Load Balancing and Topology Aware Mapping for Petascale Machines

Abhinav S Bhatele

Parallel Programming Laboratory, Illinois

Outline

- Scalable Load Balancing
- Current Supercomputers and their Topologies
- Topology Aware Mapping
  - Motivation
  - Examples
  - Tools and Techniques

Load Balancing

Load Balancing

- Load Imbalance can severely hurt performance
- The time to completion is bound by the slowest processor
- Hence, when dividing 121 units of work among 10 processors
  - 12 units to 9 processors and 13 units to 1 (simple round-robin scheme)
  - 13 units to 9 processors and 4 units to 1 (intelligent blocking)

Load Balancing Strategies

- Static: Achieved through initial data distribution
- Dynamic
  - Blocks of data which can be moved around during the execution
  - Measurement-based
  - Automatic (by the runtime)

Another classification

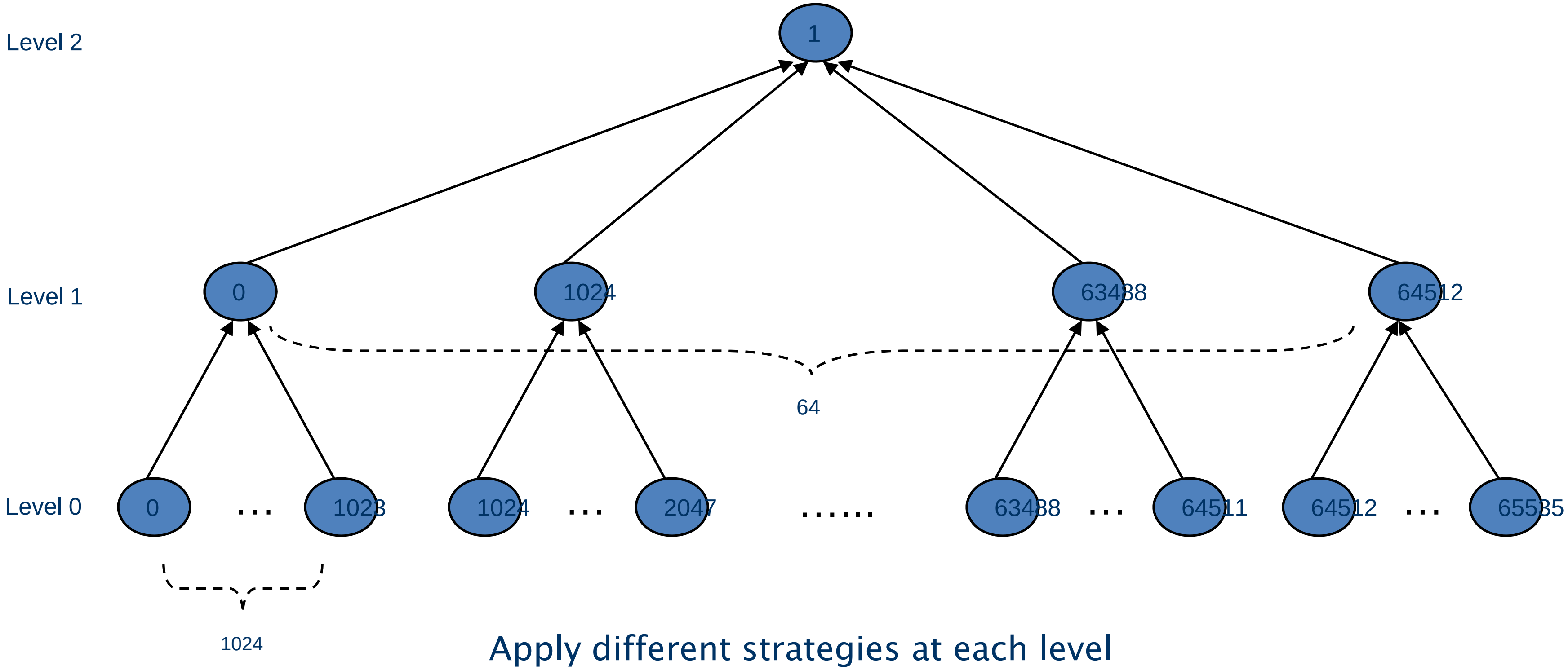
- Balance computational load
- Balance computational as well as communication load
  - Minimize communication volume on the network
- Minimize communication traffic
  - Bytes passing through each link on the network

Scalable Load Balancing

- Centralized Strategies perform better but take longer
  - Memory/communication bottleneck
- Distributed Strategies are fast but have incomplete knowledge
- Hybrid/ Hierarchical Strategies
  - Allow using different strategies at different levels

Hierarchical Strategies

64K processor hierarchical tree



Apply different strategies at each level

References

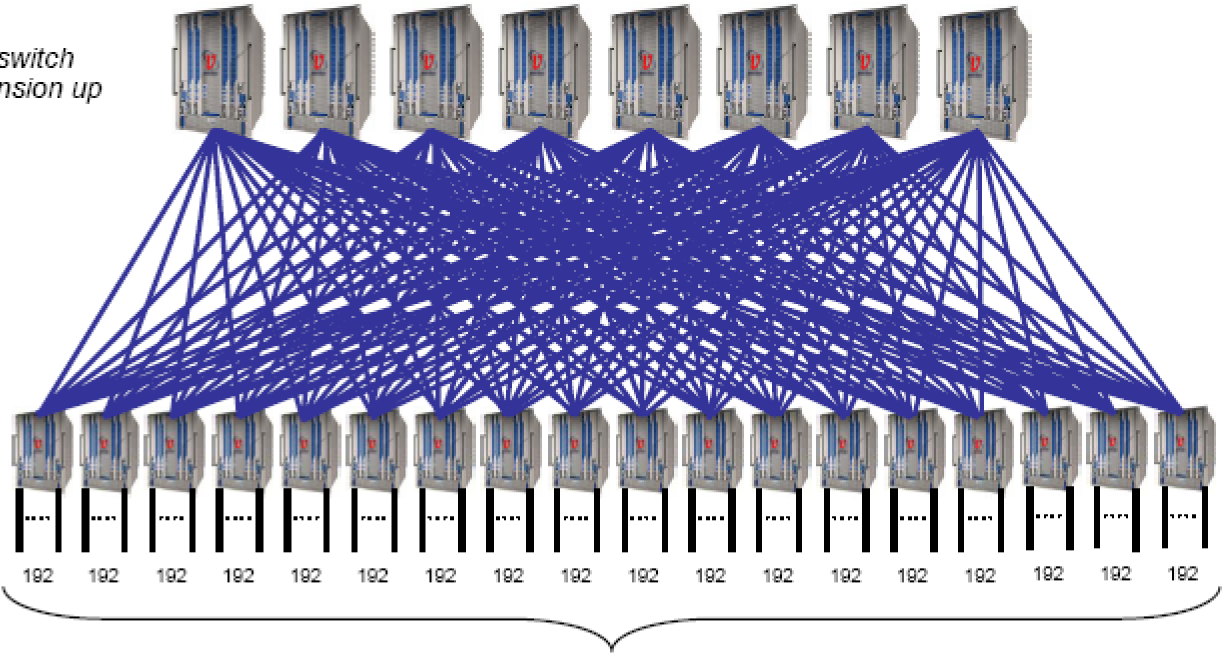
- Trinilos (developed at Sandia): http://trilinos.sandia.gov/
- Charm++ Load Balancing Framework: https://charm.cs.uiuc.edu/research/ldbal/

Topology Aware Mapping

Current Machines and their Topologies

- 3D Mesh – Cray XT3/4/5
- 3D Torus – Blue Gene/L, Blue Gene/P
- Fat-tree, CLOS network – Infiniband, Federation
- Kautz Graph – SiCortex
- Future Topologies?

RoadRunner



Eight 2nd–stage switches

Extra 2nd–stage switch ports allow expansion up to 24 CUs

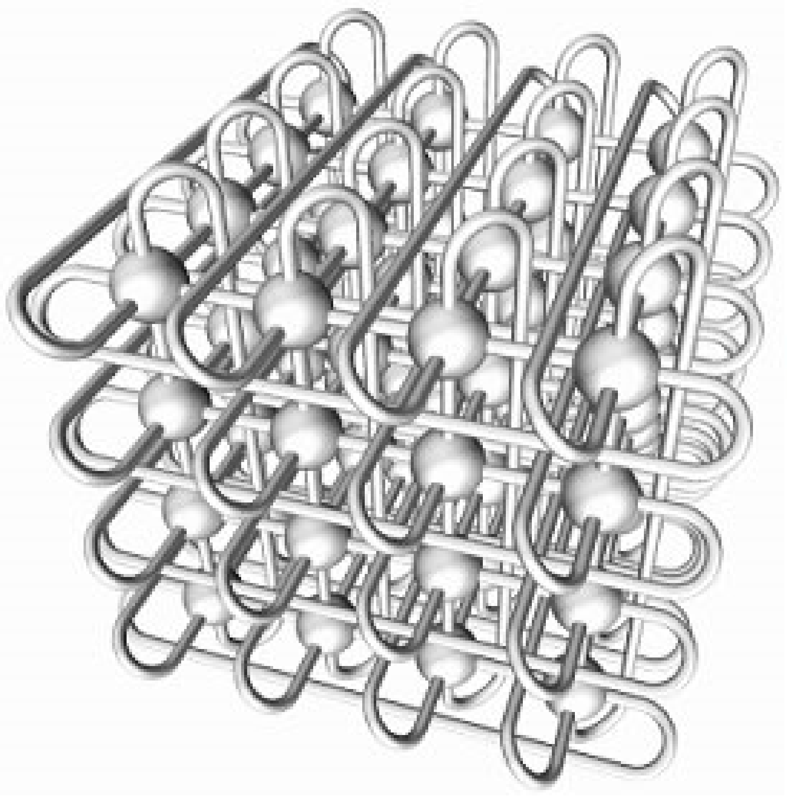
12 uplinks per CU to each 2nd–stage switch provide half-bandwidth

Smaller systems (e.g. 4 CUs) by disabling links to other CUs for improved robustness during standup & stabilization

18 CUs with CU switches, 3456 IB nodes

Blue Gene/P

- Midplane: smallest unit with torus in all dimensions: 8 x 8 x 8
- All bigger partitions formed from this unit of 512 nodes
- Intrepid at ANL: 40960 nodes



Cray XT5

- The full installation is a torus
- Typically the batch scheduler does not allocate cuboidal shapes
- Jaguar (XT4) at ORNL: 8,064 nodes, torus dimensions 21 x 16 x 24
- Jaguarpf (XT5) at ORNL: 21 x 32 x 24

Is topology important?

- For machines as large as these?

  Yes.

- For all applications?

  No.

