BG/Q Code status

Chulwoo Jung
Brookhaven National Laboratory
(with sincere apologies & thanks to Bob Mawhinney,
Balint Joo, James Osborn..)

May 4, 2012
USQCD All hands meeting
Fermilab

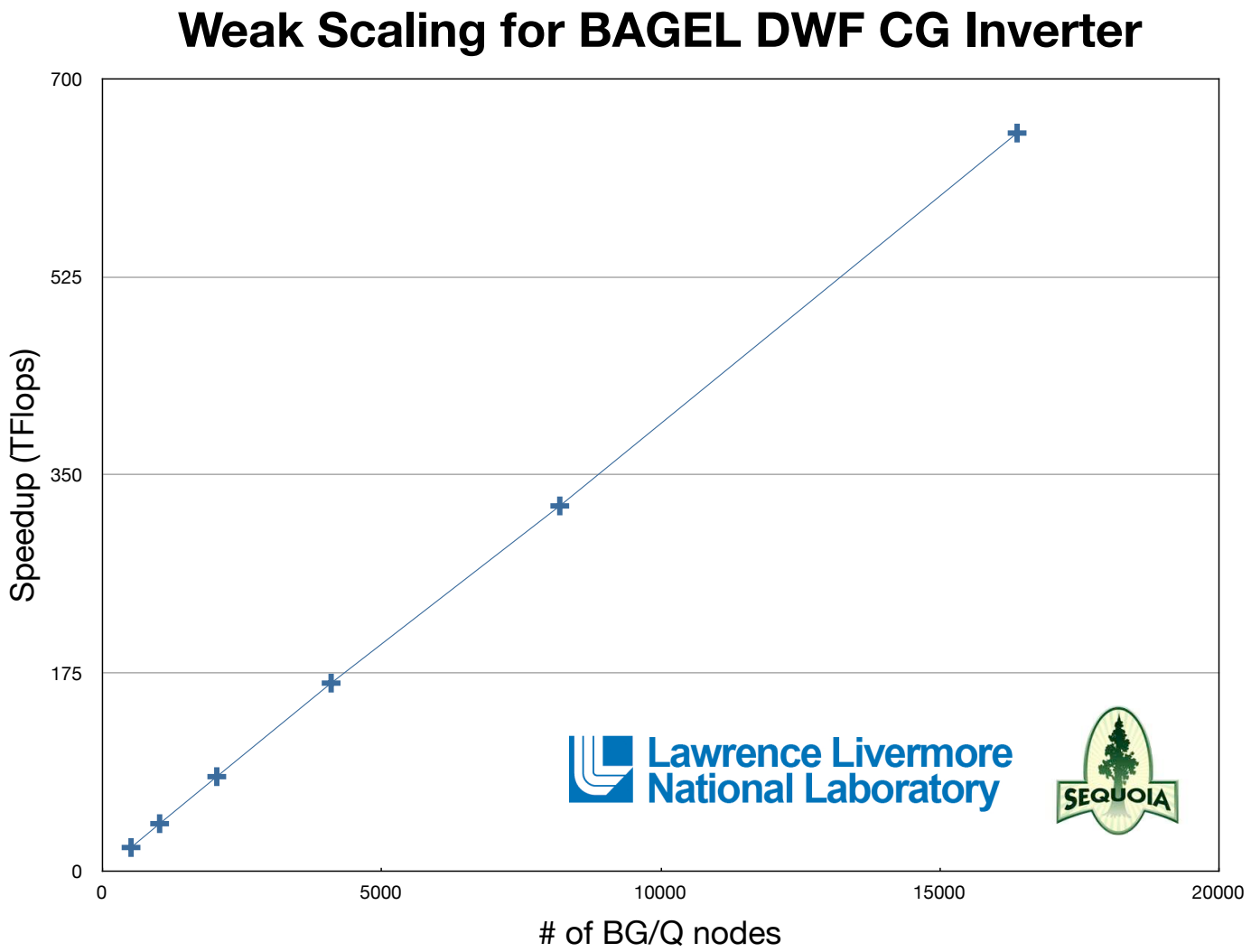
IBM BG/Q

	BG/L	BG/P	BG/Q
Core/node	2	4	16+1(+1)
SIMD/core	2	2	4
Clock speed(Mhz)	700	850	1600
Threads/node	2	4	16×4
Peak/node(GFlops)	5.6	13.9	205
L1/core(KB)	32	32	16
L3/node(MB)	4	8	32 (L2)
Memory(GB)/node	1	2	16
DDR Bandwidth(GB/s)	5.6	13.6	40
Torus Dimension	3	3	5
Bandwidth (GB/s)	2.1	5.1	∼40
Watt/Rack (KW)	∼20	∼40	∼80
Linpack GF/W	0.23	0.37	1.68

BFM CG performance (GFlops/node,1 rack, w SPI)

Fermion	V	L_s	Double	Single*
DWF(Shamir)	$6^3 \times 8$	8	34	43
	$6^3 \times 8$	16	41	51
	$8^3 \times 8$	8	43	55
	$8^3 \times 8$	16	34	60
	$10^3 \times 8$	8	32	59
	$10^3 \times 8$	16	24	52
	$12^3 \times 12$	8	20	47
Wilson	$6^3 \times 8$		14	15
	$8^3 \times 8$		24	26
	$10^3 \times 8$		33	37
	$12^3 \times 12$		26	50
WilsonTM	$6^3 \times 8$		12	13
	$8^3 \times 8$		21	23
	$10^3 \times 8$		29	33
	$12^3 \times 12$		24	45

Scaling plot (LLNL/EPCC up to 16 rack)



Tests were performed with the STFC funded DiRAC facility at Edinburgh

up to

16K $8^4 \times 8$ local volume double precision

Integration/optimization status

BFM/BAGEL (Peter Boyle)

- DWF(5d,4d), Mobius, Wilson, WilsonTM Inverters, with QMP and/or SPI
- QDP++ interface included. In use with UKHADRON(Chroma)
- Provide threading framework rather than just dslash for other routines(eg. Arnoldi/Lanczos(R. Arthur) EigCG (Qi Liu))
- Single precision with SPI not yet fully debugged/optimized. So far only works with 1 MPI per node & contiguous mapping (logical nearest nodes should be also physically nearest) A bit slower with QMP/MPI
- Clover? : Similar structure to WilsonTM. Just needs the clover term!

CPS

- DWF(5d), WilsonTM integrated into CPS. Mobius in progress(Hantao Yin).
- Mixed precision solver, MADWF.... forthcoming
- Other parts??
  Relevant part for DWF evolution threaded with openMP, getting 5-10GFlops in double precision.
  So far only with openMP and GCC, no QPX, room for improvement.
  A lot of routines in Lattice QCD are memory bandwith bounded (flops/Byte < 1)
  $\rightarrow$ should be able to get reasonably close to "theoretical" peak by organizing the code to minimize the bandwidth, with better compiler(XLC..)

Chroma (From Balint Joo)

- Current status
  - Chroma built and run on prototype hardware in 2011
  - No linkage with optimized/assembly kernels yet
  - Currently working on deployment at ANL (also LLNL) and Bagel/BFM linkage
- Targets for 2012
  - Deploy QDP++/Chroma, link to optimized libraries (e.g. Bagel)
  - Optimize important QDP++ expressions
  - Port and check effectiveness of site-vectorized code for BG/Q
- Threading?
  - QDP++ has a simple thread dispatching mechanism
  - Should work with OpenMP and QMP (if it is ported)

MILC : See James' talk!

Software Targets for 2012

- Site Vectorized Dslash and Intel MIC (c.f. Chips talk earlier)
  - In collaboration with Intel Parallel Computing Labs
  - Aim initially for "level 3" Wilson-Clover solver
  - QDP++ & Chroma with compatible data layout ('scalarvec')
  - All x86 compatible targets benefit: Xeon, Interlagos (& Cray XE)
- BG/Q
  - Deploy QDP++/Chroma, link to optimized libraries (e.g. Bagel)
  - Optimize important QDP++ expressions
  - Port and check effectiveness of site-vectorized code for BG/Q
- Gauge Generation on GPU based Leadership Class Systems
  - Chroma on CPU + QUDA, Chroma over QDP-JIT + QUDA
  - In collaboration with QUDA developers, QDP-JIT with Frank Winter
- Chroma general maintenance: e.g. integrating updated QIO etc

Current Status

- Site Vectorized Dslash
  - Dslash exists, ongoing optimization & cleanup
  - Most optimization so far on single socket,
  - NUMA aware branch, & MPI boundary communications exist.
- GPUs
  - Have run Wilson HMC (2 flavor, with Hasenbusch prec.) on TitanDev
  - Both Chroma(CPU)+QUDA and Chroma(QDP-JIT)+QUDA
  - For Cray XK, QDP-JIT needs kernels pre-generated elsewhere
  - Work needed on Clover, and non-solver Level 3
  - Concern: lack of mixed precision multi-shift solver for RHMC.
    - various workarounds available (eg. solve in SP, polish up in DP etc)
- BG/Q
  - Chroma built and run on prototype hardware last year
  - No linkage with optimized/assembly kernels yet
  - Currently working on deployment at ANL (also LLNL) and Bagel BFM linkage

