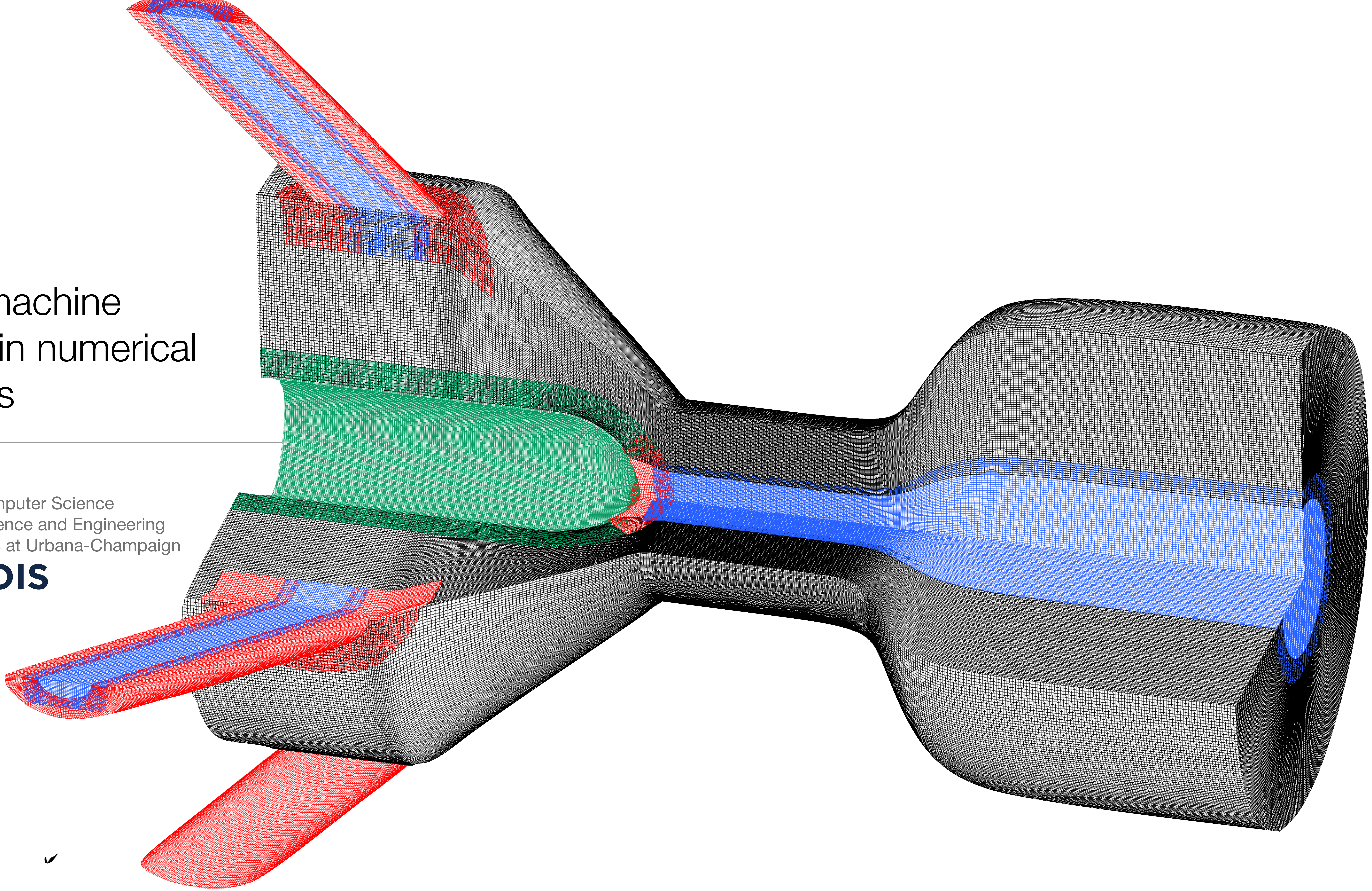
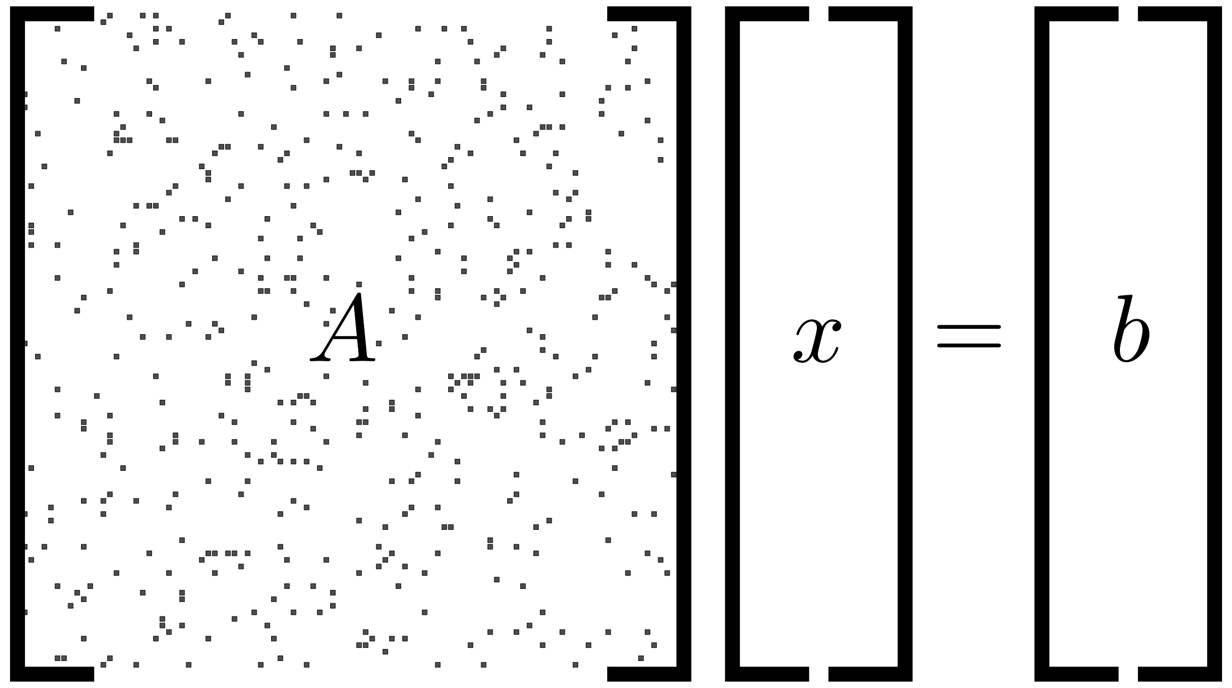
Utilizing machine topology in numerical algorithms

Luke Olson
Department of Computer Science
Computational Science and Engineering
University of Illinois at Urbana-Champaign

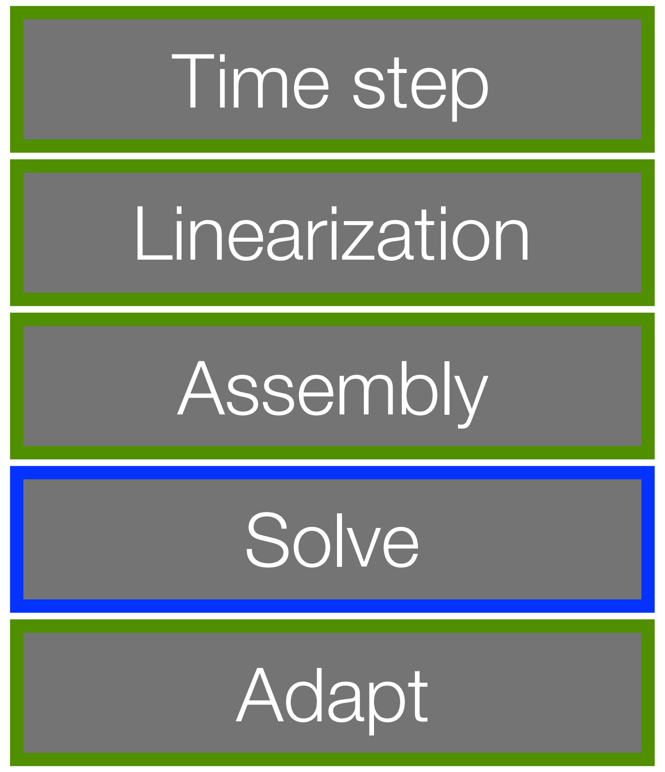


Overview

- **I use Blue Waters to solve large sparse linear systems and to study the performance**



- **Why?**



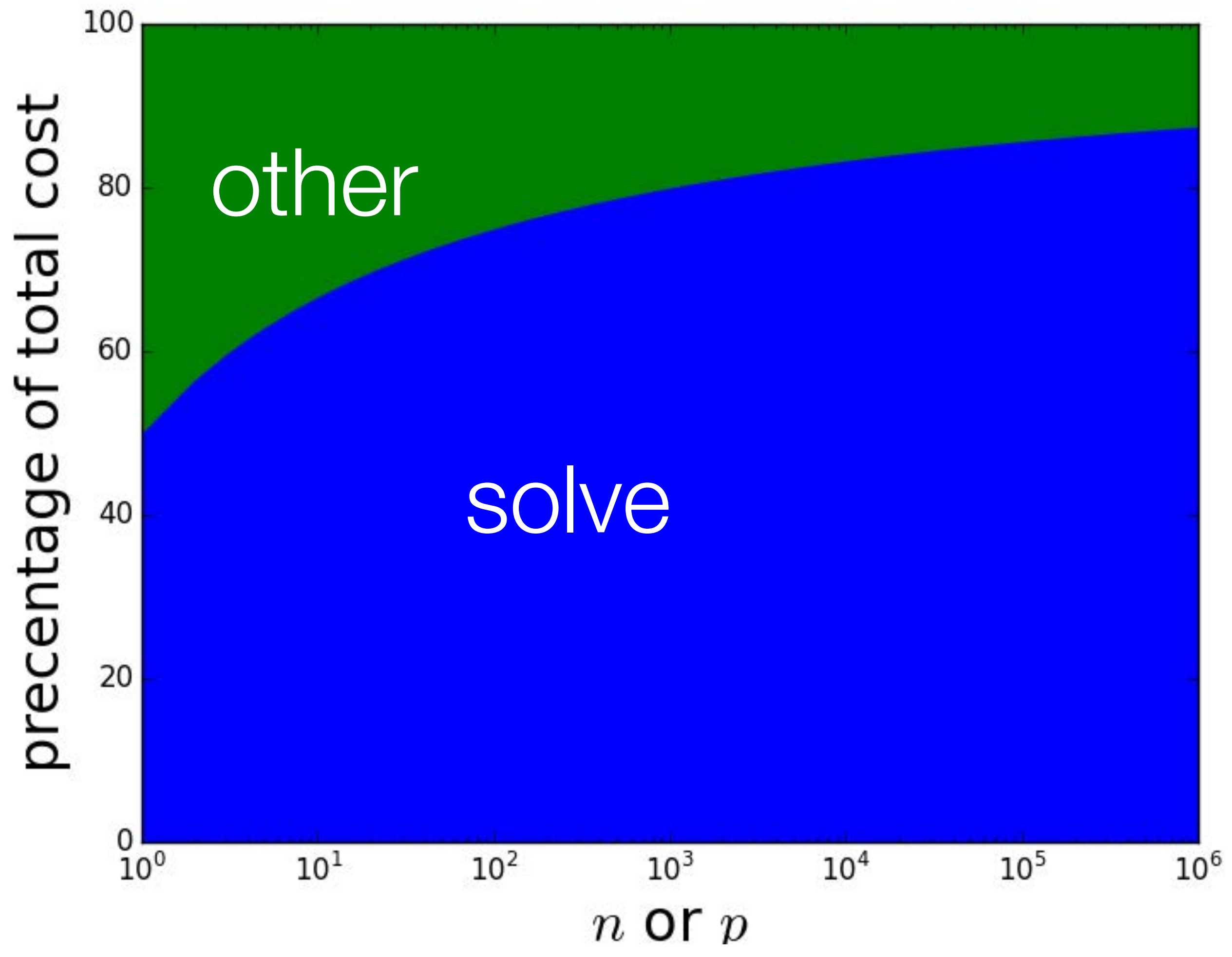
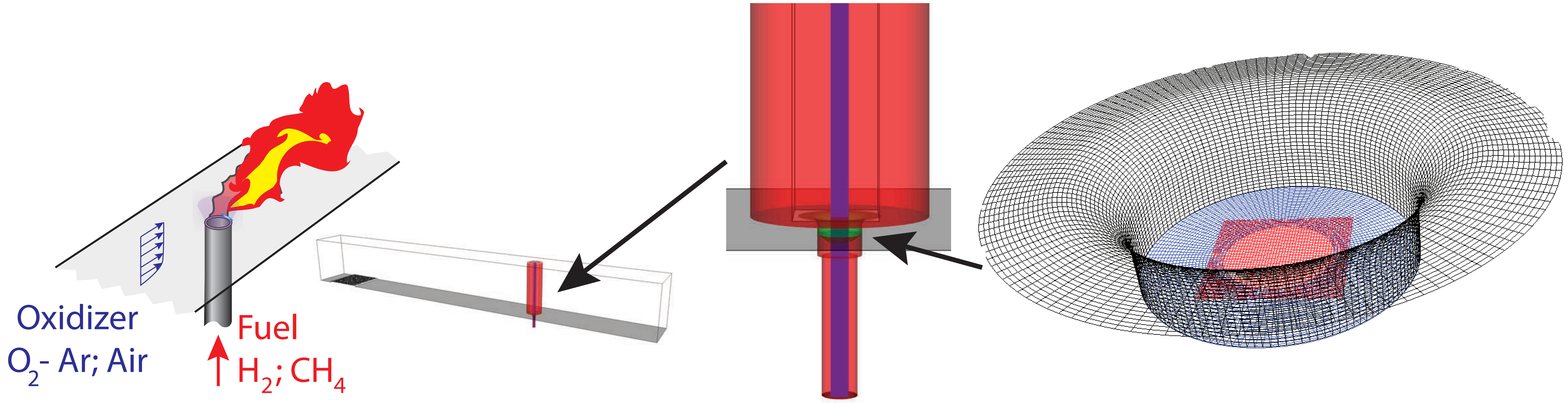
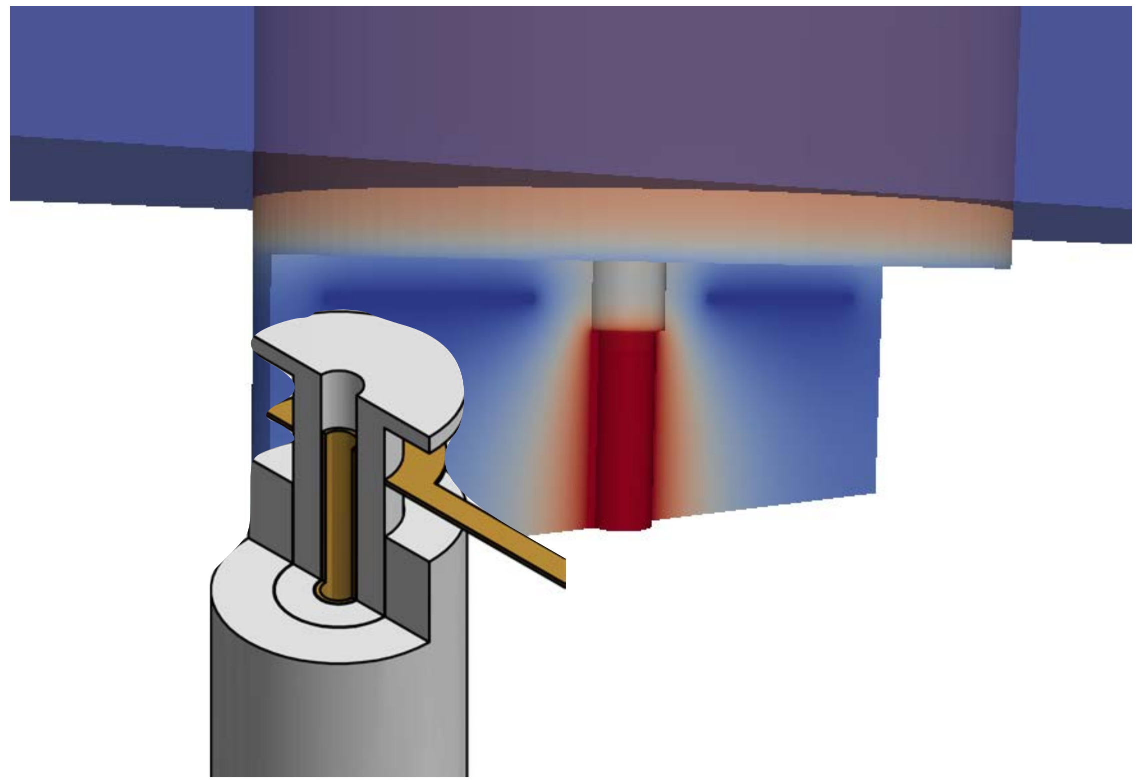


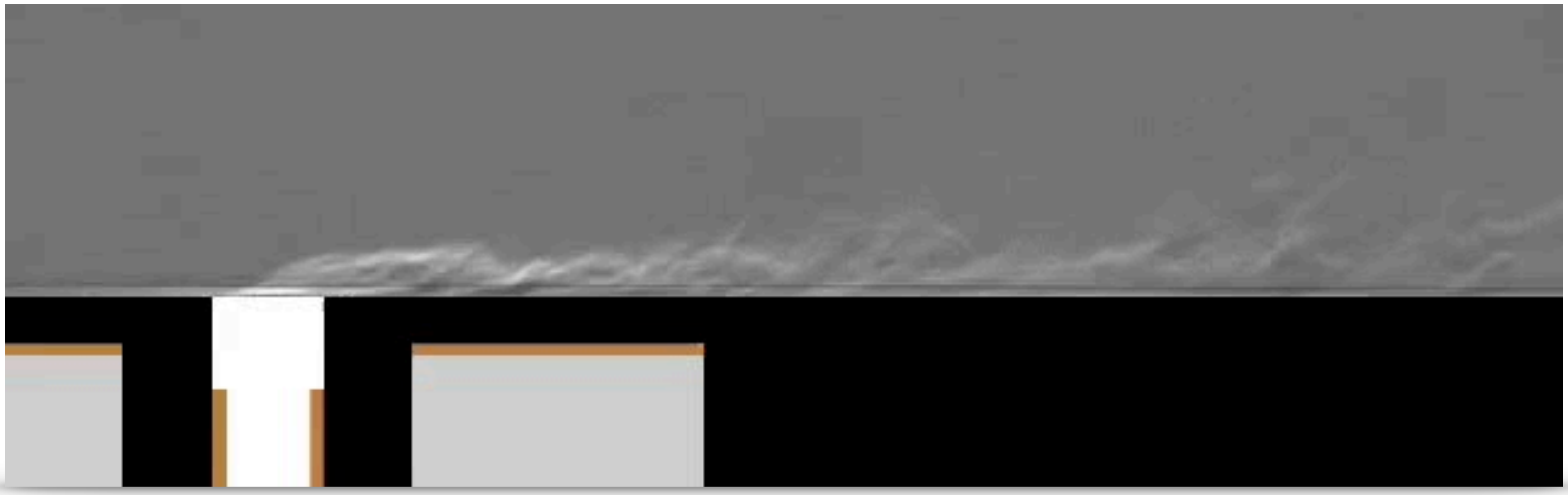
Image: David Keyes??