Motivation

- Consider a 3D mesh/torus interconnect
- Message latencies can be modeled by

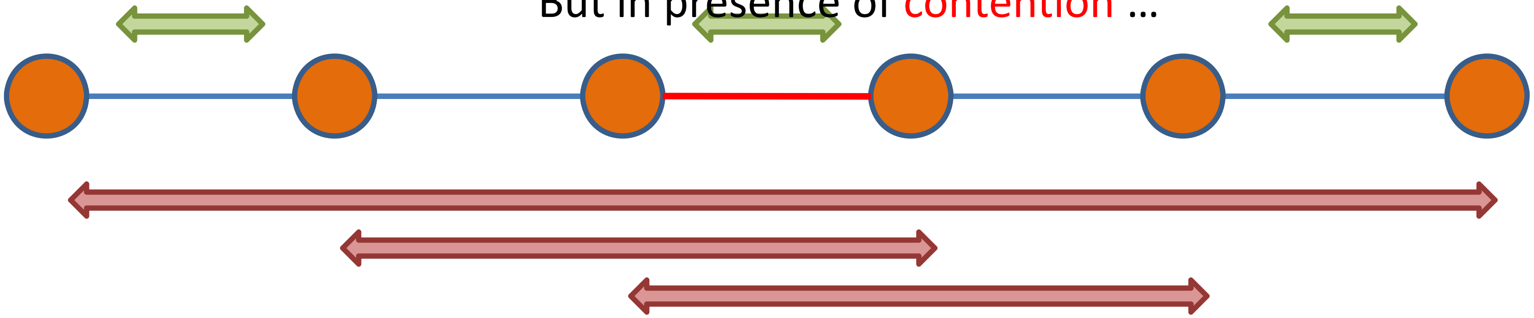
$$(L_f/B) \times D + L/B$$

L_f = length of flit, B = bandwidth,

D = hops, L = message size

When (L_f * D) << L, first term is negligible

But in presence of contention …



Validation through Benchmarks

MPI Benchmarks†

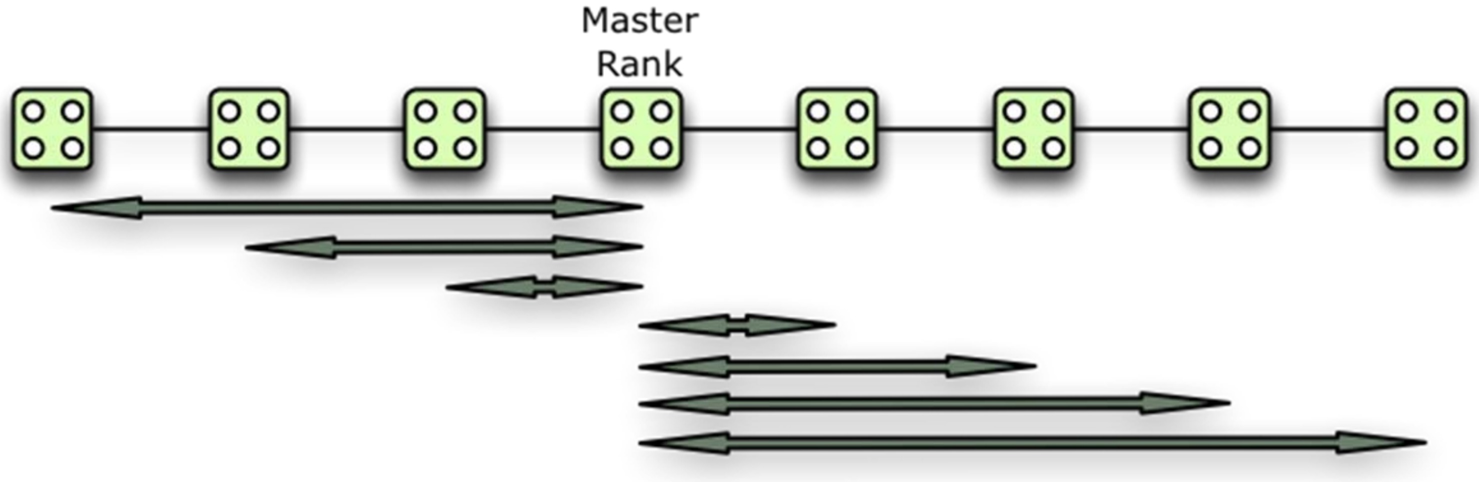
- Quantification of message latencies and dependence on hops
  - No sharing of links (no contention)
  - Sharing of links (with contention)

† http://charm.cs.uiuc.edu/~bhatele/phd/contention.htm

Abhinav Bhatele, Laxmikant V. Kale, **An Evaluation of the Effect of Interconnect Topologies on Message Latencies in Large Supercomputers**, In *Workshop on Large-Scale Parallel Processing (IPDPS)*, 2009.

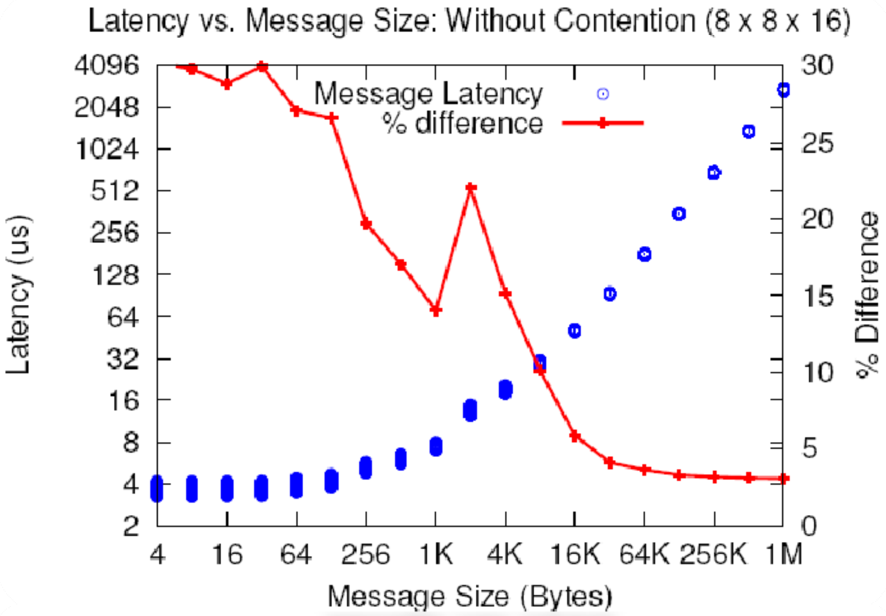
WOCON: No contention

- A master rank sends messages to all other ranks, one at a time (with replies)

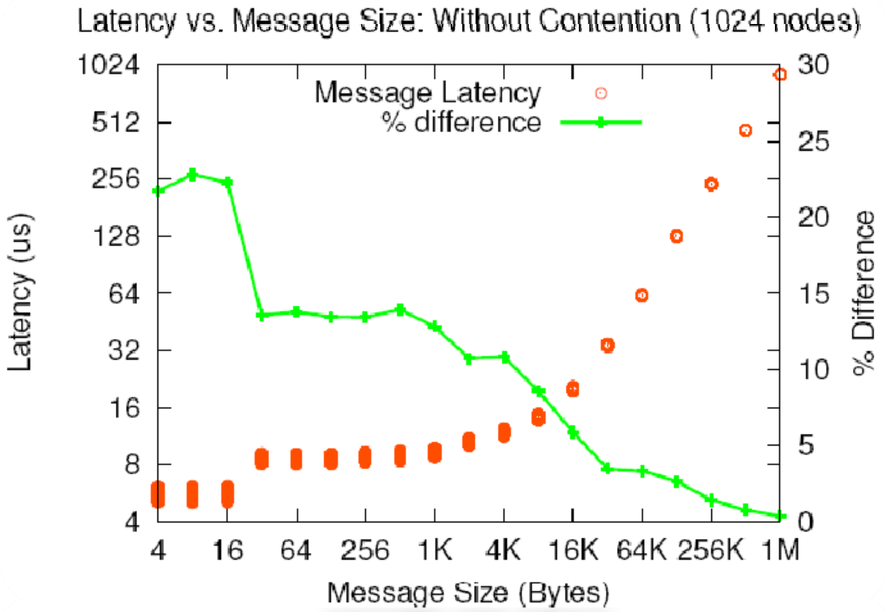


WOCON: Results

$(L_f/B) \times D + L/B$



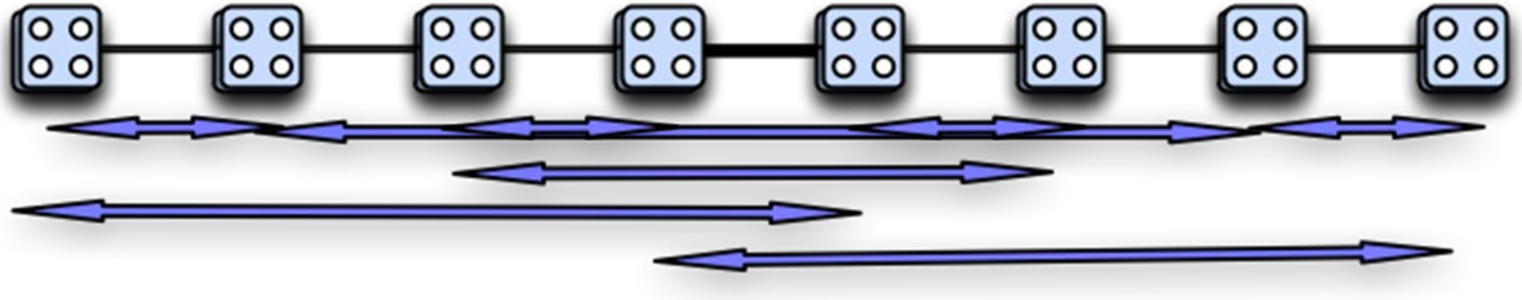
ANL Blue Gene/P



PSC XT3

WICON: With Contention

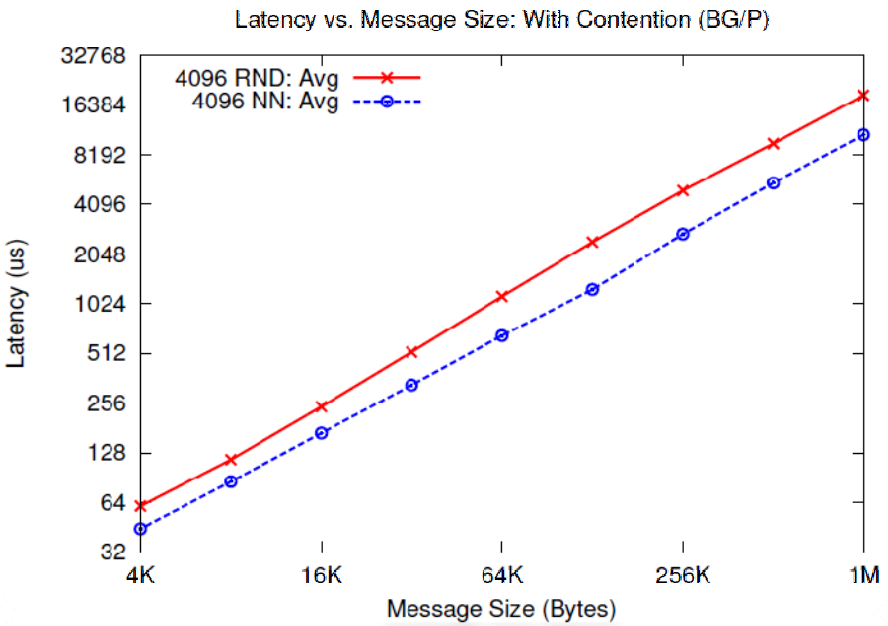
- Divide all ranks into pairs and everyone sends to their respective partner simultaneously



