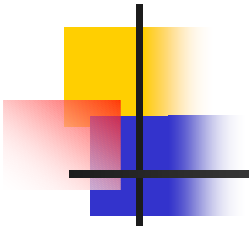
Performance of Lattice QCD Codes

Don Holmgren
Fermilab
djholm@fnal.gov
HEPiX Fall 2007, Nov 8, 2007



Outline

- Context
- LQCD Machines
  - Flops and bytes
  - Constraints
- Performance
  - Memory bandwidth
  - Single node applications performance
  - NUMA issues
  - Cluster performance
- Conclusions and speculation

Context: QCD and Lattice QCD

- Quantum ChromoDynamics is the theory of the strong force
  - The strong force describes the binding of quarks by gluons to make particles such as neutrons and protons
  - The QCD action, which expresses the strong interaction between quarks mediated by gluons:

$$S_{Dirac} = \overline{\psi}(\not{D} + m)\psi$$

where the Dirac operator ("*dslash*") is given by

$$\not{D}\psi = \sum_{\mu} \gamma_{\mu}(\partial_{\mu} + igA_{\mu}(x))\psi(x)$$

Context: QCD and Lattice QCD

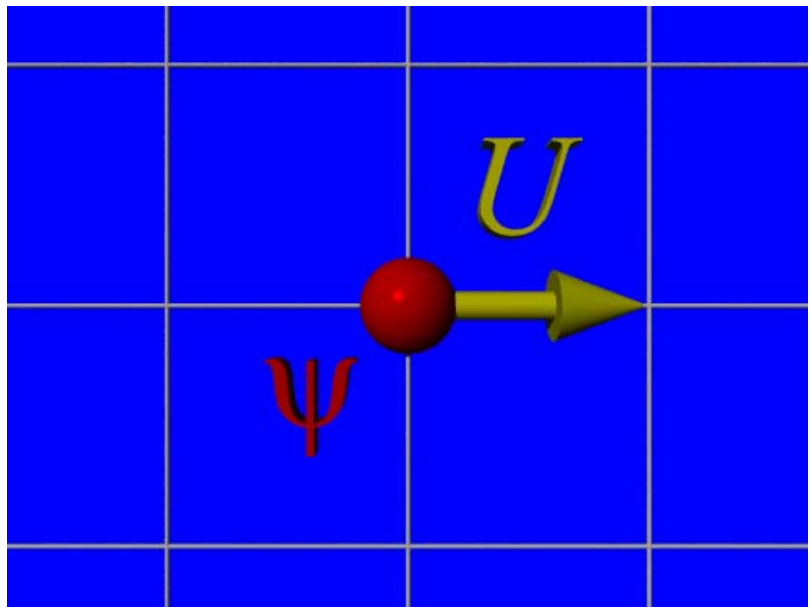
- Lattice QCD is the numerical simulation of QCD
  - Lattice QCD uses discretized space and time
  - A very simple discretized form of the Dirac operator is

$$\not{D}\psi(x) = \frac{1}{2a}\sum_{\mu}\gamma_{\mu}[U_{\mu}(x)\psi(x+a\hat{\mu}) - U_{\mu}^{\dagger}(x-a\hat{\mu})\psi(x-a\hat{\mu})]$$

where a is the lattice spacing

Context: QCD and Lattice QCD

- A quark, $\psi(x)$, depends upon $\psi(x + a\mu)$ and the local gluon fields U_μ
  - $\psi(x)$ is complex 3x1 vector, and the U_μ are complex 3x3 matrices. Interactions are computed via matrix algebra
  - On a parallel computer, the space-time lattice is distributed across all of the nodes



Context: Scale

- LQCD uses Monte Carlo to estimate observables, such as particle masses and decay constants
  - This is done by “tying together” valence quark propagators that are simulated in snapshots of the QCD vacuum known as *vacuum gauge configurations*
  - Gauge configurations are generated in a single Markov chain – these calculations are done on 10 TFlops and larger peak machines and require many months per ensemble
  - Propagator generation can be done on smaller machines
    - Embarrassingly parallel, similar to event reconstruction
    - About 50% of Flops are spent on gauge configuration generation, and about 50% on propagator generation and analysis