Application: Plasma-coupled Combustion

- XPACC The Center for Exascale Simulation of Plasma-Coupled Combustion @ Illinois

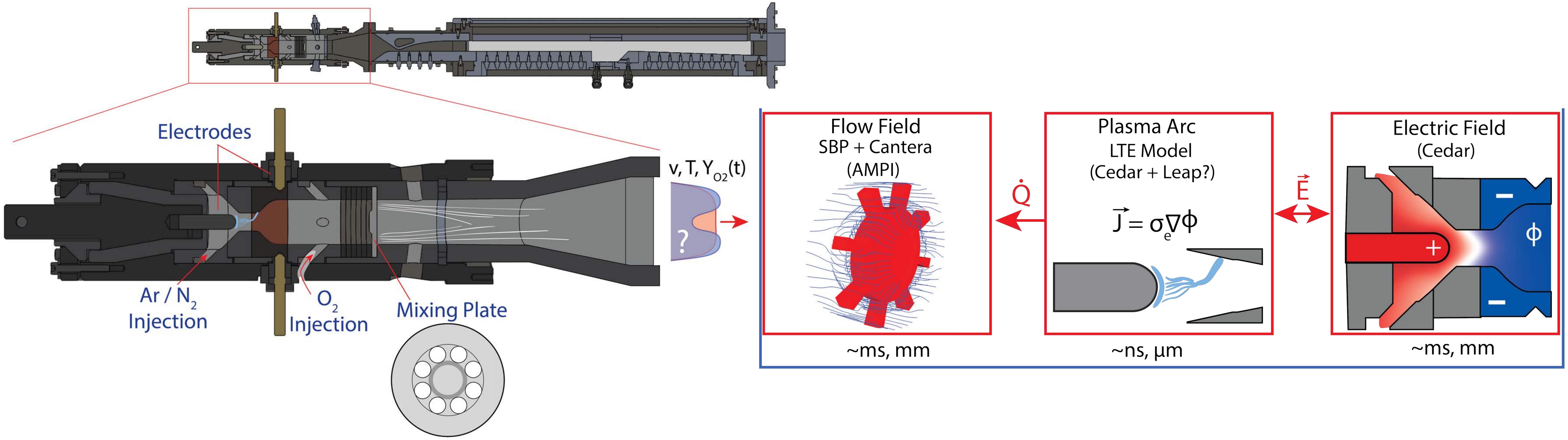






Application: Plasma-coupled Combustion

- XPACC The Center for Exascale Simulation of Plasma-Coupled Combustion @ Illinois

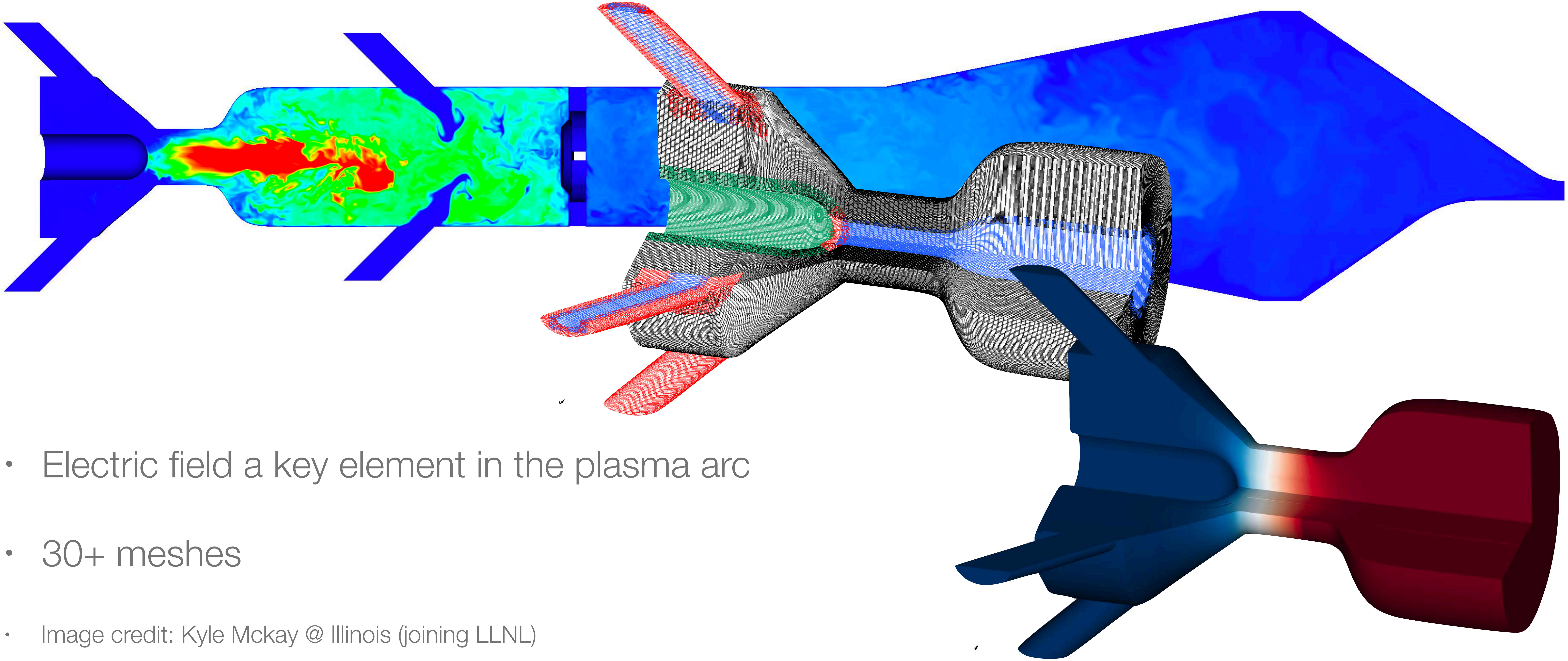


- Electric conductivity influences the electric field and current density over time

$$\nabla \cdot \sigma_r \nabla \phi = g$$

Application: Plasma-coupled Combustion

- XPACC The Center for Exascale Simulation of Plasma-Coupled Combustion @ Illinois



- Electric field a key element in the plasma arc
- 30+ meshes
- Image credit: Kyle Mckay @ Illinois (joining LLNL)

Why Blue Waters?

- Sparse matrix operations are communication dominant — performance models play a key role.

- Exploiting machine layout plays an important role in addressing bottlenecks in communication.

- Blue Waters has enabled us to develop/test/scales codes to address these issues

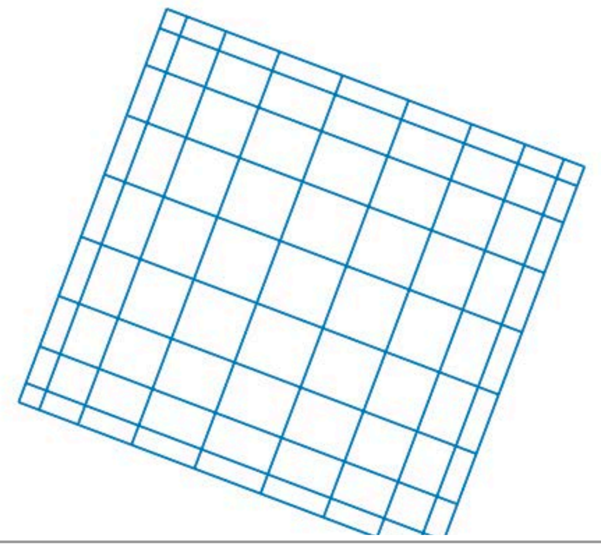
Structured Multigrid

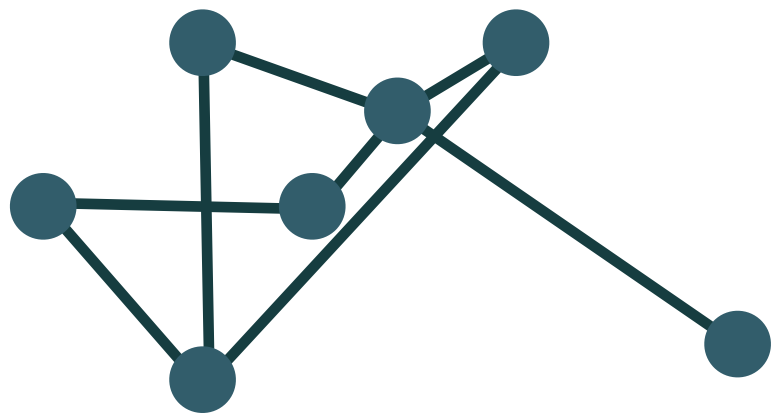
Cedar
Parallel structured multigrid solvers

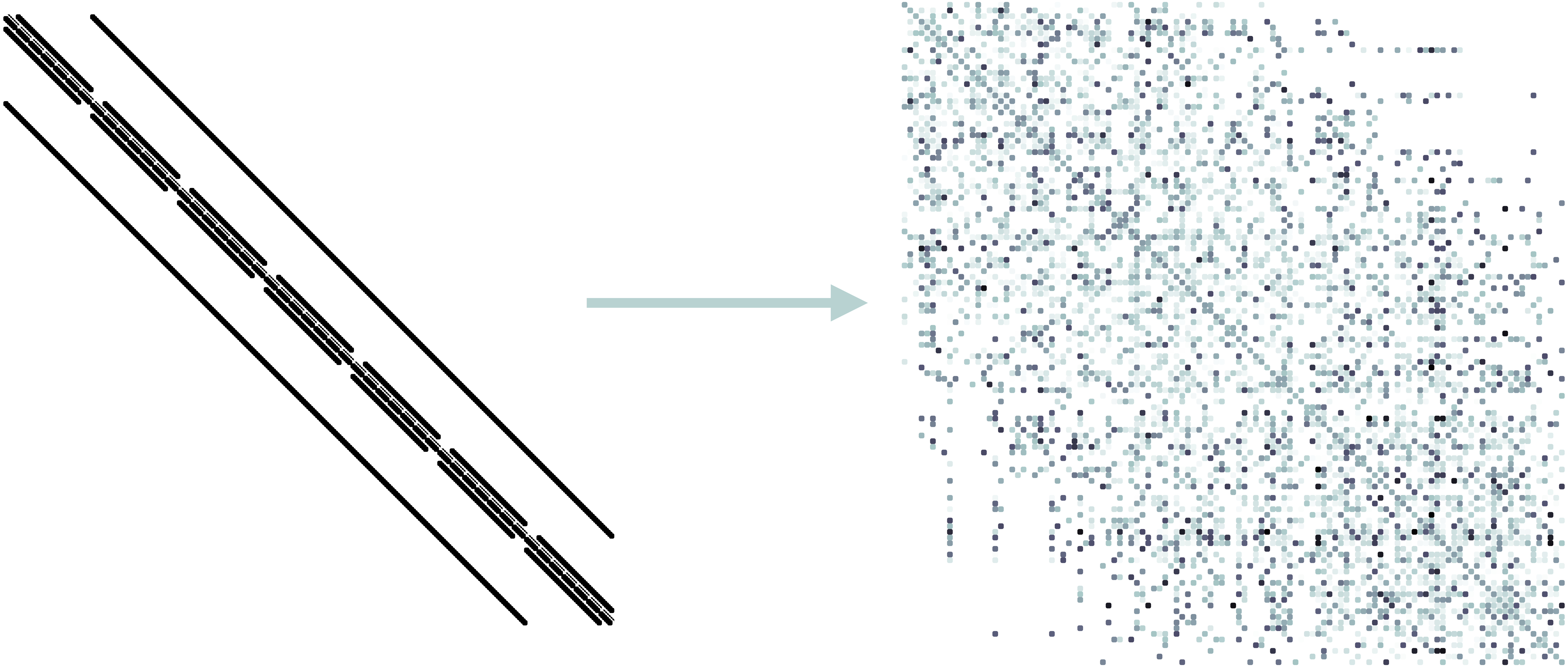
Algebraic Multigrid

RAPtor: parallel algebraic multigrid

Sparsity (a.k.a. *data relationships*)

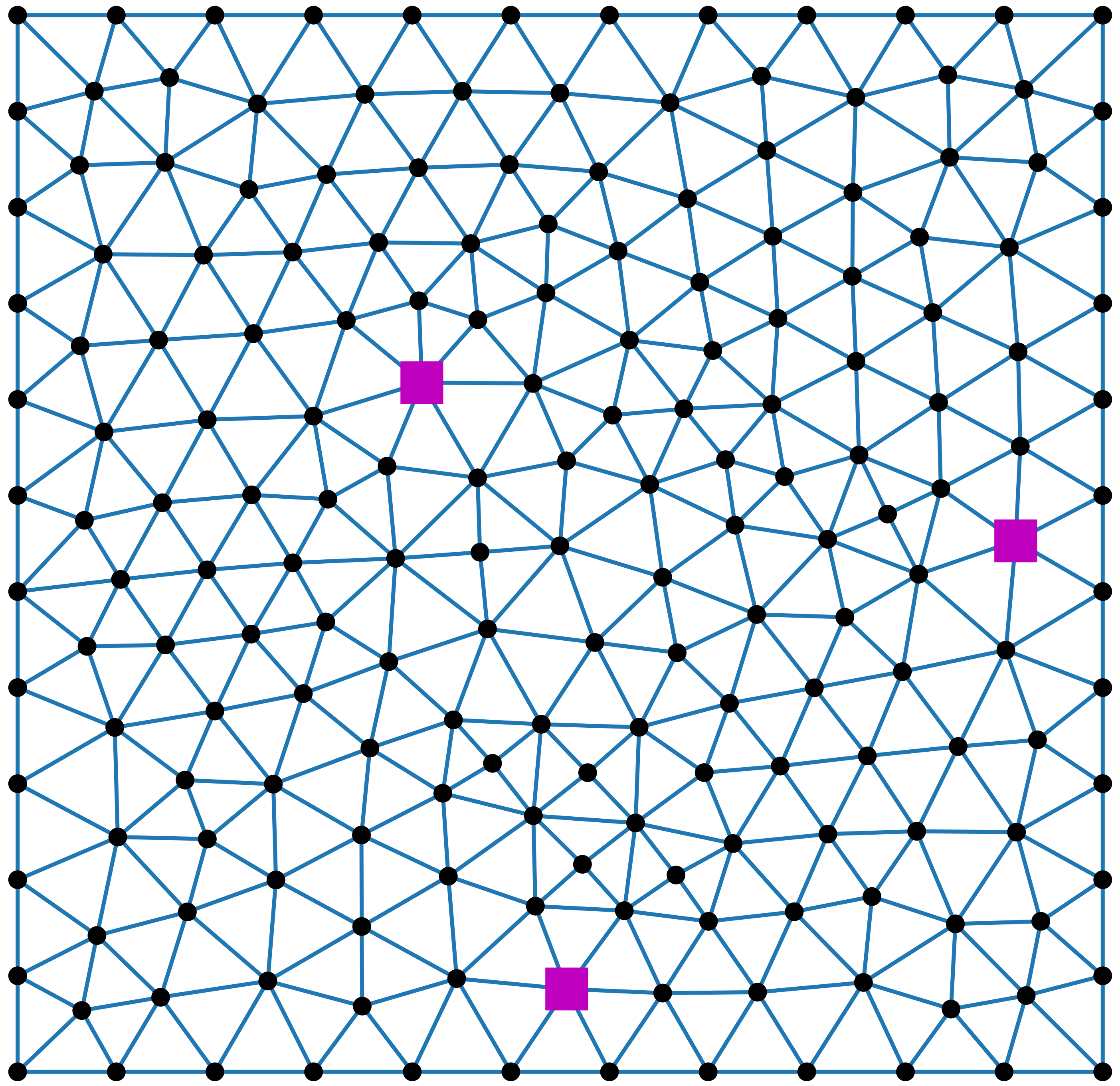




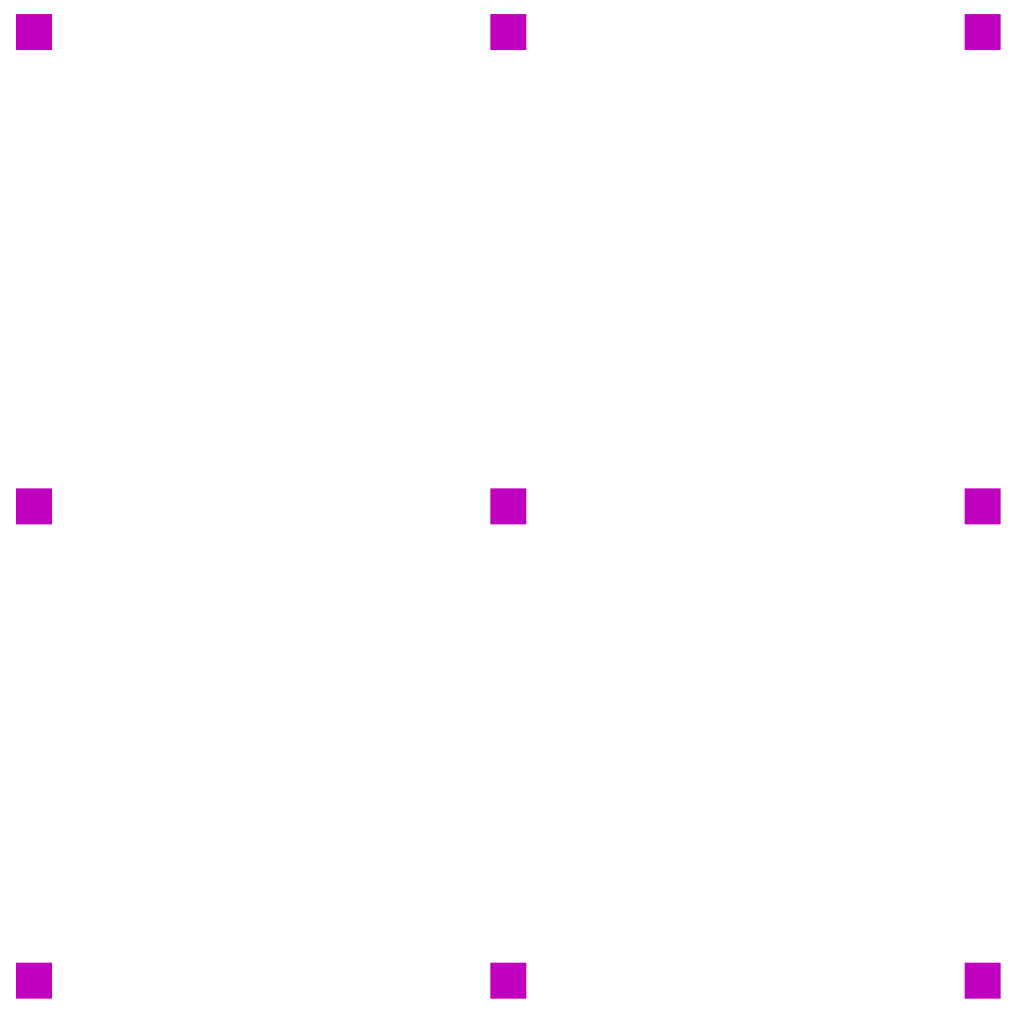


Multilevel solvers for solving $Ax = b$

- Series or hierarchy of successively smaller (and more dense) problems
- Iteratively annihilate the error in the solution through this hierarchy of problems



