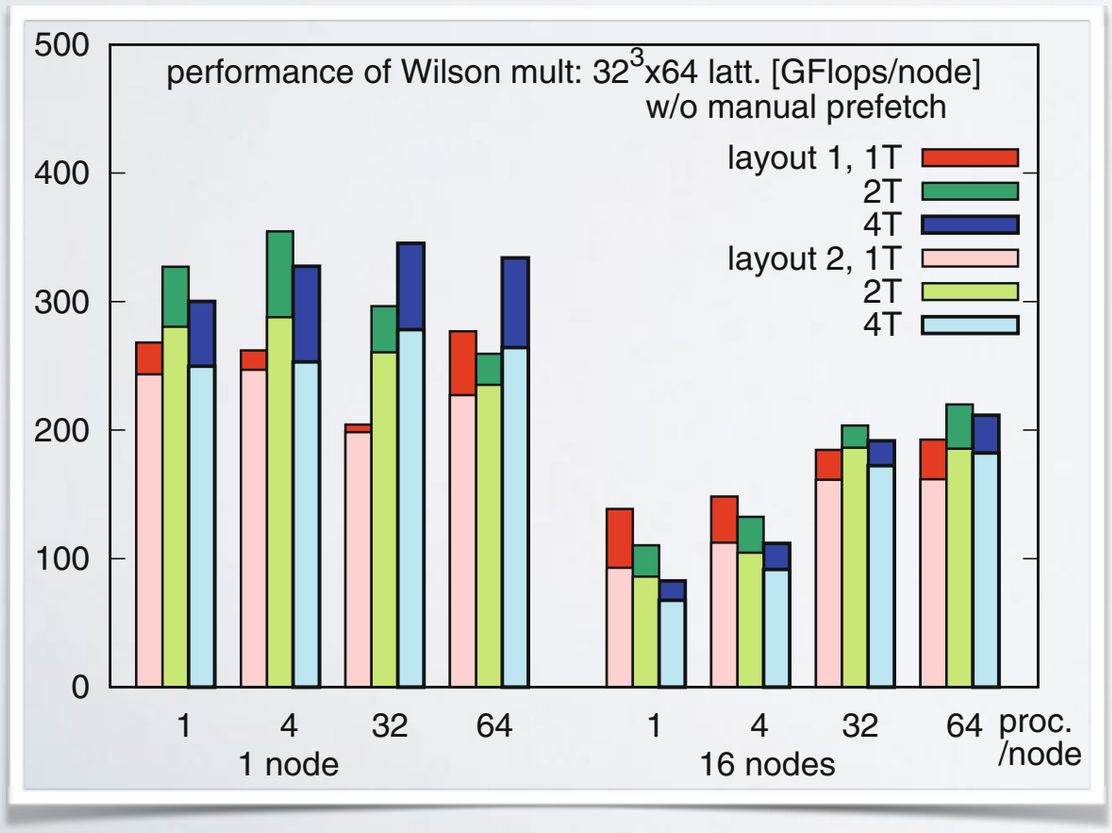
- Object-oriented C++ LQCD library (http://bridge.kek.jp/Lattice-code/).
- For AVX512, experimented with two different SIMD data layouts.