Context: The National Program

- Gauge configuration has typically been done at the various DOE and NSF computing facilities (NERSC, Oak Ridge, NCSA, PSC, SDSC, etc.) and on custom hardware (QCDSP, QCDOC, ACPMAPS)
- A 4-year DOE major IT project for FY06-FY09 operates the QCDOC at BNL, and builds and operates clusters at FNAL and JLab
  - Funded by OHEP (80%) and ONP (20%)
  - As of today, in terms of sustained LQCD TFlops, this program operates 4.2 TF at BNL, 3.6 TF at FNAL, and 4.1 TF at Jlab, with 4.2 TF additional planned at FNAL in late calendar 2008
- Also, DOE has funded, via SciDAC, LQCD software infrastructure development (libraries, machine specific optimizations)

LQCD Machines: Flops and Bytes

- Fundamental LQCD math kernel:
  - Most flops occur in multiplications of SU(3) Vectors (complex 3X1) by SU(3) Matrices (complex 3X3)
  - In single precision: 66 Flops, 96 bytes read, 24 bytes written
    - This 2:1 bytes:flops ratio (4:1 for double precision) stresses memory
  - To invert the *dslash* operator a preconditioned conjugate gradient iterative solve is used
  - The vectors live on the 4-D lattice sites, and the matrices live on the links connecting the sites
  - On each sweep of the lattice, each site is updated with the results of multiplying neighbor vectors by links in each of the eight directions
  - On a parallel machine, message passing (MPI) is used to gather vectors on the surface of the 4-D sublattices or neighboring nodes

LQCD Machines: Constraints

- Either memory bandwidth, floating point performance, or network performance (bandwidth at message sizes used) will be the limit on performance on a given parallel machine
- On current single nodes, using lattices sizes of interest, memory bandwidth is the constraint
- On current parallel computers, the constraint is either memory bandwidth or network performance, depending on how many nodes are used
  - Network performance limits scaling:
    Surface area to volume ratio increases as more nodes are used, causing relatively more communications

LQCD Machines: Constraints

- We buy machines with the best LQCD price/performance
- This means:
  - Machines with the best memory bandwidth
  - Machines with modest memory size (0.5 GB/core)
  - High performance interconnects (Infiniband, Myrinet, Quadrics, Gigabit Ethernet meshes)
- 5+ years ago interconnect costs were 50% of total machine costs
  - Infiniband has commoditized HPC interconnects
  - Interconnect costs are 30% of total (and dropping)
  - Interconnect can typically be re-used after a cluster is retired