NVIDIA

QCD on GPUs: Current Status and Future Plans

Mike Clark

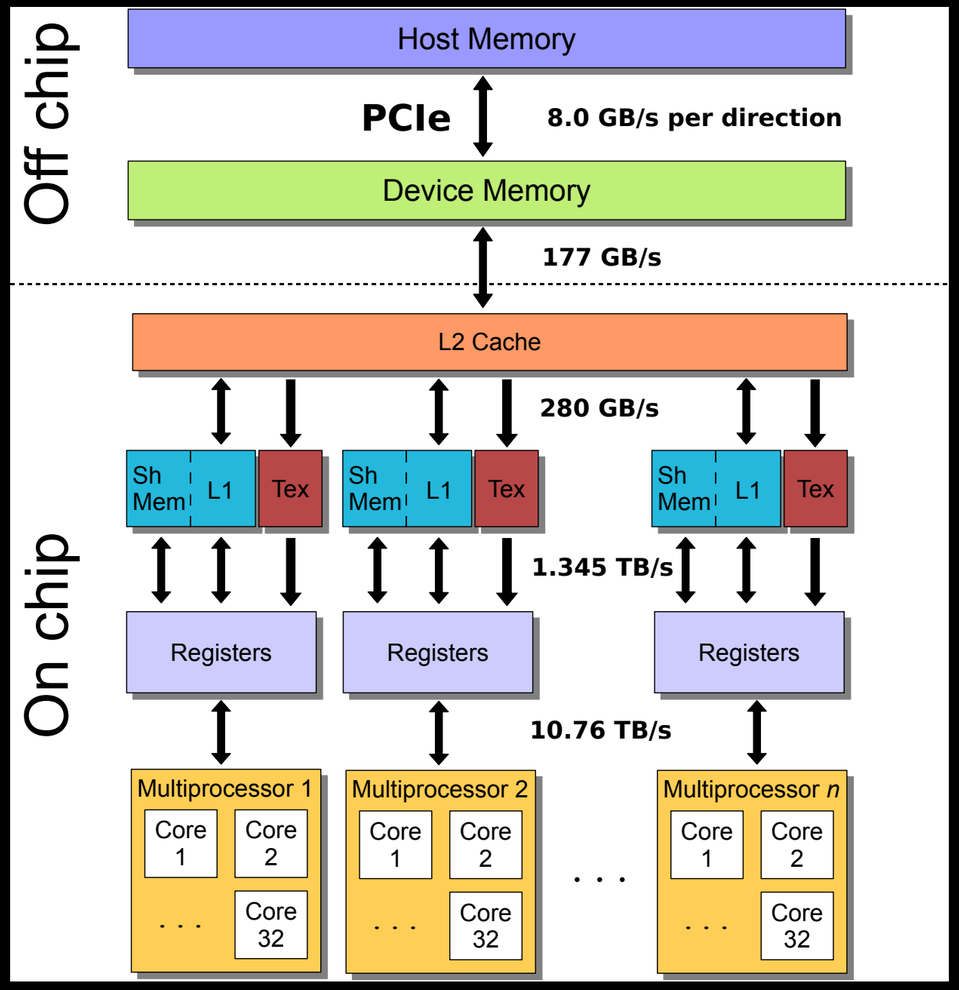
FNAL

4th May 2012

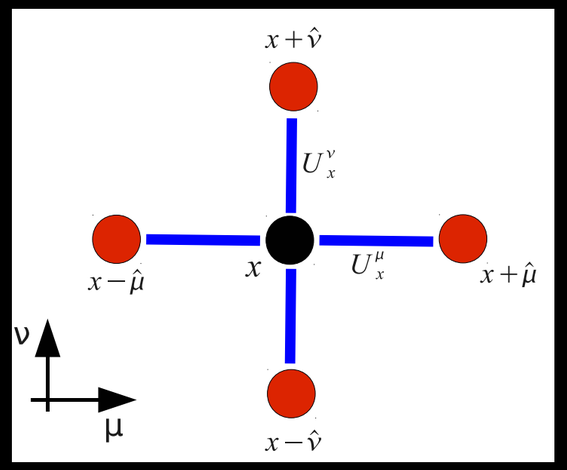
Outline

- Current Status of QUDA
- QUDA development roadmap
- Kepler preview

Anatomy of a GPU



- Deep memory hierarchy (bandwidth or latency)
- This is what the exascale will look like
- Efficient implementation must use exploit all locality
- Fortunately LQCD has lots of locality



QCD and NVIDIA

- Why does NVIDIA care about QCD?
  - Important HPC application
  - QUDA used to debug current toolkit and SDK
- QCD is one of the HPC benchmarks run on all future GPU simulators
  - Ron Babich working on NVIDIA Exascale GPU project
  - Ensures that future GPUs will always be great at QCD
  - Influencing design decisions (e.g., cache size)

QUDA

- QUDA is a library of solvers and performance critical kernels for lattice QCD on CUDA GPUs
- Highly optimized multi-GPU implementations of the Dirac operators
  - ‘Standard’ Krylov Solvers for QCD: CG(NE), BiCGStab
  - Domain-decomposition preconditioned solvers (additive and multiplicative)
- Now includes auxiliary kernels for HMC
- Key Optimizations
  - Spatial-cache blocking
  - Memory-traffic-reducing transformations
    - Mixed-Precision (16 bit, 32 bit, 64 bit) solvers
    - Field Compression
  - Aggressive Autotuning

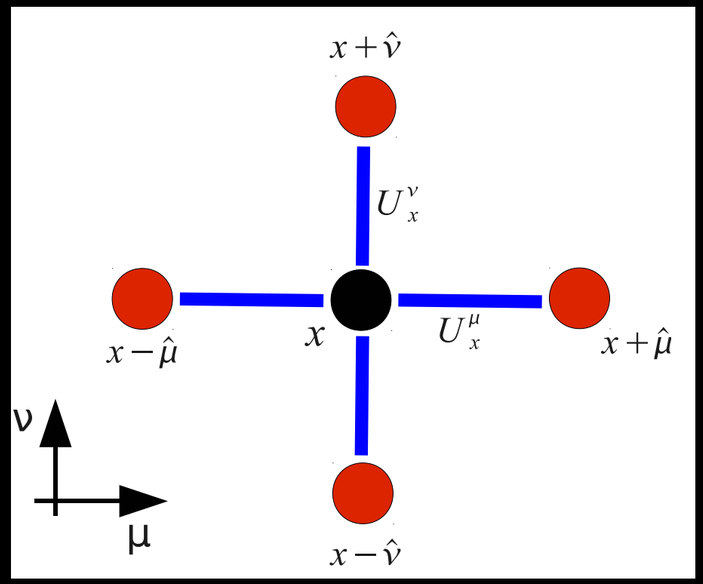
http://lattice.github.com/quda

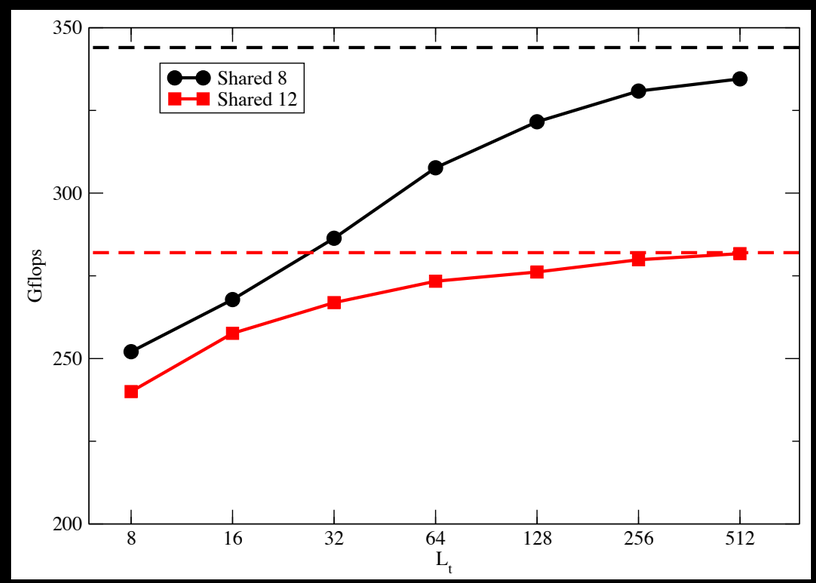
QUDA Consortium:

Ron Babich (NVIDIA), Rich Brower (BU), Mike Clark (NVIDIA), Justin Foley (Utah), Bálint Joó (Jefferson Lab), Guochun Shi (NCSA), Alexei Strelchenko (Cyprus), Kostya Petrov (Peta QCD), Joel Giedt (RPI), Steve Gottlieb (Indiana)

