Performance of MILC Lattice QCD Code on Commodity Clusters

Don Holmgren (djholm@fnal.gov)

Fermi National Accelerator Laboratory

P.O. Box 500

Batavia, IL 60510–0500, USA

DESY Zeuthen – 26 August 2002

S. Gottlieb[2], D. Panda[3], R. Rechenmacher[1], S. Senapathi[3], J. Simone[1]

[1]Fermilab

[2]Indiana University

[3]The Ohio State University

http://qcdhome.fnal.gov/DESY-Zeuthen.pdf

Outline

- Introduction to MILC Lattice QCD Code
- Fermilab Hardware
- MILC Single Node Performance
- MILC SMP Performance
- MILC Cluster Performance
- SciDAC Initiative (if time)

Lattice QCD Codes

- Quantum Chromodynamics (QCD) is the theory of the strong interaction between quarks, mediated by gluons, expressed by the Dirac action for quarks

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

where the Dirac operator $\not{D}$ ("*dslash*") is given by

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

and ψ is the quark wavefunction

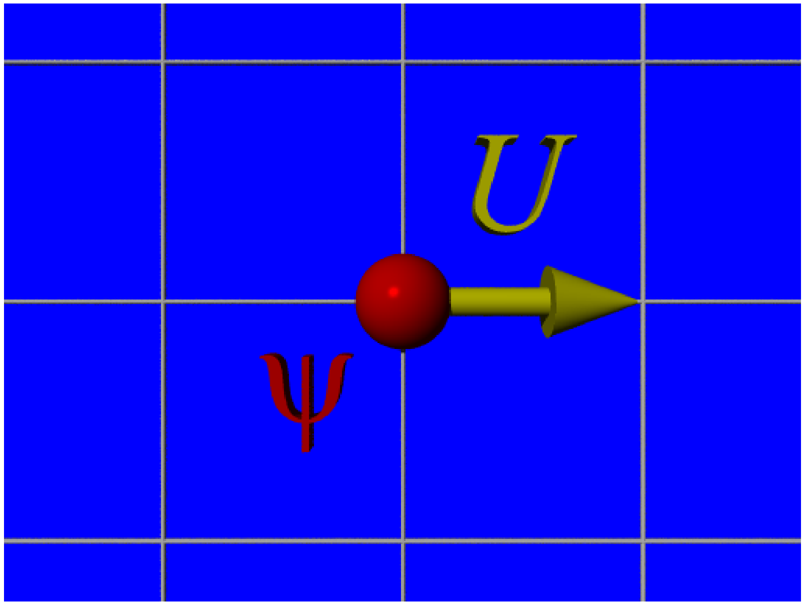
- Lattice QCD is the numerical simulation of QCD, using discretized space and time
- A very simple discretized form of the Dirac operator is

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

where a is the lattice spacing.

- Alternate representations of ψ or additional operators higher order in a lead to alternate discrete quark *actions*
- $(\not{D} + m)$ is a sparse matrix, invertable using techniques such as conjugate gradient

- A quark, $\psi(x)$, depends upon $\psi(x + a\hat{\mu})$ and the local gluon fields U_μ



- $\psi(x)$ is expressed as a number of SU3 vectors (complex, 3X1), and the U_μ are expressed as SU3 matrices (complex, 3X3). Interactions between the $\psi(x)$ and U_μ are computed via matrix algebra.

- On a clustered system, the lattice is divided among the computers. On the boundaries between computers, data must be interchanged to compute $\not{D}\ \psi(x)$

- MILC (MIMD Lattice Computation) is an implementation of Lattice QCD.

  See http://media4.physics.indiana.edu/ sg/milc.html

MILC "Improved Staggered"

- "*Improved Kogut-Susskind*", also called "*improved staggered*", is one of the actions available in MILC
- Sequence of operations in MILC for inverting $\not{D}$ (conjugate gradient technique):
  - start gather of data from positive directions
  - multiply quark vectors by U-matrices (gluon fields) along negative directions
  - start gathers from negative directions
  - wait on gathers from positive directions
  - multiply quark vectors by U-matrices (gluon fields) along positive directions
  - wait on gathers from negative directions
  - accumulate results
  - check convergence (global sums)
  - loop while not converged
- MILC code stresses:
  - floating point performance
  - memory bandwidth
  - network performance

Fermilab Hardware

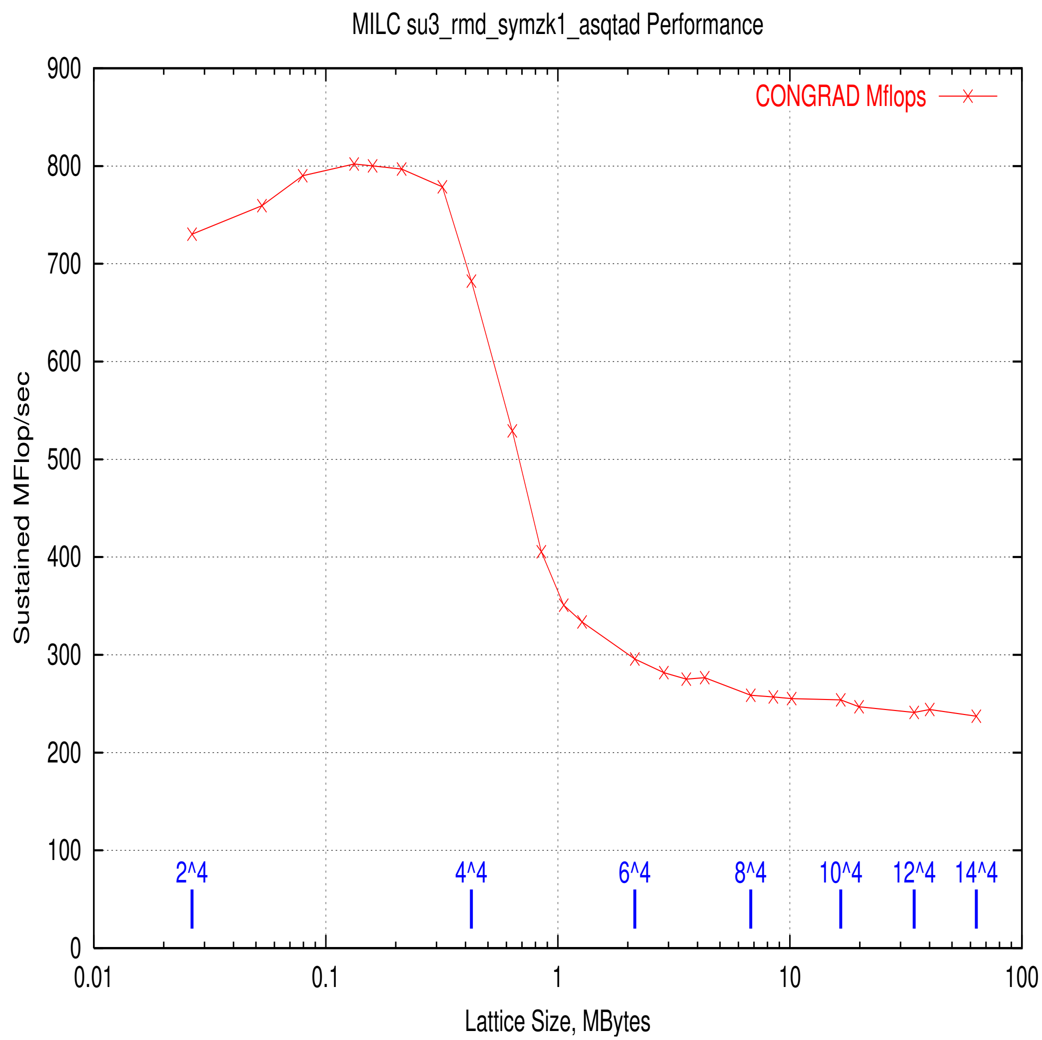
Lattice QCD Clusters at Fermilab

- QCD80 (entered production January 2001)
  - 80 Dual 700 MHz Pentium III systems
  - 256 MB SDRAM (100 MHz) Memory
  - Intel L440GX+ motherboards, with integrated IPMI
  - Myrinet 2000 (133 MHz Lanai)
- NQCD (entered production August 2002)
  - 48 Dual 2.0 GHz Xeon (“Prestonia”) systems
  - 1.0 GB memory (interleave 200 MHz DDR)
  - SuperMicro P4DPE motherboards (E7500 chipset)
  - IPMI option card
  - Myrinet 2000 (133 MHz Lanai), shares switch with QCD80
- Near Future (in production November 2002)
  - 128 additional Dual 2.0 GHz Xeon systems (same as NQCD), or
  - Approximately 64 Itanium 2 (900 MHz) systems
  - Myrinet 2000

Single Node Performance

MILC Performance on 2.0 GHz Xeon

- Sample Performance Graph
  - Typical lattice has dimensions of Lˆ3 X T
  - Each site is 1656 bytes large
  - Blue ticks mark $(2,4,6,8,10,12,14)^4$
  - L2 cache (512K) near 4^4
  - FPU dominates for lattices smaller than 4^4
  - Memory bandwidth dominates for lattices larger than 4^4



Single Node Performance

In-Cache Floating Point Performance

Processor	Matrix-Vector	Matrix-HWVector	Matrix-Matrix
700 MHz Pentium III	305	300	302
1.2 GHz Athlon	679	806	637
1.4 GHz / 400 MHz P4	622	627	661
2.0 GHz / 400 MHz Xeon	905	823	954
2.26 GHz / 533 MHz P4	1046	1053	1111

- Performance in MFlop/sec, single precision, using MILC “C” library
- MILC SU3 matrices and vectors:

  Matrix = 3x3 complex

  Vector = 3x1 complex

  HWVector = 3X2 complex
- See http://qcdhome.fnal.gov/qcdstream/

Single Node Performance

Memory Bandwidth

Processor	Memory Type	Chipset	Bandwidth MB/sec
Pentium III	100 MHz SDRAM	440GX	330
Athlon	DDR200	760MP	700
Xeon	DDR200	GC-HE	935
	DDR200	E7500	1240
	PC800 RDRAM	i860	1305
	(SSE assist)	i860	2121
Pentium 4	PC800 RDRAM	i850	1320
	PC1066 RDRAM	i850E	2035
Itanium 2	DDR266	zx1	2460

- Measured with the Streams benchmark
- Shows rate of copying *double* to *double* at 100% CPU utilization
- Values depend on *memory type, interleave, bus width*
- Why are these numbers so far below the theoretical maximum of 3.2 GByte/sec?

  SSE assist = 128 bit loads/stores, cache bypass writes

Single Node Performance

SU3 Linear Algebra Performance

Access Pattern	Matrix-Vector (MFlop/Sec)	Matrix-Matrix (MFlop/sec)
In Cache	905	954
Sequential	710	815
Strided	139	292
Mapped	131	265

- Measured on 2.0 GHz Xeon system (E7500, interleaved DDR) using qcdstreams
- Measured during loop over `i` with these patterns:
  - In Cache: `mat_vec(a[0], b[0], c[0])`
  - Sequential: `mat_vec(a[i], b[i], c[i])`
  - Strided: `mat_vec(a[i*s], b[i*s], c[i*s])`, s=stride constant
  - Mapped: `mat_vec(a[map[i]], b[map[i]], c[map[i]])`

Single Node Performance

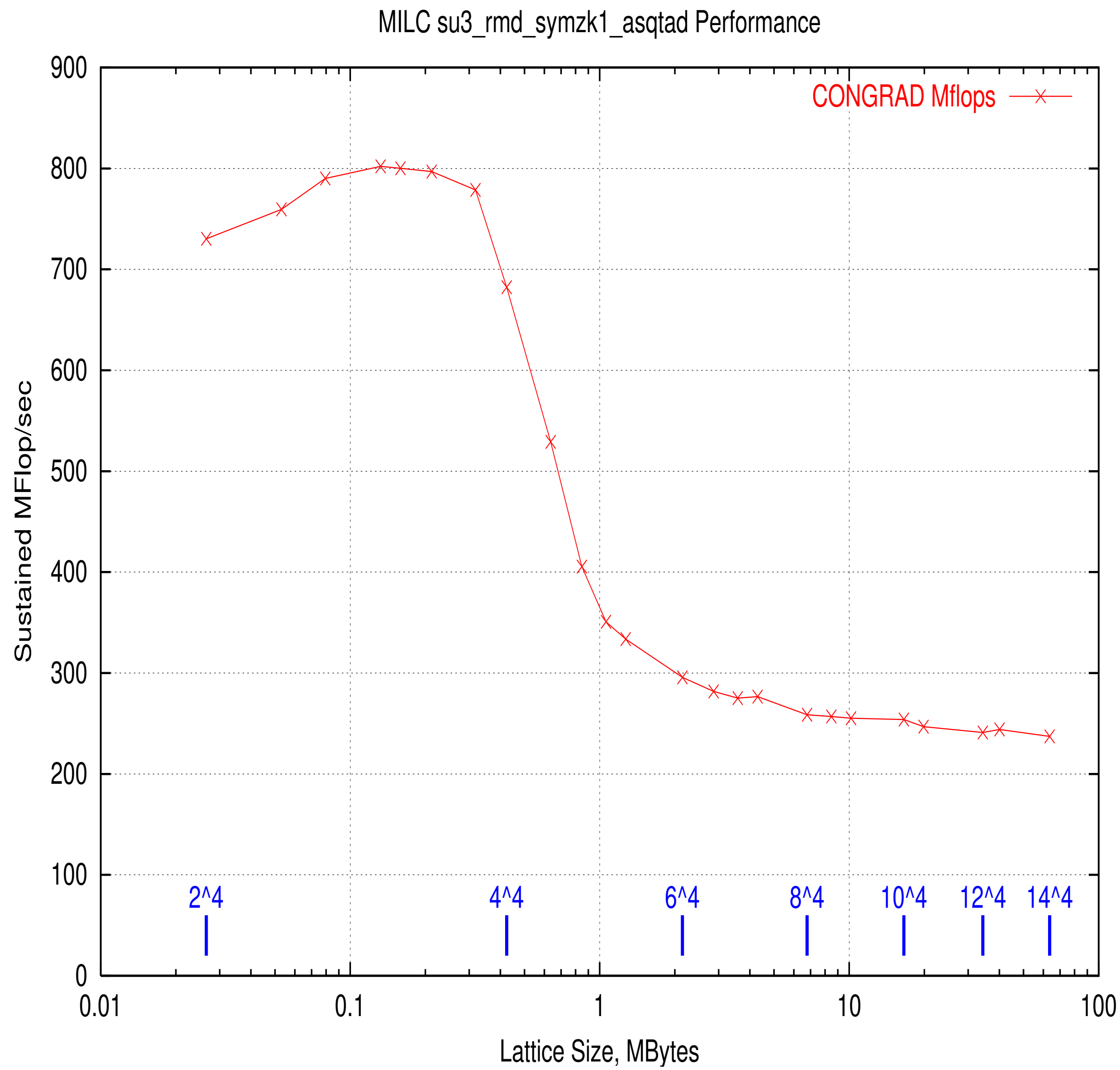
Predicting MILC Performance - 2.0 GHz Xeon, E7500

- Assumption:

  all Matrix-Vector
- In-cache upper bound: 905
- In-memory bounds:

  130 (strided/mapped)

  710 (sequential)



Single Node Performance

SU3 Linear Algebra Performance

Access Pattern	Matrix-Vector (MFlop/Sec)	Matrix-Matrix (MFlop/sec)
In Cache	905	954
Sequential	710 (1553)	815 (2590)
Strided	139 (540)	292 (1326)
Mapped	131 (483)	265 (1202)

- Measured on 2.0 GHz Xeon system (E7500, interleaved DDR) using qcdstreams
- Measured on a 900 MHz Itanium 2 system (interleaved DDR) using John Dupuis' optimized kernels
- Measured during loop over `i` with these patterns:
  - In Cache: `mat_vec(a[0], b[0], c[0])`
  - Sequential: `mat_vec(a[i], b[i], c[i])`
  - Strided: `mat_vec(a[i*s], b[i*s], c[i*s])`, s=stride constant
  - Mapped: `mat_vec(a[map[i]], b[map[i]], c[map[i]])`