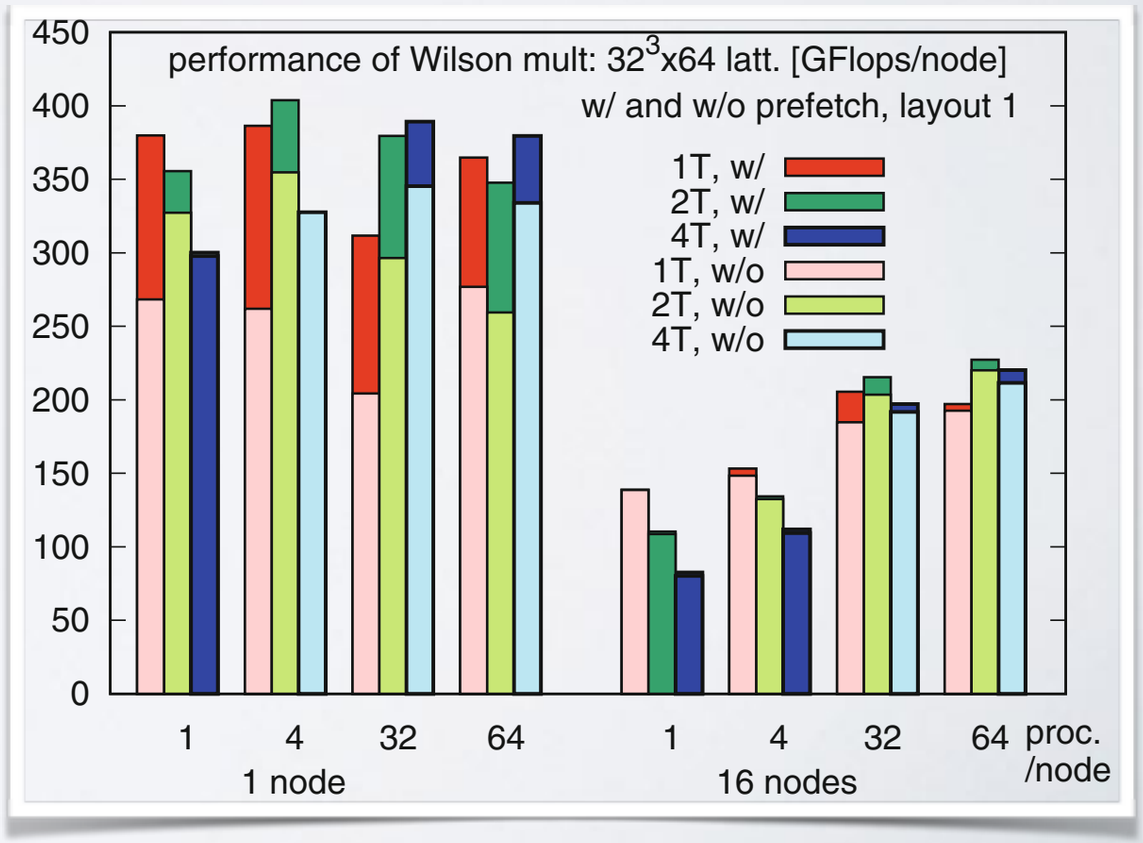


Kanamori & Matsufuru,ICCSA 2018

KNL nodes on Oakforest-PACS

Effects of prefetching

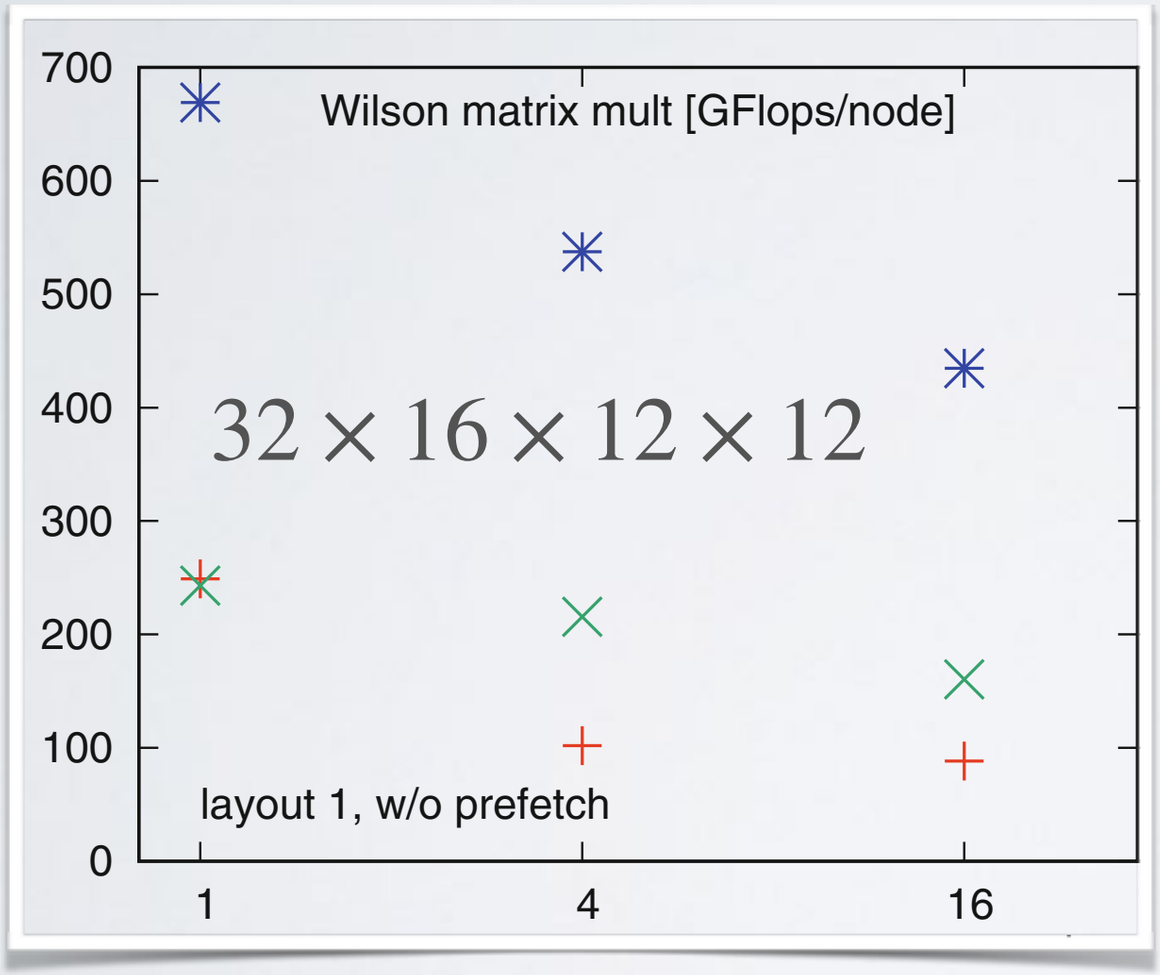




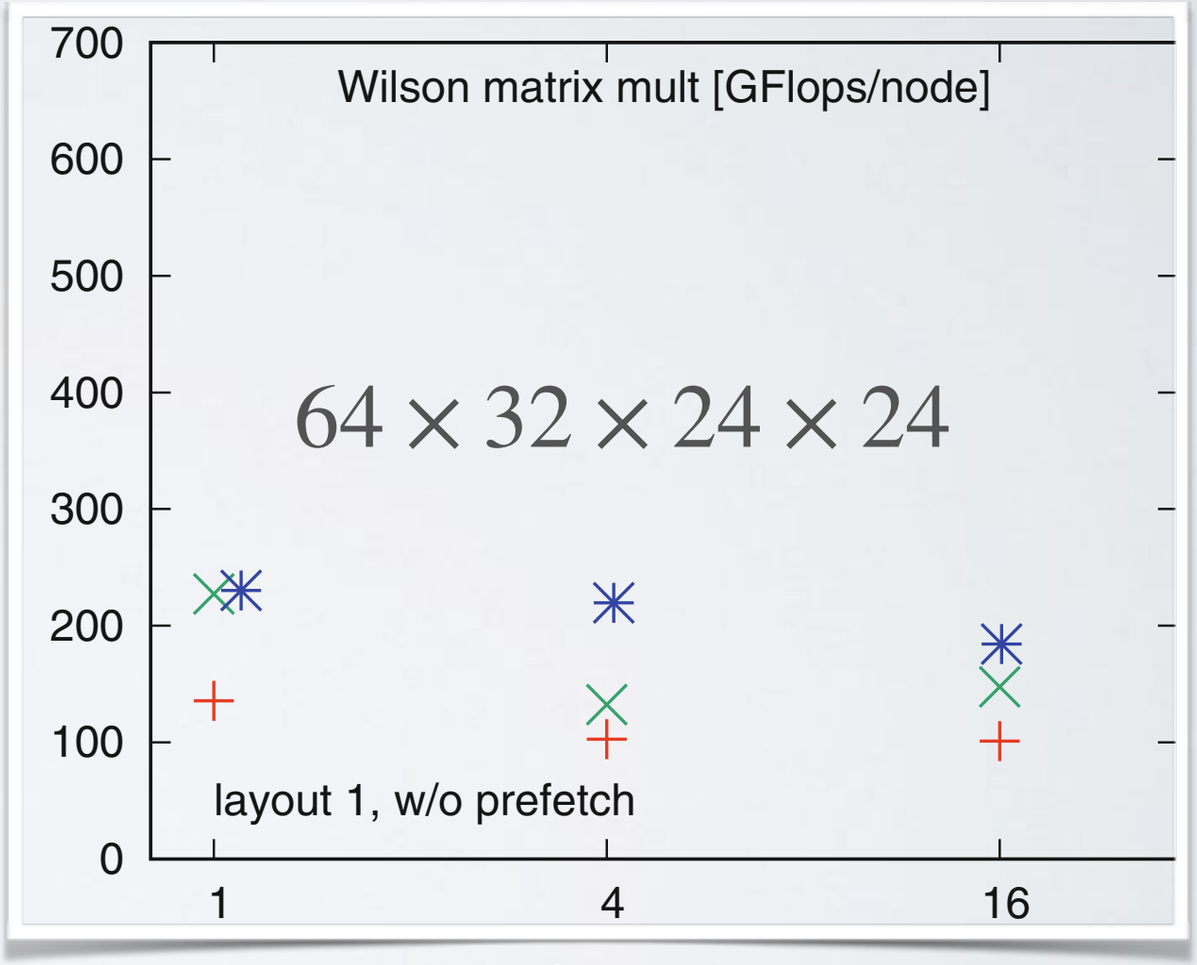
Bridge++

- For Skylake, uses the same KNL code. Kanamori & Matsufuru,ICCSA 2018
- Prefetching has no effect.
- Cache reuse is important.

