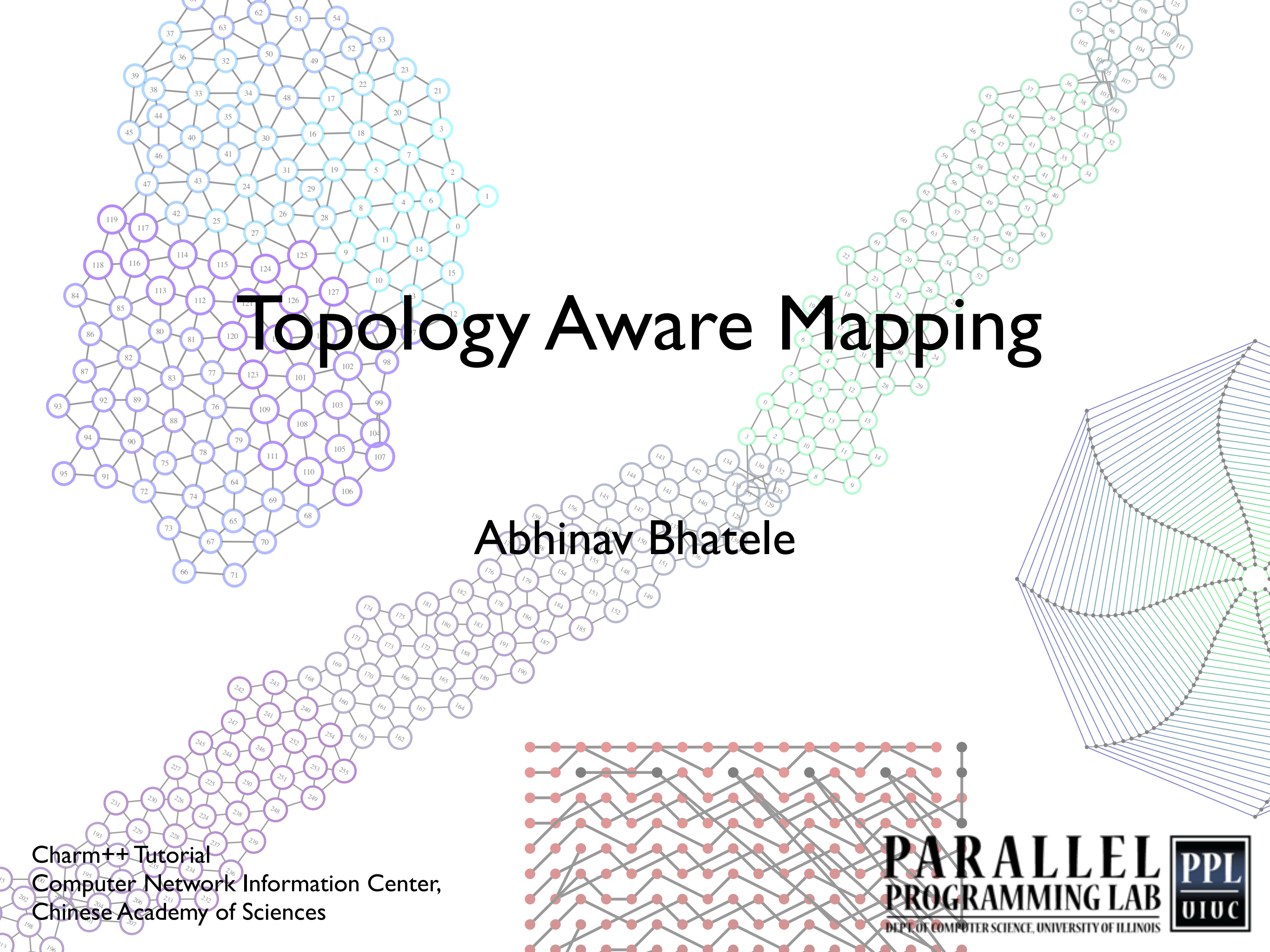


Topology Aware Mapping

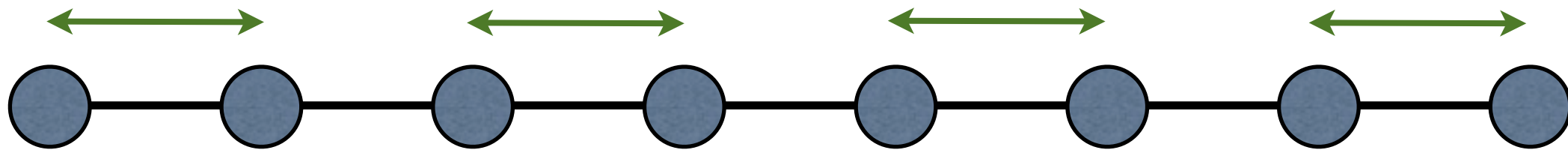
Abhinav Bhatele



Charm++ Tutorial
Computer Network Information Center,
Chinese Academy of Sciences

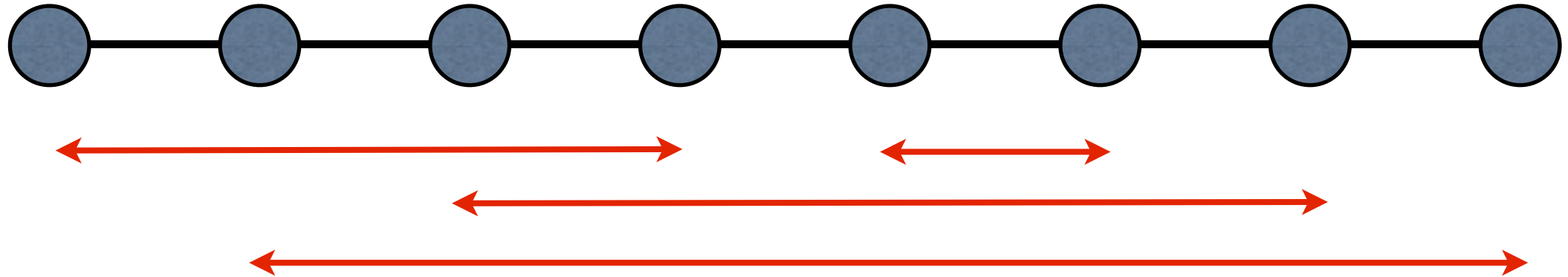
Motivation

- Running a parallel application on a linear array of processors:



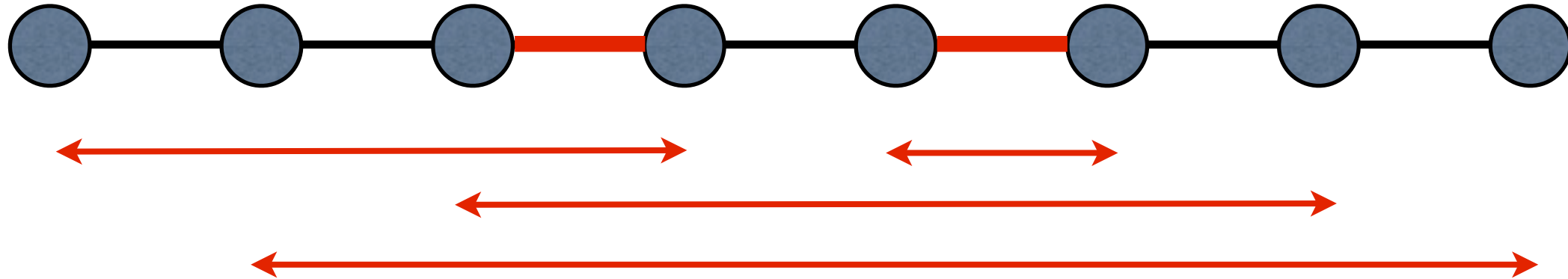
Motivation

- Running a parallel application on a linear array of processors:



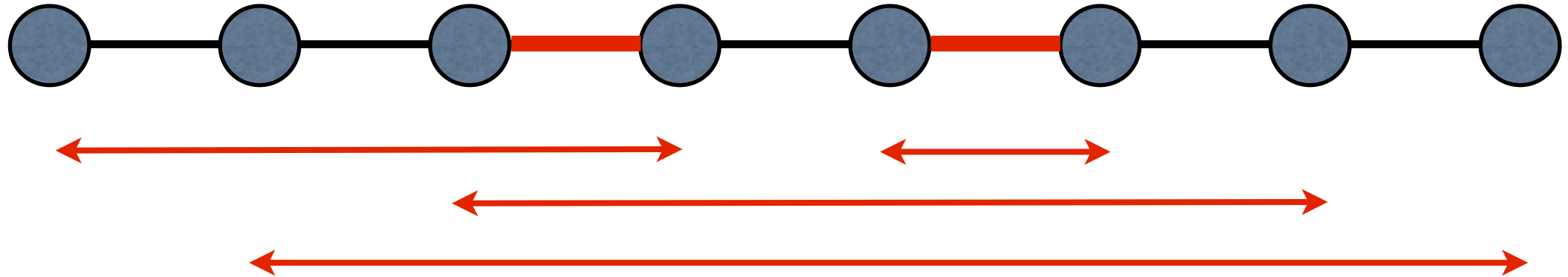
Motivation

- Running a parallel application on a linear array of processors:



Motivation

- Running a parallel application on a linear array of processors:

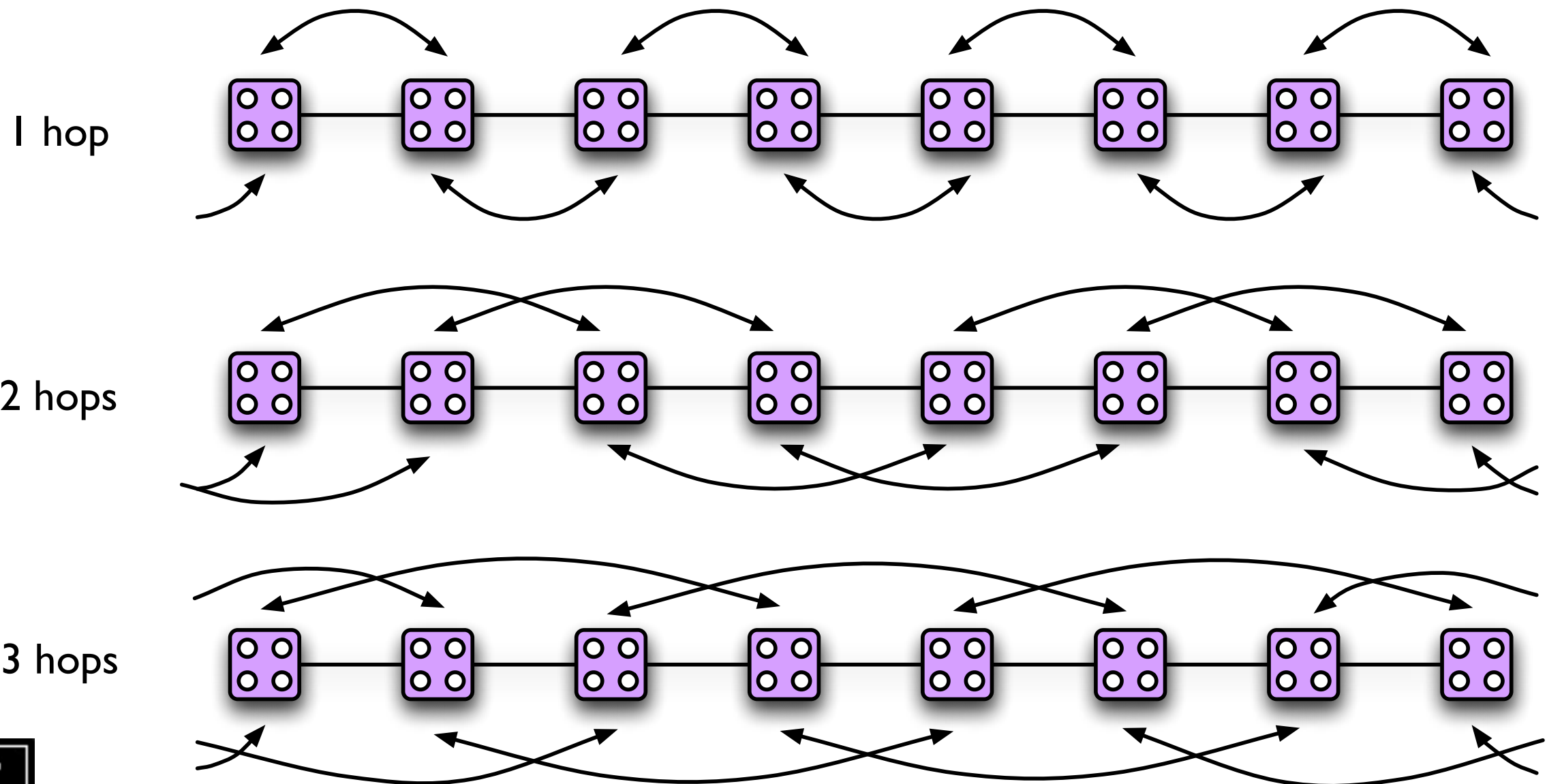


- Typical communication is between random pairs of processors simultaneously



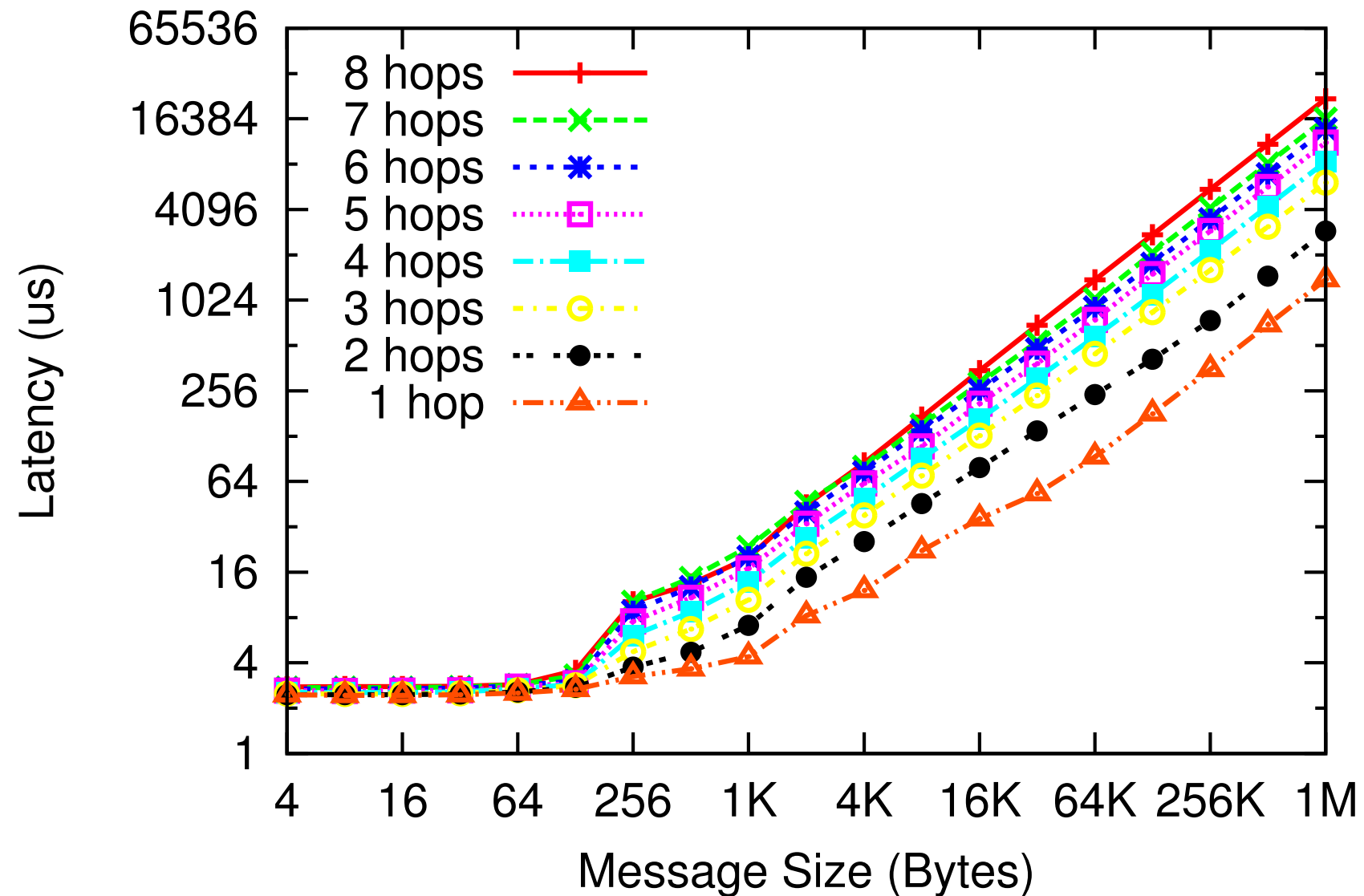
Benchmark Creating Artificial Contention

- Pair each processor with a partner that is n hops away



Results: Contention

Effect of distance on latencies (Torus - 8 x 8 x 16)



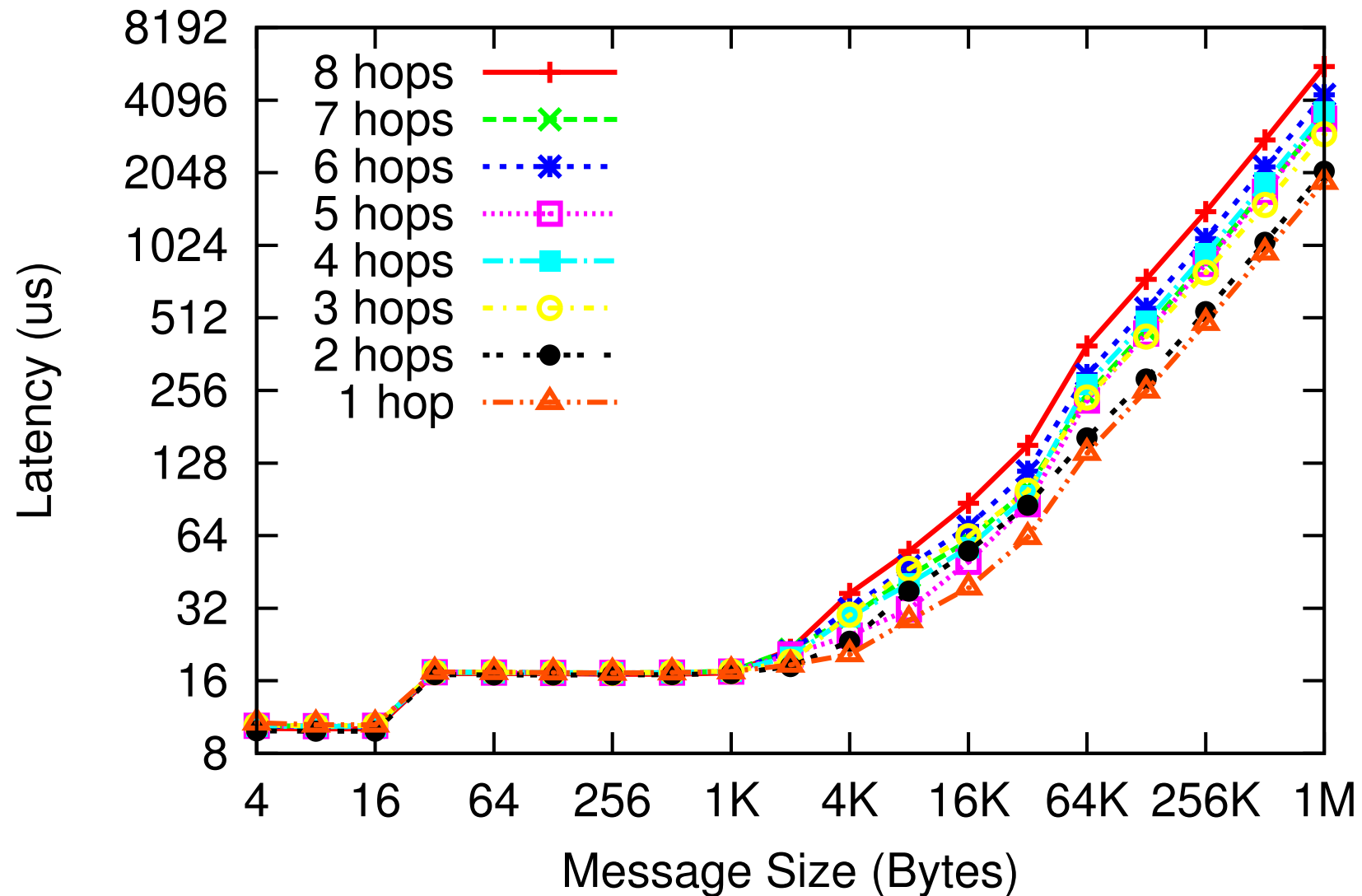
Blue Gene/P

Bhatele A., Kale L.V., Quantifying Network Contention on Large Parallel Machines, *Parallel Processing Letters (Special Issue on Large-Scale Parallel Processing)*, 2009. [Best Poster Award, ACM Student Research Competition, Supercomputing 2008, Austin, TX.](#)



Results: Contention

Effect of distance on latencies (Torus - 8 x 8 x 16)



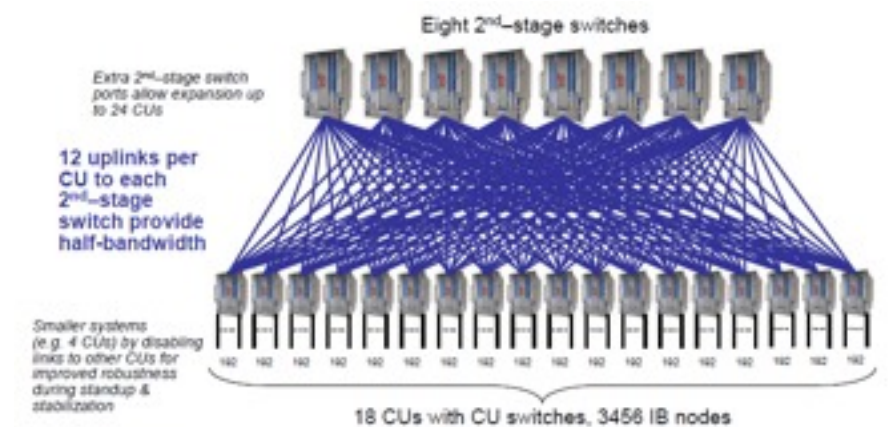
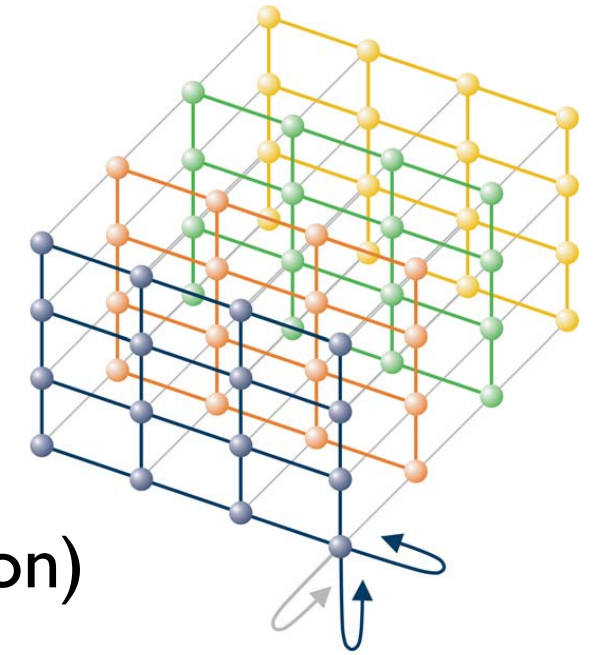
XT4

Bhatele A., Kale L.V., Quantifying Network Contention on Large Parallel Machines, *Parallel Processing Letters (Special Issue on Large-Scale Parallel Processing)*, 2009. [Best Poster Award, ACM Student Research Competition, Supercomputing 2008, Austin, TX.](#)



Interconnect Topologies

- Three dimensional meshes
 - 3D Torus: Blue Gene/L, Blue Gene/P, Cray XT4/5
- Trees
 - Fat-trees (Infiniband) and CLOS networks (Federation)
- Dense Graphs
 - Kautz Graph (SiCortex), Hypercubes
- Future Topologies?
 - Blue Waters, Blue Gene/Q

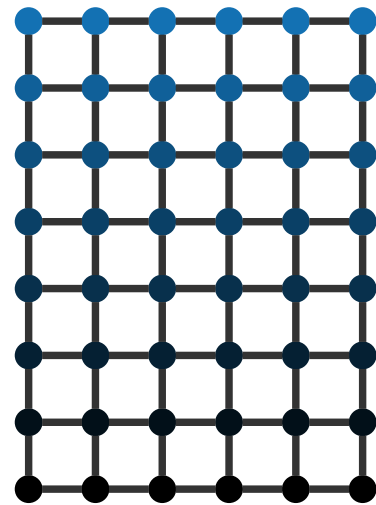
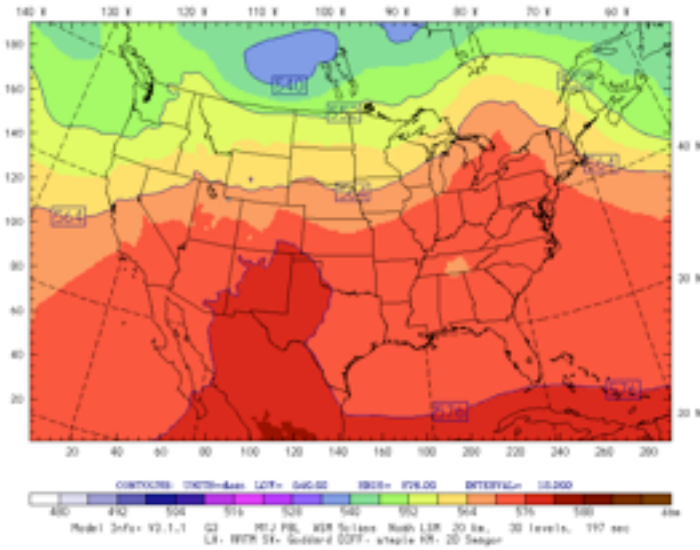


Roadrunner Technical Seminar Series, March 13th 2008, Ken Koch, LANL

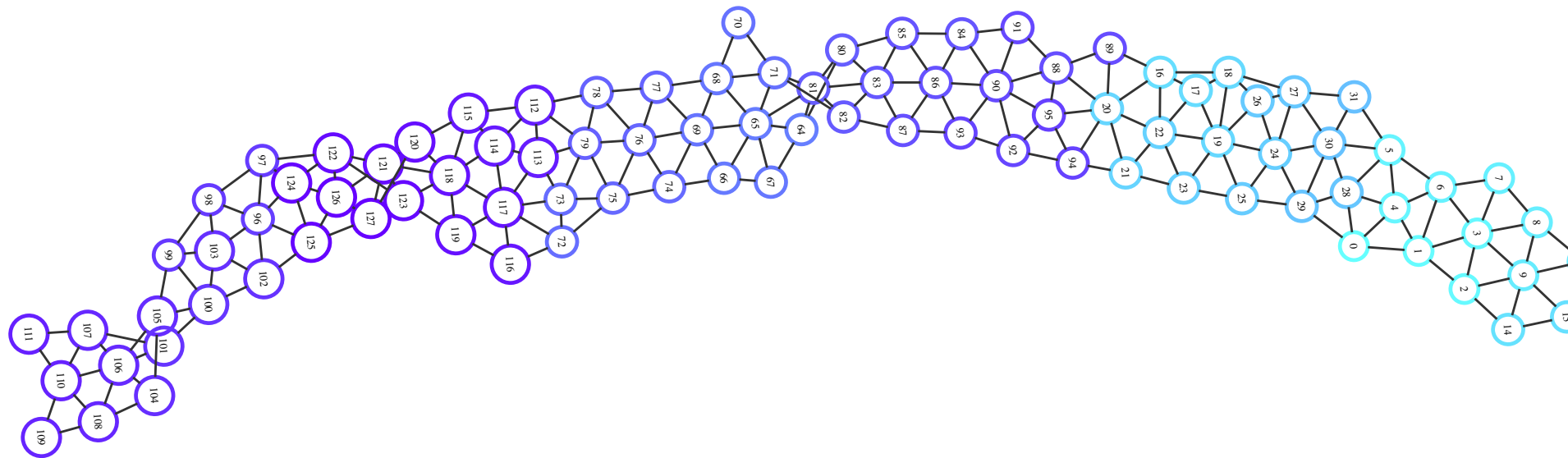
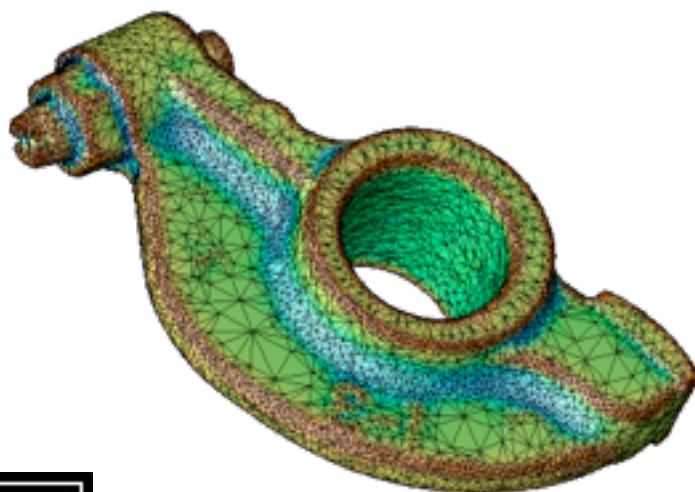
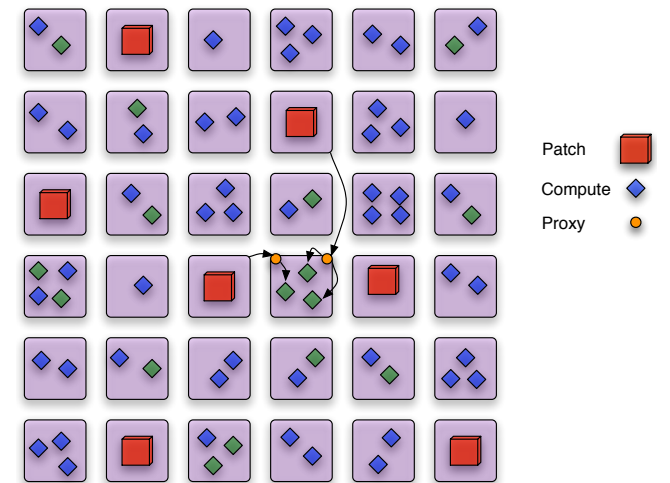
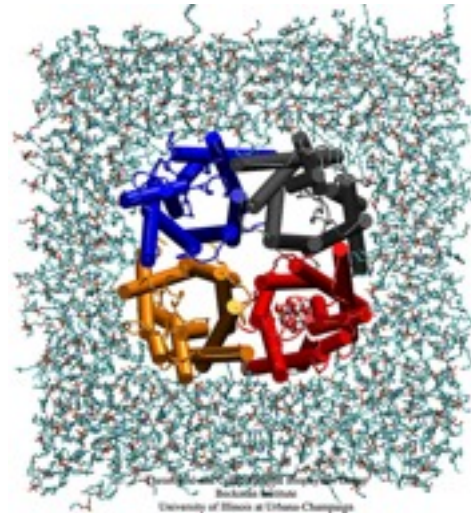


Application Topologies

http://wrf-model.org/plots/realtime_main.php



<http://www.ks.uiuc.edu/Gallery/Science/>



<http://math.lanl.gov/Research/Projects/meshing.shtml>

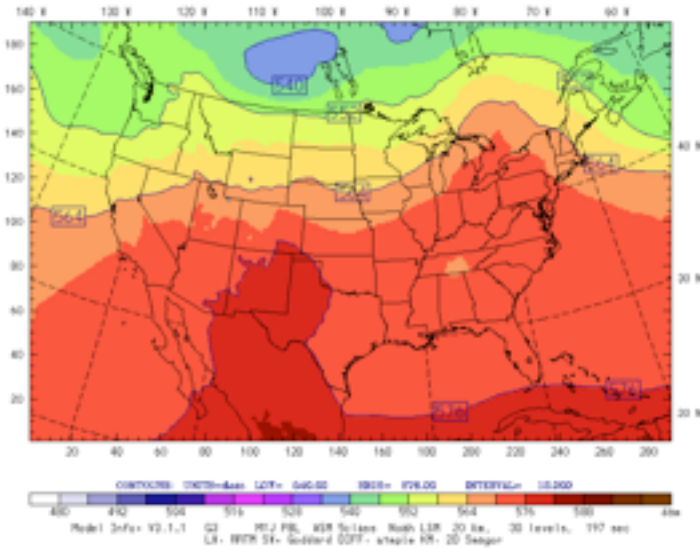
December 16th, 2010

CNIC @ CAS © Abhinav Bhatele

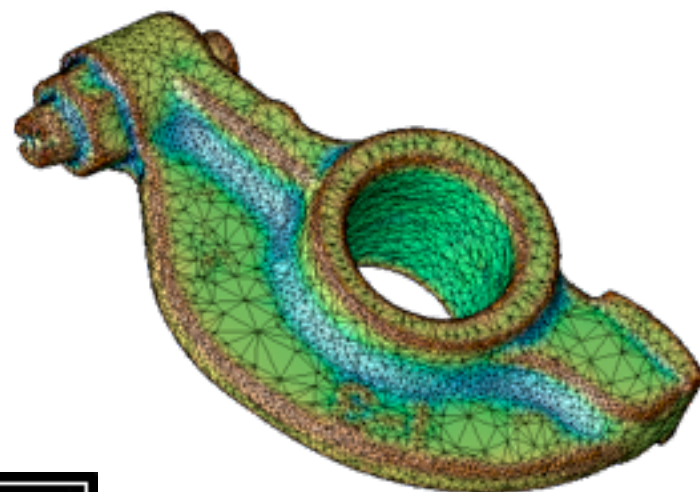
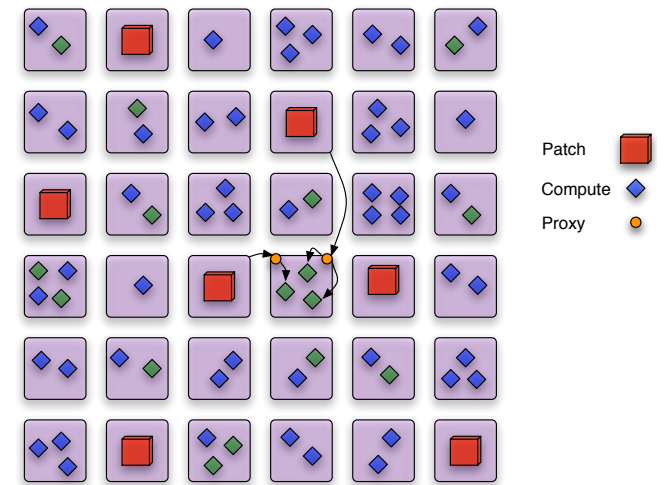
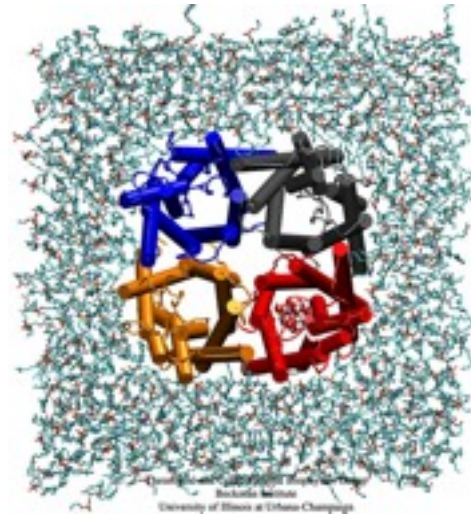


Application Topologies

http://wrf-model.org/plots/realtime_main.php



<http://www.ks.uiuc.edu/Gallery/Science/>



We want to map communicating objects closer to one another



<http://math.lanl.gov/Research/Projects/meshing.shtml>

December 16th, 2010

CNIC @ CAS © Abhinav Bhatele



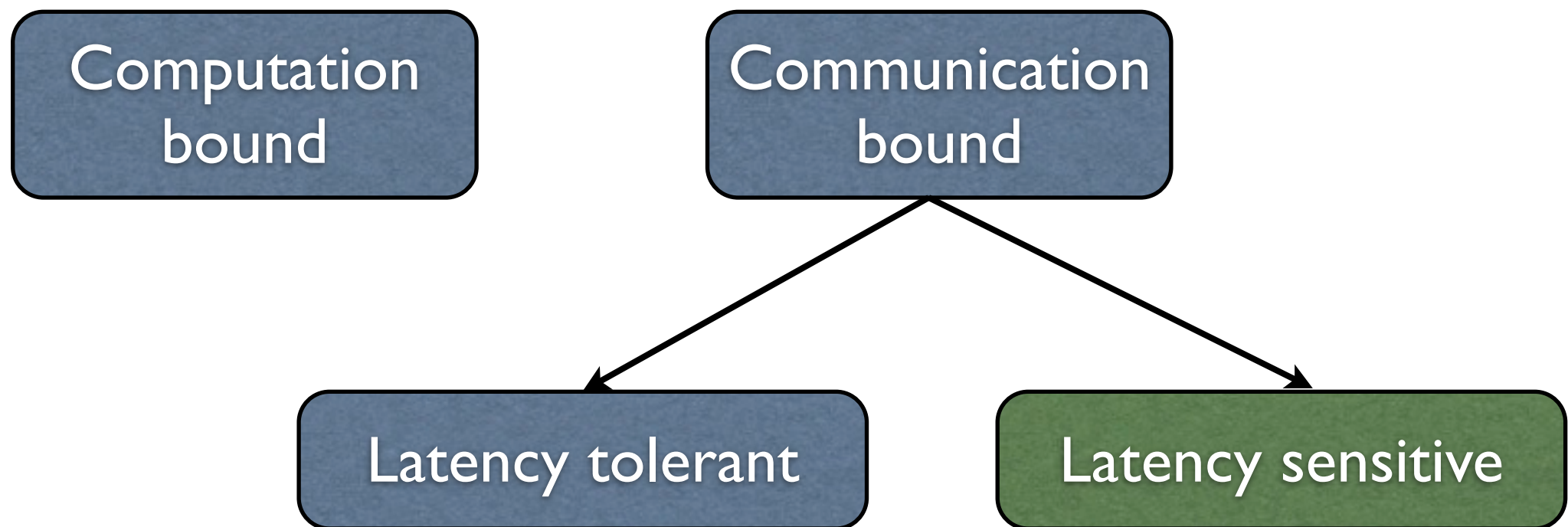
The Mapping Problem

- Applications have a communication topology and processors have an interconnect topology
- Definition: Given a set of communicating parallel “entities”, map them on to physical processors to optimize communication
- Goals:
 - Minimize communication traffic and hence contention
 - Balance computational load (when $n > p$)



Scope of this work

- Currently we are focused on 3D mesh/torus machines
- For certain classes of applications



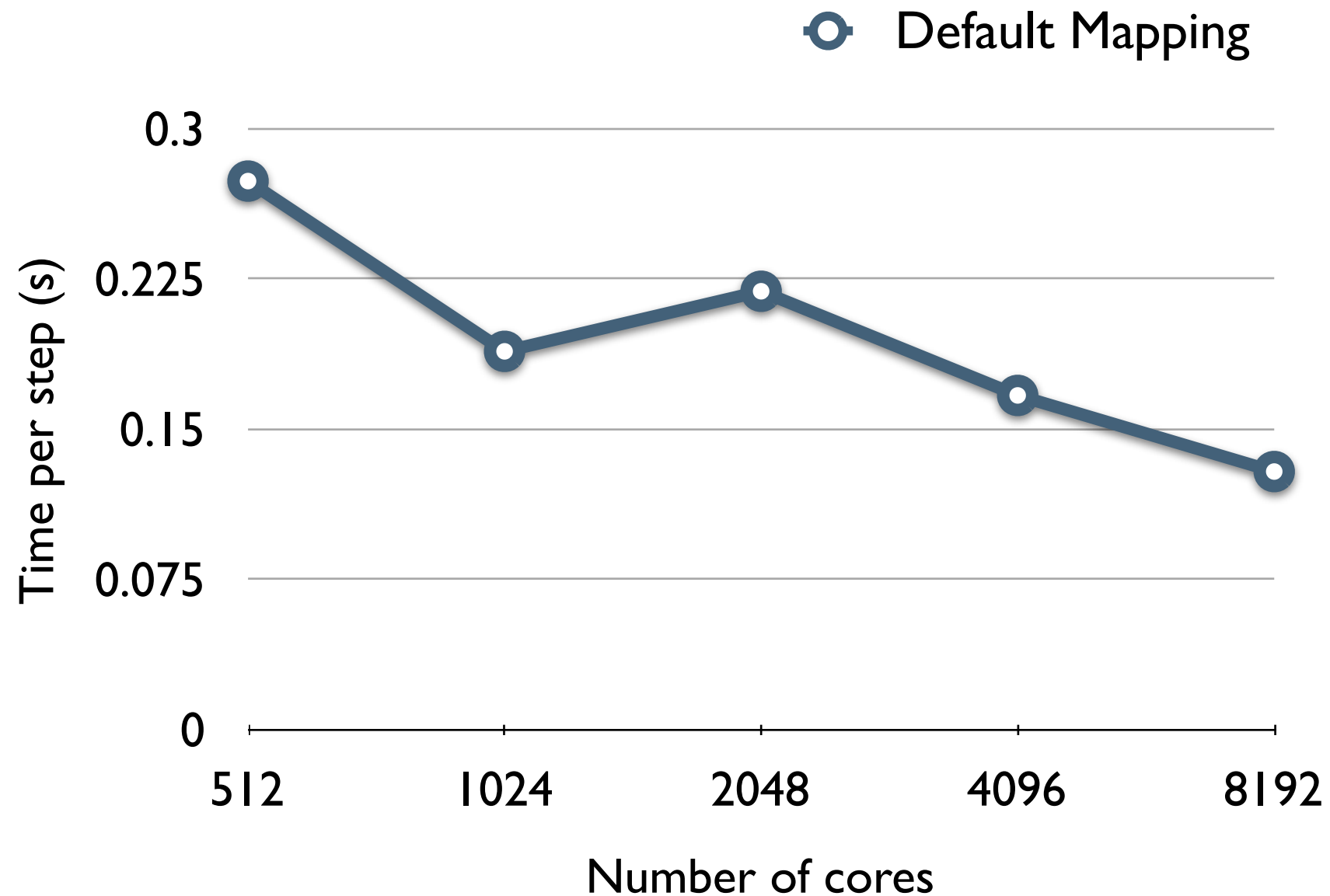
Outline

- Case studies:
 - OpenAtom
 - NAMD
- Automatic Mapping Framework
 - Pattern matching
- Heuristics for Regular Graphs
- Heuristics for Irregular Graphs