Performance: Memory Bandwidth

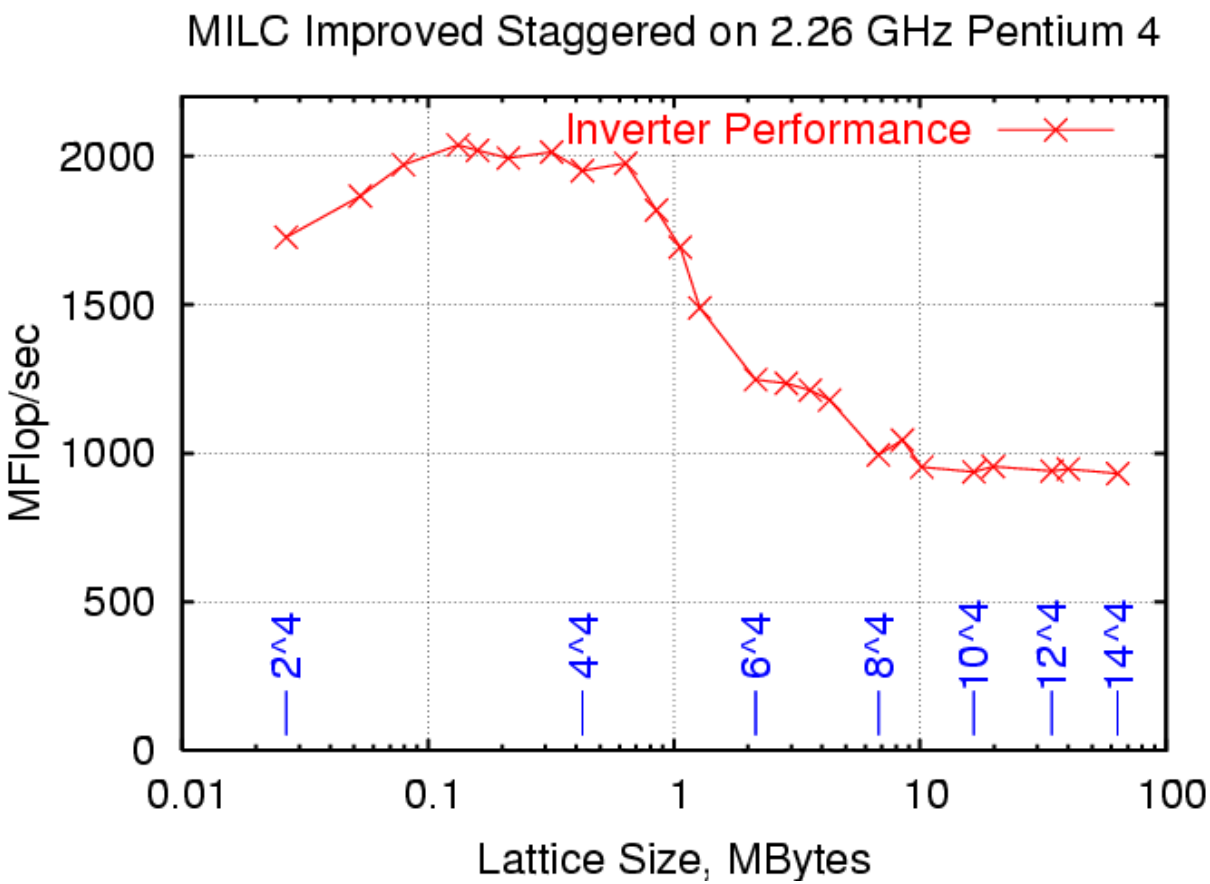
- How we measure:
  - McCalpin's STREAMS "Copy" benchmark correlates well with LQCD application performance
  - On multi-core machines, we use an OpenMP version of STREAMS to thread the inner loops and measure aggregate performance across all threads
  - Very important: aggressive optimizations improve STREAMS numbers but are not relevant to LQCD code
    - Biggest performance gain comes from writing around cache (non-temporal writes)
    - Unfortunately these writes require memory alignments that are not compatible with the sizes of SU(3) data structures

Performance: STREAMS

CPU	Machine	Memory Type	Single Core	All cores (#)
2.93 GHz Pentium dual core	Single socket	DDR2 533	3104 MB/sec	3085 MB/sec (2 cores)
2.66 GHz Xeon dual core	Dual socket	FB-DIMM 1333	2712	5043 (4 cores)
2.4GHz Xeon Quad	Quad Socket	FB-DIMM 1066	2732	8194 (16 cores)
2.0 GHz AMD Dual	Dual Socket	DDR 667	2368	5426 (4 cores)
2.6 GHz AMD Dual	Dual Socket	DDR2 667	2590	6693 (4 cores)
1.9 GHz AMD Quad	Dual Socket	DDR2 667	2725	8123 (8) cores
1.9 GHz AMD Quad	Dual Socket	DDR2 667	3236	10667 (8 cores)
2.1 GHz AMD Quad	Dual Socket	DDR2 667	3900	13056 (8 cores)

- All Intel cpus shown are “Core” microarchitecture
- AMD Quad (Barcelona): split-power plane (clock memory and processor separately) significantly increases bandwidth (indicated in blue)

Generic Single Node Performance



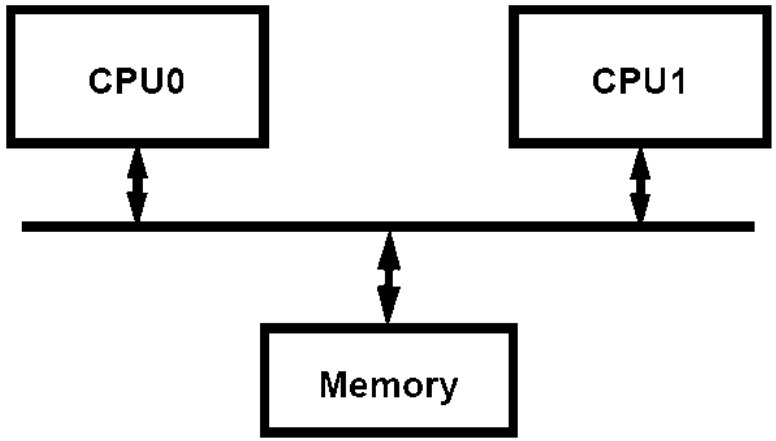
- Graph shows performance of the conjugate gradient Dirac operator (*dslash*) inverter
- Cache size = 512 KB
- Floating point capabilities of the CPU limits in-cache performance
- Memory bus limits performance out-of-cache
- We care about 12^4 and larger lattices
  - 48^3 X 144 gauge configurations are currently being produced for analysis