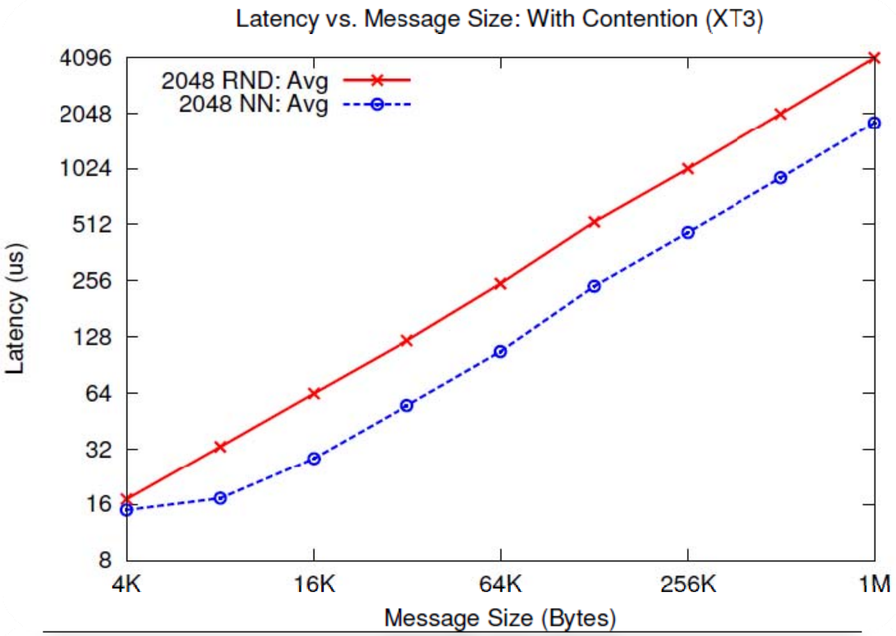
Random: RND

Neighbor: NN

WICON: Results



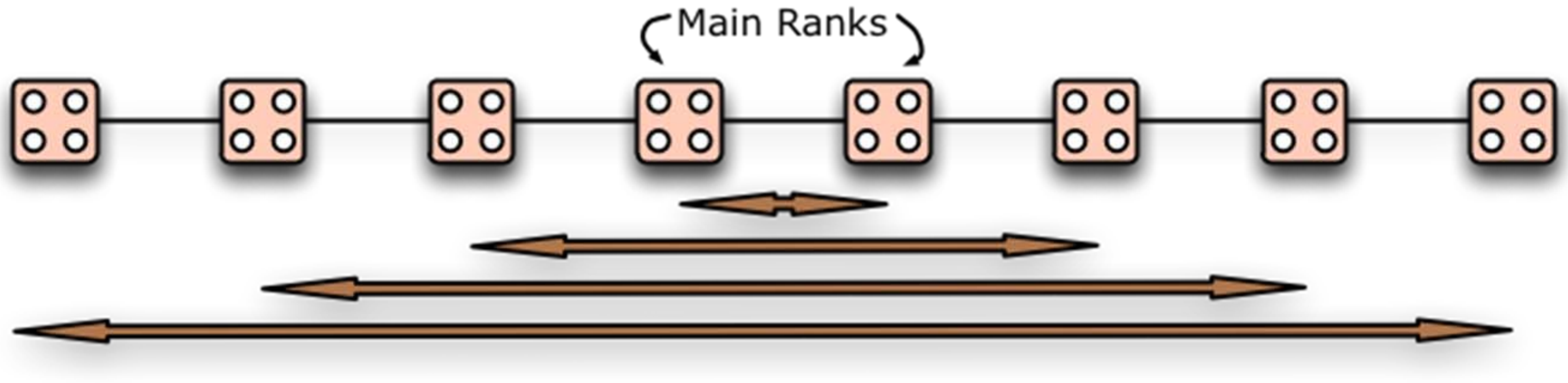
ANL Blue Gene/P

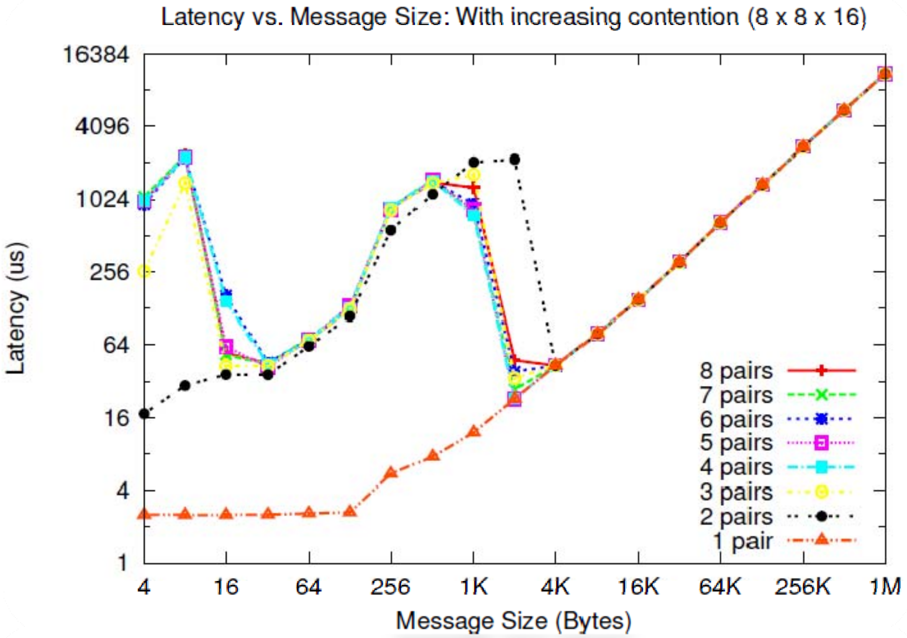


PSC XT3

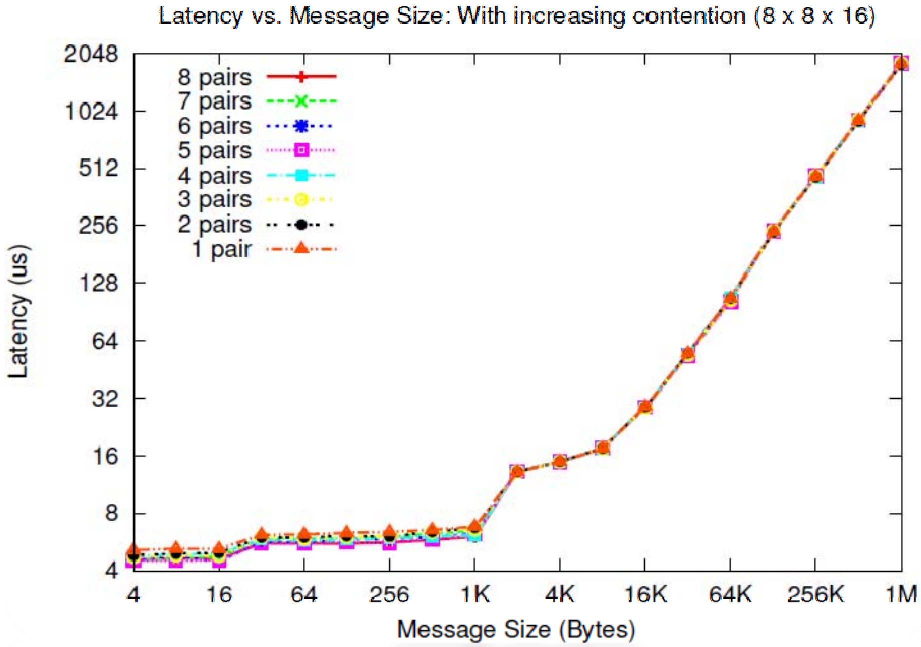
Stressing-a-link Benchmark

- Controlled Congestion
- Quantify contention effects on a particular link





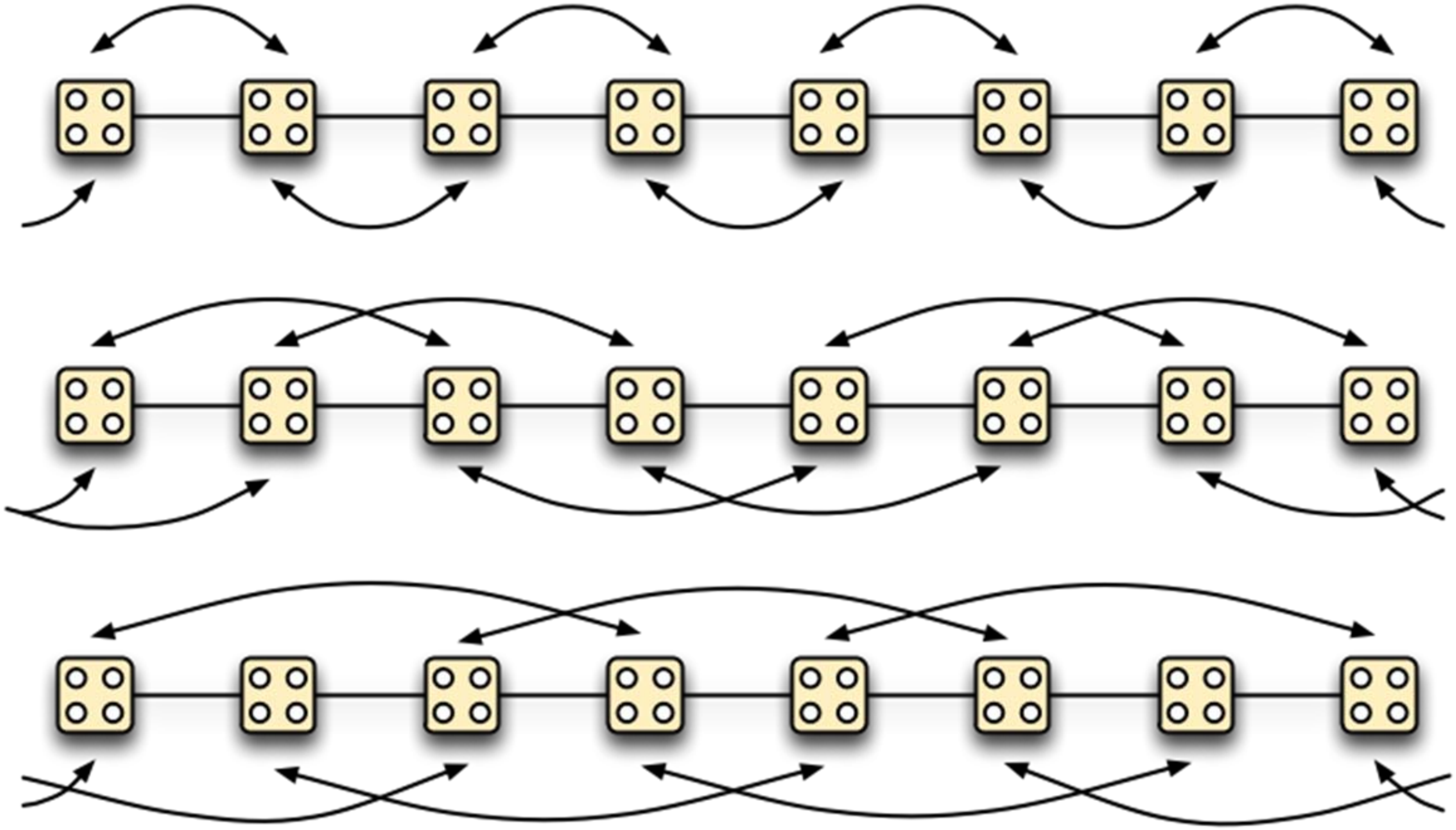
ANL Blue Gene/P



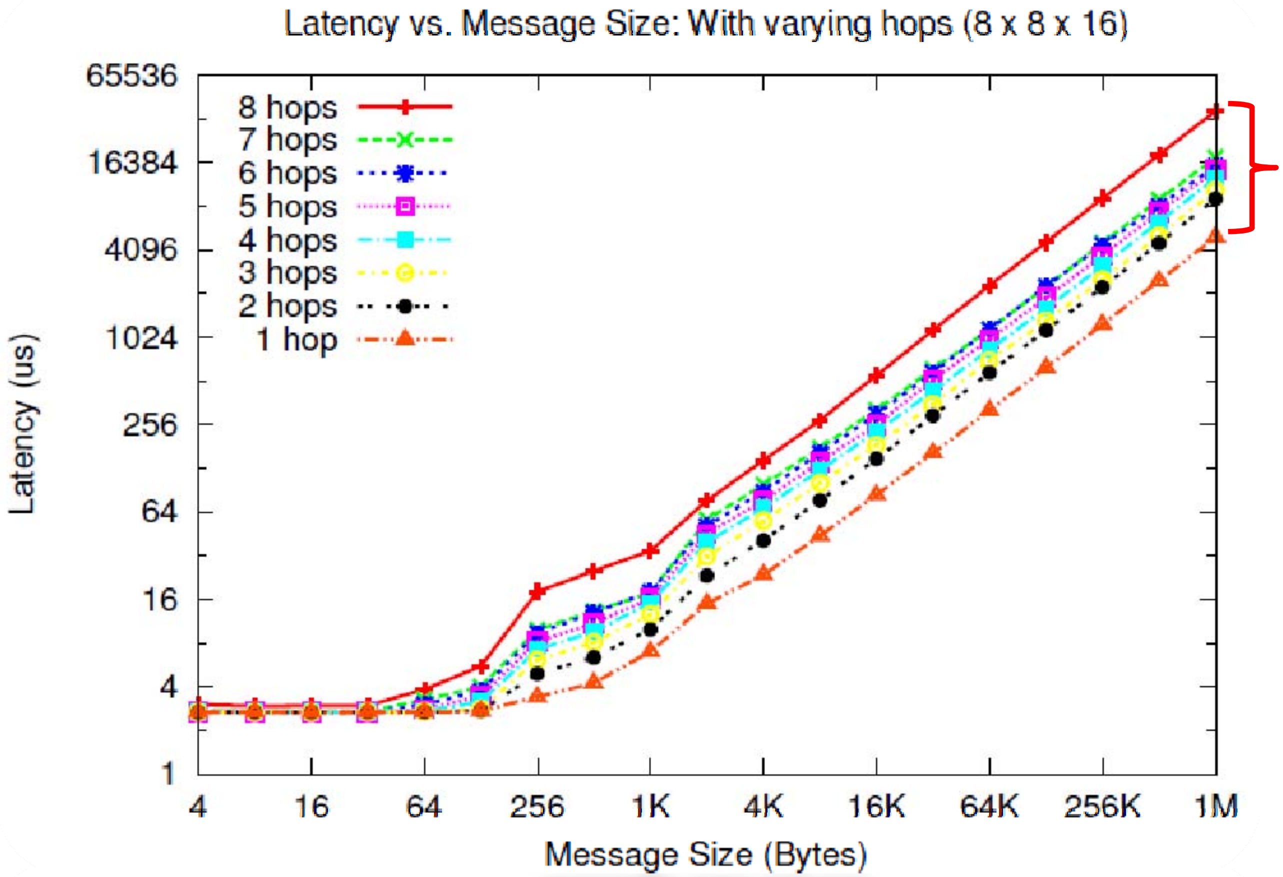
PSC XT3

Equidistant-pairs Benchmark

- Pair each rank with a partner which is ‘n’ hops away

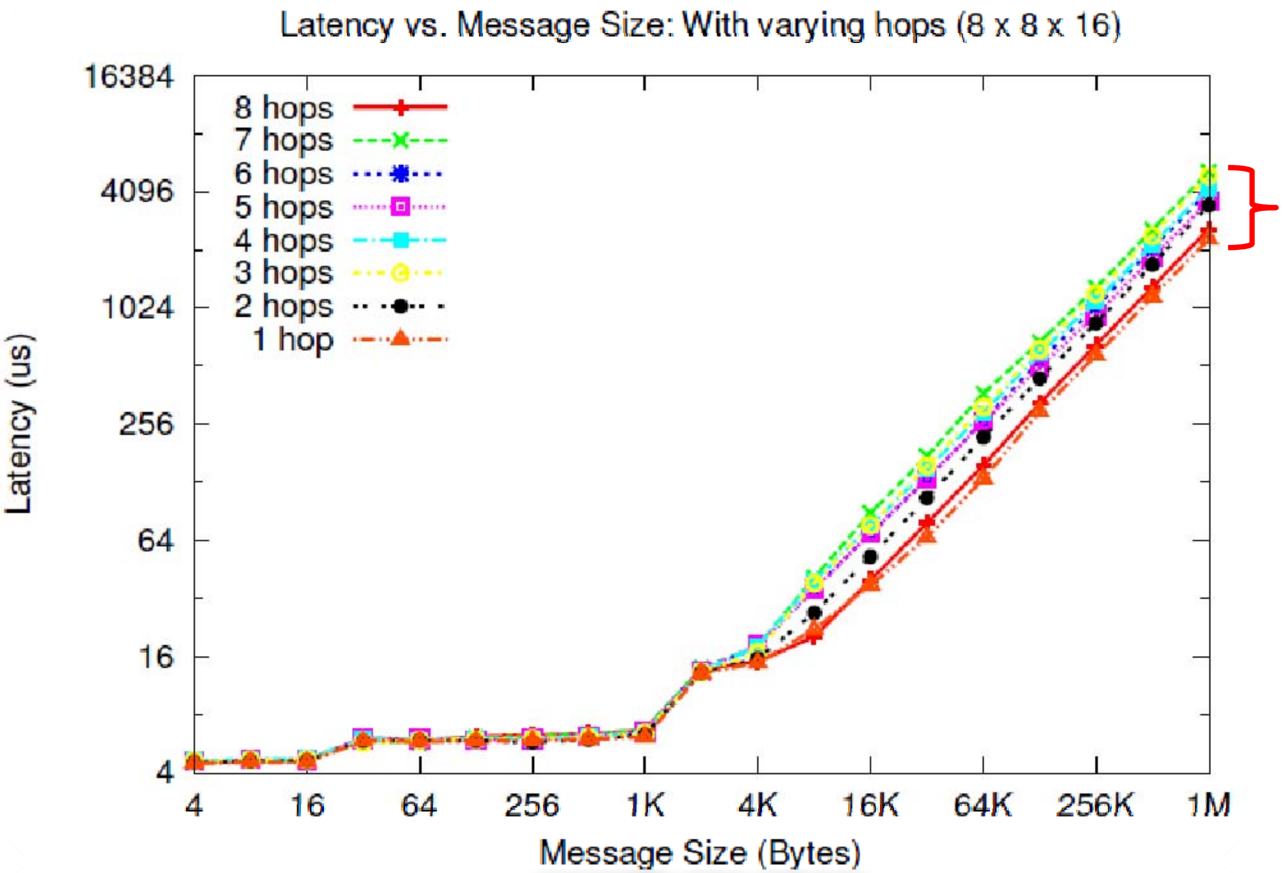


Blue Gene/P



7.39 times

Cray XT3



2.23 times

Can we avoid congestion?

- Yes. If we can minimize the distance each message travels
- Solution: topology aware mapping