Problem



A

SpMM $A * B$

SpMV $A * v$

Constructing the solver
~SpMM + several SpMVs

Applying the solver
~many SpMVs

Sparse Matrices

$$y = A * x$$

	0	1	2	3	4
0	1.1		2.2		
1		3.3		4.4	
2	5.5		6.6		7.7
3		8.8		9.9	
4				10.0	11.1

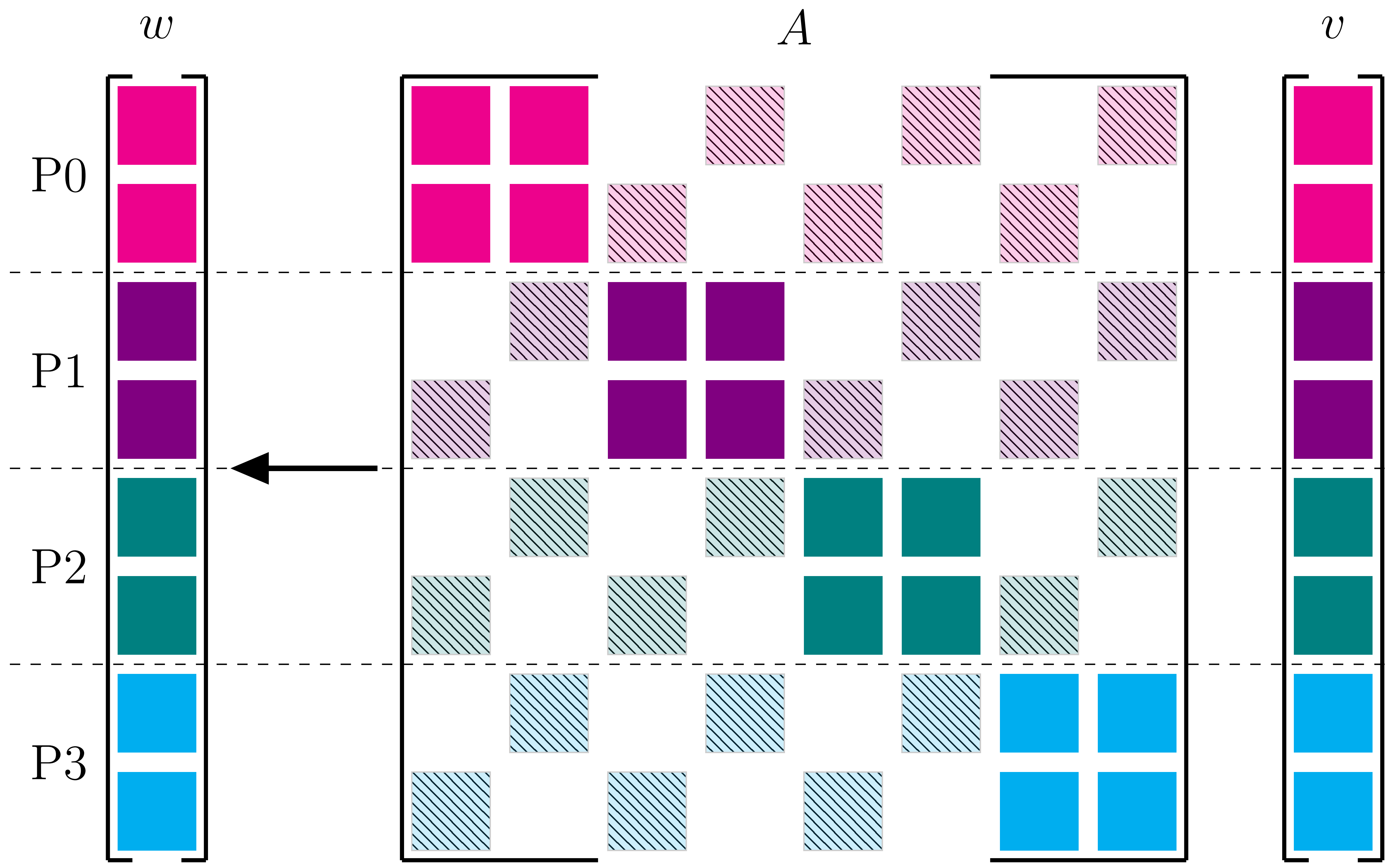
```
Arowptr = [0    2    4    7    9    11]
Acol    = [0    2    1    3    0    2    4    1    3    3    4]
Aval    = [1.1  2.2  3.3  4.4  5.5  6.6  7.7  8.8  9.9  10.0  11.1]
```

```
for(i = 0; i < n; i++){
    sum = y[i];

    for(jj = Arowptr[i]; jj < Arowptr[i+1]; jj++){
        sum += Aval[jj] * x[ Acol[jj] ];
    }

    y[i] = sum;
}
```

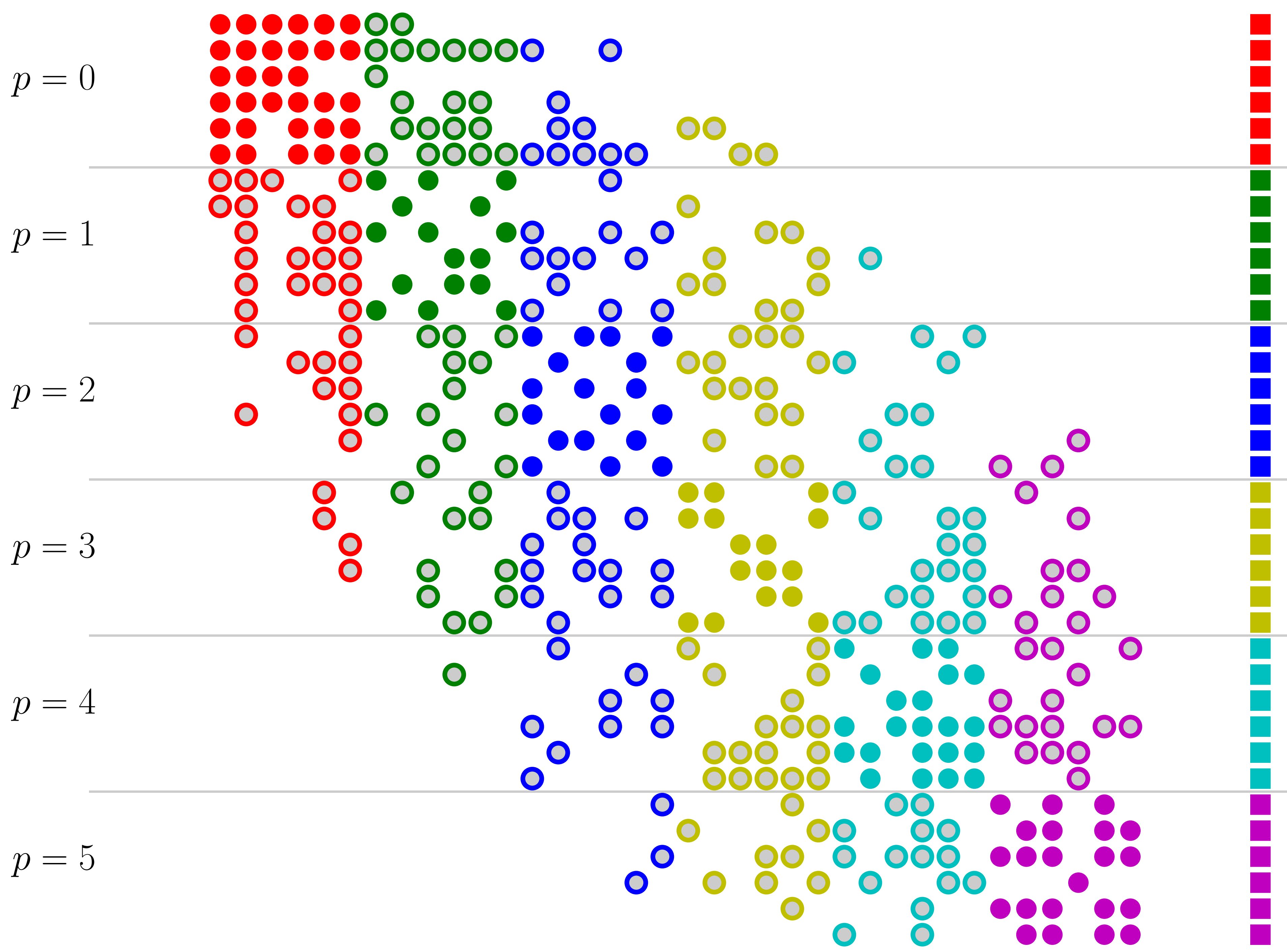
- Complexity: O(nnz)
- This is CSR, other formats are similar (in cost, not memory access)

Key Challenge: Parallel Efficiency of Sparse Operations $w \leftarrow A * v$

- Solid blocks: on-process portion
- Patterned blocks: off-process portion (requires communication of the input vector)