Performance: Single Node LQCD

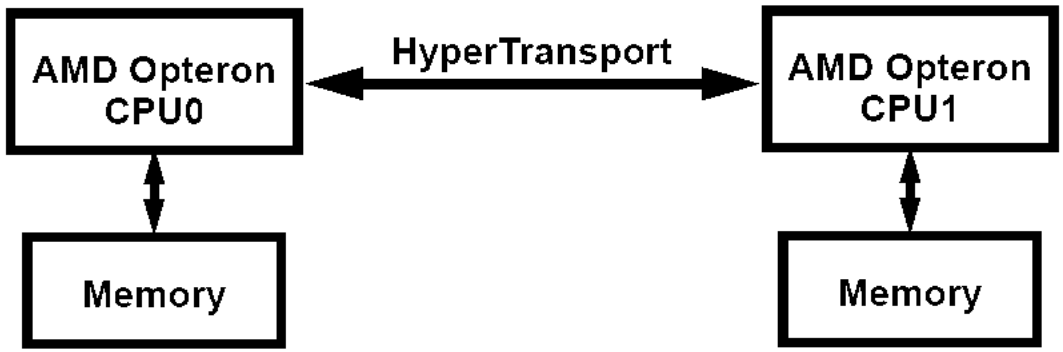
CPU	Machine	Memory Type	Single Core	All cores (#)
2.93 GHz Pentium dual	Single socket	DDR2 533	3367 MFlops	3637 MFlops (2 cores)
2.66 GHz Xeon dual core	Dual socket	FB-DIMM 1333	2363	4745 (4 cores)
2.4GHz Xeon Quad	Quad Socket	FB-DIMM 1066	1800	6872 (16 cores)
2.0 GHz AMD Dual	Dual Socket	DDR 667	1387	4415 (4 cores)
2.6 GHz AMD Dual	Dual Socket	DDR2 667	1650	4807 (4 cores)
1.9 GHz Barcelona	Dual Socket	DDR2 667	1624	5556 (8) cores
1.9 GHz Barcelona	Dual Socket	DDR2 667	1792	7490 (8 cores)
2.1 GHz Barcelona	Dual Socket	DDR2 667	1965	8291 (8 cores)

- Performance of MILC *dslash* inverter using fixed 14^4 lattice (strong scaling)
- Shared memory MPICH used for message passing