ITO: SKylake Gold 6154, 18 cores, dual-socket, 24.75 MB L3 cache



# of nodes



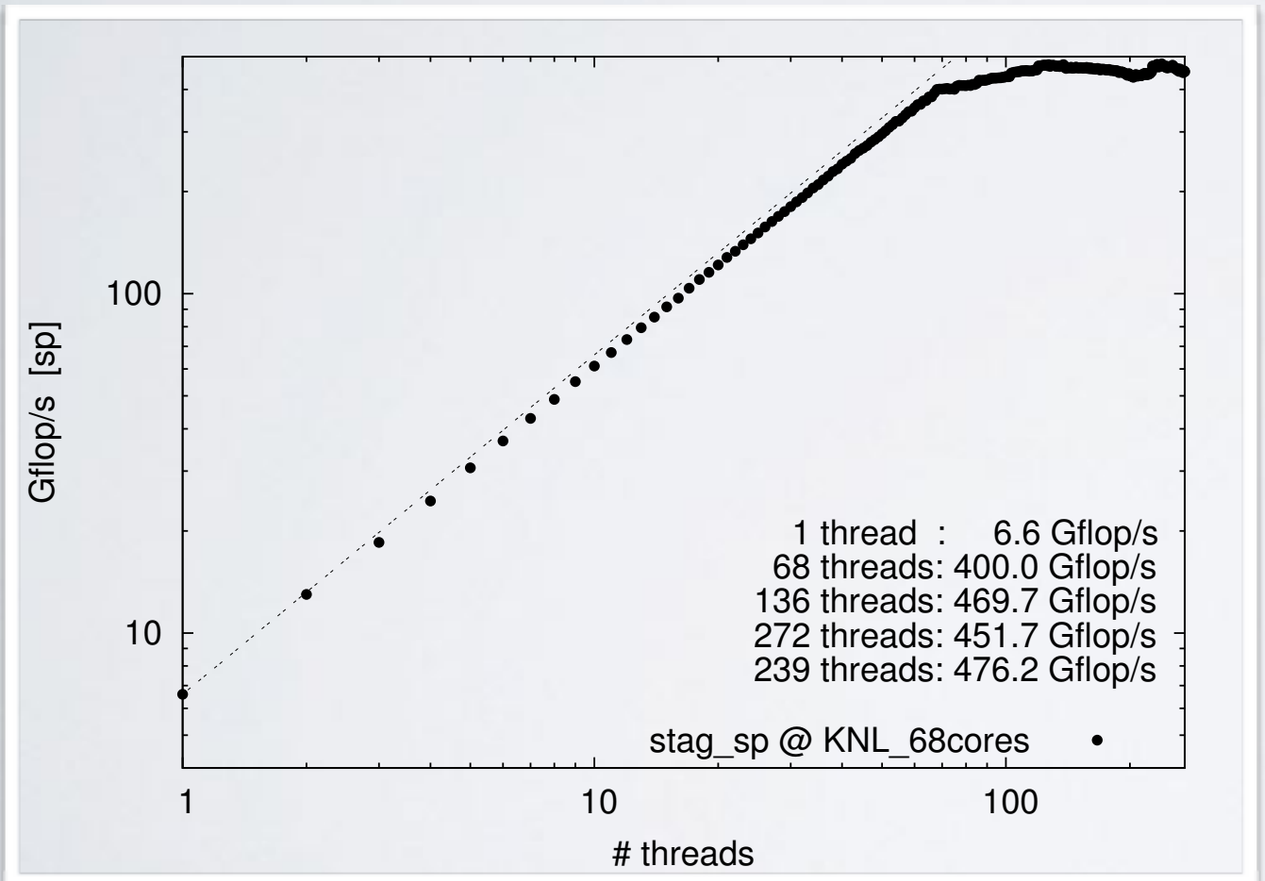
# of nodes

Weak Scaling

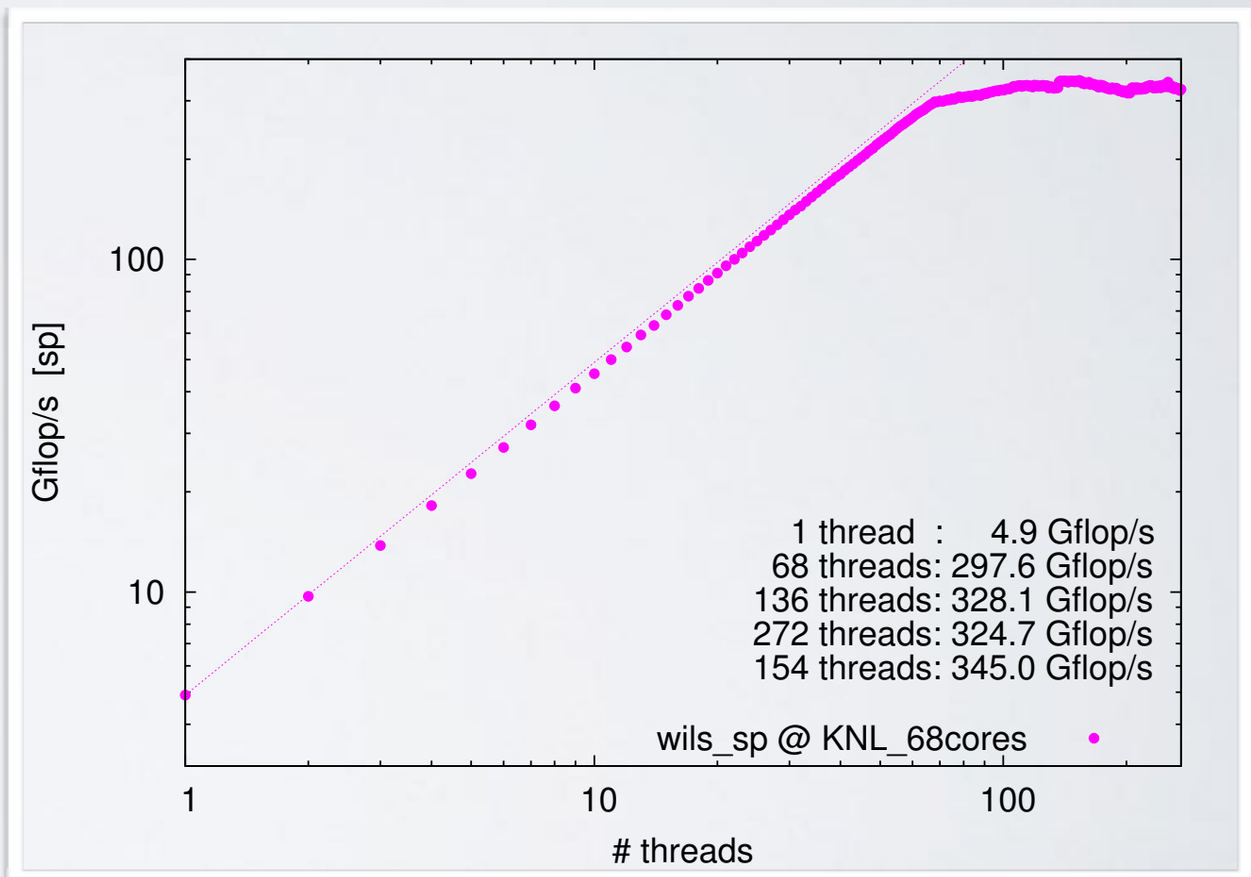
Three Dirac operators on two architectures With one piece of code

Stephan DURR on 23 Jul 2018 at 5:10 PM

- Uses a high-level language (Fortran in this case) and simple OpenMP SIMD pragmas
- Implemented three Dirac operators: Brillouin, Wilson and Staggered
- Targets different architectures (portability): KNL and core i7 performances reported



Staggered, Single Precision, KNL



Wilson, Single Precision, KNL

Grid Status Update

- Grid is a C++ Lattice QCD library. [Boyle, Yamaguchi, Cossu & Portelli, arXiv: 1512.03487]
- https://github.com/paboyle/Grid

Boyle, Cossu, Portelli, Yamaguchi + additional contributions

- Continue multicore performance portability
  - Intel, AMD multicore & many core
  - IBM BlueGene/Q
  - NEW U. Regensburg (Meyer/Wettig) ARM/Neon, ARM/SVE
- New Functionality
  - Hadrons measurement package (Portelli + others)