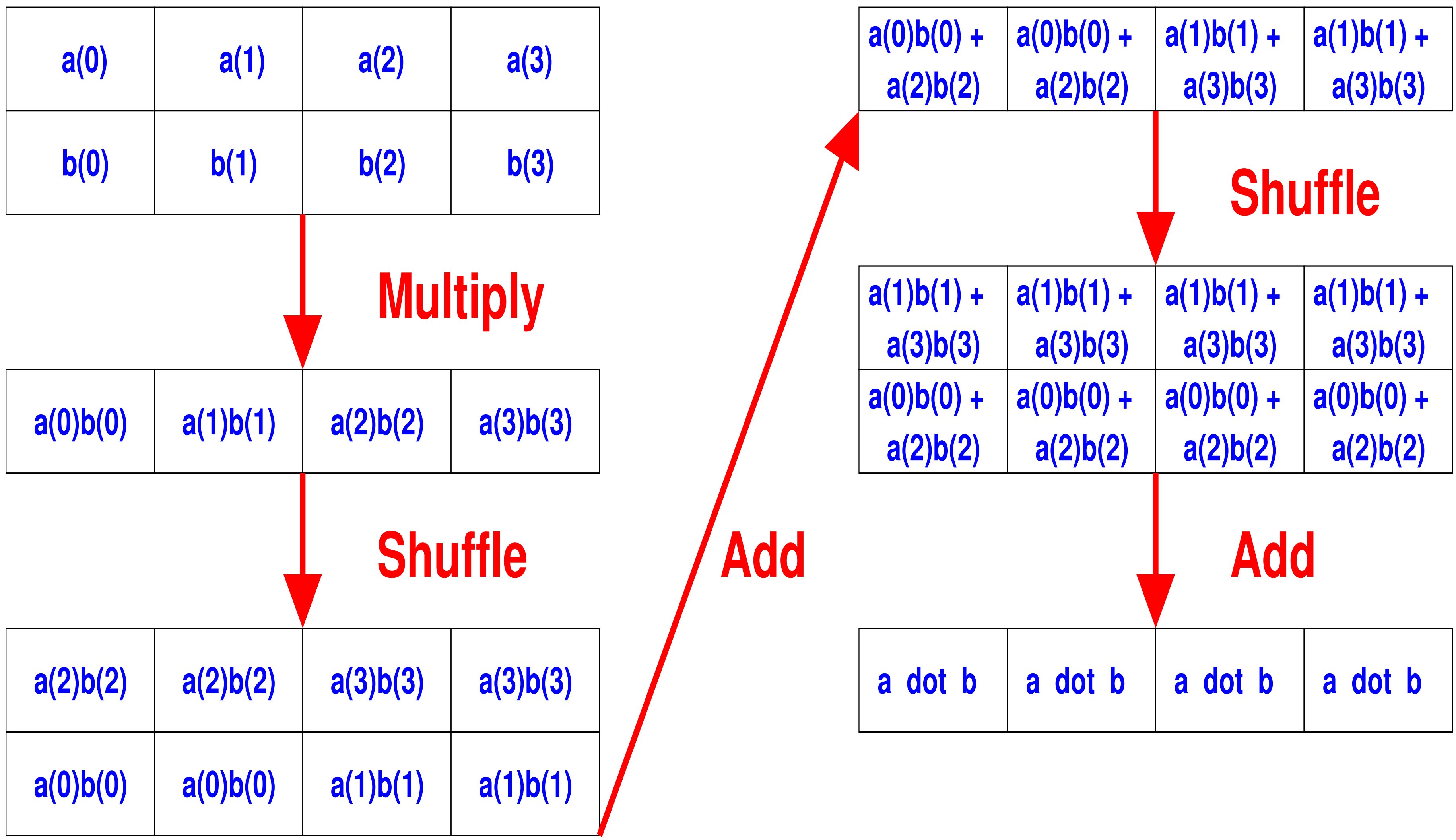
Single Node Performance

SSE Optimizations

- SSE = Streaming SIMD Extensions
  - Implemented via eight 128-bit wide XMM registers
  - Available on Pentium 4, Xeon
  - Also emulated on Athlon
- SSE allows 4-wide SIMD single precision floating point operations
  - SSE2 allows 2-wide double precision operations
- M. Luescher communicated very encouraging P-III and P4 results in 2001
- Access from compilers:
  - Intrinsics in Intel *ICC* and newer *GCC* (3.1) (inline)
  - Via *GCC* inline assembler (compatible with latest Intel *ICC*) (inline)
  - Link to object files from an assembler (*GNU as, nasm*) (subroutine only)

Single Node Performance

SSE Example - Inner Product



Single Node Performance

SSE SU3 Matrix Algebra - In Cache Performance

Operation	"C" Cycles	SSE Cycles	MFlops/GHz
Matrix-Vector Multiply	124	57	1158
Matrix-Matrix Multiply	414	130	1523
Matrix-HWVector Multiply	268	73	1808

- In cache performance
- Results shown are for Pentium 4 (Xeon results are equivalent)
- See http://qcdhome.fnal.gov/sse/
- Matrix-HWVector is M. Luescher's code
- Implemented via inline GCC assembler macros - subroutine calls cost 20-30 cycles

Single Node Performance

SSE SU3 Linear Algebra Performance

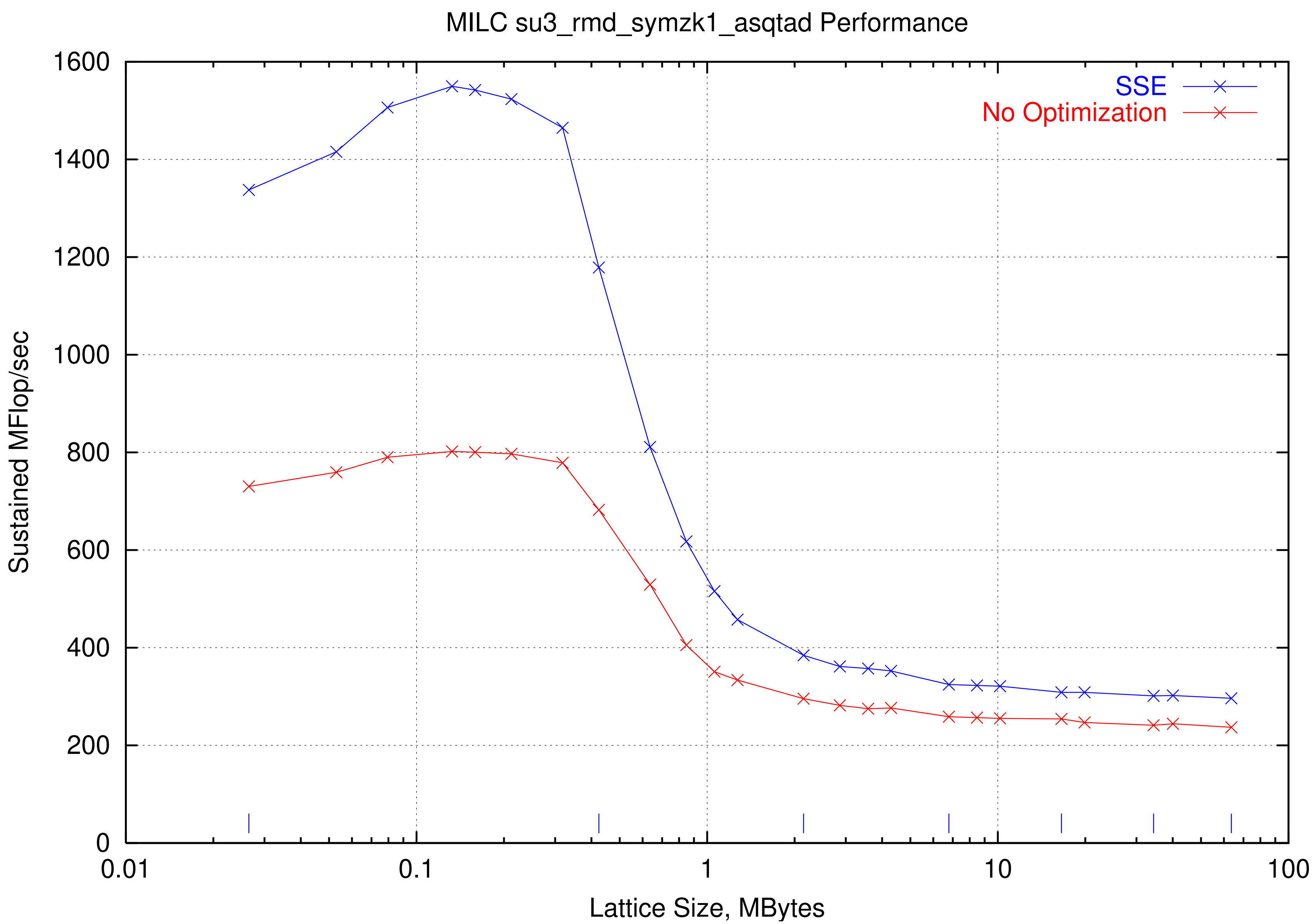
- Performance degradation out of cache:

Code	Pattern	Matrix-Vector	Matrix-HWVector	Matrix-Matrix
"C"	In Cache	905	823	954
	Sequential	710	729	815
	Strided	139	212	292
	Mapped	131	196	265
SSE	In Cache	2124	3514	2985
	Sequential	846	1135	1248
	Strided	148	273	316
	Mapped	156	283	248

- Results shown are MFlops/sec on 2.0 GHz Xeon (E7500 interleaved DDR)
- Optimization is clearly constrained by memory bandwidth

Single Node Performance

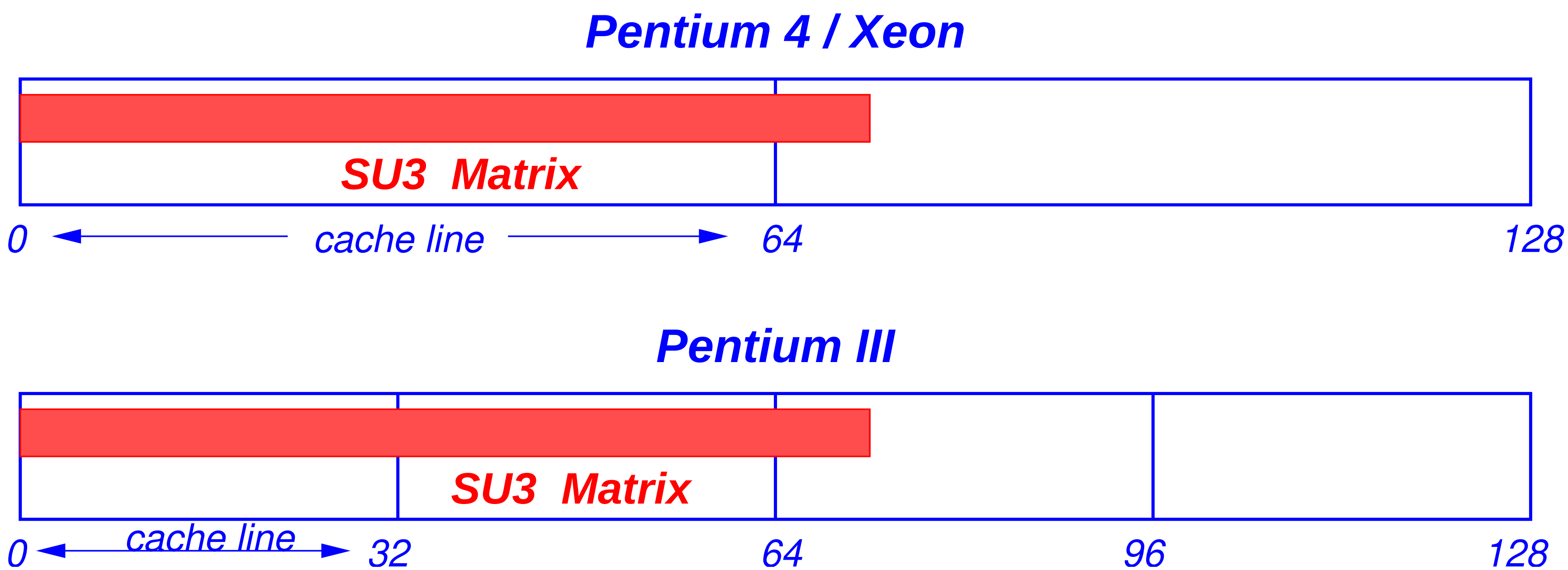
MILC Performance - SSE Optimizations - 2.0 GHz Xeon, E7500



Single Node Performance

Field Major Optimization

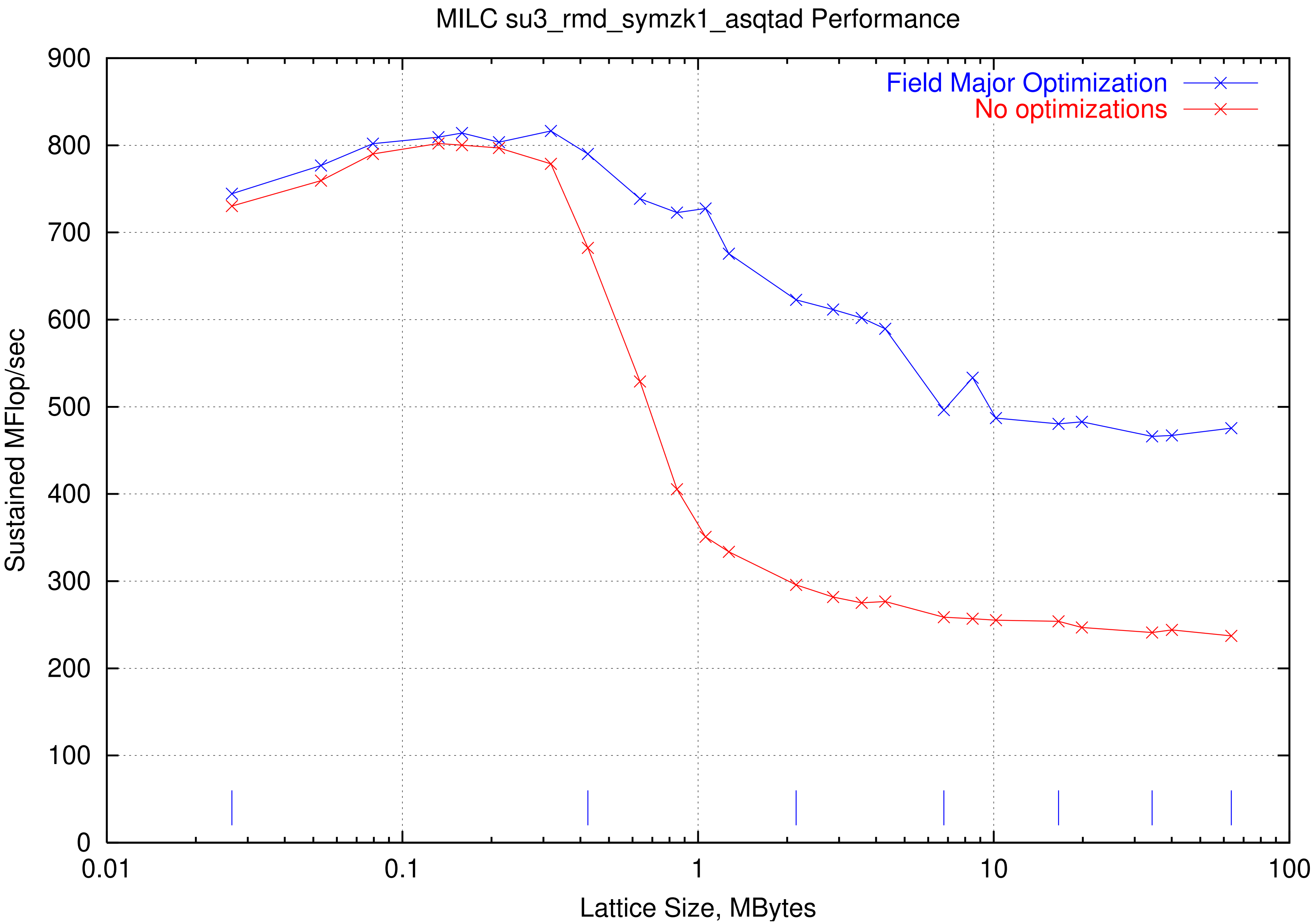
- Standard MILC lattice layout is site major
  - each lattice site has a number of fields (vectors, matrices, scalars)
  - stride between corresponding fields is large (1656 bytes on improved staggered)



- SU3 matrix size = 72 bytes
  - Load efficiency on Pentium III = 75%
  - Load efficiency on Pentium 4 = 56%
- Field major layout boosts load efficiency
- MILC concept - Steve Gottlieb, MILC Implementation - Dick Foster