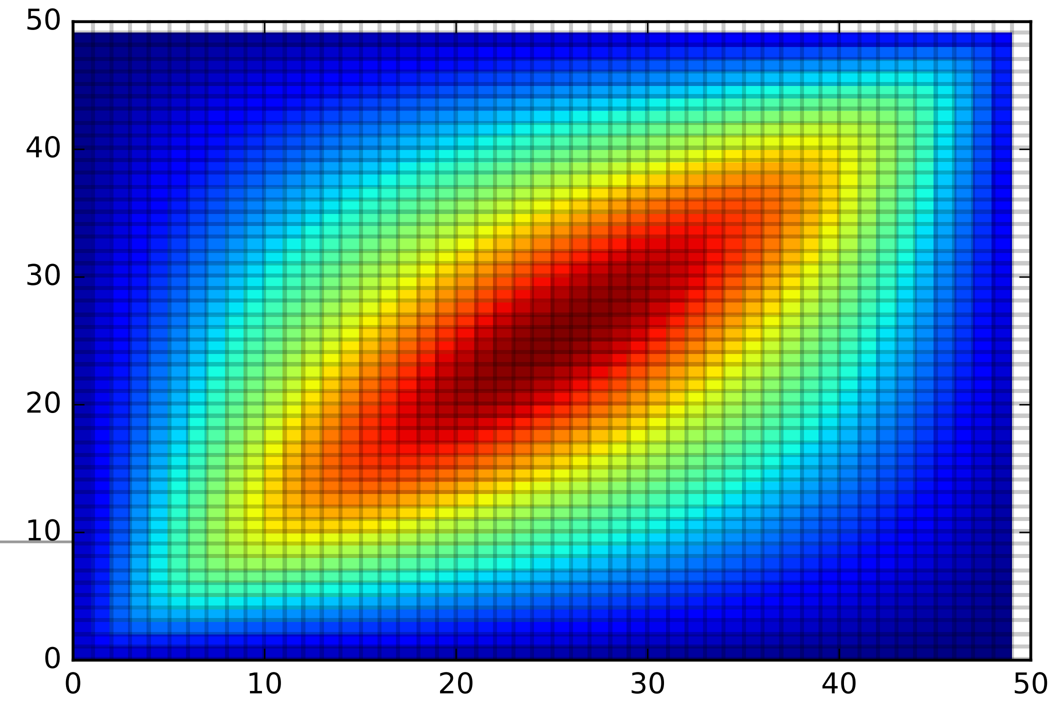


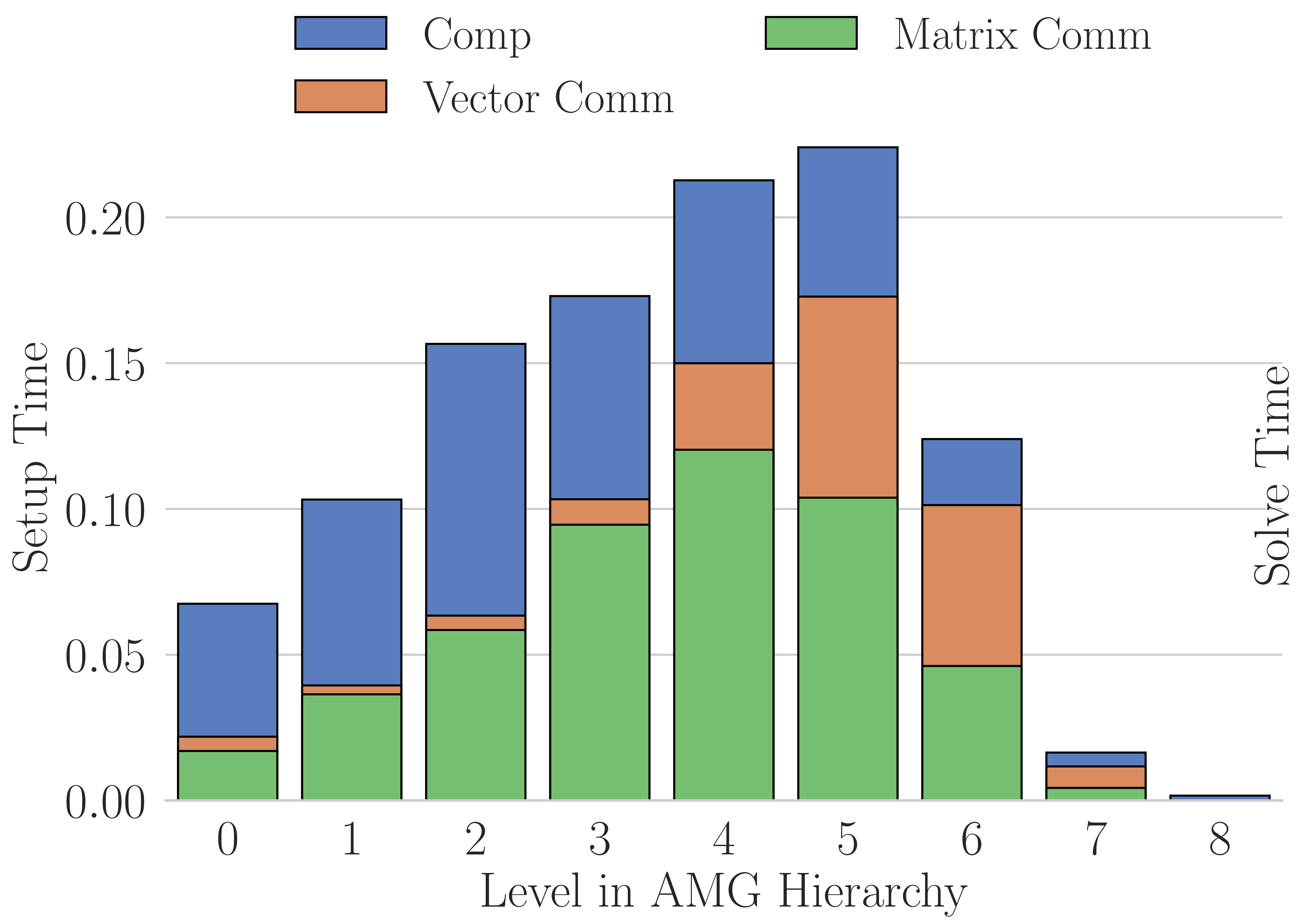
Data layout

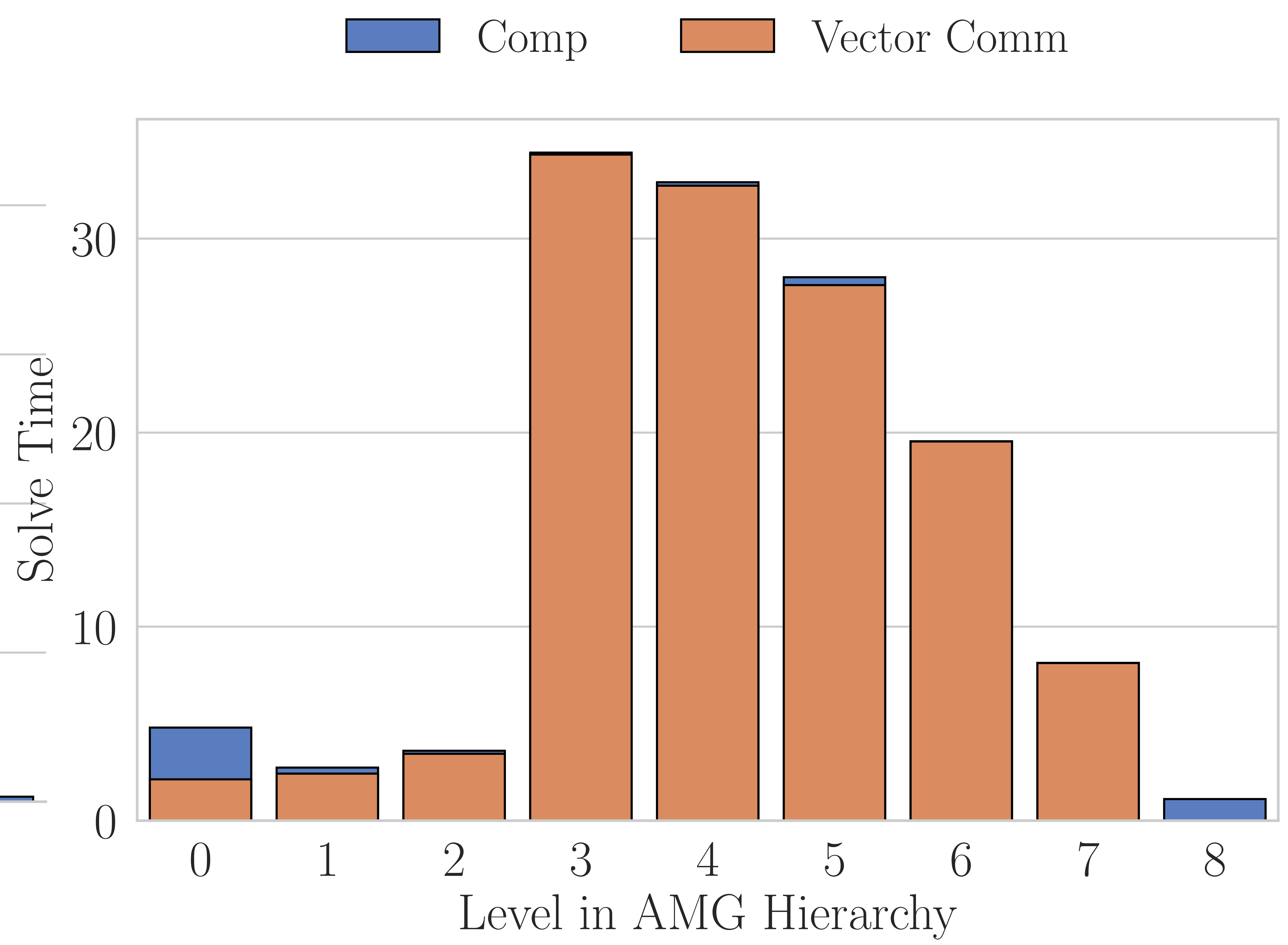


Where data is sent

Blue Waters Case Study: Laplacian

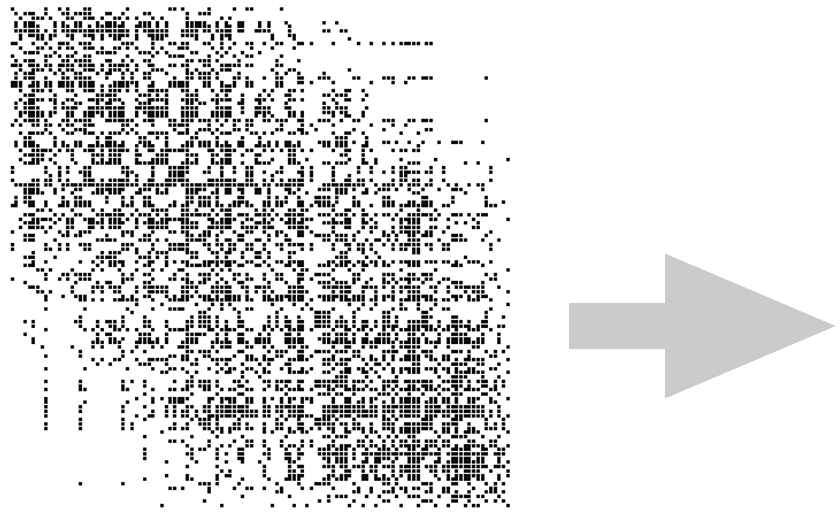


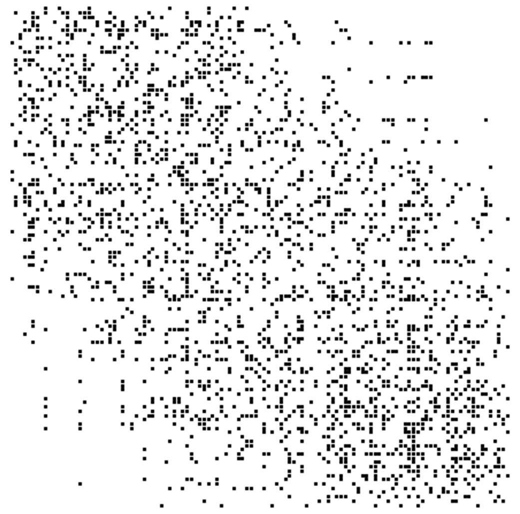




8192 processors, 512 nodes, ~200 rows per processor

How do we address this?





1. Remove data

R. D. Falgout and J. B. Schroder, “Non-Galerkin coarse grids for algebraic multigrid,” SISC, 2014

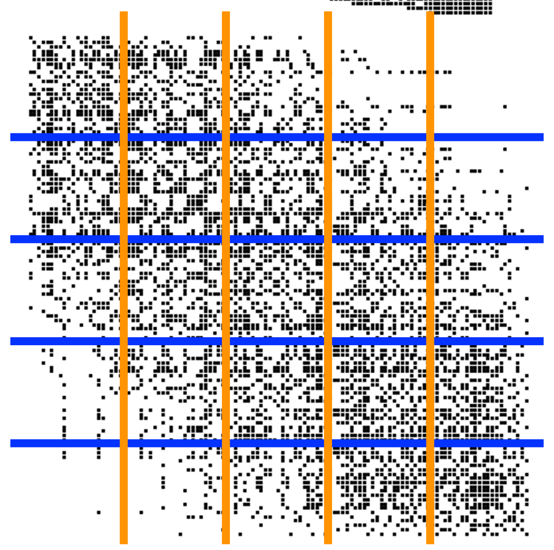
E. Treister and I. Yavneh, “Non-Galerkin multigrid based on sparsified smoothed aggregation,” SISC, 2015

A. Bienz, R. D. Falgout, W. Gropp, L. N. Olson, and J. B. Schroder, “Reducing parallel communication in algebraic multigrid through sparsification,” SISC, 2016

2. Data layout

Graph reorderings
Redistribution

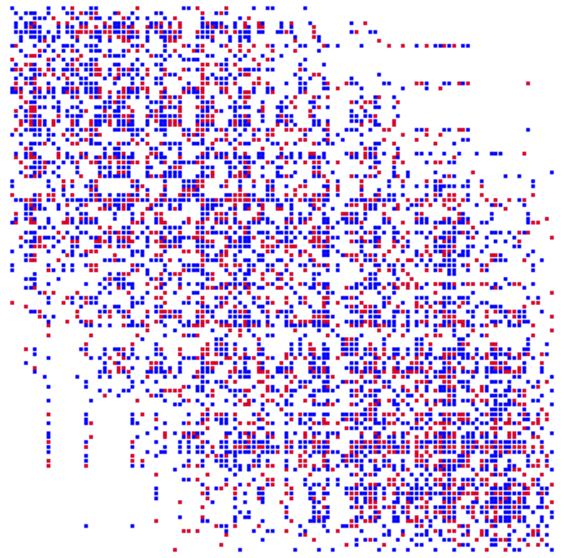
Gahvari, Gropp, Jordan, Schulz, Meier Yang, Systematic Reduction of Data Movement in Algebraic Multigrid Solvers, 2014



3. Data partition

Mets, Scotch, Zoltan

Bowman, Wolf, “A Nested Dissection Partitioning Method forParallel Sparse Matrix-Vector Multiplication”

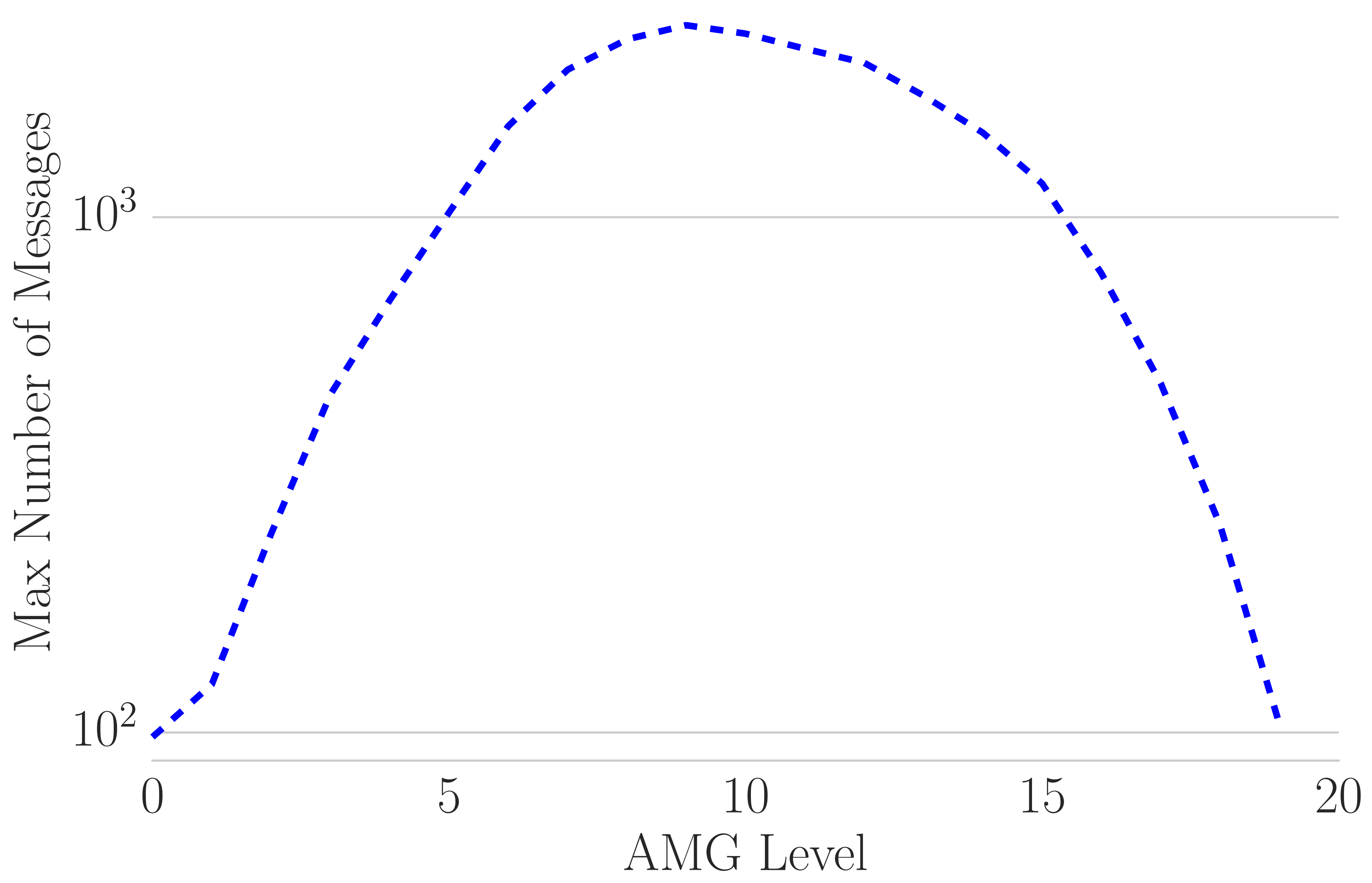


4. Data traffic

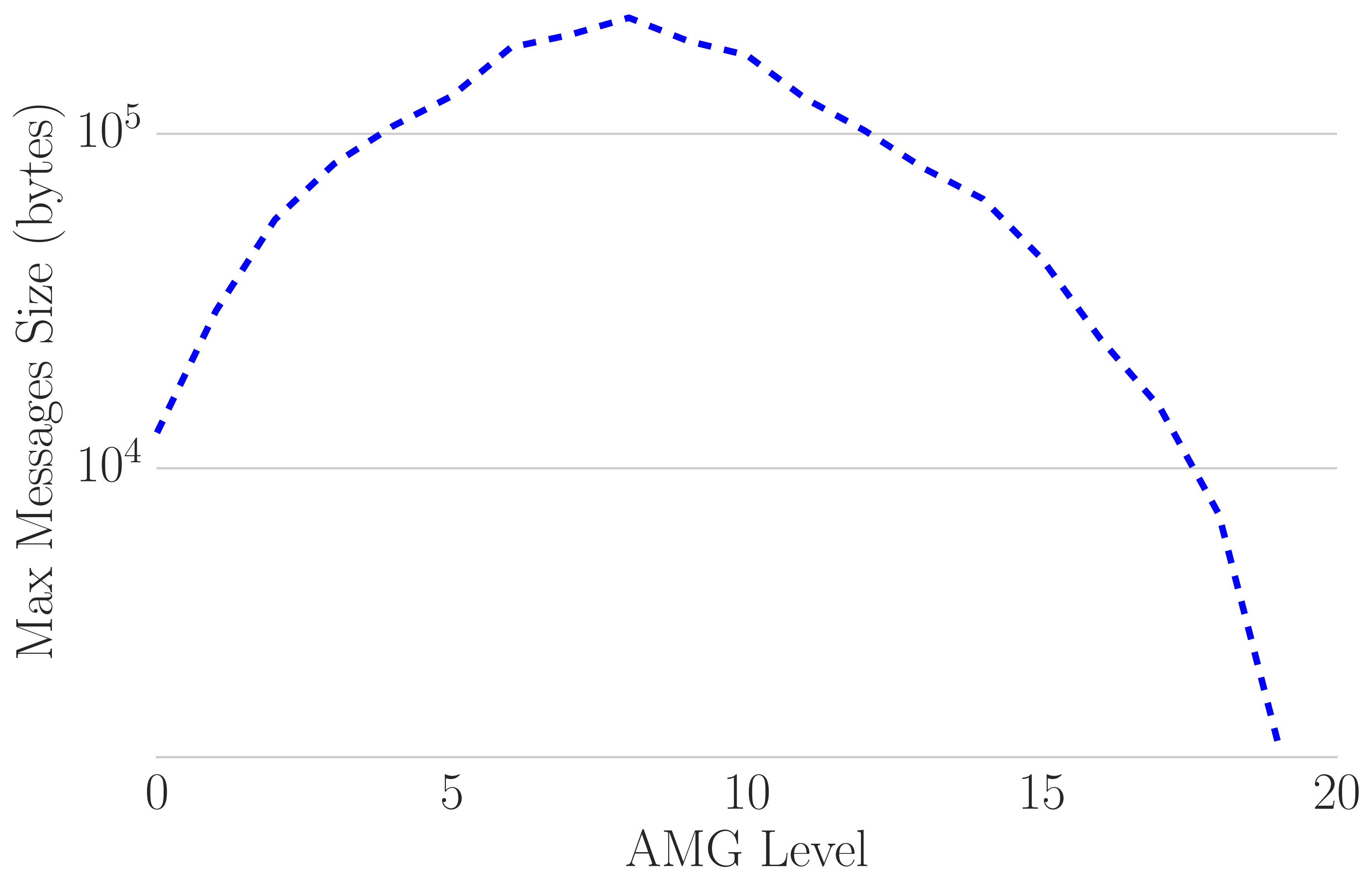
Reorganize communication

- Recognize limits of communication
- Recognize opportunities in the machine hierarchy