Case Study I: OpenAtom

Performance on Blue Gene/L



Diagnosis

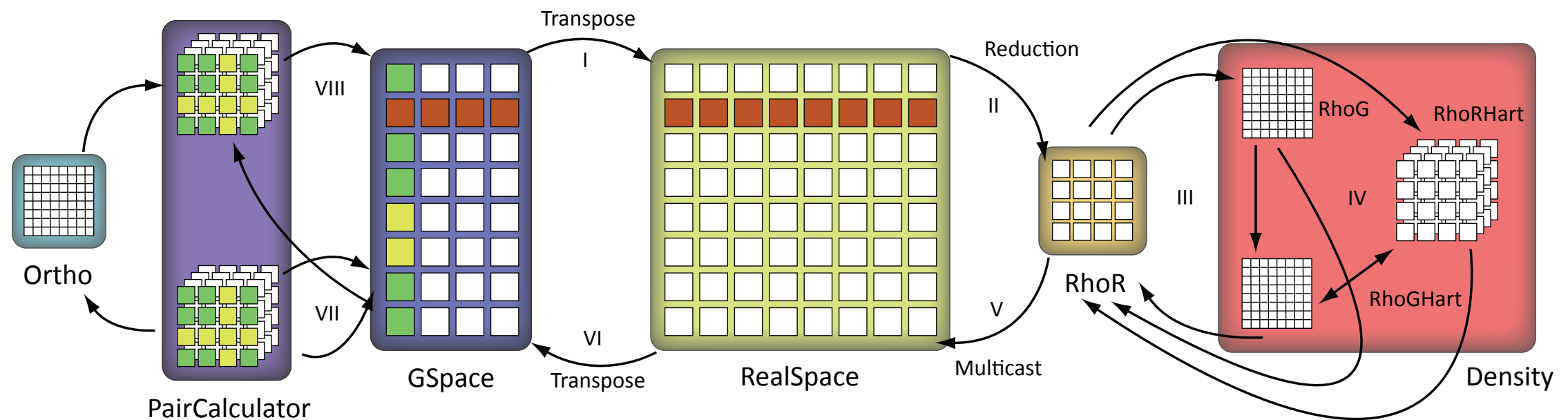
8.48 secs



Timeline view (OpenAtom on 8,192 cores of BG/L) using the performance visualization tool, Projections



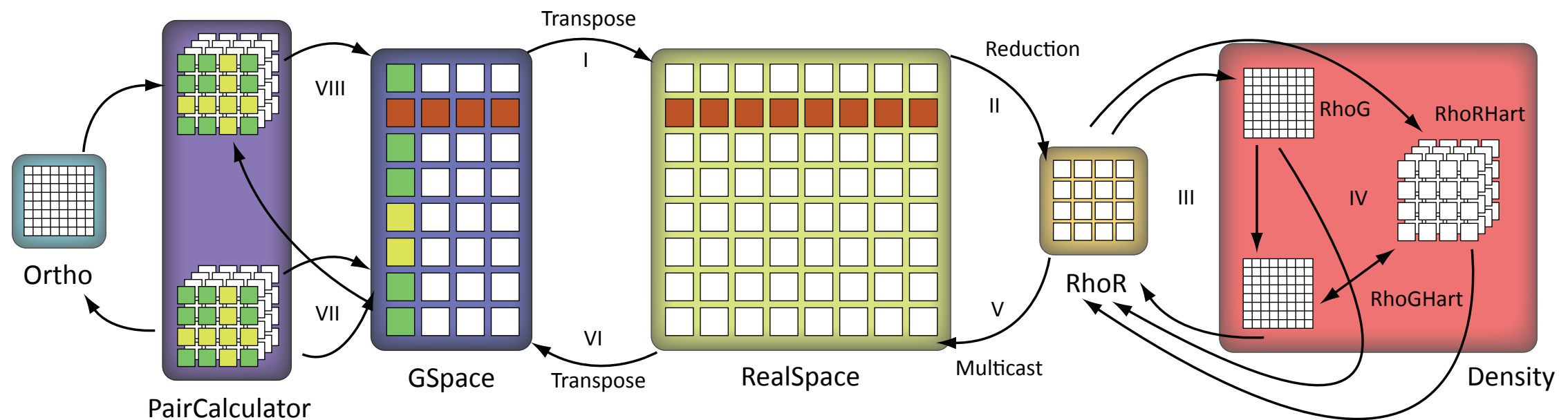
Mapping of OpenAtom Arrays



A. Bhatele, E. Bohm, and L.V. Kale. A Case Study of Communication Optimizations on 3D Mesh Interconnects. In Euro-Par, LNCS 5704, pages 1015–1028, 2009. *Distinguished Paper Award, Feng Chen Memorial Best Paper Award*



Mapping of OpenAtom Arrays



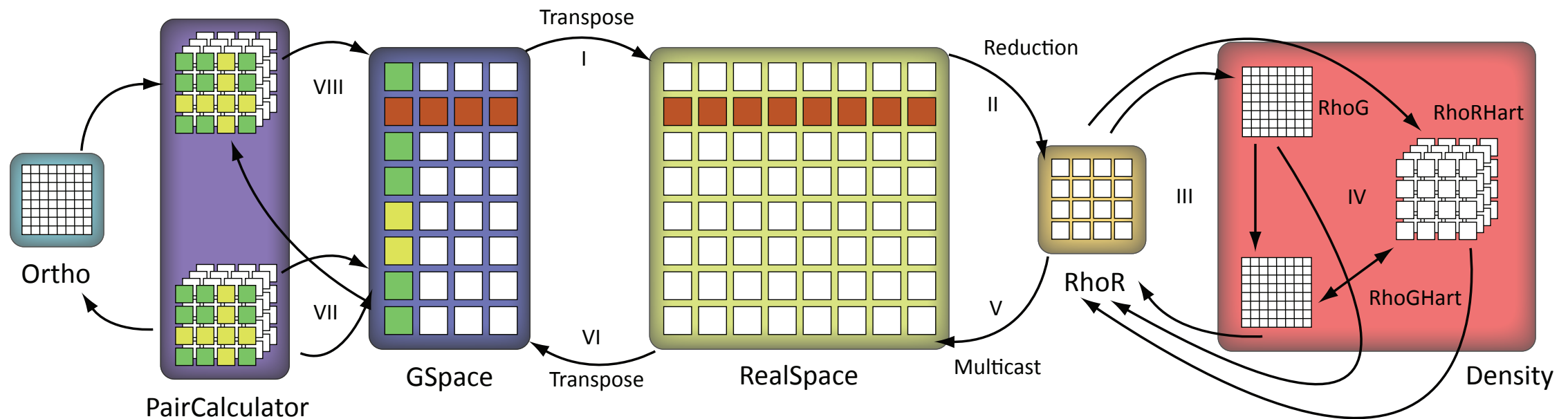
Paircalculator and GSpace have plane-wise communication

RealSpace and GSpace have state-wise communication

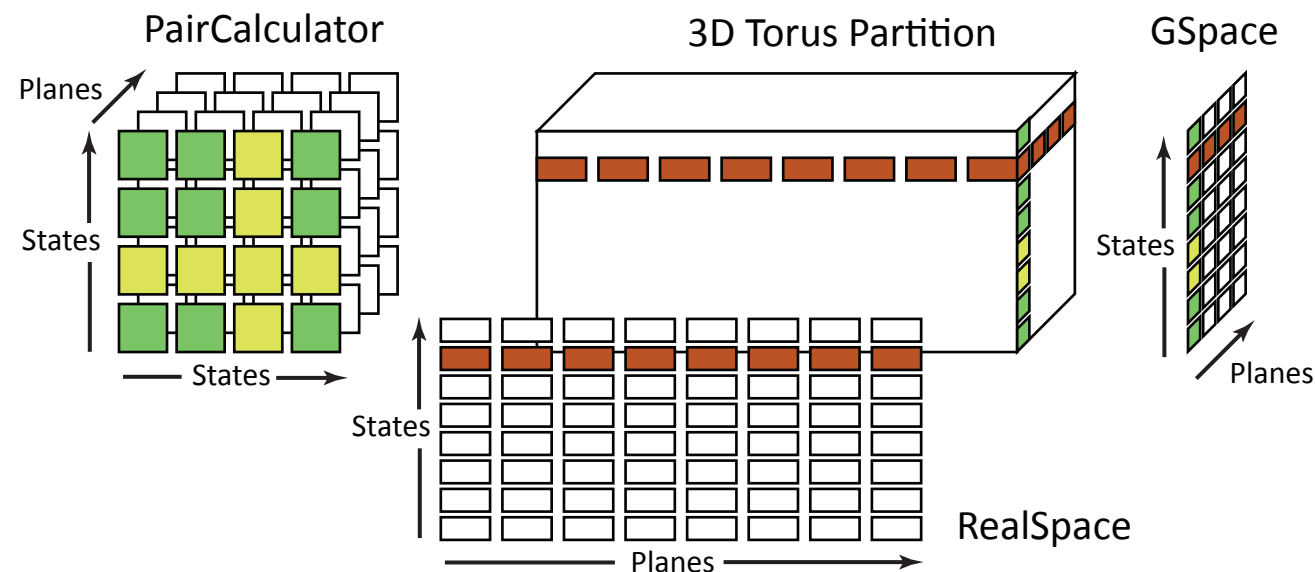
A. Bhatele, E. Bohm, and L.V. Kale. A Case Study of Communication Optimizations on 3D Mesh Interconnects. In Euro-Par, LNCS 5704, pages 1015–1028, 2009. *Distinguished Paper Award, Feng Chen Memorial Best Paper Award*



Mapping of OpenAtom Arrays



Paircalculator and GSpace have plane-wise communication



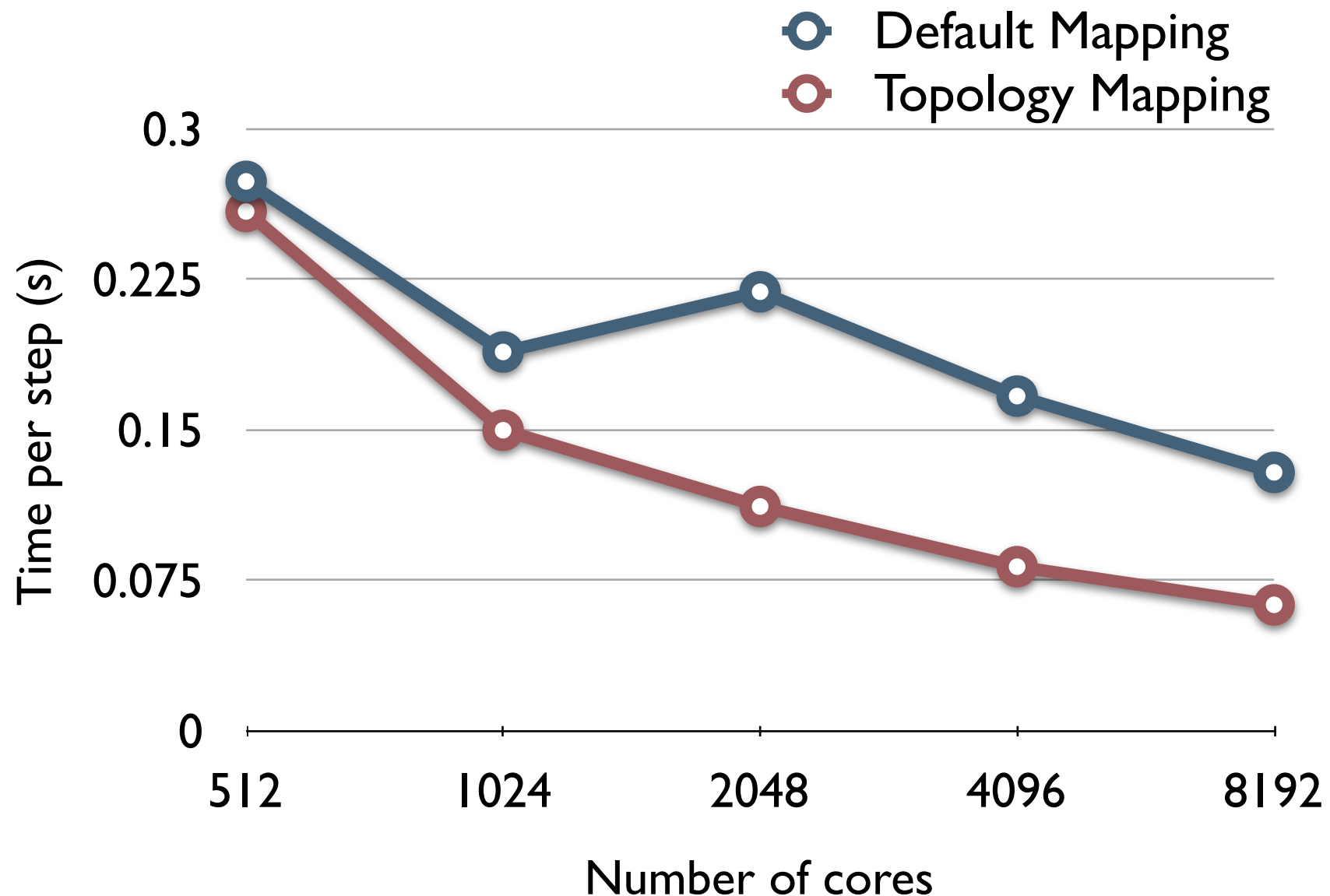
RealSpace and GSpace have state-wise communication

A. Bhatele, E. Bohm, and L.V. Kale. A Case Study of Communication Optimizations on 3D Mesh Interconnects. In Euro-Par, LNCS 5704, pages 1015–1028, 2009. *Distinguished Paper Award, Feng Chen Memorial Best Paper Award*

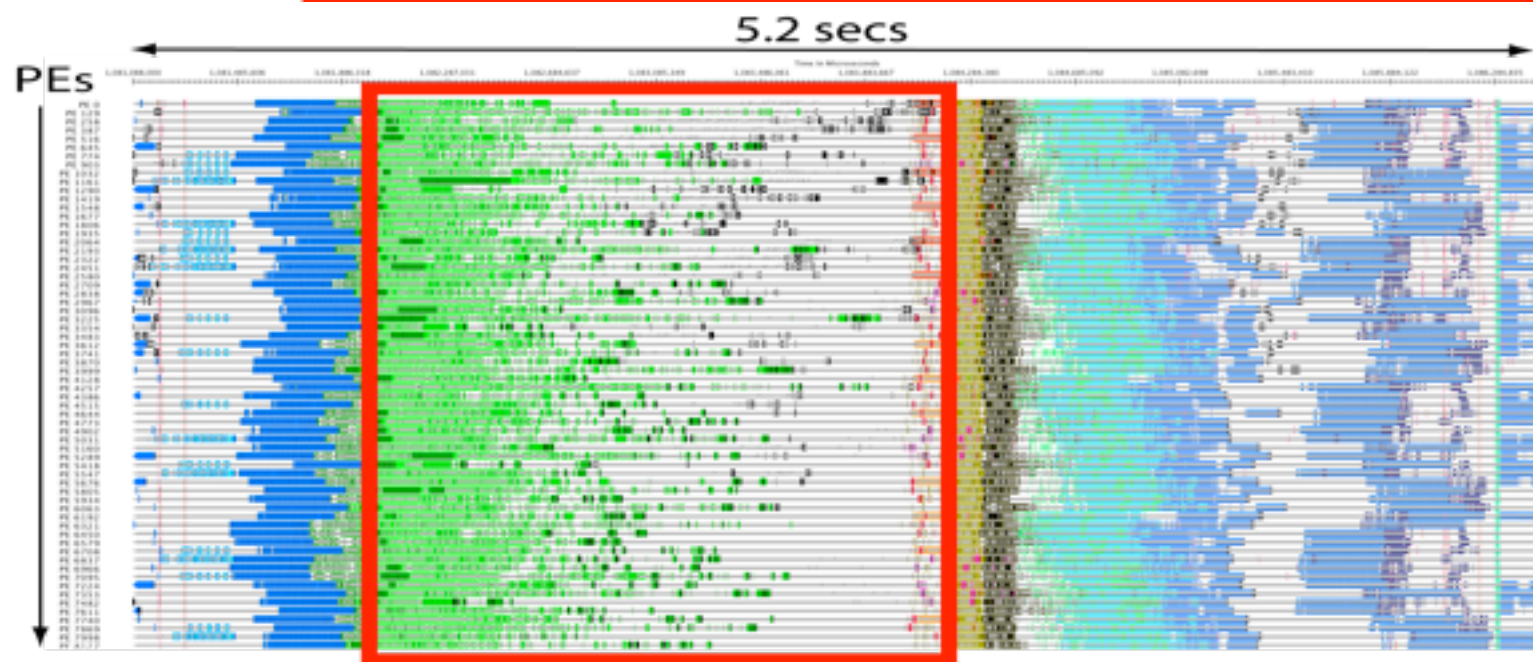
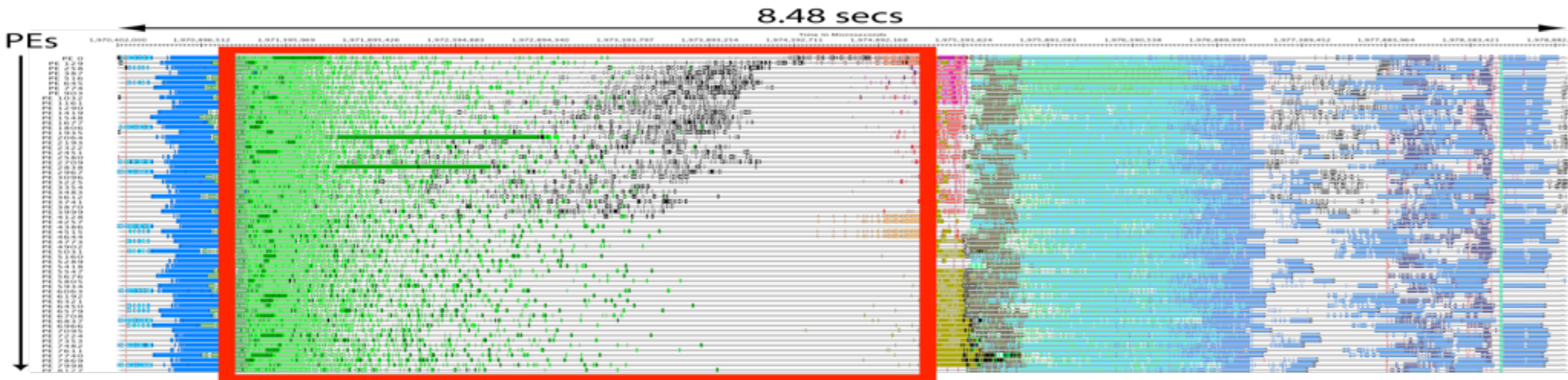


Performance Benefits from Mapping

Performance on Blue Gene/L



Diagnosis of Improvement



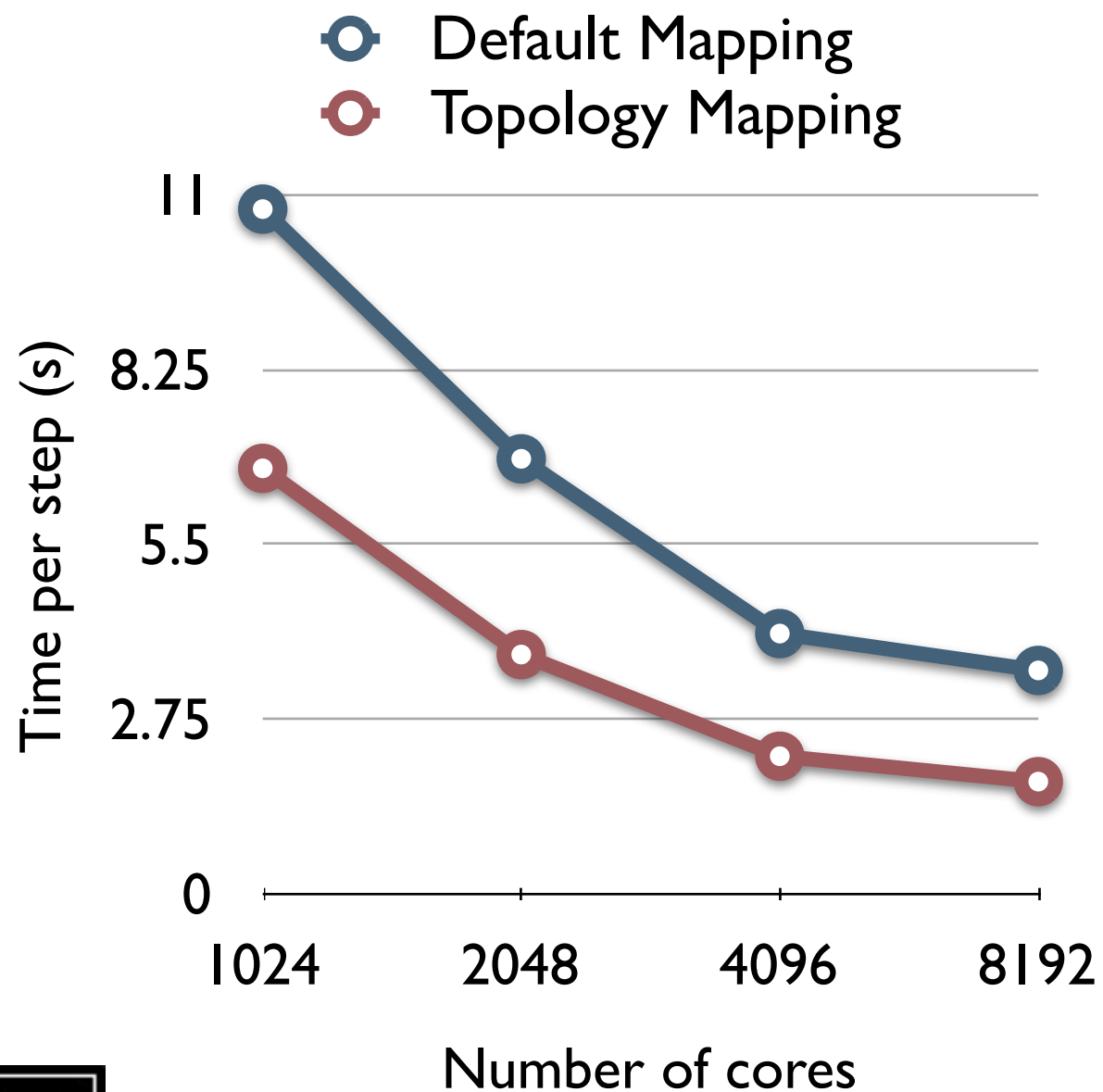
Timeline of 1 iteration of
OpenAtom running
WATER_256M_70Ry on
8192 cores of BG/L

Timeline view using the performance visualization tool, Projections



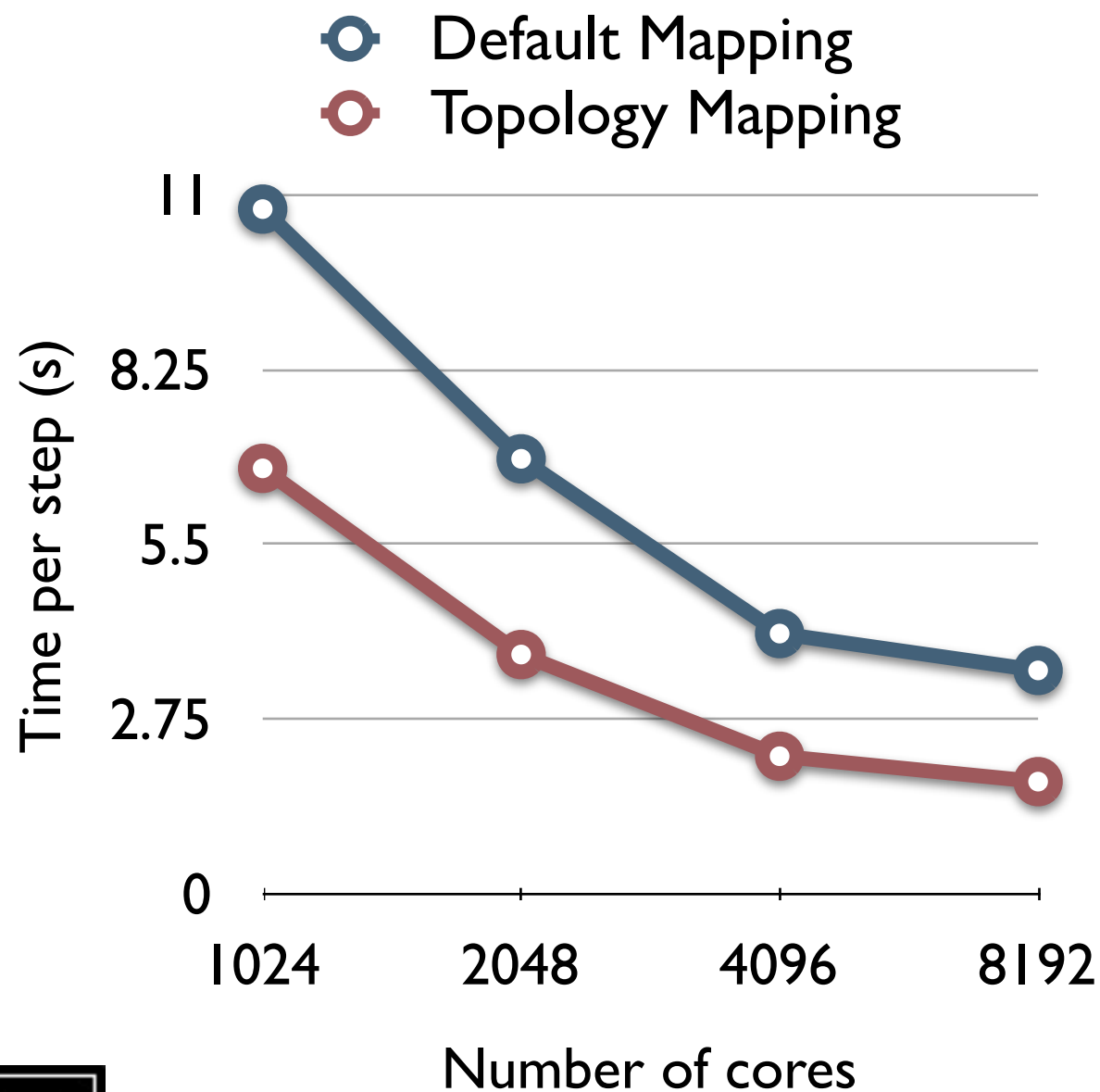
OpenAtom Performance on Blue Gene/P

Application Performance

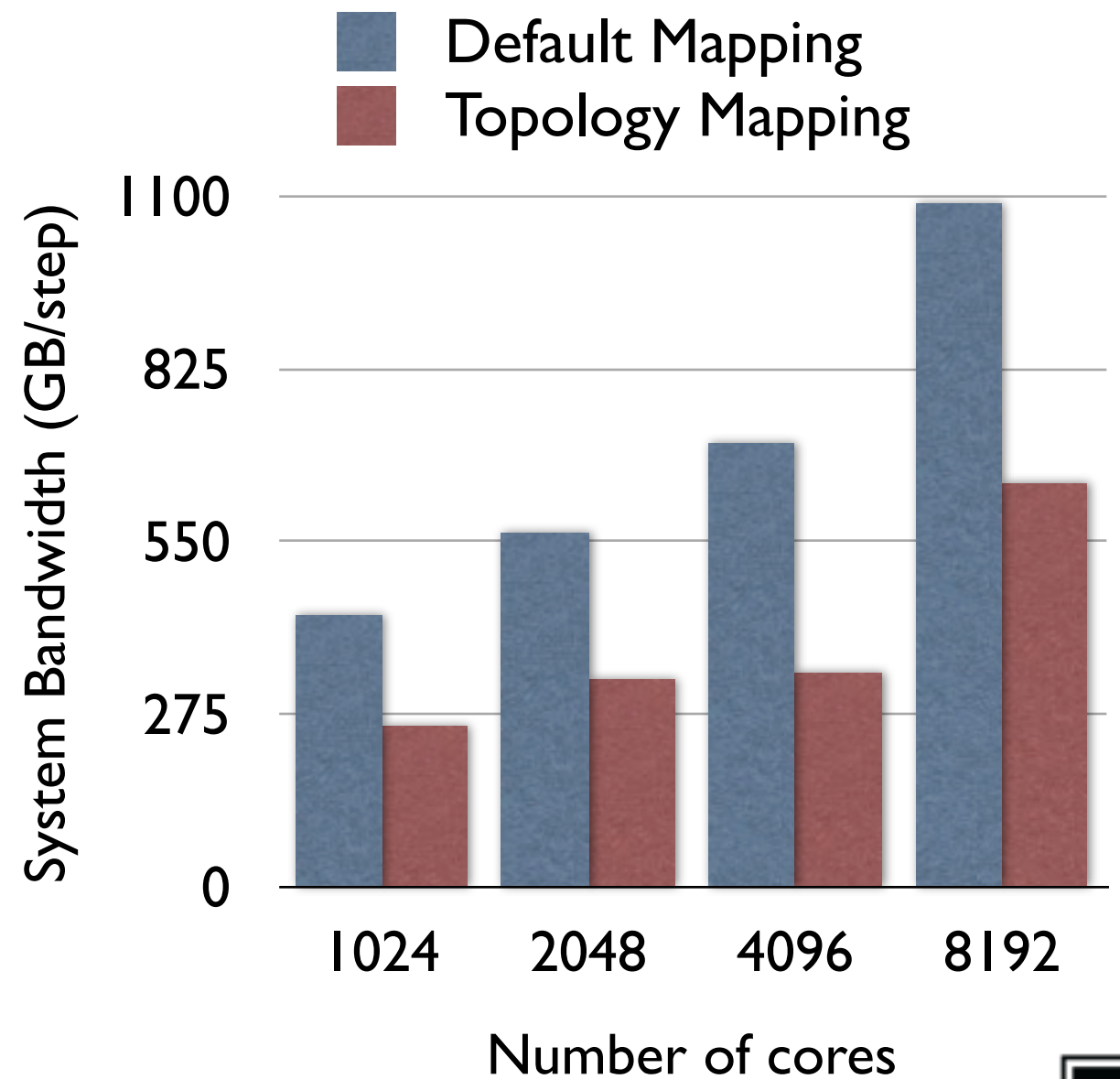


OpenAtom Performance on Blue Gene/P

Application Performance

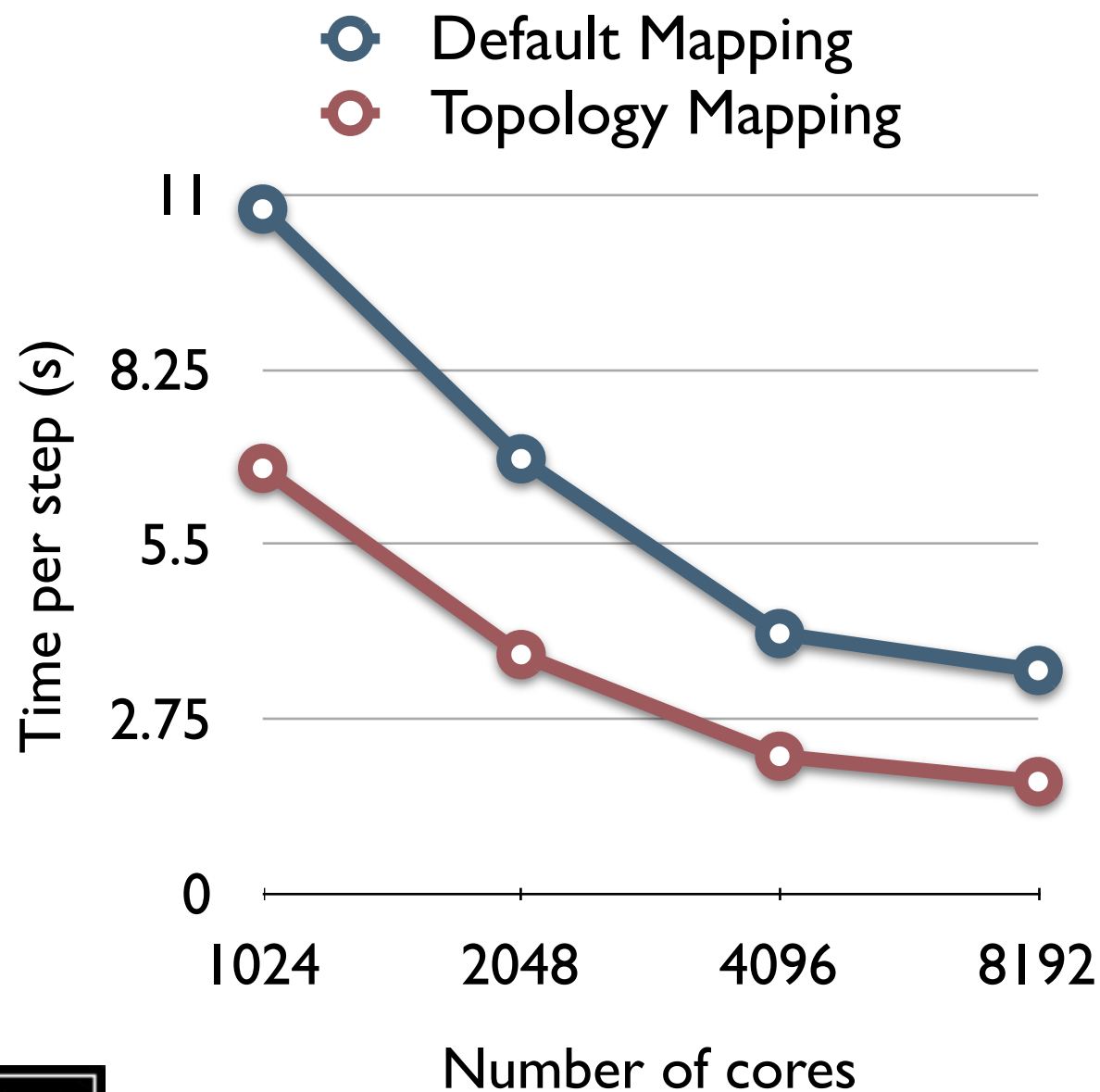


Performance Counters

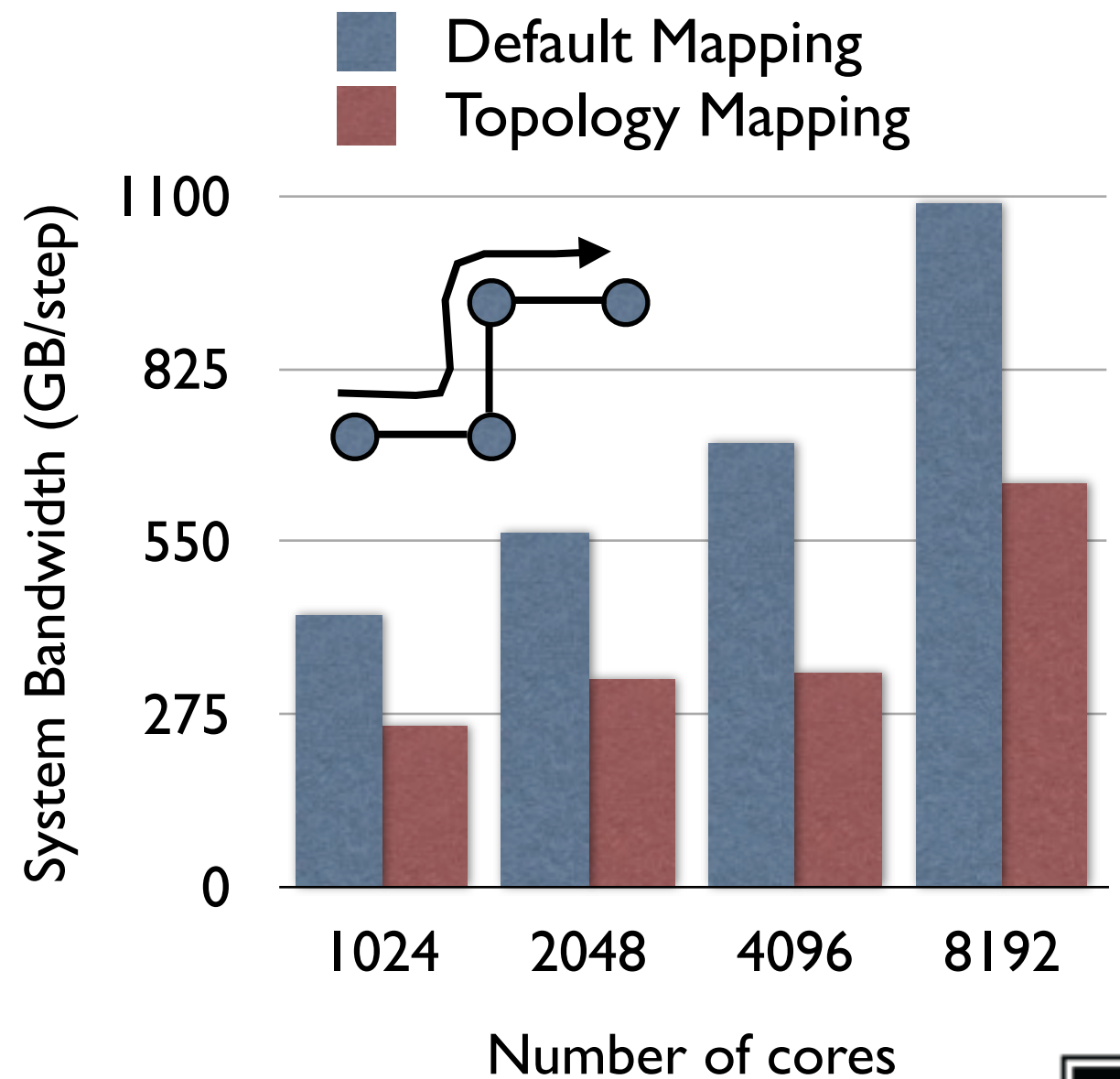


OpenAtom Performance on Blue Gene/P

Application Performance



Performance Counters



OpenAtom Performance on Cray XT3



December 16th, 2010

CNIC @ CAS © Abhinav Bhatele



16

OpenAtom Performance on Cray XT3

- Cray XT3:
 - Link bandwidth - 3.8 GB/s (XT3), 0.425 (BG/P), 0.175 (BG/L)
 - Bytes per flop - 8.77 (XT3), 0.375 (BG/P and BG/L)



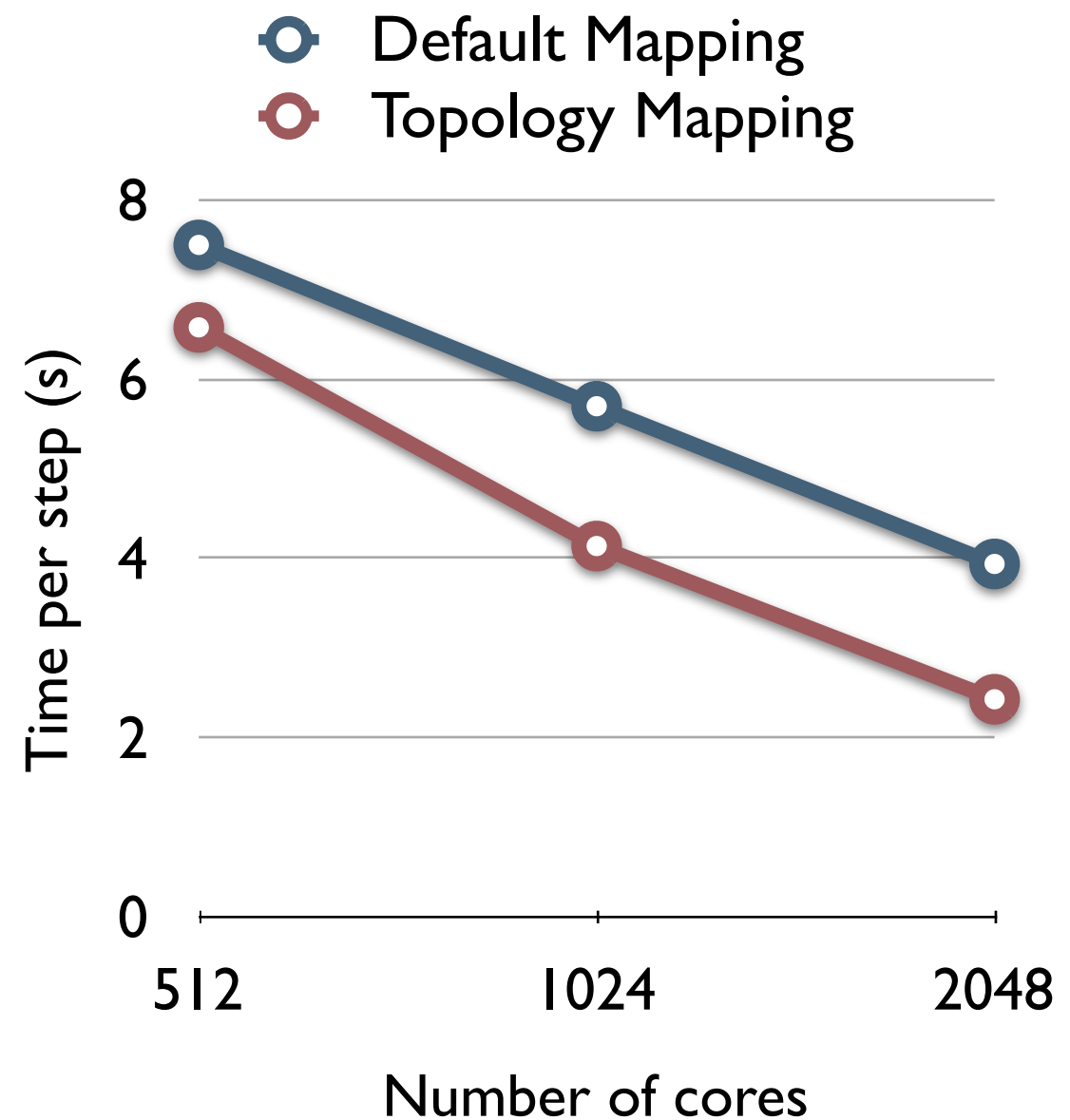
OpenAtom Performance on Cray XT3

- Cray XT3:
 - Link bandwidth - 3.8 GB/s (XT3), 0.425 (BG/P), 0.175 (BG/L)
 - Bytes per flop - 8.77 (XT3), 0.375 (BG/P and BG/L)
- Job schedulers on Cray are not topology aware