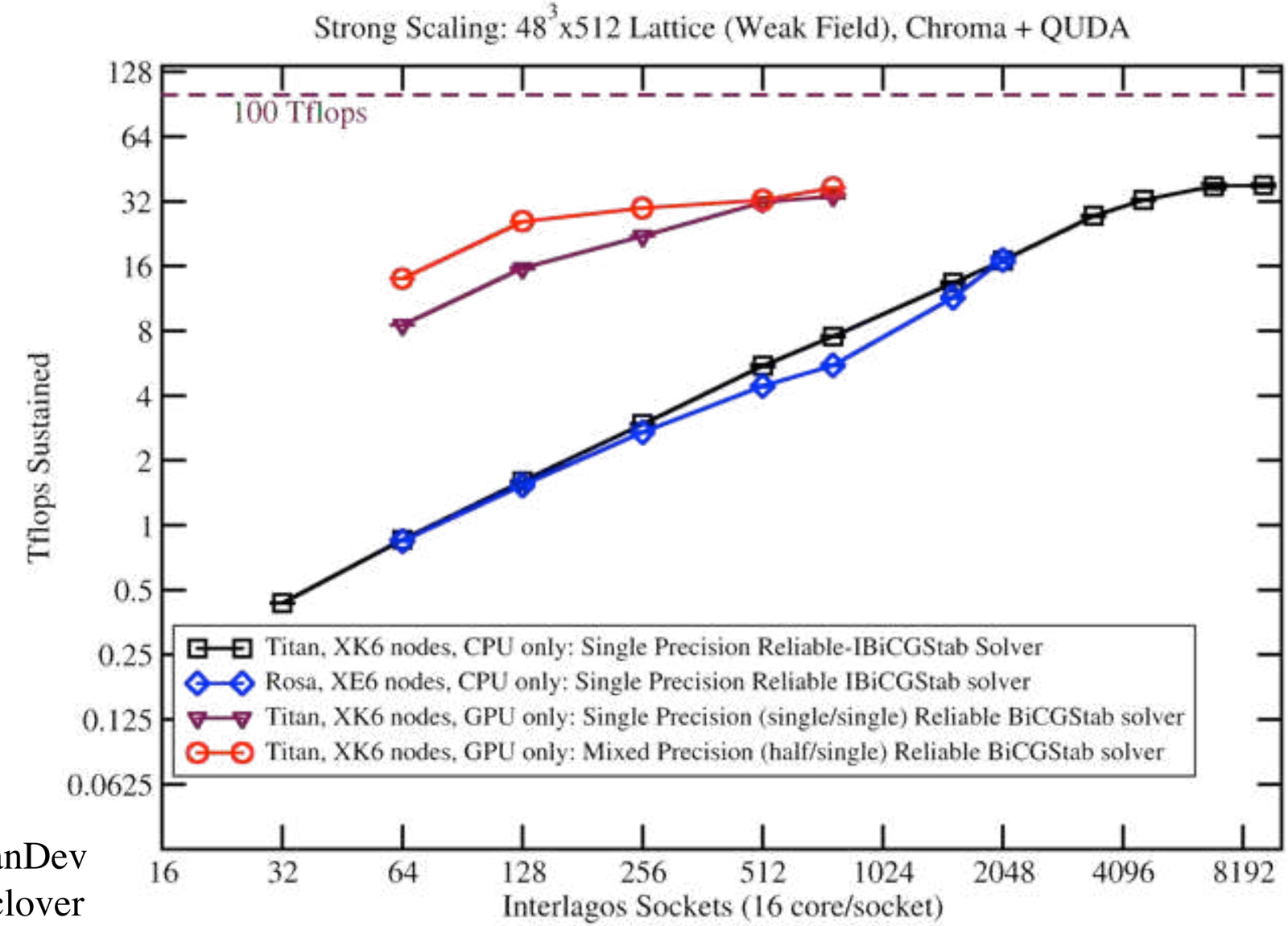
QUDA on Fermi

- Fermi is a two-year-old architecture so performance very well understood
- Fallacy that Fermi has too small a cache for QCD
  - Actually ideal balance between registers and shared memory for Wilson spatial cache blocking (SP)
  - Wilson spatial-cache blocking achieves 79% of an infinite cache (SP)
- Capacity solver performance:
  - Chroma Wilson-clover: 1 GPU ~ 100 cores
  - MILC Improved staggered: 1 GPU ~ 50 cores
- Issues
  - PCIe 2.0 and IB comms must be pipelined through host memory
  - Register-per-thread limitation causes large reduction in DP perf.



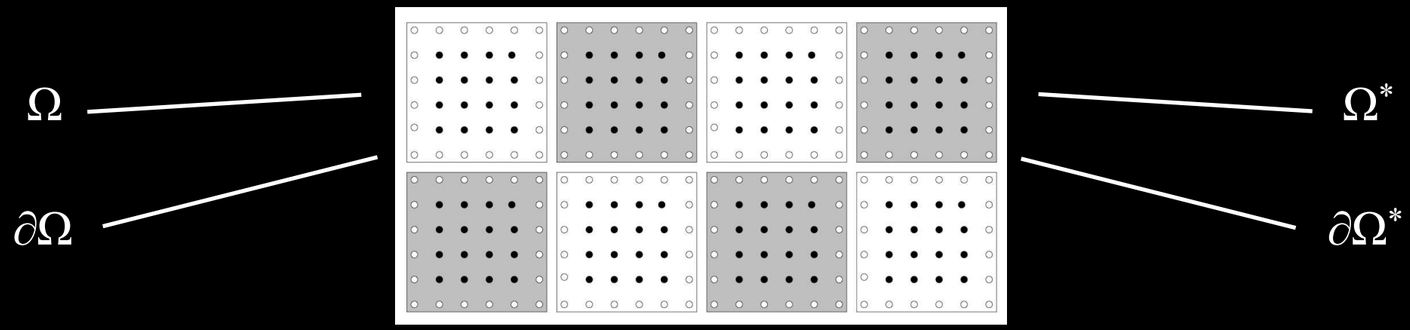


Benchmarks: + GPU (BiCGStab)



Results from TitanDev
- 48^3x512 aniso clover
- scaling up 768 GPUs

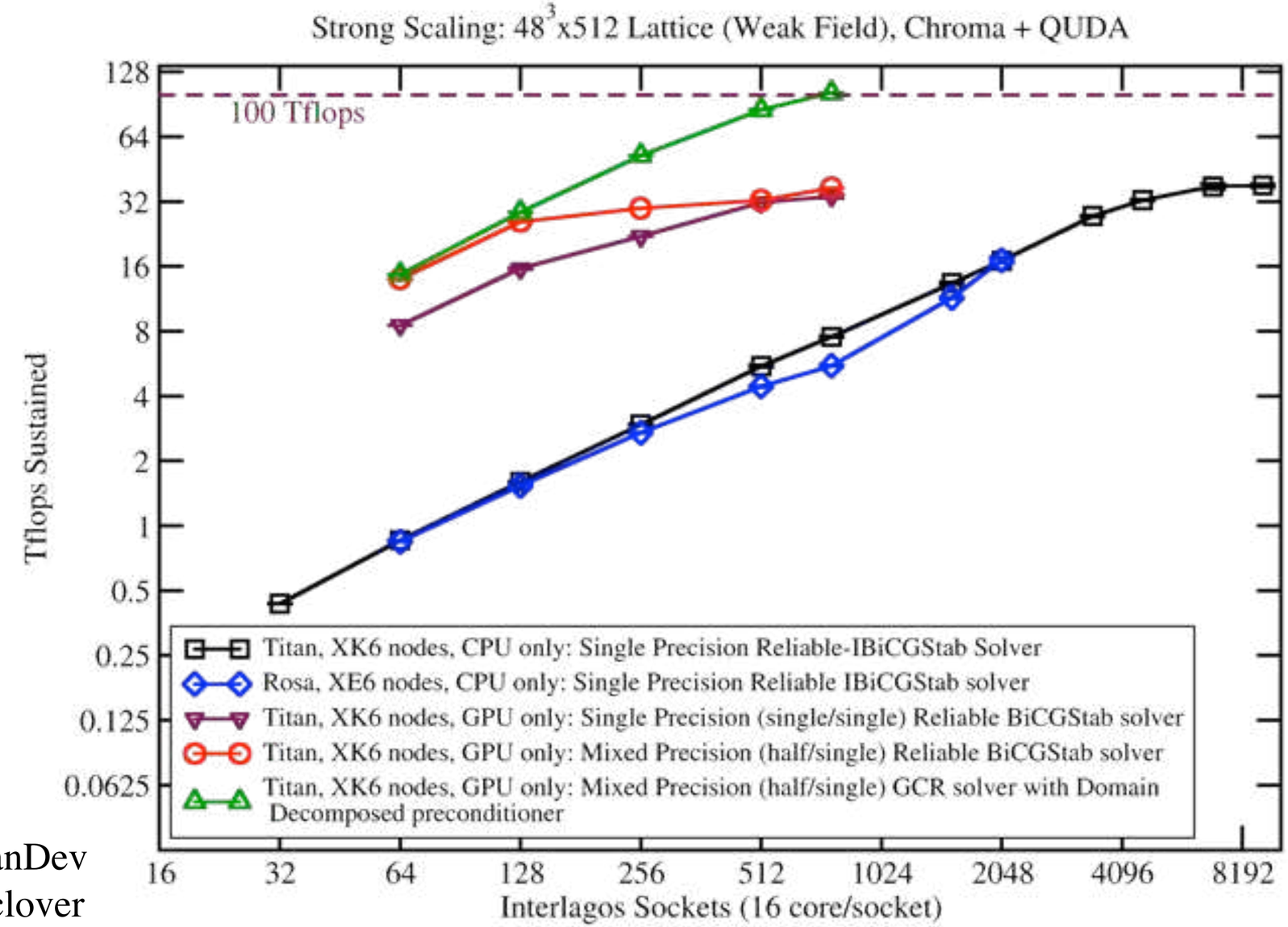
Domain Decomposition



- Schwarz Methods aka Domain Decomposition (Schwarz, 1870)
  - Partition the problem into domains
  - Impose Dirichlet BC and solve independently
  - Use union of domain solutions as a preconditioner
- Multiplicative Schwarz:
  - Update domains sequentially and use most recent solutions Block Gauss Seidel
  - Coloring algorithms used to parallelize $K = D_{\Omega}^{-1} + D_{\Omega^*}^{-1} + D_{\Omega^*}^{-1} D_{\partial\Omega^*} D_{\Omega}^{-1}$
- Additive Schwarz:
  - Update domains simultaneously Block Jacobi
  - Algorithm trivially parallel $K = D_{\Omega}^{-1} + D_{\Omega^*}^{-1}$