Observation 1: high volume/number of messages



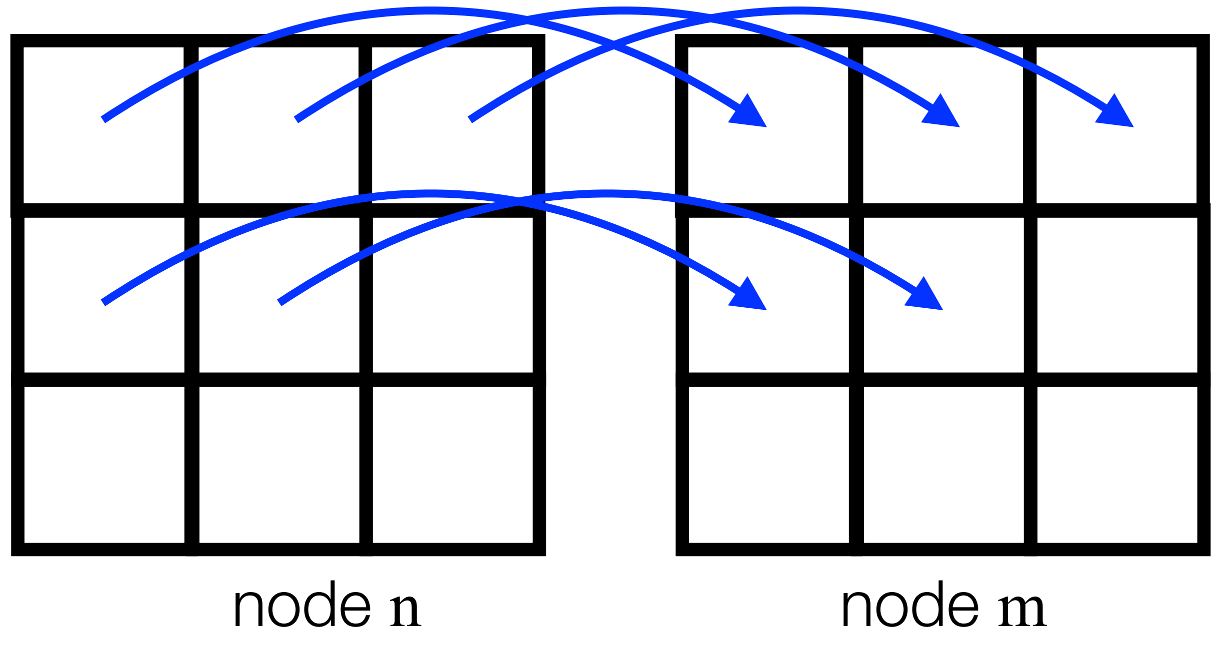
Maximum number of messages

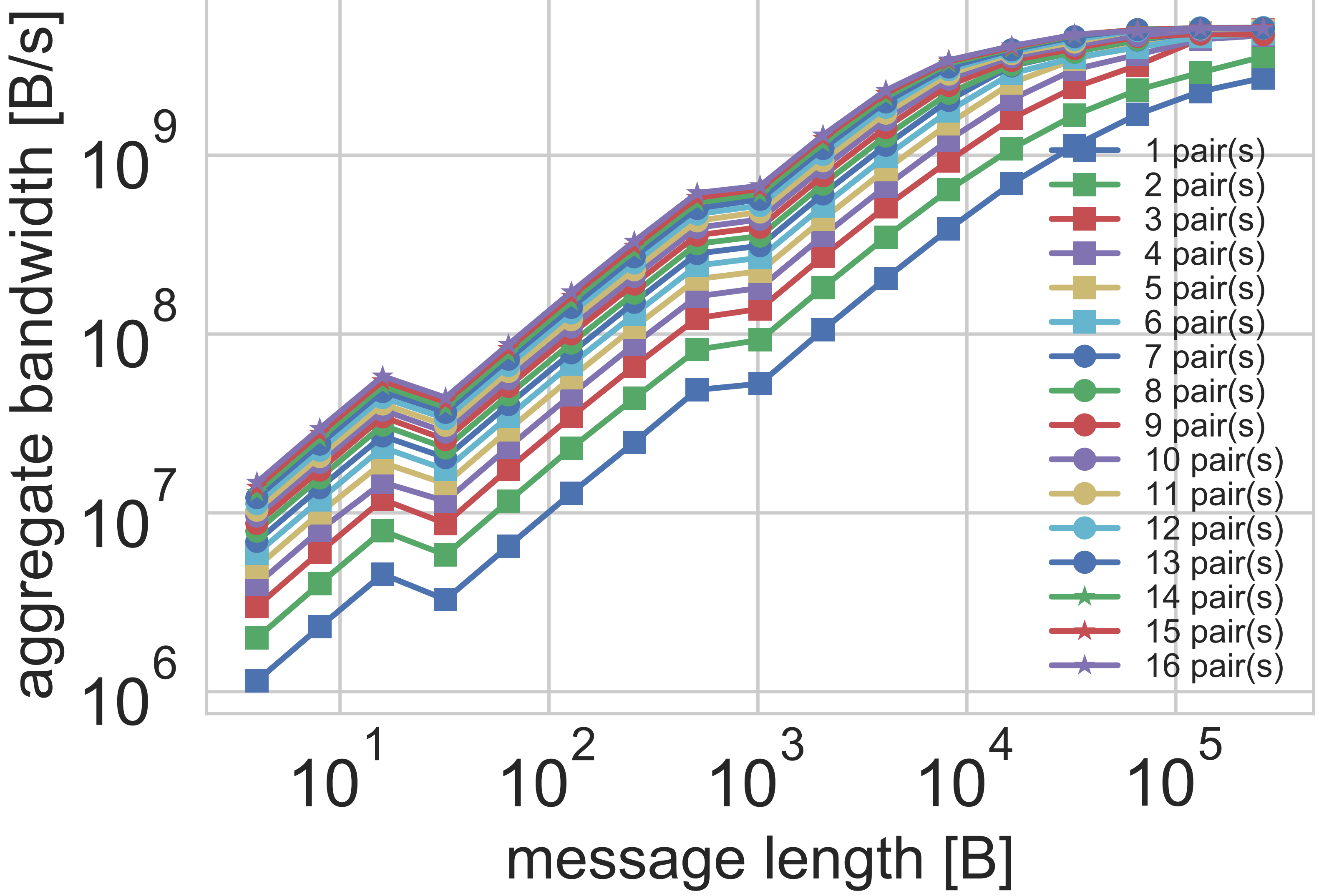


Maximum size of messages

np = 16384

Observation 2: diminishing returns with higher communicating cores





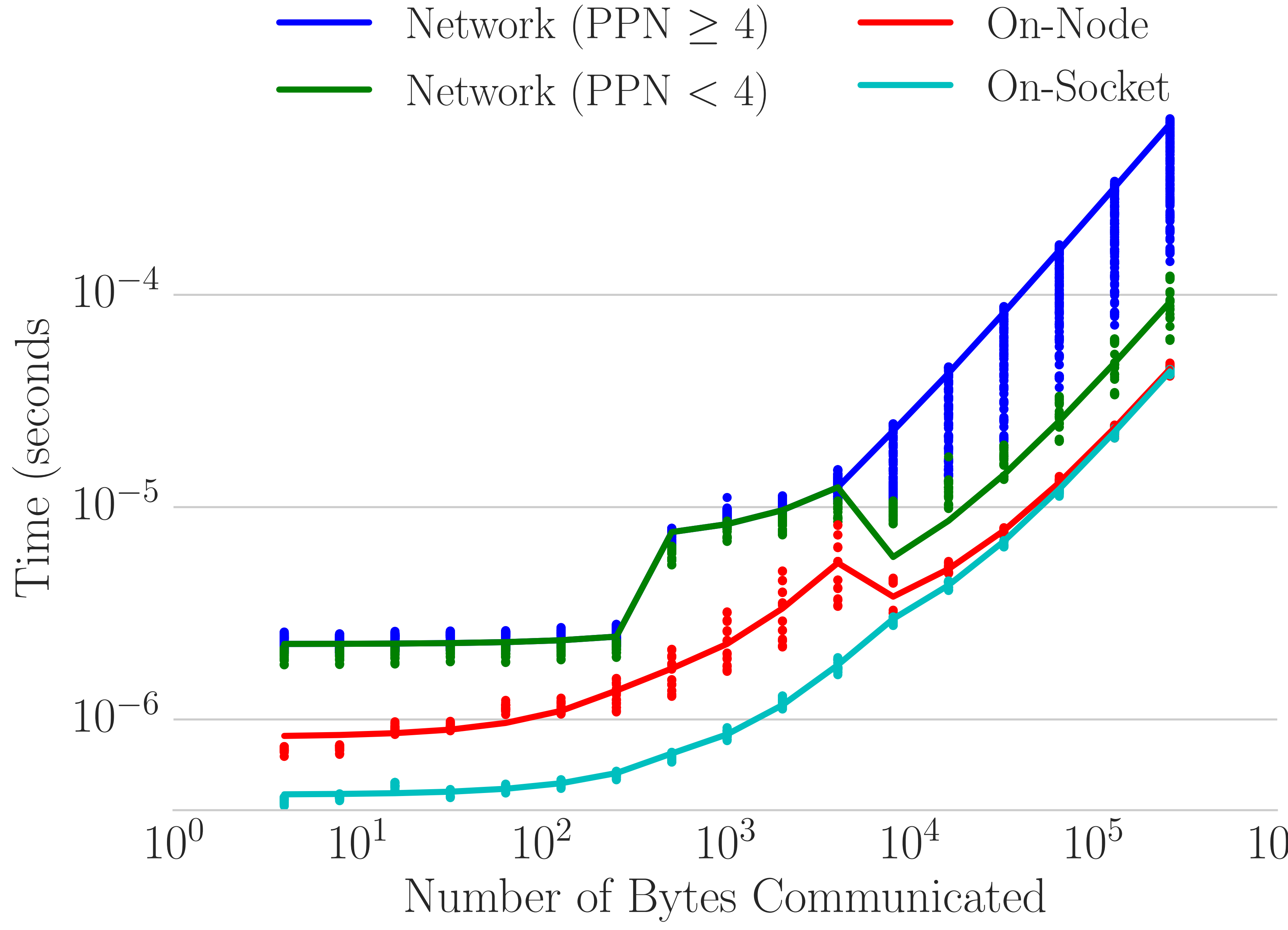
latency

message size

$$T = \alpha + \frac{\mathrm{ppn} \cdot s}{\min\left(R_N,\, \mathrm{ppn} \cdot R_B\right)}$$

Node injection bandwidth

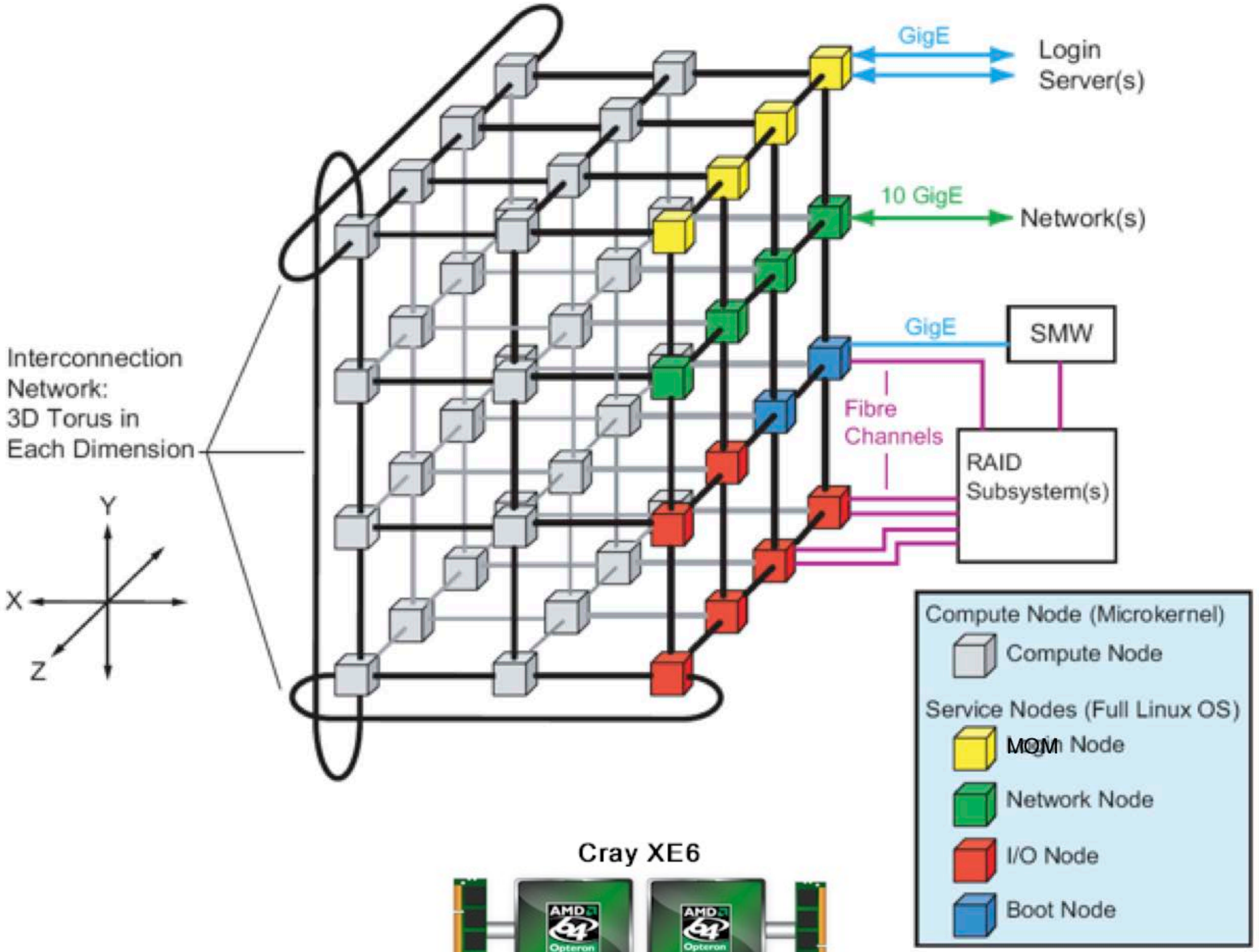
Bandwidth between two processes

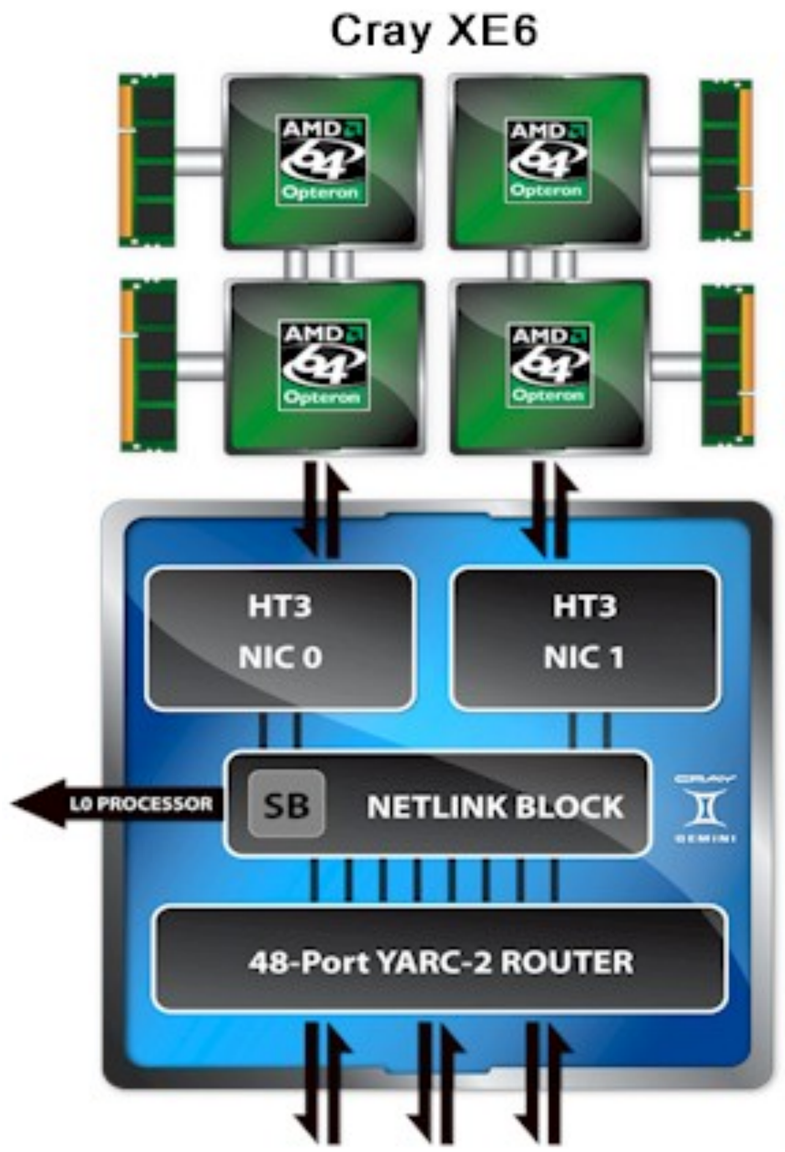


Modeling MPI Communication Performance on SMP Nodes: Is it Time to Retire the Ping Pong Test,
Gropp, Olson, Samfass, EuroMPI 2016.

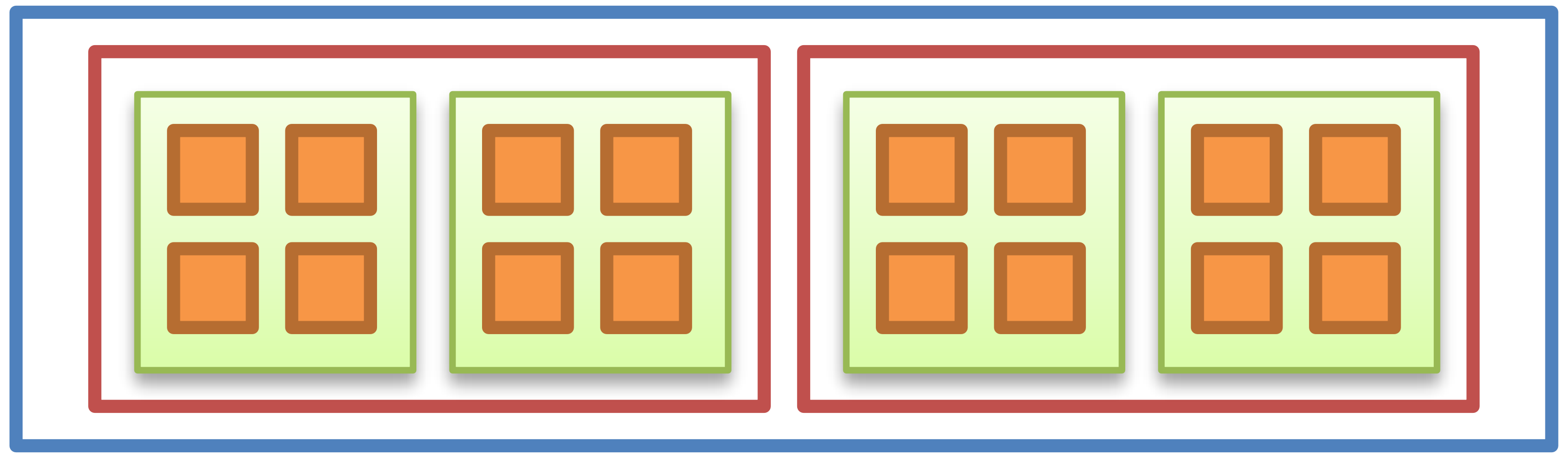
Observation 3: node locality

- Concurrency increasing
- Hierarchy of compute nodes (sockets , nodes, etc)
- Range of compute units (power 9, GPU, etc)
- **Blue Waters is providing a roadmap**