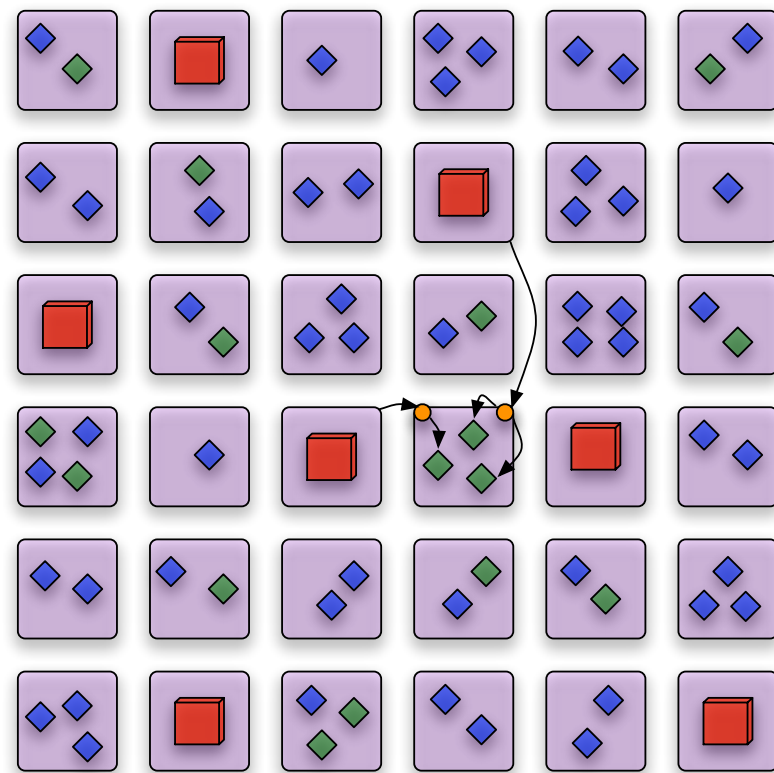


OpenAtom Performance on Cray XT3

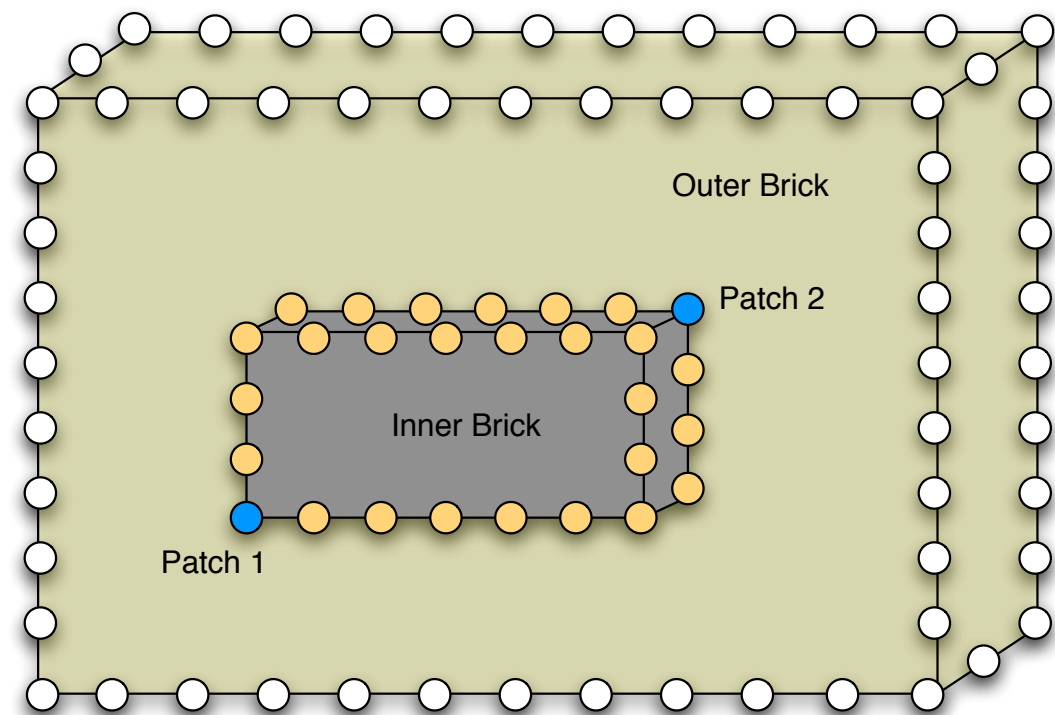
- Cray XT3:
 - Link bandwidth - 3.8 GB/s (XT3), 0.425 (BG/P), 0.175 (BG/L)
 - Bytes per flop - 8.77 (XT3), 0.375 (BG/P and BG/L)
- Job schedulers on Cray are not topology aware
- Performance Benefit at 2048 cores: 40% (XT3), 45% (BG/P), 41% (BG/L)



Case Study II: NAMMD



Communication between patches and computes



Topology aware placement of computes

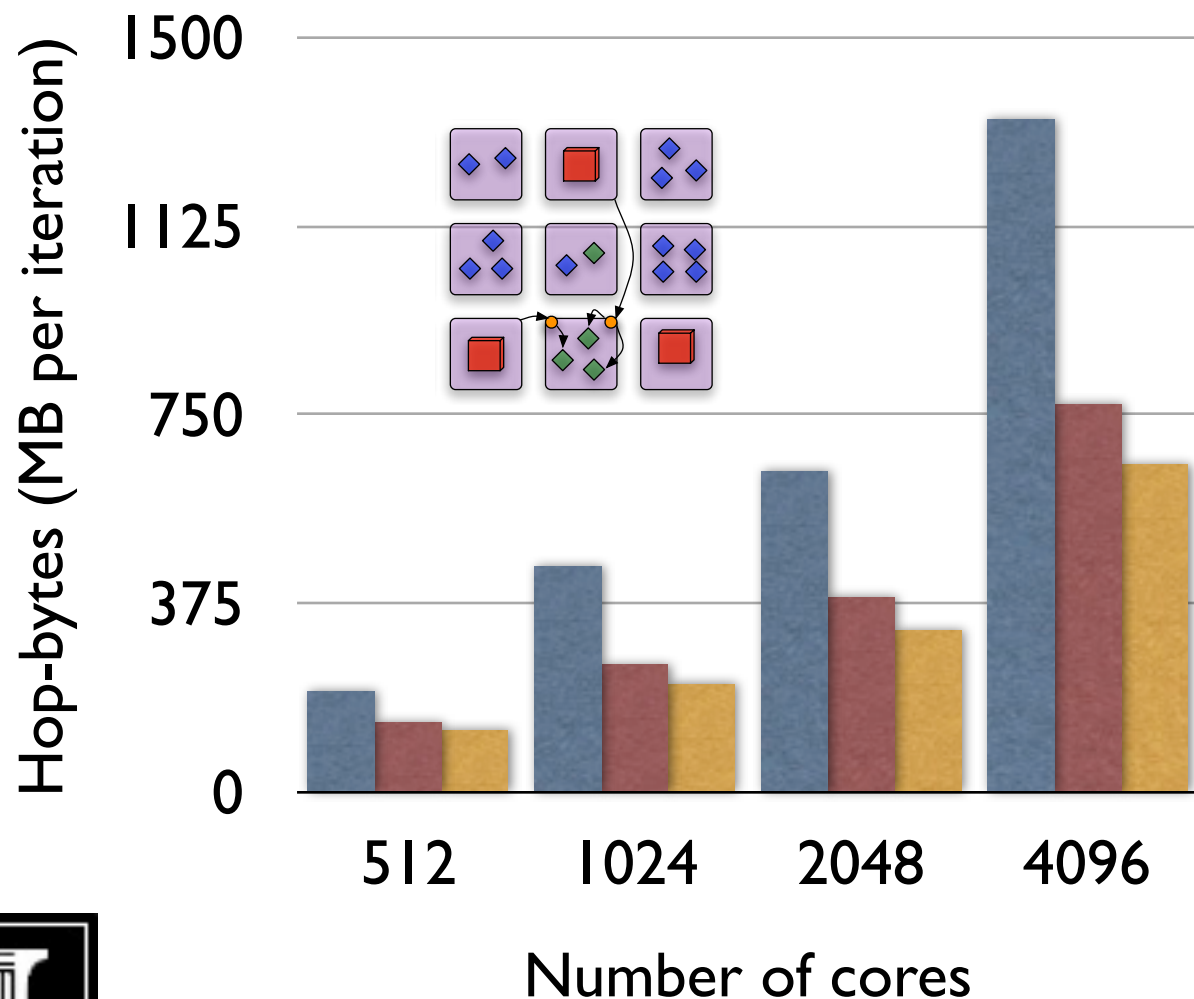
A. Bhatele, L.V. Kale and S. Kumar, Dynamic Topology Aware Load Balancing Algorithms for Molecular Dynamics Applications, In 23rd ACM International Conference on Supercomputing (ICS), 2009.



NAMD Performance on Blue Gene/P

Measured Hop-bytes

- Topology Oblivious
- TopoAware Patches
- TopoAware Computes



- Evaluation Metric:
Hop-bytes

$$HB = \sum_{i=1}^n d_i \times b_i$$

d_i = distance
 b_i = bytes
 n = no. of messages

- Indicates amount of traffic and hence contention on the network
- Previously used metric: maximum dilation

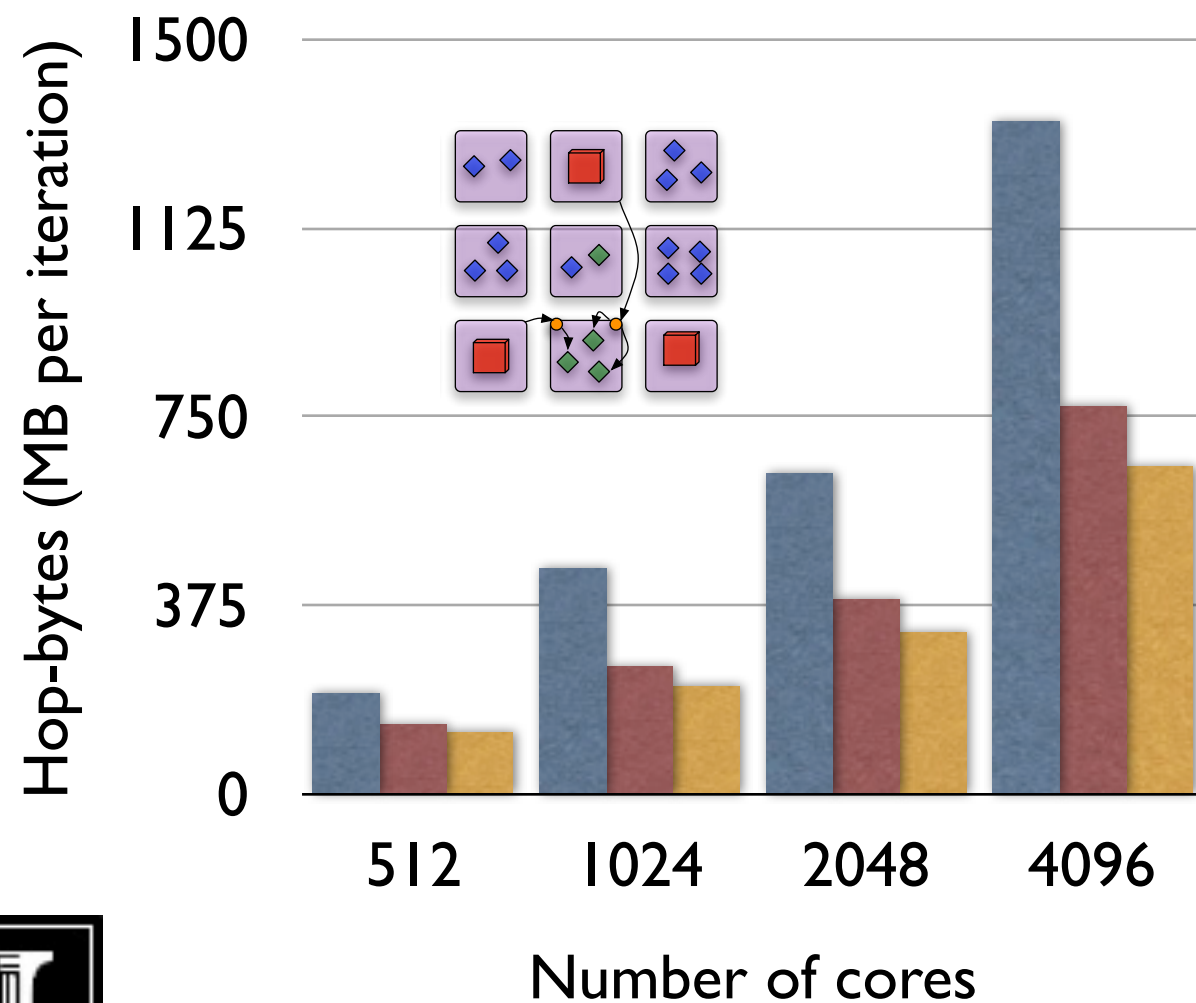
$$d(e) = \max\{d_i | e_i \in E\}$$



NAMD Performance on Blue Gene/P

Measured Hop-bytes

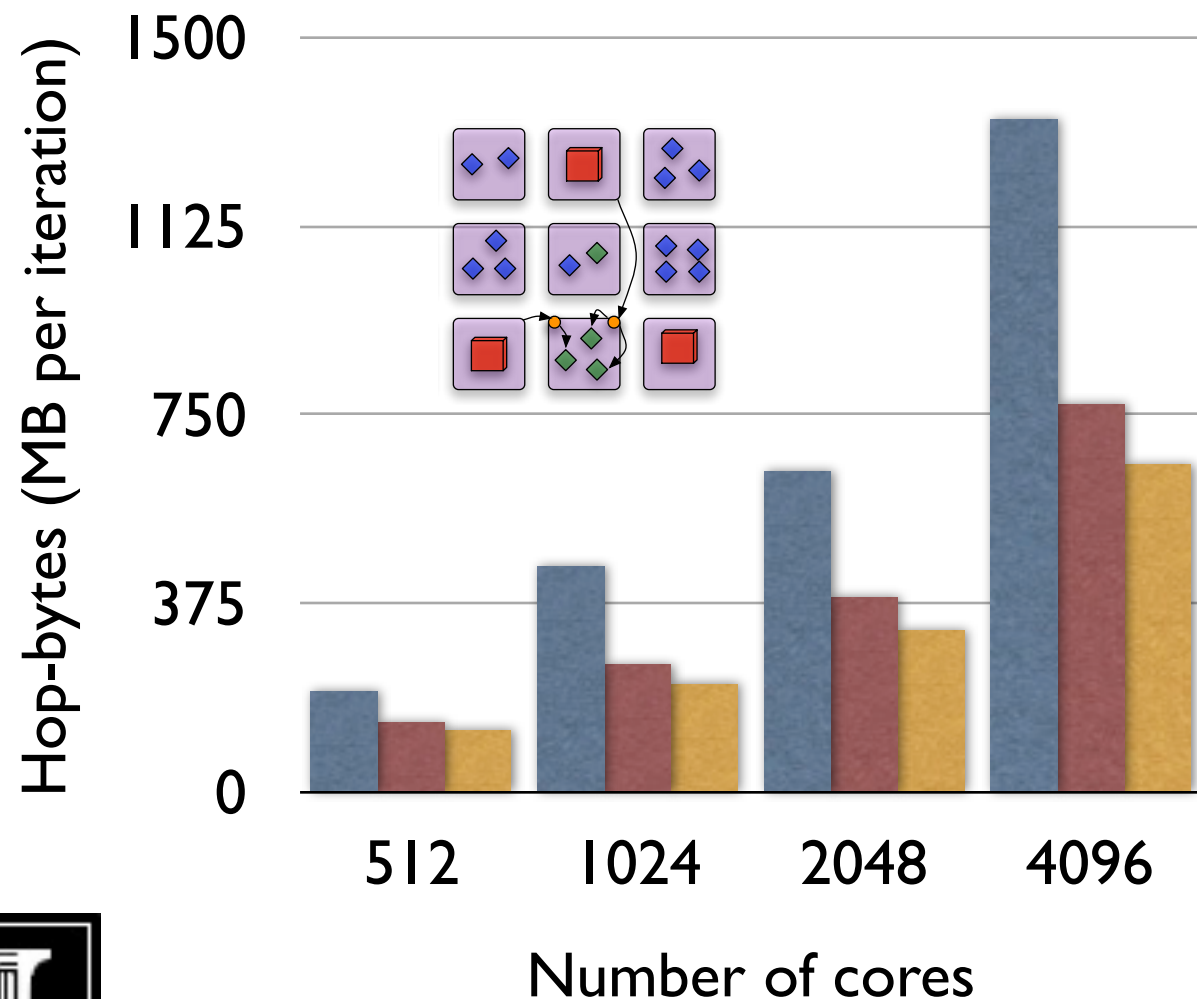
- Topology Oblivious
- TopoAware Patches
- TopoAware Computes



NAMD Performance on Blue Gene/P

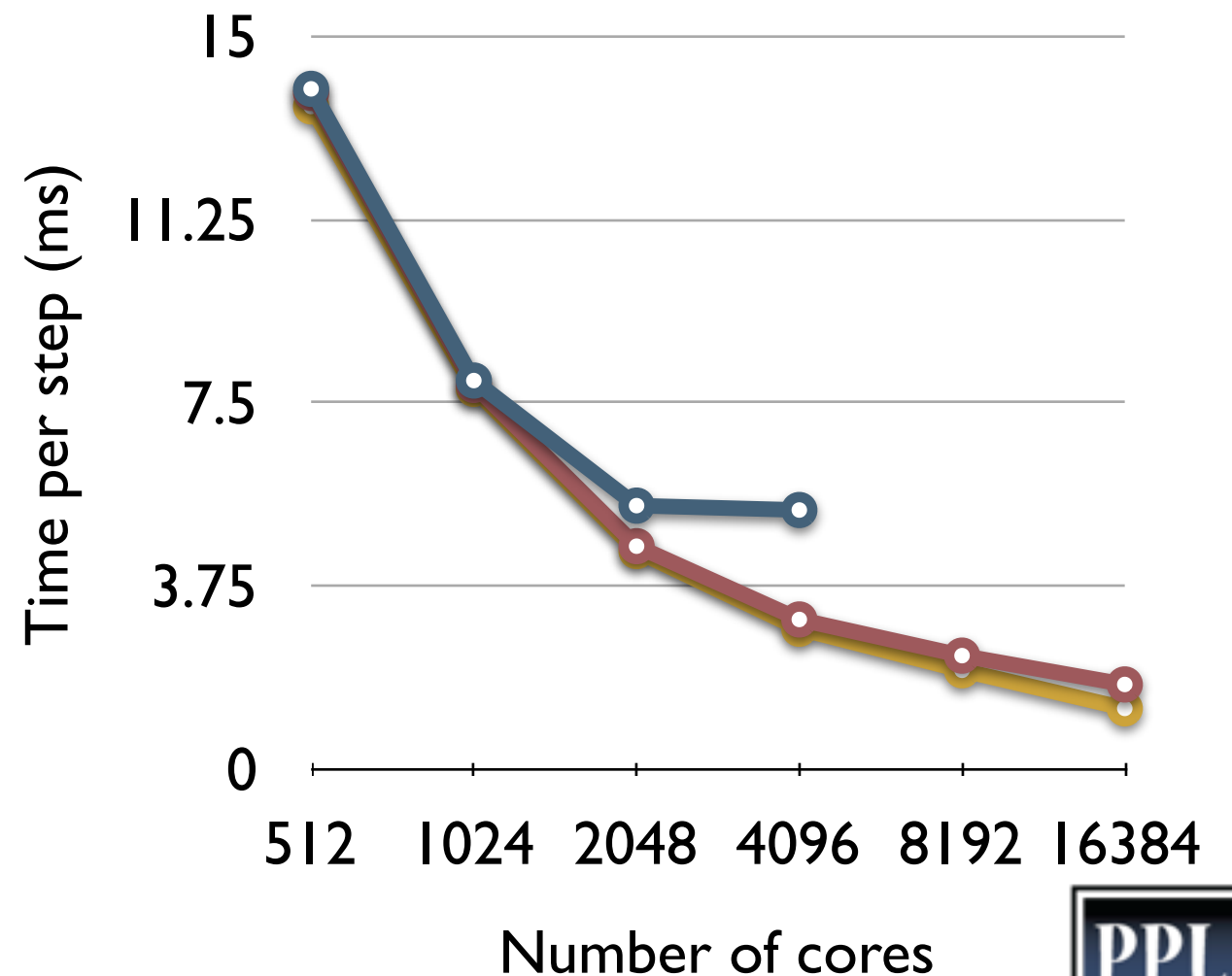
Measured Hop-bytes

- Topology Oblivious
- TopoAware Patches
- TopoAware Computes



Application Performance

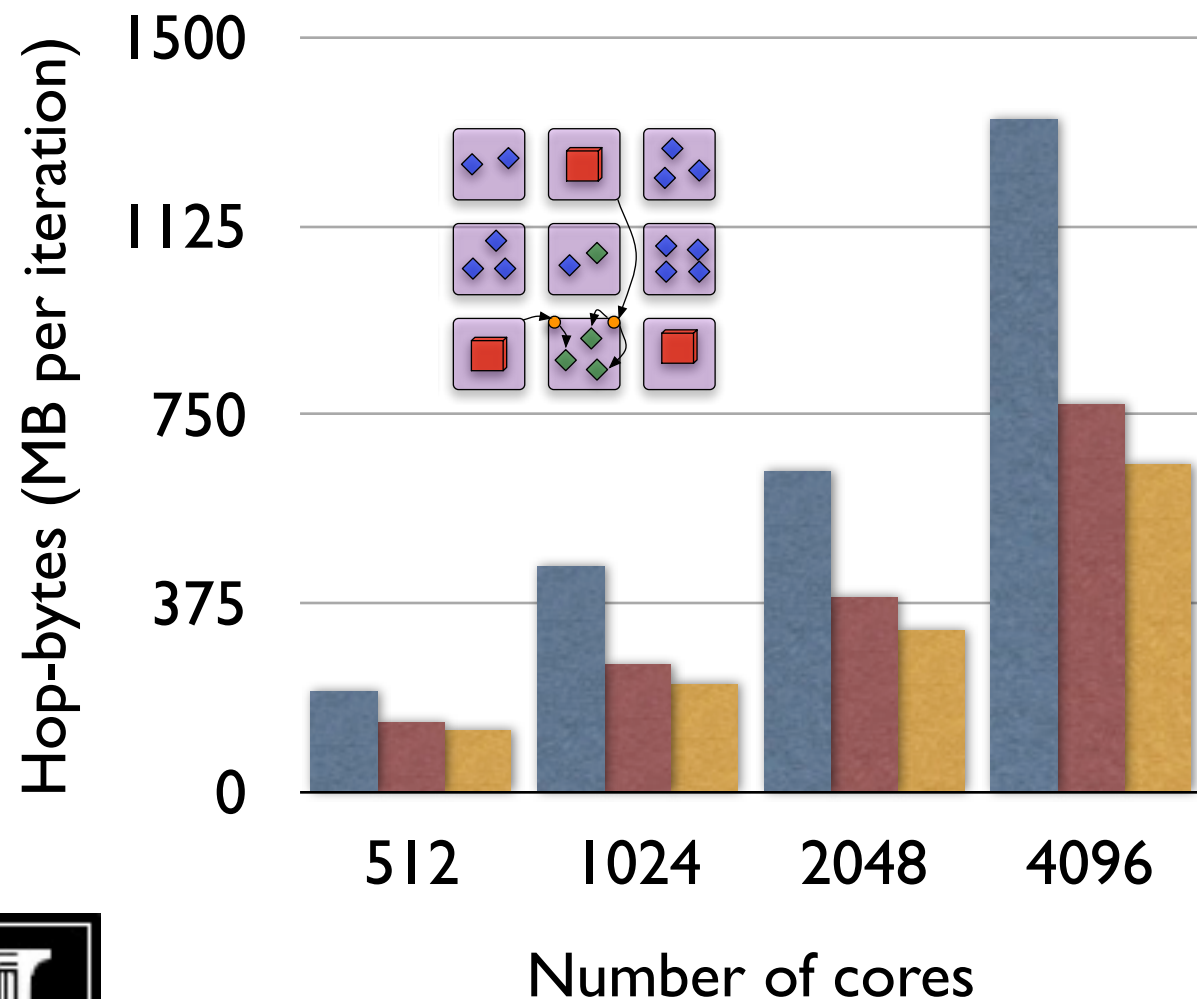
- Topology Oblivious
- TopoAware Patches
- TopoAware Computes



NAMD Performance on Blue Gene/P

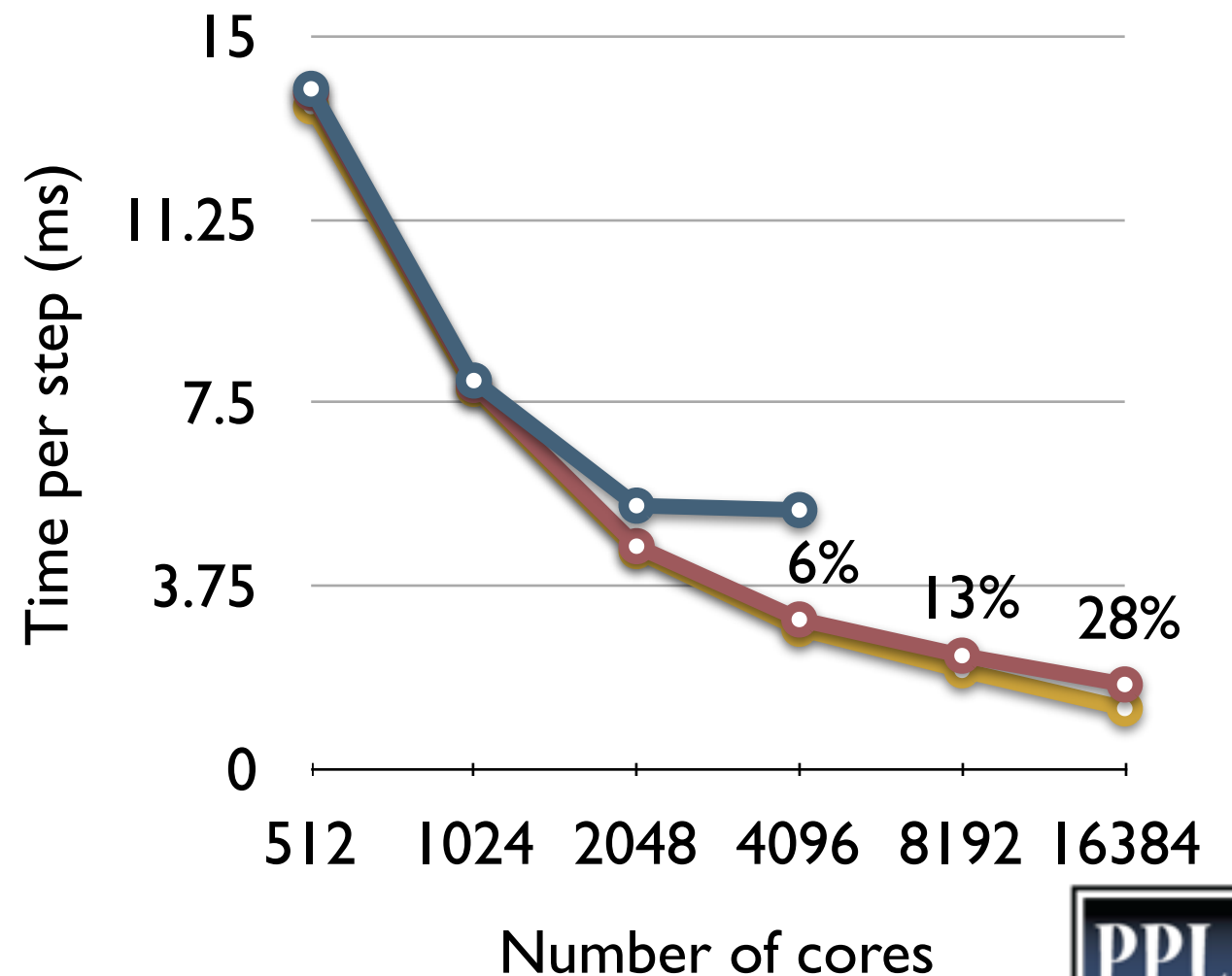
Measured Hop-bytes

- Topology Oblivious
- TopoAware Patches
- TopoAware Computes



Application Performance

- Topology Oblivious
- TopoAware Patches
- TopoAware Computes

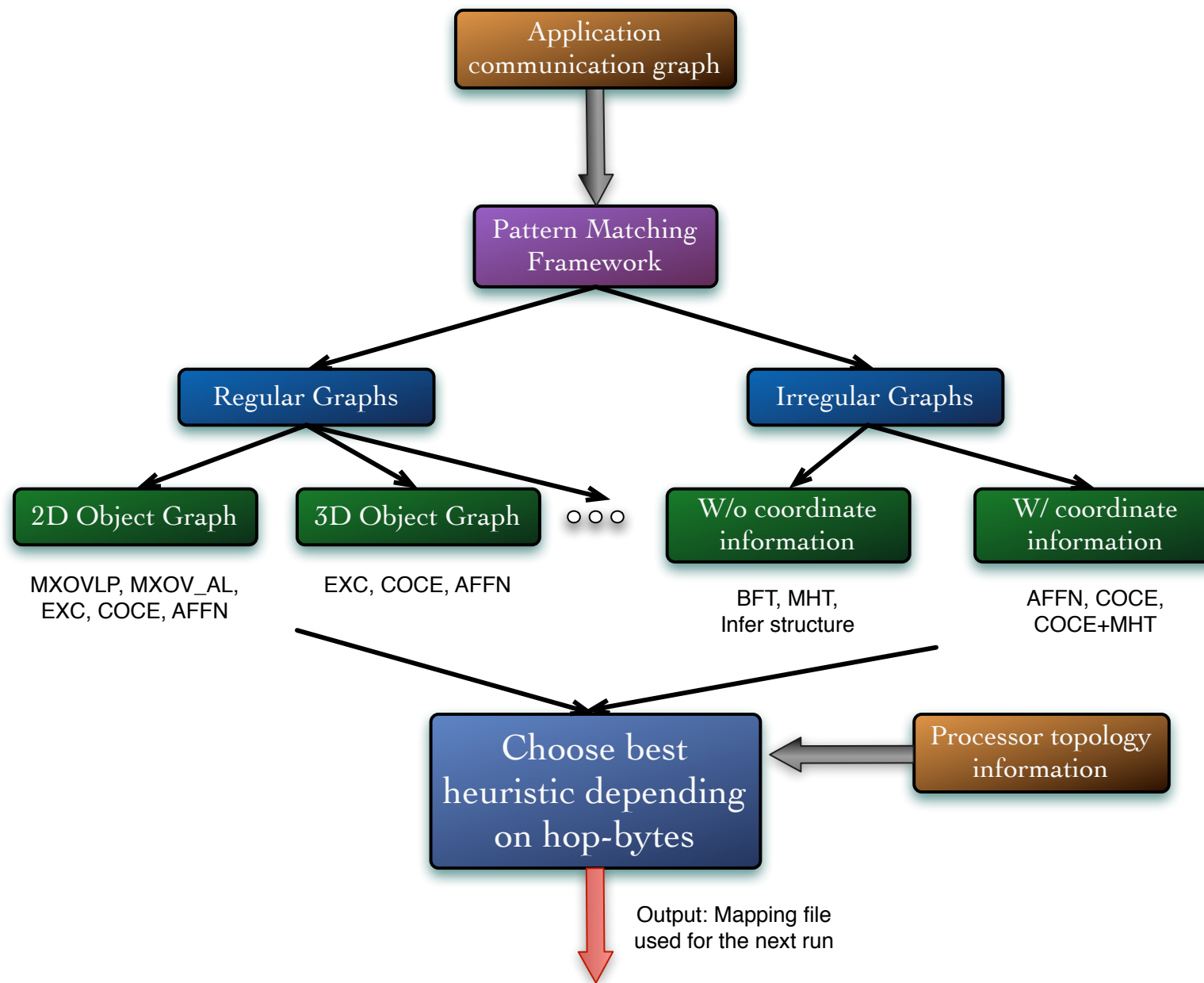


Outline

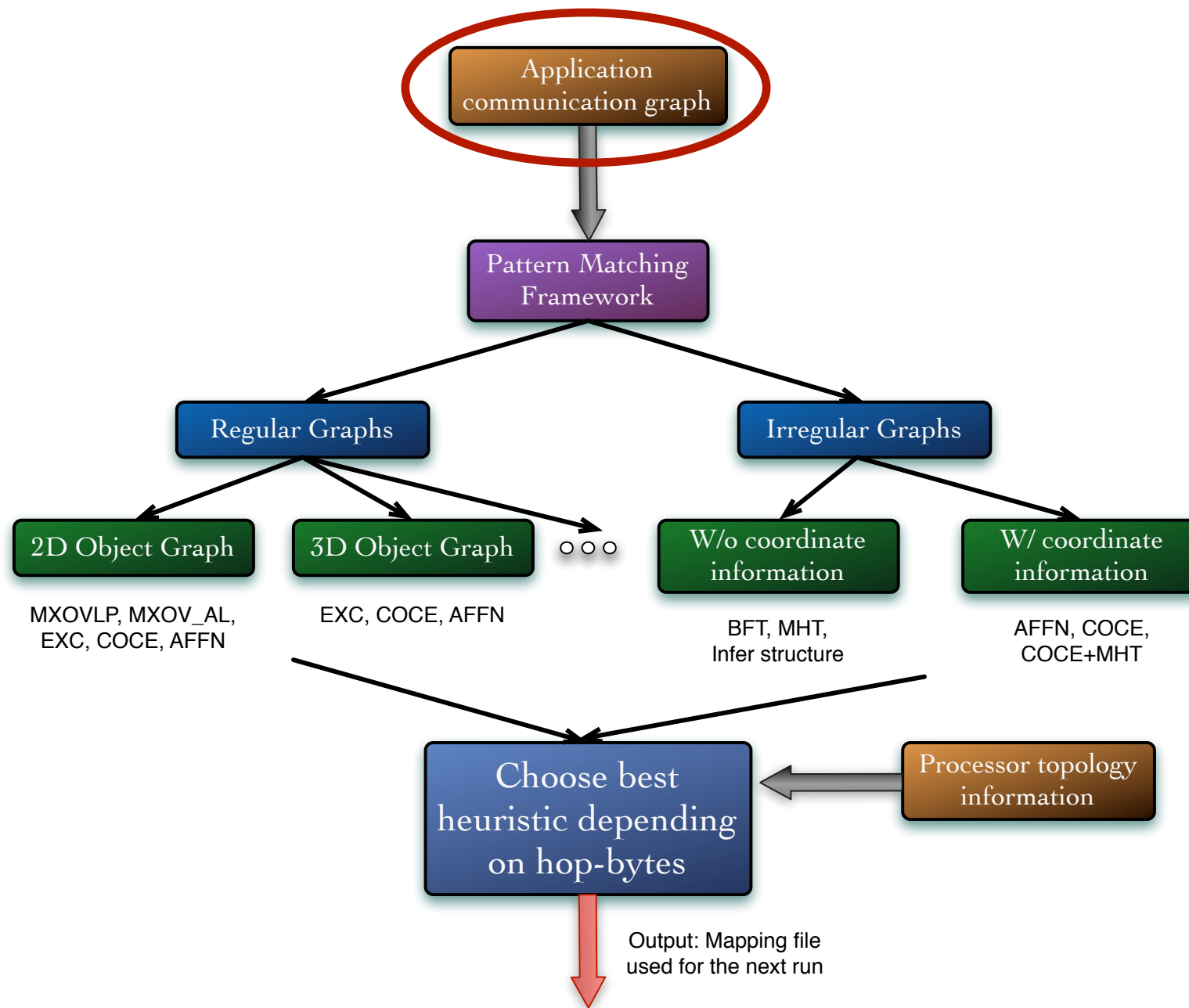
- Case studies:
 - OpenAtom
 - NAMD
- Automatic Mapping Framework
 - Pattern matching
- Heuristics for Regular Graphs
- Heuristics for Irregular Graphs



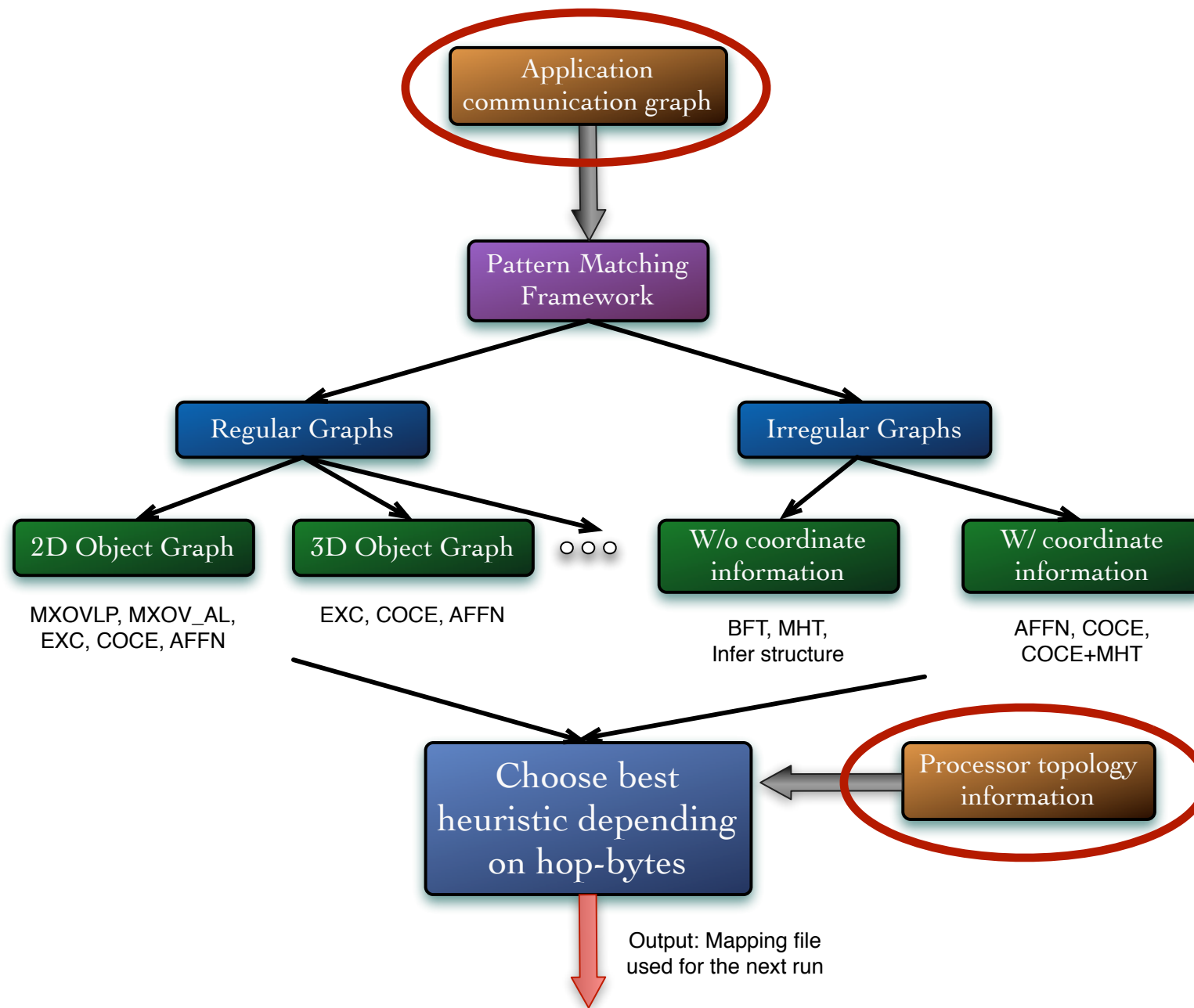
Automatic Mapping Framework



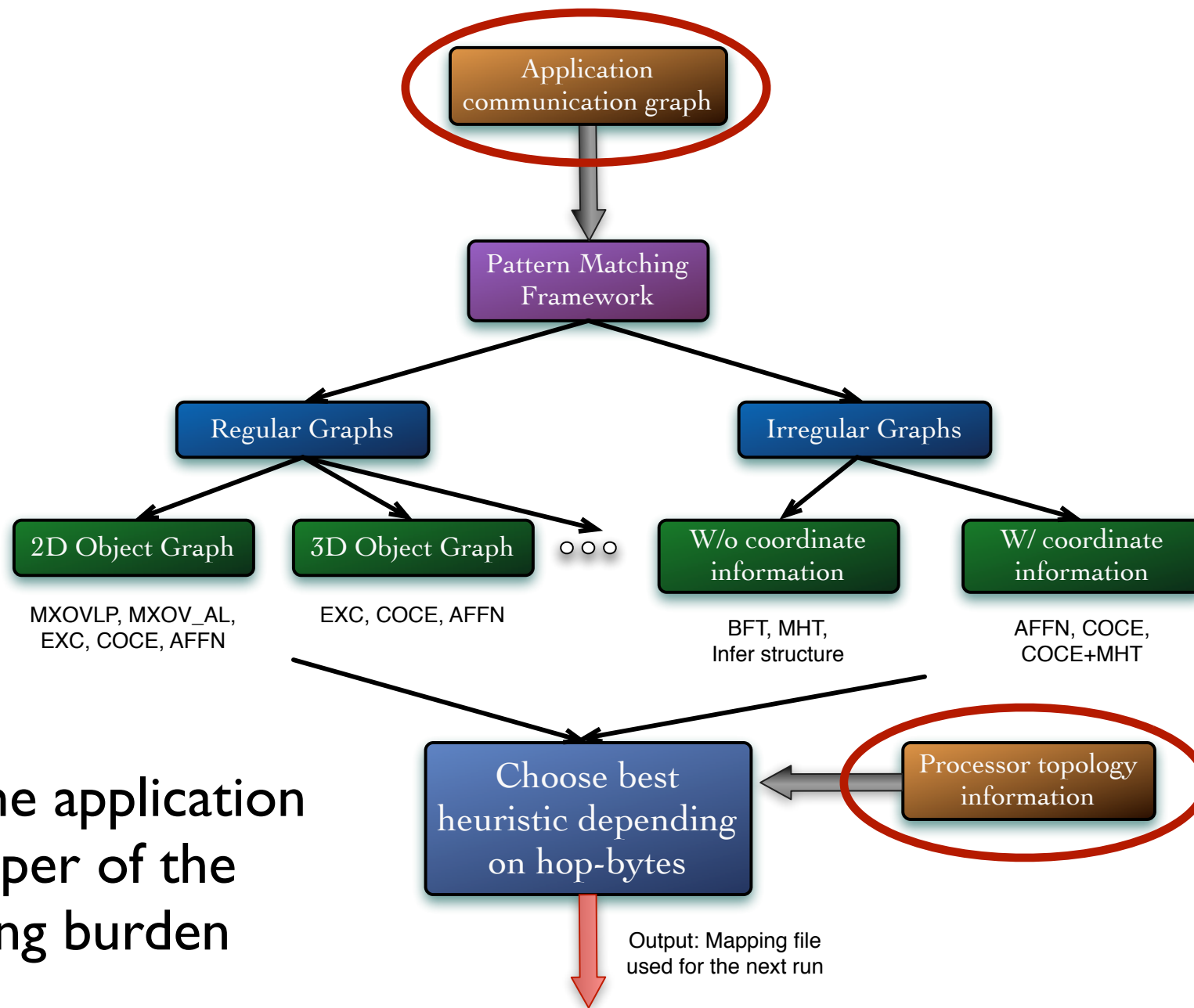
Automatic Mapping Framework



Automatic Mapping Framework



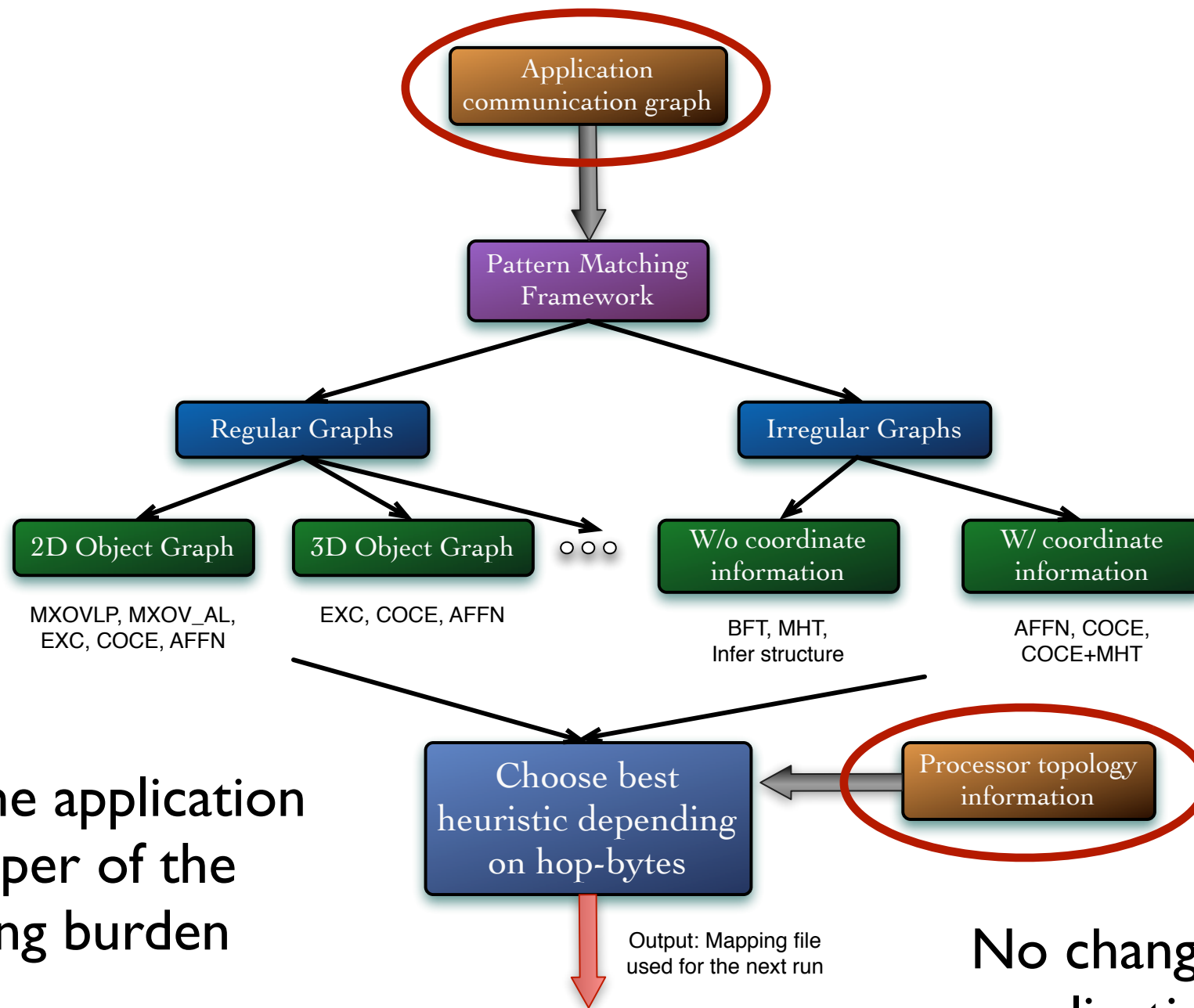
Automatic Mapping Framework



Relieve the application developer of the mapping burden



Automatic Mapping Framework



Relieve the application developer of the mapping burden

No change to the application code



Topology Discovery

- Topology Manager API: for 3D interconnects (Blue Gene, XT)
- Information required for mapping:
 - Physical dimensions of the allocated job partition
 - Mapping of ranks to physical coordinates and vice versa
- On Blue Gene machines such information is available and the API is a wrapper
- On Cray XT machines, jump several hoops to get this information and make it available through the same API