Figure by Luescher

Benchmarks: + GPU (DD+GCR)



Results from TitanDev
- 48^3x512 aniso clover
- scaling up 768 GPUs

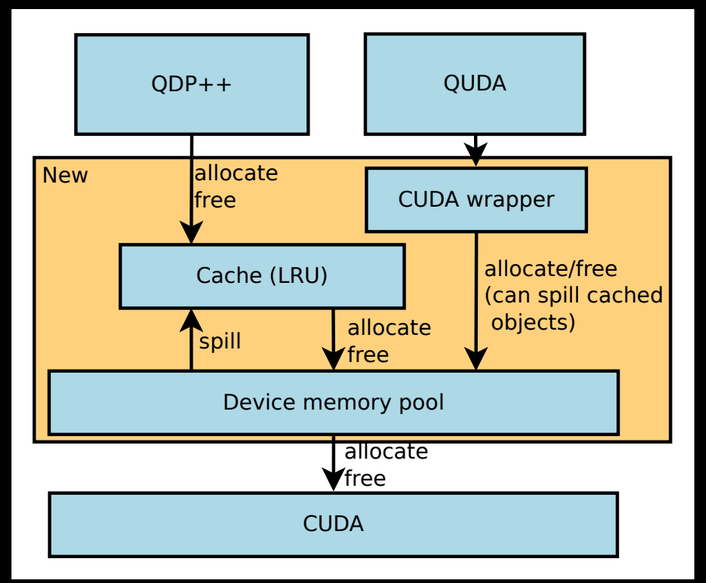
Bound by what?

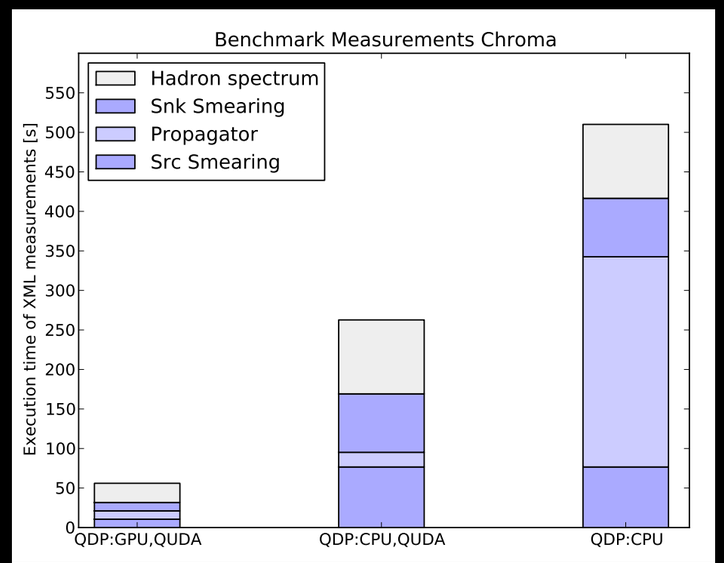
- Timing breakdown at 768 GPUs
- This tells us how we will scale
  - Doubling the GPU speed, same PCIe ⟹ 1.63x
  - Doubling the GPU and PCIe ⟹ 1.80x
- Increasingly latency bound due to orthogonalization
- Now need to think about about both comms and latency
  - Do more useful local work ⟹ overlapping blocks?
  - Do less reductions ⟹ modGMRES?

65%	kernel with no communication
15%	kernel with local communication
11%	orthogonalization (global reductions)
7%	solver restart (local and global)

QDP-JIT

F. Winter

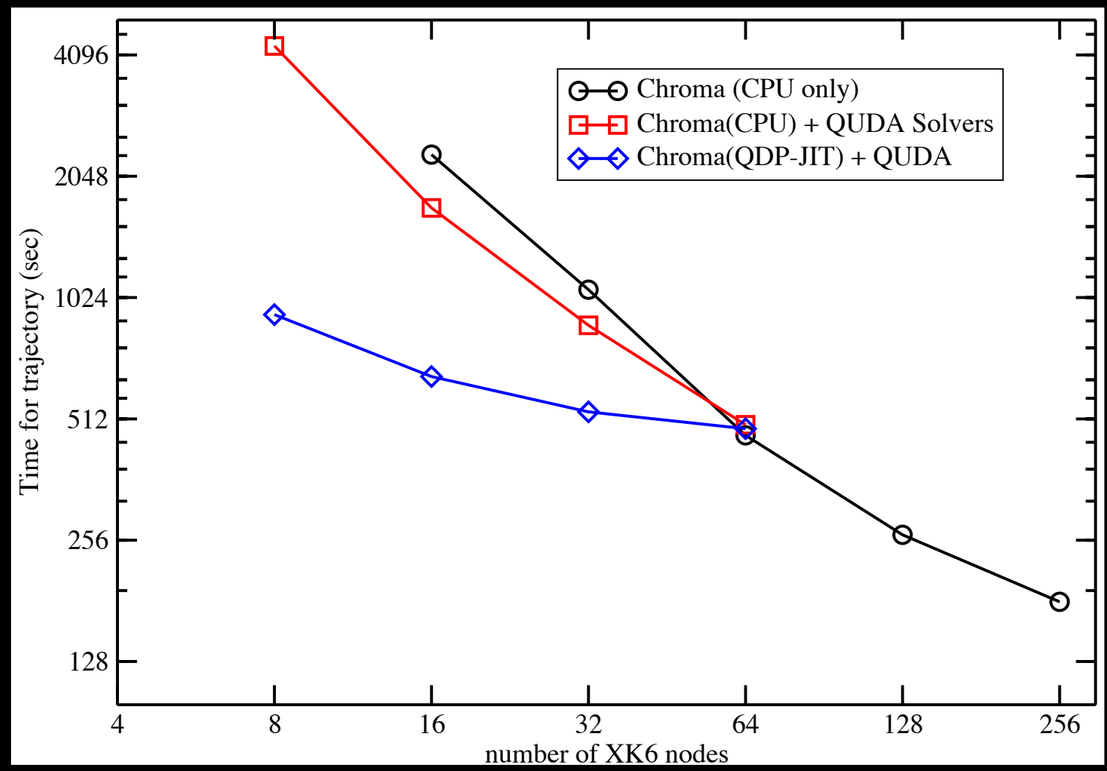




- What about the parts of Chroma that QUDA doesn't accelerate?
- QDP-JIT - port QDP++ to CUDA https://github.com/fwinter/qdp
  - Designed to interoperate with QUDA
  - QUDA accelerates time-critical routines
  - QDP-JIT accelerates everything else
- Uses JIT to create CUDA kernels at runtime
- Allows Chroma to run unaltered on (multiple) GPUs

Chroma HMC

- Hot-off-the-press benchmark
  - Thanks Balint and Frank
- Wilson HMC, $32^3 \times 96$
  - Simple CG solver
- Proof of concept
- Vast improvements possible
  - Solo Wilson Dslash done on CPU
  - GPUDirect
  - QUDA ⇔ QDP-JIT interop
  - DD solver



QUDA Clover Plans

- Implement missing clover HMC functionality
- Clover term
  - $F_{\mu\nu}$ kernel in place
  - Need clover computation, trace det and clover inverse
- Generalize fat-link computation for stout-link generation
- Allow QDP-JIT to directly share send/receive GPU buffers
- All ingredients then in place for full clover gauge generation
- Target is summer

HISQ Fermions

- QUDA 0.4.1 will include support for
  - Thanks to Justin and Guochun
  - Multi-GPU HISQ force, fat-link, gauge-force
- Almost everything in place for full RHMC
- Proof-of-concept V = 64^3 x 96 RHMD evolution
  - Run on 128 C2070 at FNAL
  - 1x C2070 = 24 CPU cores
  - GPUDirect issue?