Single Node Performance

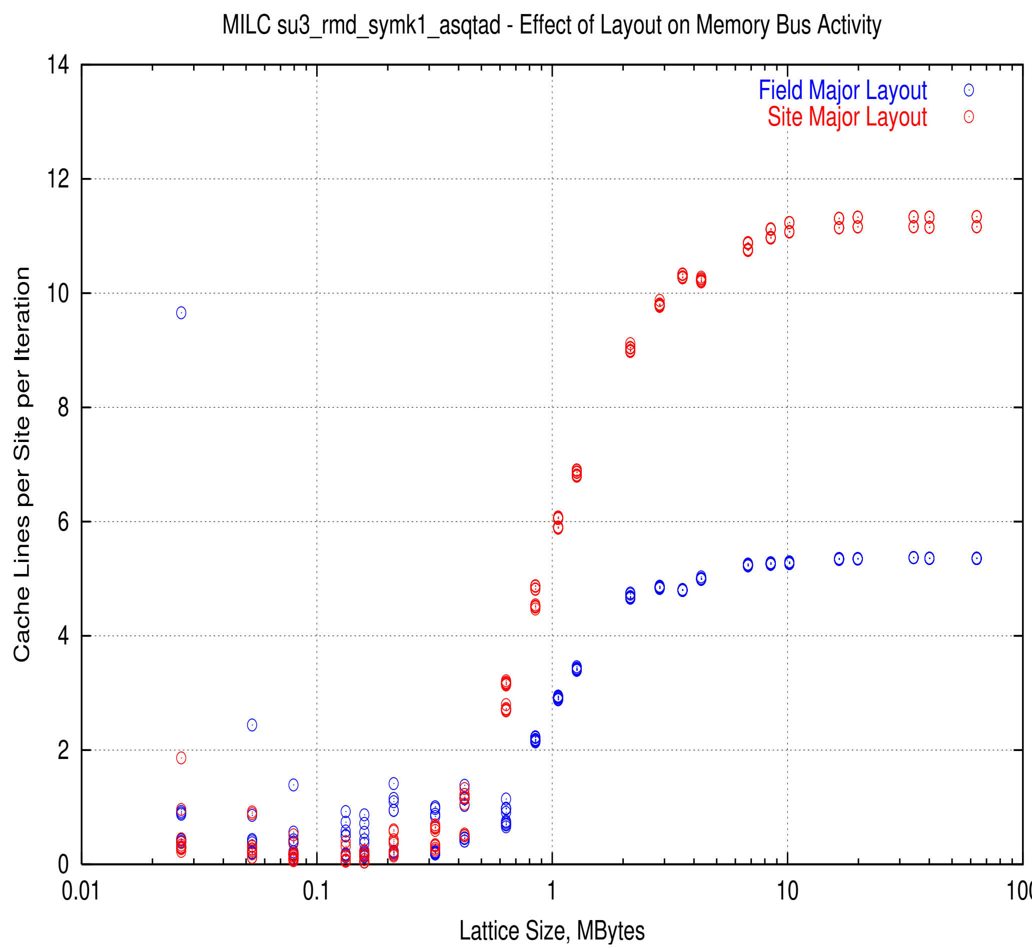
MILC Performance - Field Major Optimization - 2.0 GHz Xeon, E7500



Single Node Performance

Field Major Optimization - Cache Line Loads

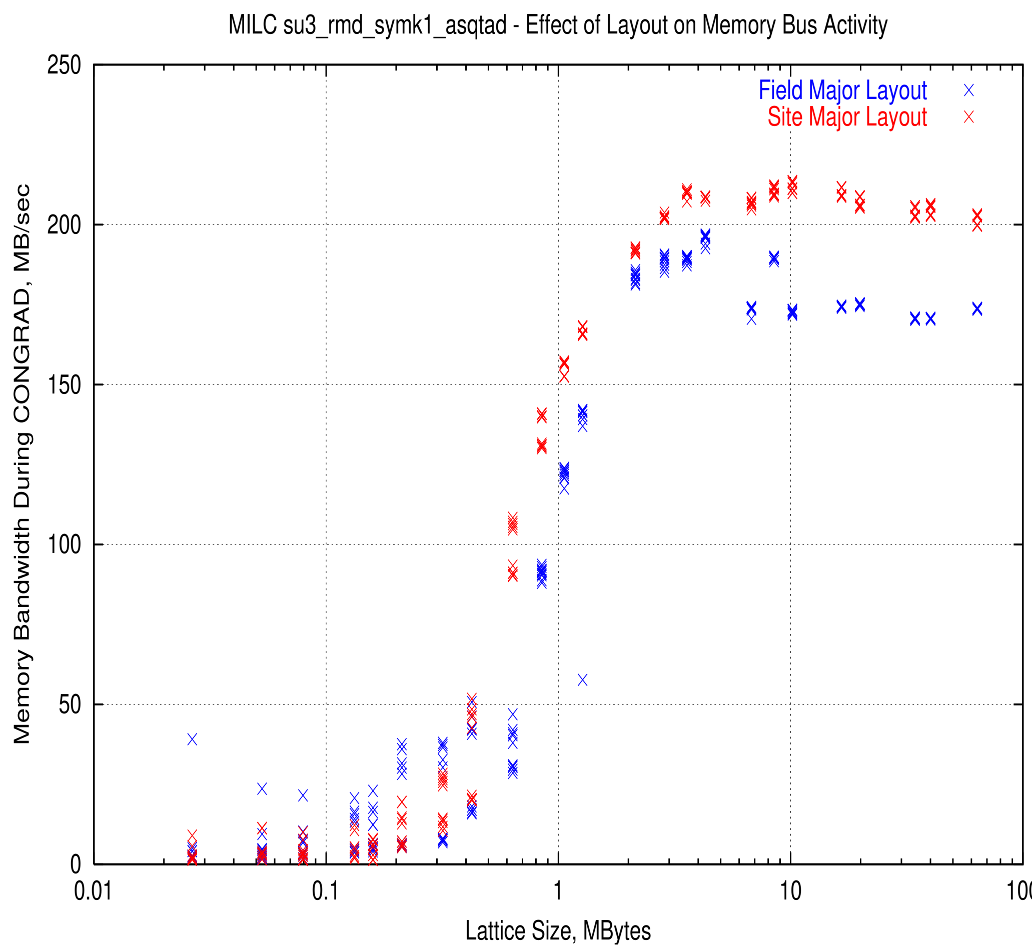
- Instrument code:
  - Use Performance Counters
  - Count cache line loads/stores
  - Count retired flops
- Cache line loads per lattice site per iteration shown
- I haven't done the simple back of the envelope excercise to verify that these are the expected out-of-cache asymptotes



Single Node Performance

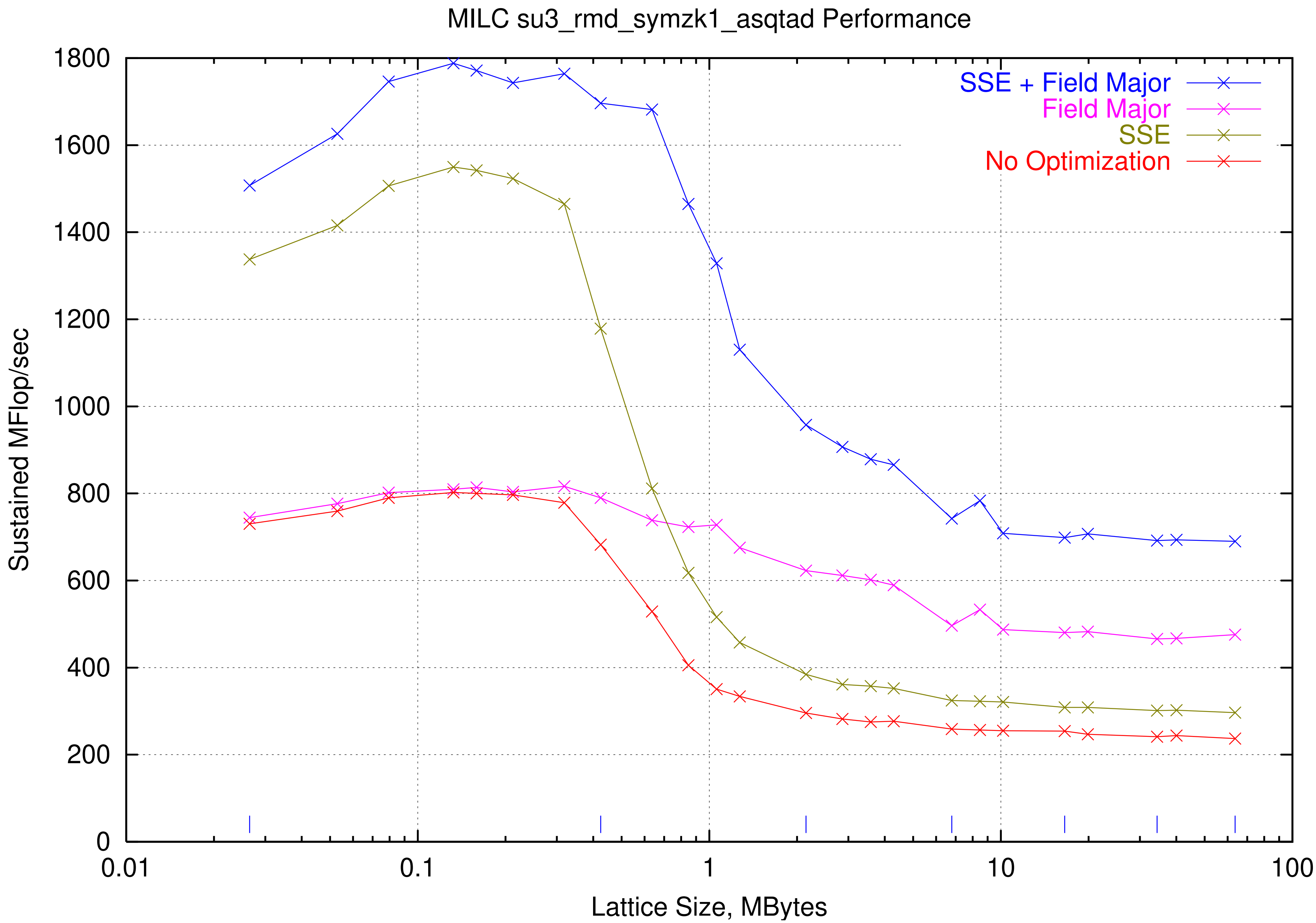
Field Major Optimization - Utilized Memory Bandwidth

- Average memory bandwidth utilization during CONGRAD
- Field major doubles performance and reduces demand on memory bus
- Suggests further optimization by prefetching (note *streams* value is approx. 1300 MB/sec)
- Why aren't these bandwidths equal to 1300 MB/sec if the code is memory bandwidth bound?