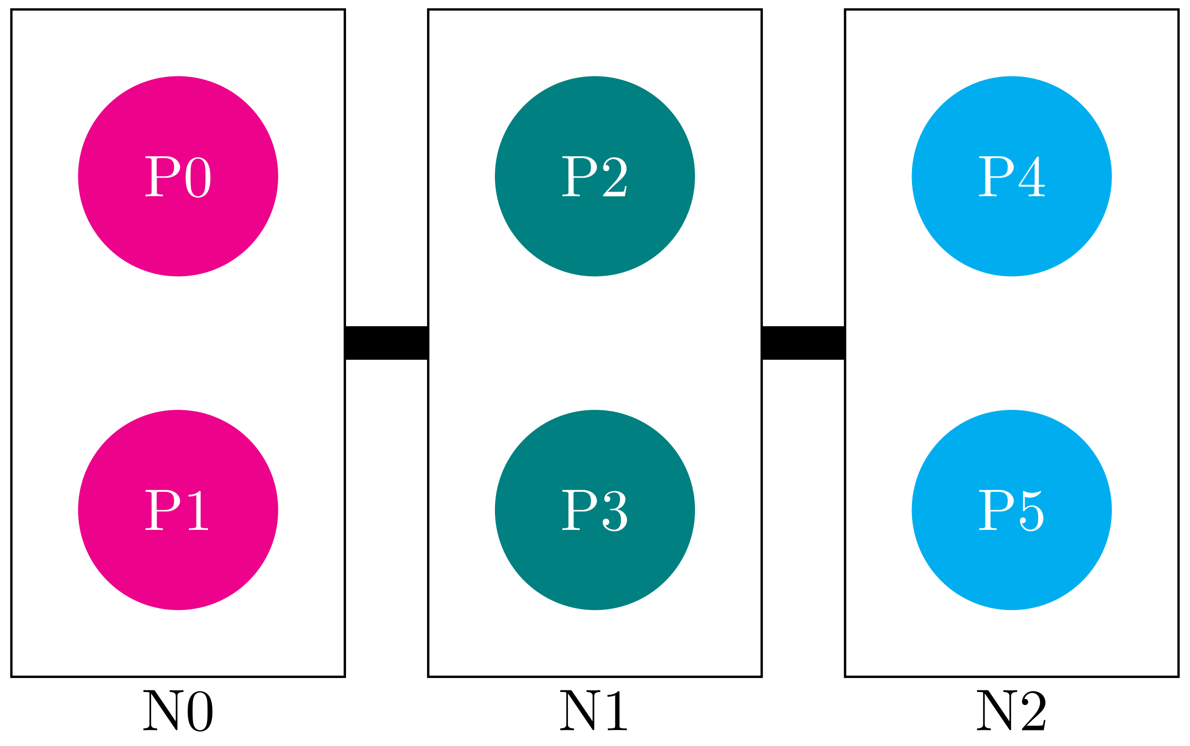




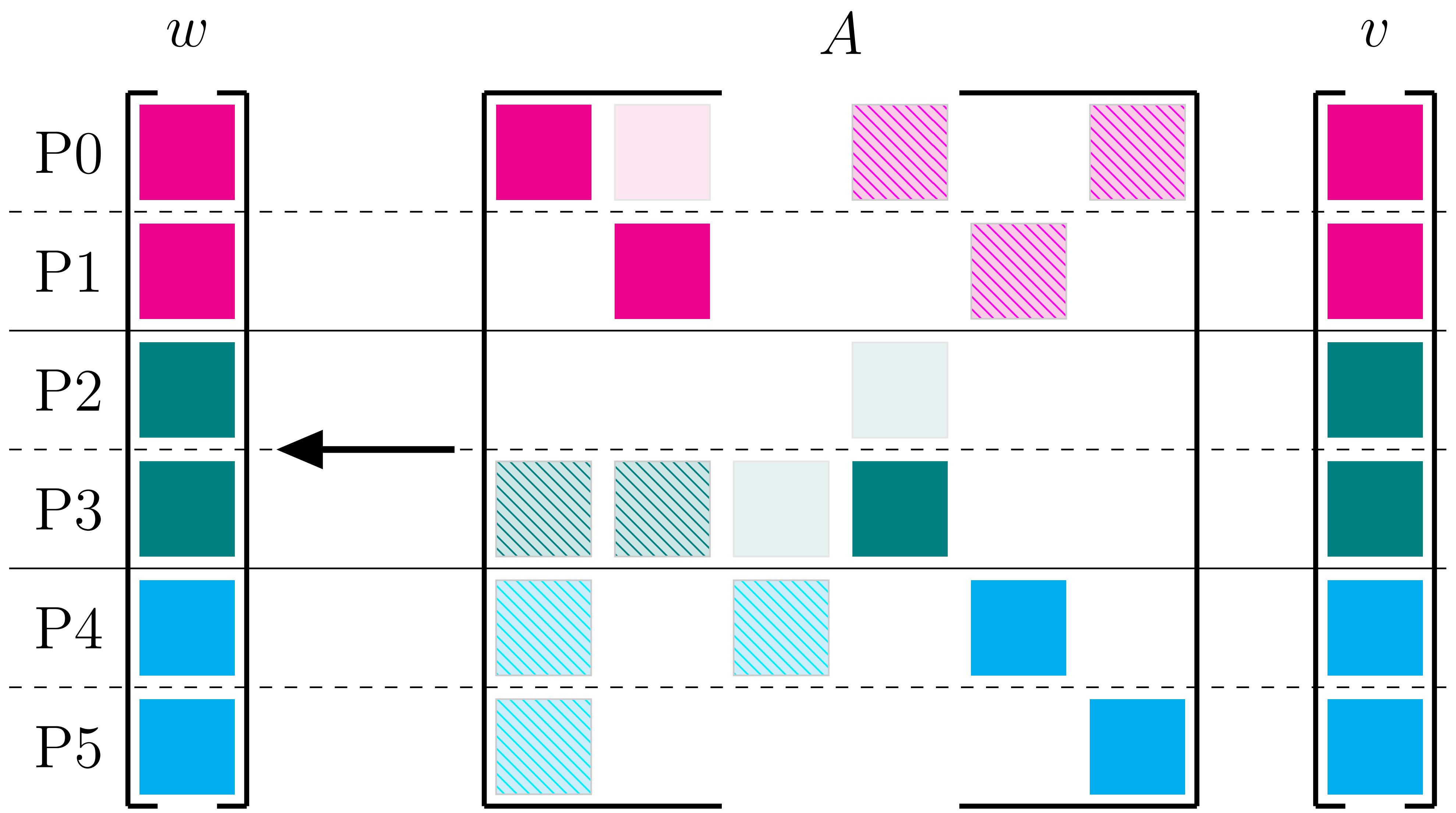
node
socket
die
cores



A **node** level approach to a SpMV



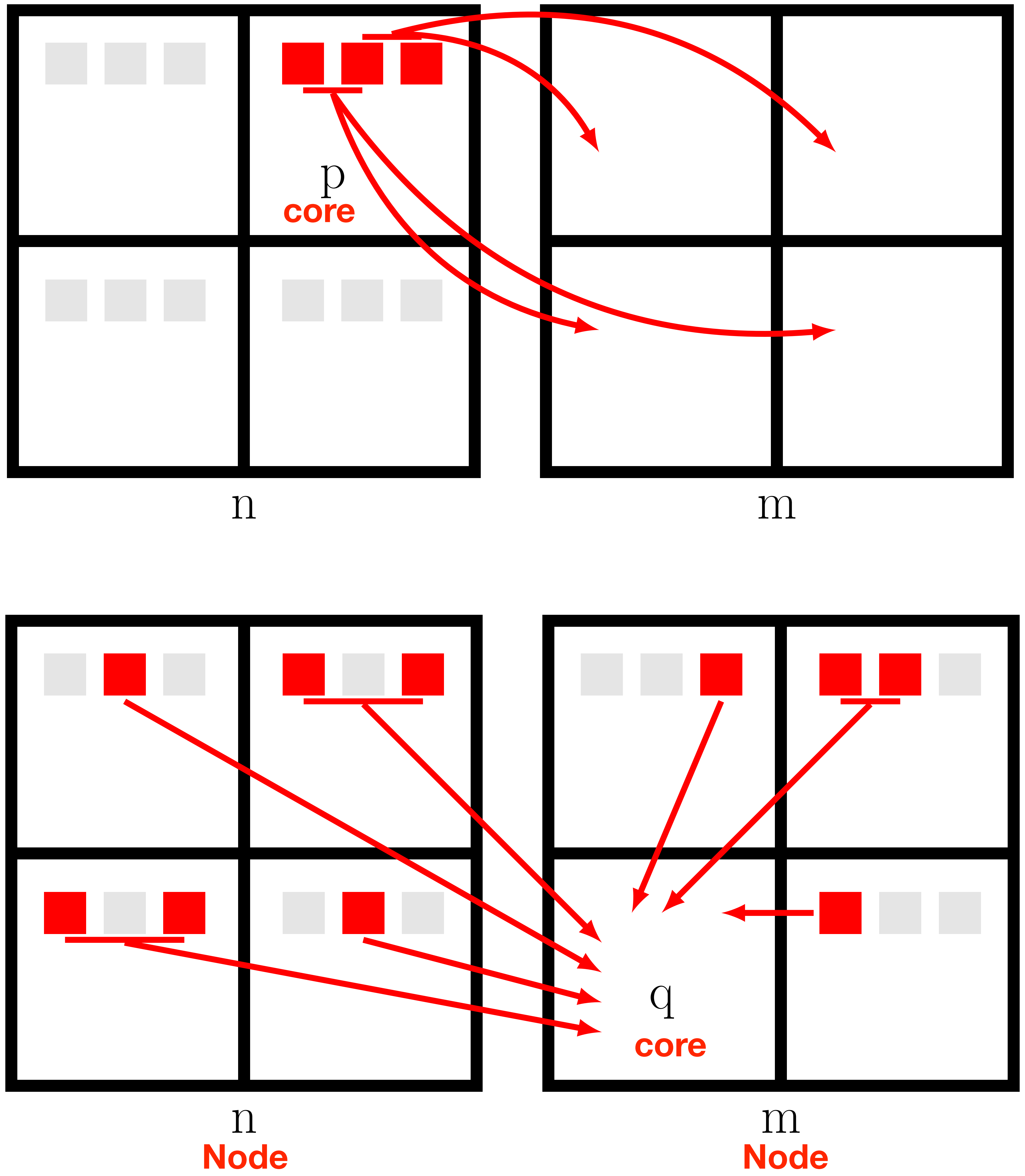
Six processes distributed across three nodes



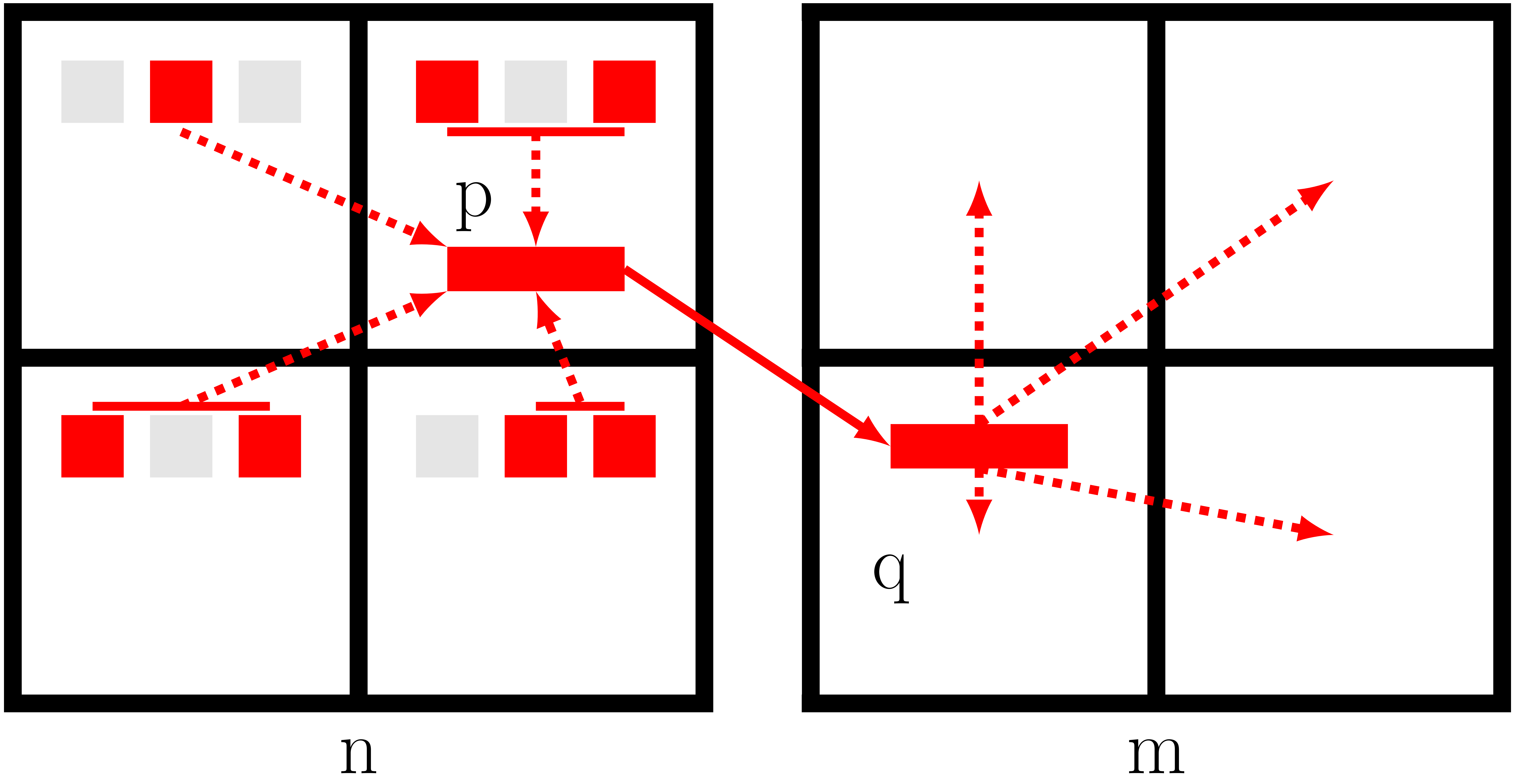
Linear system distributed across the processes