current

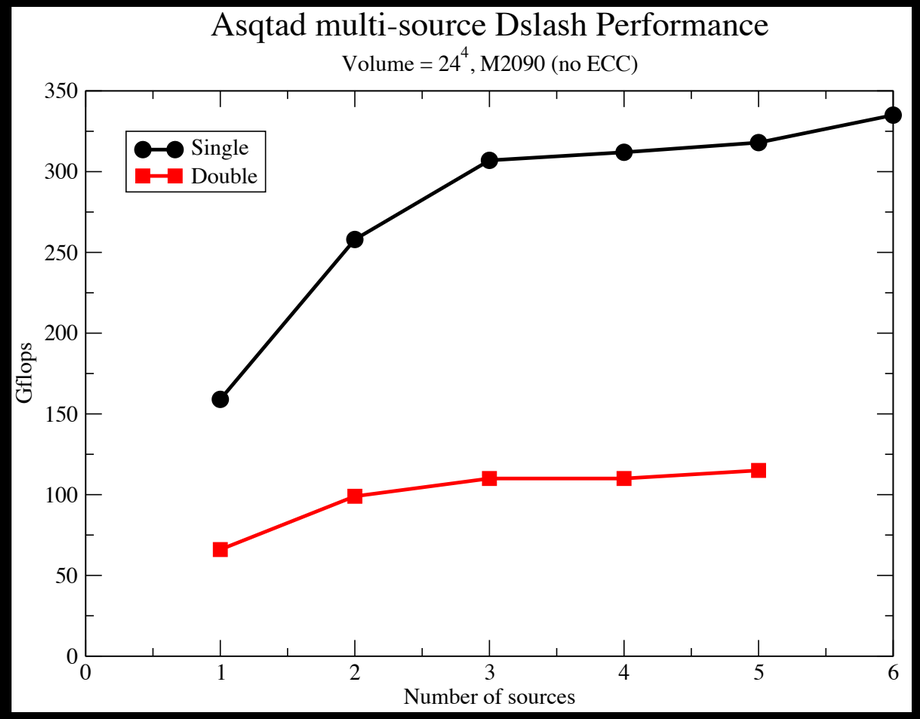
f	calls	time(s)	percentage
CG	6	149.8	2.2%
multi-CG	515	5832.46	62.23%
Fatlink	102	410.323	6.03%
GF	300	282.855	4.16%
FF	100	1162.96	13.8425%
Others		562.988	6.70113%
total_time		8.401388e+03	

fixed (3 weeks from now)

f	calls	times	percentage
multi-CG	515	3597.6	63%
Fatlink	102	410.323	7%
GF	300	282.855	5%
FF	100	814.07	14%
Others		562.988	11%
total_time		5.667e+03	

HISQ Fermions Plans

- Performance potholes
  - Missing kernels: outer products, long link
  - Fully GPU-resident pipeline
  - Fix naik-epsilon refinement issue
  - Adopt QDP/C-style HISQ force
- Staggered fermion has less spatial locality than Wilson
  - Performance is significantly worse



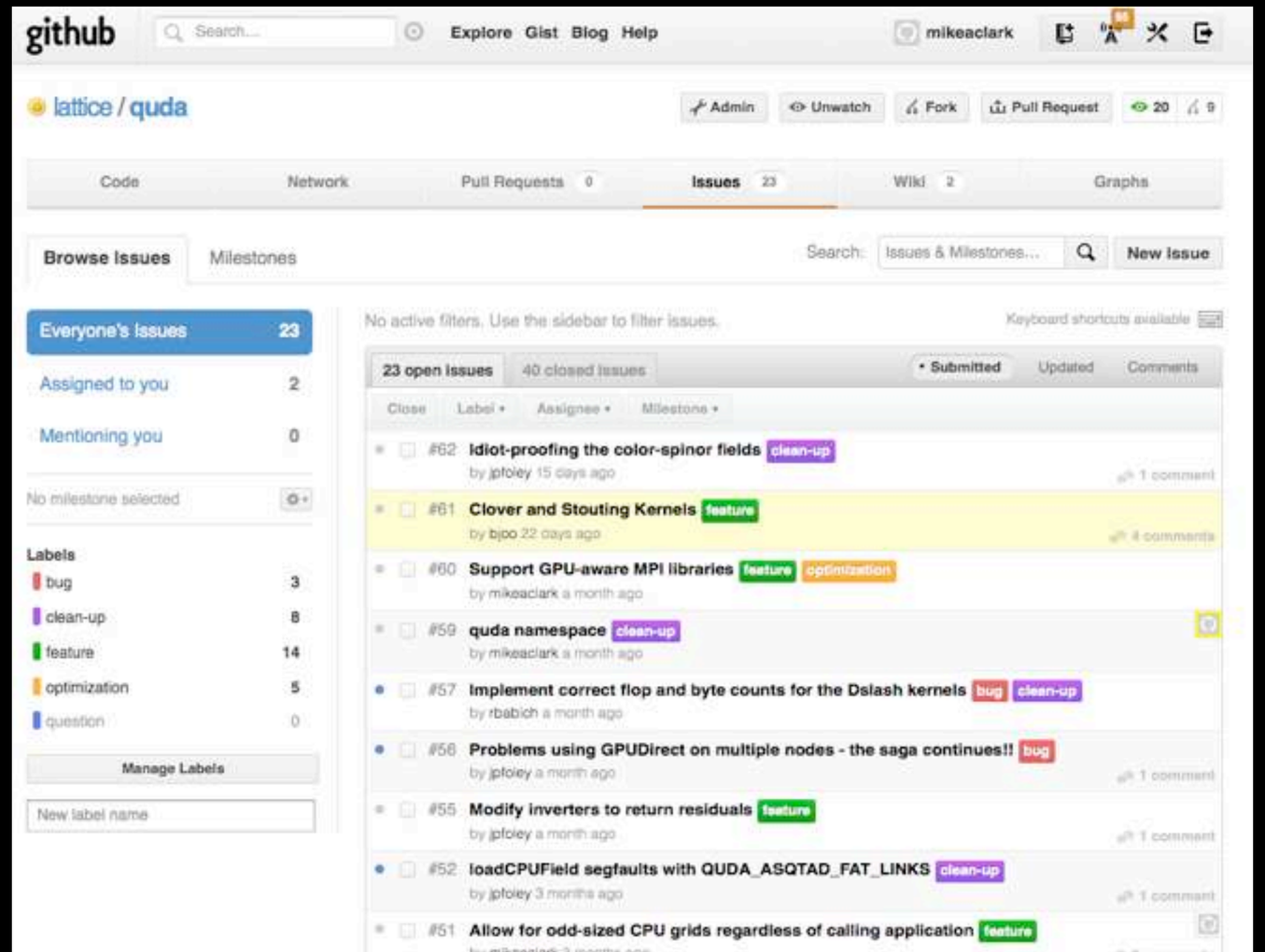
- Deploy multi-source solvers to vastly improve the spatial locality $\Longrightarrow$ more pseudofermions in RHMC
- Implement a domain-decomposition solver
- Use OpenACC for non-accelerated portions of MILC?

Domain-Wall Fermions

- QUDA 0.4.1 will include support for multi-GPU domain-wall fermions
  - Thanks to Joel Giedt, Alexei Strelchenko and Chris Schroeder
- Single GPU dslash (Fermi SP, not yet optimized): 188 Gflops
  - Add spatial-cache blocking (as Wilson): 246 Gflops
  - Gauge-field reuse (as staggered): 420 Gflops
- Not yet deployed (in the US, in production in Cyprus)
  - Need to get the CPS-QUDA interface into main trunk of CPS
  - Joel working on qcdlib interface for solver

QUDA plans

- Exploit GPU Direct 2.0
  - peer-2-peer communicaton within a node
  - improves intra-GPU bandwidth and latency
  - leaves PCIe bandwidth for IB
- Two summer projects
  - Adaptive multigrid
  - Kepler performance optimizations
- Please make requests
- Or even better - please join the fun



and now a word from my sponsors.....

GPU TECHNOLOGY CONFERENCE

May 14th-17th
San Jose, California

300 Sessions
120 Research Posters
100 Sponsors and Exhibitors
2 Co-located Events - LANL AHPC Symposium & InPar
1 Emerging Companies Summit