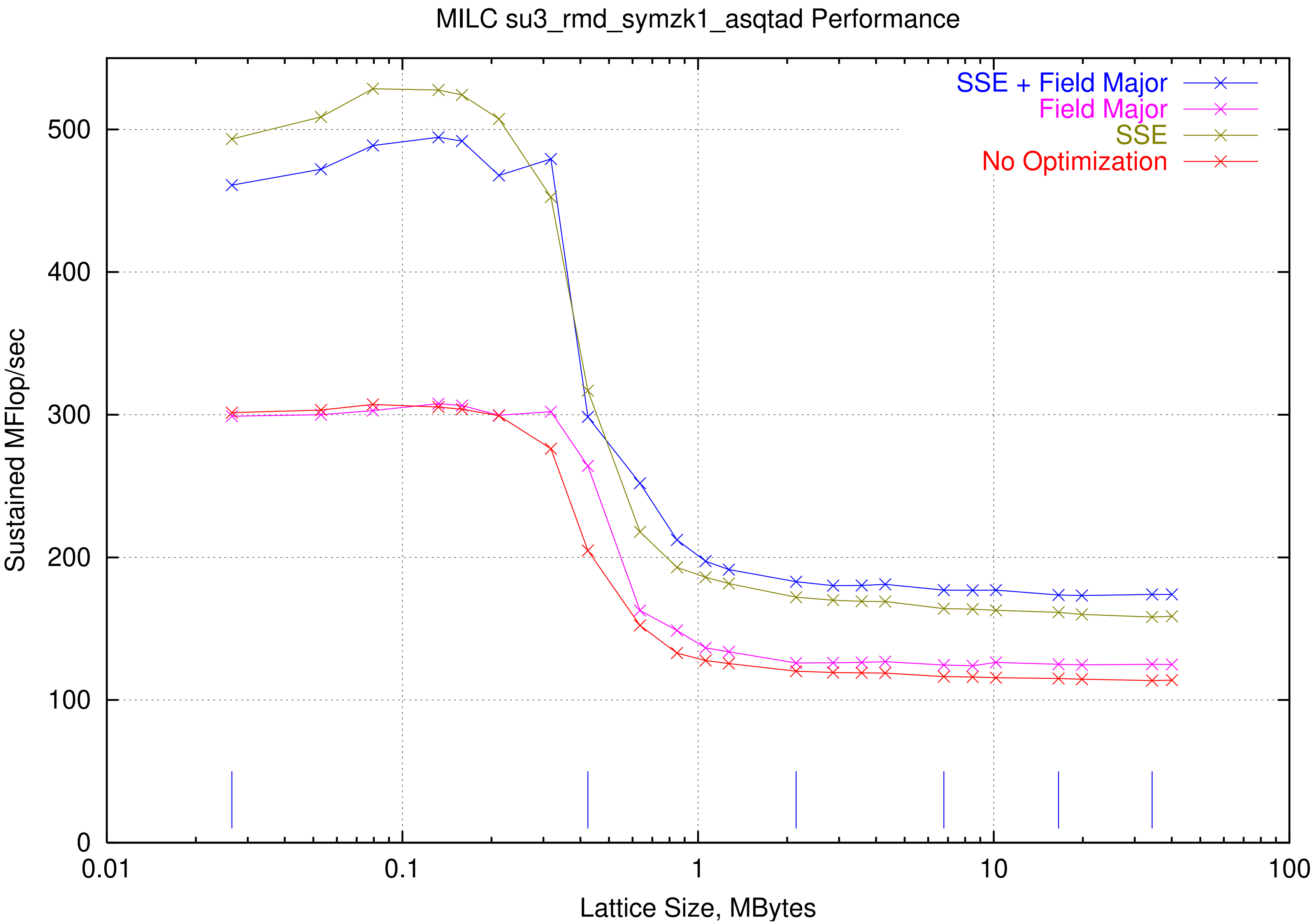
Single Node Performance

All Optimizations - 2.0 GHz Xeon, E7500



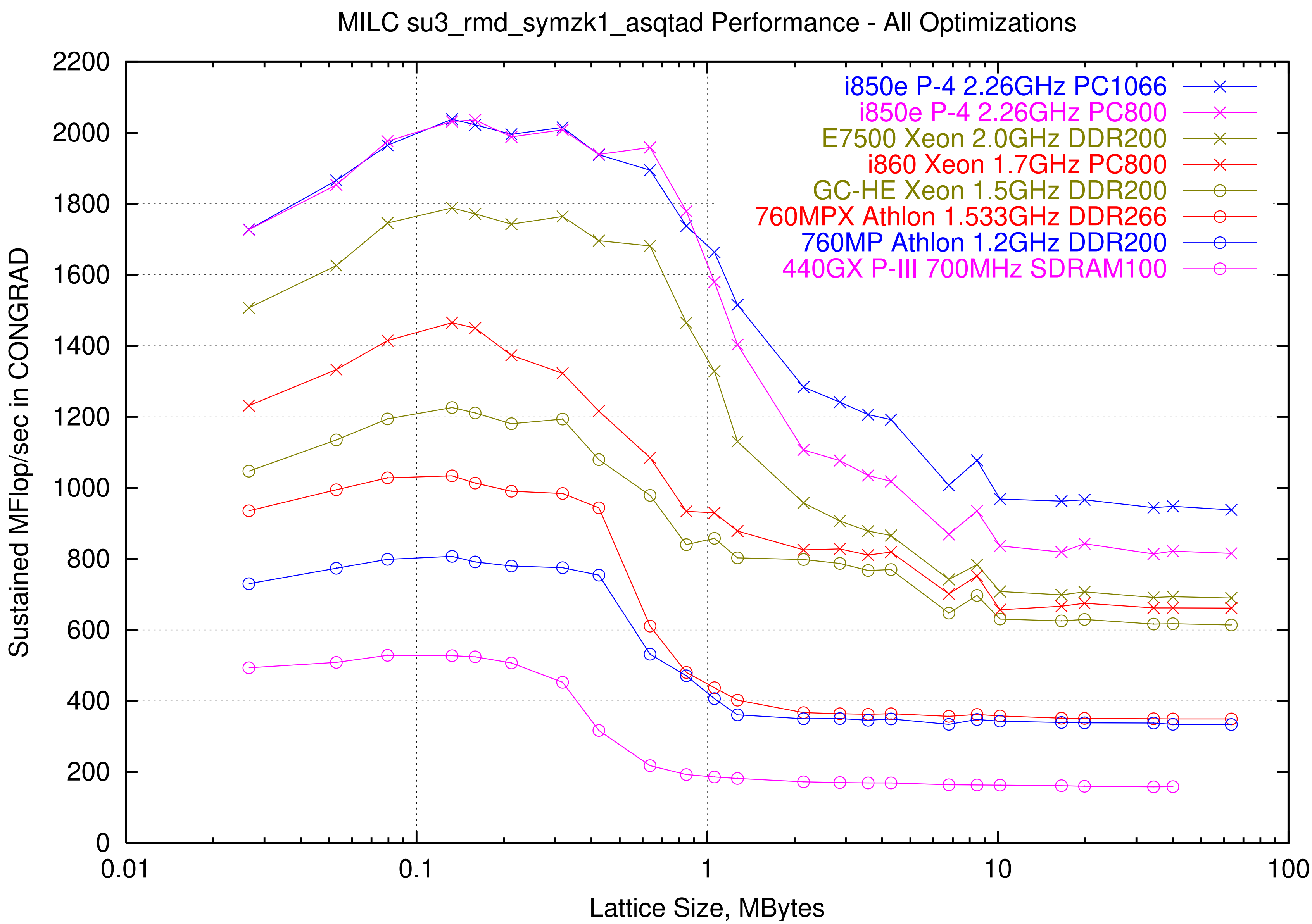
Single Node Performance

All Optimizations - 700 MHz Pentium III, 440GX



Single Node Performance

Survey of MILC Performance with All Optimizations



Single Node Performance

Other Potential Optimizations

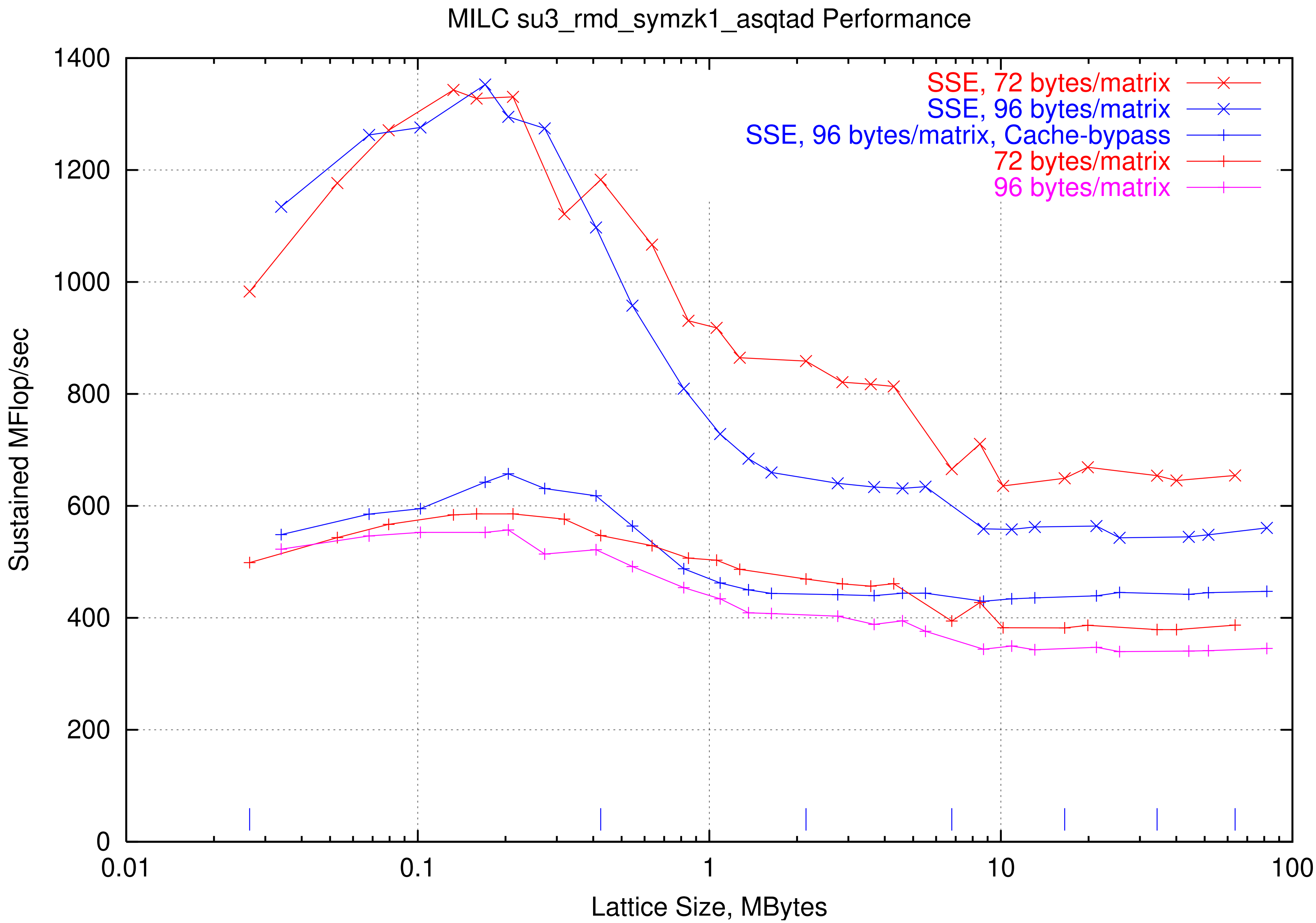
- “Paragraph alignment” (16 bytes)
  - SSE aligned moves are faster than unaligned
  - Requires padding in MILC layout:

    Increase matrices from 72 to 96 bytes

    Increase vectors from 24 to 32 bytes
- Cache bypass writes
  - SSE allows aligned writes directly to memory
  - Done carefully, minimizes cache thrashing
- Early results are negative
  - Evidently, not implemented carefully enough!
  - But, red-black decomposition implies that I *don’t* have to be careful to minimize thrashing the cache

Single Node Performance

Performance of Additional "Optimizations" - 1.4 GHz P4 - RDRAM



SMP Performance

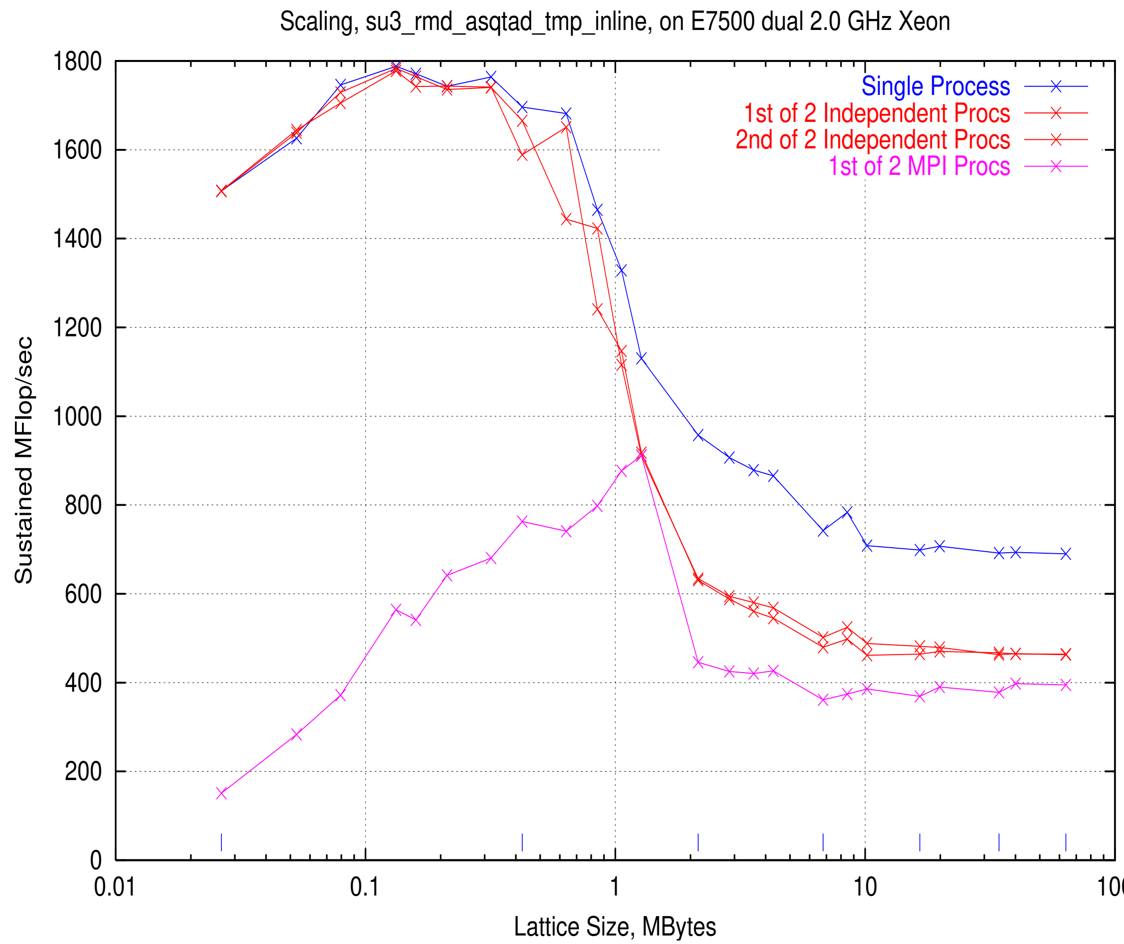
Why Consider SMP?

- If communication bandwidth sufficient, minimize cost of network interfaces
- Fast, wide PCI buses, and PCI-X
  - Single Pentium 4 mainboards have only narrow, slow PCI
- Incremental cost of second processor is low
  - If codes scale well, better performance/price
- Server class features
  - Hardware management - IPMI, BIOS redirect
  - Integrated video, network
- Minimize number of machines to manage

SMP Performance

SMP Performance on Dual Processor 2.0 GHz Xeon - E7500 Interleaved DDR

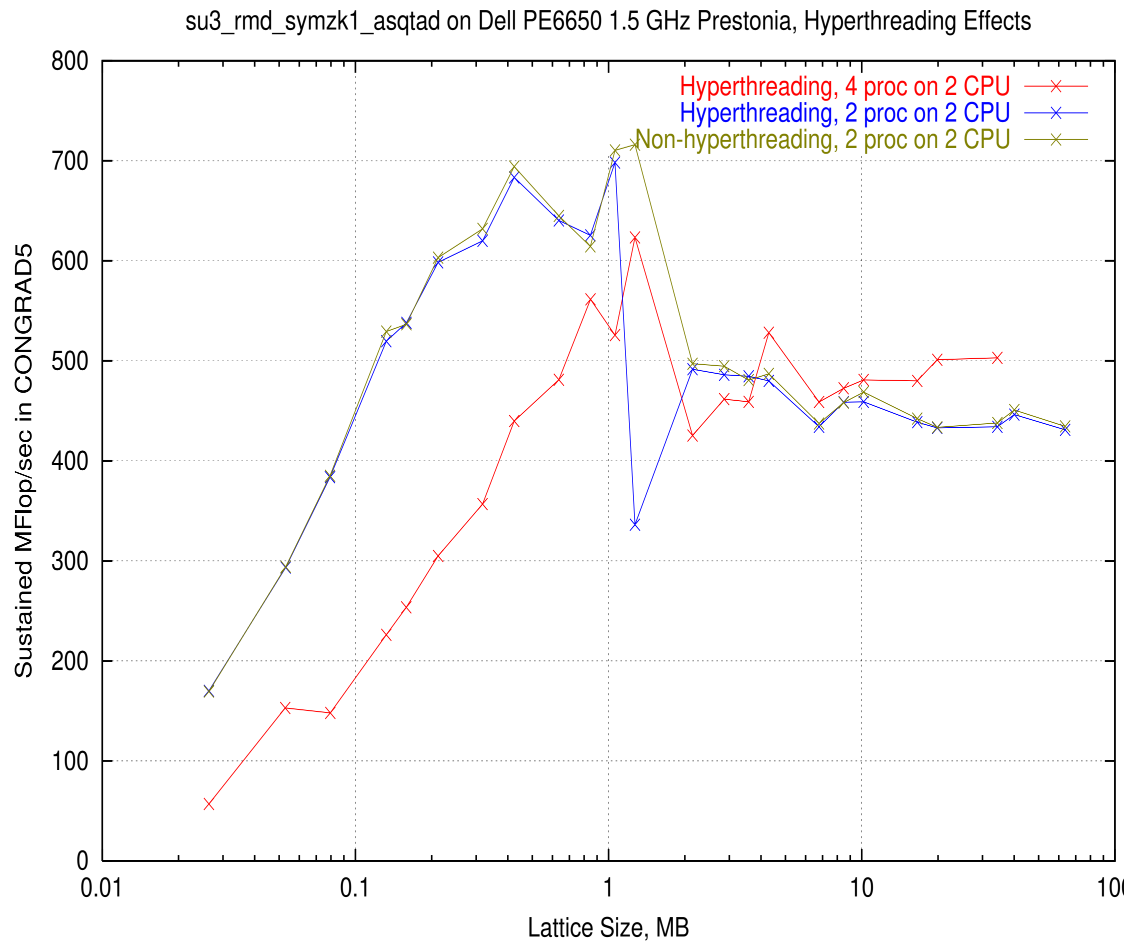
- Scaling on dual cpu machine:
  - 65% on independent processes
  - 55% on cooperative processes
- Not surprising since single processes are memory bandwith bound



SMP Performance

SMP Performance on GC-HE - Hyperthreading

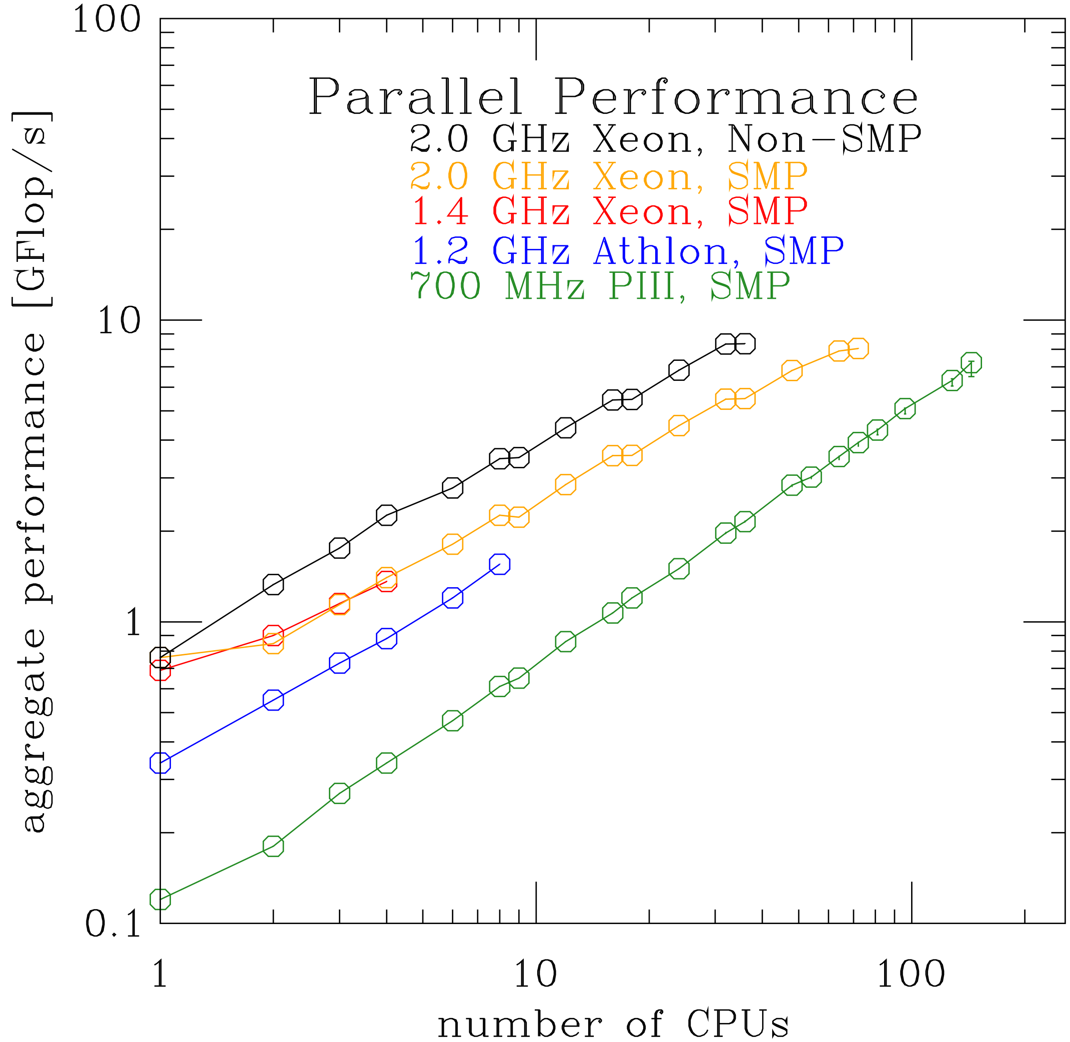
- Hyperthreading - 2 virtual CPUs per physical CPU
- Slower in cache, faster in main memory