Memory Architectures

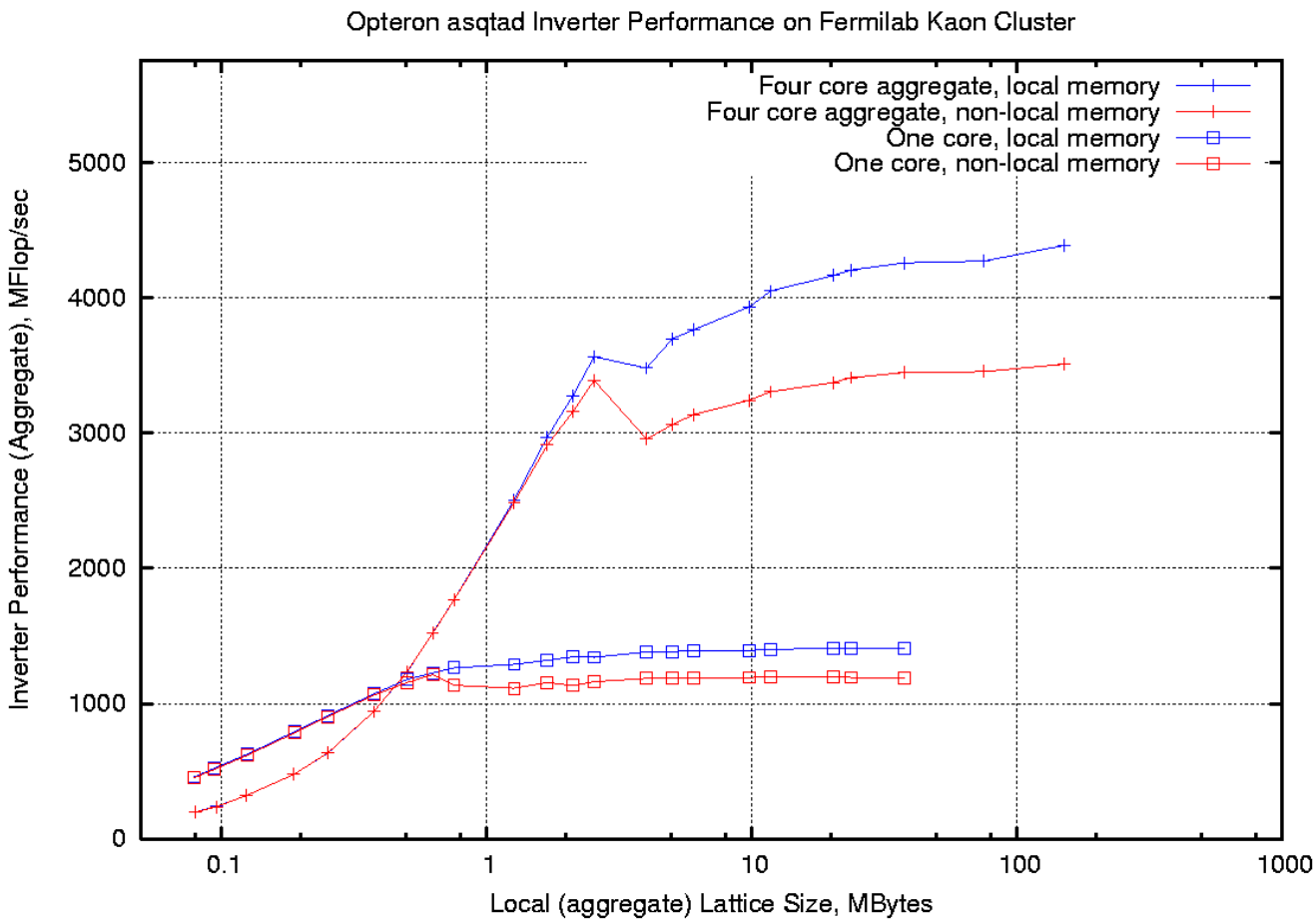


Intel Xeon SMP Architecture



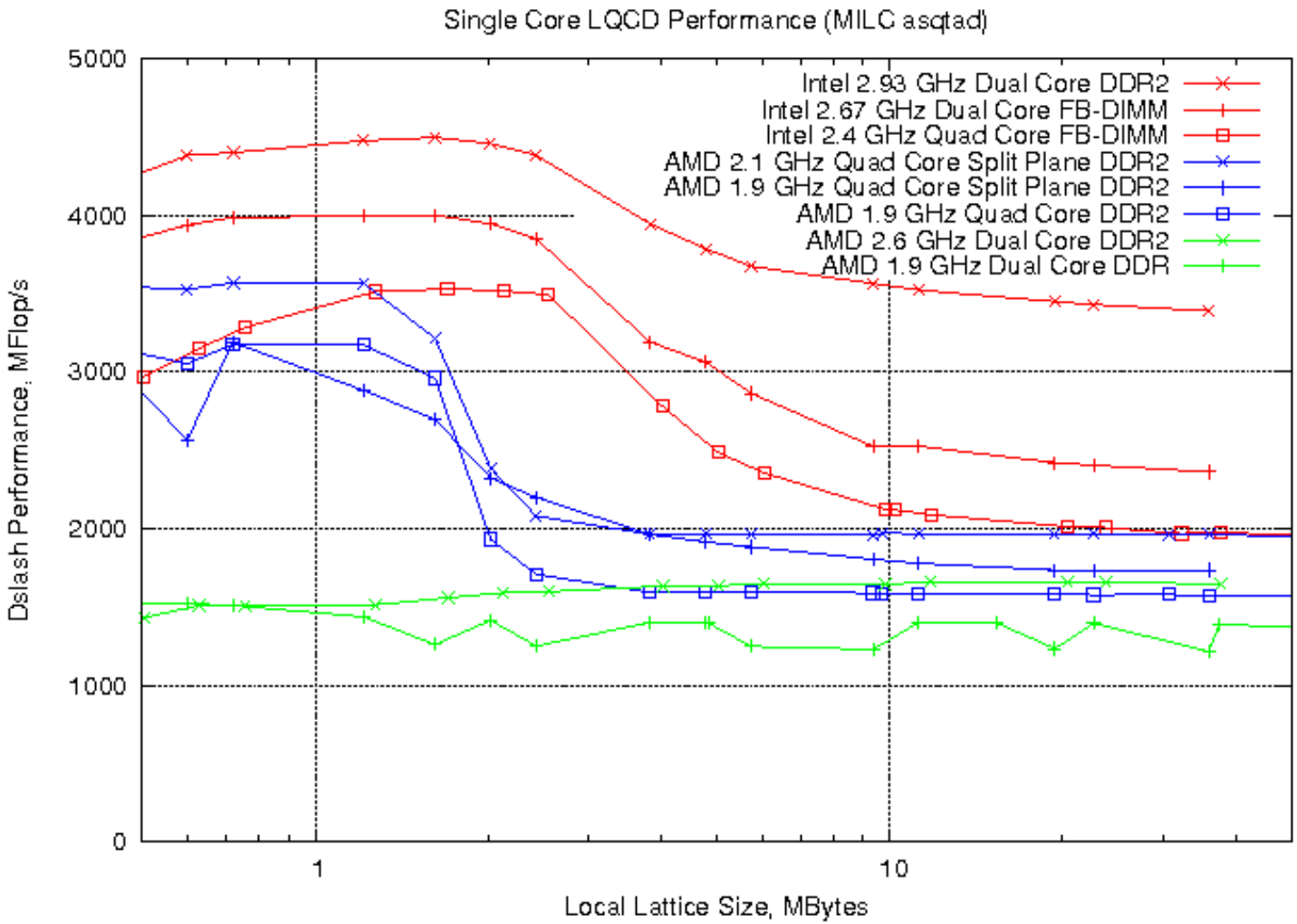
AMD Opteron SMP Architecture

NUMA Effects



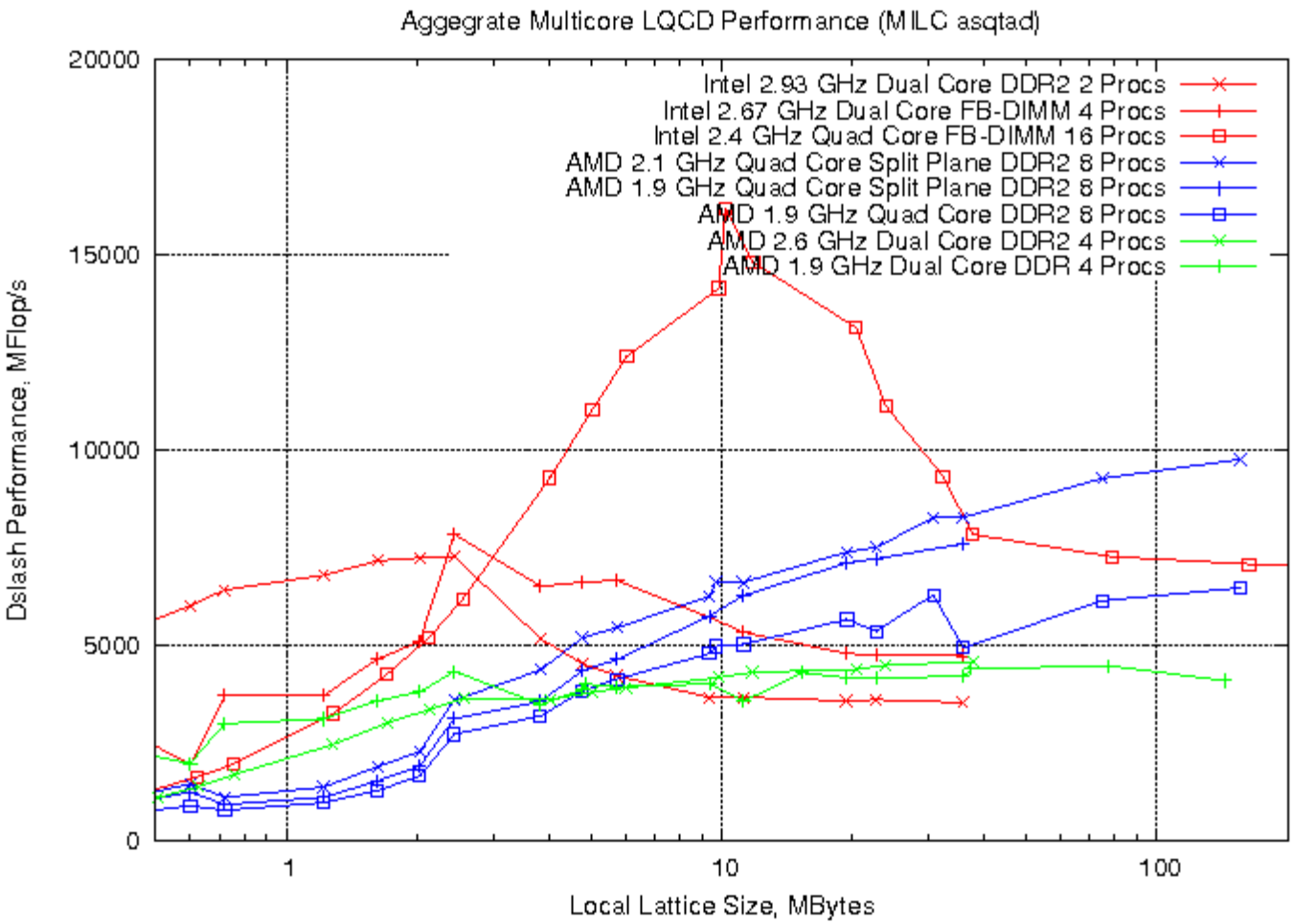
- Plots show performance of (bottom lines) single code instances and (top lines) aggregate performance of 4 MPI processes
- Non-local memory used (via numactl)
- Local memory used (via numactl)
- On NUMA clusters local memory must be used *and* processes must be locked to cores