  - Initially
  - At runtime (as a part of load balancing)

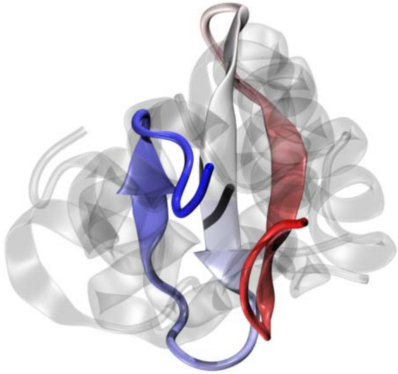
Example Application: NAMD

Molecular Dynamics

- A system of [charged] atoms with bonds
- Use Newtonian Mechanics to find the positions and velocities of atoms
- Each time-step is typically in femto-seconds
- At each time step
  - calculate the forces on all atoms
  - calculate the velocities and move atoms around

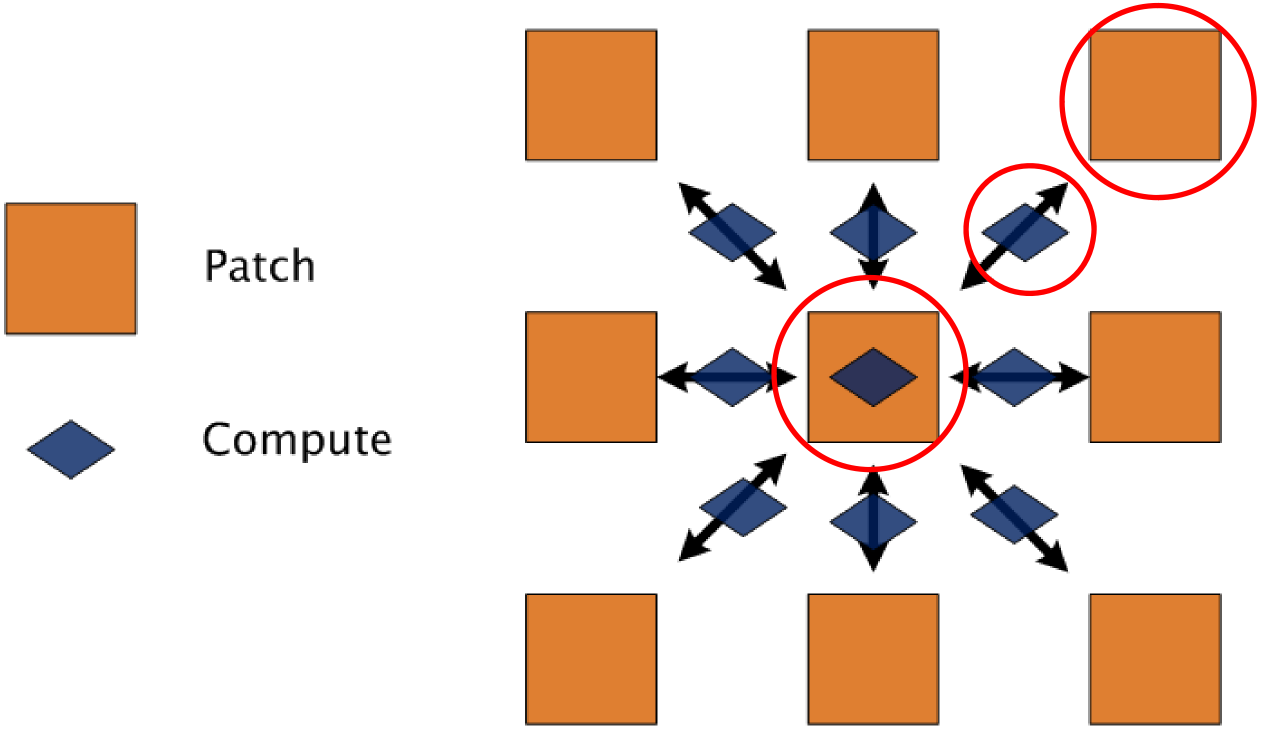
NAMD: NAnoscale Molecular Dynamics

- Naïve force calculation is O(N^2)
- Reduced to O(N logN) by calculating
  - Bonded forces
  - Non-bonded: using a cutoff radius
    - Short-range: calculated every time step
    - Long-range: calculated every fourth time-step (PME)



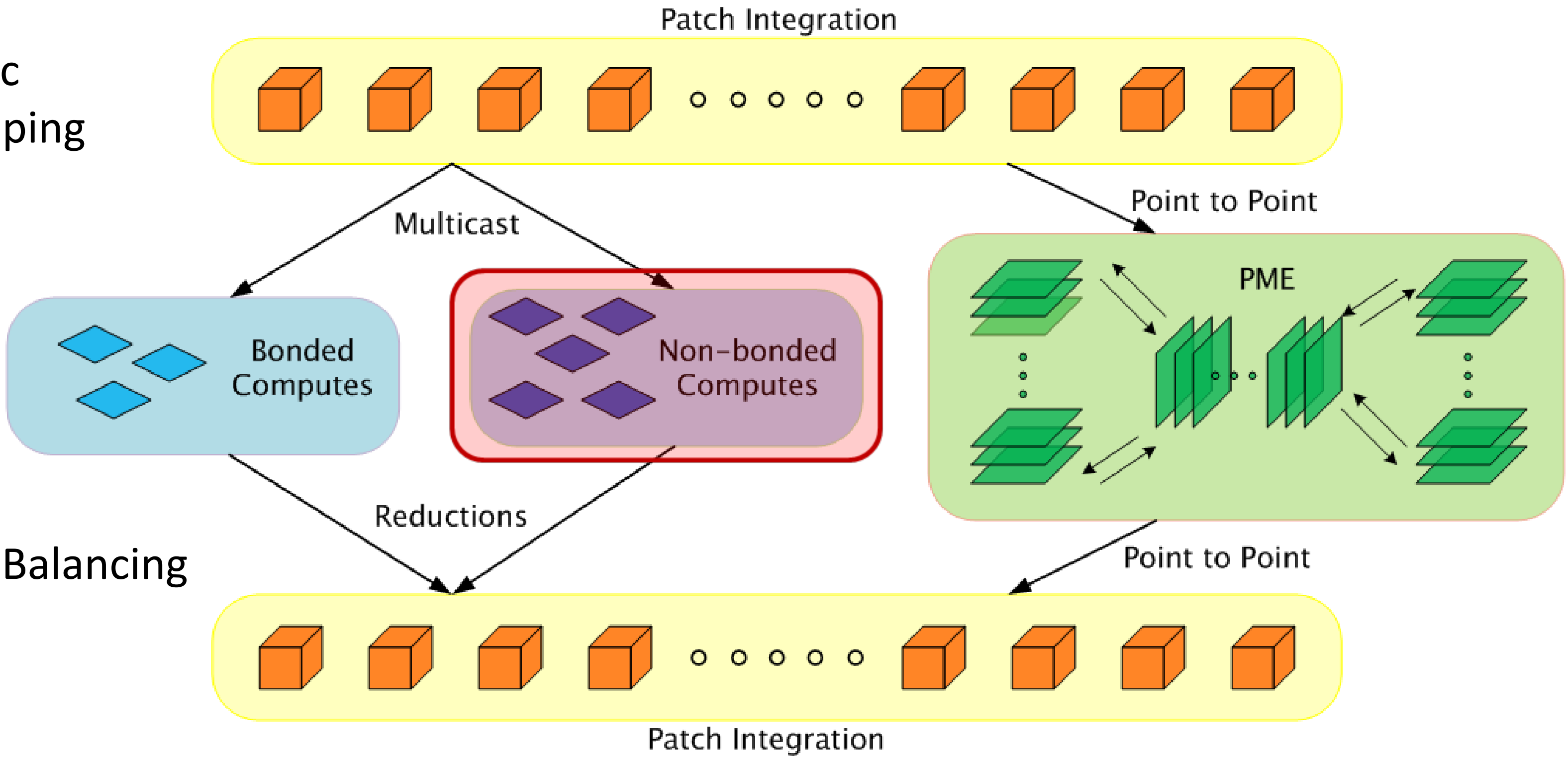
NAMD’s Parallel Design

- Hybrid of spatial and force decomposition



Parallelization using Charm++

Static Mapping

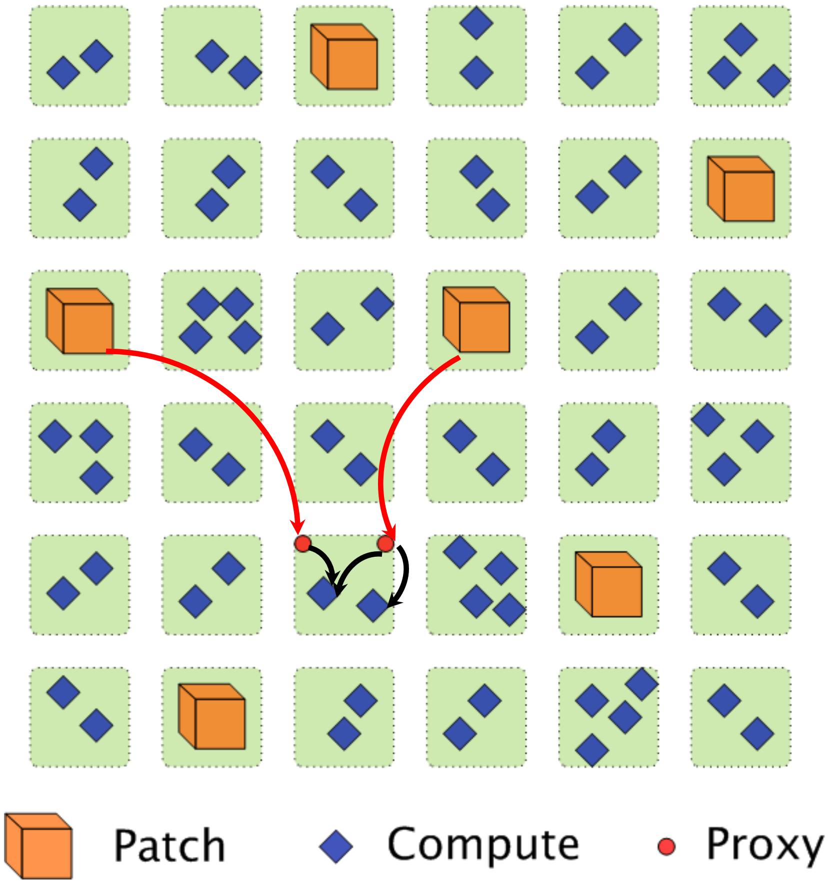


Load Balancing

Bhatele, A., Kumar, S., Mei, C., Phillips, J. C., Zheng, G. & Kale, L. V., **Overcoming Scaling Challenges in Biomolecular Simulations across Multiple Platforms**. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium*, Miami, FL, USA, April 2008.

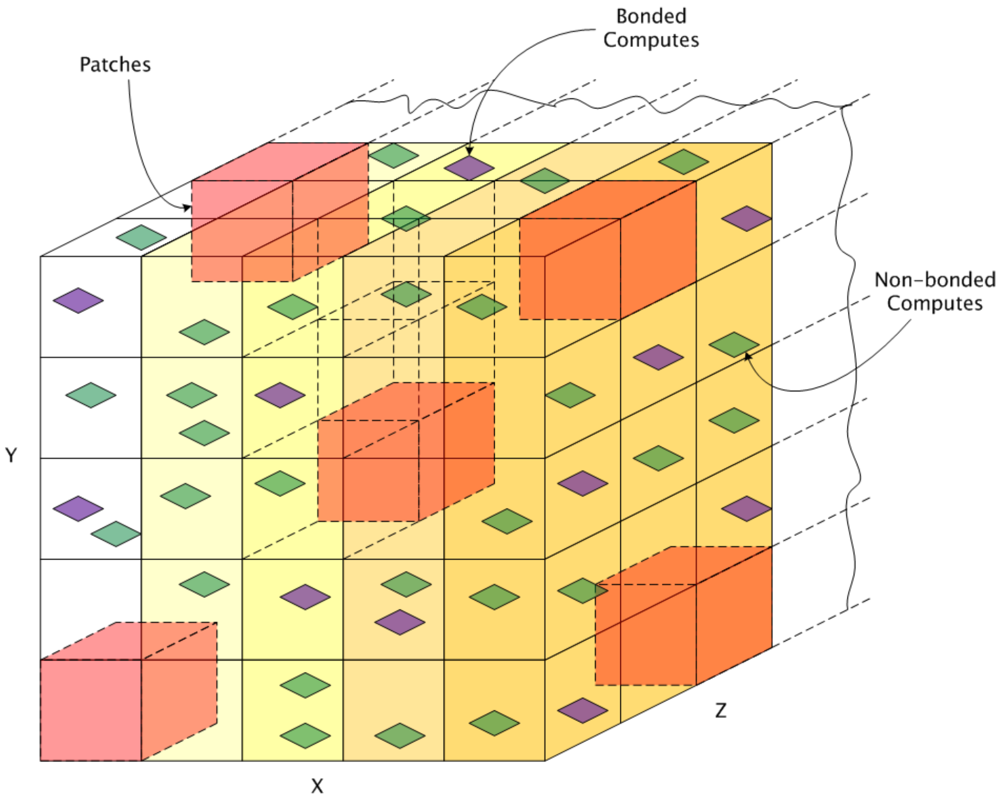
Communication in NAMD

- Each patch multicasts its information to many computes
- Each compute is a target of two multicasts only
- Use 'Proxies' to send data to different computes on the same processor



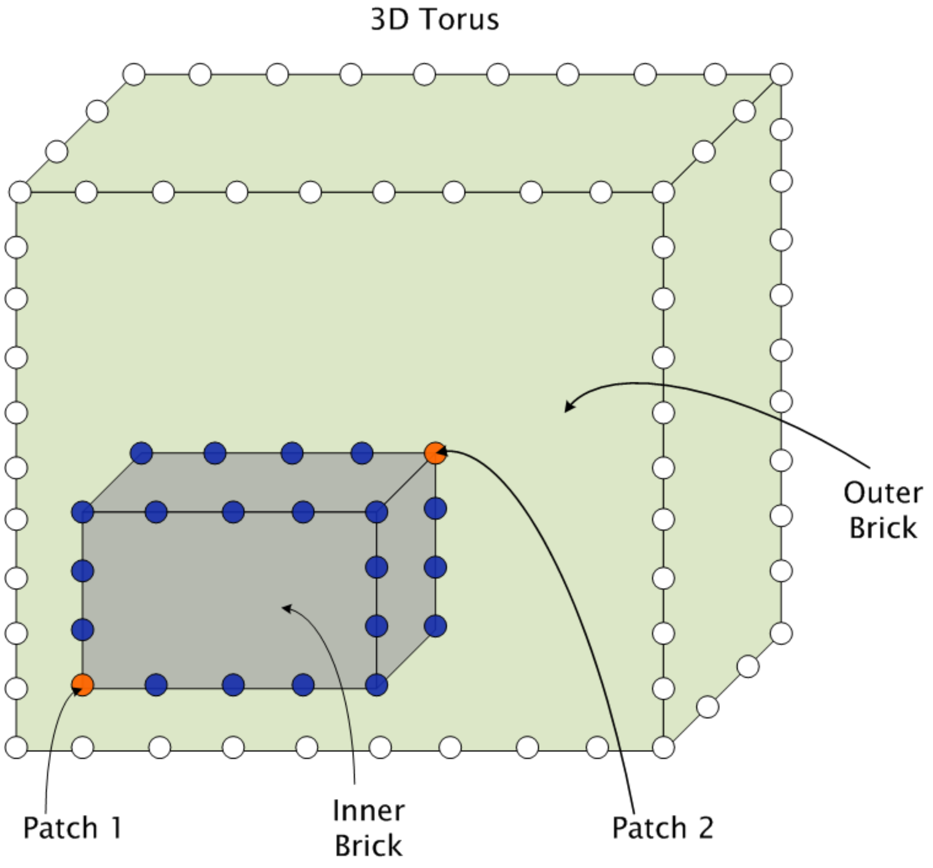
Topology Aware Techniques

- Static Placement of Patches



Topology Aware Techniques (contd.)

- Placement of computes



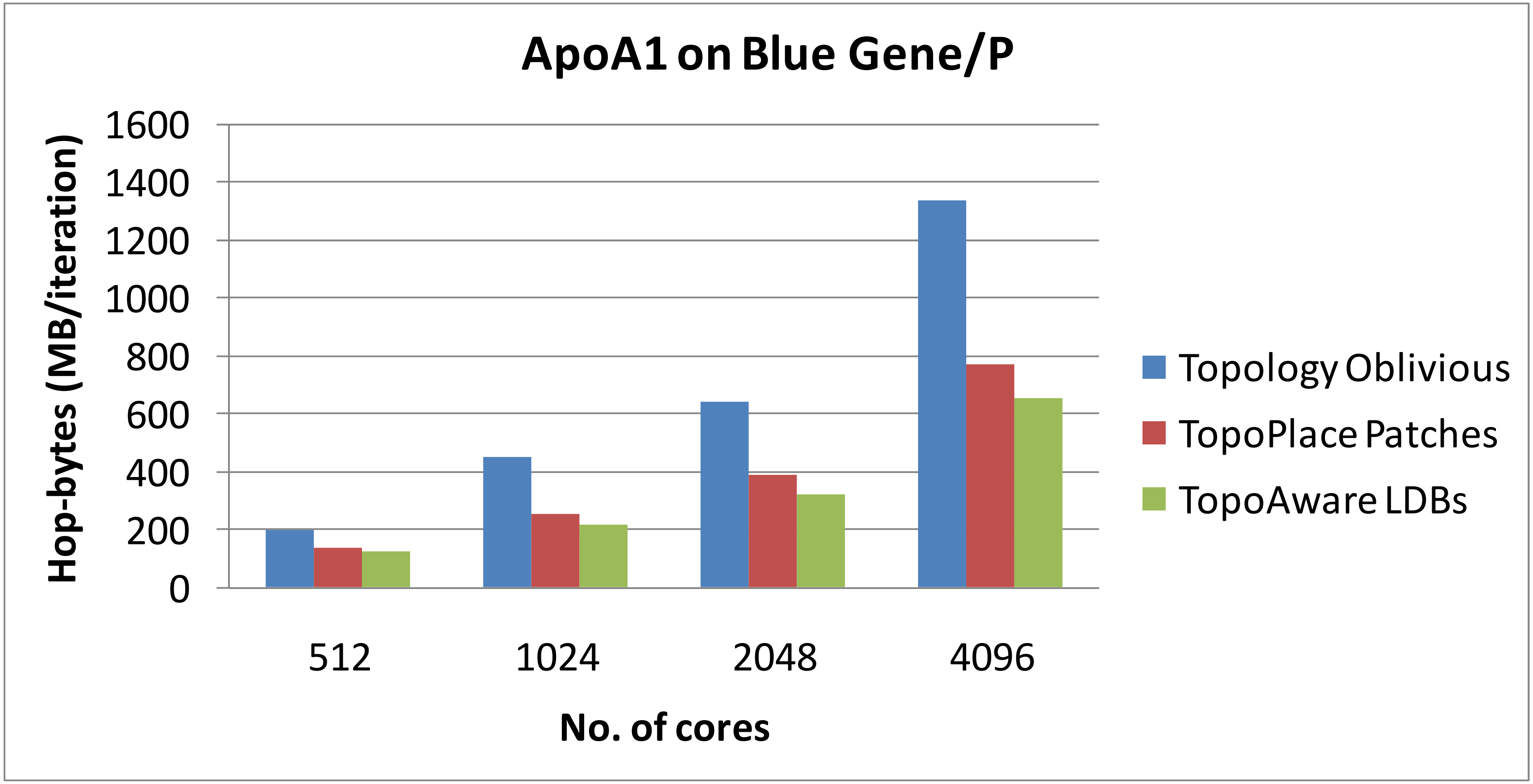
Load Balancing Metrics

- Load Balance: Bring Max-to-Avg Ratio close to 1
- Communication Volume: Minimize the number of proxies
- Communication Traffic: Minimize hop bytes

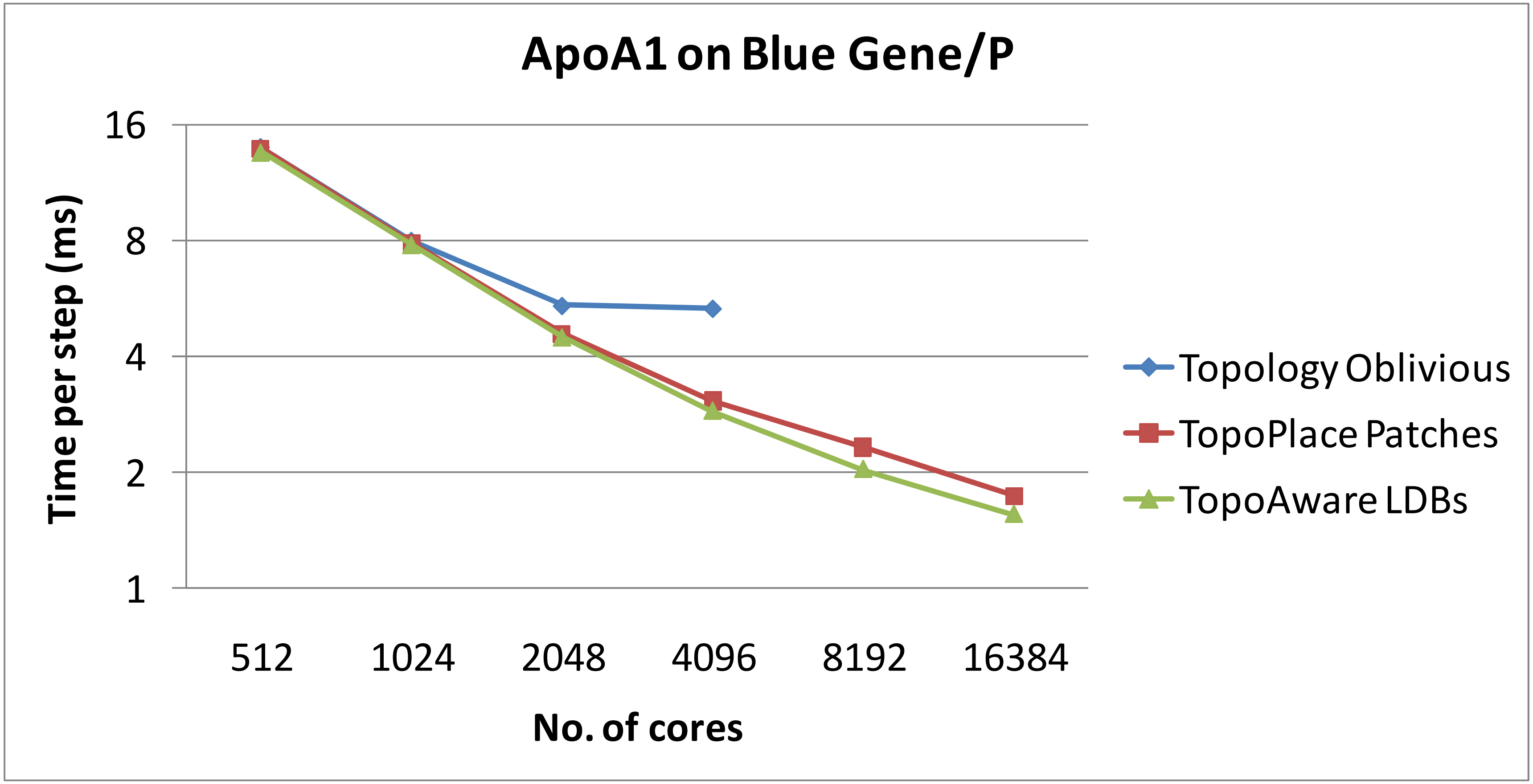
Hop-bytes = Message size X Distance traveled by message

Agarwal, T., Sharma, A., Kale, L.V., **Topology-aware task mapping for reducing communication contention on large parallel machines,** In *Proceedings of IEEE International Parallel and Distributed Processing Symposium*, Rhodes Island, Greece, April 2006.

Results: Hop-bytes



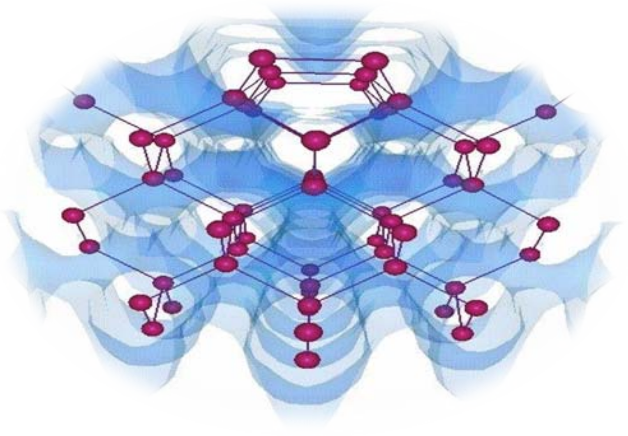
Results: Performance



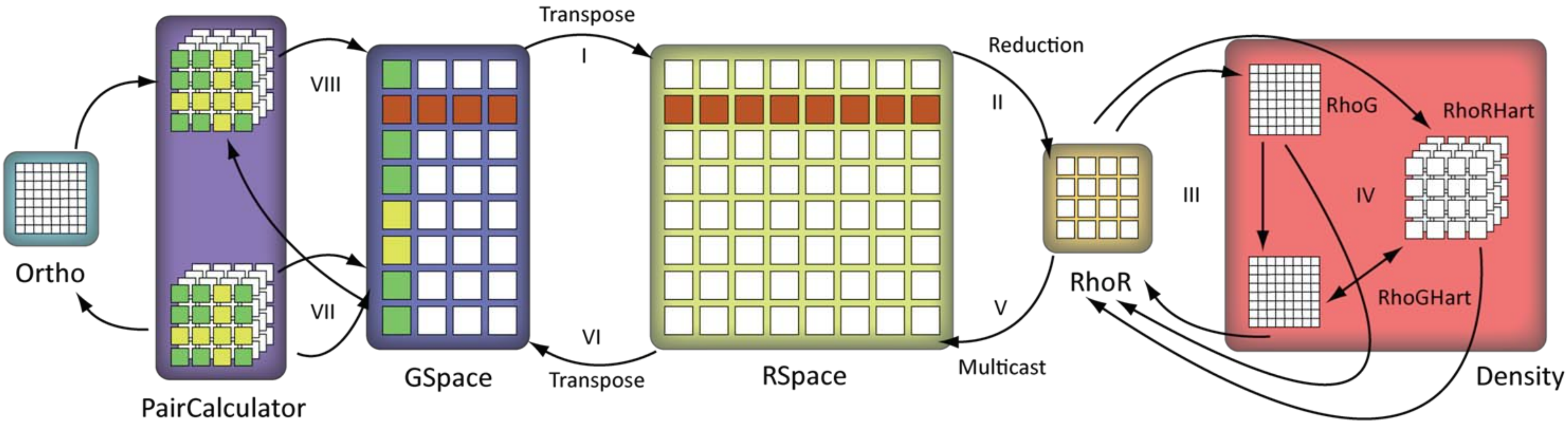
Abhinav Bhatele and Laxmikant V. Kale. **Dynamic Topology Aware Load Balancing Algorithms for MD Applications**. In Proceedings of International Conference on Supercomputing, 2009.

OpenAtom

- Ab-Initio Molecular Dynamics code
- Communication is static and structured
- Challenge: Multiple groups of objects with conflicting communication patterns



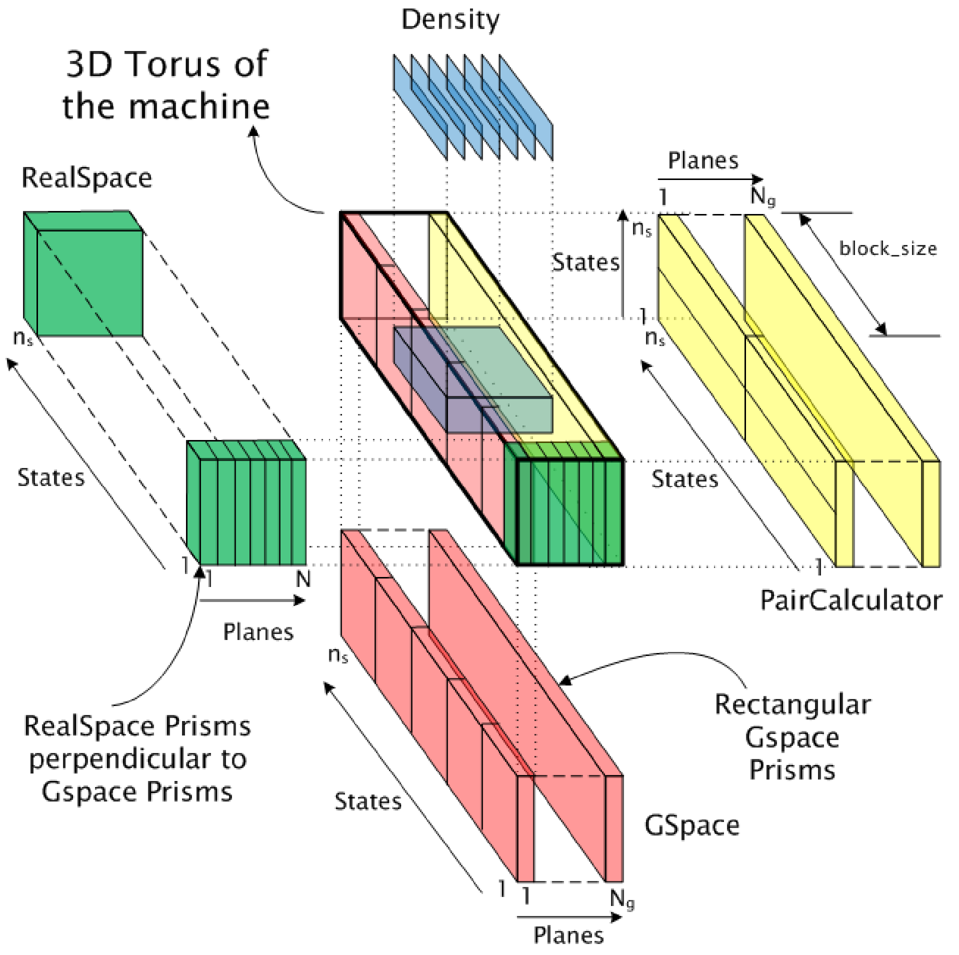
Parallelization using Charm++



Eric Bohm, Glenn J. Martyna, Abhinav Bhatele, Sameer Kumar, Laxmikant V. Kale, John A. Gunnels, and Mark E. Tuckerman. **Fine Grained Parallelization of the Car-Parrinello ab initio MD Method on Blue Gene/L**. *IBM J. of R. and D.: Applications of Massively Parallel Systems, 52(1/2):159-174*, 2008.

Abhinav Bhatele, Eric Bohm, Laxmikant V. Kale, **A Case Study of Communication Optimizations on 3D Mesh Interconnects**, To appear in *Proceedings of Euro-Par (Topic 13 - High Performance Networks)*, 2009.

Topology Mapping of Chare Arrays

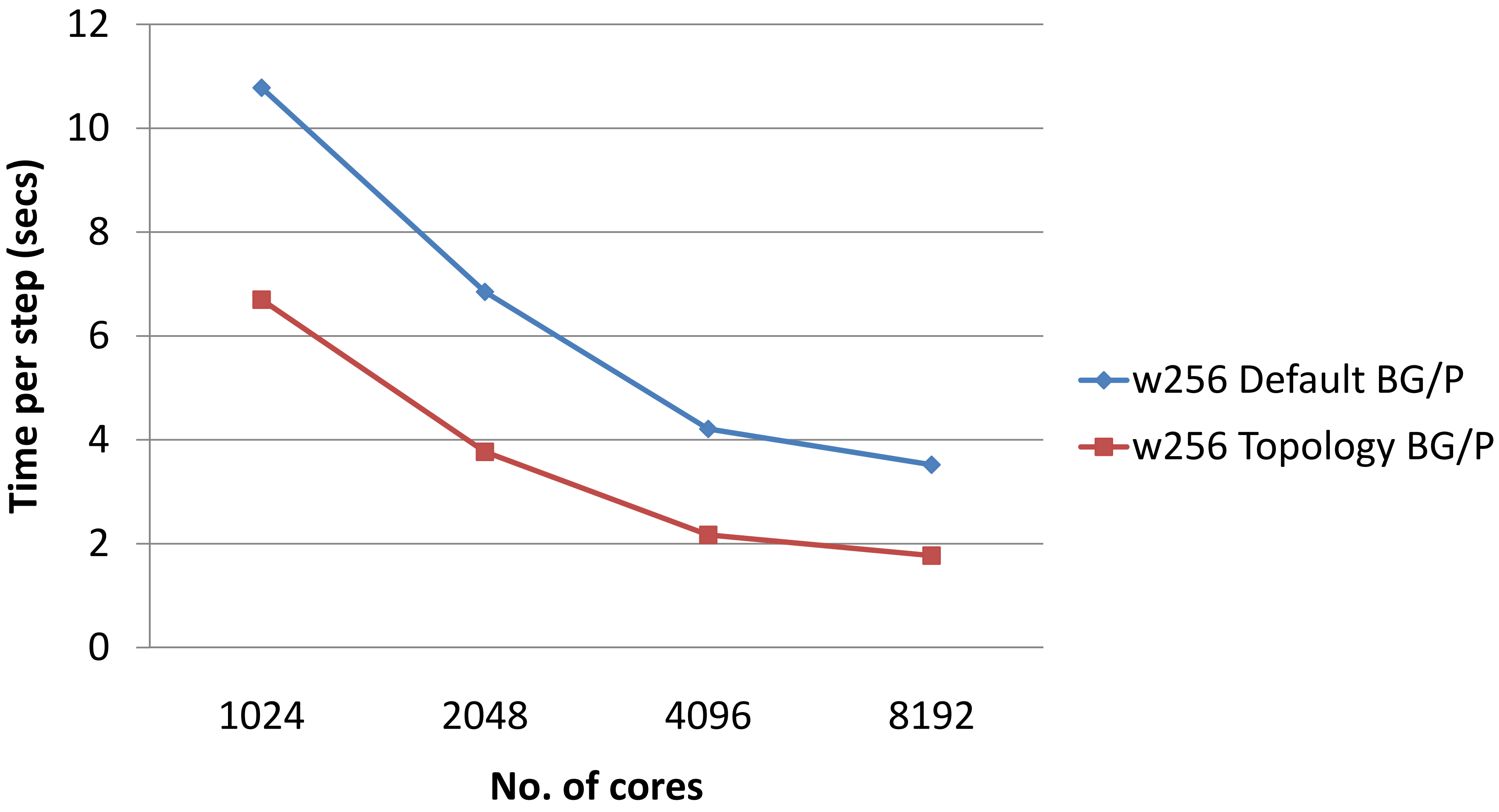


Plane-wise communication

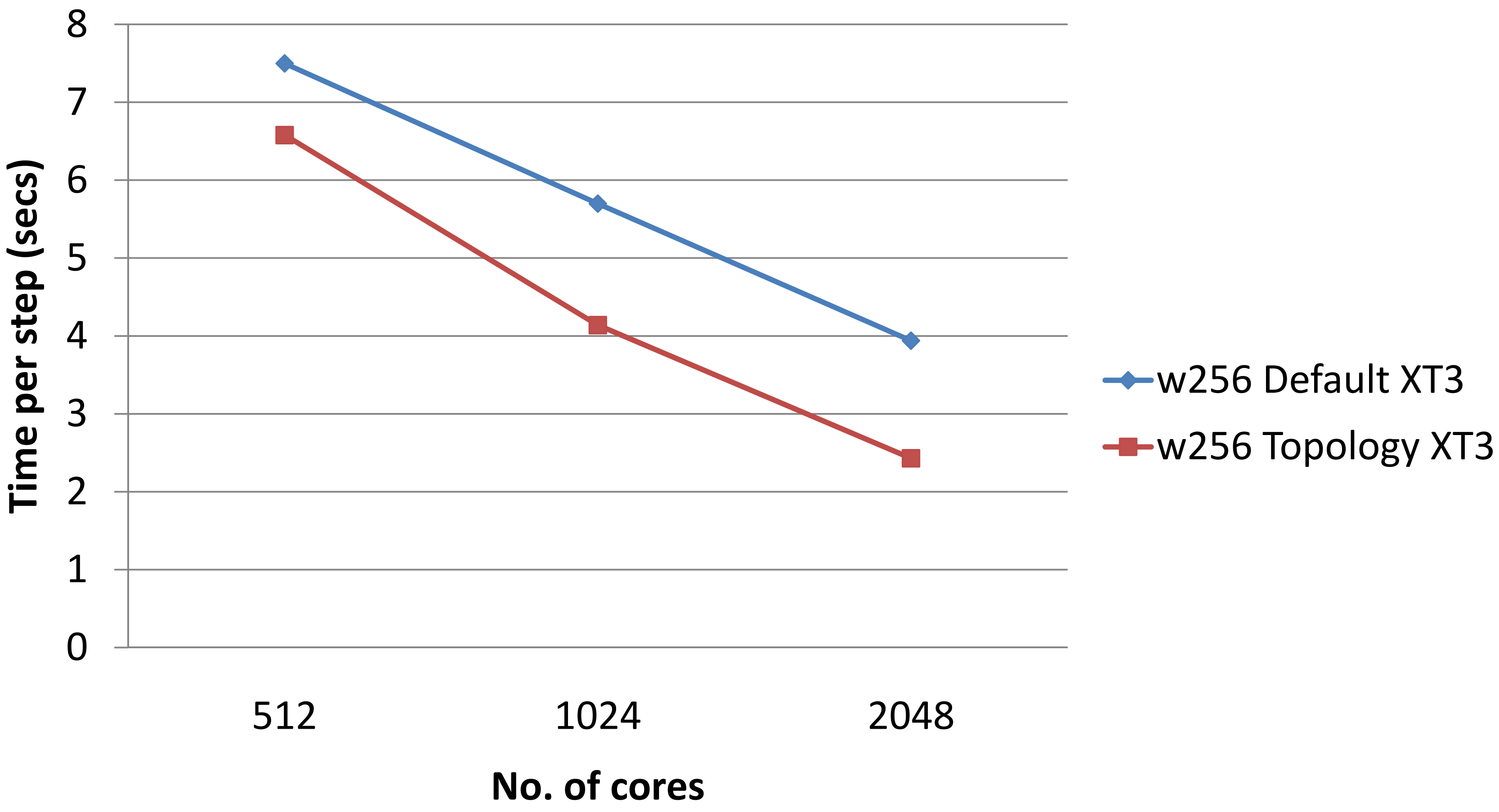
State-wise communication

Joint work with Eric J. Bohm

Results on Blue Gene/P (ANL)



Results on XT3 (Bigben@PSC)



Tools and Techniques

Topology Aware Mapping

- Broadly three requirements
  - Processor topology graph
  - Object communication graph
  - Efficient and scalable mapping algorithms

Application Characteristics

- Computation-bound applications
- Communication-heavy applications
  - Latency tolerant: NAMD
  - Latency sensitive: OpenAtom

Topology Manager API†

- The application needs information such as
  - Dimensions of the partition
  - Rank to physical co-ordinates and vice-versa
- TopoManager: a uniform API
  - On BG/L and BG/P: provides a wrapper for system calls
  - On XT3/4/5, there are no such system calls
  - Provides a clean and uniform interface to the application

† http://charm.cs.uiuc.edu/~bhatele/phd/topomgr.htm

TopoMgrAPI

- getDimNX(), getDimNY(), getDimNZ(), getDimNT()
- rankToCoordinates(), coordinatesToRank()
- getHopsBetweenRanks()

Object Communication Graph

- Obtaining this graph:
  - Manually
  - Profiling (e.g. IBM’s HPCT tools)
  - Charm++’s instrumentation framework
- Visualizing the graph
- Pattern matching

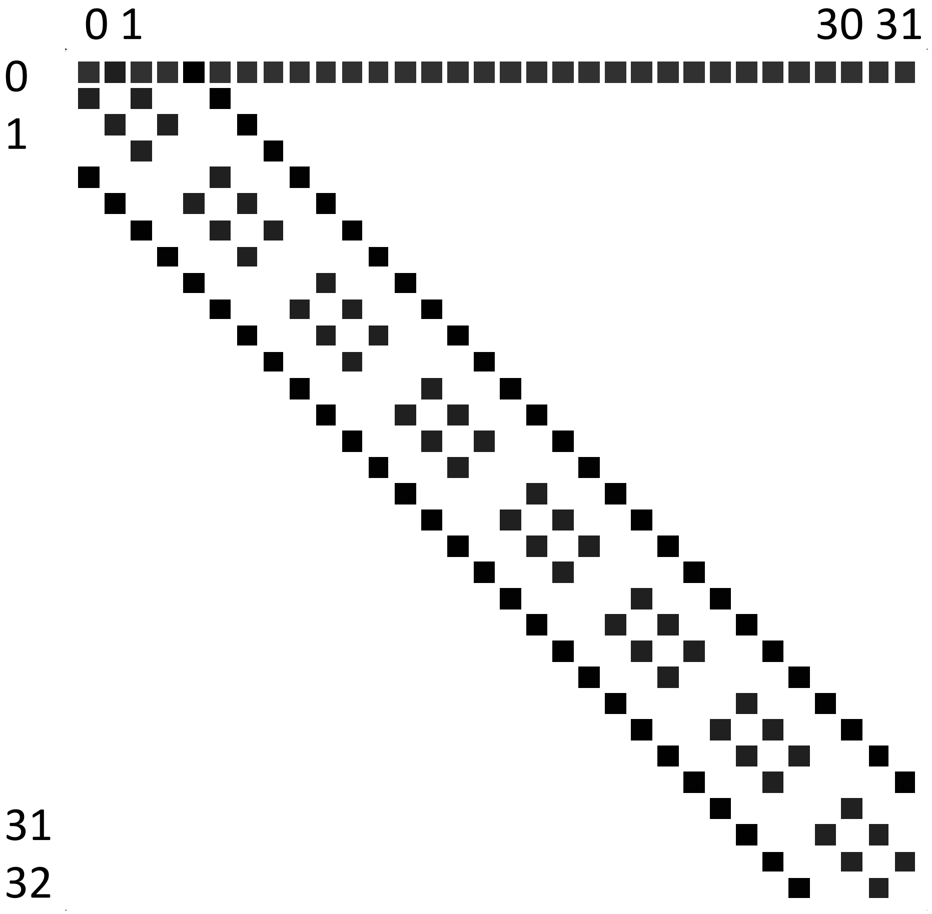
Communication Scenarios

- Static Regular Graph: 3D Stencil
  - Structured Mesh applications such as MILC
  - POP, MILC, WRF
- Static Irregular Graph: Unstructured Mesh applications such as UMT2K
- Dynamic Communication Graph: MD codes, Unstructured Mesh codes with AMR
- Use pattern matching to get an idea

Simple 2D/3D Graph

- Many science applications have a near-neighbor communication pattern
  - POP: Parallel Ocean Program
  - MILC: MIMD Lattice Computation
  - WRF: Weather Research and Forecasting

WRF Communication Graph

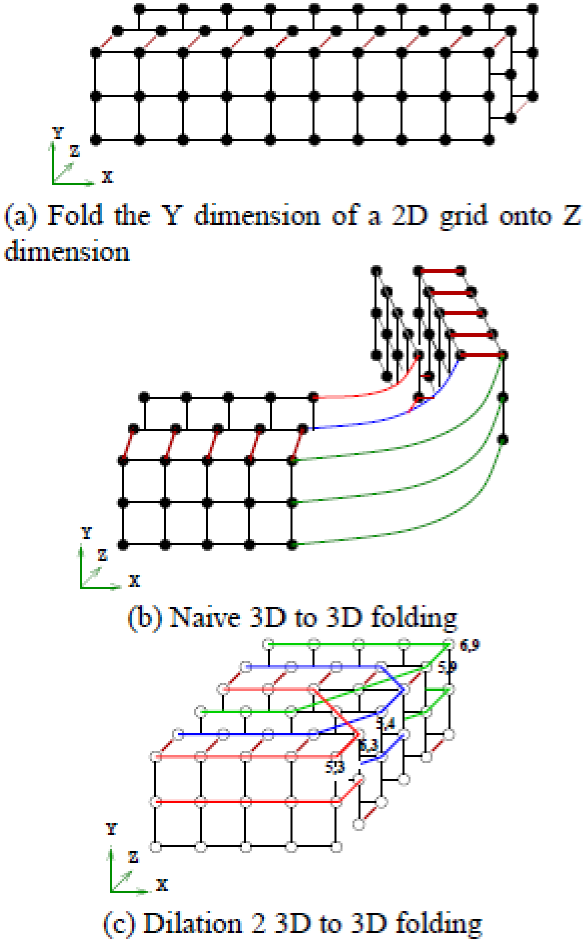


Folding of 2D/3D to 3D

For more objects than processors:

- Blocking Techniques

SMP nodes ?



(a) Fold the Y dimension of a 2D grid onto Z dimension

(b) Naive 3D to 3D folding

(c) Dilation 2 3D to 3D folding

Hao Yu, I-Hsin Chung and Jose Moreira, **Topology Mapping for Blue Gene/L Supercomputer**, In *Proceedings of the ACM/IEEE Supercomputing Conference*, 2006

Irregular Graphs

- Heuristic Techniques
  - Pairwise Exchanges
  - Greedy Strategies

Automatic Mapping Framework

- Obtain the communication graph (previous run, at runtime)
- Pattern matching to identify communication patterns
- Apply heuristic techniques for different scenarios to arrive at a mapping solution automatically

Summary

- Scalable Load Balancing: Hierarchical
- Machines of Petascale Era: ‘Exploitable’ Topologies
- MPI Benchmarks show:
  - Contention for the same links by different messages slows all of them down
- Topology aware mapping to avoid contention
  - Carefully see if application would benefit from this
- Tools and Techniques
  - TopoMgrAPI
  - Obtaining the object graph: Instrumentation
  - Pattern Matching
  - Techniques: Folding, Heuristics

Summary

1. Topology is important again
2. Even on fast interconnects such as Cray

3. In presence of contention, bandwidth occupancy effects message latencies significantly
4. Increases with the number of hops each message travels

5. Topology Manager API: A uniform API for IBM and Cray machines
6. Different algorithms depending on the communication patterns
7. Eventually, an automatic mapping framework

Acknowledgements:

1. Argonne National Laboratory: Pete Beckman, Tisha Stacey
2. Pittsburgh Supercomputing Center: Chad Vizino, Shawn Brown
3. Oak Ridge National Laboratory: Patrick Worley, Donald Frederick
4. Grants: DOE Grant B341494 funded by CSAR, DOE grant DE-FG05-08OR23332 through ORNL LCF, and a NIH Grant PHS 5 P41 RR05969-04 for Molecular Dynamics

References:

1. Abhinav Bhatele, Laxmikant V. Kale, **Quantifying Network Contention on Large Parallel Machines**, *Parallel Processing Letters (Special issue on Large-Scale Parallel Processing)*, 2009

2. Abhinav Bhatele, Laxmikant V. Kale, **Benefits of Topology-aware Mapping for Mesh Topologies**, *Parallel Processing Letters (Special issue on Large-Scale Parallel Processing), Vol: 18 Issue:4, Pages: 549-566*, 2008

E-mail: bhatele, kale @ illinois.edu
Webpage: http://charm.cs.illinois.edu

Thanks!