  - All-to-all and AMA (O'Haigan)
  - Wilson/Clover multigrid (Richtmann, talk this conf).

Poster by Nils MEYER on 24 Jul 2018 at 6:45 PM

Daniel RICHTMANN on 23 Jul 2018 at 4:50 PM

Common source GPU port is functional, but incomplete (Boyle)

- Broadly follow strategy developed in collab. with USQCD ECP project https://arxiv.org/abs/1710.09409 (Boyle, Clark, DeTar, Lin, Rana, Vaquero)
- 80%-95% of QUDA performance from native code.
- Data parallel site local expressions saturate memory bandwidth
- Compiles for GPU "naturally" using advanced C++ features

Peter Boyle

Grid Single-Node Performance

MultiRHS Wilson operator, DWF. Choose optimal local volume on each.

Architecture	Cores	GF/s (Ls x Dw)	peak
Intel Knight's Landing 7250	68	770	6100
Intel Knight's Corner	60	270	2400
Intel Skylakex2	48	1200	9200
Intel Broadwellx2	36	800	2700
Intel Haswellx2	32	640	2400
Intel Ivybridgex2	24	270	920
AMD EPYCx2	64	590	3276
AMD Interlagosx4	32 (16)	80	628
1xVolta	84 SM	1900	15700

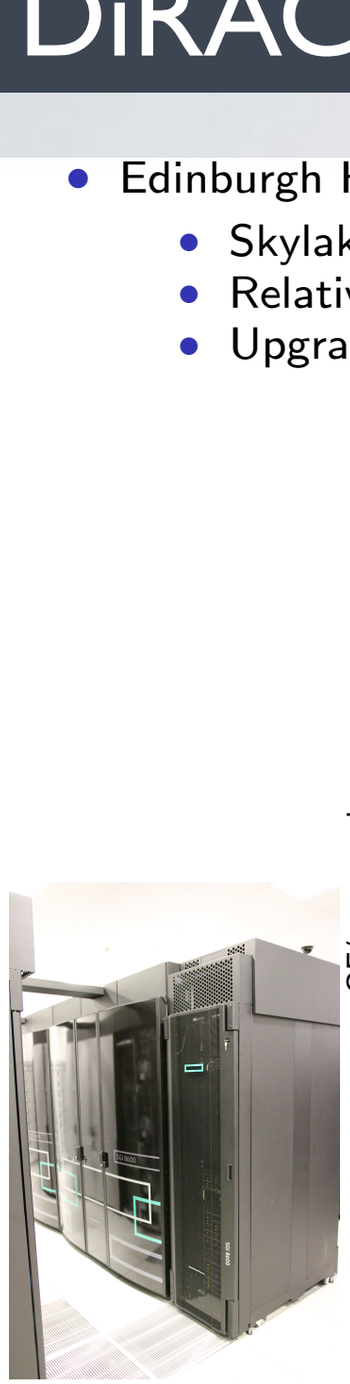
Notes:

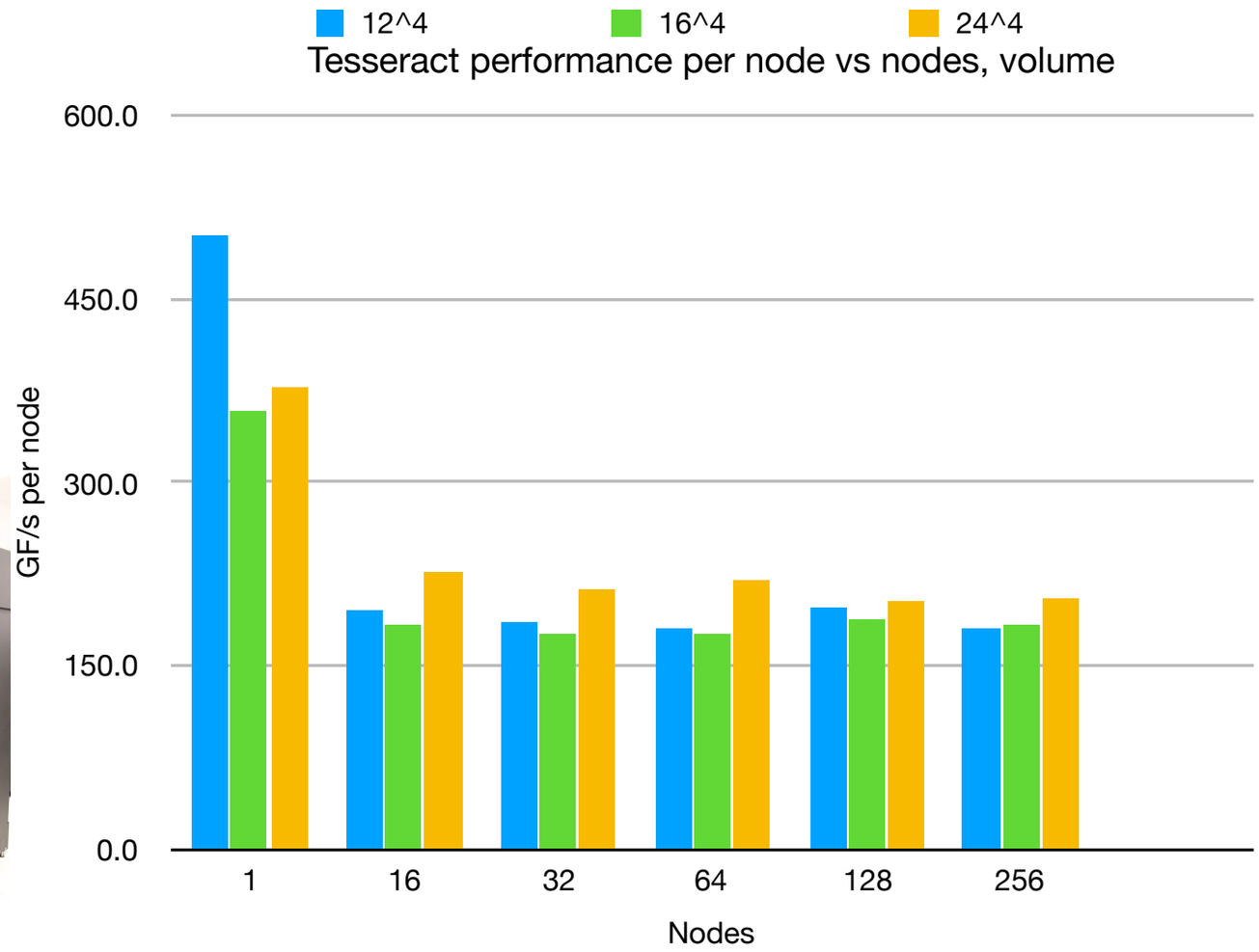
- Dropped to inline assembly for key kernel in KNL and BlueGene/Q
- EPYC is MCM; ran 4 MPI ranks per socket, one per die
- GPU-port not yet production ready

Peter Boyle

DiRAC HPE ICE-XA, hypercube network

- Edinburgh HPE 8600 system, 844 dual socket nodes (March 2018)
  - Skylake Silver 4116, 12 core; Omnipath interconnect
  - Relatively cheap node: enables high node count and scalability
  - Upgrade to over 1400 nodes in 2018/19.





- 512 nodes topology aware code bidirectional 19GB/s (25GB/s peak)
  - 76% wirespeed using every link in system concurrently
- Collaboration w. Intel, Brookhaven Lab: introduced concurrency in Intel MPI, OPA
- https://arxiv.org/pdf/1711.04883.pdf

Peter Boyle

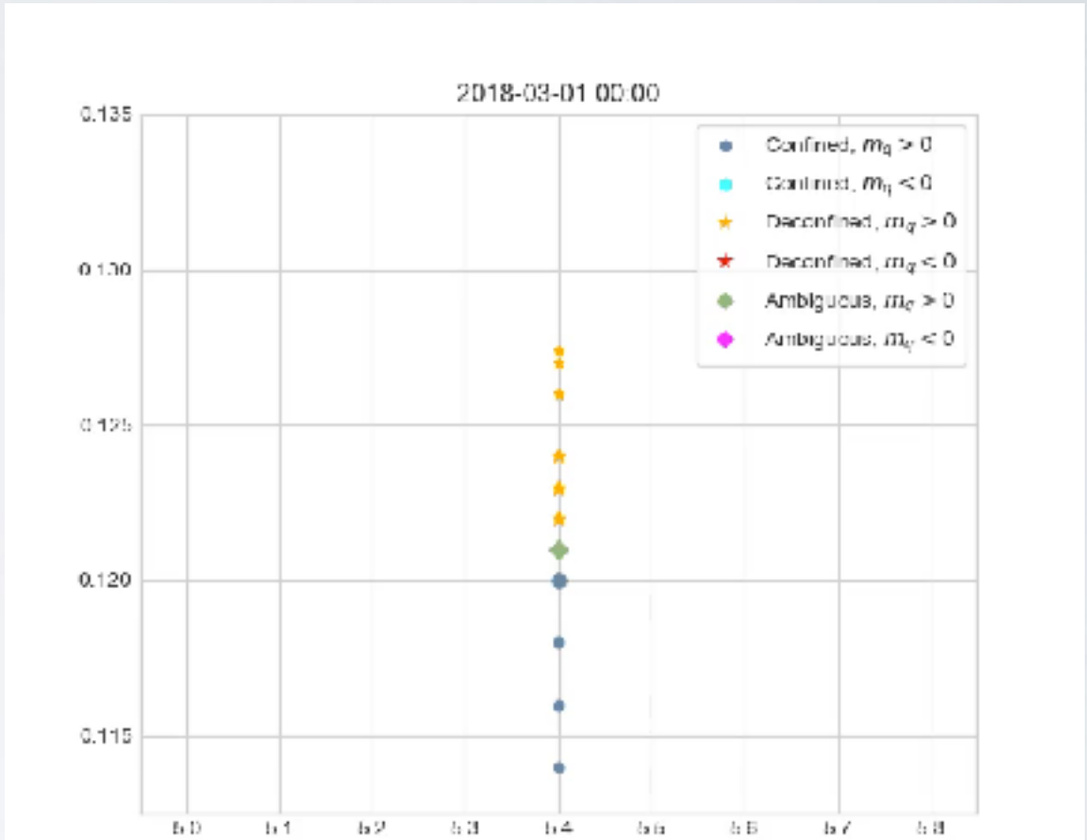
Workflow Management Software

Colorado lattice group* is developing tools for <u>end-to-end workflow management and automation</u>:

- Central SQL database stores all information, from ensemble parameters to gauge config metadata, correlators, intermediate analysis results & final outputs

- Workflow manager (e.g. **taxi** Python module - <u>https://github.com/dchackett/taxi</u>) manages simulations and measurements on remote machines

- Automation scripts on the back end "close the loop", allow for cyclic workflows that start and end in the database. Enables things like <u>automated phase diagram exploration</u>:



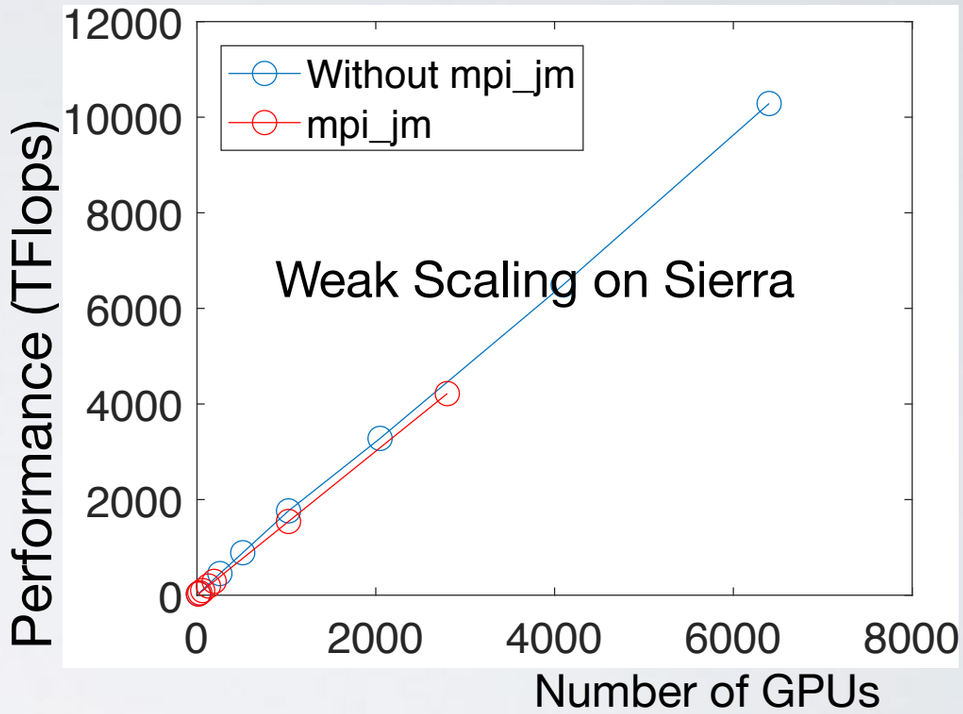
Wilson finite-T phase diagram for SU(3) $N_f=2$, $12^3 \times 6$. 8200 configs across 138 ensembles. Some adjustments and debugging through the run, but **no** simulations launched by hand!

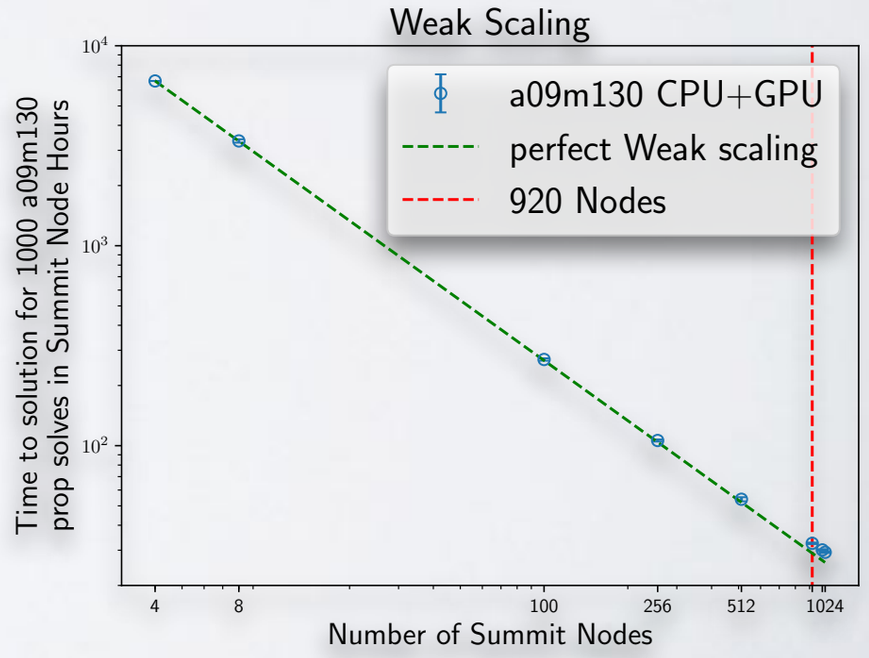
**V. Ayyar, D.C. Hackett, W.I. Jay, E.T. Neil*

Workflow Management Software

developed a job management system for efficiently distributing work over a large number of compute nodes.

- **METAQ** - Bash script that bundles a large number of small-node-count jobs.
- **mpi_jm** - Other issues motivated a second generation job management tool:
  - Packing: Independent CPU and GPU jobs on the same nodes. 2GPU with 4 GPU on same node.
  - Interconnect: Jobs always placed on node sets with local connections. Yields higher and more consistent performance.
  - Low overhead on service node. Jobs started with MPI_Comm_spawn. Scheduling and job control/monitoring is implemented in C++.
  - Job collection/generation and configuration is done in Python - very configurable.





EPJ Web Conf. 175 (2018) 09007 Berkowitz, Jansen, McElvain, and Walker-Loud

New/Emerging Architectures

- It's possible that exascale computers will be based on unconventional architectures.
- Japan already announced that their Post-K exascale computer will be based on 512-bit ARM v8 SVE (Scalable Vector Extension) instruction set.