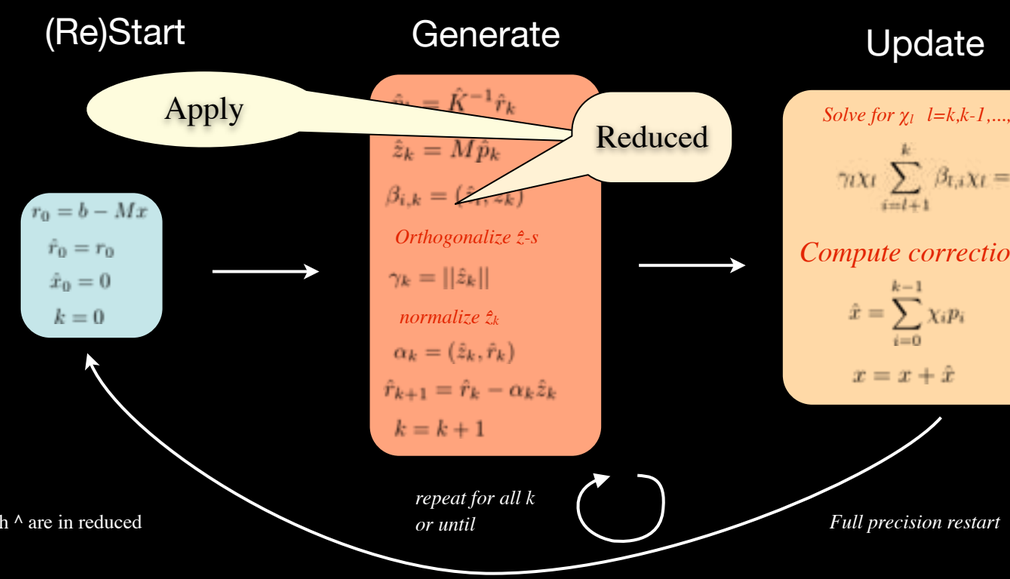
http://www.gputechconf.com

- Aon Benfield Securities
- BAE Systems
- Barcelona Supercomputing Center
- BGI (China)
- Boston University
- CERN (Switzerland)
- Chevron
- Chinese Academy of Sciences, Institute of Process Engineering
- CSIRO (Australia)
- Citrix
- Cray
- Dell
- DOE Joint Genome Institute
- eBay Research Labs
- Forschungszentrum Julich (Germany)
- GE Intelligent Platforms
- Georgia Tech Research Institute
- Harvard University
- HP
- ING Bank
- Inria
- Institute for Molecular Science (Japan)
- Irish Centre for High-End Computing
- Italian Air Force Meteorological Center
- Johannes Gutenberg University of Mainz
- Johns Hopkins University
- Karlsruhe Institute of Technology
- KISTI (Korea)
- Lawrence Berkeley National Laboratory
- Lawrence Livermore National Laboratory
- LEGO
- Lockheed Martin Solar & Astrophysics Laboratory
- Microsoft
- MIT Lincoln Laboratory
- MITRE Corporation
- Moscow Institute of Physics and Technology
- Nanyang Technological University
- NASA Langley Research Center
- National Tsing Hua University
- Naval Research Laboratory
- Oak Ridge National Laboratory
- Pacific Northwest National Laboratory
- Pittsburgh Supercomputing Center
- Pixar Animation Studios
- Princeton University
- Sandia National Laboratories
- Shanghai Jiao Tong University
- Siemens Corporate Research
- SLAC National Accelerator Laboratory
- Sony Electronics Inc.
- Stanford University
- Synopsys
- Tata Motors Limited
- Technicolor
- Texas A&M University
- The Boeing Company
- The University of Queensland
- Tokyo Institute of Technology
- Tsinghua University
- Unilever
- Universidade Federal Fluminense (Brazil)
- University College London
- University of Bonn
- University of Calgary, Department of Chemical & Petroleum Engineering
- University of California at Berkeley
- University of California, Davis
- University of Delaware
- University of Groningen
- University of Hamburg, Institute of Applied Physics and Microstructure Research Center
- University of Hong Kong
- University of Illinois at Urbana-Champaign
- University of Los Angeles
- University of Michigan
- University of Pennsylvania
- University of Texas at Austin
- VMware
- Wake Forest University
- Walt Disney Animation Studios

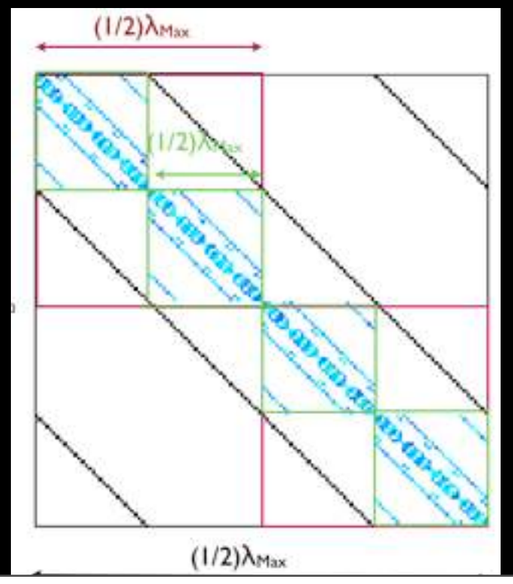
We’re hiring...

Domain Decomposition

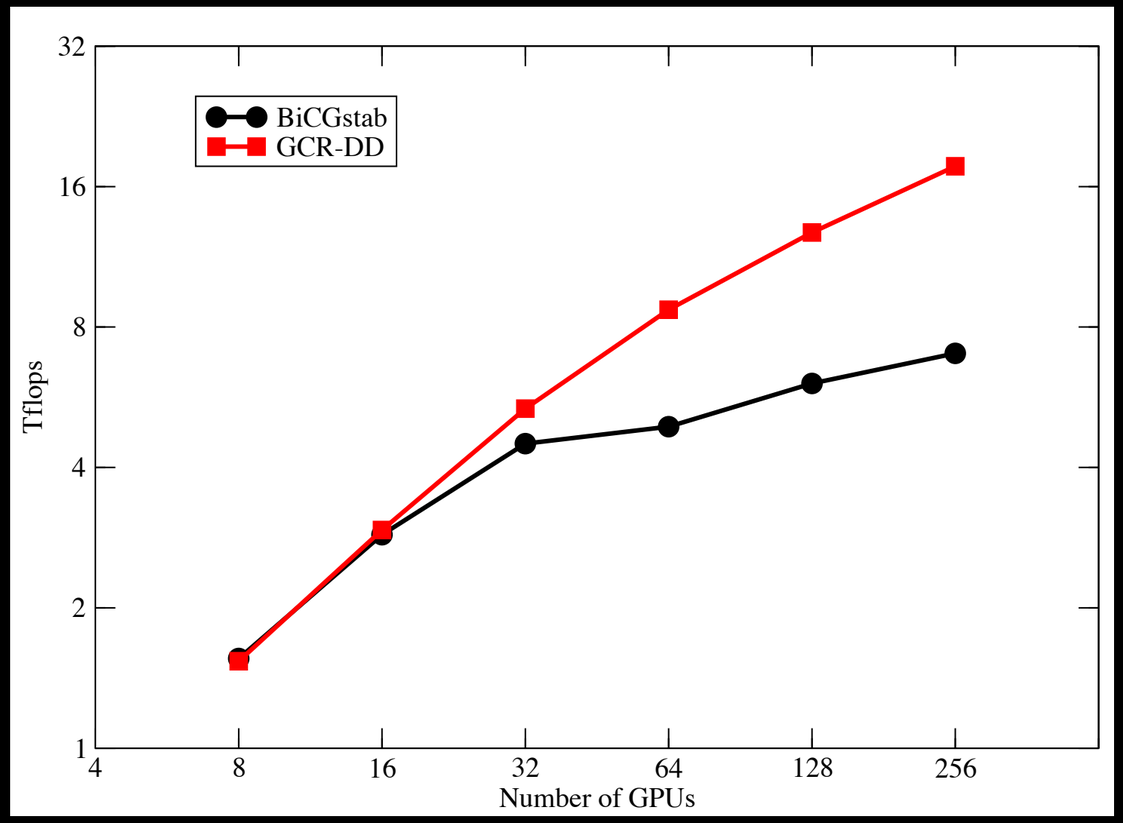
- Non-overlapping blocks - simply have to switch off inter-GPU communication
- Preconditioner is a gross approximation
  - Use an iterative solver to solve each domain system
  - Require only 10 iterations of domain solver $\Longrightarrow$ 16-bit
- Need to use a flexible solver $\Longrightarrow$ GCR
- Block-diagonal preconditoner impose λ cutoff
- Finer Blocks lose long-wavelength/low-energy modes
  - keep wavelengths of ~ $O(\Lambda_{QCD}^{-1})$, $\Lambda_{QCD}^{-1} \sim 1$fm
- Aniso clover: (a_s=0.125fm, a_t=0.035fm) $\Longrightarrow$ 8^3x32 blocks are ideal
  - 48^3x512 lattice: 8^3x32 blocks $\Longrightarrow$ 3456 GPUs