Application communication graph

- Several ways to obtain the graph
- MPI applications:
 - Profiling tools (IBM's HPCT tools)
 - Collect information using the PMPI interface
 - Manually provided by the application end user
- Charm++ applications:
 - Instrumentation at runtime
 - Profiling tools (HPCT): when $n = p$



Pattern Matching

- We want to identify regular 2D/3D communication patterns

Input: $CM_{n,n}$ (communication matrix)

Output: $isRegular$ (boolean, true if communication is regular)
 $dims[]$ (dimensions of the regular communication graph)

for $i = 1$ to n **do**

 find the maximum number of neighbors for any rank in $CM_{i,n}$

end for

if max neighbors ≤ 5 **then**

 // this might be a case of regular 2D communication

 select an arbitrary rank $start_{pe}$ find its distance from its neighbors

$dist$ = difference between ranks of $start_{pe}$ and its top or bottom neighbor

for $i := 1$ to n **do**

if distance of all ranks from their neighbors $== 1$ or $dist$ **then**

$isRegular = true$

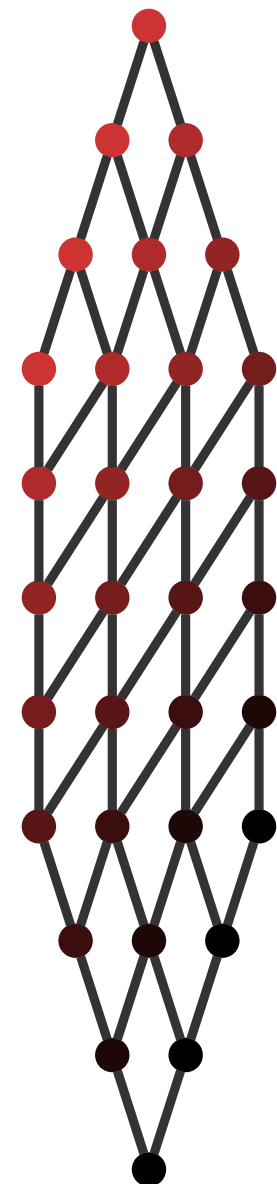
$dim[0] = dist$

$dim[1] = n/dist$

end if

end for

end if



Pattern Matching

- We want to identify regular 2D/3D communication patterns

Input: $CM_{n,n}$ (communication matrix)

Output: $isRegular$ (boolean, true if communication is regular)
 $dims[]$ (dimensions of the regular communication graph)

for $i = 1$ to n **do**

 find the maximum number of neighbors for any rank in $CM_{i,n}$

end for

if max neighbors ≤ 5 **then**

 // this might be a case of regular 2D communication

 select an arbitrary rank $start_{pe}$ find its distance from its neighbors

$dist$ = difference between ranks of $start_{pe}$ and its top or bottom neighbor

for $i := 1$ to n **do**

if distance of all ranks from their neighbors $== 1$ or $dist$ **then**

$isRegular = true$

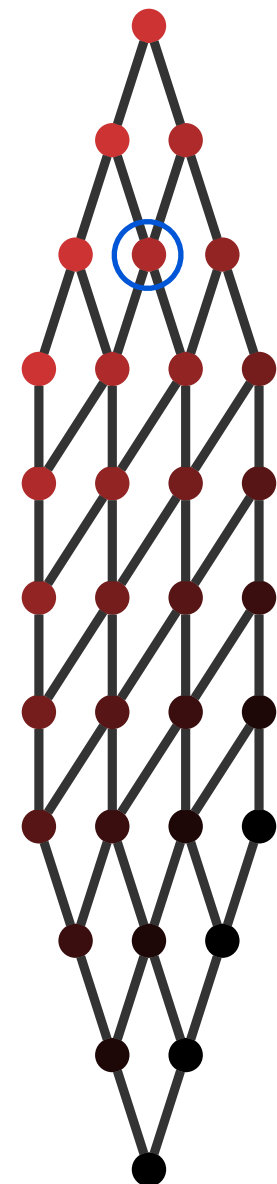
$dim[0] = dist$

$dim[1] = n/dist$

end if

end for

end if



Pattern Matching

- We want to identify regular 2D/3D communication patterns

Input: $CM_{n,n}$ (communication matrix)

Output: $isRegular$ (boolean, true if communication is regular)
 $dims[]$ (dimensions of the regular communication graph)

for $i = 1$ to n **do**

 find the maximum number of neighbors for any rank in $CM_{i,n}$

end for

if max neighbors ≤ 5 **then**

 // this might be a case of regular 2D communication

 select an arbitrary rank $start_{pe}$ find its distance from its neighbors

$dist$ = difference between ranks of $start_{pe}$ and its top or bottom neighbor

for $i := 1$ to n **do**

if distance of all ranks from their neighbors $== 1$ or $dist$ **then**

$isRegular = true$

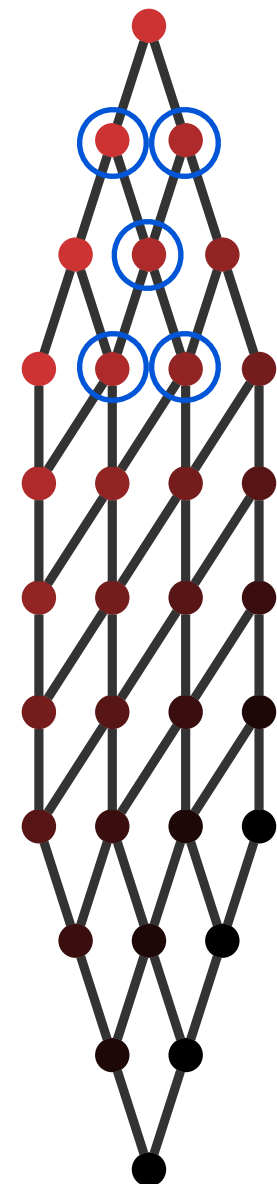
$dim[0] = dist$

$dim[1] = n/dist$

end if

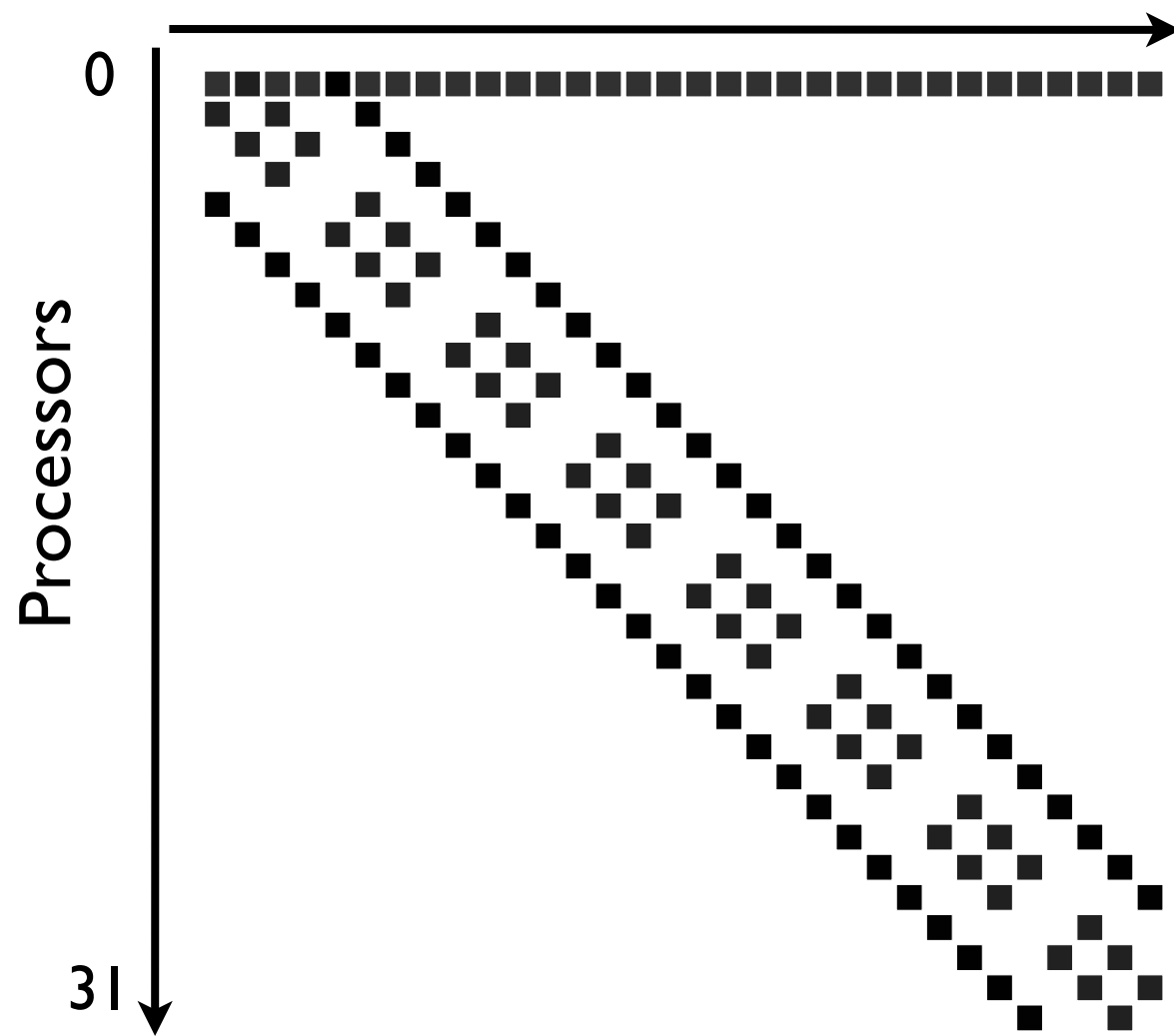
end for

end if



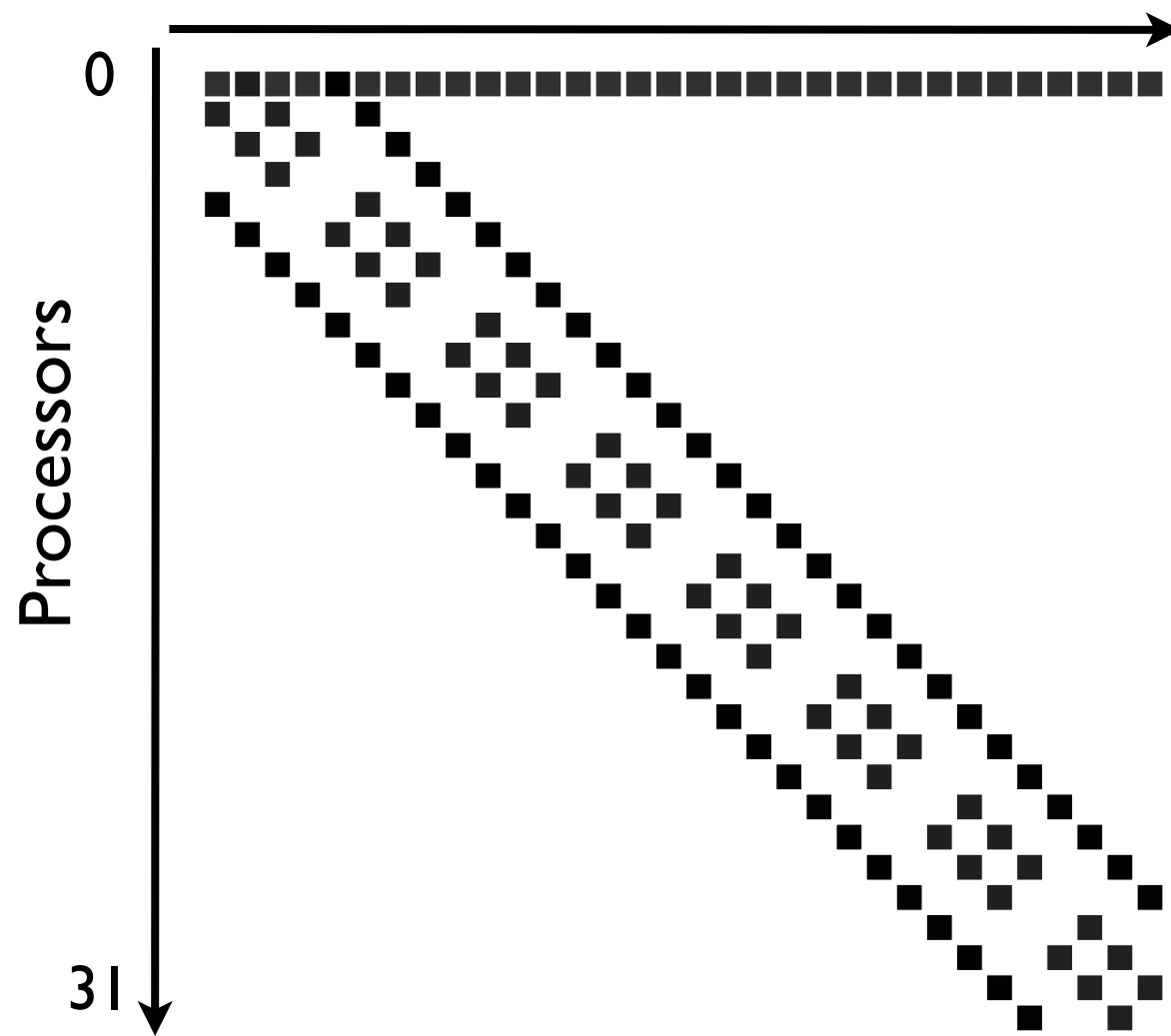
Example

- WRF running on 32 cores of Blue Gene/P



Example

- WRF running on 32 cores of Blue Gene/P

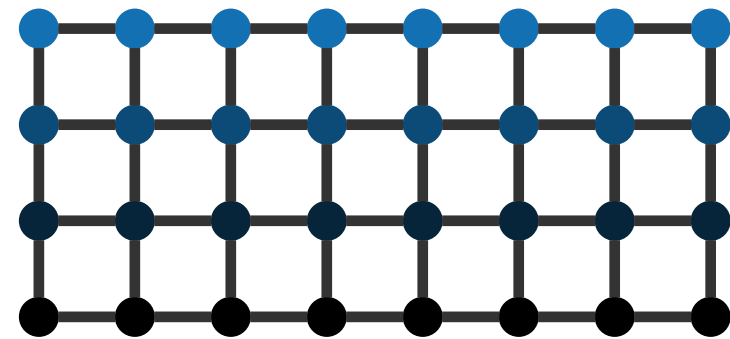
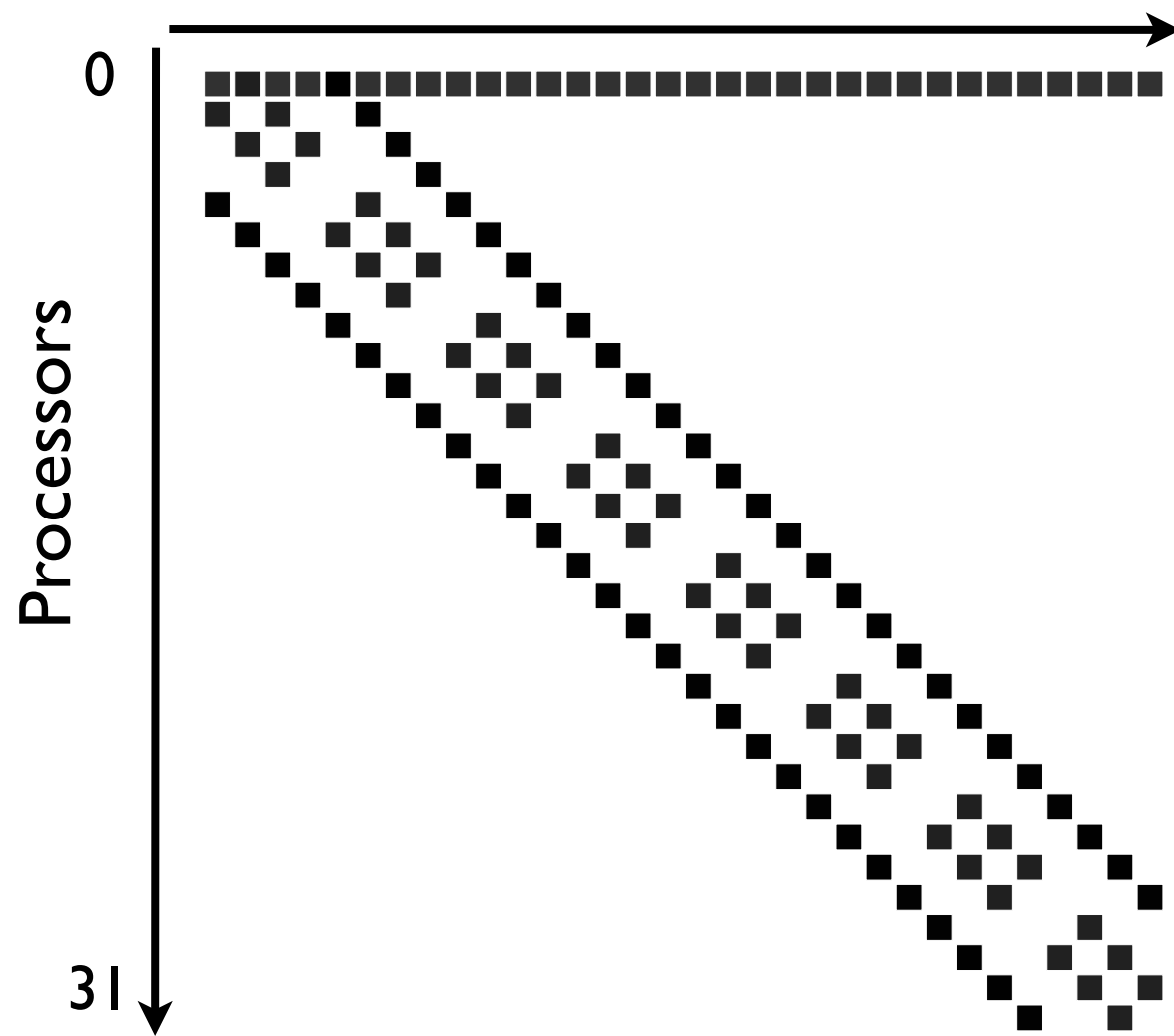


Pattern matching to identify regular communication patterns such as 2D/3D near-neighbor graphs



Example

- WRF running on 32 cores of Blue Gene/P



Pattern matching to identify regular communication patterns such as 2D/3D near-neighbor graphs



Communication Graphs

- Regular communication:
 - POP (Parallel Ocean Program): 2D Stencil like computation
 - WRF (Weather Research and Forecasting model): 2D Stencil
 - MILC (MIMD Lattice Computation): 4D near-neighbor
- Irregular communication:
 - Unstructured mesh computations: FLASH, CPSD code
 - Many other classes of applications



Outline

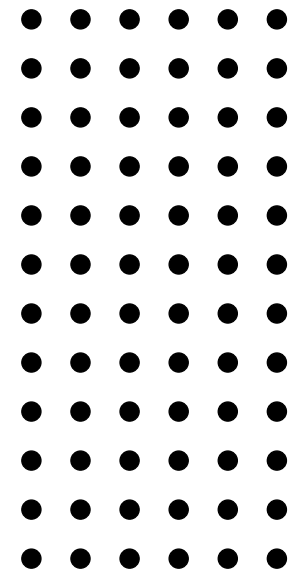
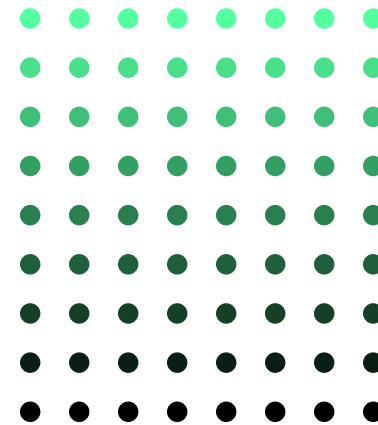
- Case studies:
 - OpenAtom
 - NAMD
- Automatic Mapping Framework
 - Pattern matching
- Heuristics for Regular Graphs
- Heuristics for Irregular Graphs



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

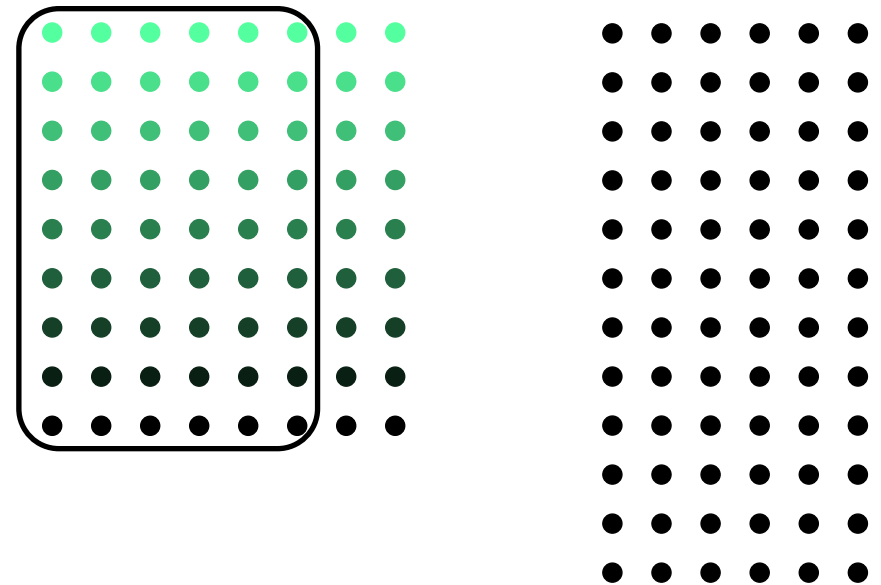
Object Graph: 9 x 8
Processor Graph: 12 x 6



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

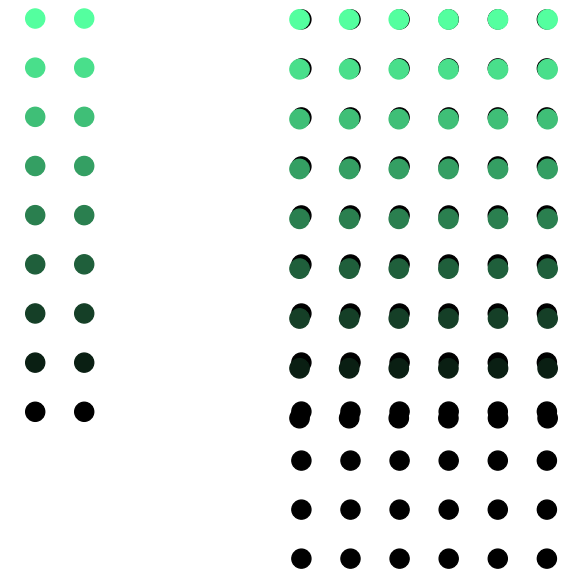
Object Graph: 9 x 8
Processor Graph: 12 x 6



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

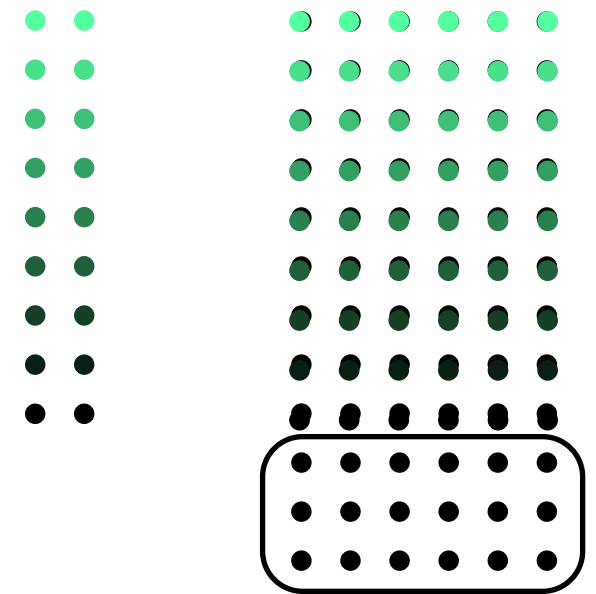
Object Graph: 9 x 8
Processor Graph: 12 x 6



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

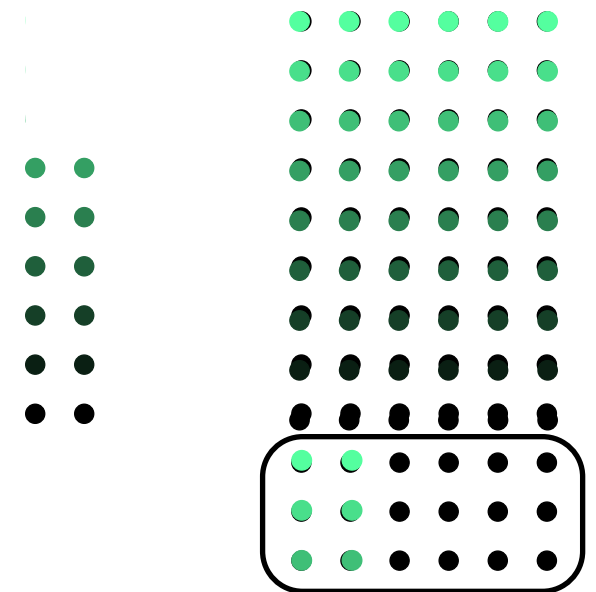
Object Graph: 9 x 8
Processor Graph: 12 x 6



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

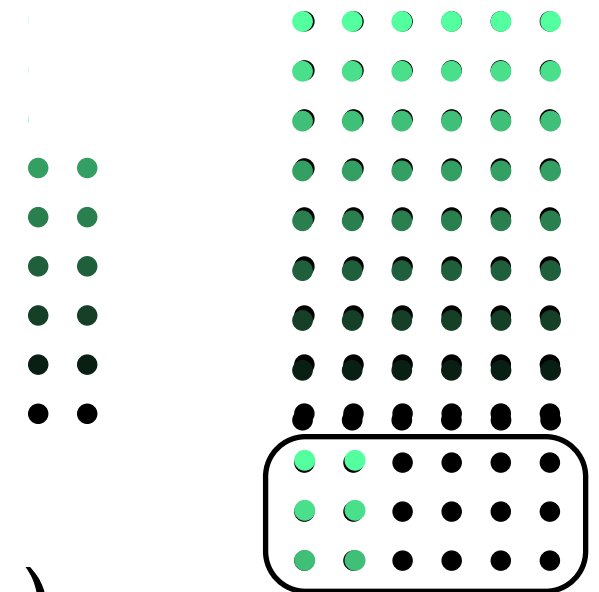
Object Graph: 9 x 8
Processor Graph: 12 x 6



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

Object Graph: 9 x 8
Processor Graph: 12 x 6



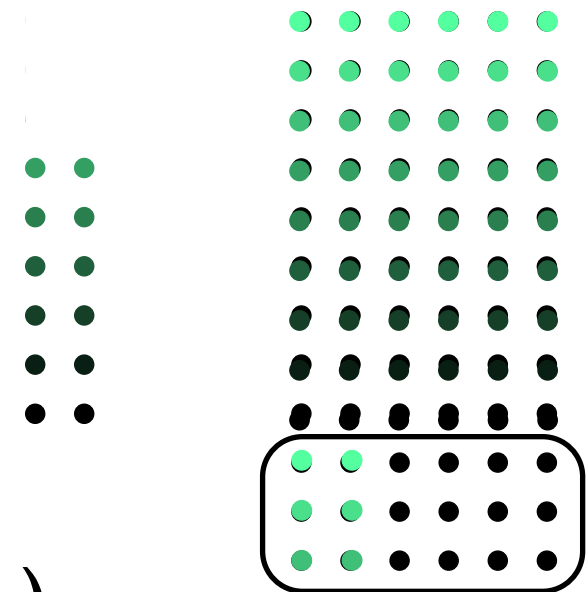
- Maximum Overlap with Alignment (MXOV+AL)
 - Alignment at each recursive call



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

Object Graph: 9 x 8
Processor Graph: 12 x 6



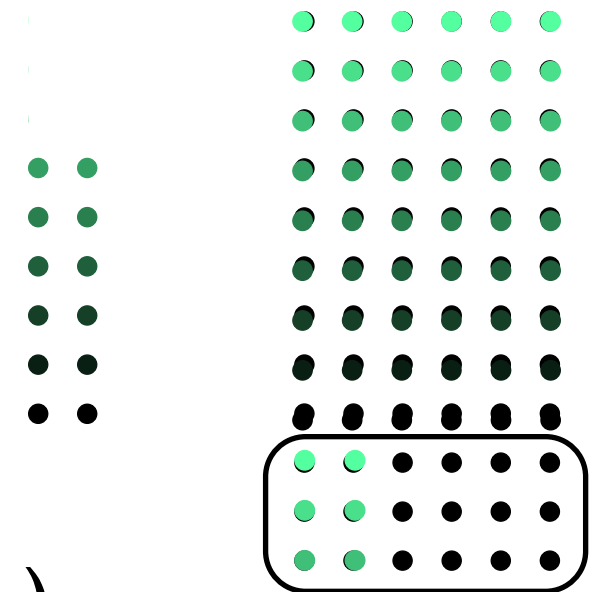
- Maximum Overlap with Alignment (MXOV+AL)
 - Alignment at each recursive call
- Expand from Corner (EXCO)



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

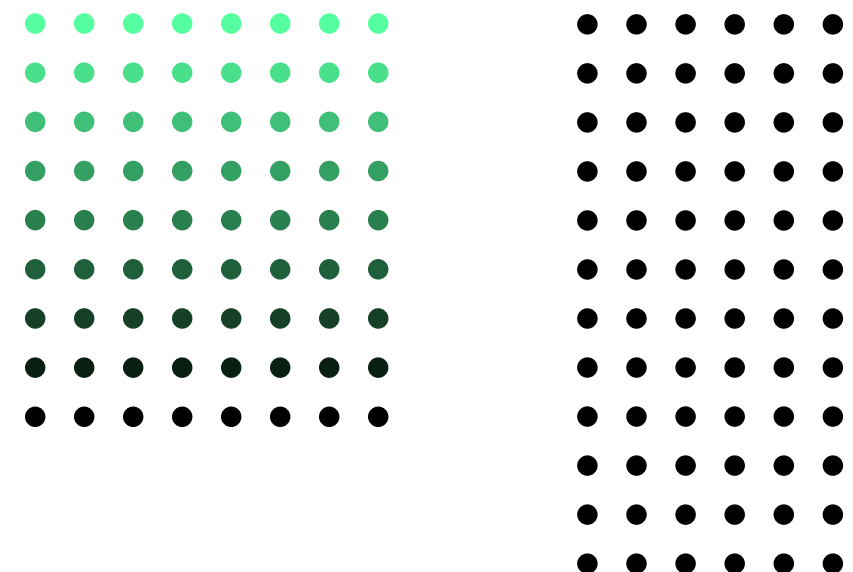
Object Graph: 9 x 8
Processor Graph: 12 x 6



- Maximum Overlap with Alignment (MXOV+AL)

- Alignment at each recursive call

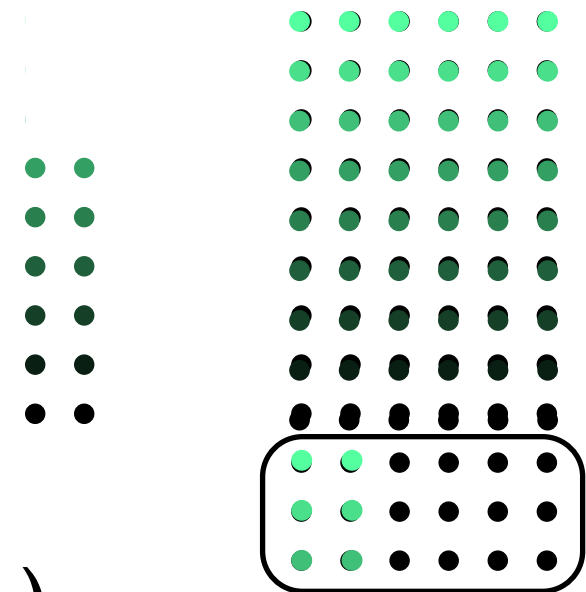
- Expand from Corner (EXCO)



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

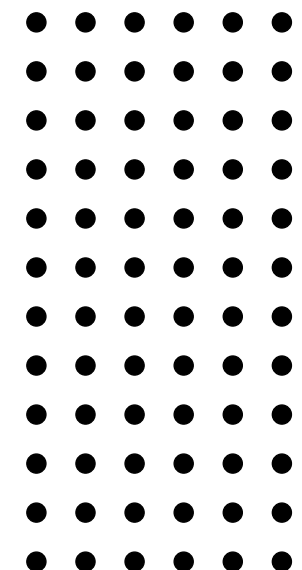
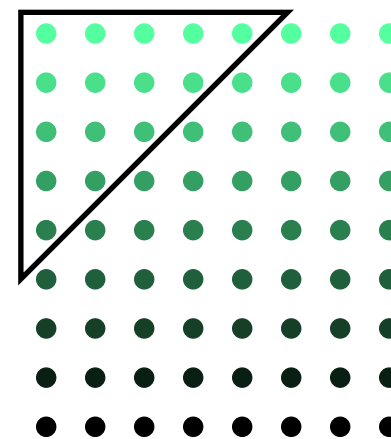
Object Graph: 9 x 8
Processor Graph: 12 x 6



- Maximum Overlap with Alignment (MXOV+AL)

- Alignment at each recursive call

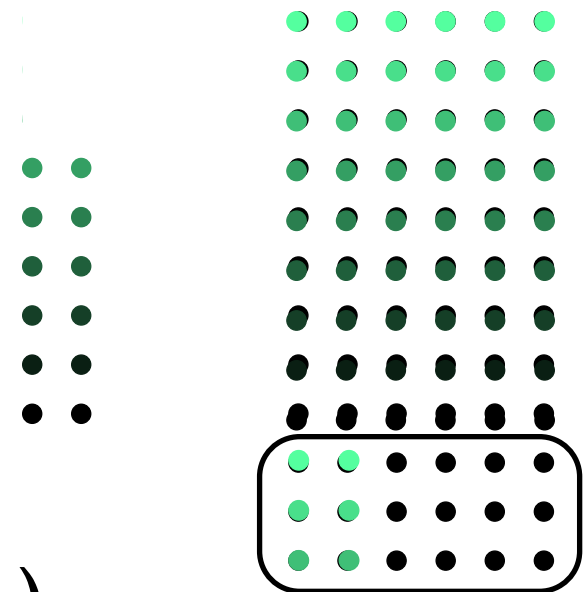
- Expand from Corner (EXCO)



Mapping Regular Graphs (2D)

- Maximum Overlap (MXOVLP)

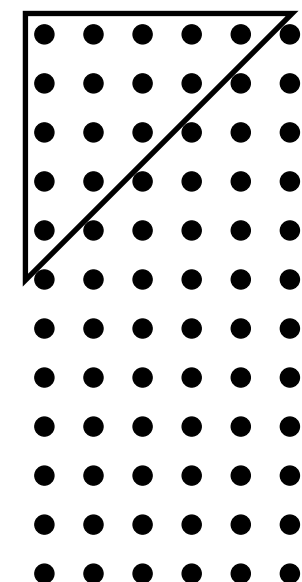
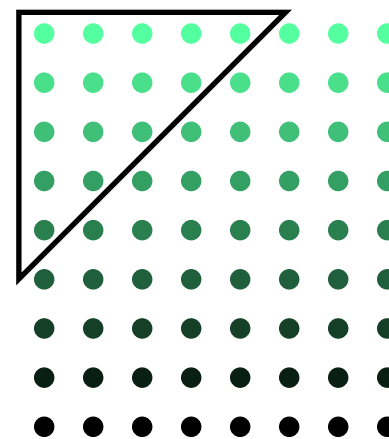
Object Graph: 9 x 8
Processor Graph: 12 x 6



- Maximum Overlap with Alignment (MXOV+AL)

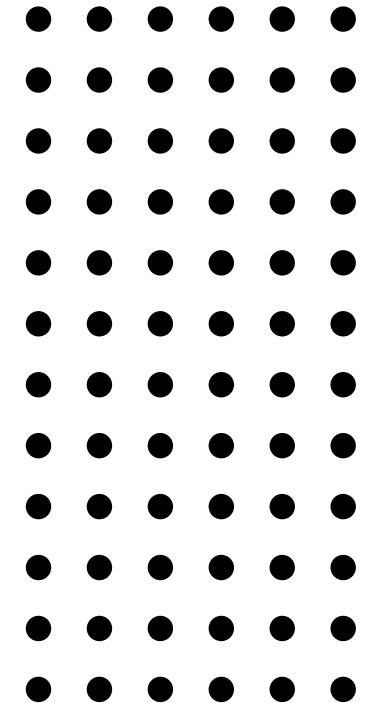
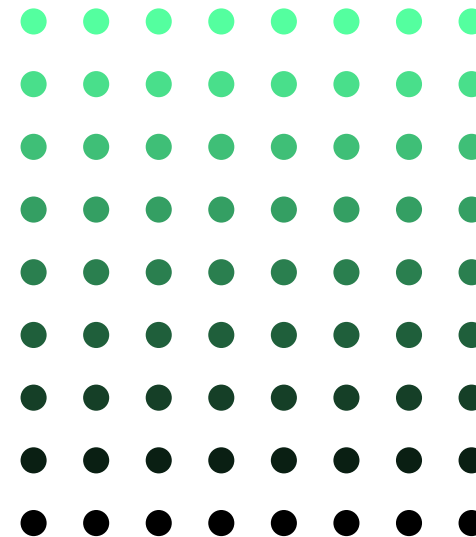
- Alignment at each recursive call

- Expand from Corner (EXCO)



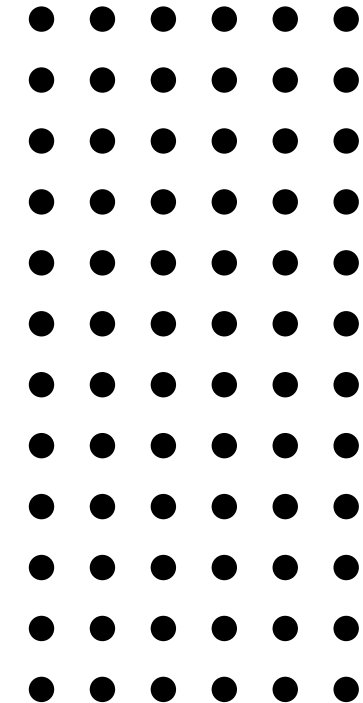
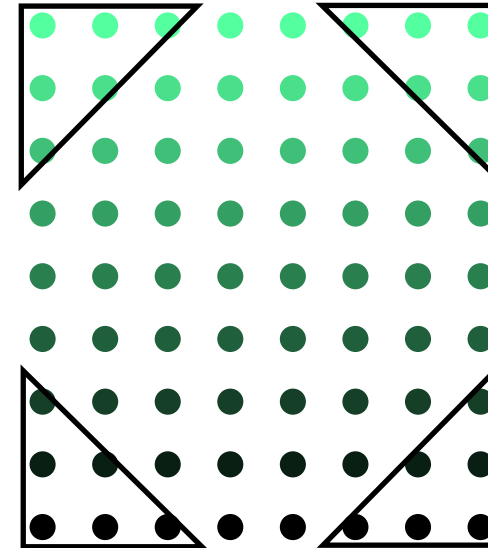
More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners



More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners

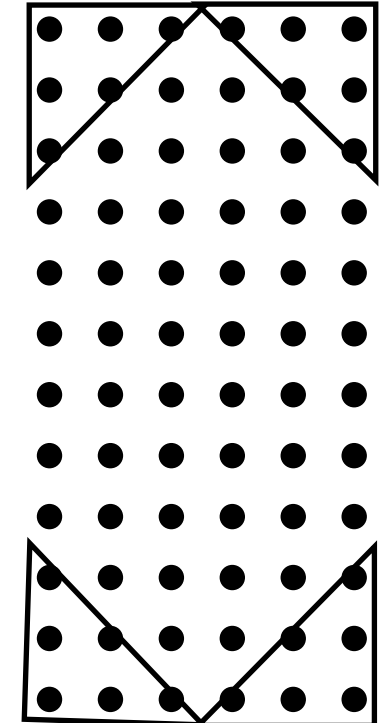
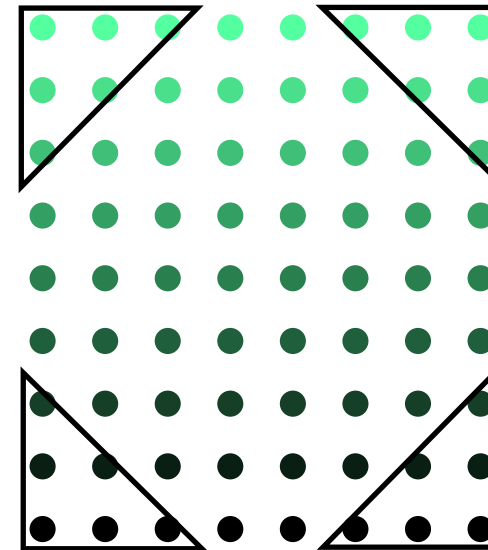


Abhinav Bhatele, Gagan Gupta, Laxmikant V. Kale and I-Hsin Chung, "Automated Mapping of Regular Communication Graphs on Mesh Interconnects", Proceedings of International Conference on High Performance Computing (HiPC), 2010.