- **ARM** (Nils Meyer, poster)
- **FPGA** (2 presentations at this conference)
- **Sunway Taihu Light** (Ming Gong, poster)
- **Quantum Computers** (3 presentations at this conference)

Lattice QCD on ARM SVE

QPACE 4 project

- Aims to design an ARM SVE environment optimized for Lattice QCD
  - Prototype at the University of Regensburg, Germany, by 2020
- Pursues evaluation and enhancement of existing SVE software toolchain and upcoming SVE hardware technologies for Lattice QCD applications
- Enables Lattice QCD code optimized for SVE architectures, e.g., the Japanese flagship supercomputer "Post-K" announced for 2021

ARM SVE

- The ARM Scalable Vector Extension (SVE) targets the HPC market [1]
- Key features of SVE hardware
  - Wide vector units, ranging from 128 bit to 2048 bit
  - Vectorized native 16 bit floating point operations, including arithmetics
  - Vectorized arithmetics of complex numbers
  - The silicon provider chooses the vector register length and defines the performance characteristics of the SVE hardware
  - The first SVE hardware officially announced is the Post-K, which comprises 512 bit wide vector units, 48 compute cores per node and high performance stacked memory [2]

Lattice QCD on ARM SVE (cont'd)

ARM SVE Development Environment

Arm C/C++/Fortran Compiler

Commercial C/C++/Fortran compiler from Arm for Linux user-space HPC applications with complete SVE support.

Arm Instruction Emulator

Run SVE binaries on existing Armv8-A hardware.

Arm Code Advisor

Simple, actionable advice to tune your HPC application on Arm. Combines static and dynamic information to produce actionable insights.

Enabling SVE in Grid

- We use ACLE for our SVE implementation to access the SVE features
- We minimize implications of the VLA programming model and bypass restrictions on usage of ACLE data types
  - We declare the vector length `VL` (in bytes) as a compile-time constant
  - Superfluous loops implied by VLA are omitted
  - The variety of predications is minimized to fit into the register file
  - The templated struct `acle<T>` (`T = double, float`) provides convenient access to ACLE definitions, e.g., predications and data types

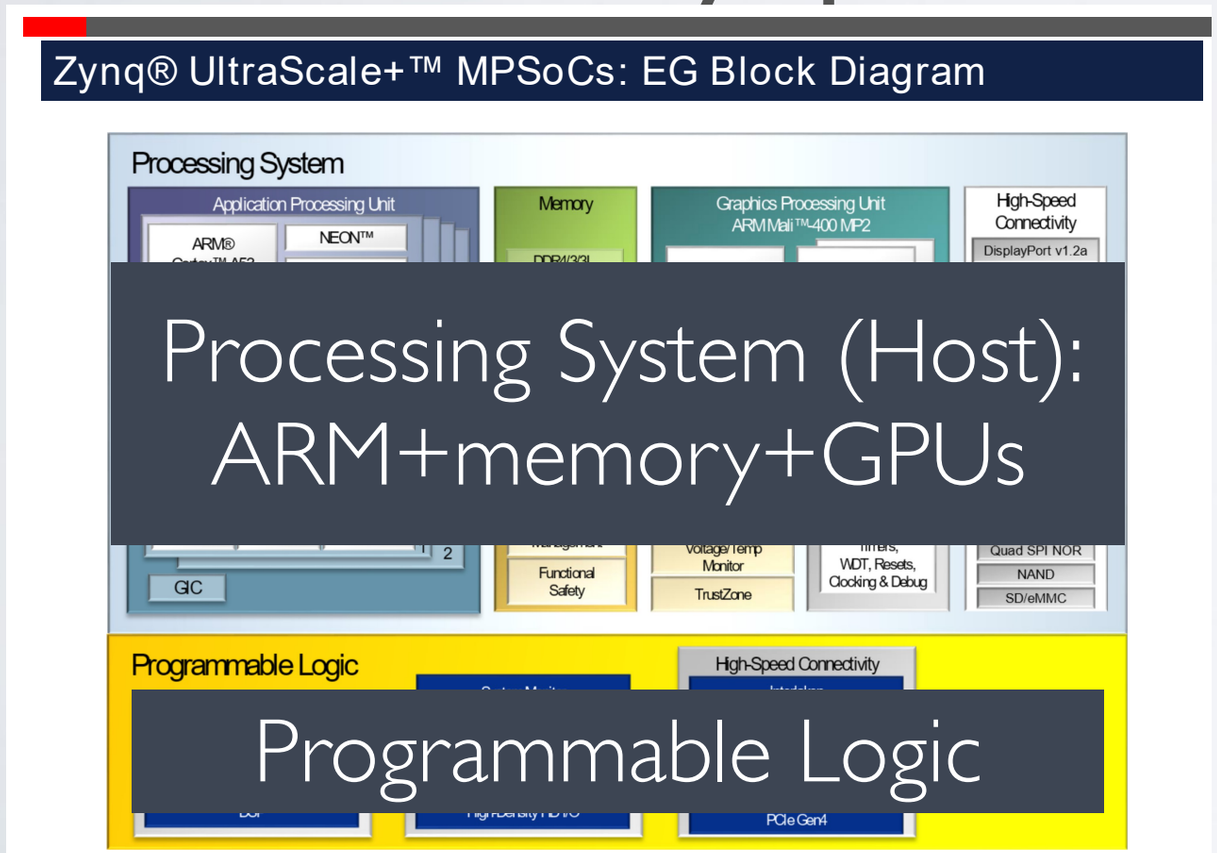
Poster by Nils MEYER on 24 Jul 2018 at 6:45 PM

Lattice QCD on FPGA

Poster by Piotr KORCYL on 24 Jul 2018 at 6:45 PM

FPGA: **Field Programmable Gate Arrays**
Dynamic reconfiguration
Natural parallelization
Power efficient
No OS overhead

Xilinx Zynq



CG on FPGA devices

$$
\begin{aligned}
&\psi \leftarrow \psi_0 \\
&r \leftarrow \eta - DD^\dagger\psi \\
&p \leftarrow r \\
&\textbf{while } |r| \geq r_{min} \textbf{ do} \\
&\quad r_{old} \leftarrow |r| \\
&\quad \alpha \leftarrow \frac{r_{old}}{\langle p|DD^\dagger|p\rangle} \\
&\quad \psi \leftarrow \psi + \alpha p \\
&\quad r \leftarrow r - \alpha DD^\dagger p \\
&\quad \beta \leftarrow \frac{|r|}{r_{old}} \\
&\quad p \leftarrow r + \beta p \\
&\textbf{end while}
\end{aligned}
$$

Run entirely on programmable logic

The rest run on ARM

8. Performance in GFLOPs

Estimates with * are based on the Zynq XCZU9EG board transfer infrastructure and assume 500MHz clock.