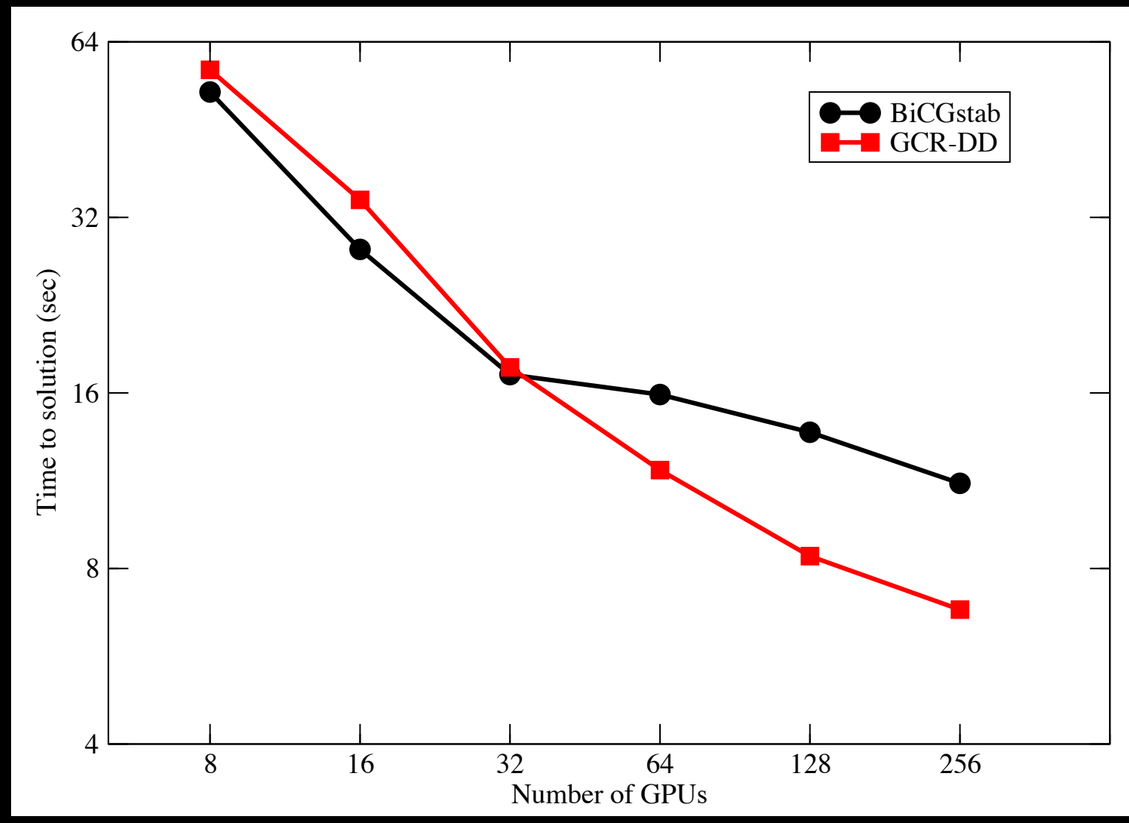


Quantities with ^ are in reduced



Flops versus speedup





Babich, Clark, Joo, Shi, Brower, Gottlieb, “Scaling Lattice QCD beyond 100 GPUs,” SC’11

Argonne NATIONAL LABORATORY

Near-term Software for SciDAC 3
ALCF, MILC/HISQ, QDP/C, Multigrid, FUEL edition

James C. Osborn

Argonne Leadership Computing Facility

USQCD All Hands Meeting

May 4-5, 2012

FNAL

U.S. DEPARTMENT OF ENERGY

ALCF BG/Q “Mira” Racks

T&D racks “Cetus” and “Vesta”


IBM
Blue Gene Q supercomputer

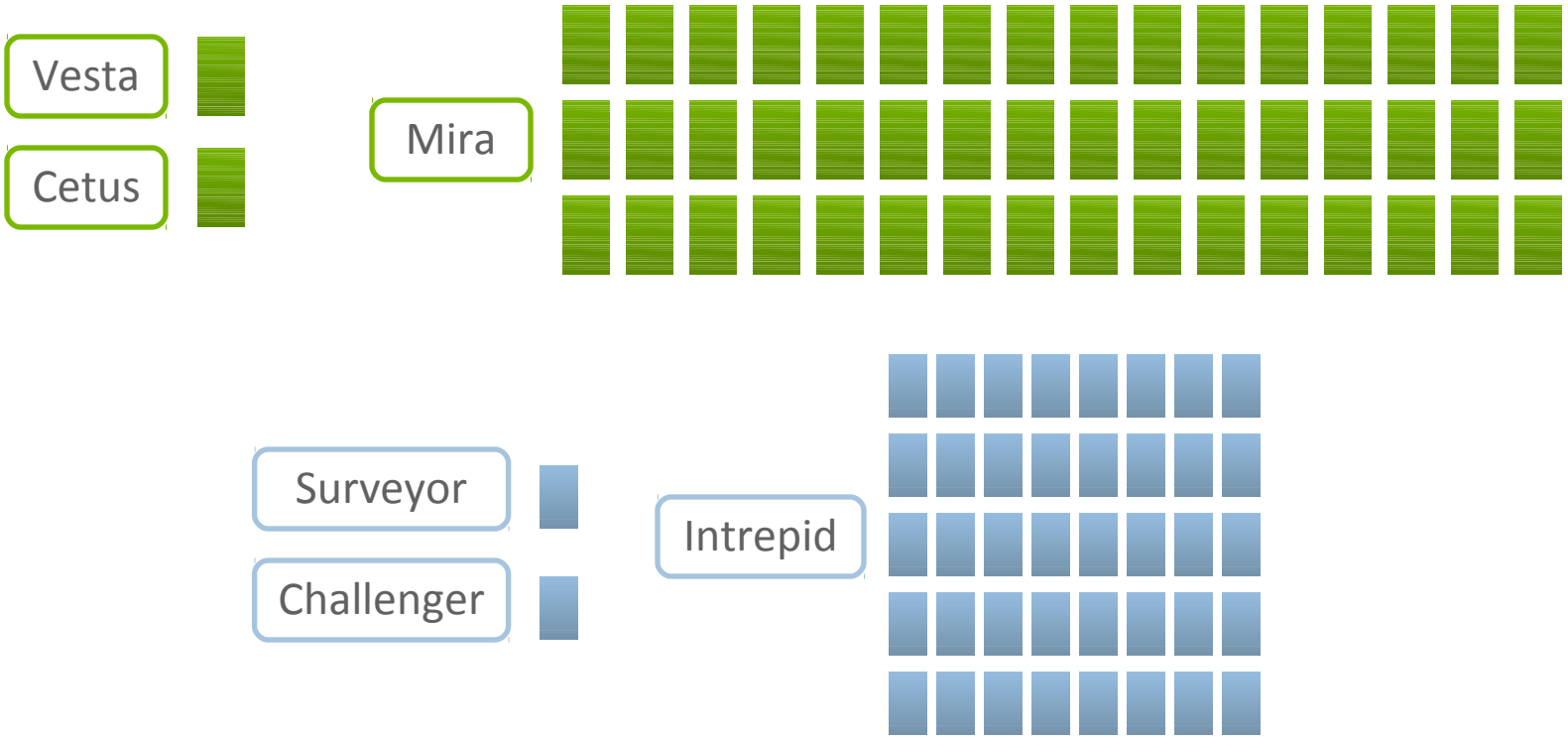
ALCF BG/Q Hardware

First two racks delivered 1/14/2012

Test & Development (T&D) machines

One rack: Very Early Access System (VEAS)

Sixteen *Mira* racks delivered



Evolving timeline for Mira availability

ESP access to Mira likely soon

- **ESP projects *may* have access to Mira as early as summer 2012**
- **Time for Early Science runs is likely to be earlier than expected**
  - Possibly second half of CY2012
- **Get on the T&D *now***
  - ESP is currently a top priority but there are many pressures on the hardware
  - Important for INCITE proposal preparation, especially computational readiness on BG/Q

Plan for a 2013 INCITE proposal on Mira

- **Mira is committed to go live for INCITE on October 1, 2013 with 768M core-hours for allocation**
- **Start date of production INCITE time is likely to happen earlier in CY 2013**
- **Guidance for INCITE 2013**
  - Propose science based on a 3B core-hour pool, 100-300M per project
  - 2013 INCITE Allocation scenarios

768M

2B core-hours

3B core-hours

HISQ on Q

- MILC + SciDAC QOPQDP and supporting libraries
  - QOPQDP provides optimized HISQ solver, fermion and gauge forces
- Need to use 32-64 threads per node
- Threading strategies:
  - "MPI everywhere": multiple MPI ranks per node, no threading
    - no changes to MILC
    - may not be as efficient
  - OpenMP (parallel loops)
    - Exists in QLA, and also some parts of MILC
    - Noticeable overhead for fine-grained parallelism
  - Application level threads
    - Whole application (or major routines) are fully threaded
    - Threads work on independent parts of loops as they are encountered
    - Needs occasional synchronization
    - Exists in development version of QDP/C, works, needs further testing and cleanup
    - Haven't tested it in QOPQDP yet

Initial tests and optimization for BG/Q

- Focus on MPI-everywhere to start,
  transition to fully threaded code once up and running with reasonable efficiency

- Initial optimizations
  - QLA: add optimized QPX code for key routines
    - IBM provided inline asm versions of a few QLA routines a while back
    - Have rewritten some in xlc intrinsics to aid modification and maintainability
    - Staggered matrix-vector product gets up to 30% peak

  - QMP: use low level (SPI) communications instead of MPI (with Heechang Na)
    - Avoids MPI overheads, reduces latency by up to 8x for small messages (1.5-3x typical)
    - Have working version of QMP using SPI
    - SPI use not automatic, user must declare send/recv pair to use SPI instead of MPI
    - Added synchronization optimization calls to QMP
      - QMP_clear_to_send(message, FLAG)

      FLAG = { QMP_CTS_DISABLED, QMP_CTS_NOT_READY, QMP_CTS_READY }

HISQ CG solver performance

128 nodes using 32^3x64 lattice (8x8x16x16 per node)

Plain MILC or QOPQDP compiled with xlc -O3 MPI, simple mapping	~ 10 Gflops/node (5% peak)
As above, with optimized mapping (64 threads = 2x2x4x4 block)	~ 11 Gflops/node (5.3% peak)
As above, compiled with -O5 (looks correct, but not thoroughly checked)	~ 12 Gflops/node (5.9% peak)
QOPQDP + QPX code in QLA optimized mapping (c32 mode)	~ 16 Gflops/node (7.6% peak)
As above + QMP-SPI	~ 20 Gflops/node (9.7% peak)