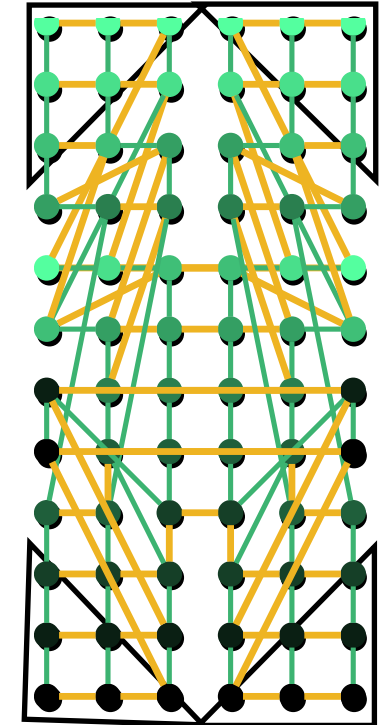
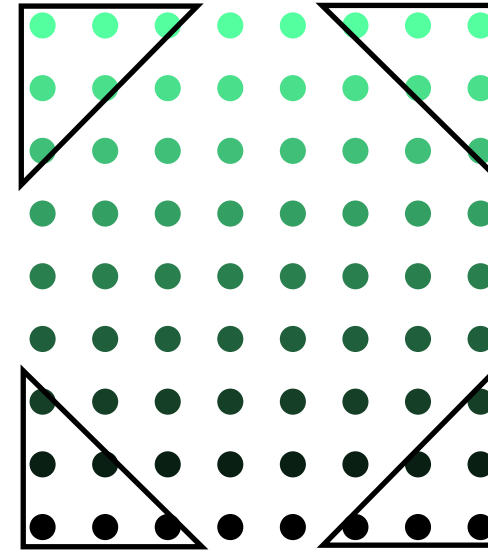
More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners



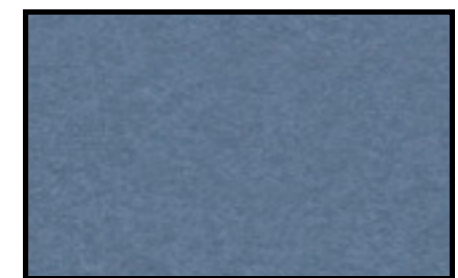
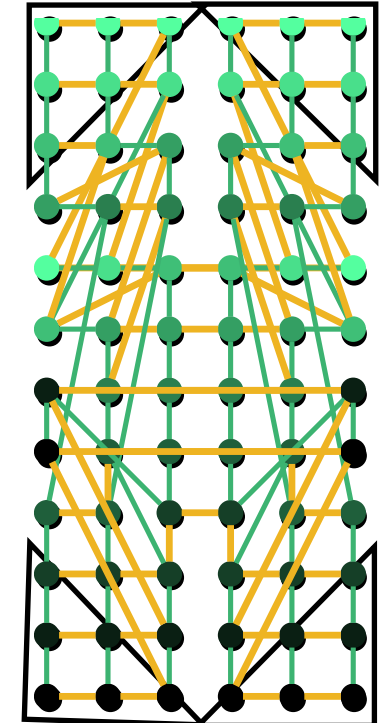
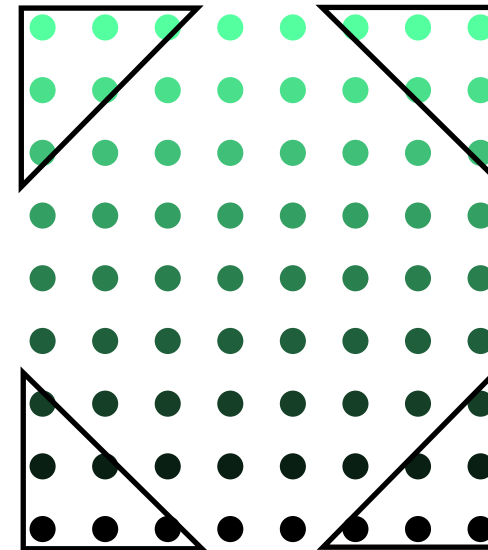
More heuristics ...

- Corners to Center (COCE)
- Start simultaneously from all corners



More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners
- Affine Mapping (AFFN)

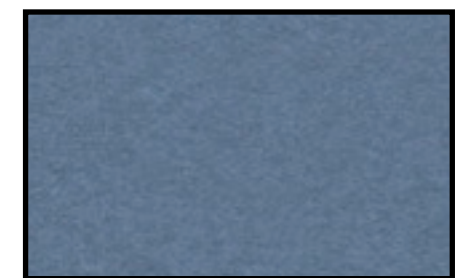
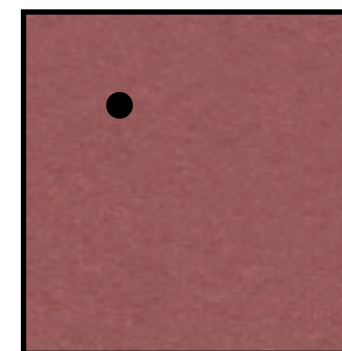
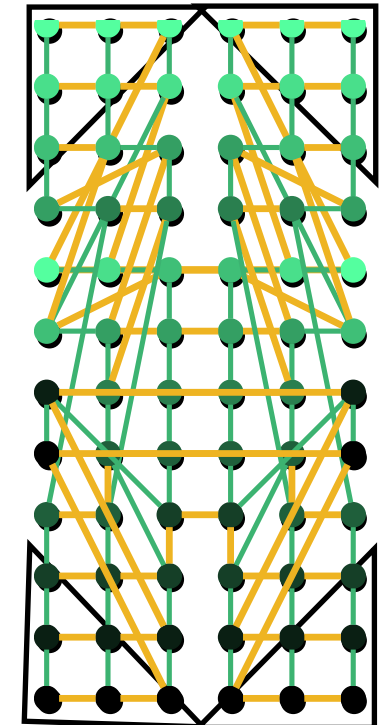
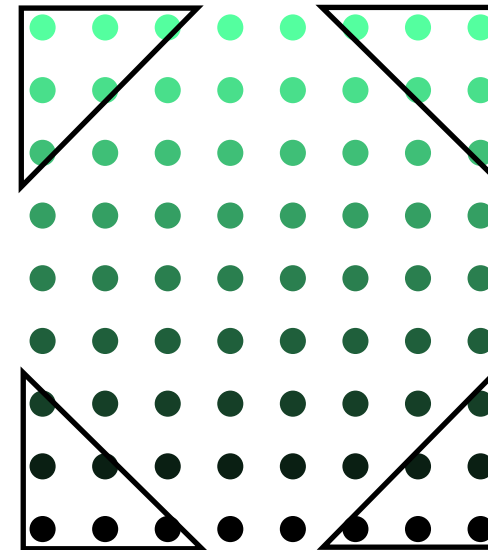


Abhinav Bhatele, Gagan Gupta, Laxmikant V. Kale and I-Hsin Chung, "Automated Mapping of Regular Communication Graphs on Mesh Interconnects", Proceedings of International Conference on High Performance Computing (HiPC), 2010.



More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners
- Affine Mapping (AFFN)

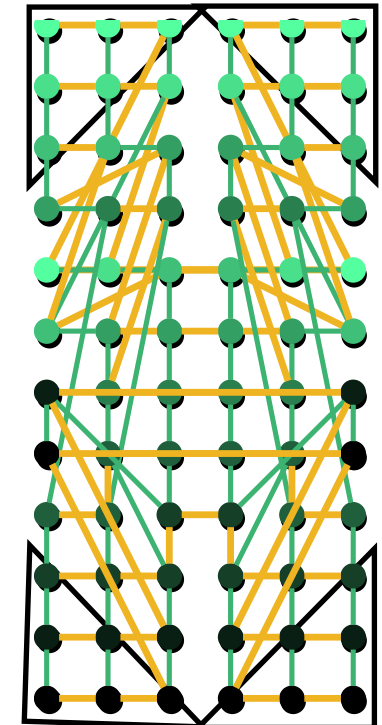
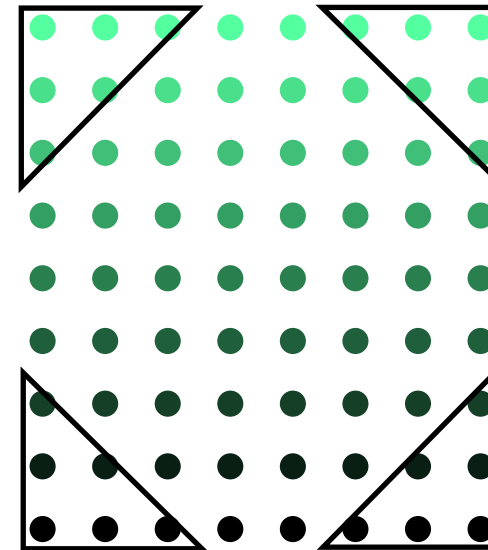


Abhinav Bhatele, Gagan Gupta, Laxmikant V. Kale and I-Hsin Chung, "Automated Mapping of Regular Communication Graphs on Mesh Interconnects", Proceedings of International Conference on High Performance Computing (HiPC), 2010.



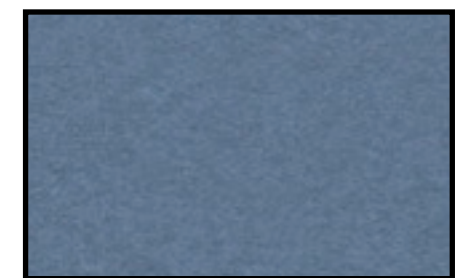
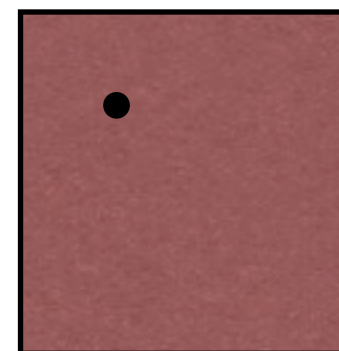
More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners



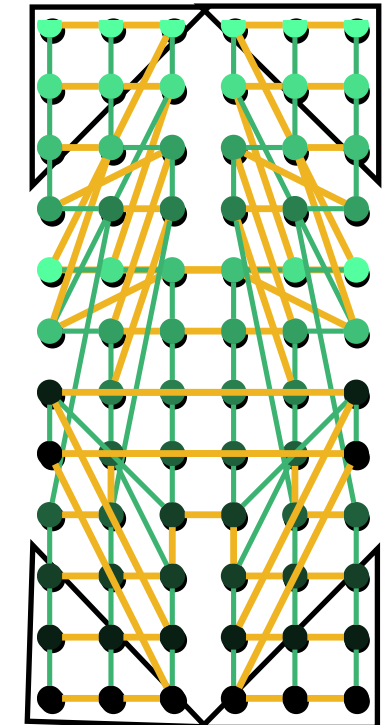
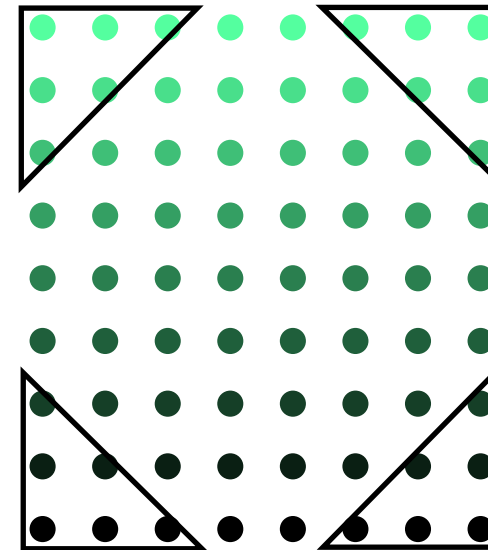
- Affine Mapping (AFFN)

$$(x, y) \rightarrow \left(\lfloor P_x * \frac{x}{O_x} \rfloor, \lfloor P_y * \frac{y}{O_y} \rfloor \right)$$



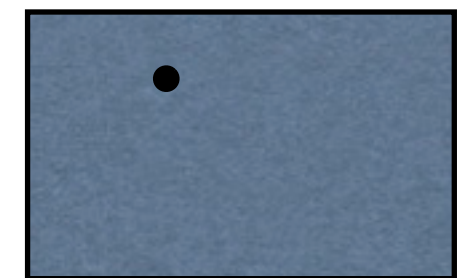
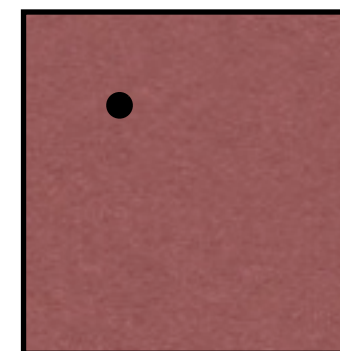
More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners



- Affine Mapping (AFFN)

$$(x, y) \rightarrow \left(\lfloor P_x * \frac{x}{O_x} \rfloor, \lfloor P_y * \frac{y}{O_y} \rfloor \right)$$

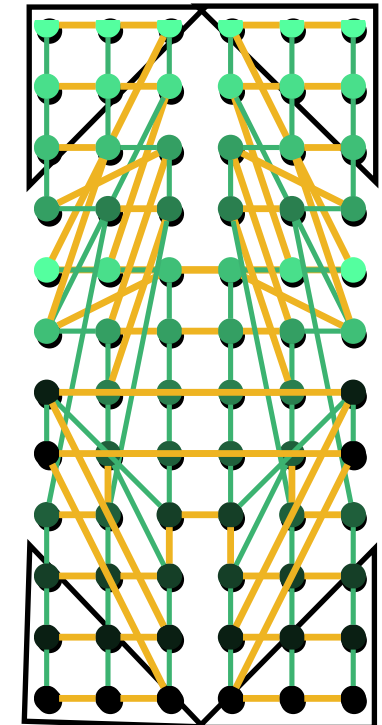
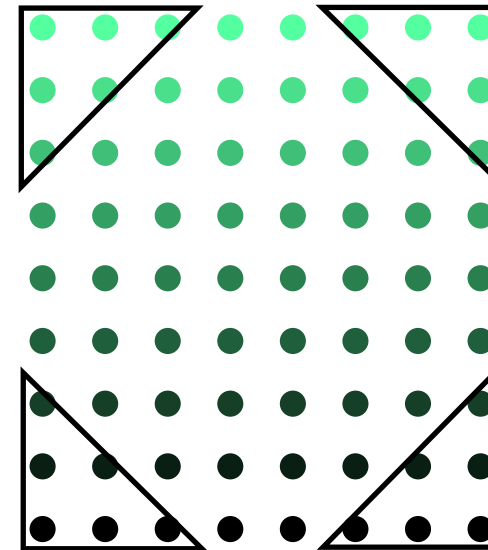


Abhinav Bhatele, Gagan Gupta, Laxmikant V. Kale and I-Hsin Chung, "Automated Mapping of Regular Communication Graphs on Mesh Interconnects", Proceedings of International Conference on High Performance Computing (HiPC), 2010.



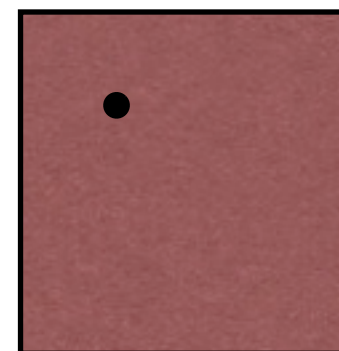
More heuristics ...

- Corners to Center (COCE)
 - Start simultaneously from all corners



- Affine Mapping (AFFN)

$$(x, y) \rightarrow \left(\lfloor P_x * \frac{x}{O_x} \rfloor, \lfloor P_y * \frac{y}{O_y} \rfloor \right)$$

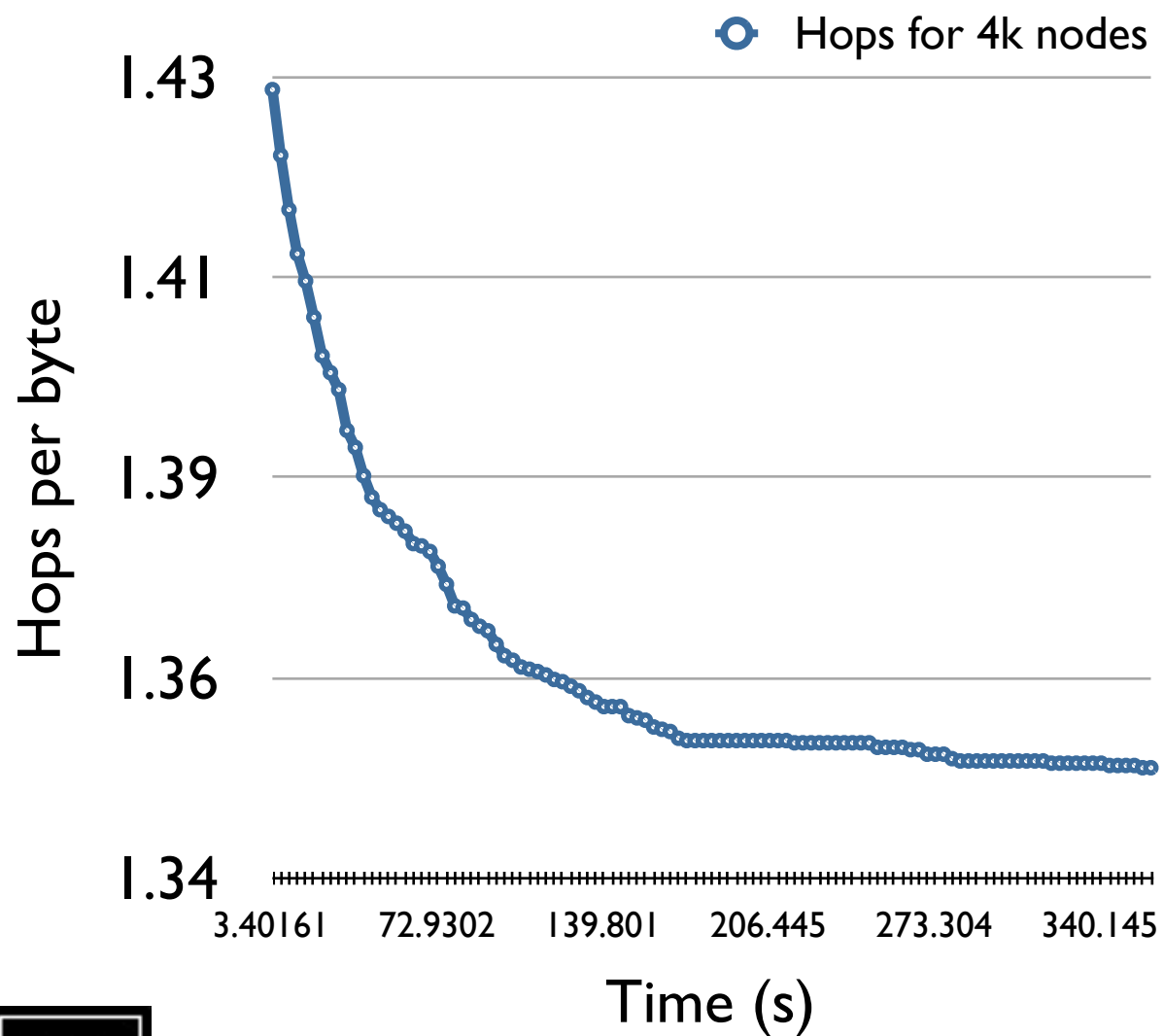


Abhinav Bhatele, Gagan Gupta, Laxmikant V. Kale and I-Hsin Chung, "Automated Mapping of Regular Communication Graphs on Mesh Interconnects", Proceedings of International Conference on High Performance Computing (HiPC), 2010.



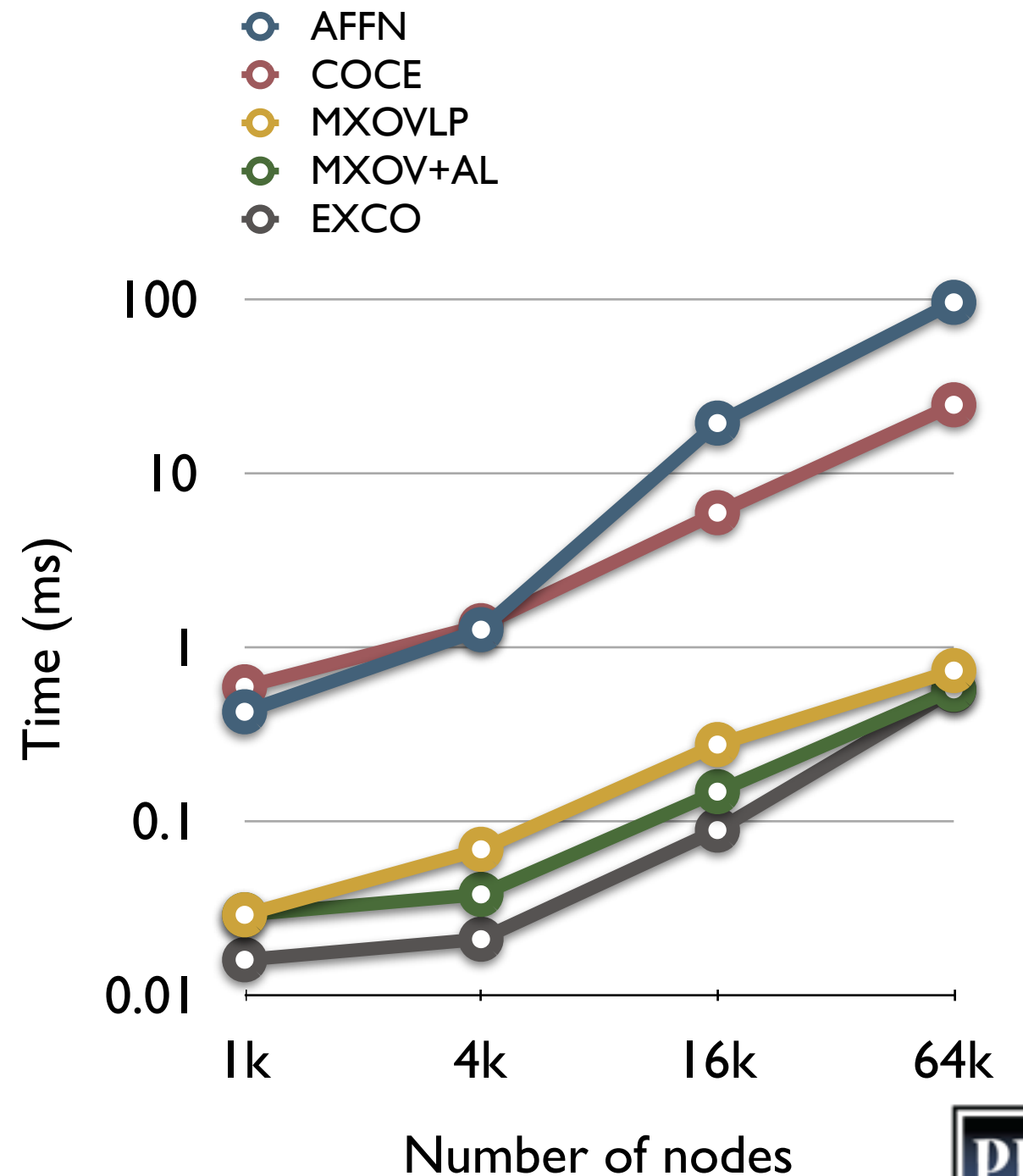
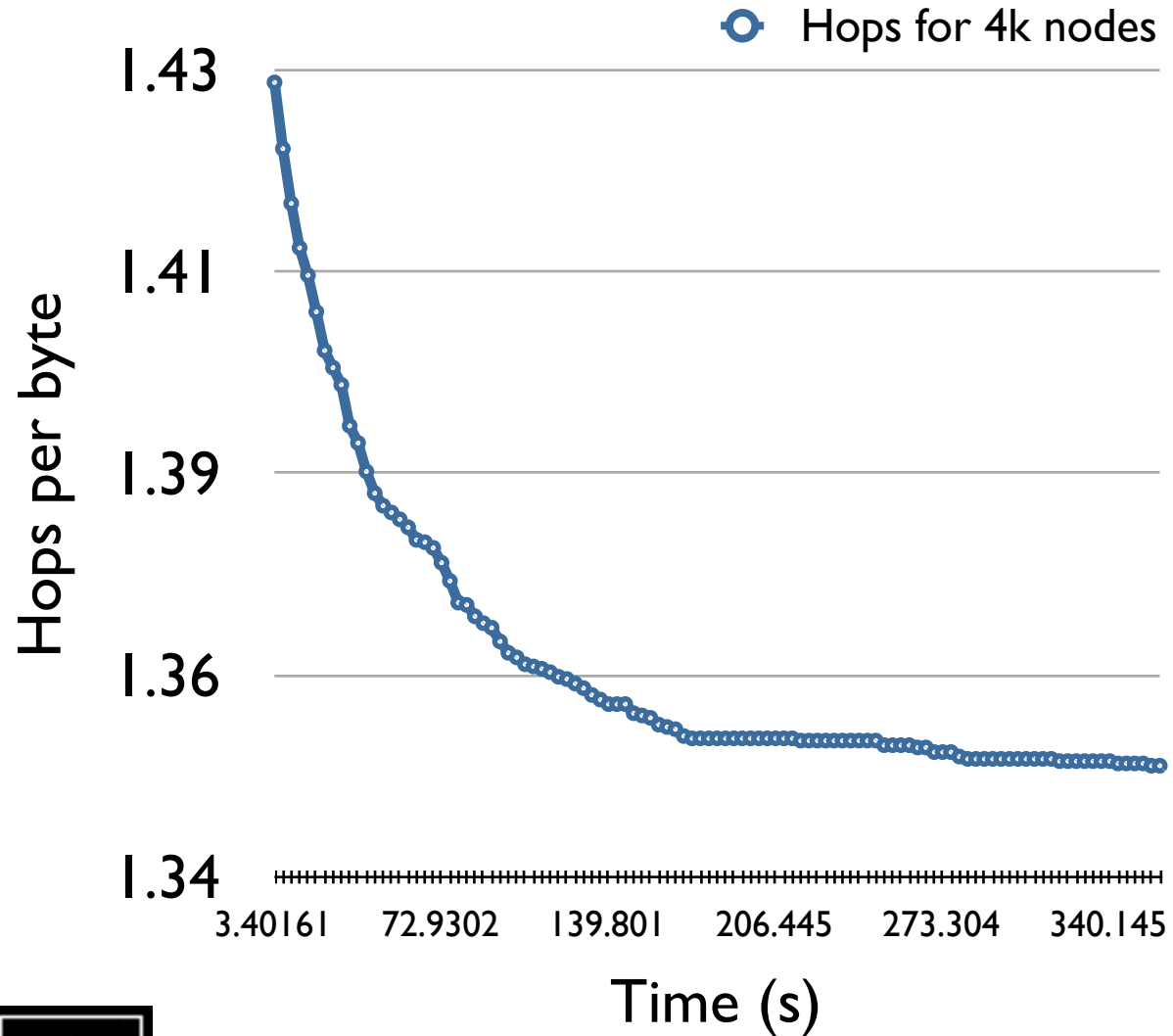
Running Time

- Pairwise Exchanges (PAIRS)
- Bokhari, Lee et al.

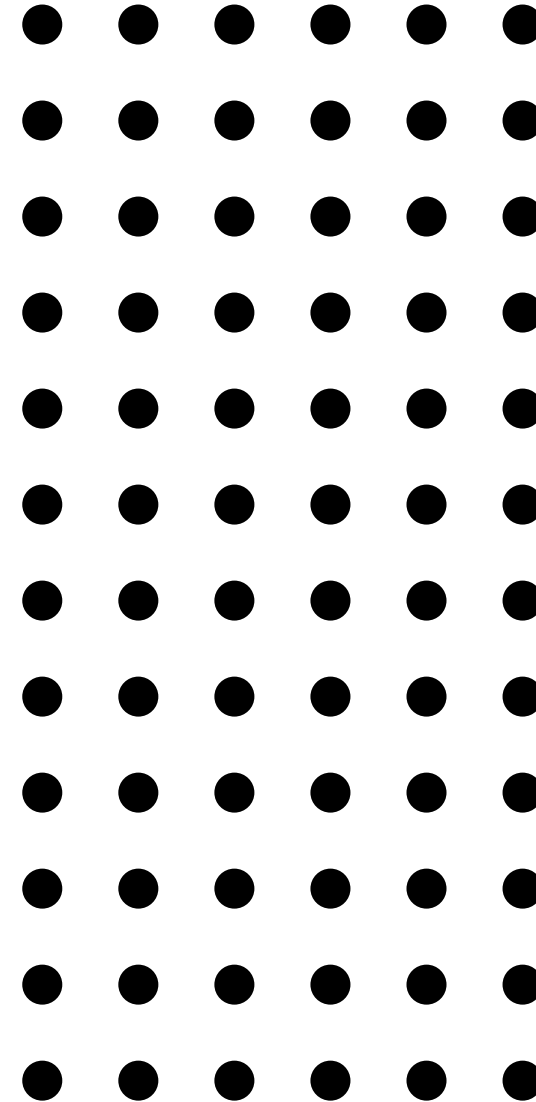
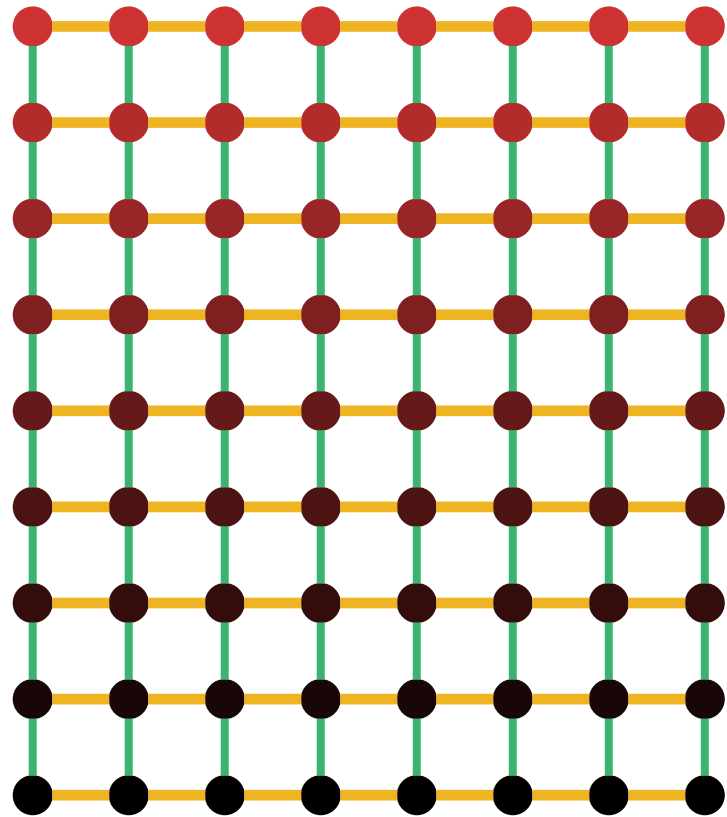


Running Time

- Pairwise Exchanges (PAIRS)
- Bokhari, Lee et al.



Example Mapping

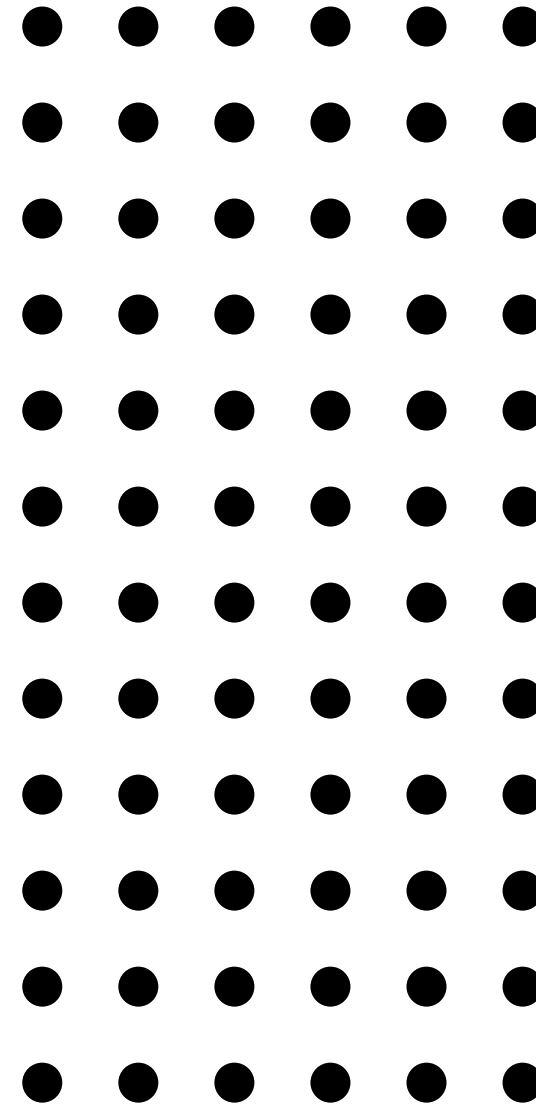
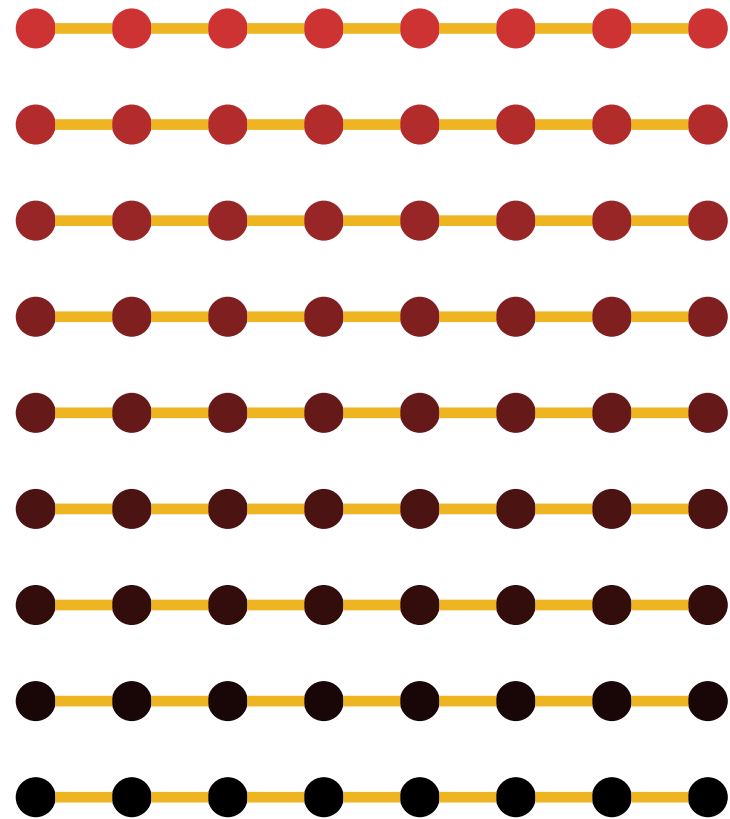


Object Graph: 9×8
Processor Graph: 12×6

Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907-913, 1982



Example Mapping

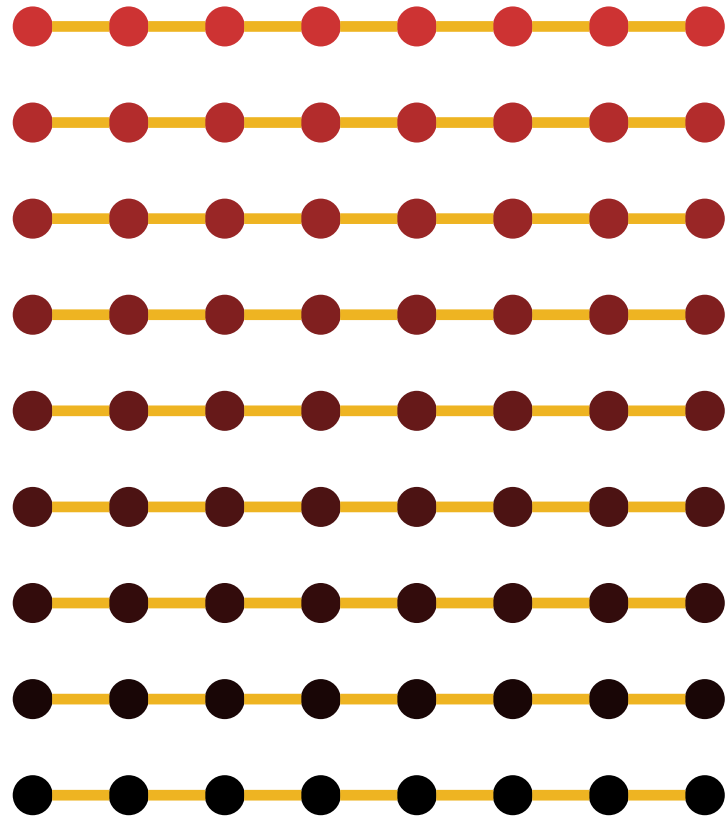


Object Graph: 9×8
Processor Graph: 12×6

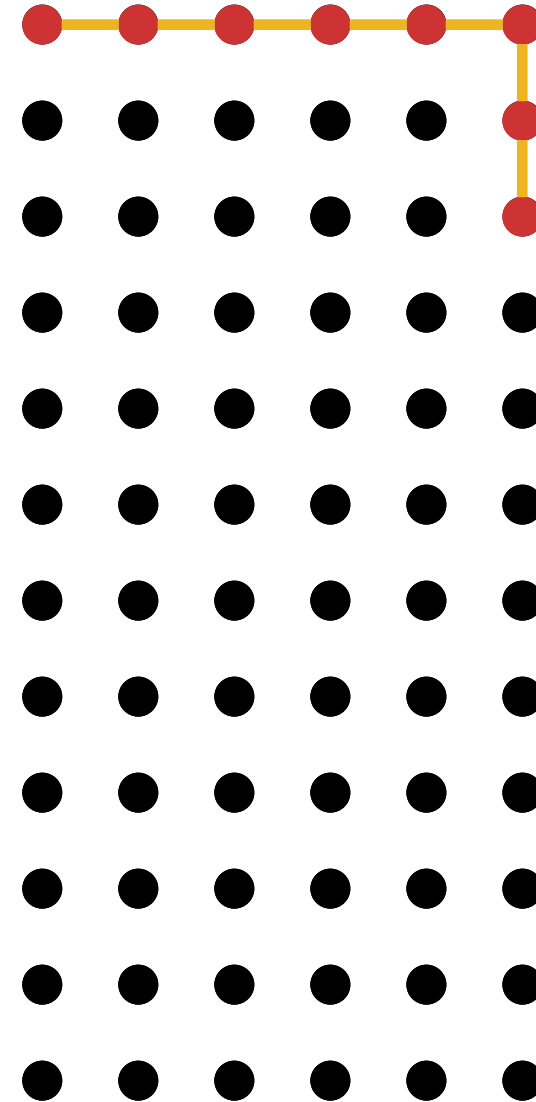
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907–913, 1982



Example Mapping



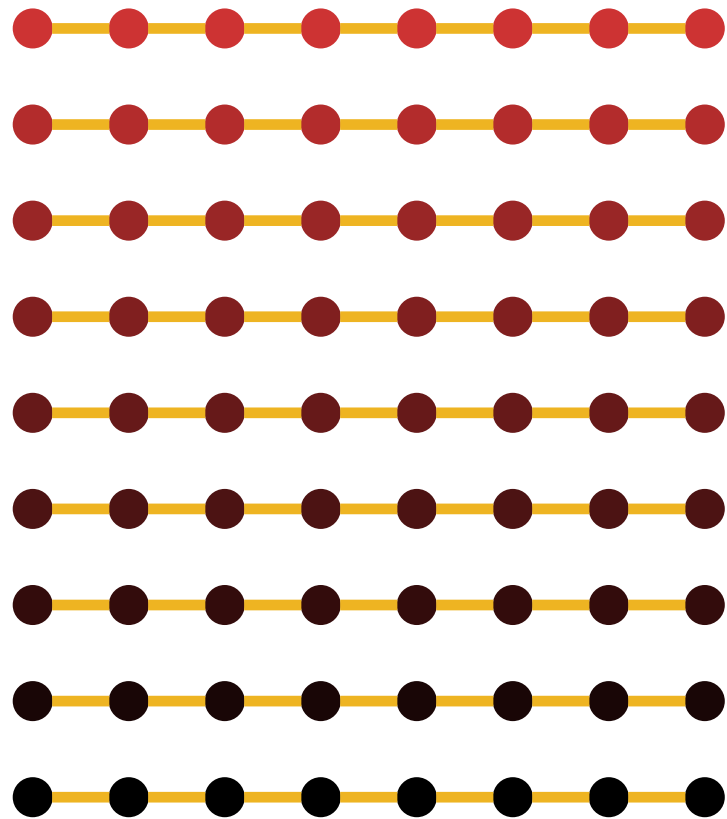
Object Graph: 9 x 8
Processor Graph: 12 x 6



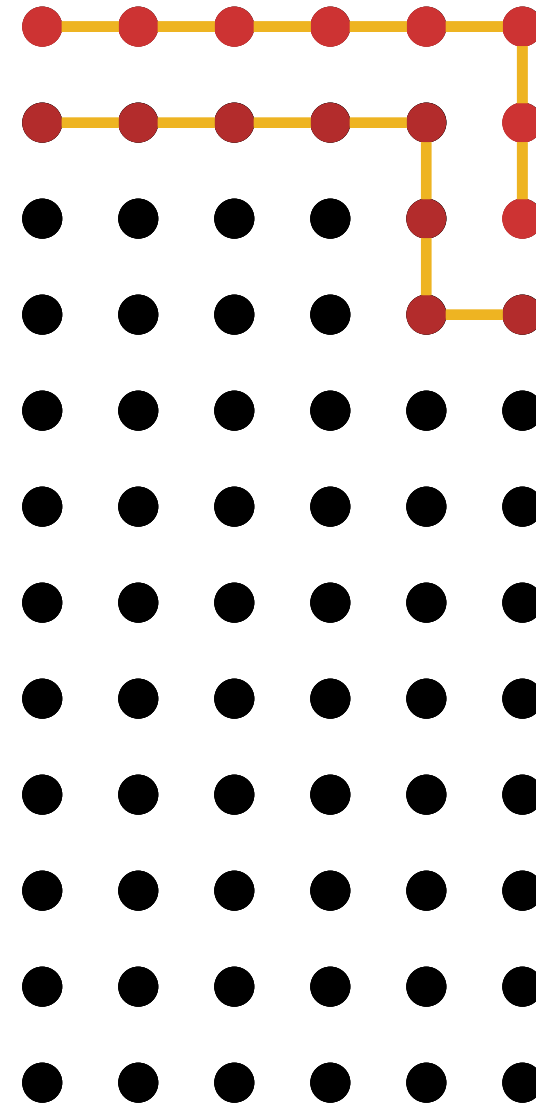
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907-913, 1982



Example Mapping



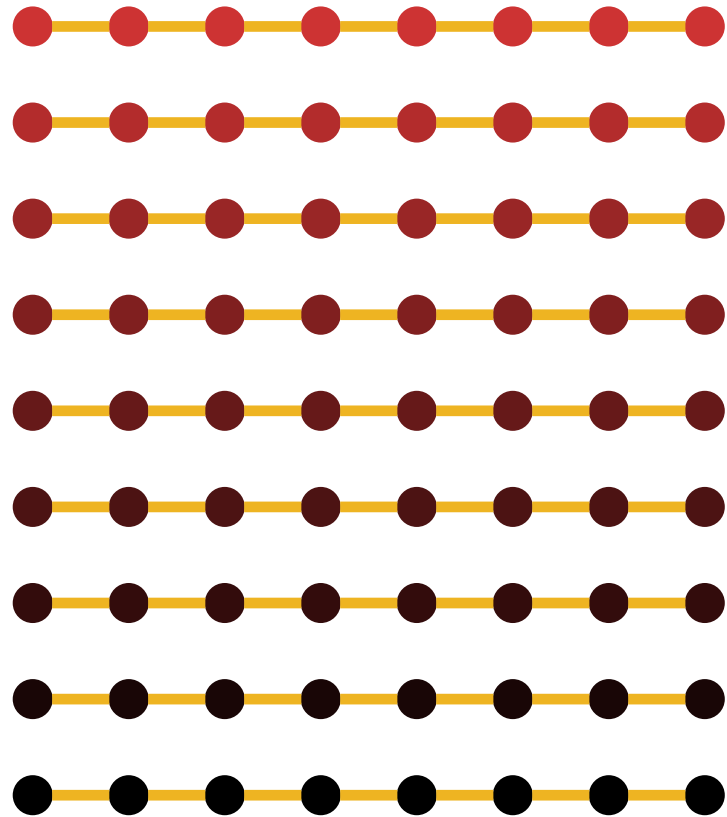
Object Graph: 9 x 8
Processor Graph: 12 x 6



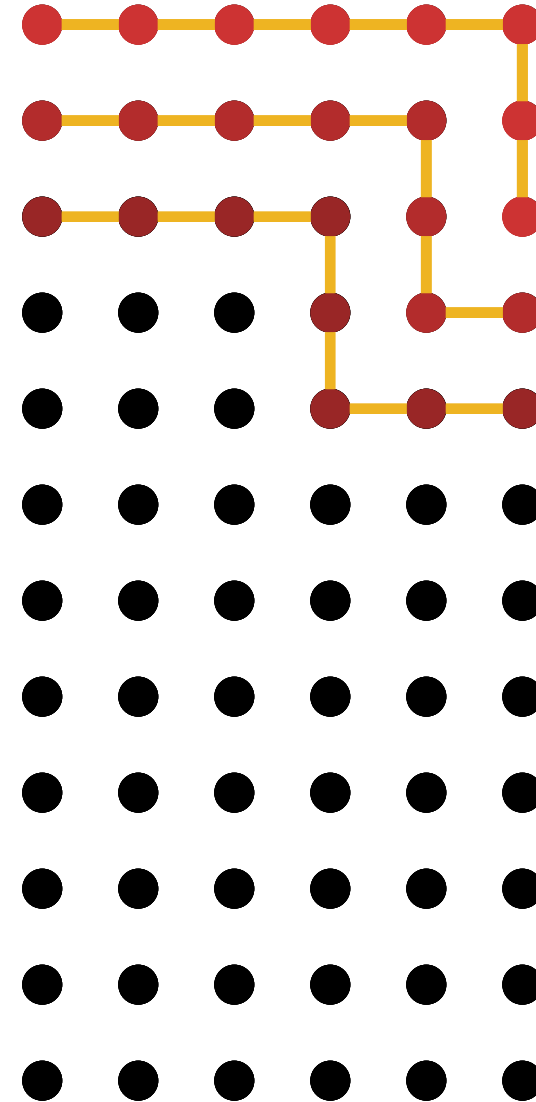
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907-913, 1982



Example Mapping



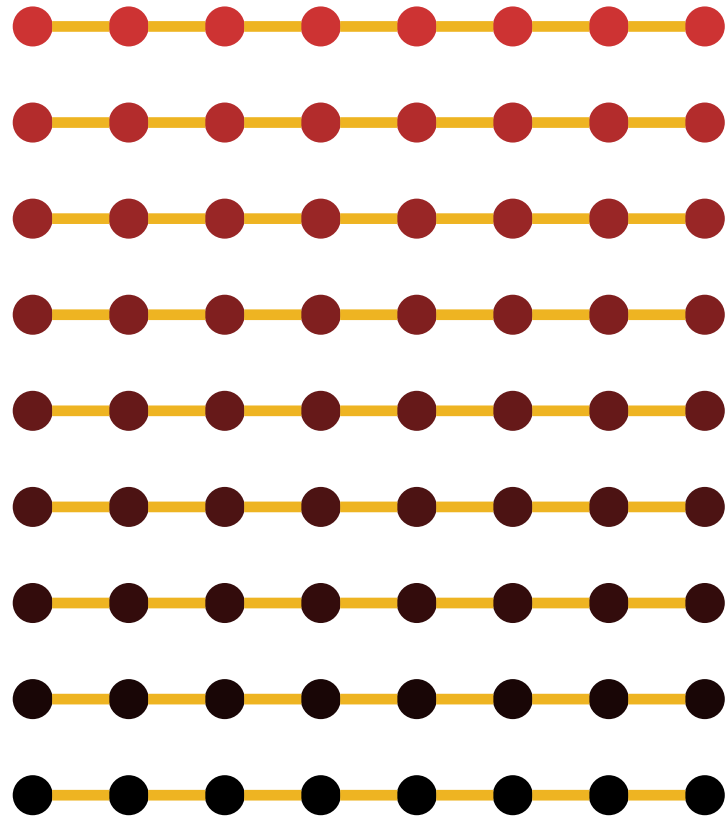
Object Graph: 9 x 8
Processor Graph: 12 x 6



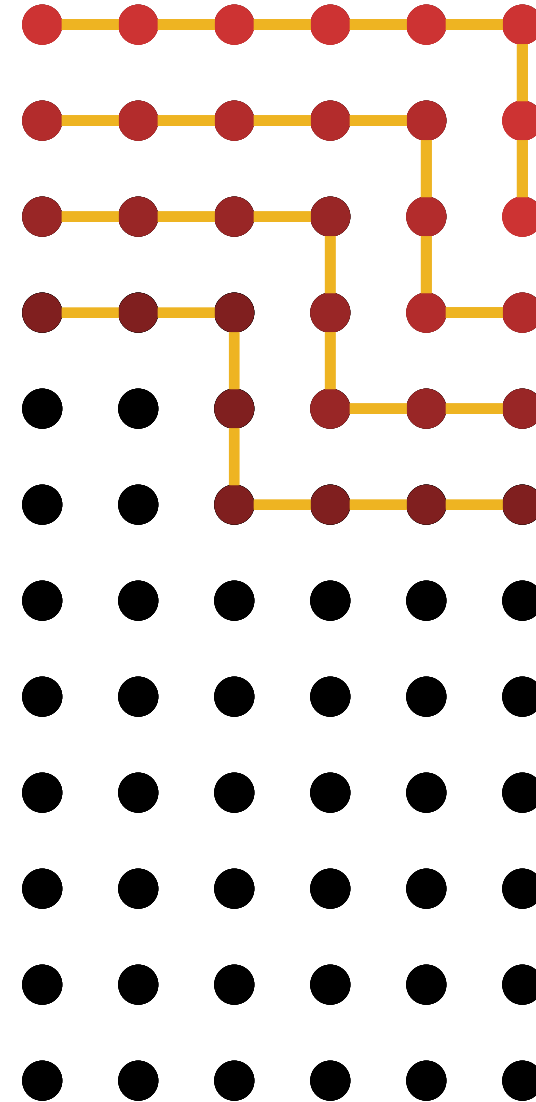
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907-913, 1982



Example Mapping



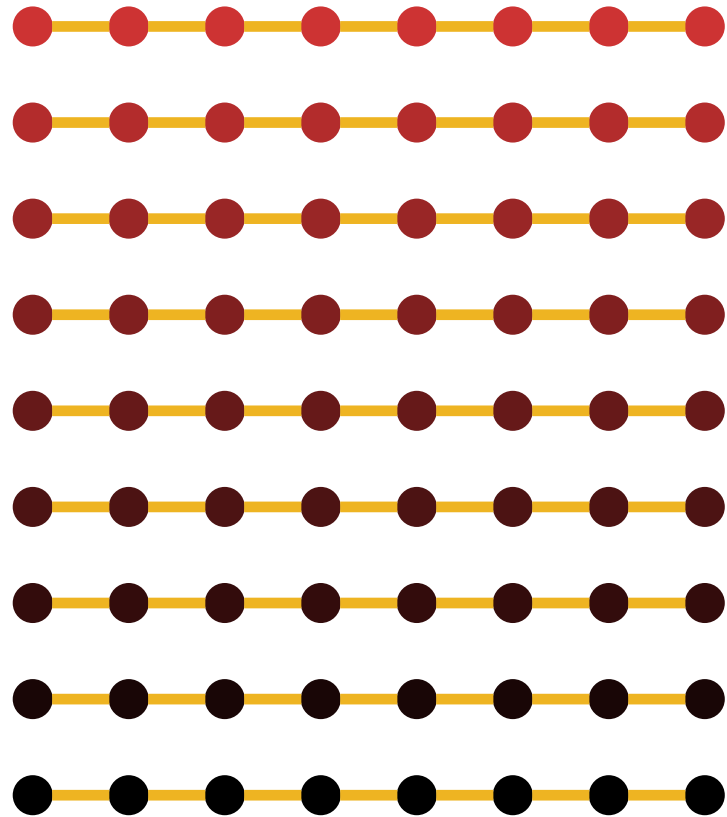
Object Graph: 9×8
Processor Graph: 12×6



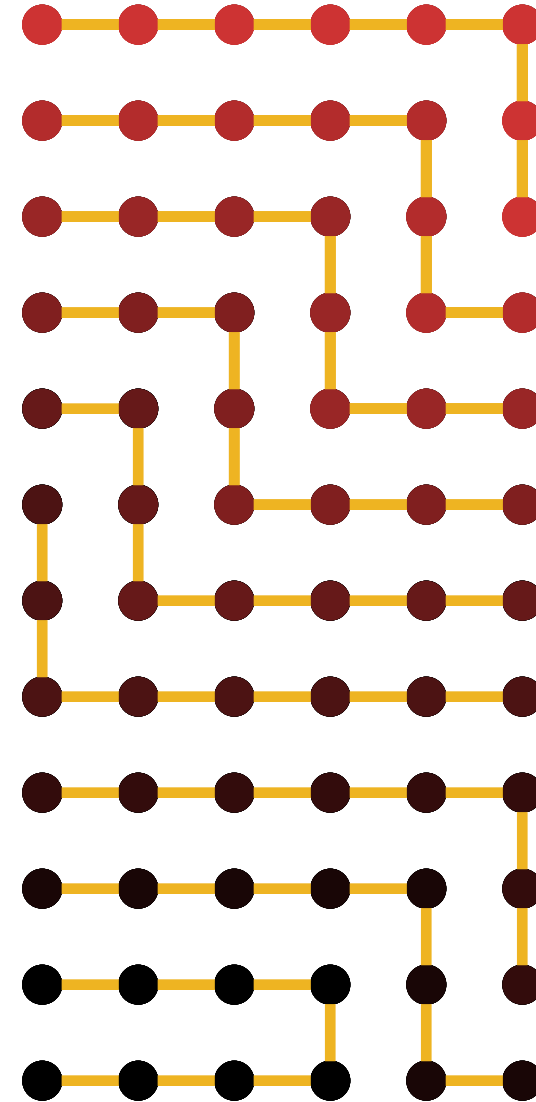
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907–913, 1982



Example Mapping



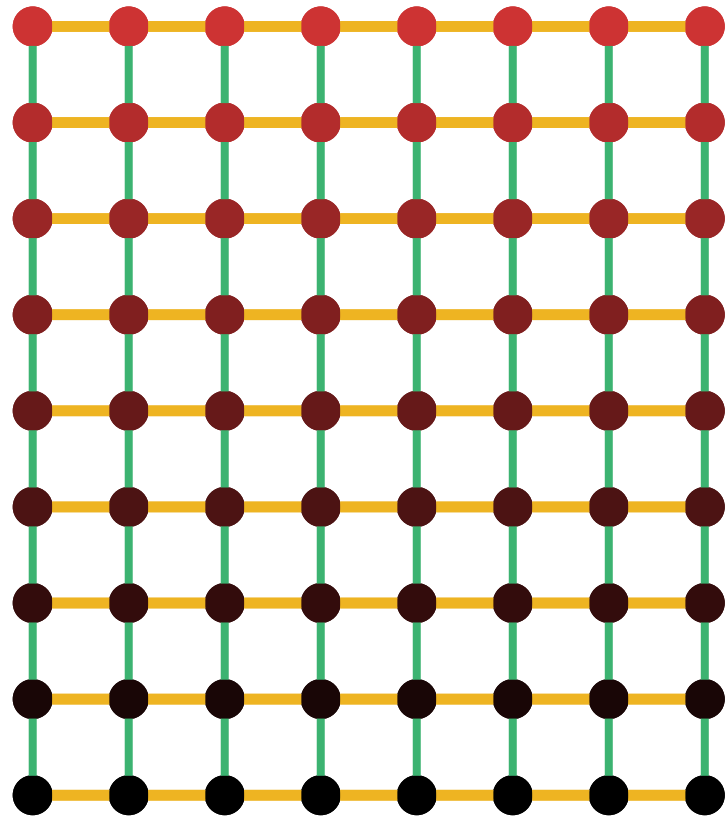
Object Graph: 9 x 8
Processor Graph: 12 x 6



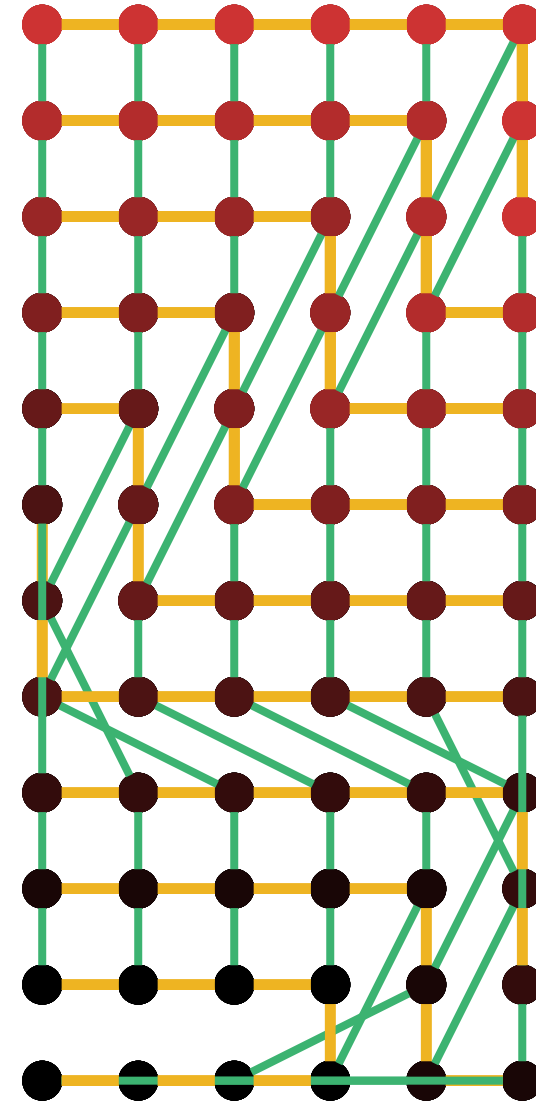
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907-913, 1982



Example Mapping



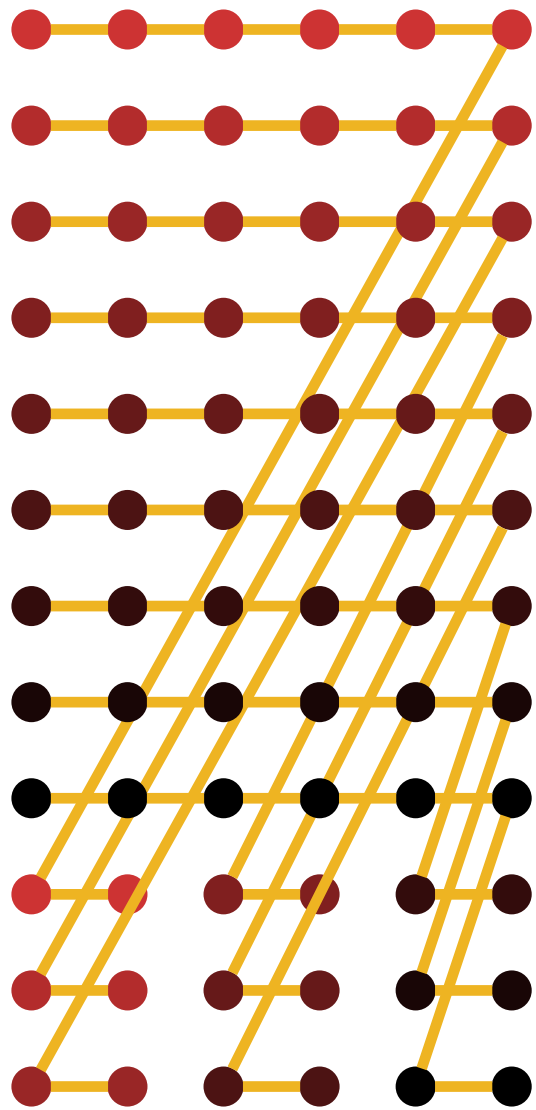
Object Graph: 9 x 8
Processor Graph: 12 x 6



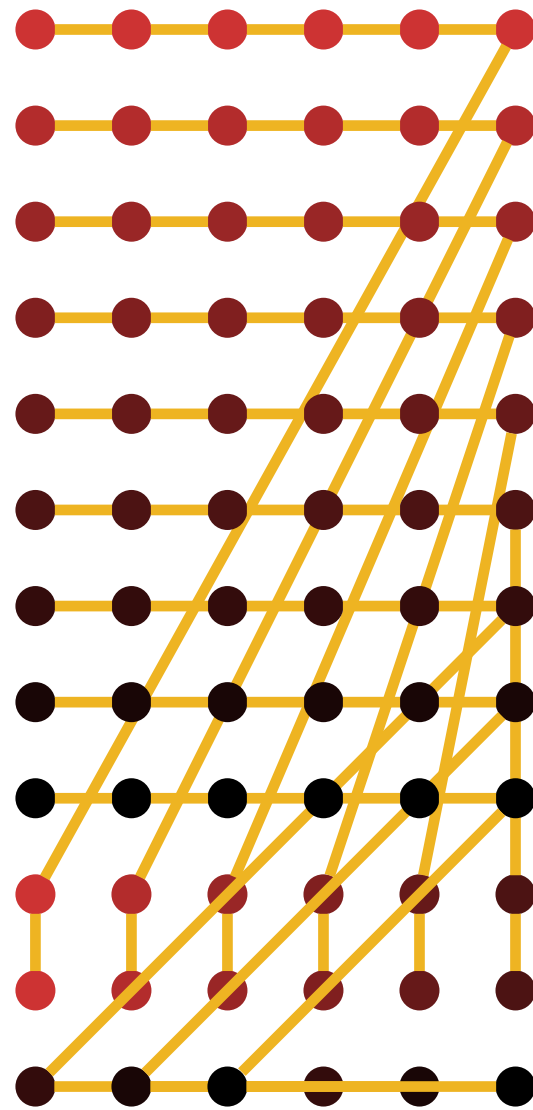
Aleliunas, R. and Rosenberg, A. L. On Embedding Rectangular
Grids in Square Grids. IEEE Trans. Comput., 31(9):907-913, 1982



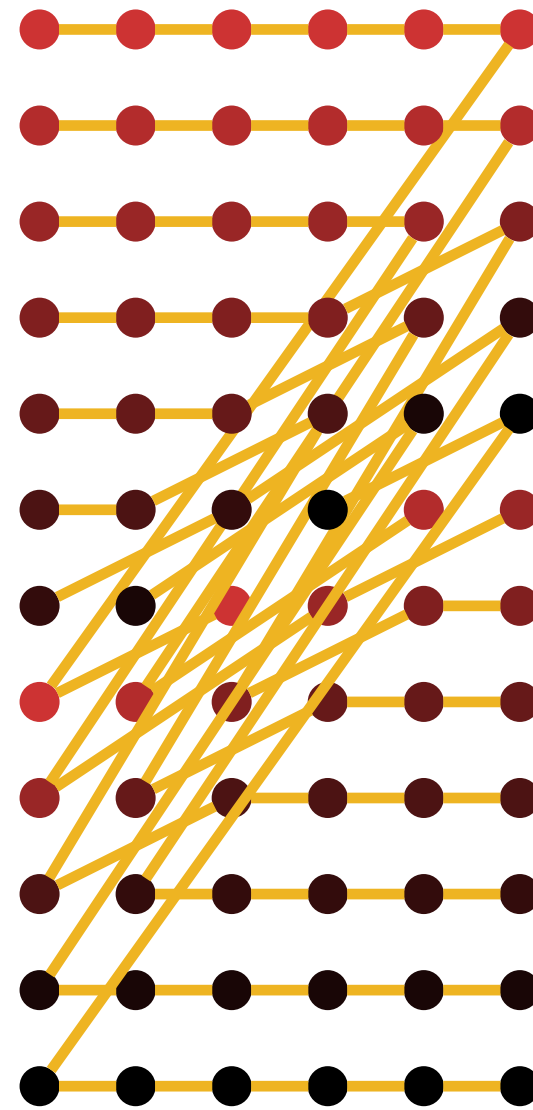
Mapping of 9x8 graph to 12x6 mesh



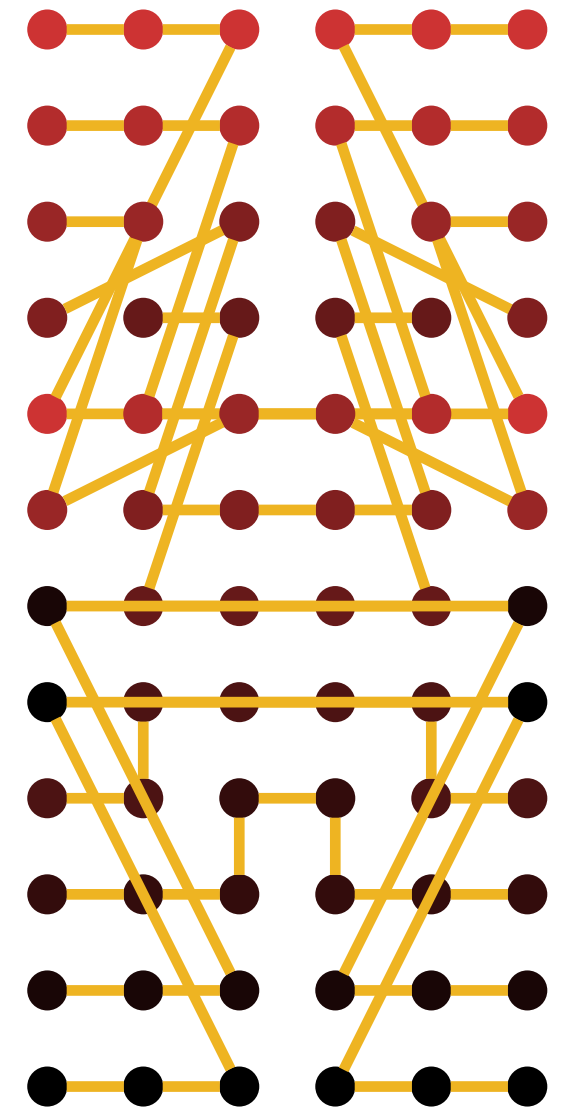
MXOVLP: 1.66



MXOV+AL: 1.65



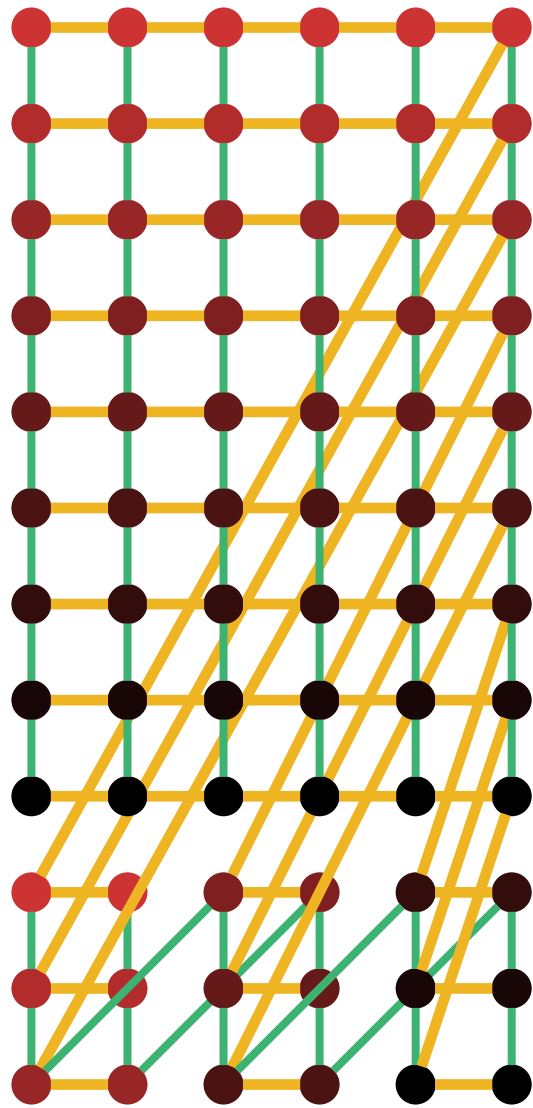
EXCO: 2.31



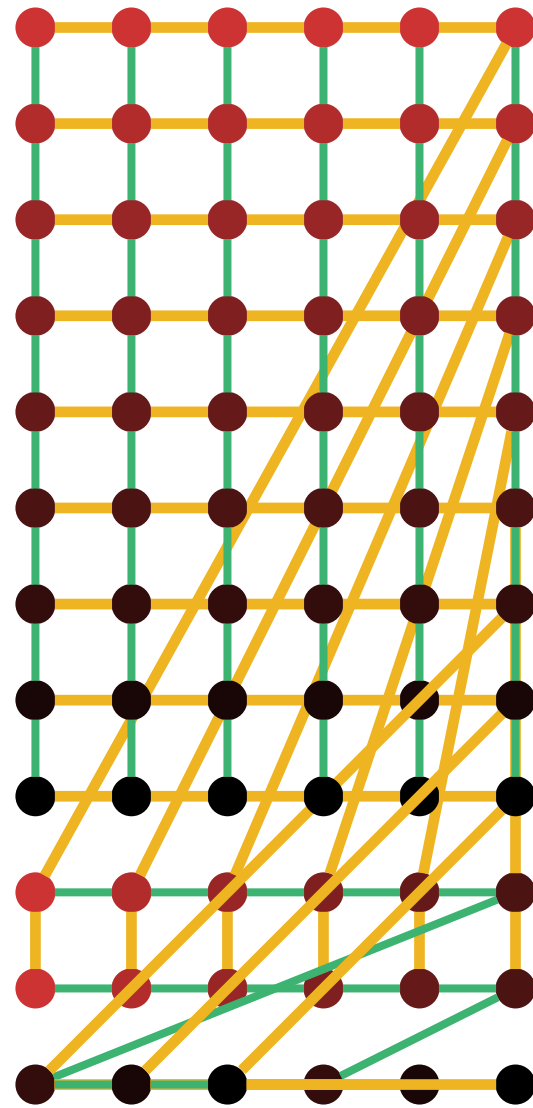
COCE: 1.91



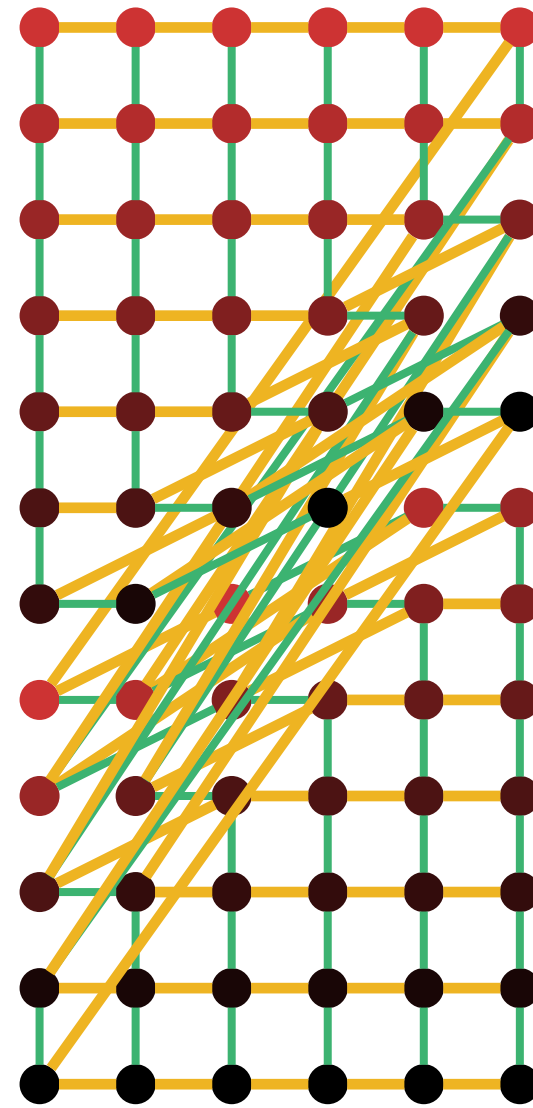
Mapping of 9x8 graph to 12x6 mesh



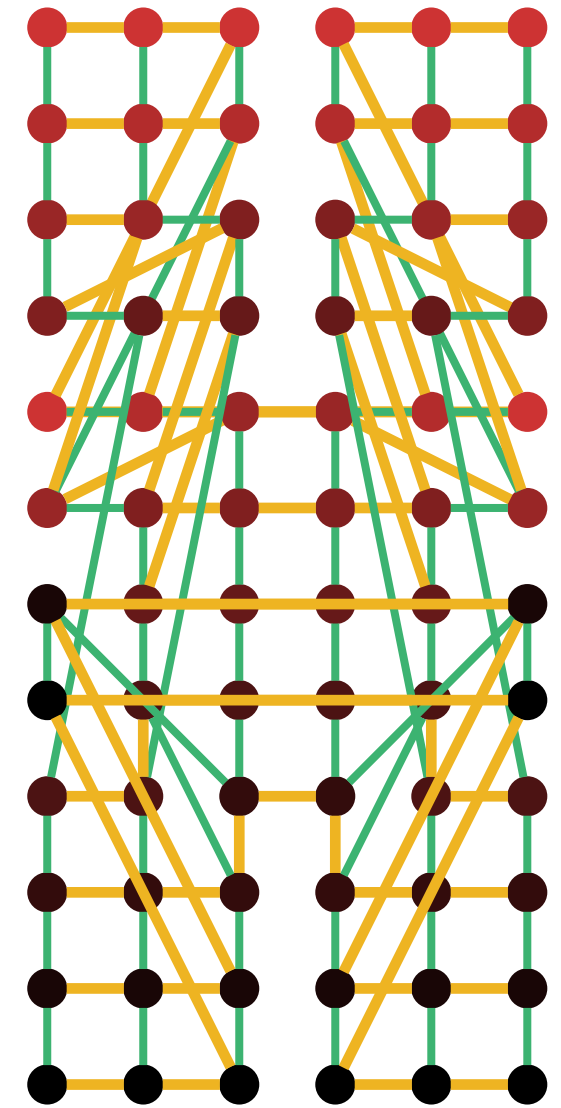
MXOVLP: 1.66



MXOV+AL: 1.65



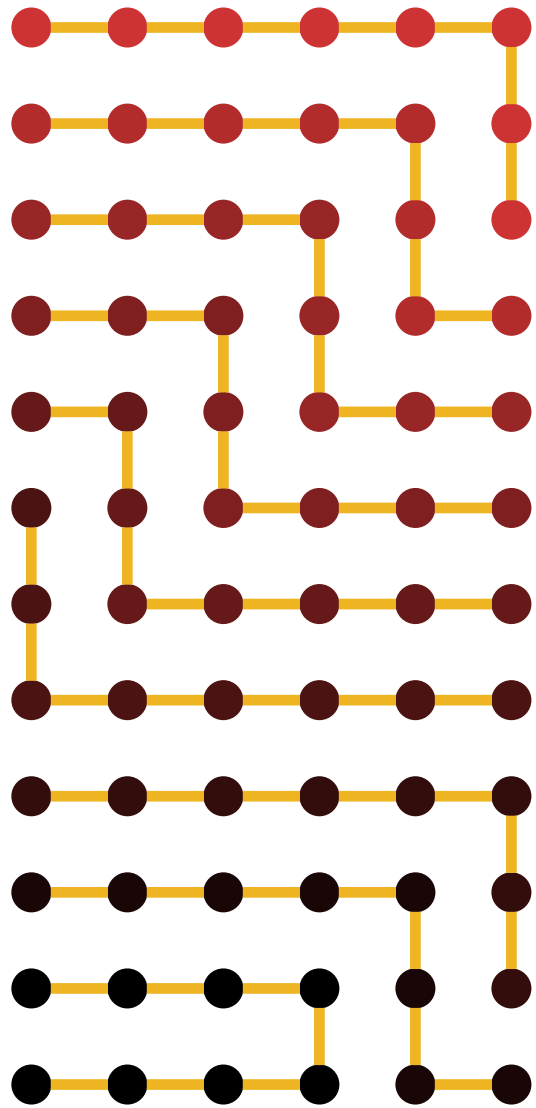
EXCO: 2.31



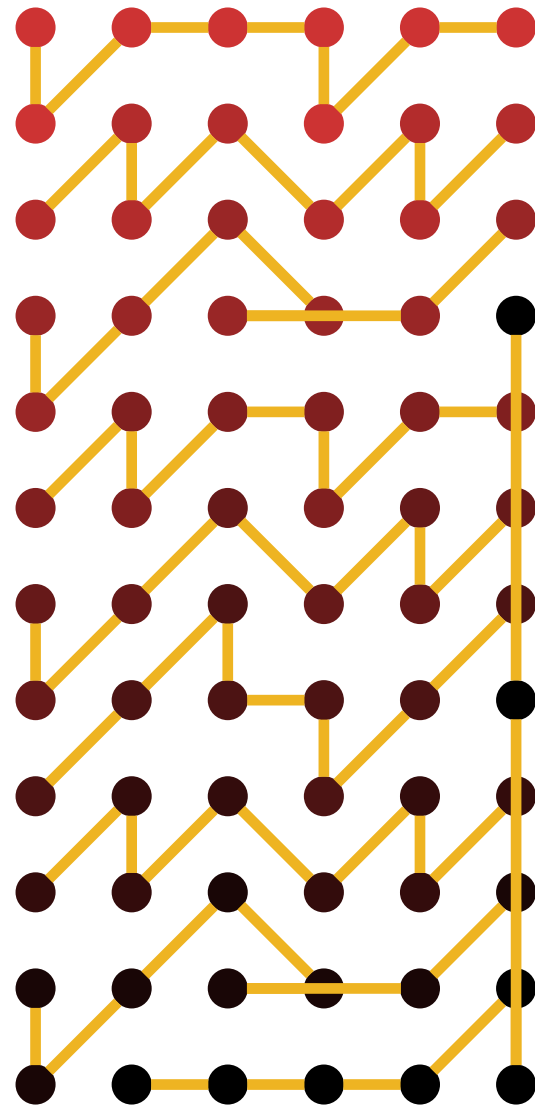
COCE: 1.91



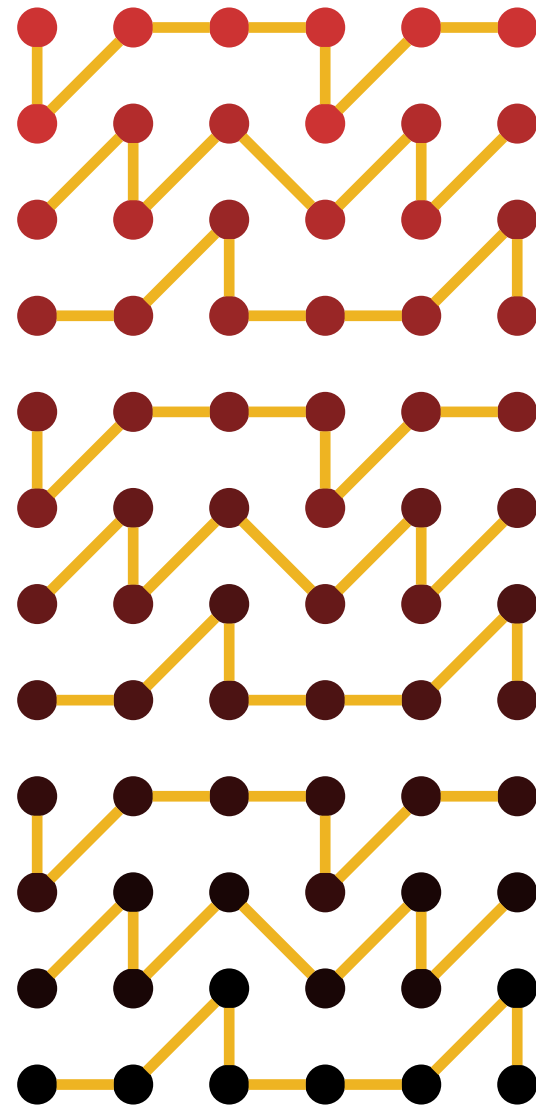
Mapping of 9x8 graph to 12x6 mesh



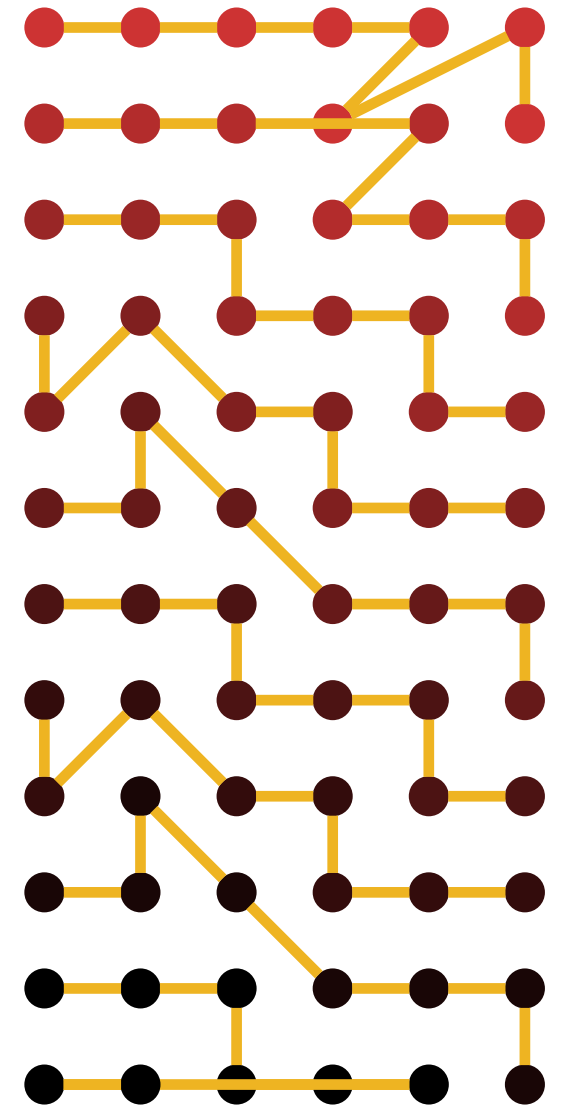
STEP: 1.39



AFFN1: 1.77



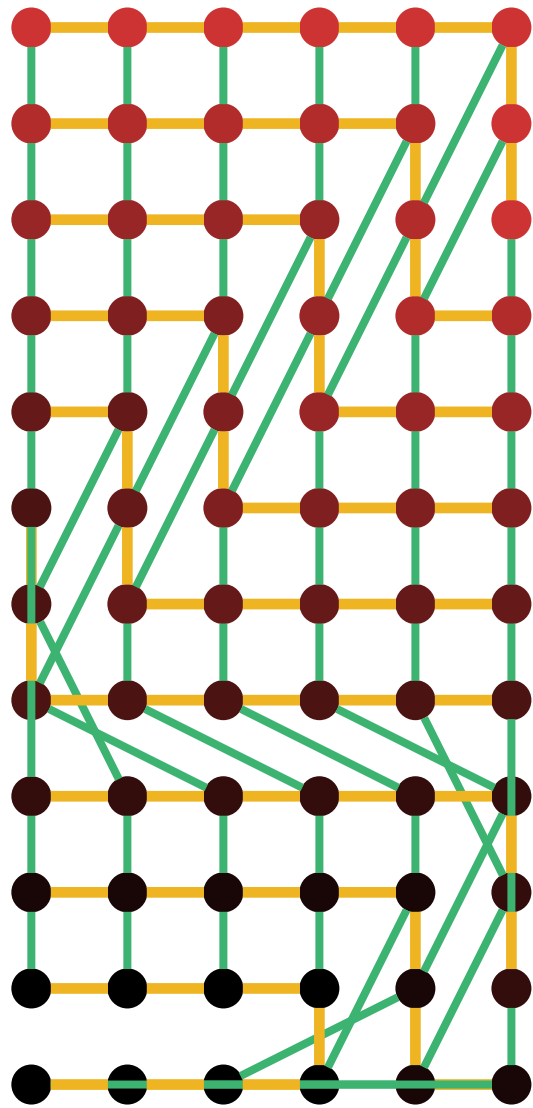
AFFN2: 1.53



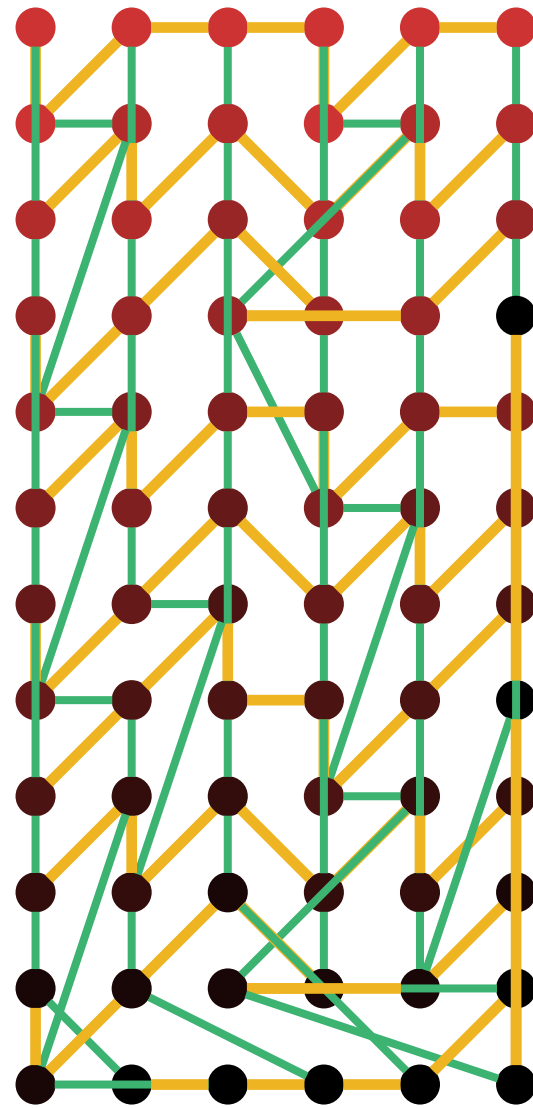
AFFN3: 1.91



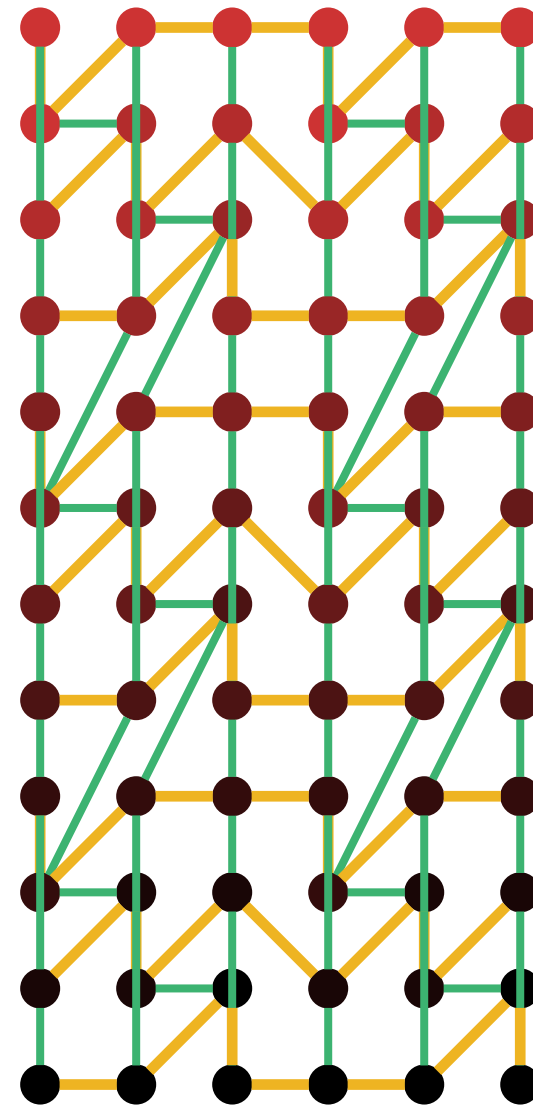
Mapping of 9x8 graph to 12x6 mesh



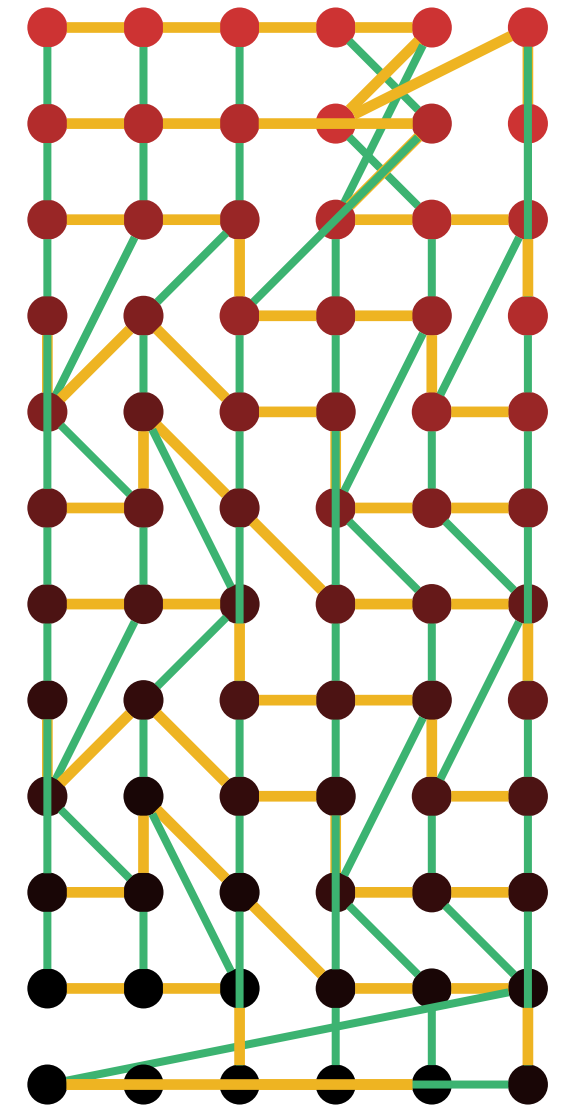
STEP: 1.39



AFFN1: 1.77



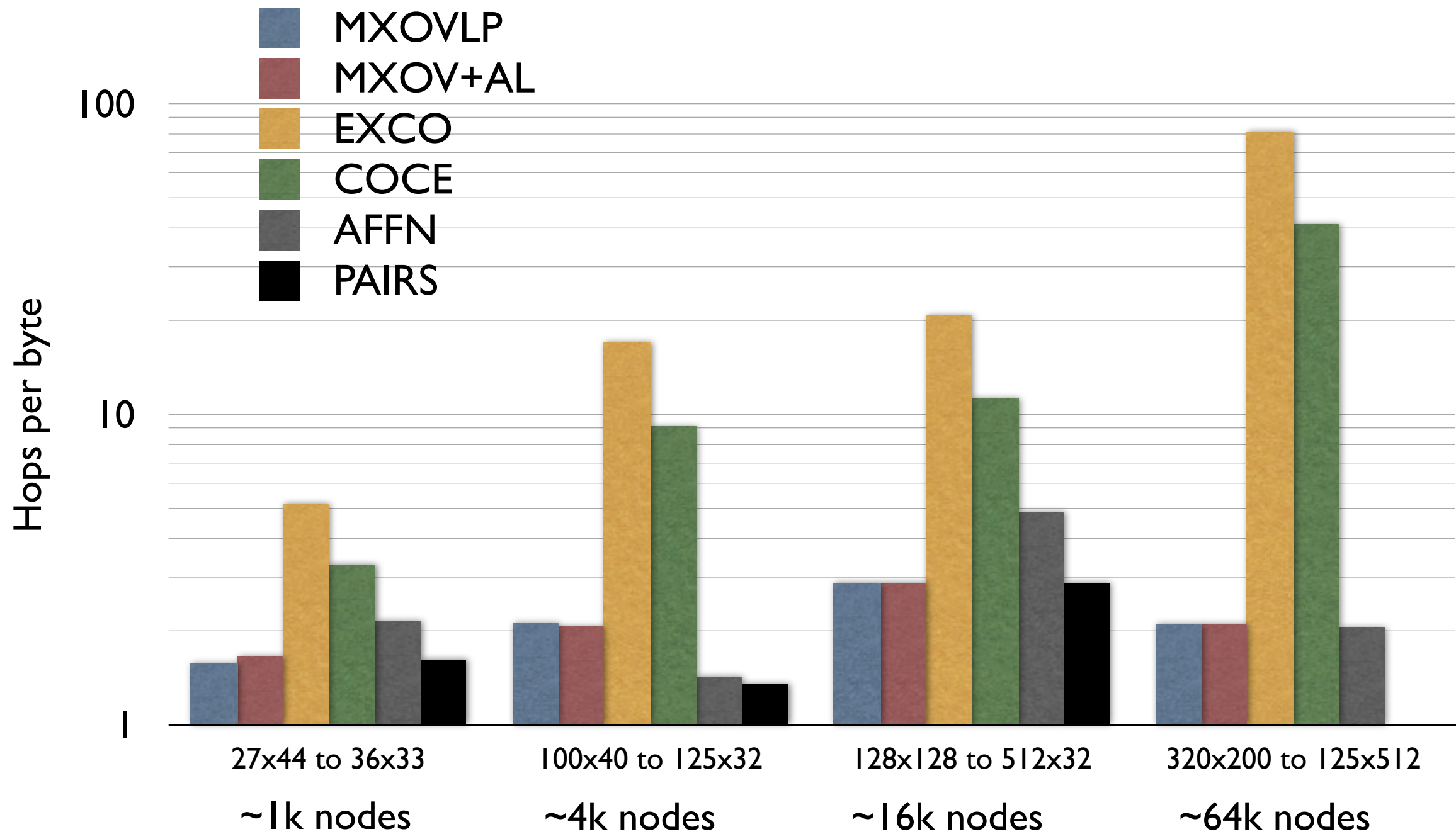
AFFN2: 1.53



AFFN3: 1.91



Evaluation

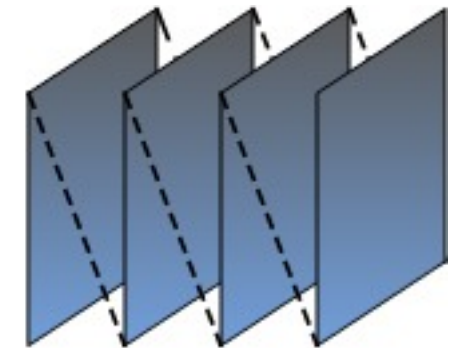


Mapping 2D Graphs to 3D

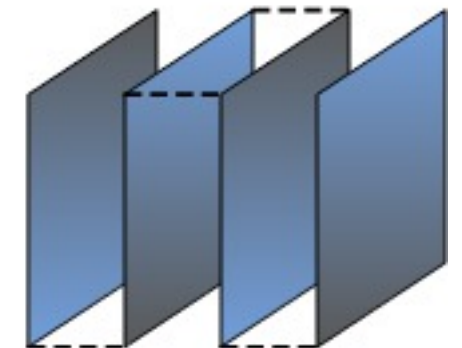
- Map a two-dimensional object graph to a three-dimensional processor graph
- Divide object graph into subgraphs once each for the number of planes
 - Stacking
 - Folding
- Best 2D to 2D heuristic chosen based on hop-bytes



2D Object Graph



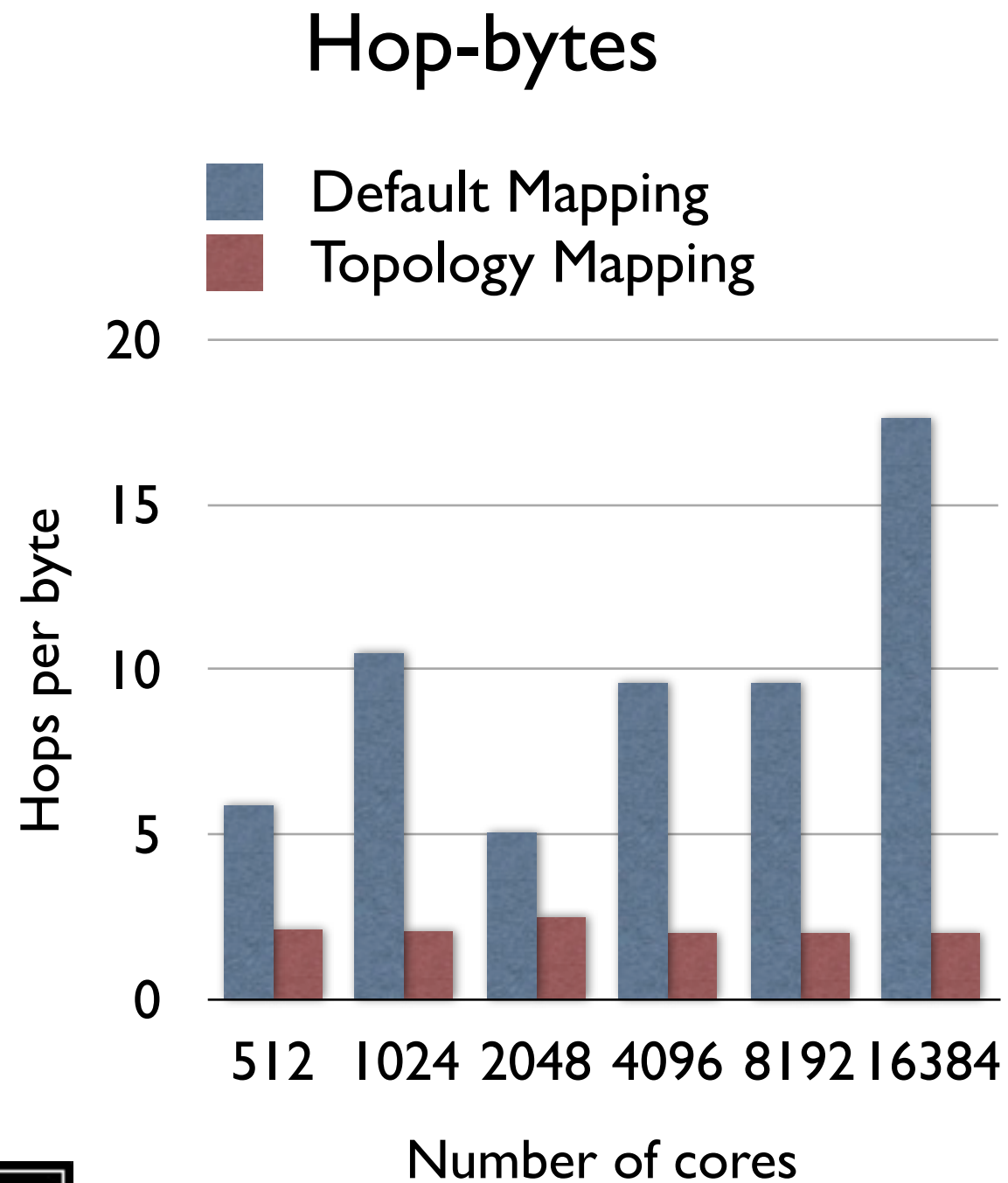
Stacking



Folding

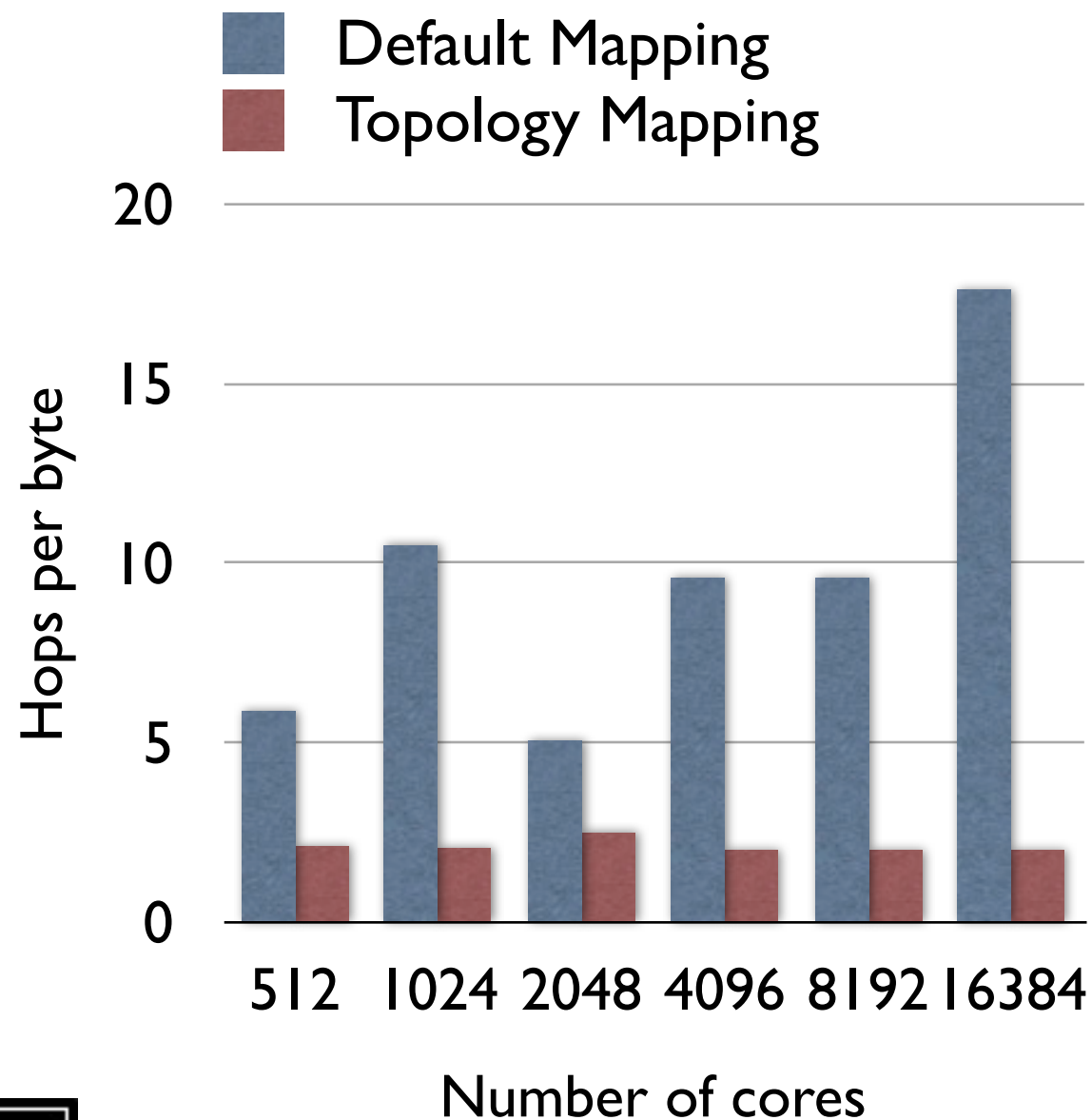


Results: 2D Stencil on Blue Gene/P

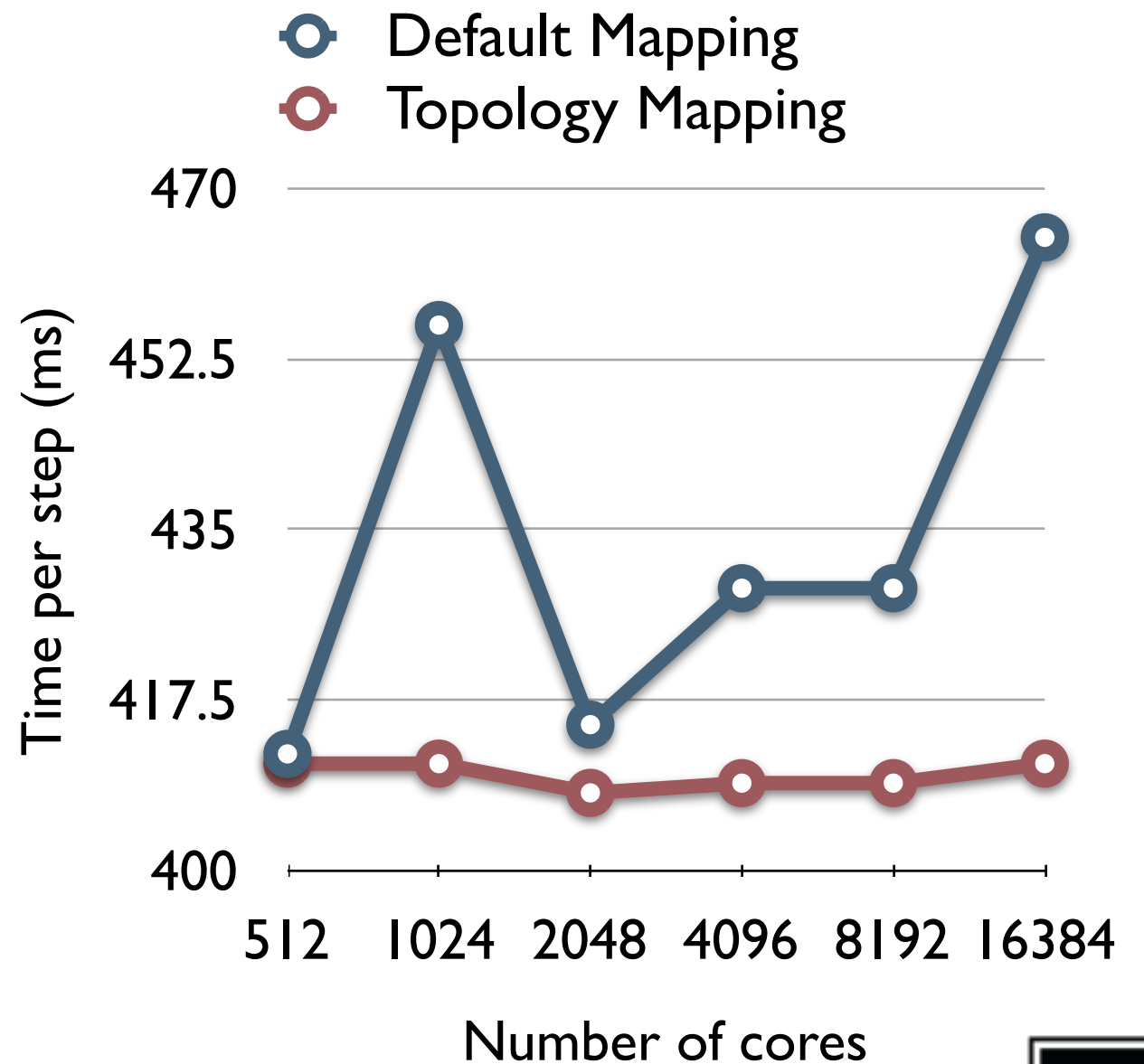


Results: 2D Stencil on Blue Gene/P

Hop-bytes



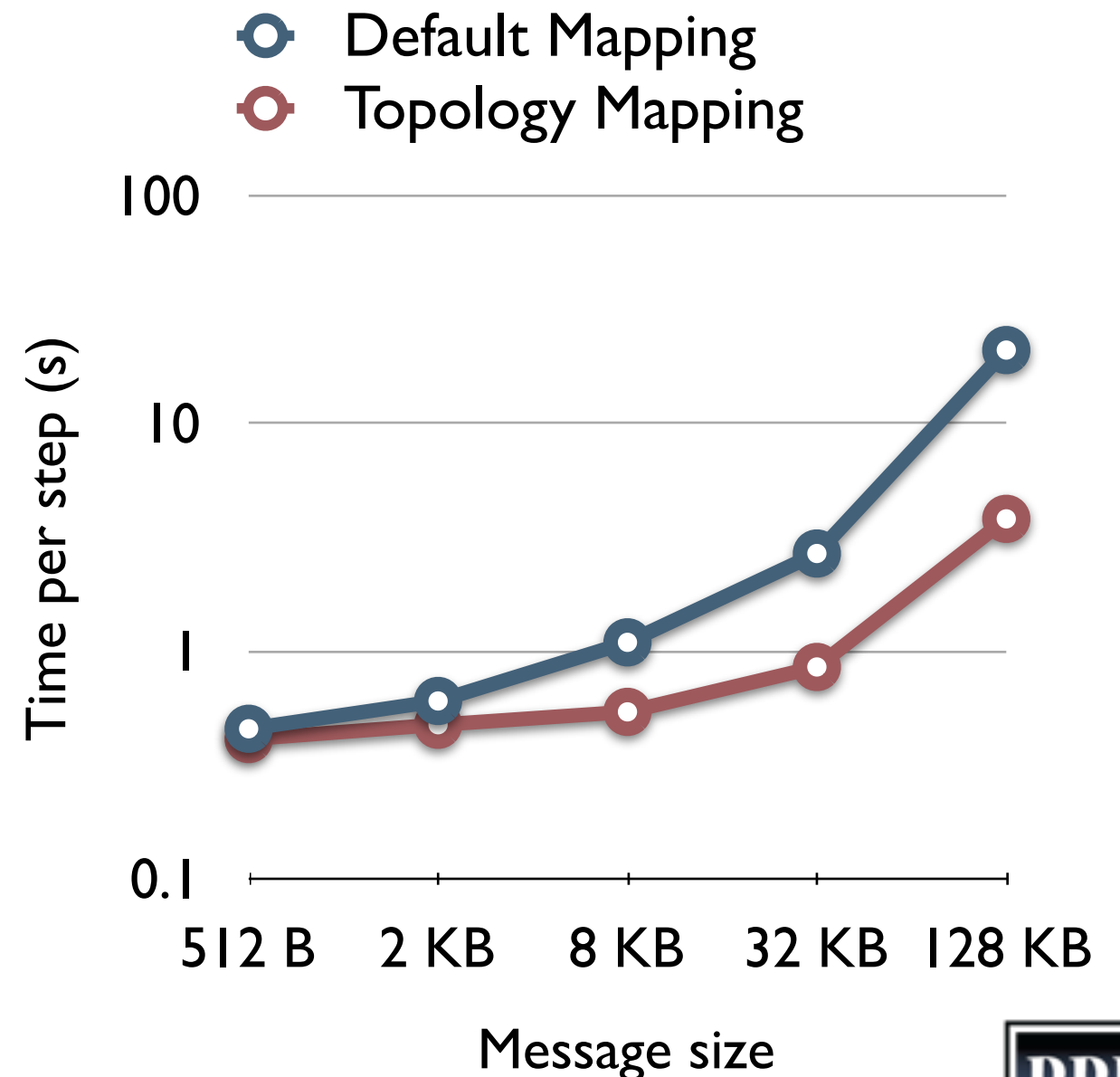
Performance



Increasing communication

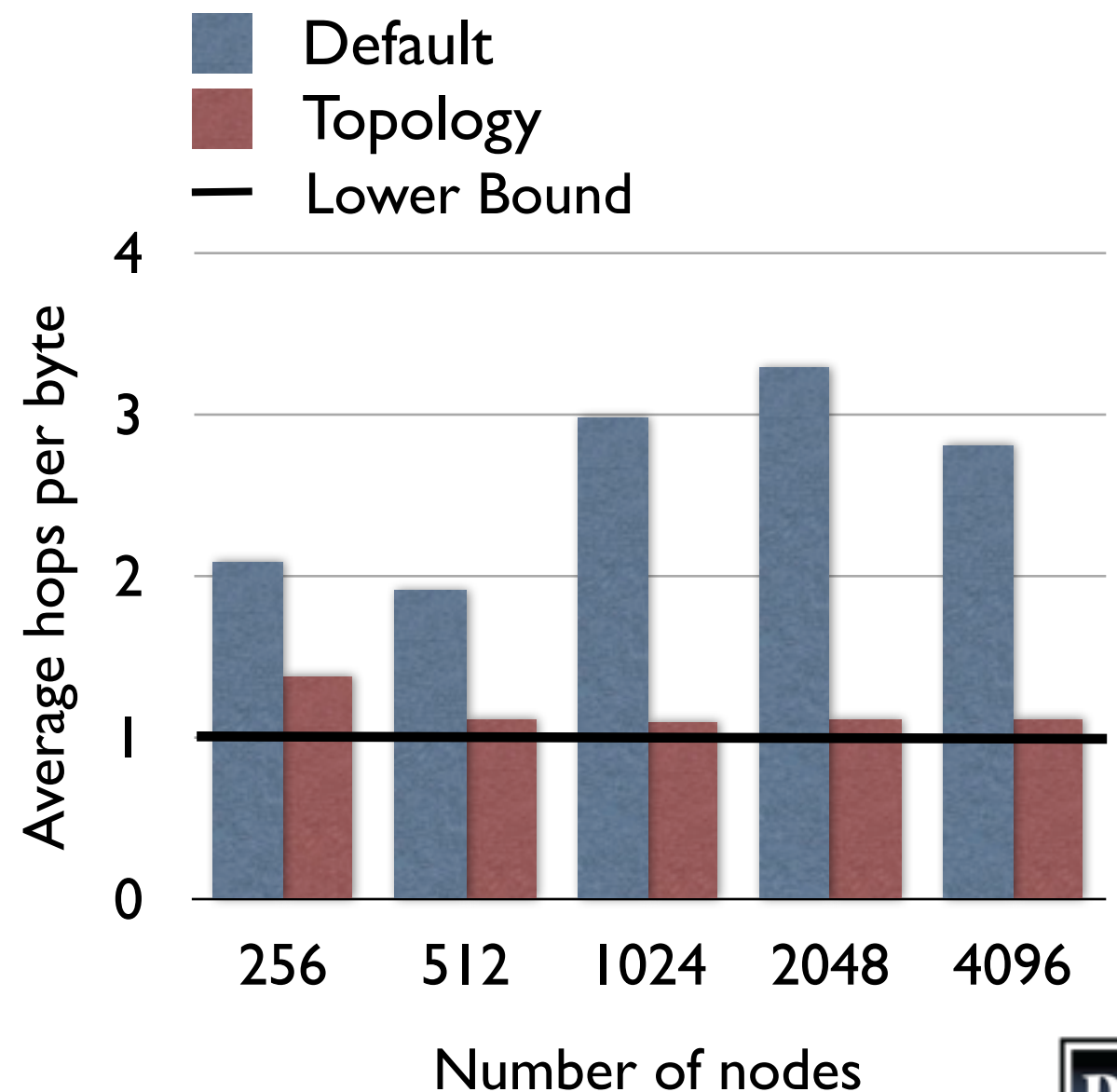
- With faster processors and constant link bandwidths
 - computation is becoming cheap
 - communication is a bottleneck
- Trend for bytes per flop
 - XT3: 8.77
 - XT4: 1.357
 - XT5: 0.23

2D Stencil on BG/P



Results: WRF on Blue Gene/P

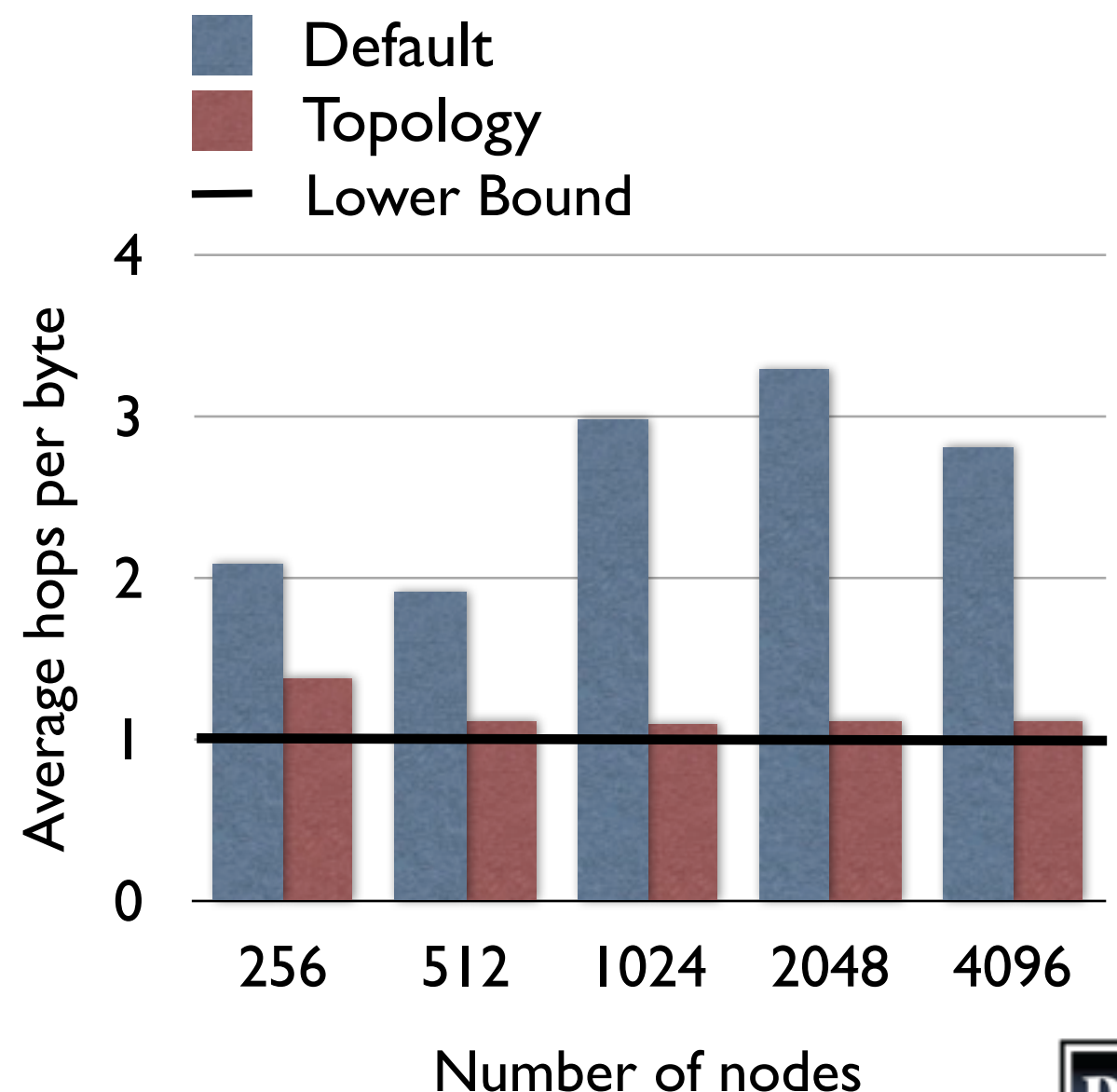
Hops from IBM HPCT



Results: WRF on Blue Gene/P

- Performance improvement negligible on 256 and 512 cores

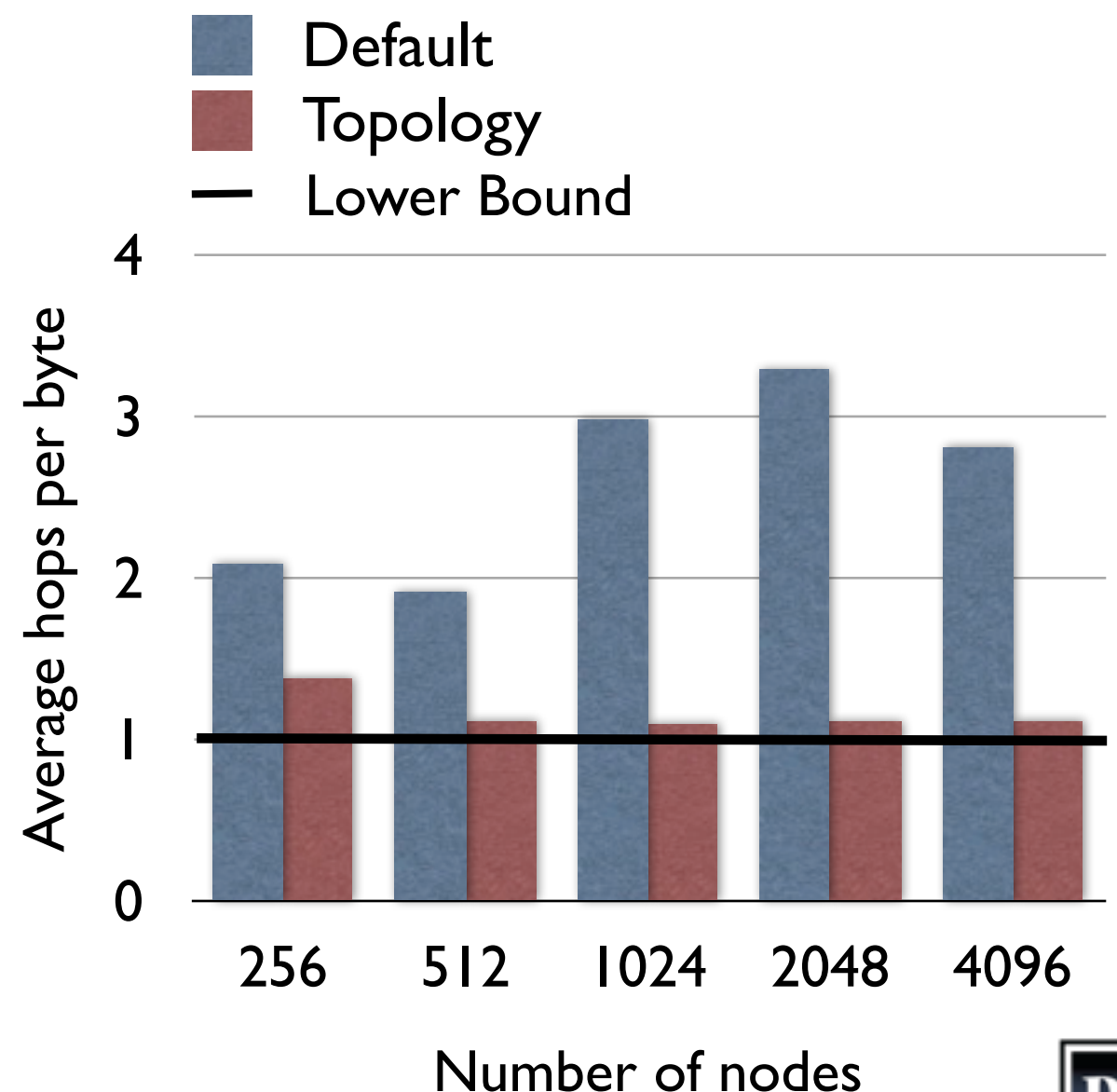
Hops from IBM HPCT



Results: WRF on Blue Gene/P

- Performance improvement negligible on 256 and 512 cores
- On 1024 nodes:
 - Hops reduce by: 63%
 - Time for communication reduces by 11%
 - Performance improves by 17%

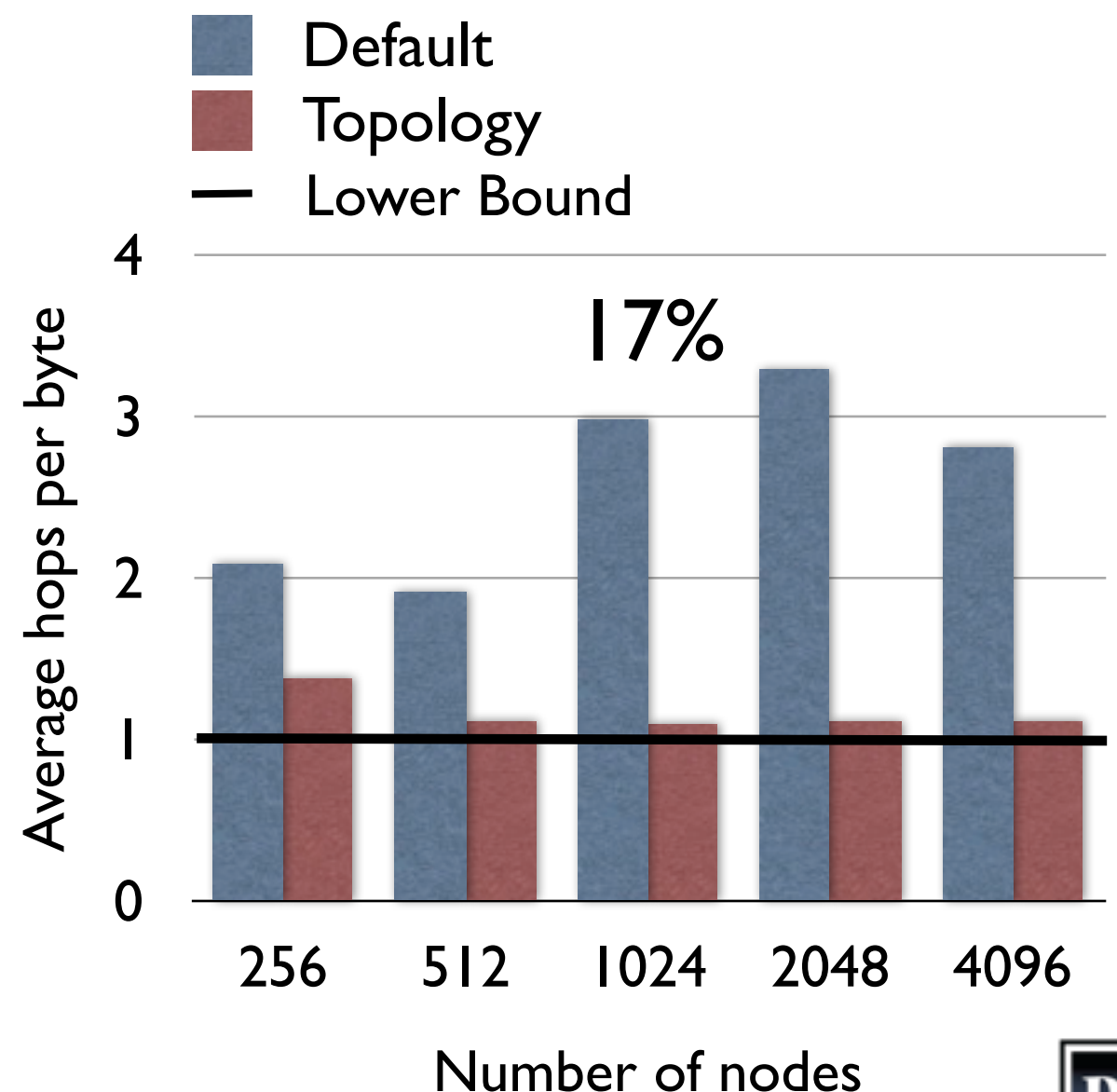
Hops from IBM HPCT



Results: WRF on Blue Gene/P

- Performance improvement negligible on 256 and 512 cores
- On 1024 nodes:
 - Hops reduce by: 63%
 - Time for communication reduces by 11%
 - Performance improves by 17%

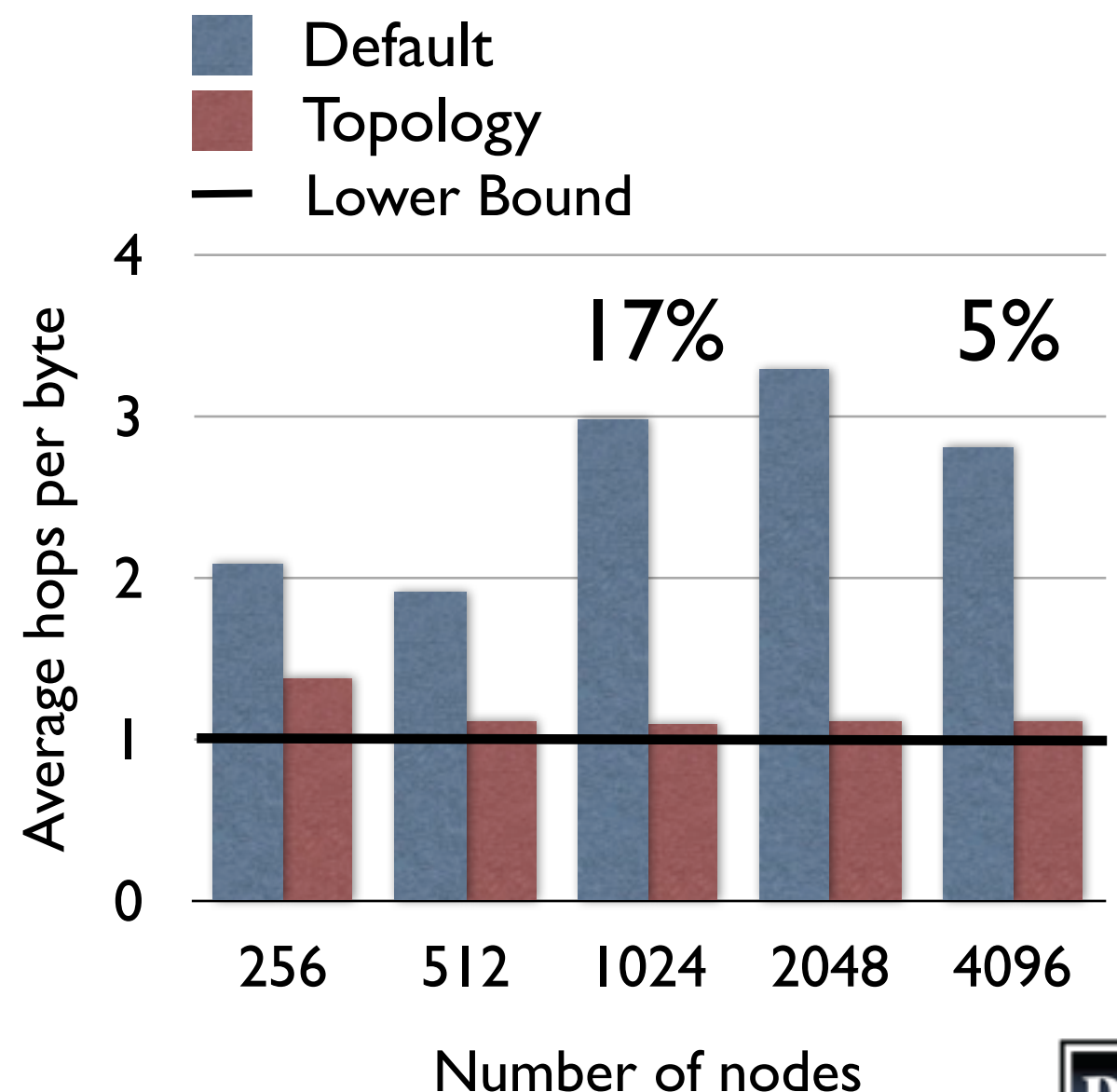
Hops from IBM HPCT



Results: WRF on Blue Gene/P

- Performance improvement negligible on 256 and 512 cores
- On 1024 nodes:
 - Hops reduce by: 63%
 - Time for communication reduces by 11%
 - Performance improves by 17%

Hops from IBM HPCT

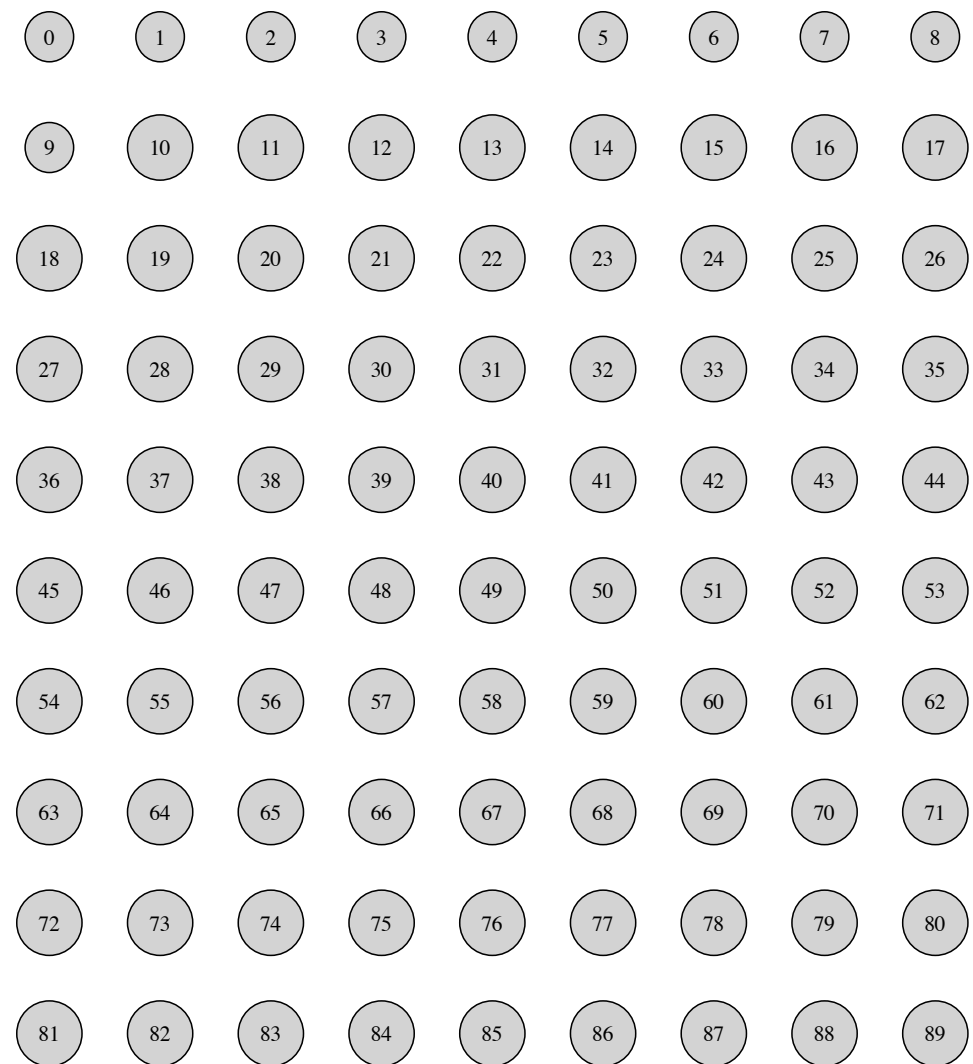
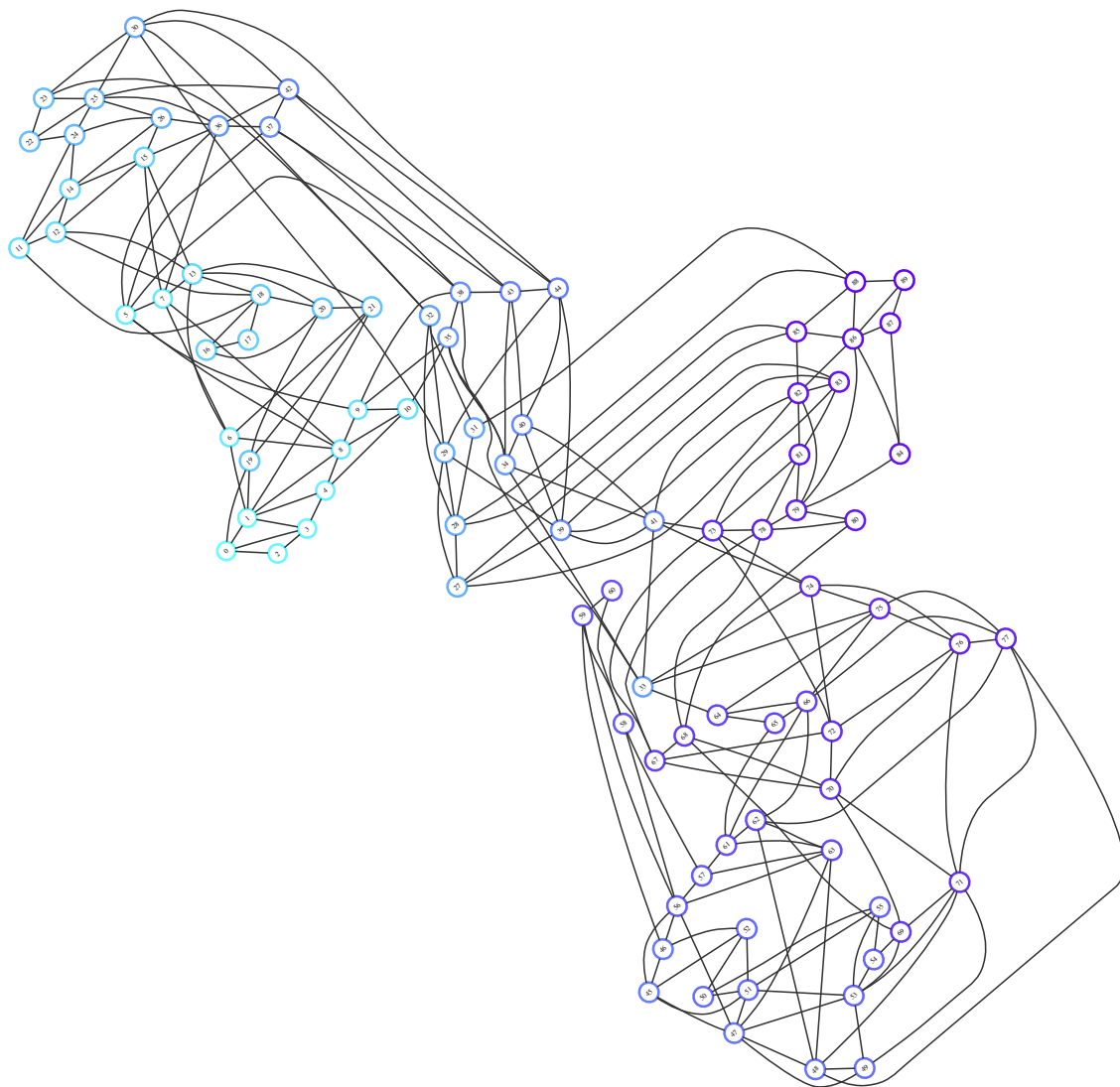


Outline

- Case studies:
 - OpenAtom
 - NAMD
- Automatic Mapping Framework
 - Pattern matching
- Heuristics for Regular Graphs
- Heuristics for Irregular Graphs



Mapping Irregular Graphs



Object graph: 90 nodes

Processor Mesh: 10 x 9



Two different scenarios

- There is no spatial information associated with the node
 - Option 1: Work without it
 - Option 2: If we know that the simulation has a geometric configuration, try to infer the structure of the graph
- We have geometric coordinate information for each node
 - Use coordinate information to avoid crossing of edges and for other optimizations



No coordinate information



No coordinate information

- Breadth first traversal (BFT)
 - Start with a random node and one end of the processor mesh
 - Map nodes as you encounter them close to their parent

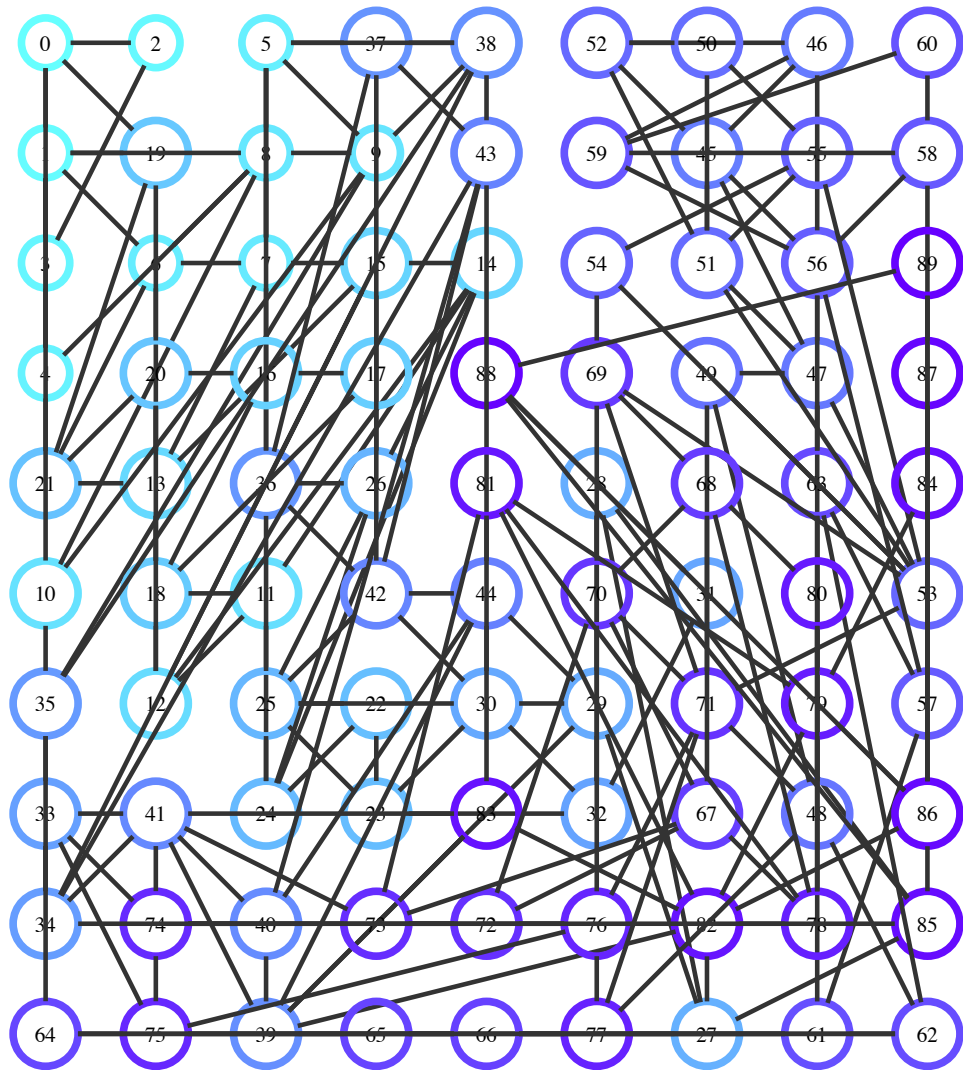


No coordinate information

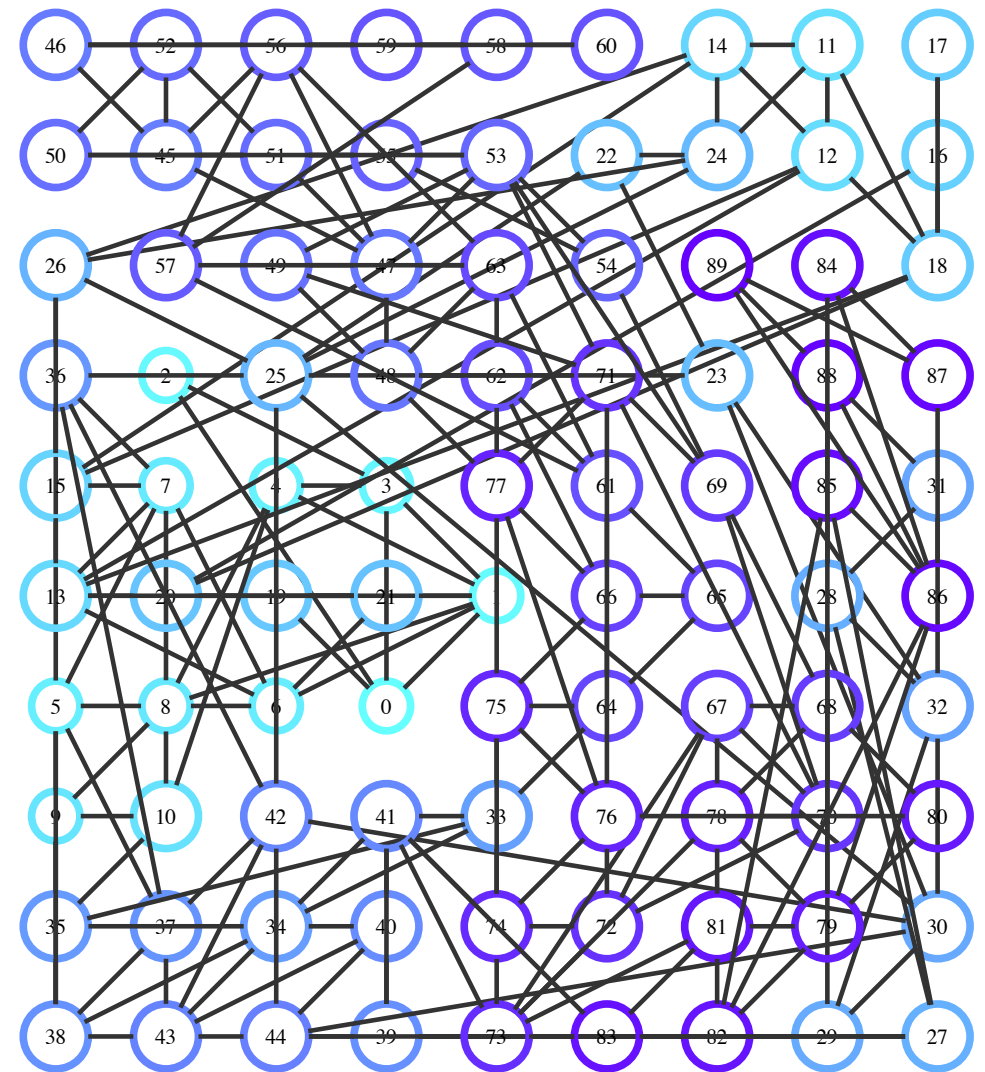
- Breadth first traversal (BFT)
 - Start with a random node and one end of the processor mesh
 - Map nodes as you encounter them close to their parent
- Max heap traversal (MHT)
 - Start with a random node and one end/center of the mesh
 - Put neighbors of a mapped node into the heap (node at the top is the one with maximum **number of mapped neighbors**)
 - Map elements in the heap one by one around the centroid of their mapped neighbors



Mapping visualization



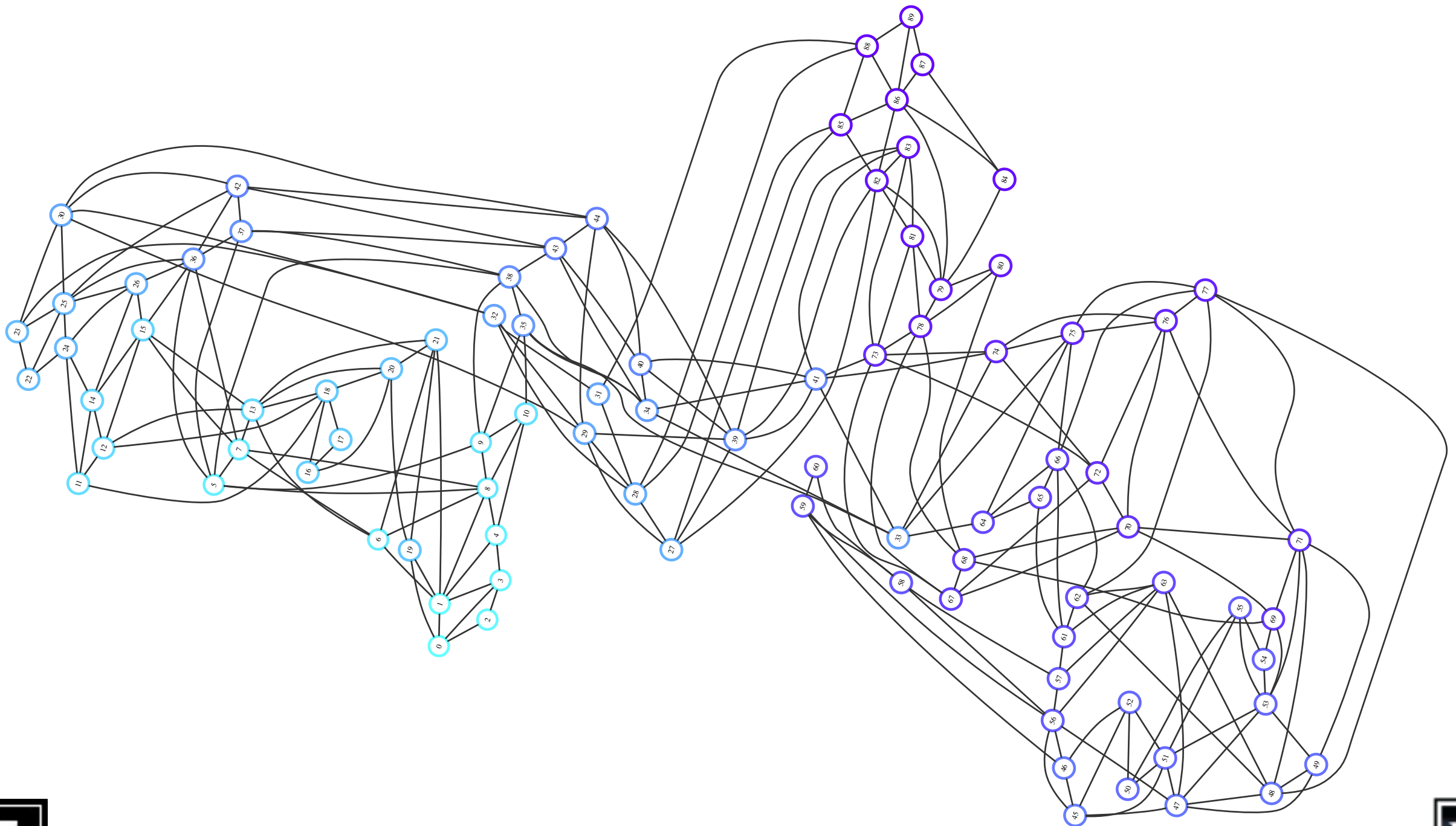
BFT: 2.89



MHT: 2.69



Inferring the spatial placement



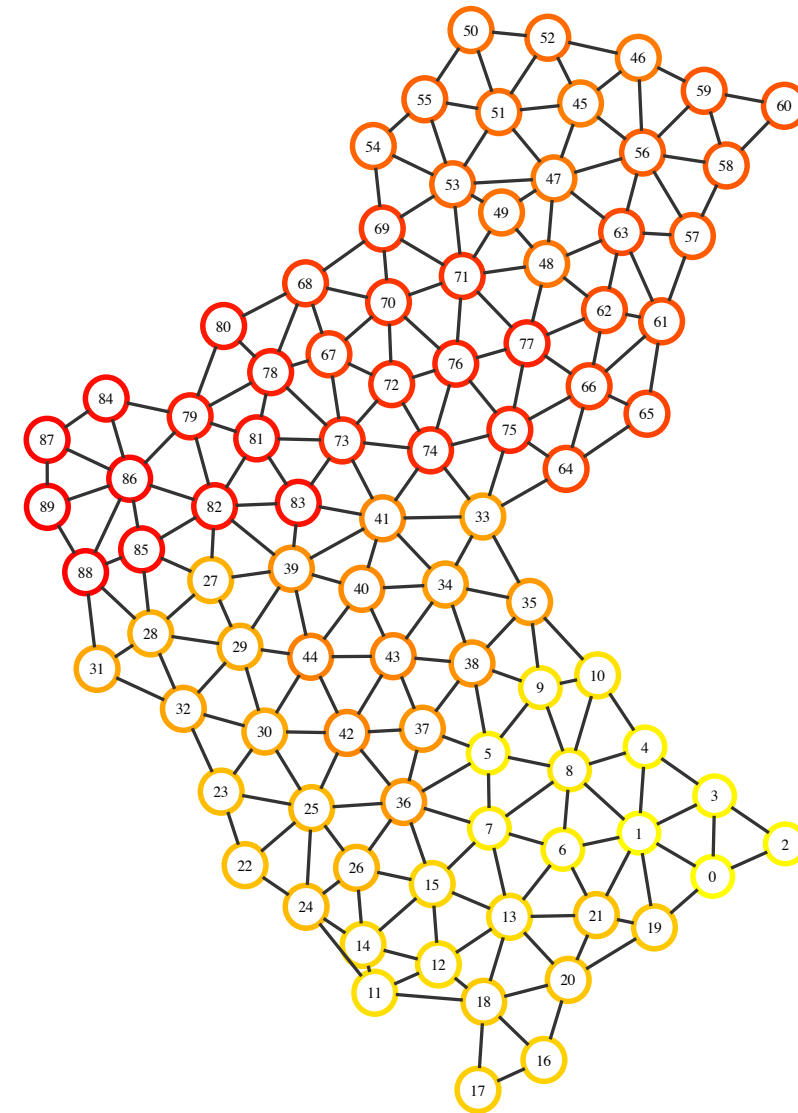
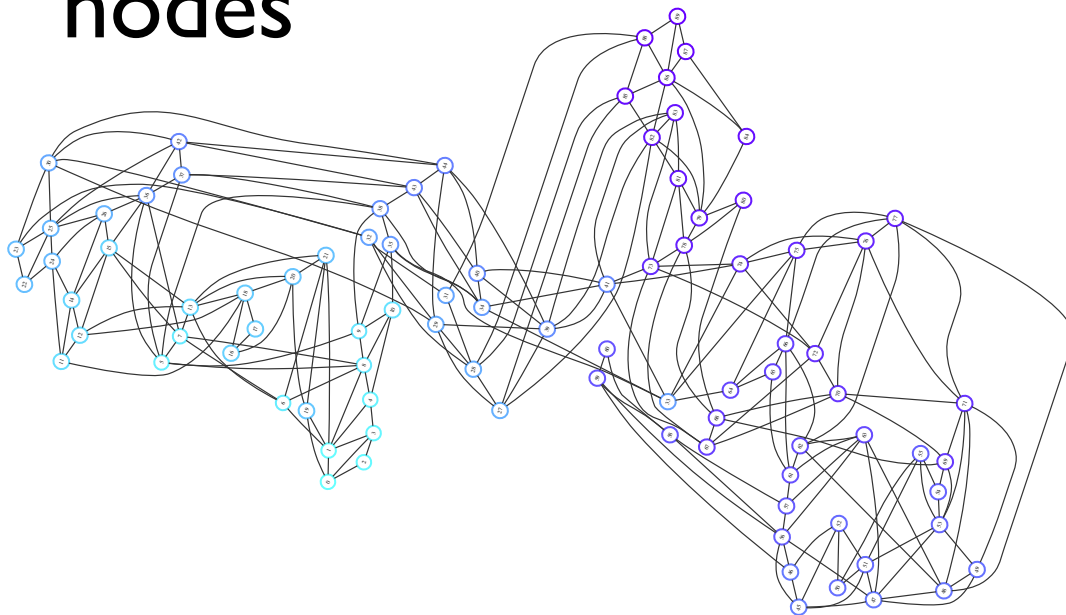
Inferring the spatial placement

- Graph layout algorithms
 - Force-based layout to reduce the total energy in the system
- Use the graphviz library to obtain coordinates of the nodes



Inferring the spatial placement

- Graph layout algorithms
 - Force-based layout to reduce the total energy in the system
- Use the graphviz library to obtain coordinates of the nodes

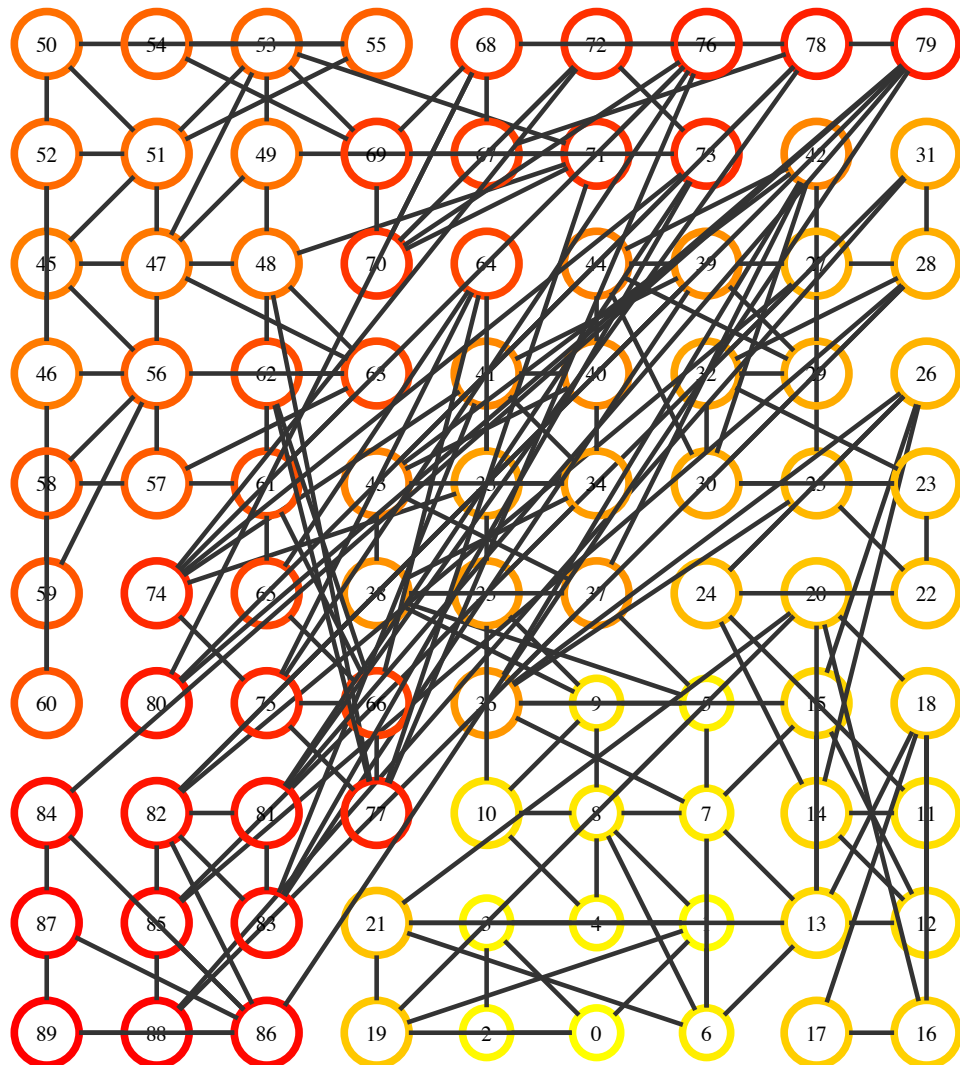


With coordinate information

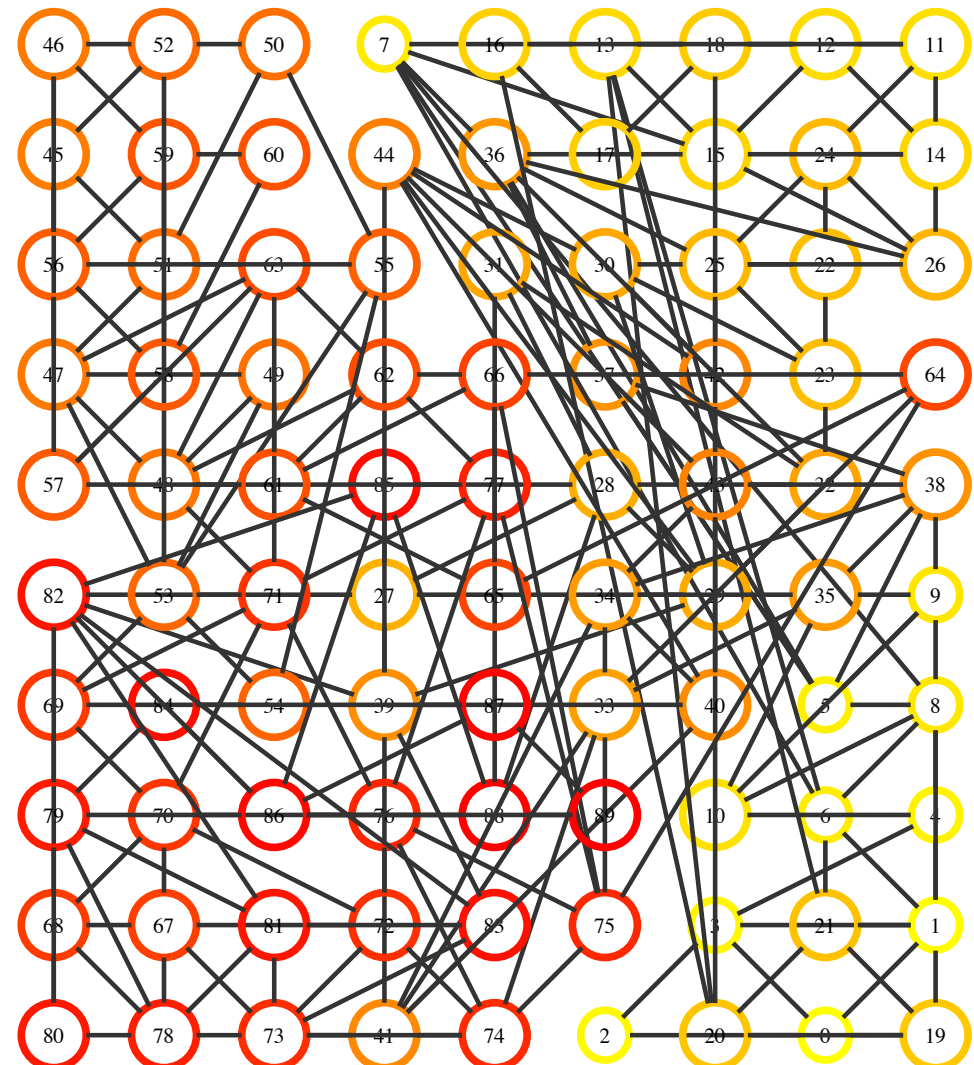
- **Affine Mapping (AFFN)**
 - Stretch/shrink the object graph (based on coordinates of nodes) to map it on to the processor grid
 - In case of conflicts for the same processor, spiral around that processor
- **Corners to Center (COCE)**
 - Use four corners of the object graph based on coordinates
 - Start mapping simultaneously from all sides
 - Place nodes encountered during a BFT close to their parents



Mapping visualization



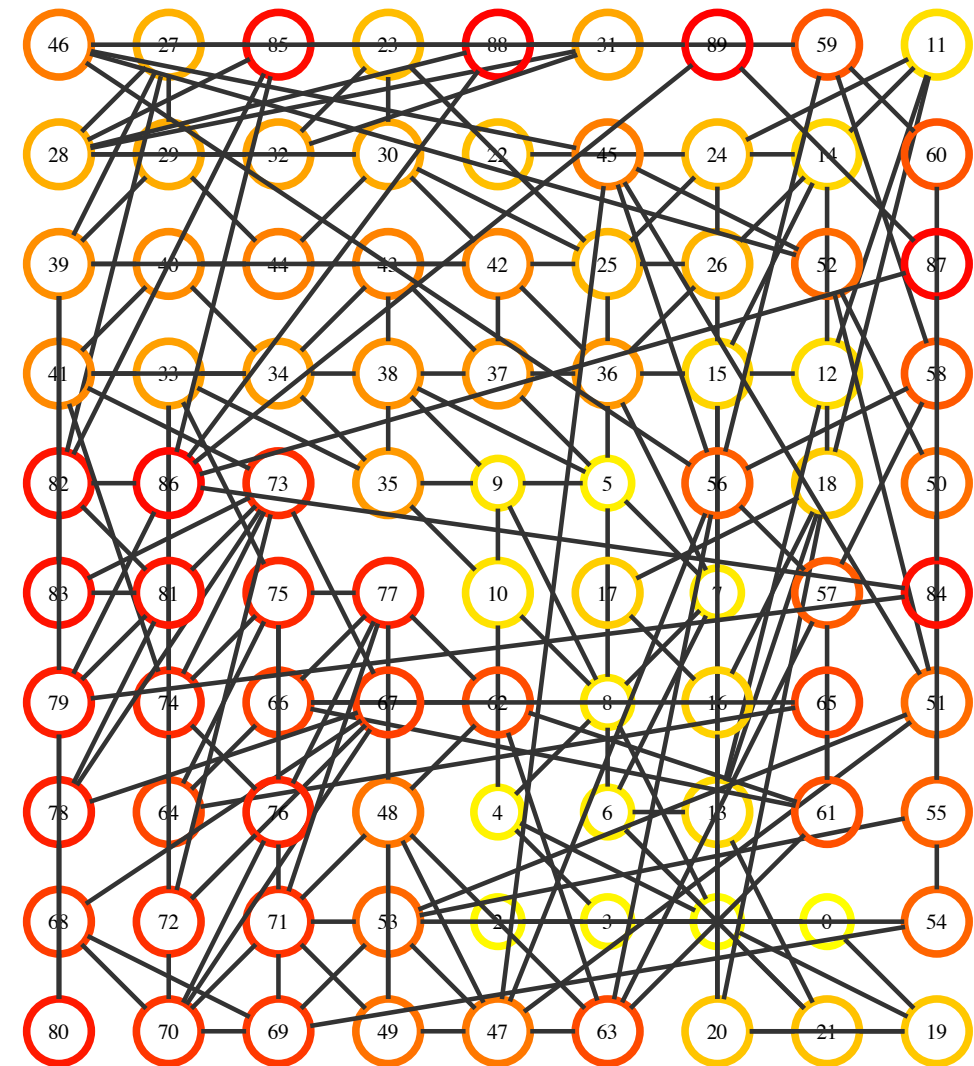
AFFN: 3.17



COCE: 2.88



- **COCE+MHT Hybrid:**
 - We fix four nodes at geometric corners of the mesh to four processors in 2D
 - Put neighbors of these nodes into a max heap
- **Map from all sides inwards**
 - Starting from centroid of mapped neighbors



COCE: 2.78



Time Complexity



December 16th, 2010

CNIC @ CAS © Abhinav Bhatle



49

Time Complexity

- All algorithms discussed above choose a desired processor and spiral around it to find the nearest available processor
- Heuristics generally applicable to any topology



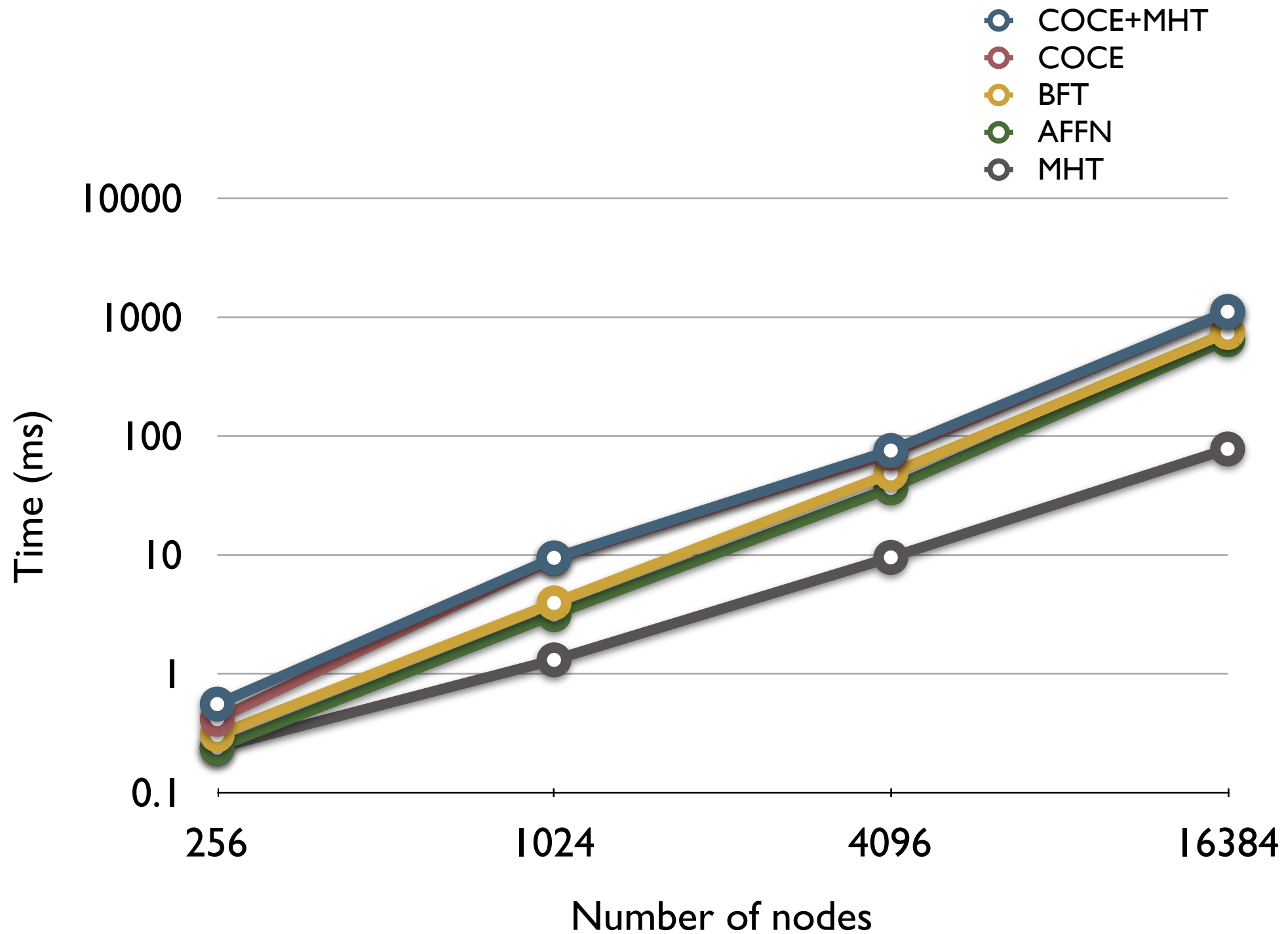
Time Complexity

- All algorithms discussed above choose a desired processor and spiral around it to find the nearest available processor
 - Heuristics generally applicable to any topology
- Depending on the running time of findNext:

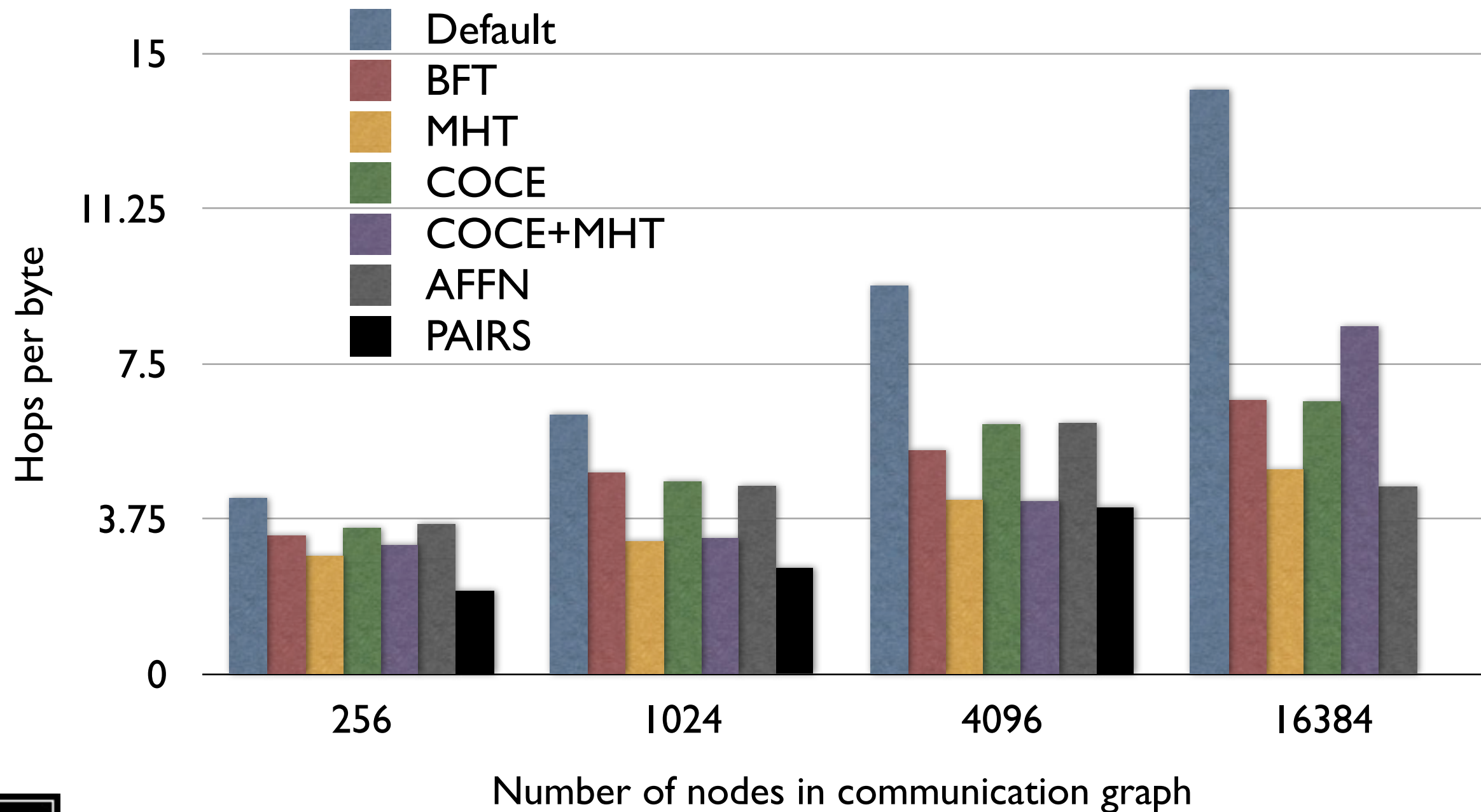
BFT	COCE	AFFN	MHT	COCE+MHT
$O(n)$	$O(n)$	$O(n)$	$O(n \log n)$	$O(n \log n)$
$O(n (\log n)^2)$	$O(n (\log n)^2)$	$O(n (\log n)^2)$	$O(n (\log n)^2)$	$O(n (\log n)^2)$



Running Time



Results: simple2D



Summary

- Contention in modern day supercomputers can impact performance: makes mapping important
- Certain classes of applications (latency sensitive, communication bound) benefit most
 - OpenAtom shows performance improvements of up to 50%
 - NAMD - improvements for > 4k cores
- Developing an automatic mapping framework
 - Relieve the application developer of the mapping burden



Summary

- Topology discovery: Topology Manager API
- Object Communication Graph: Profiling, Instrumentation
- Pattern matching
 - Regular graphs
 - Irregular graphs
- Suite of heuristics for mapping
- Distributed strategies with global view



Future Work

- More sophisticated algorithms for pattern matching and mapping
 - Multicast and many-to-many patterns
- Handling multiple communication graphs
 - Simultaneous or occurring in different phases
- Extension of the work on distributed load balancing



Thanks

Abhinav Bhatele. Automating Topology Aware Mapping for Supercomputers, PhD Thesis, Department of Computer Science, University of Illinois. <http://hdl.handle.net/2142/16578>