Standard Communication

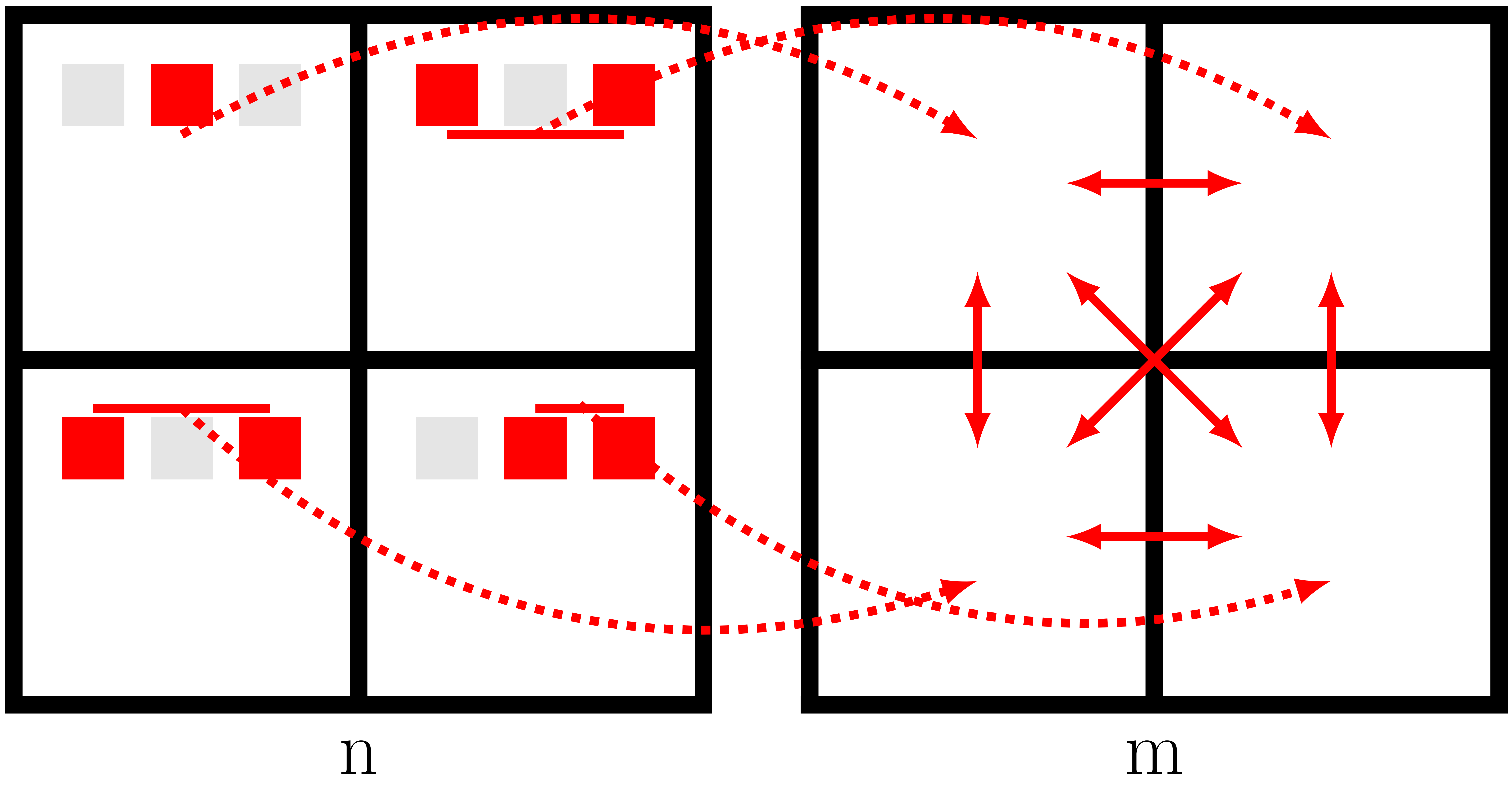
- Duplicate data
- Many messages



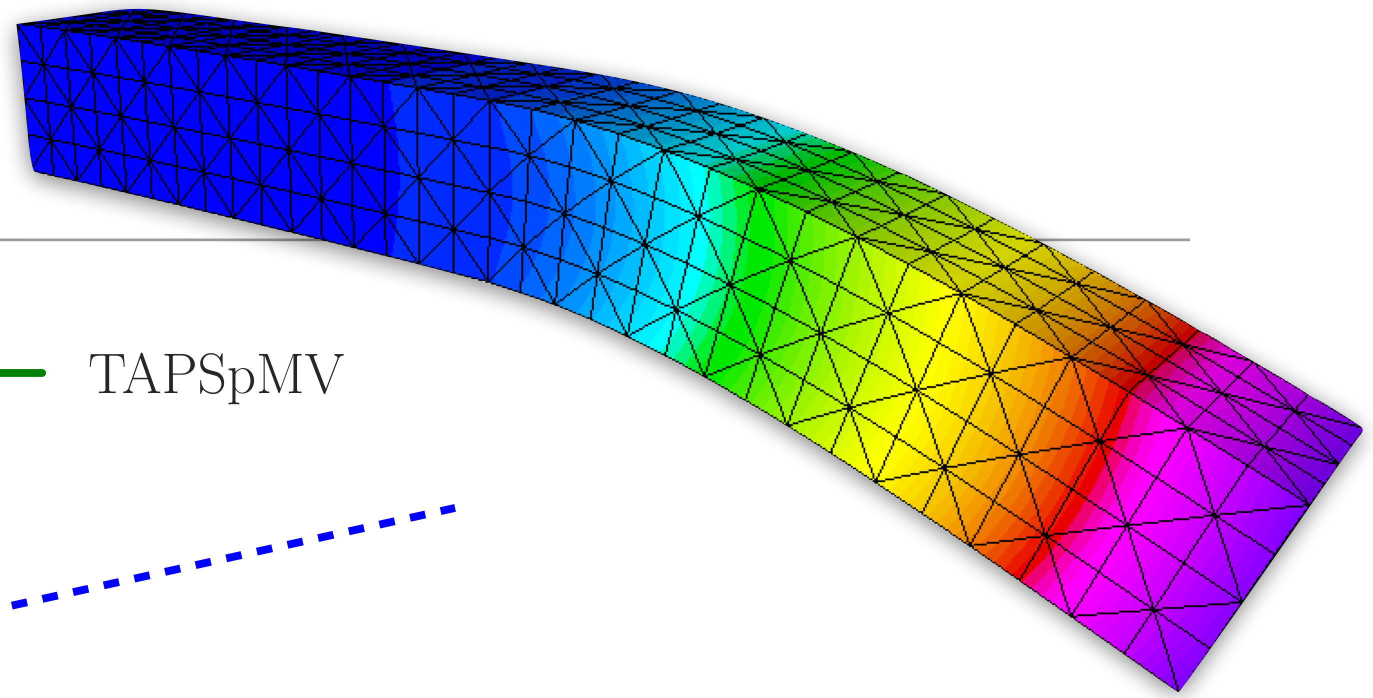
3-step Communication

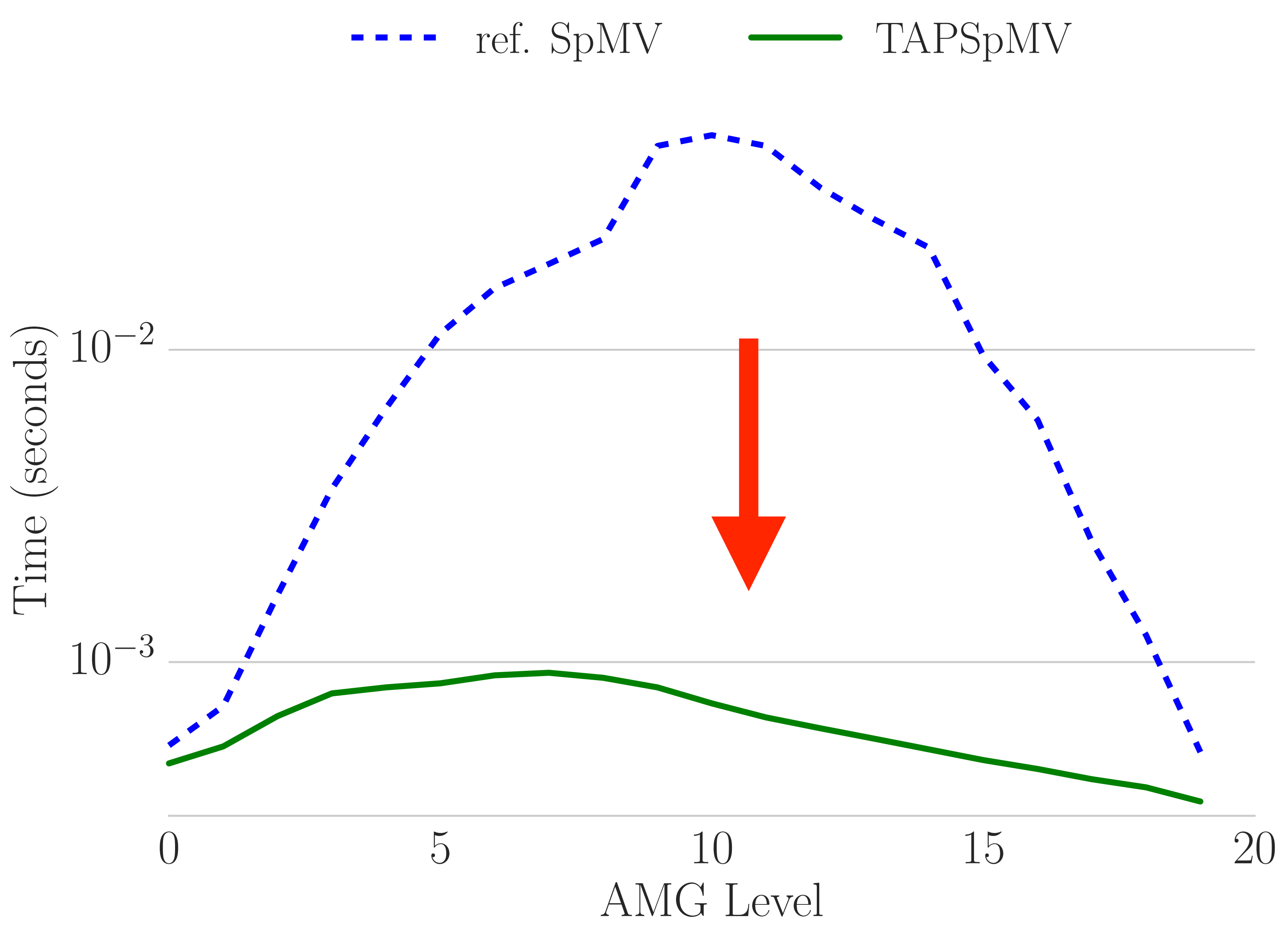


2-step Communication;

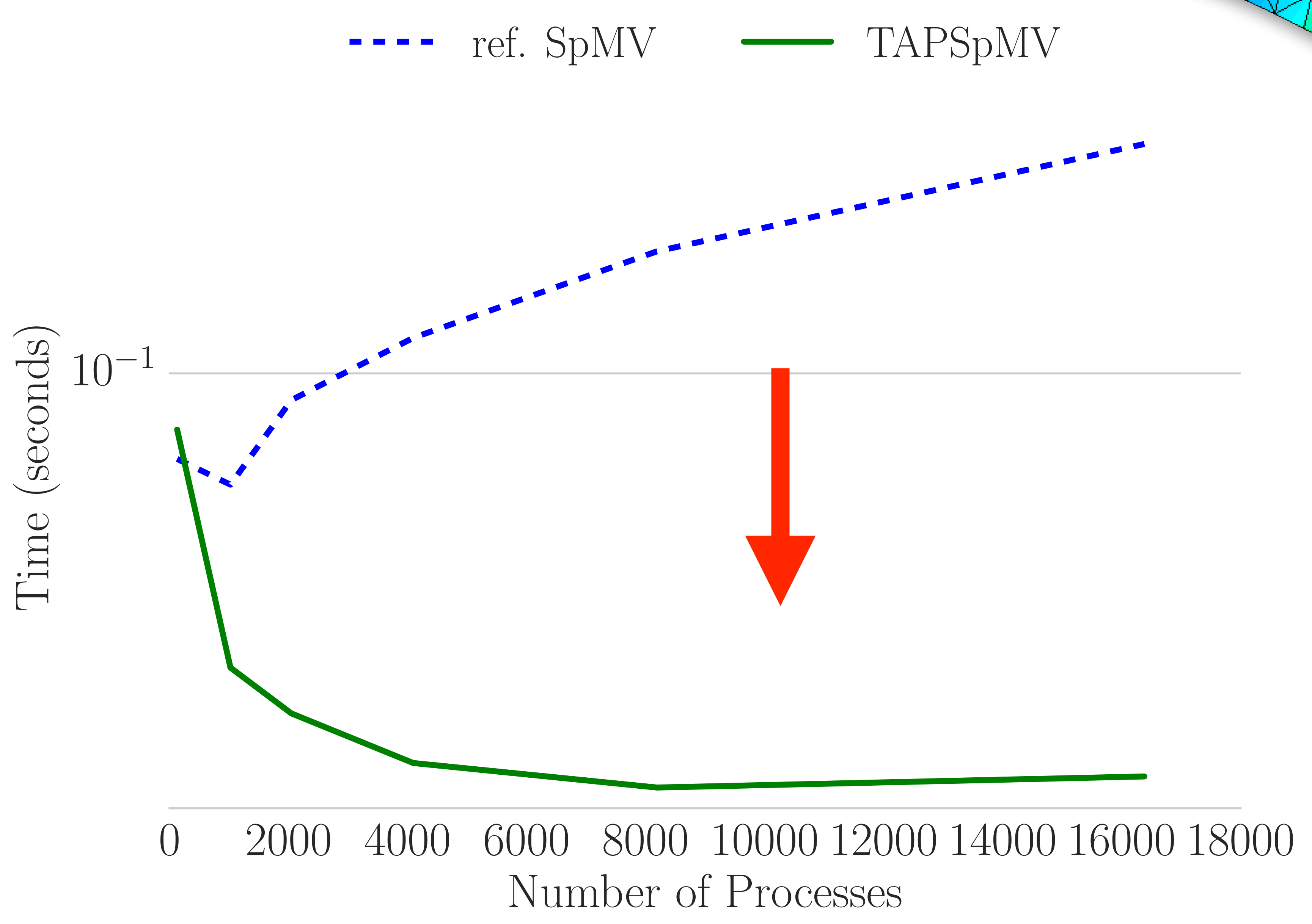


Case Study: 3 step communication, Linear Elasticity





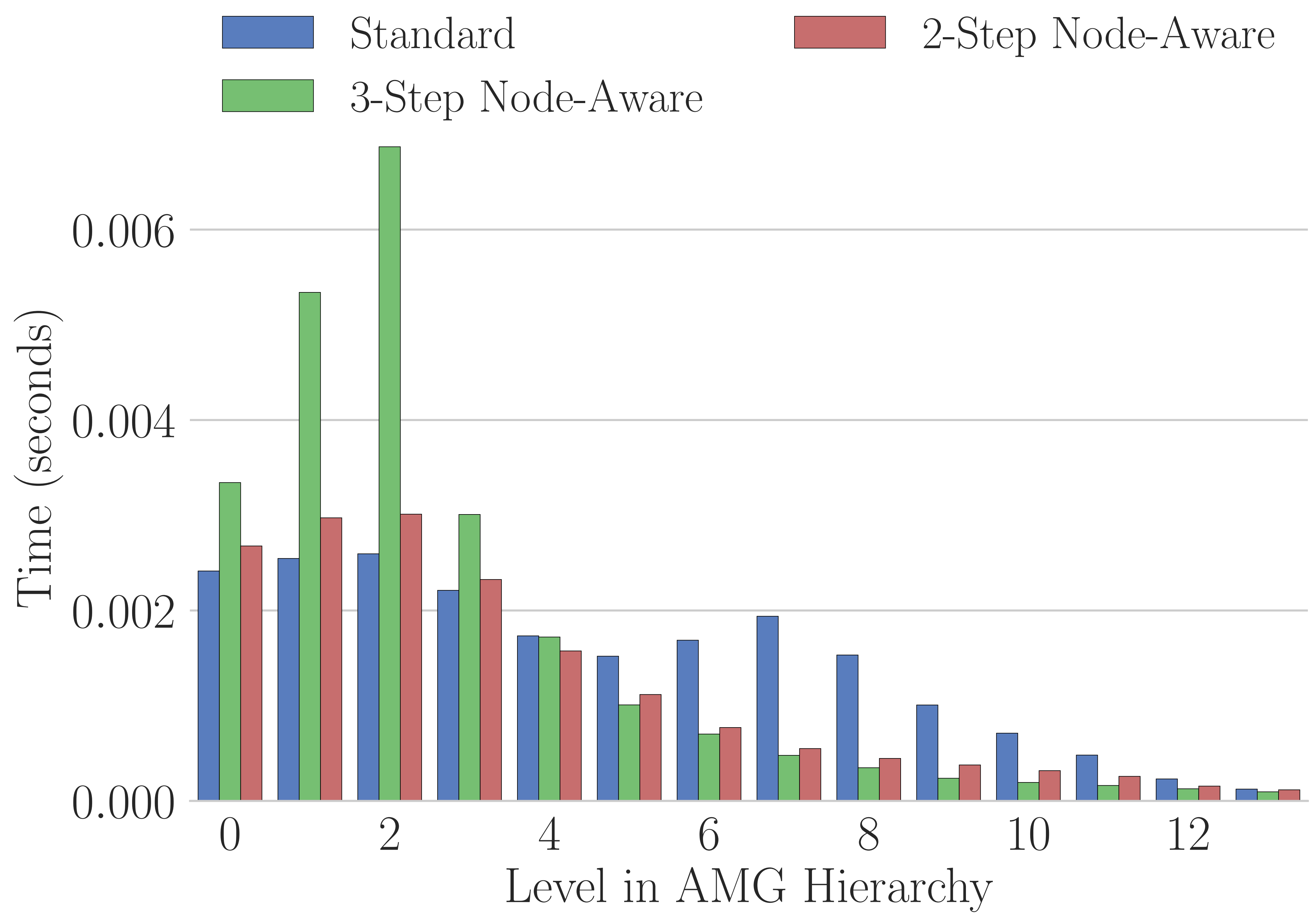
Total Time



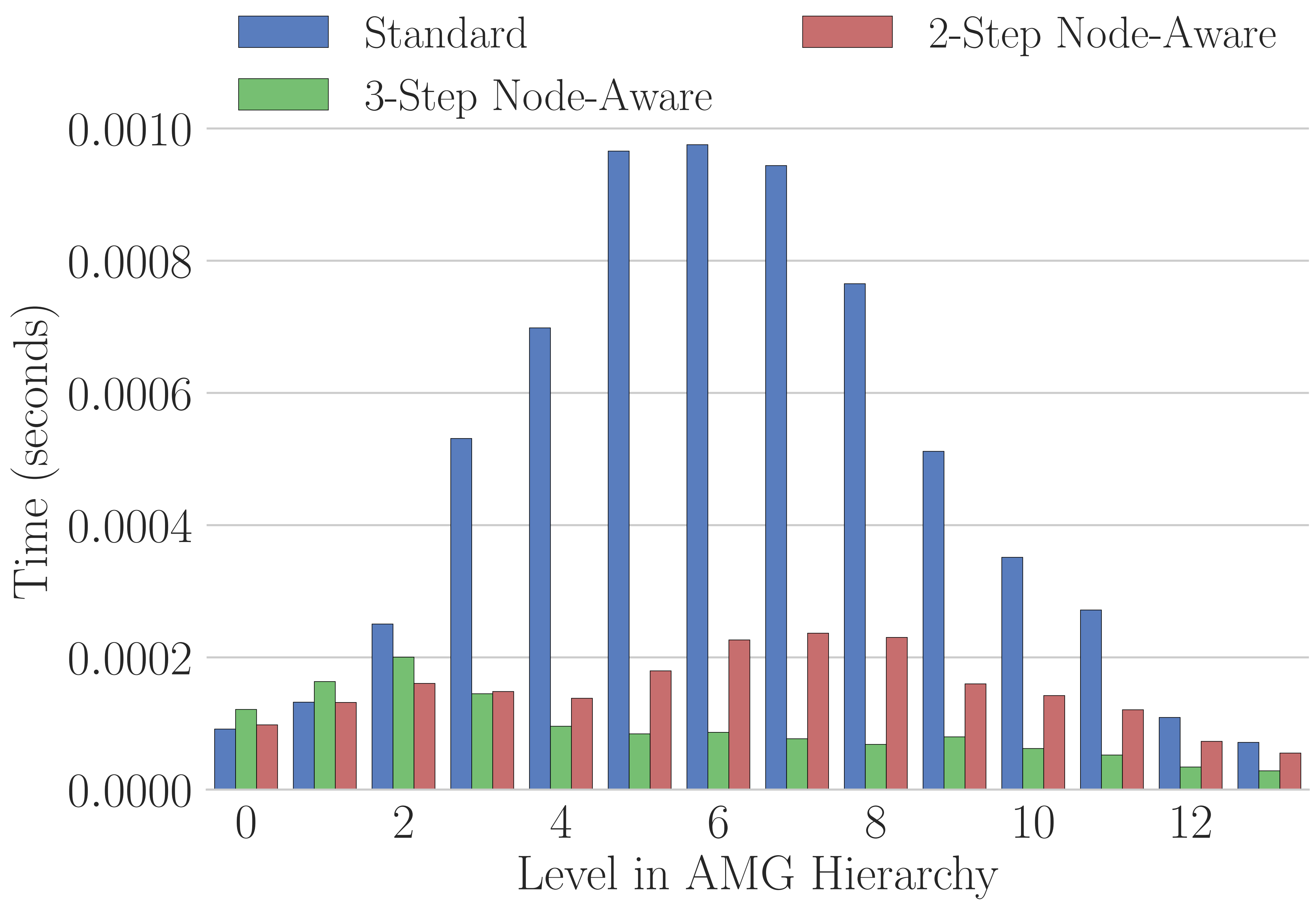
Strong Scaling

Node aware sparse matrix-vector multiplication,
Bienz, Gropp, Olson, `JPDC`, 2019.