Some overheads remain: MPI global sums, QDP communications bookkeeping

MPI-everywhere upper limit probably around 12-15%

Now time probably better spent working on fully threaded code

Wilson clover

128 nodes using 32^3x64 lattice (8x8x16x16 per node)

Plain QOPQDP Compiled with xlc -O3 QMP-MPI, optimal mapping	~ 12 Gflops/node (5.9% peak)
with QMP-SPI	~ 15 Gflops/node (7.1% peak)

QPX code not available yet

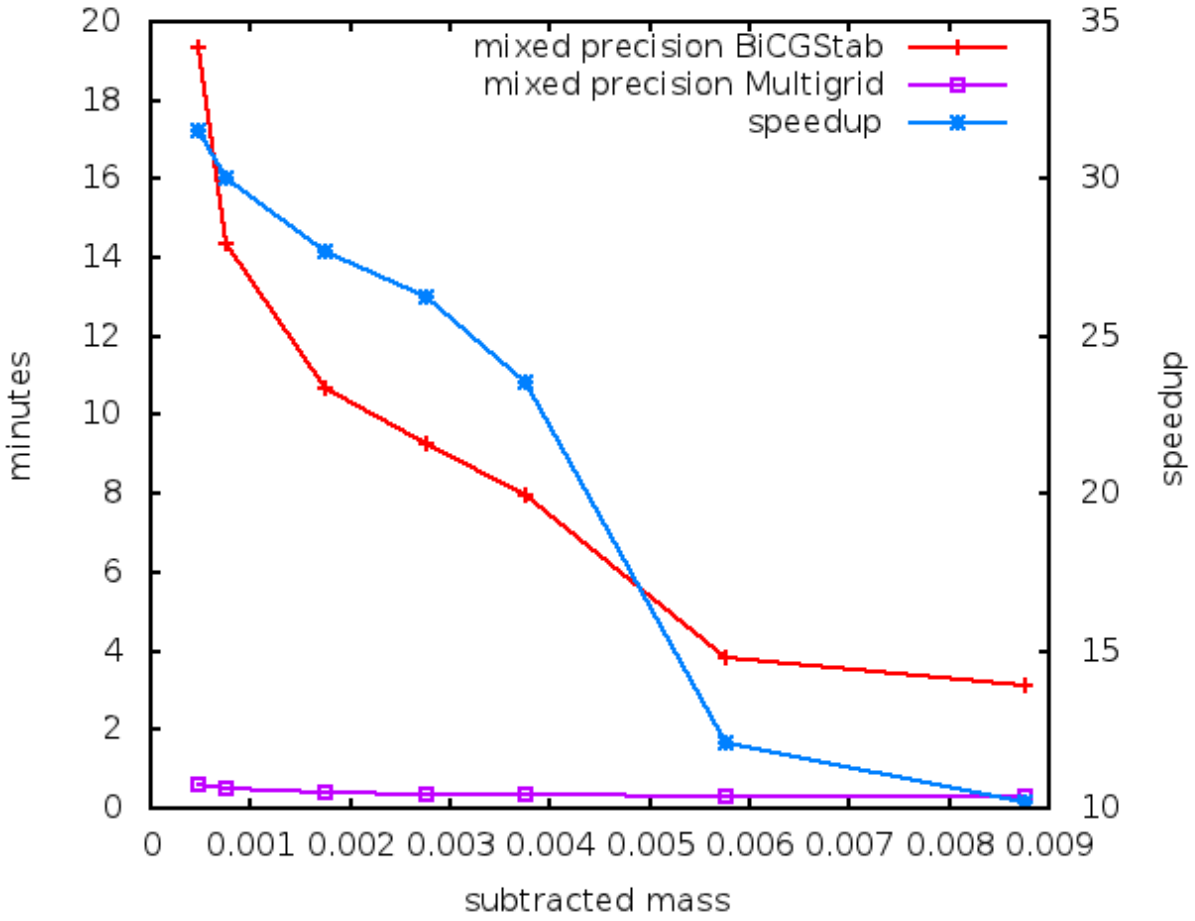
Wilson clover multigrid (3 level)

128^3x96 quenched lattice (Karsch+collabs.)

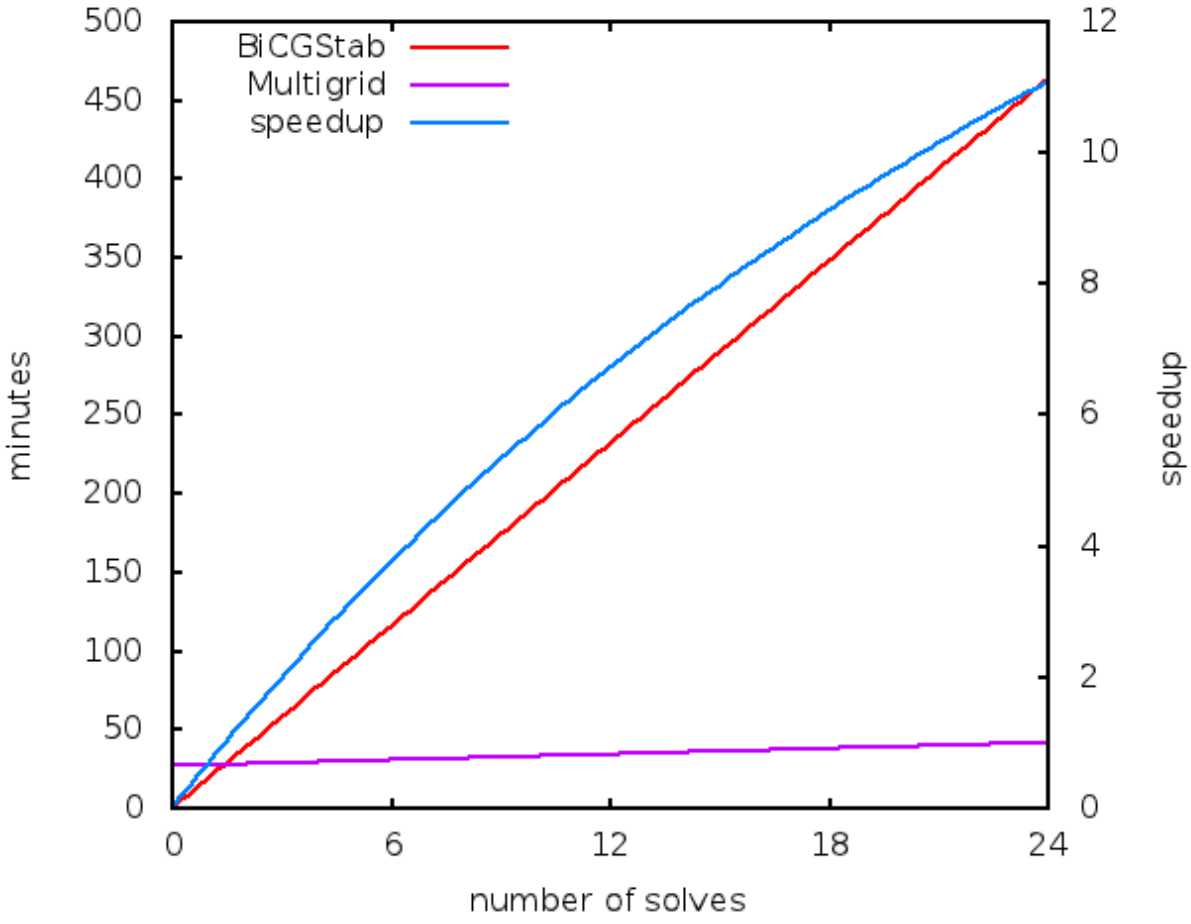
512 BG/Q nodes (64 ranks/node), plain C, xlc -O3, QMP-MPI

BiCGStab total speed
~ 5.8 Tflops
(5.6% peak)

Multigrid effective rate
~ 184 Tflops
(176% peak)



Wilson clover multigrid
total time for fixed number of solves



New FUEL HMC framework (Framework for Unified Evolution of Lattices)

- High level layer focused on gauge configuration generation
  - motivation is to have flexible HMC framework to support wide range of beyond standard model theories
  - algorithmic abstraction: generation algorithm independent of gauge group, action, etc.
  - easy to write new high-level algorithms, tune parameters
  - serves as wrapper for efficient “level 3” routines
  - easy to plug in new routines
  - new routines can be written in any other language/framework

- Uses scripting language Lua
  - Small
  - Easy to port (ANSI C89)
  - Easy to use, yet powerful
  - Easy to embed and interface with libraries

Initial prototype

- Initially using SciDAC QDP/C and QOPQDP libraries to provide needed routines
- Supports SU(3) lattice generation with HISQ
  - using RHMC (Rational Hybrid Monte Carlo) algorithm
  - also supports HMC with mass preconditioning
- Matches conventions of current MILC code (and can parse same input files)
- Being used by Lattice BSM collaboration for 8f HISQ (G. Fleming's talk)

Current plans

- Use as testbed for algorithmic research
  - test HMC algorithms for large N_f
  - improved integrators (e.g. force gradient)
  - plug in improved solvers (e.g. multigrid)

- Add Wilson and domain wall quark support
- Add QUDA (GPU) routines as alternatives
- Add SU(2) support
- Plus many other actions mentioned in HEP SciDAC 3 proposal