	only compute	with transfer
4	752	32*
3	376	30*
2	376	30*
1	188	28*
	2.1	1.3

Lattice QCD on Sunway Taihu Light

- **Sunway Taihu-Light**: #2 on Top 500
- Custom architecture
- May be one of the architectures for China's exascale supercomputers

Poster by Ming GONG on 24 Jul 2018 at 6:45 PM

- **Master code:** runs on MPE
  - Communications
  - Input/Output
- **Slave code:** runs on CPE (64 cores)
  - DSlash
  - MinRes Inverters
- Metaprogramming and genetic algorithms are used to investigate the feasibility of automatic code optimization
- Interface to Chroma and other packages is work in progress

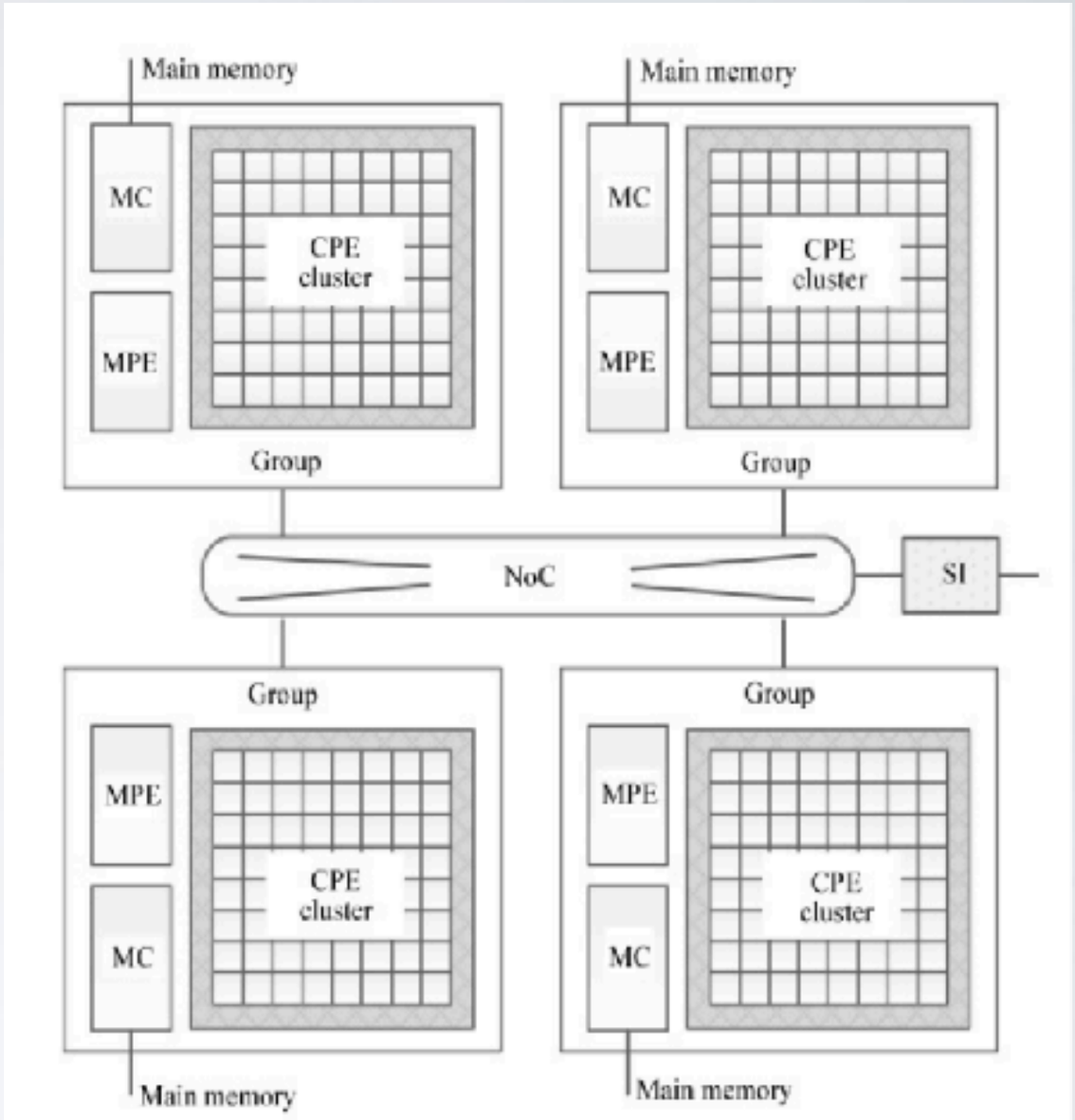


Figure 2: Basic Layout of a Node

Source: hpcwire.com

Quantum Computing

- Can quantum computers play a role in lattice QCD simulations?
- Increases in the number of qubits and the fidelity of quantum gates are reaching a threshold of becoming useful sandboxes for theoretical developments.
- Parallel presentations at this conference:

  Patrick DREHER on **24 Jul 2018** at **4:50 PM:**
  SU(2) Quantum Link Model

  Yannick MEURICE on **26 Jul 2018** at **9:10 AM:**
  Tensorial Formulation

  Jesse STRYKER on **25 Jul 2018** at **3:00 PM:**
  Hamiltonian Framework

- Plenary by John Preskill later at 11:45am.

FYI
SCIENCE POLICY NEWS FROM AIP
Bulletin

Quantum Legislation Leaps Forward

SHARE THIS

Publication date: 5 July 2018
Number: 81

National Quantum Initiative Act gains steam, spurred by international competition

Late last month, House Science Committee Chair Lamar Smith (R-TX) unveiled the "National Quantum Initiative Act," which would launch a 10-year interagency effort to accelerate progress in quantum information science (QIS) and technology development. Co-sponsored by Ranking Member Eddie Bernice Johnson (D-TX) and 30 other committee members, it received unanimous support at a June 27 meeting held to advance a set of bills to the full House.

4. Summary and Outlook

Summary and Outlook

- As we approach the physical and continuum limit in lattice QCD simulations, algorithms need to be improved.
  - Ongoing activities to reduce the critical slowing down in HMC
  - Block solvers, multigrid solvers can accelerate Dirac matrix inversions dramatically.
- Algorithms and machines are no longer separate topics.
- Algorithms need to be aware of the hardware architectures and configurations.
- Communication avoiding algorithms are likely essential for good strong scaling on pre-exascale and exascale machines.
- Exciting new possibilities for the upcoming machines.
- LQCD needs to be ready, both in terms of algorithms and software.