Cluster Performance

Scaling - Constant Volume per Run

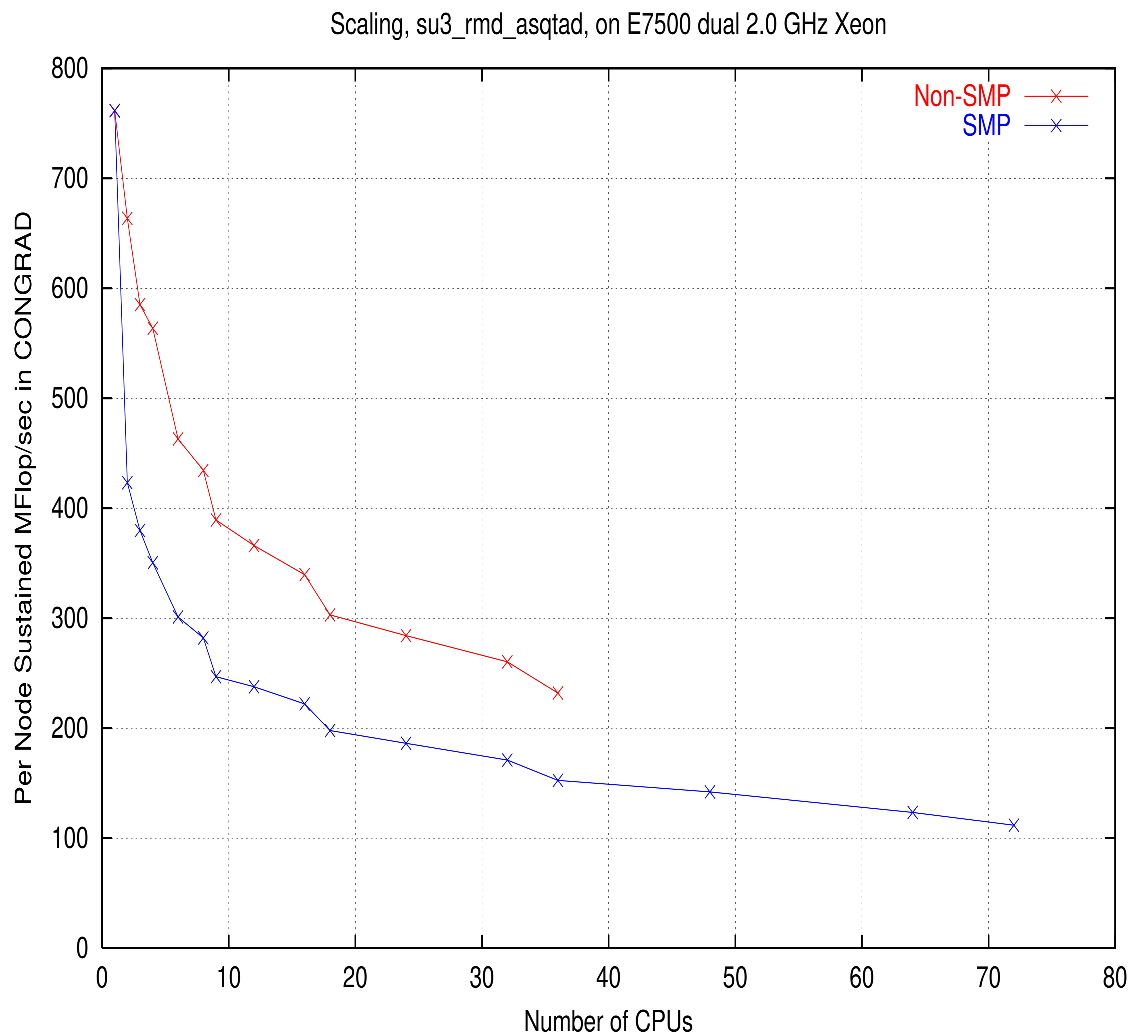
- Fixed total volume (12^3 X 24) is divided among nodes
- Graph shows aggregate performance
- Communications over Myrinet using *mpich-gm*



Cluster Performance

Scaling - Constant Volume per Run

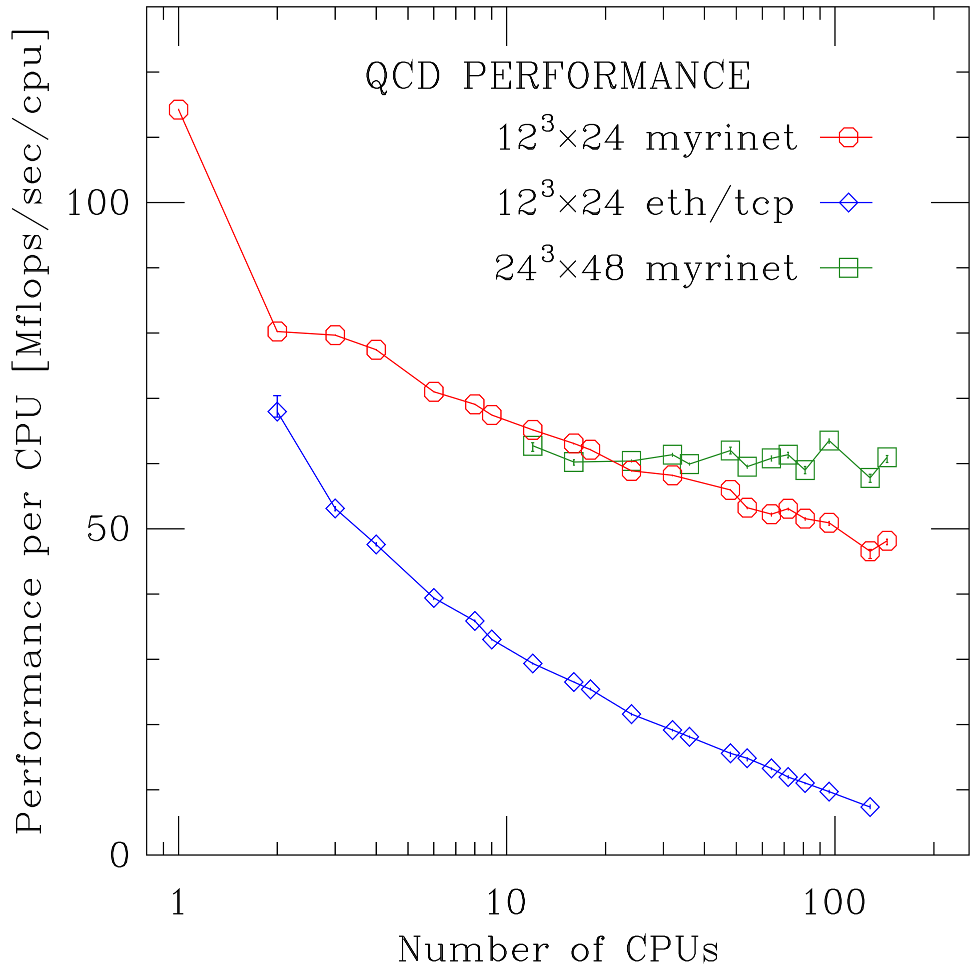
- Again, fixed volume (12^3 X 24) per run
- Performance per node is shown
- Surface area to volume ratio increases with node count



Cluster Performance

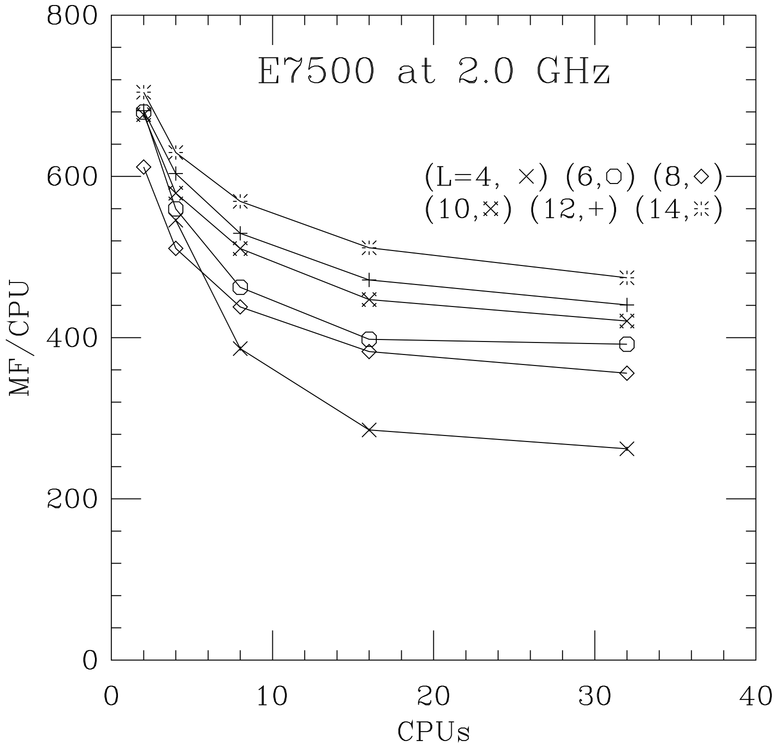
Fast Ethernet vs Myrinet

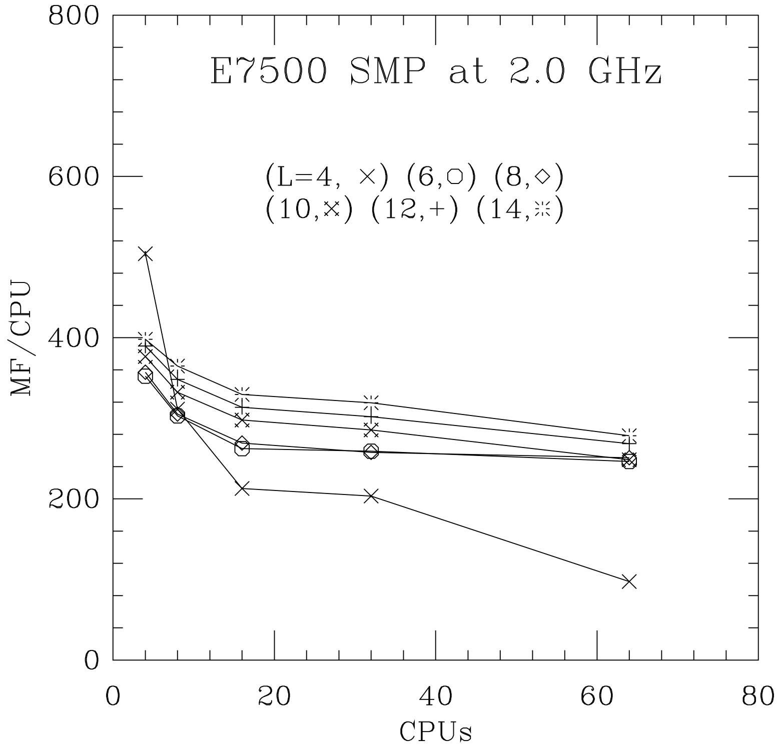
- On our Pentium III cluster, Fast Ethernet was clearly inadequate as an interconnect
- Network bandwidth demands increase with single node performance



Cluster Performance

Scaling - Constant Volume per Processor





- MILC runs with fixed lattice volume per node (L^4, L = 4,8,10,12,14)
- Number of communications directions increases with node count: For Non-SMP, 2 nodes = 1 direction, 4 nodes = 2 directions, 8 nodes = 3 directions, 16 nodes = 4 directions

Cluster Performance

PCI Performance of Common PIII/P4/Xeon Motherboards

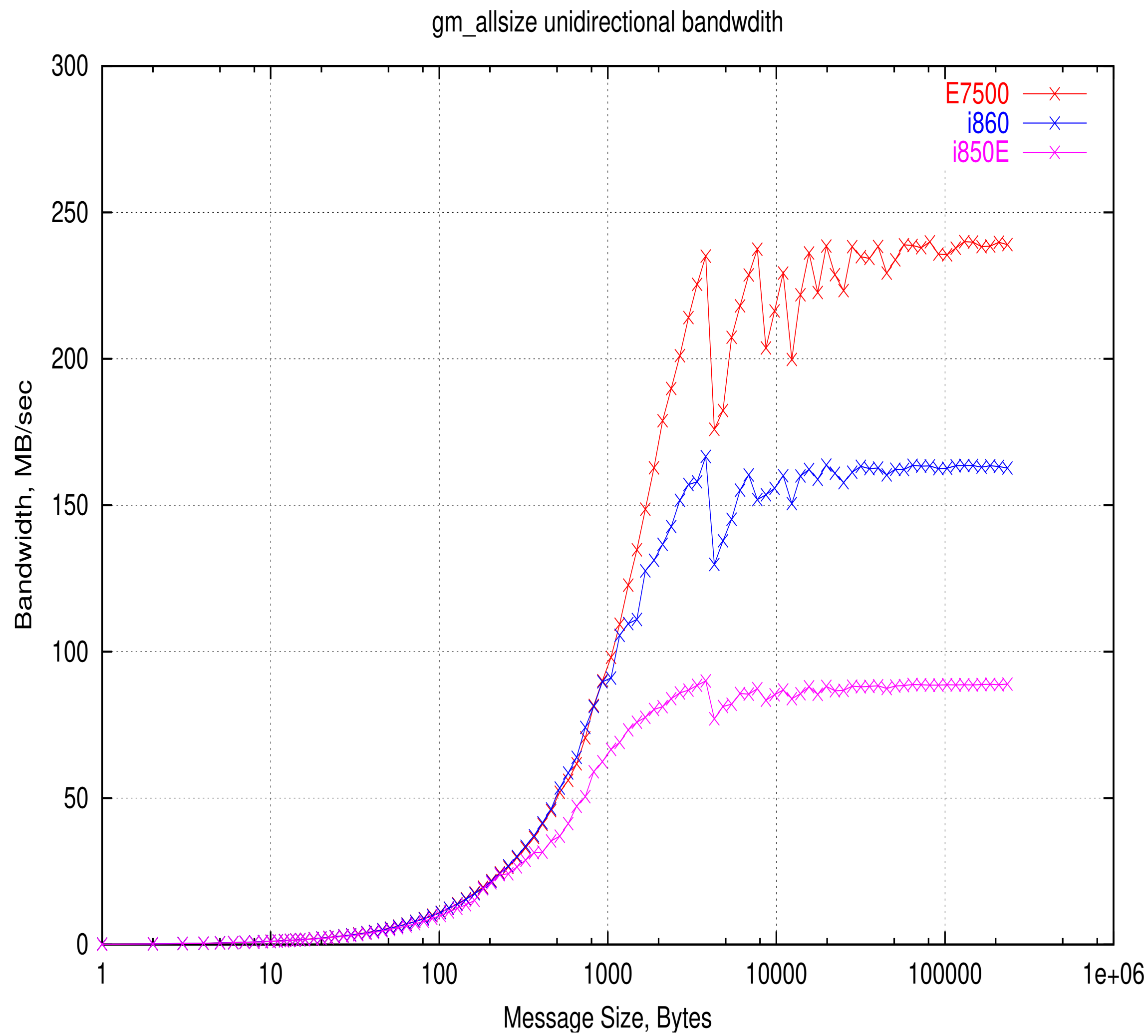
Processor	Chipset	Bus Read (MByte/sec)	Bus Write (MByte/sec)
2.26 GHz Pentium 4	i850E	100	128
700 MHz Pentium III	440GX	125	127
1.7 GHz Xeon	i860	219	294
1.7 GHz Xeon	E7500	422	477
2.0 GHz Xeon	E7500	423	476

- Shown are burst transfer rates between motherboard and NIC as measured by Myrinet GM driver
- Pentium III and Pentium 4 motherboards in table have 32-bit, 33 MHz PCI buses (theoretical maximum transfer rate 133 MB/sec)
- Pentium III motherboards with 64/66 PCI buses are available (theoretical maximum transfer rate 533 MB/sec)
- i860 motherboards with two 64/66 PCI slots must be tweaked to achieve values shown
- E7500 values in red were measured with CPU slowed to 1.7 GHz from normal 2.0 GHz
- Myrinet “wire rate” is 250 MBytes/sec - performance will be constrained on motherboards with PCI transfer rates below 250 MBytes/sec

Cluster Performance

Myrinet Performance - gm_allsize

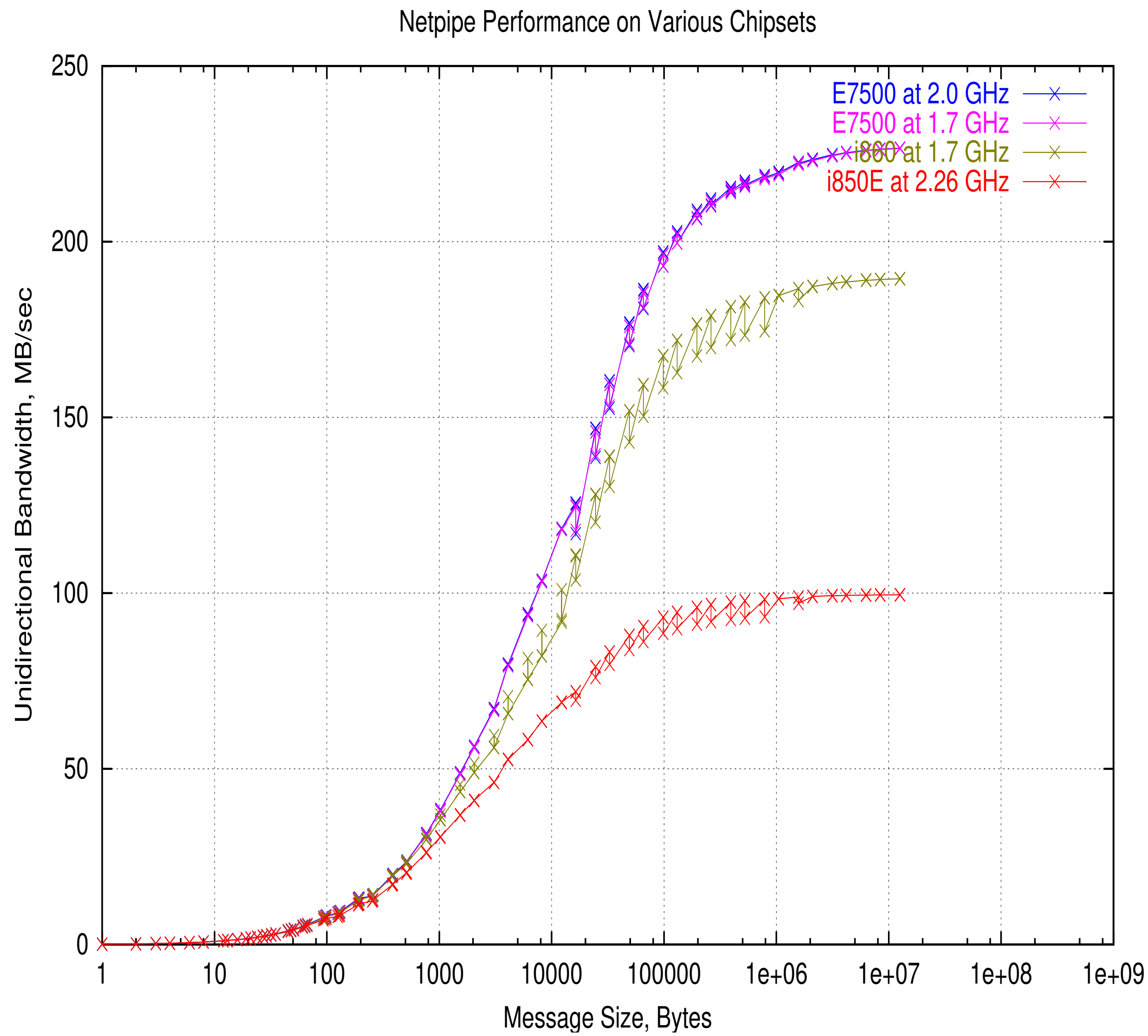
- GM is the standard Myrinet communications software
- gm_allsize benchmark supplied by Myricom
- Graph shows unidirectional bandwidth vs message size
- Measurement uses "ping-pong" exchange - message is sent to target, which sends it back
- Both i860 and i850E data were taken with E7500 computer as the other system



Cluster Performance

Myrinet Performance - Netpipe - ftp://ftp.scl.ameslab.gov/pub/netpipe

- MILC uses MPI for message passing
- The Netpipe benchmark shows “ping-pong” message exchange performance using MPI over GM
- Slower approach to maximum transfer rates, compared to “bare” GM
- E7500 2.0 GHz and E7500 1.7 GHz curves are on top of each other - no degradation in performance caused by slower CPU
- Full duplex rates are about 20% less in each direction



Cluster Performance

Estimating Performance of Slower Systems

- How important is achievable bandwidth?
- Latest Fermilab cluster has 2.0 GHz Xeon DDR systems.
  - To estimate other clusters, substitute one node
  - Frequent barrier sync's cause cluster to run at the speed of the slowest node
- Substitutions for 2.0 GHz E7500 node:

  E7500 at 1.7 GHz

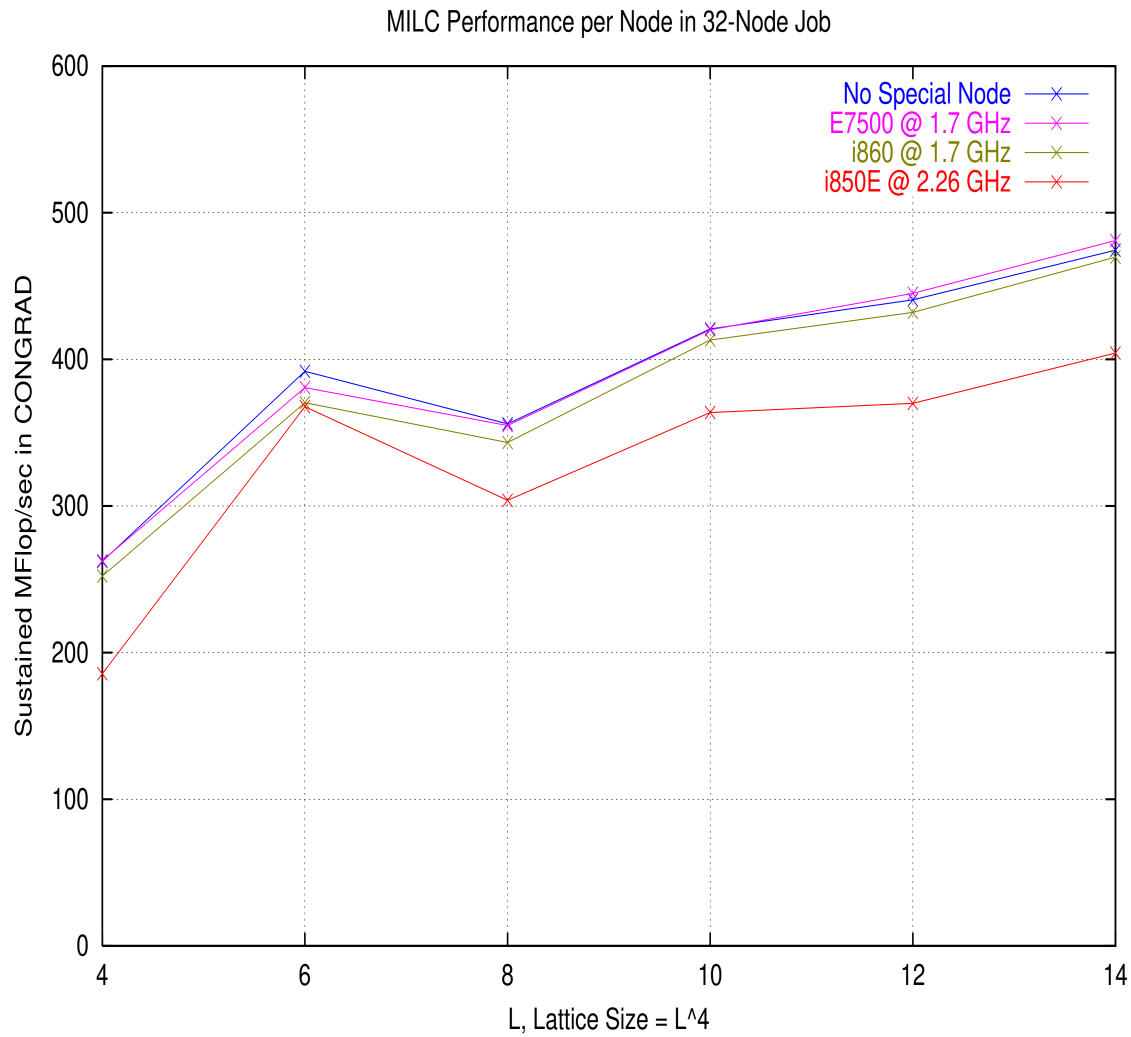
  i860 at 1.7 GHz (RDRAM, poor 64/66 PCI)

  i850E at 2.26 GHz (RDRAM at 533 MHz, poor 32/33 PCI)

Cluster Performance

Estimating Performance of Slower Systems - 32 Nodes

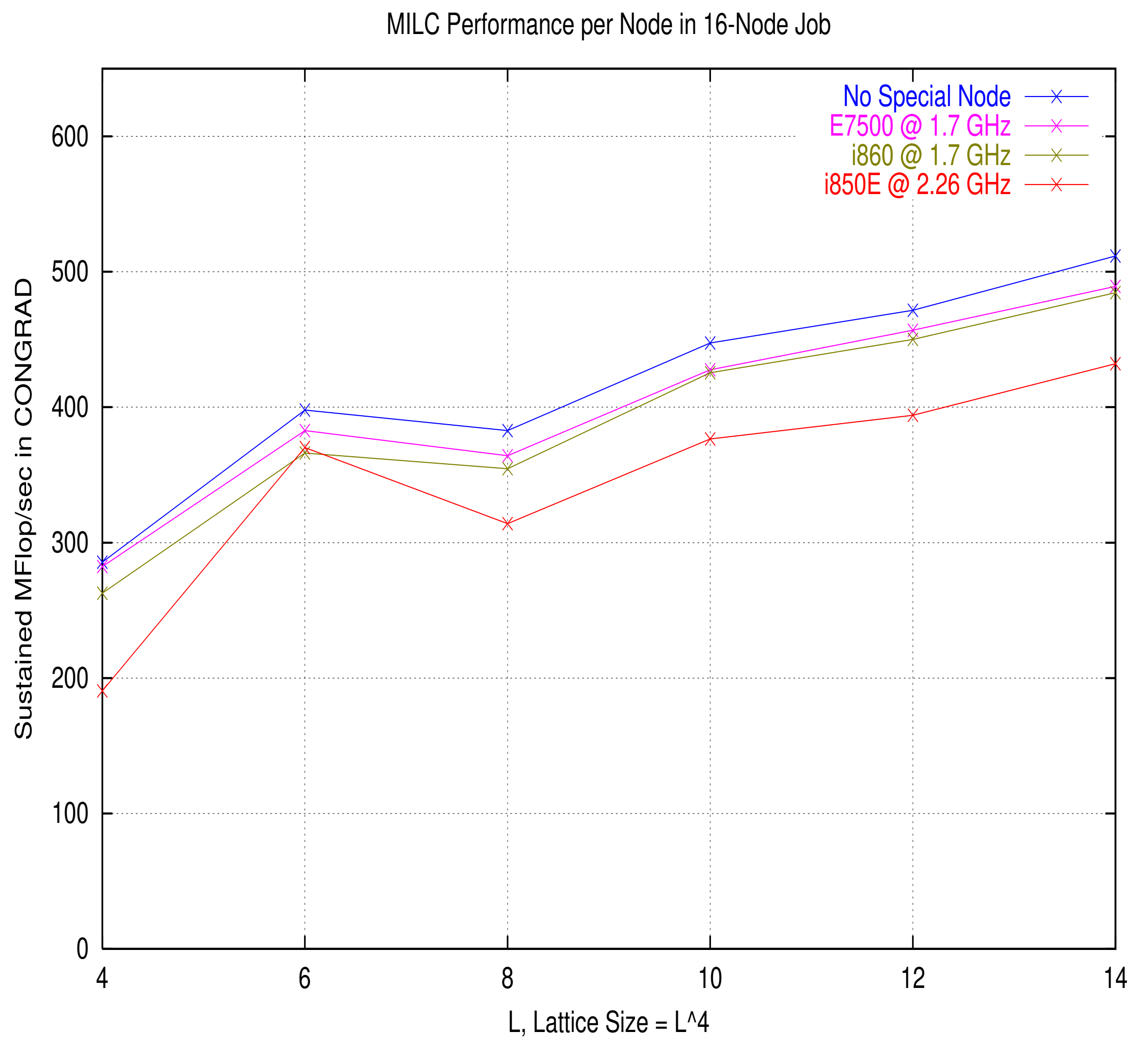
- Performance measured using constant volume per node
- Data shown for 32 nodes, non-SMP
- i860 shows little degradation compared to E7500
- Substantial degradation for i850E
- Each node communicates in 4 directions



Cluster Performance

Estimating Performance of Slower Systems - 16 Nodes

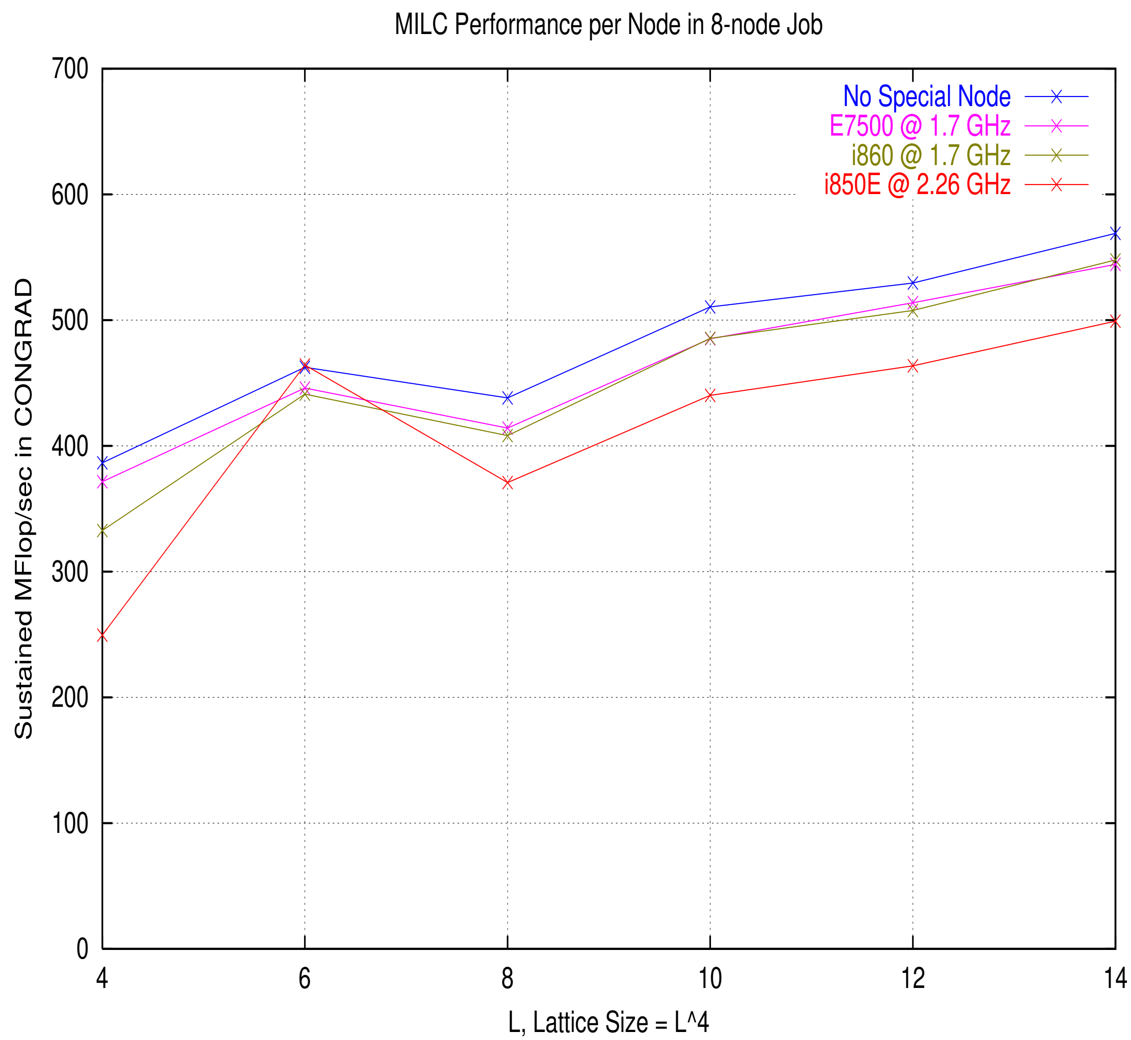
- Data for 16 nodes
- Very similar to 32 node behaviour
- Each node communicates in 4 directions



Cluster Performance

Estimating Performance of Slower Systems - 8 Nodes

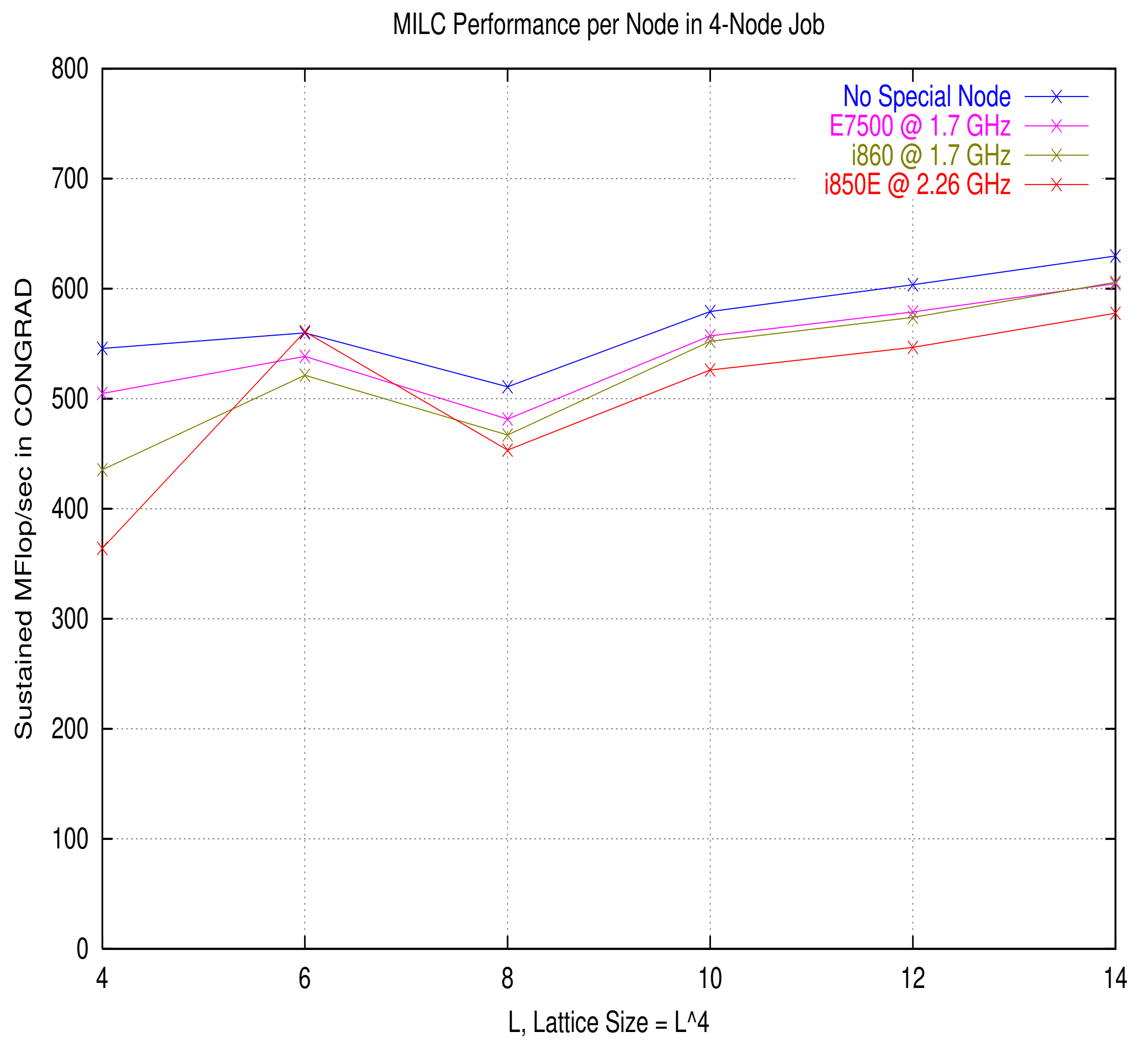
- Data for 8 nodes
- Penalty decreasing for i850E's 32/33 PCI bus
- Each node communicates in 3 directions



Cluster Performance

Estimating Performance of Slower Systems - 4 Nodes

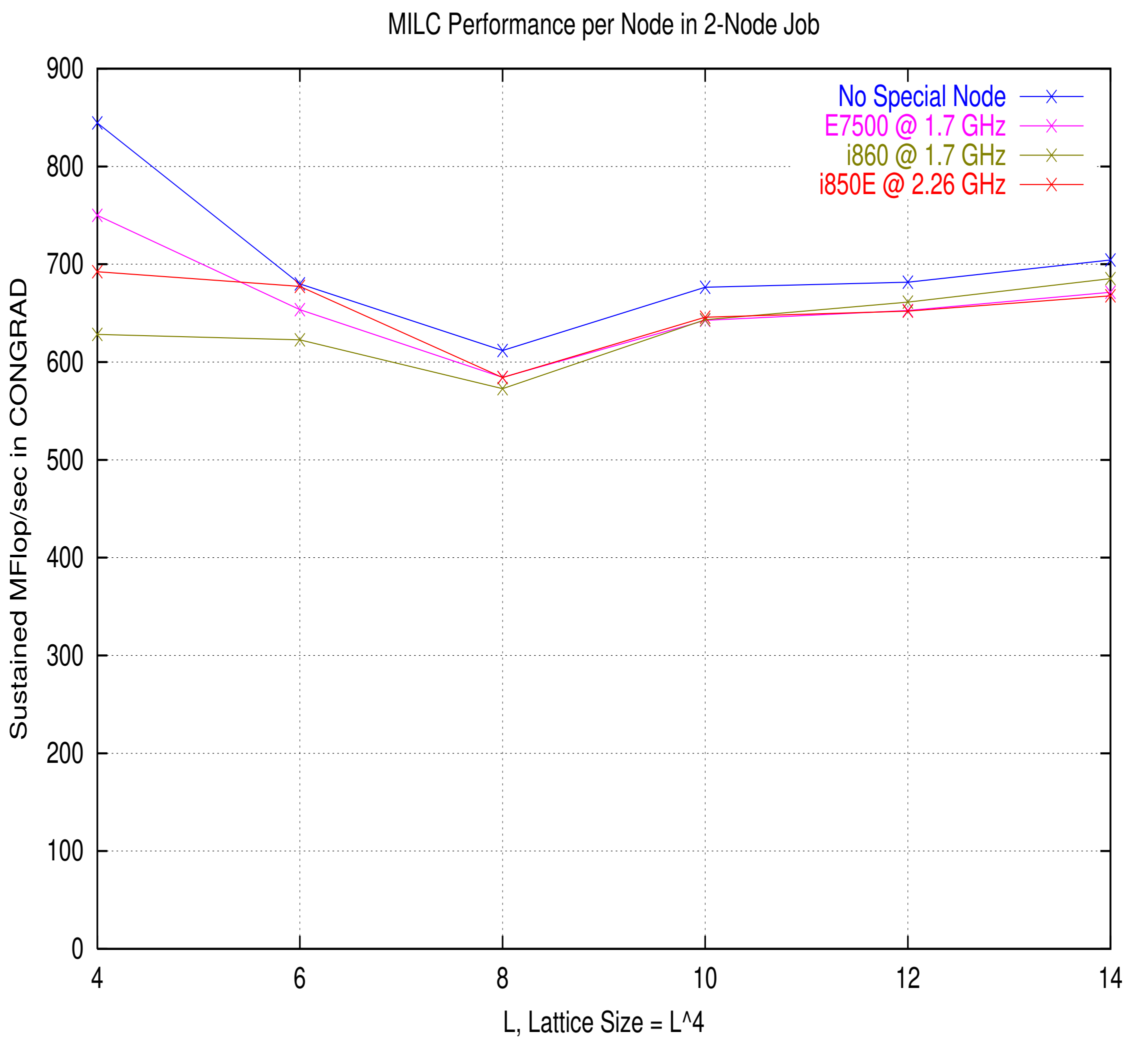
- Data for 4 nodes
- Each node communicates in 2 directions



Cluster Performance

Estimating Performance of Slower Systems - 2 Nodes

- Data for 2 nodes
- Each node communicates in 1 direction



Cluster Performance

Estimating Performance of Slower Systems - Summary

- Fast, wide PCI implementation on E7500 (and GC-LE/GC-HE) motherboards is superior to i860 implementation
  - Netpipe maximum on E7500 is 225 MB/sec, only 190 MB/sec on i860
  - Netpipe maximum on i850E is 100 MB/sec (narrow, slow PCI bus)
- However, MILC performance on i860 is very close to performance on E7500 (a few MFlop/sec per node out of 400+ MFlop/sec)
  - MILC performance on i850E is poor in comparison, particularly with large node counts (80 MFlop/sec difference at 14^4 on 32 nodes)

Cluster Performance

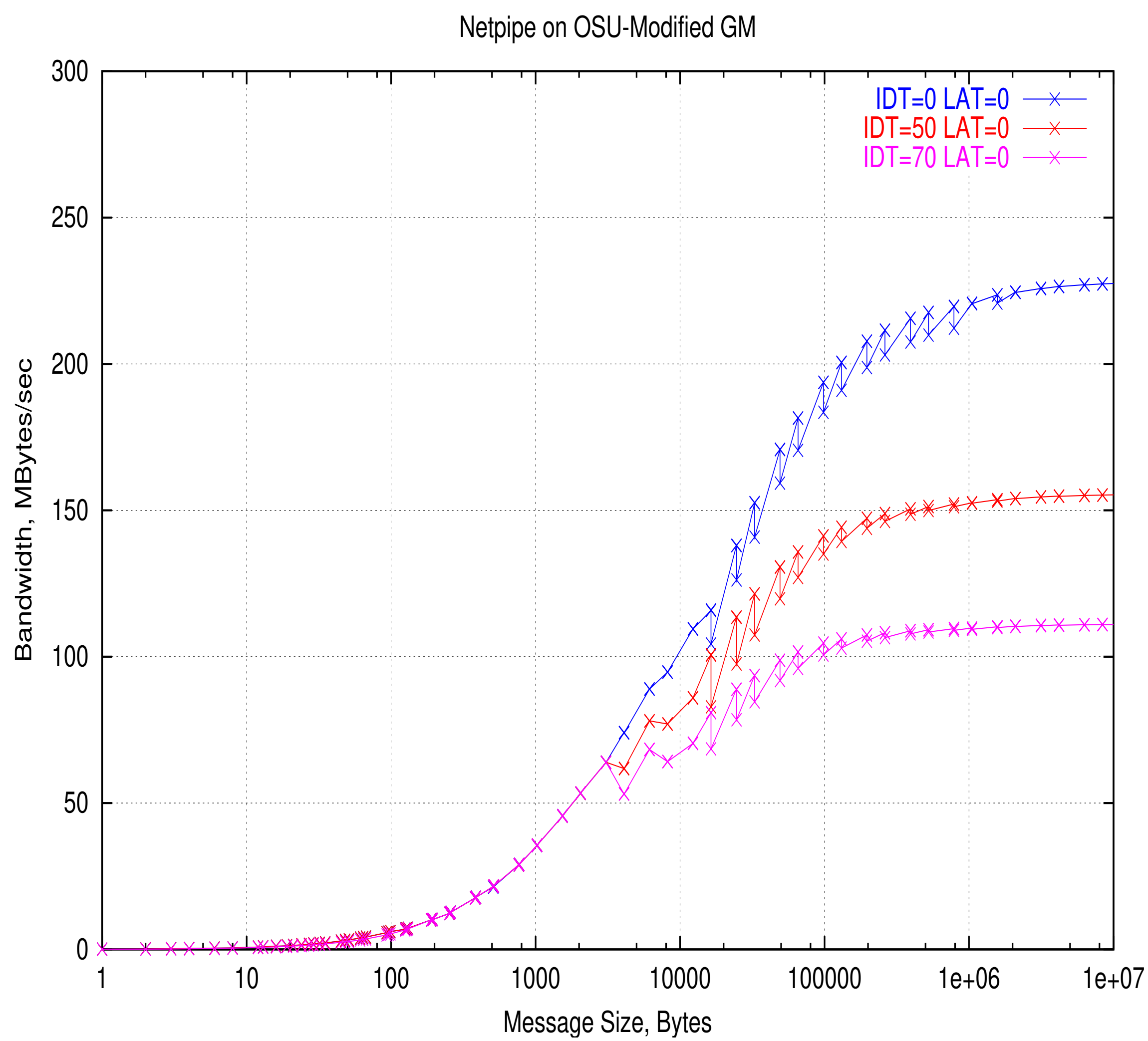
Modifying GM to Vary Latency

- How important are network latency and bandwidth?
- Pr. D.K. Panda at OSU has a QOS version of GM
  - Bandwidth throttling control per connection
  - Throttling works by injecting inter-packet delay in network interface
  - Myrinet uses 4K packets
  - All delay is in network hardware, with no effect on host CPU
- Fermilab modifications
  - Extra delay before first packet to control latency
  - Interface from MPICH-GM
- Preliminary results discussed here

Cluster Performance

Netpipe Validation of Bandwidth Control

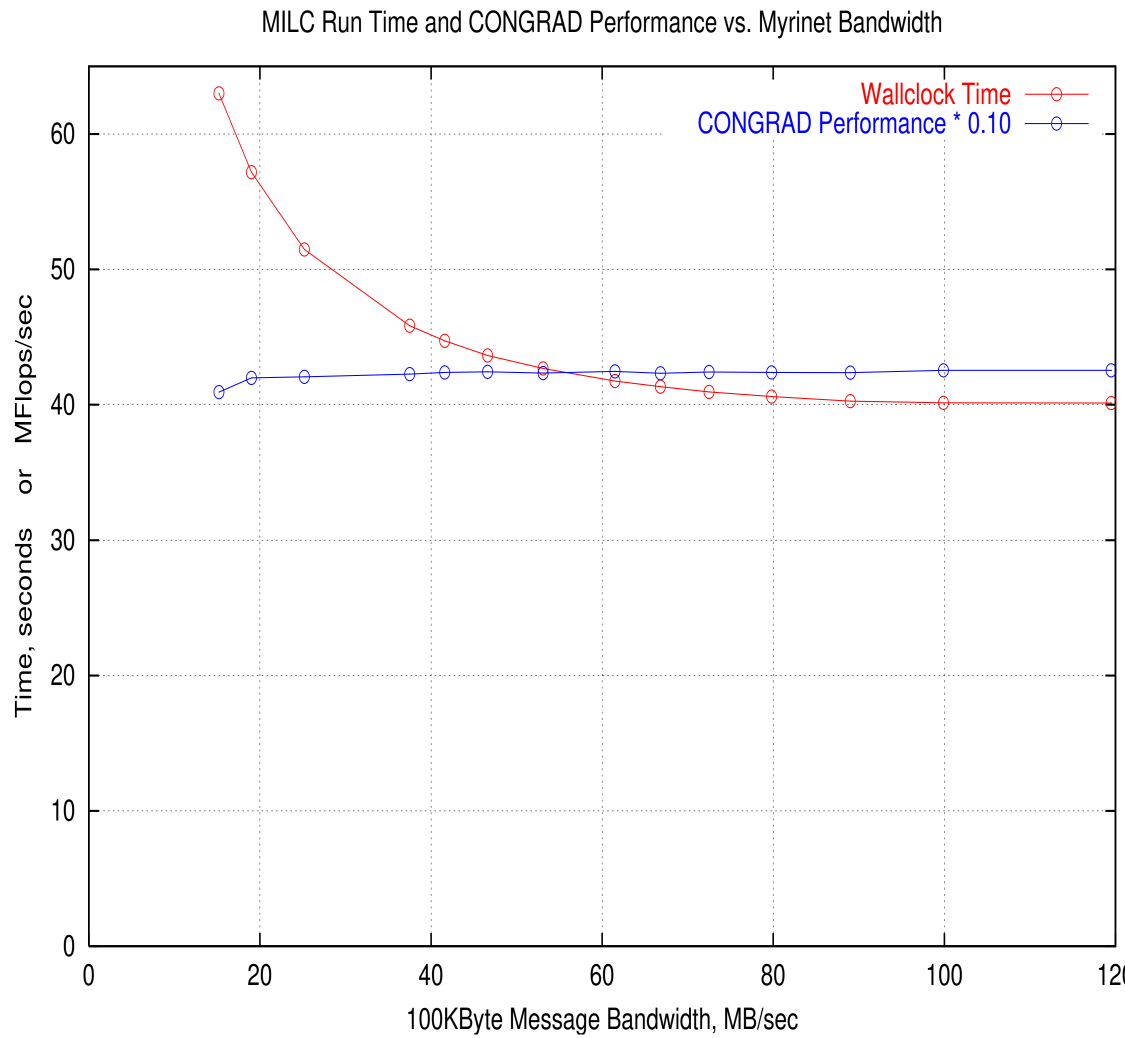
- Data for 3 values of inter-packet delay
- No noticeable effect on small messages (single packet)
- However, there is an effect on zero-length latency
- With current implementation, can't vary bandwidth without affecting latency



Cluster Performance

Netpipe: Bandwidth Affect on MILC Performance

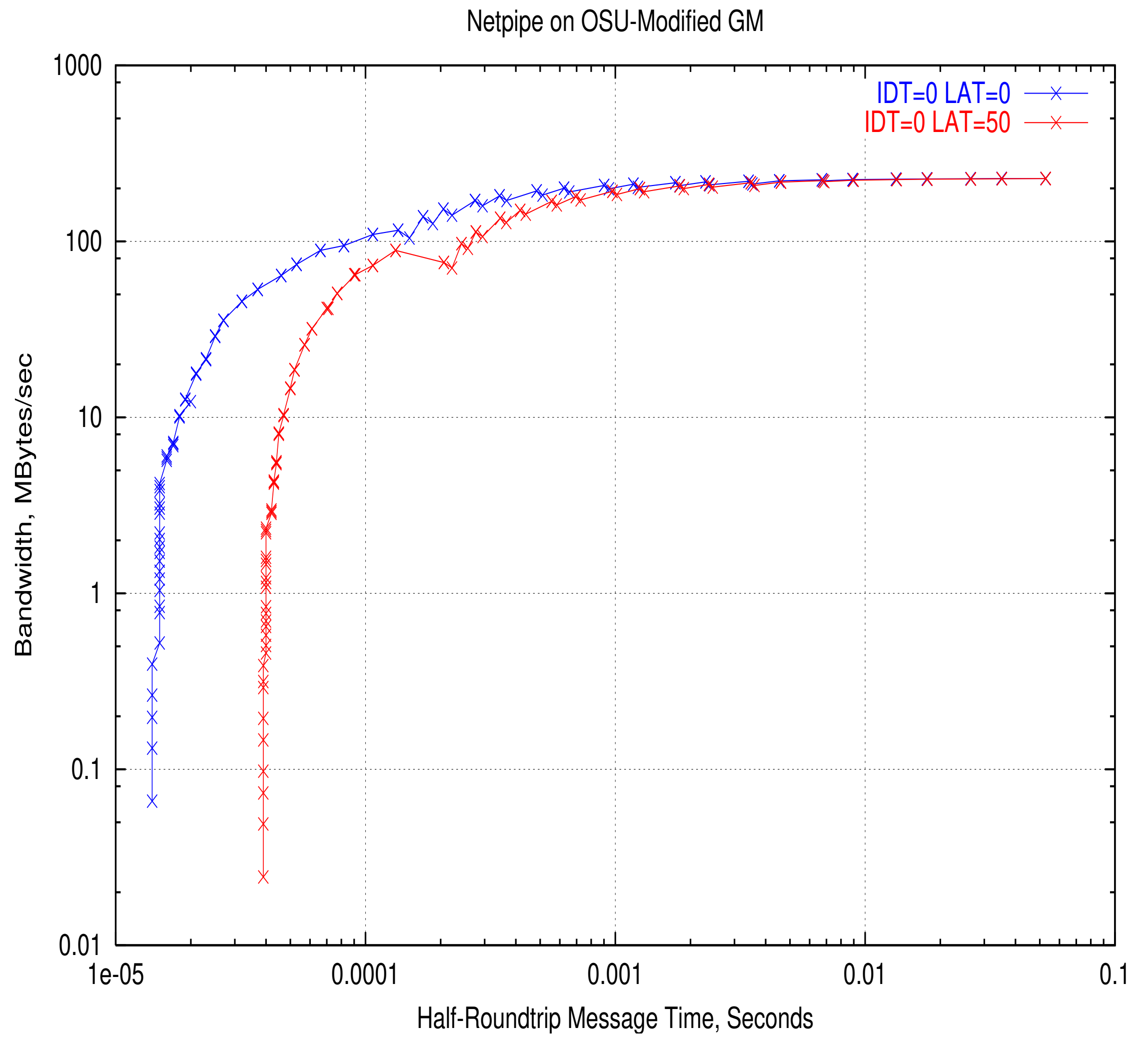
- Data for 16 nodes, 12^4
- Unmodified GM 100K message bandwidth = 209 MB/sec
- Essentially no change in CONGRAD performance
- Weak change in overall MILC runtime (includes measurement of observables)
- Confirms data taken with slow (i860) PCI node



Cluster Performance

Netpipe Validation of Latency Control

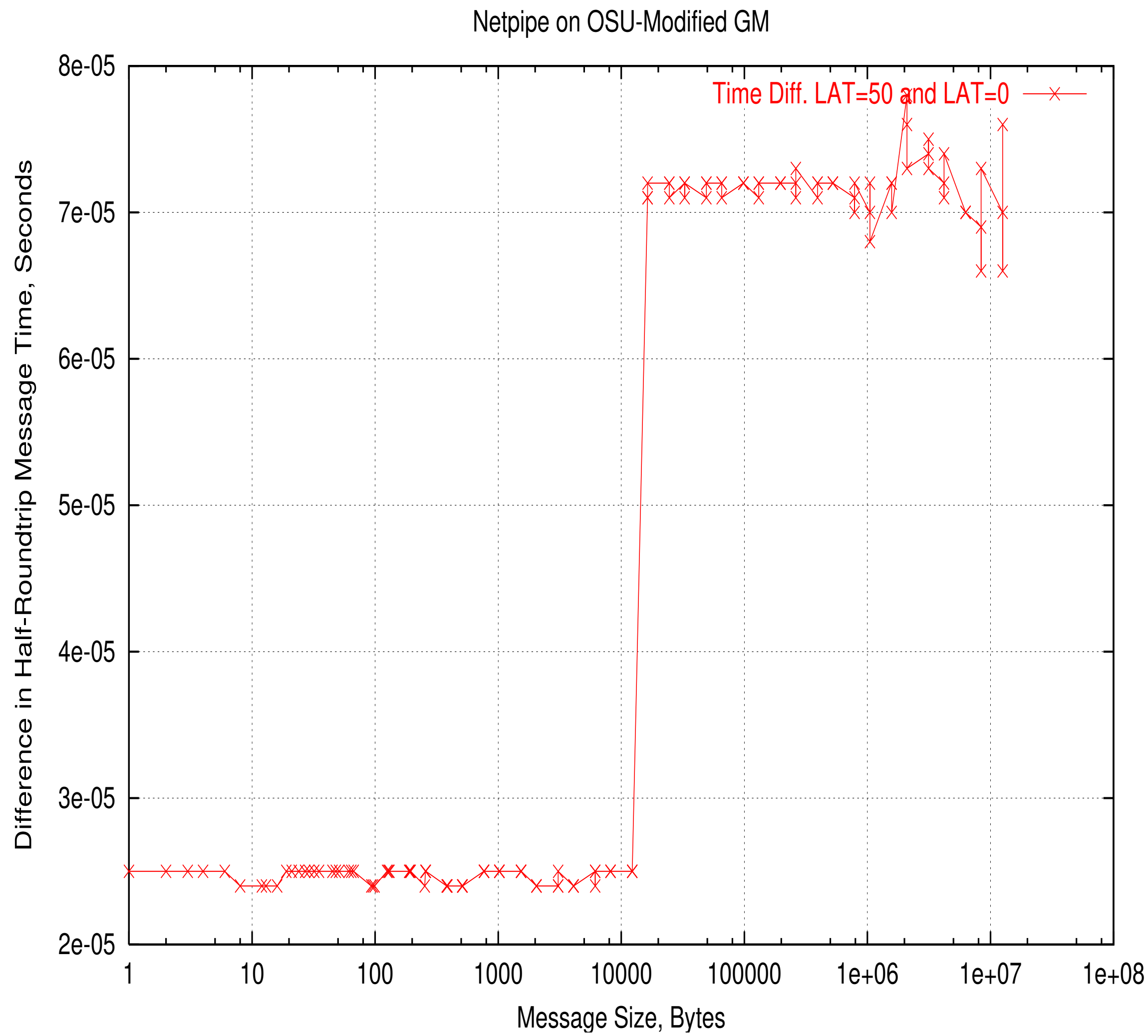
- Data for 2 values of initial packet delay
- Netpipe data plotted showing bandwidth versus message time
- Varying first packet latency does not affect long message bandwidth
- Latency without GM modifications = 9.5 microsec
- Minimum latency with GM modifications = 14 microsec
- An interesting result comes from plotting difference in corresponding message times...



Cluster Performance

Netpipe Validation of Latency Control - Eager-Rendezvous Threshold

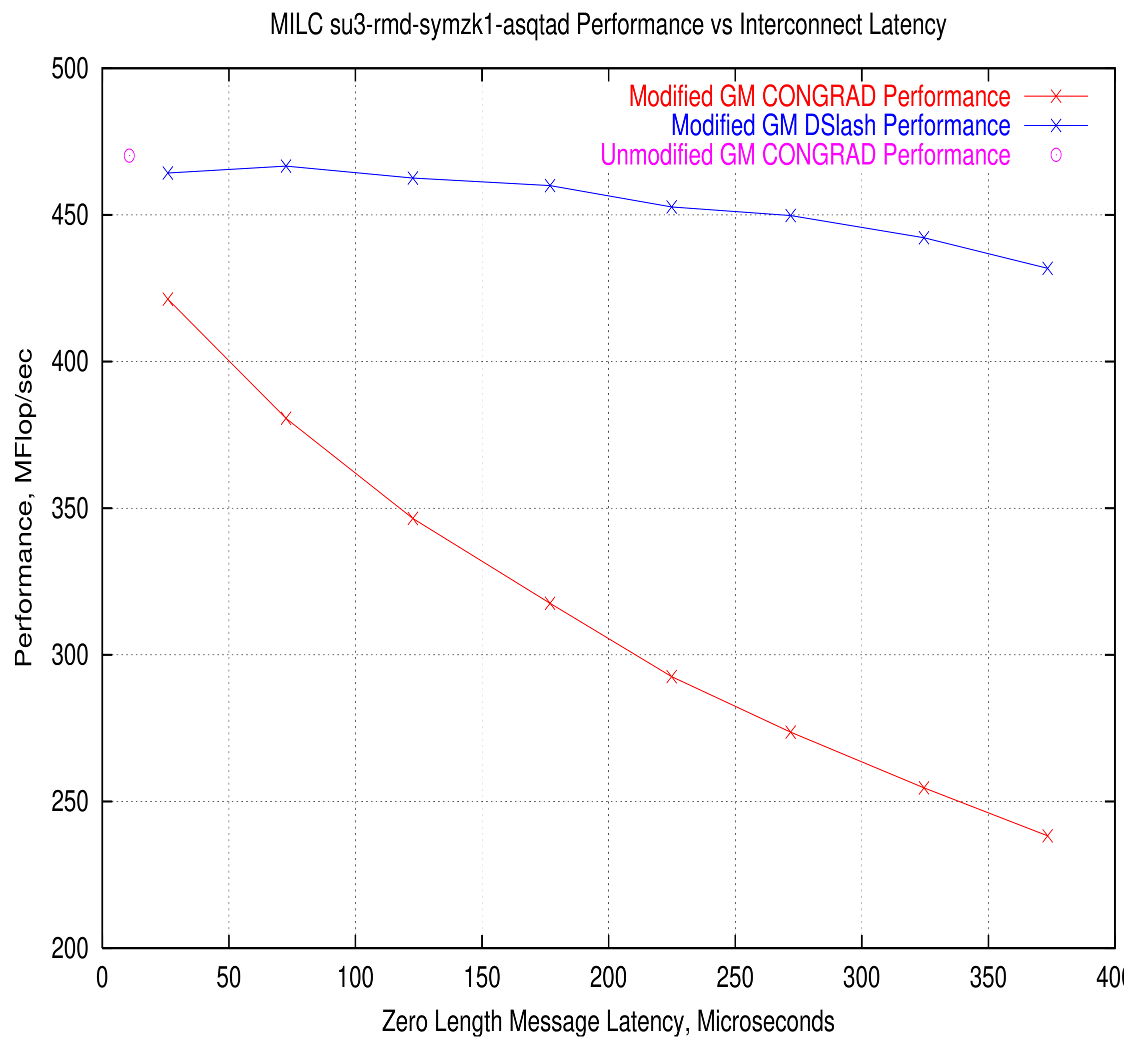
- Plotting differences in corresponding message times shows difference is multiplied by 3 for larger messages
- This effect is caused by MPI “Eager-Rendezvous” setting
  - Short messages are sent without warning, stored in temporary buffers
  - Long messages are sent after initial communication of message length
  - Long messages require 3 messages, so 3X the time difference is observed



Cluster Performance

MILC Sensitivity to Latency

- Data for 16 nodes, 12^4
- Single point is MILC performance with unmodified GM
- CONGRAD performance decreases rapidly with increasing latency
- D-slash (CONGRAD without global sums) is not very sensitive to latency



Summary

- Single Node Performance
  - Memory bandwidth dominates performance
  - Field major layouts improve performance by improving cache line efficiency
  - SSE optimizations give similar boost in performance
  - Current top-performing x86 systems are single P4 with 533 MHz FSB
- SMP Performance
  - Poor scaling because of insufficient memory bandwidth
  - Other applications may motivate use of dual nodes
- Cluster Performance
  - Wide variation in raw Myrinet performance because of PCI implementations
  - MILC CONGRAD performance is very sensitive to latency (global sums)
  - MILC CONGRAD performance is relatively insensitive to bandwidth (particularly in the range available in dual Xeon chipsets)

SciDAC Initiative

- U.S. Department of Energy is funding Lattice QCD via the SciDAC program (Scientific Discovery though Advanced Computing - http://www.lqcd.org/
  - 3 years of support (FY2002 - FY2004)
  - Collaboration includes most U.S. lattice theorists
  - Funding primarily for software development, but also prototype clusters
- Strategy
  - Two computer hardware approaches: special purpose machines (QCDOC at Columbia/Brookhaven) and commodity clusters at Fermilab and Jefferson Lab
  - Create software infrastructure to allow legacy and new LQCD codes to run on both types of hardware
  - Anticipate best performance/price on QCDOC until 2004-2005, clusters afterwards
  - If either approach fails, switch all efforts to other approach
- Software
  - QMP - communications library, basically a small subset of MPI with less overhead, will run over QCDOC mesh, Myrinet GM, MPI
  - QLA - single node linear algebra, lattice aware, optimized for some architectures
  - QDP - lattice-wide computations
  - Optimized inverters
- Hardware
  - Investigations of gigabit ethernet mesh (Z. Fodor approach) and custom network