Performance: Single Node, Using a Single Core on LQCD Code



- Plots show performance of single code instance
- Intel Dual and Quad "Core" architecture
- AMD Quad (Barcelona)
- AMD Dual Socket-940 and Socket-F

Performance: Single Node, Using All Available Cores on LQCD Code



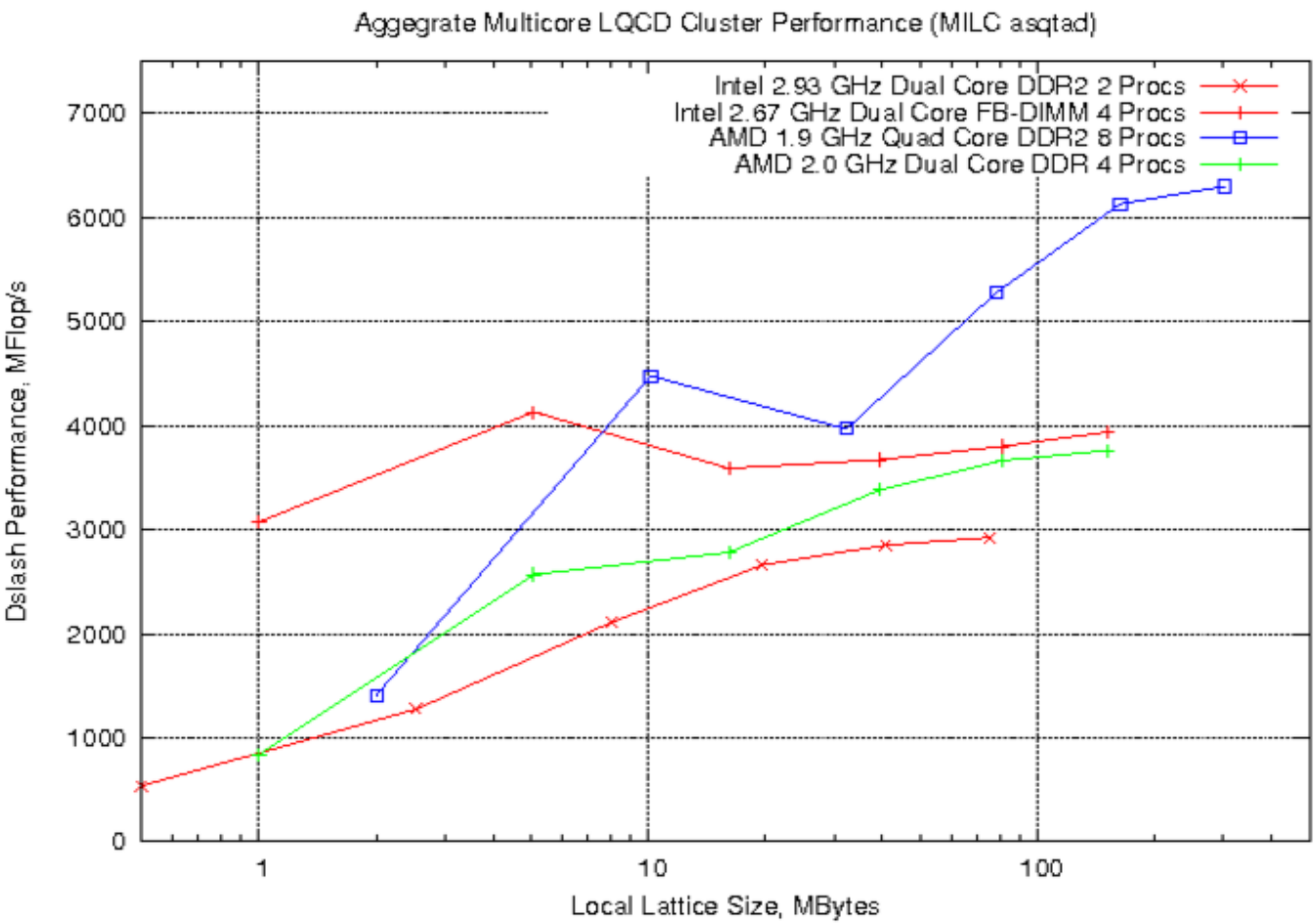
- Plots show aggregate performance of one MPI process on each core
- Intel Dual and Quad "Core" architecture
- AMD Quad (Barcelona)
- AMD Dual Core Socket-940 and Socket-F
- In Spring 2006 best price/performance was AMD Socket-F
- In Spring 2007 best price/performance was AMD Barcelona

Performance: Cluster LQCD

CPU	Machine	Interconnect	Per Node Performance (#cores)
2.93 GHz Pentium dual core	Single socket	SDR Infiniband	2925 Mflops (2 cores)
2.66 GHz Xeon dual core	Dual socket	SDR Infiniband	3939 Mflops (4 cores)
2.0 GHz AMD Dual	Dual Socket	DDR Infiniband	3761 Mflops (4 cores)
1.9 GHz Barcelona	Dual Socket	DDR Infiniband	6298 Mflops (8 cores)

- Performance of *dslash* inverter on four Infiniband clusters when running 14^4 sublattices on each available core, total of 128 processes
- Scaling on Intel is limited by shared memory bus
- MPI communications also affect scaling

Performance: Cluster LQCD



- Plots show weak scaling performance on 128 MPI processes as sublattice size varies from 4^4 to 14^4
- Intel dual-core, on single socket (DDR, lower) and dual socket (FB-DIMM, upper) nodes
- AMD dual core DDR Socket 940
- AMD quad core DDR2 (Barcelona)

Performance: Predictions

- Expect strong boost in memory bandwidth from Intel in 2008
  - DDR2 (single socket) and FB-DIMM (multiple sockets) fall far short of providing balance for LQCD codes on Intel “Core”
  - “Core” architecture is very strong for floating point, but current designs cannot feed the processors enough data for LQCD
    - Quad cores (and higher) will make the situation worse
  - Intel will introduce (late 2008) processors with imbedded triple channel memory controllers as fast as 1600 MHz (“Nehalem”)
    - Perhaps a 3-fold increase in memory bandwidth
    - NUMA, however, so we will have to apply lessons from Opterons
- Expect larger L2 and L3 (“last level”) caches from Intel and AMD
  - 16 MB pr core (at least) needed for LQCD - unlikely

Summary and Conclusions

- Memory bandwidth limits LQCD performance on current x86 architectures
- The best current architecture for LQCD in terms of price/performance is the new AMD quad core processor (Barcelona) on motherboards with independent clocking of processor and memory
- We have no performance data yet on the upcoming Intel “Penryn” processors but believe that with FB-DIMMs memory bandwidth will still limit performance
- The Intel “Nehalem” generation with DDR3 and NUMA may shift the bottleneck from memory bandwidth to floating point