Impact on SpMM and SpMV



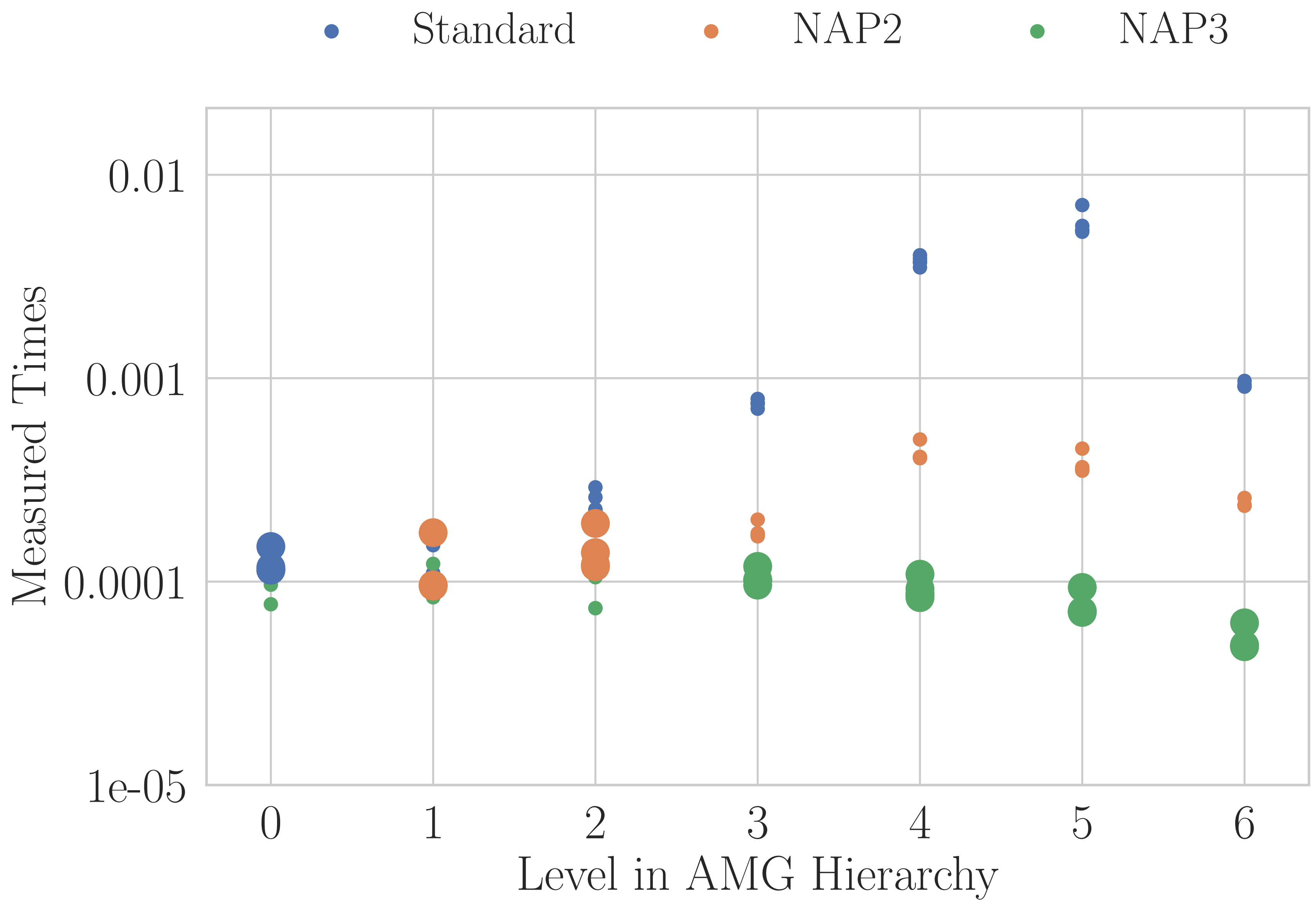
Row-Wise SpGEMM: AP



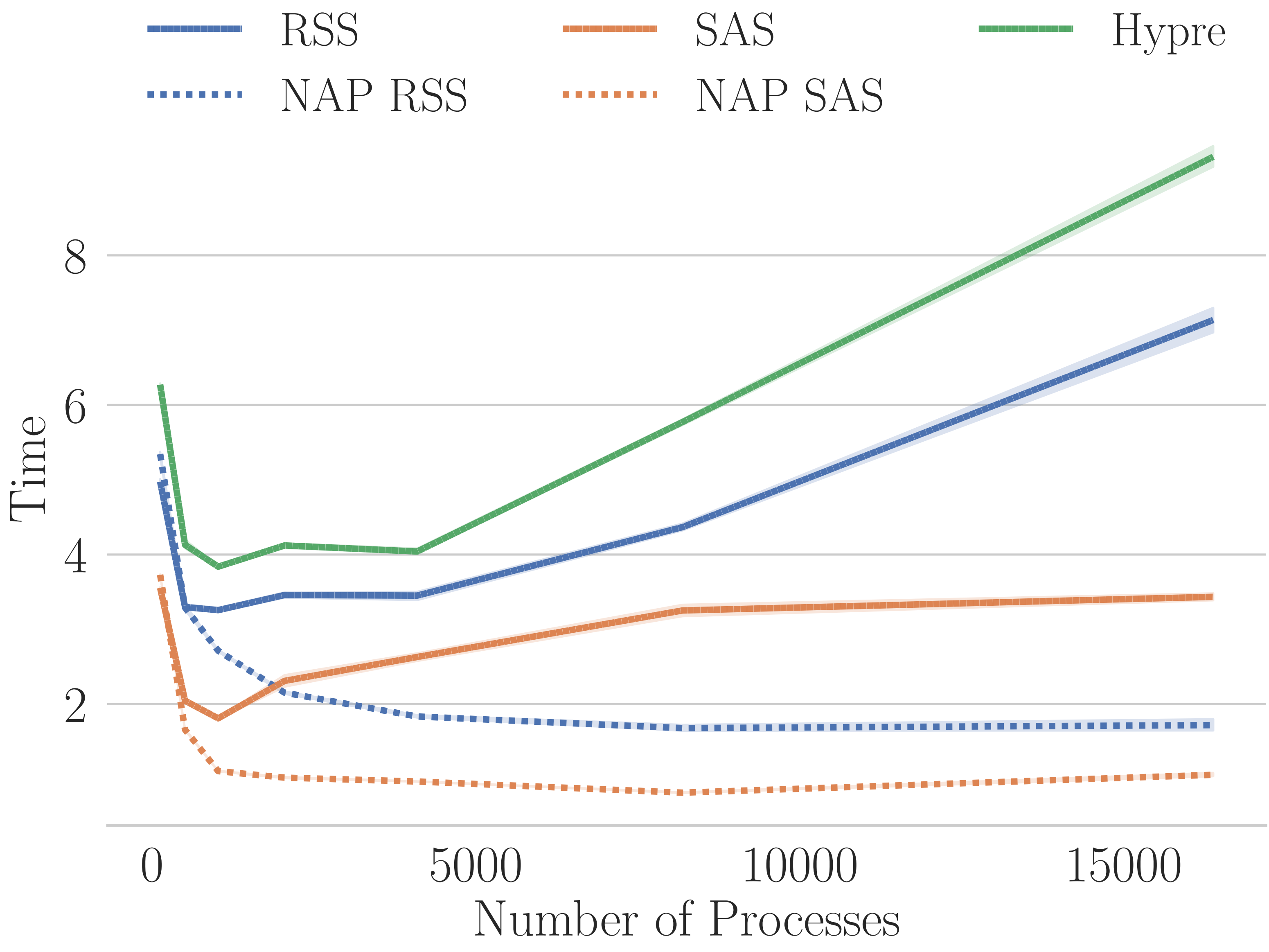
SpMV: Ax

Performance Results: Strong Scaling

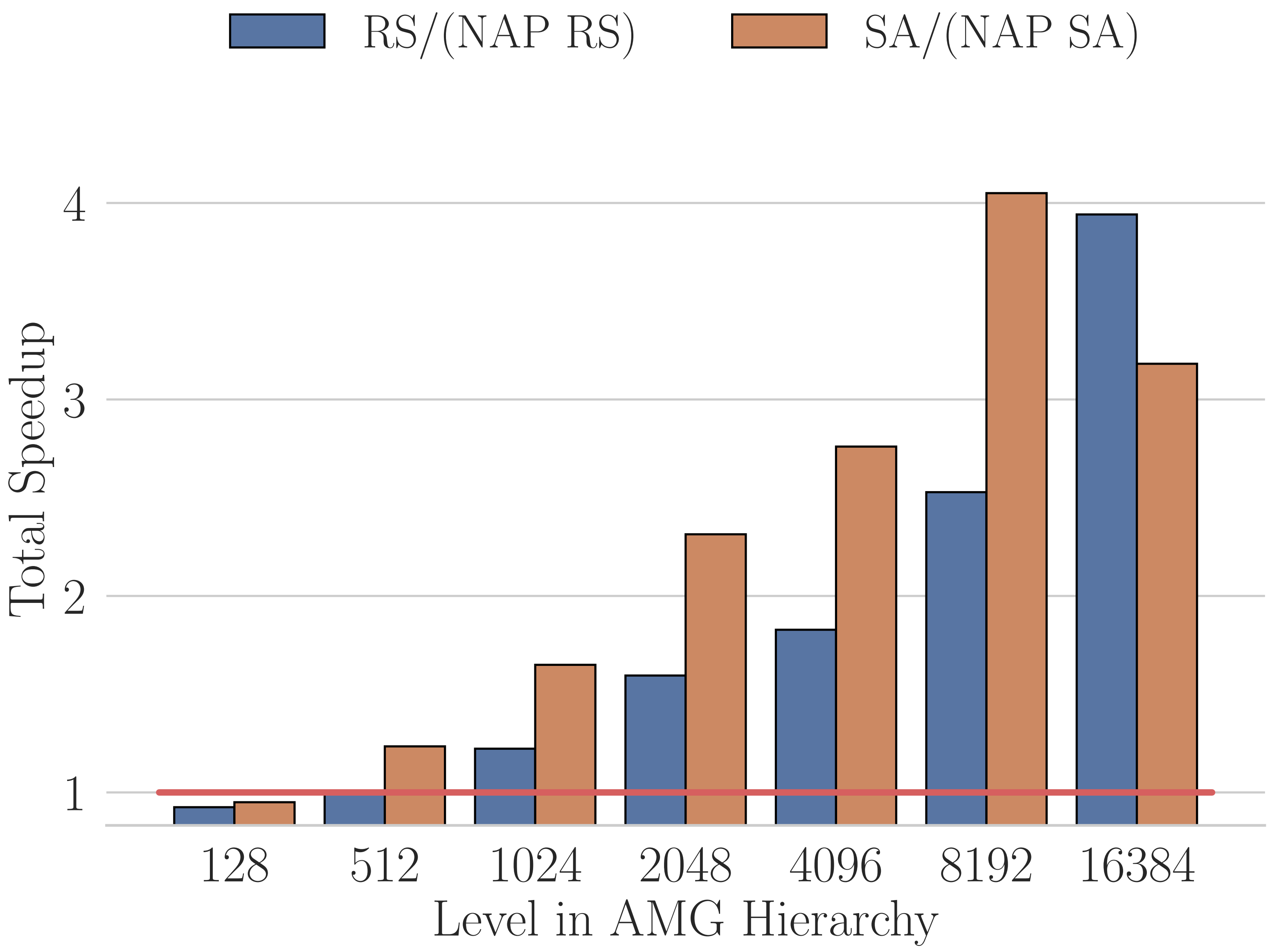
- MFEM Grad-Div problem; Blue Waters; 356,352 rows; 14,145,024 nnz
- New algorithms + performance models = extended scaling



Automatic selection



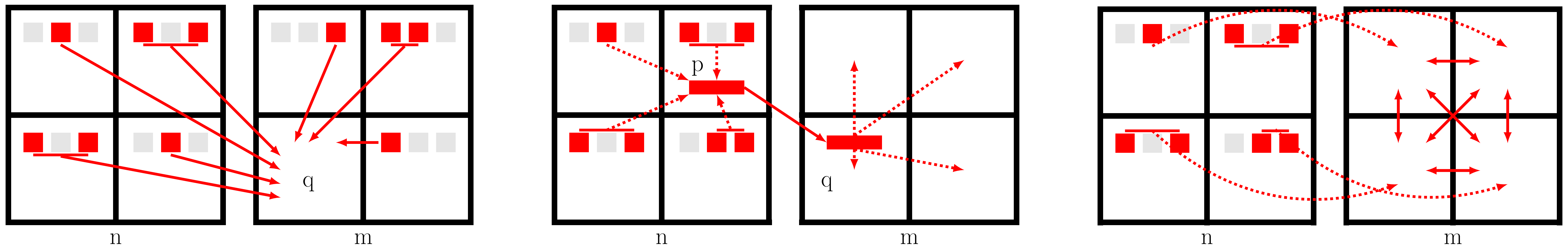
Time



Speedup

Reducing Communication in Algebraic Multigrid with Multi-step Node Aware Communication
Bienz, Gropp, Olson https://arxiv.org/abs/1904.05838

Node-Aware MPI Library




```
void MPI_NAPinit(const int n_sends, const int* send_procs, const int* send_indptr,
        const int* send_indices, const int n_recvs, const int* recv_procs,
        const int* recv_indptr, const int* recv_indices, const MPI_Comm mpi_comm,
        NAPComm** nap_comm_ptr)
```

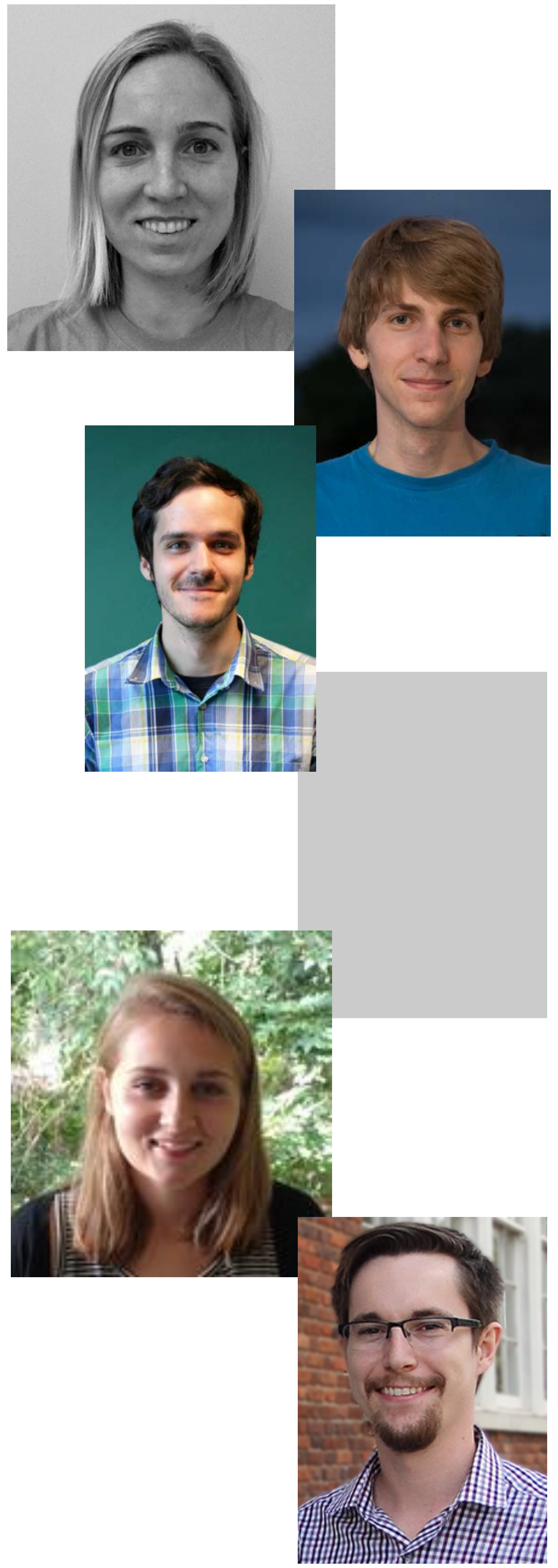
```
MPI_INAPsend(const T* send_data, const NAPComm* nap_comm, const int tag,
        const MPI_Comm mpi_comm, NapData* nap_data)
```

```
MPI_INAPrecv(T* recv_data, const NAPComm* nap_comm, const int tag,
        const MPI_Comm mpi_comm, NAPData* nap_data)
```

```
MPI_NAPwait(const NAPComm* nap_comm, NAPData* nap_data)
```

```
void MPI_NAPdestroy(NAPComm** nap_comm_ptr)
```

Impact of Blue Waters



- Amanda Bienz, PhD (2018)
  reducing communication
- Andrew Reisner, PhD (2019)
  scalable structured solvers
- Lukas Spies, PhD (current)
  communication and multi-GPU systems
- Philipp Samfass, MS (2016)
  performance modeling
- Shelby Lockhart, PhD (current)
  high performance Krylov methods
- John Calhoun, PhD (2017, **BW Fellow!**)
  fault resilience in HPC

- Blue Waters has been an invaluable **recruitment** tool, both **students and faculty**
- Blue Waters has directly contributed to the visibility and **quality** of the research
- Blue Waters has been a gateway to developing **new codes**, testing new **methods**, and anticipating new and upcoming **architectures**.

Where to find more

- **`github.com/cedar-framework/cedar`**
  - *Scaling Structured Multigrid to 500K+ Cores through Coarse-Grid Redistribution*
    Reisner, Olson, Moulton, SISC, 2018
- **`github.com/raptor-library/raptor`**
  - *Node-Aware Sparse Matrix-Vector Communication*
    Bienz, Gropp, Olson, JPDC, 2019
  - *Improving Performance Models for Irregular Point-to-Point Communication*
    Bienz, Gropp, Olson, in review EuroMPI, 2018.
  - *Reducing Communication in Algebraic Multigrid with Multi-step Node Aware Communication,* `https://arxiv.org/abs/1904.05838`
- **`github.com/bienz2/Node_Aware_MPI`**

Cedar
Parallel structured multigrid solvers

RAPtor: parallel algebraic multigrid

```
MPI_INAPsend(const T* send_data,
             const MPI_Comm mpi_comm,
```

This material is based in part upon work supported by the Department of Energy, National Nuclear Security Administration, under Award Number DE-NA0002374.

This research is part of the Blue Waters sustained petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications.