Clustering Parallel Applications to Enhance Message Logging Protocols

Esteban Meneses

Jaguar is the top 2 supercomputer in the world with 224,162 cores...
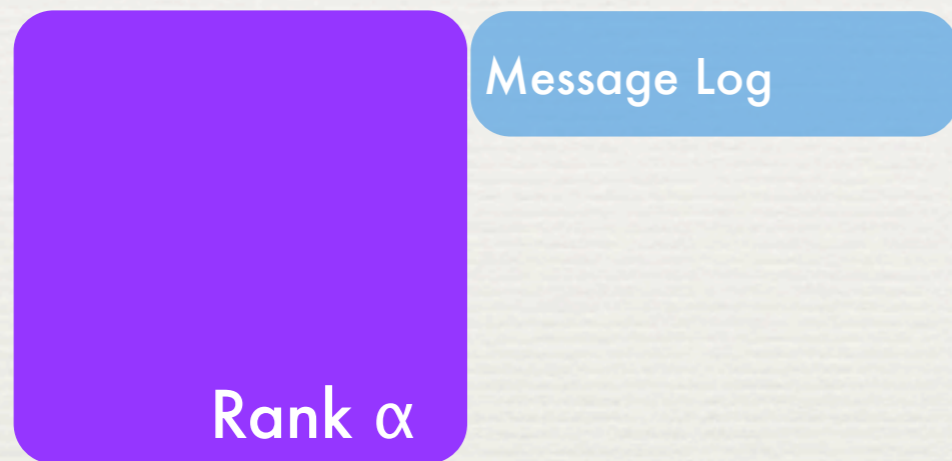During 537 days (Aug-22-2008 to Feb-10-2010)
2.33 failures per day

Sequoia will have 1.6 million cores and an exascale machine around 100 million cores...

We will see failures all the time

# Agenda

- ✦ Clusters and Message Logging.

- ✦ Static Clustering (MPI).

- ✦ Dynamic Clustering (Charm++).

- ✦ Future Work.

4th Workshop INRIA-Illinois Joint Laboratory on Petascale Computing

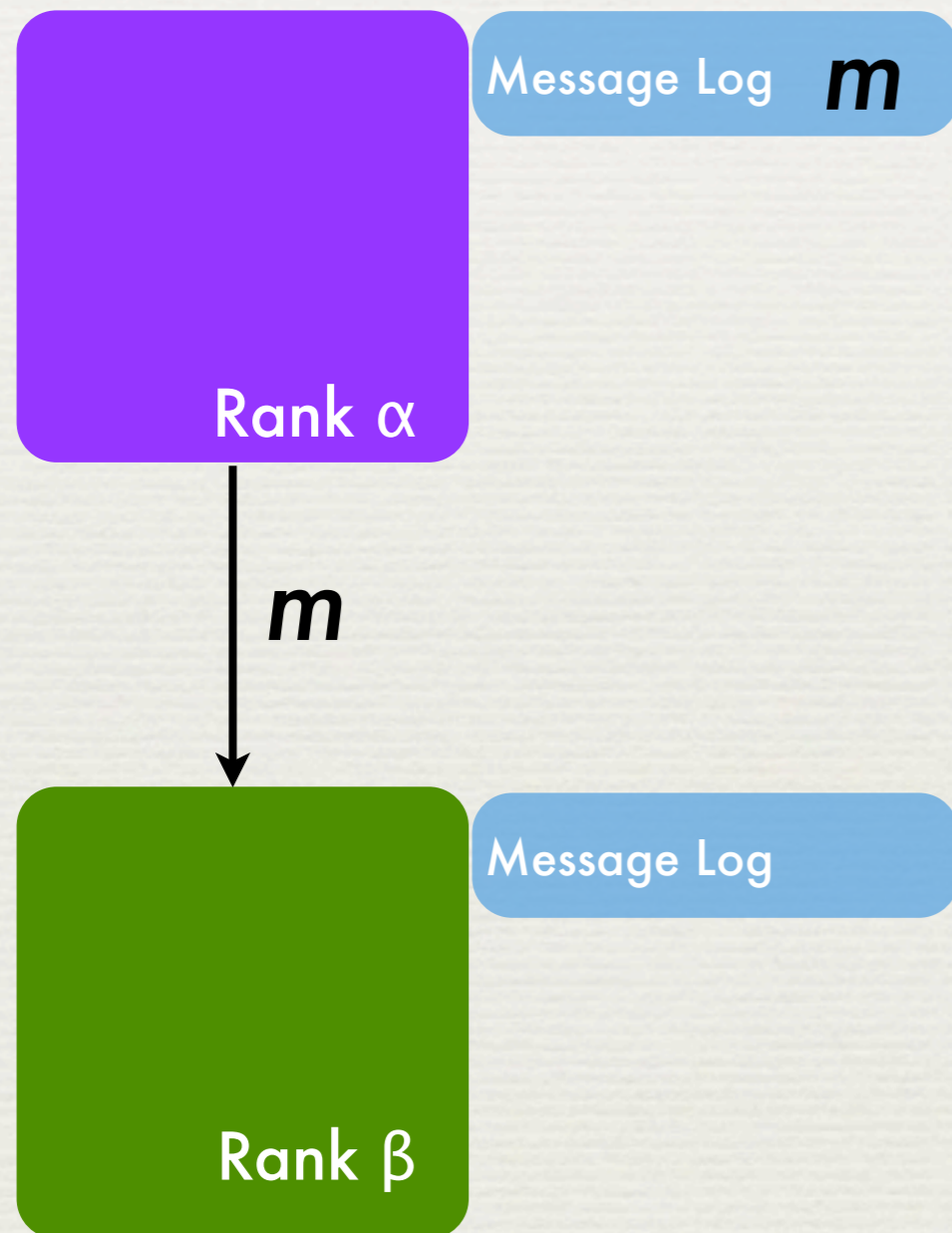# Message Logging

Rank α

Message Log

Rank β

Message Log

- Every message sent *may* be logged.

- **Advantage:** only the failed rank is rolled back.

- **Drawback**: memory overhead.

Message Logging

Rank α — Message Log

m

Rank β — Message Log

- Every message sent *may* be logged.

- **Advantage:** only the failed rank is rolled back.

- **Drawback**: memory overhead.

Message Logging

Message Log **m**

Rank α

m

Message Log

Rank β
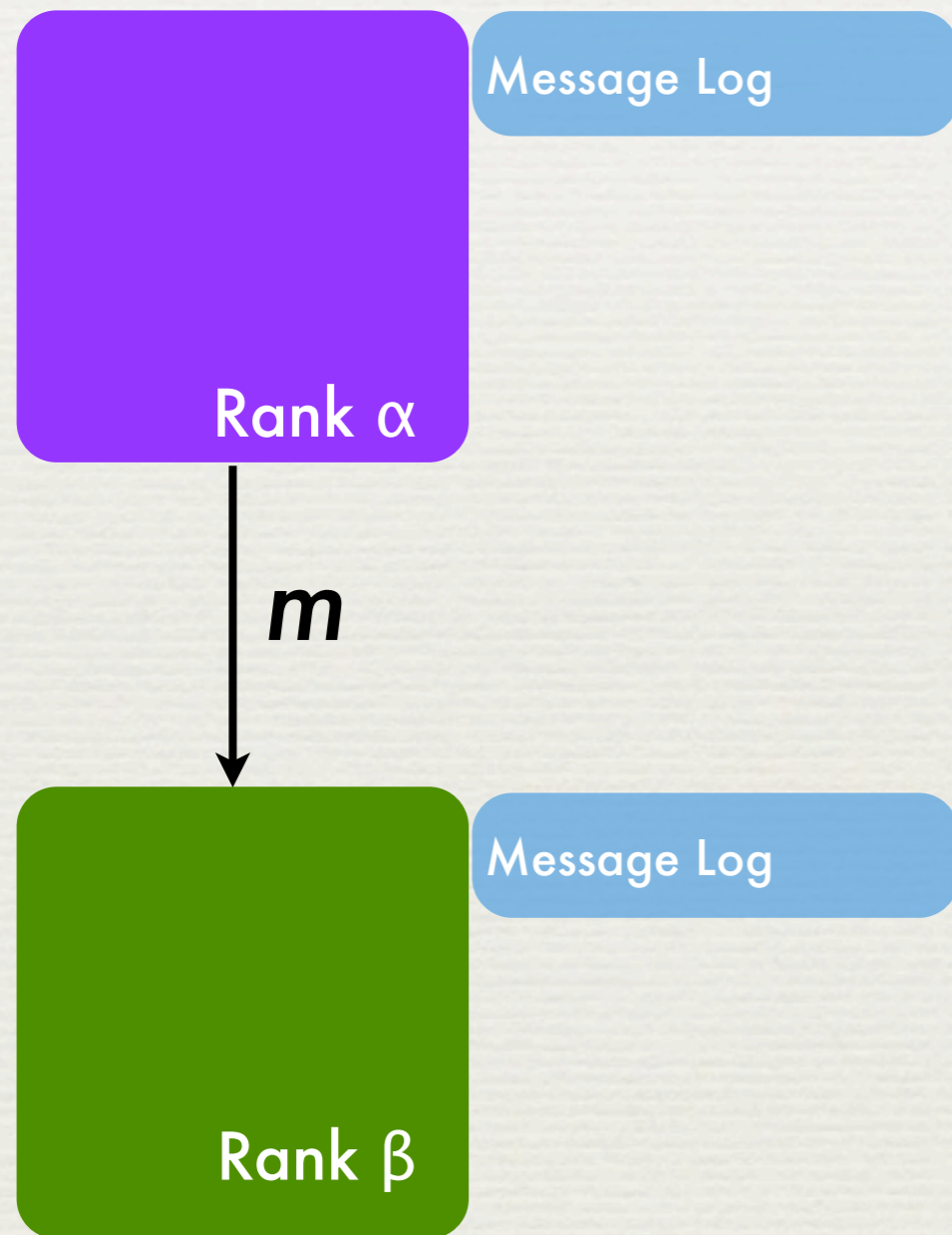
- ✦ Every message sent *may* be logged.

- ✦ **Advantage:** only the failed rank is rolled back.

- ✦ **Drawback**: memory overhead.

Teams

- **Goal**: reduce memory overhead of message log.
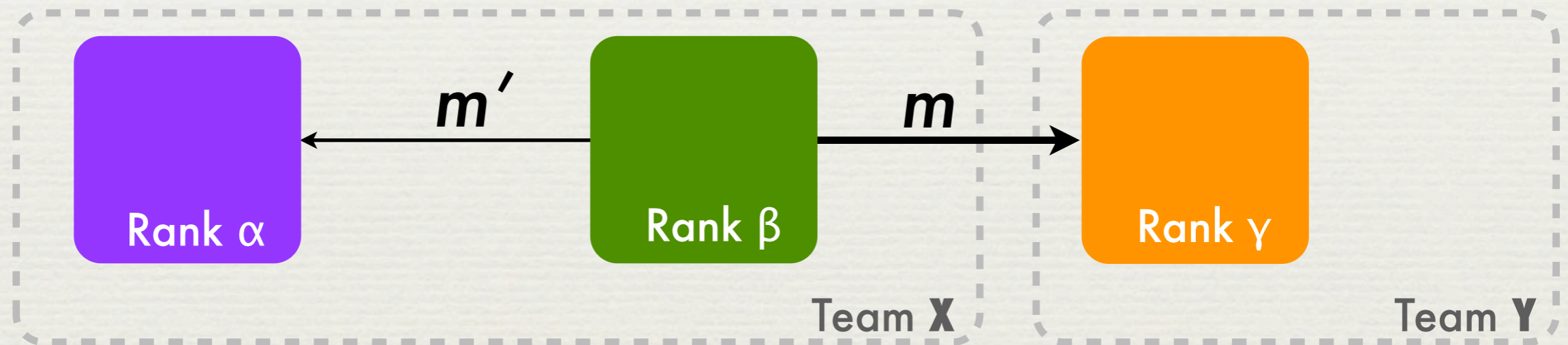
- Only messages crossing team **boundaries** are logged.

Rank α

Rank β

Team **X**

Rank γ

Team **Y**

Teams

- **Goal**: reduce memory overhead of message log.

- Only messages crossing team **boundaries** are logged.

Teams

- **Goal**: reduce memory overhead of message log.
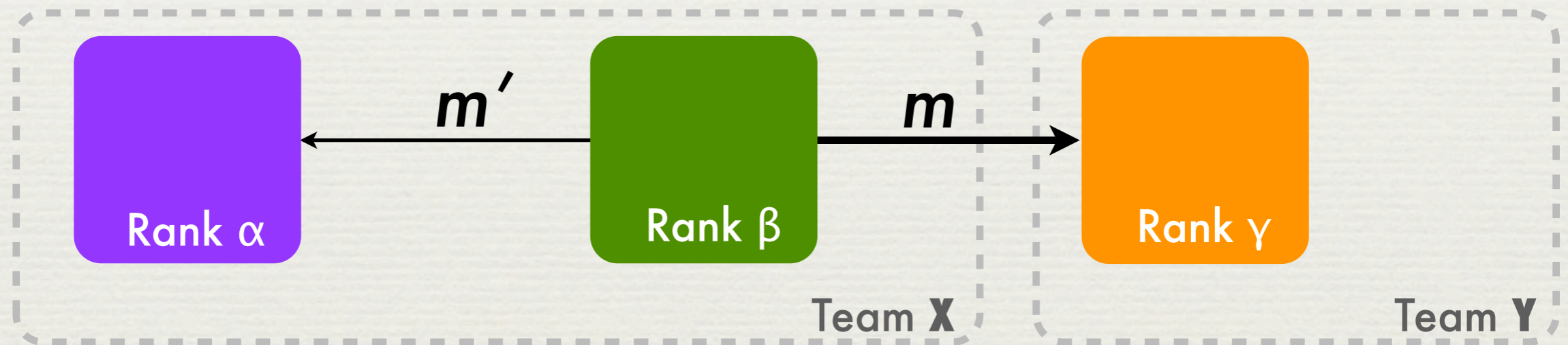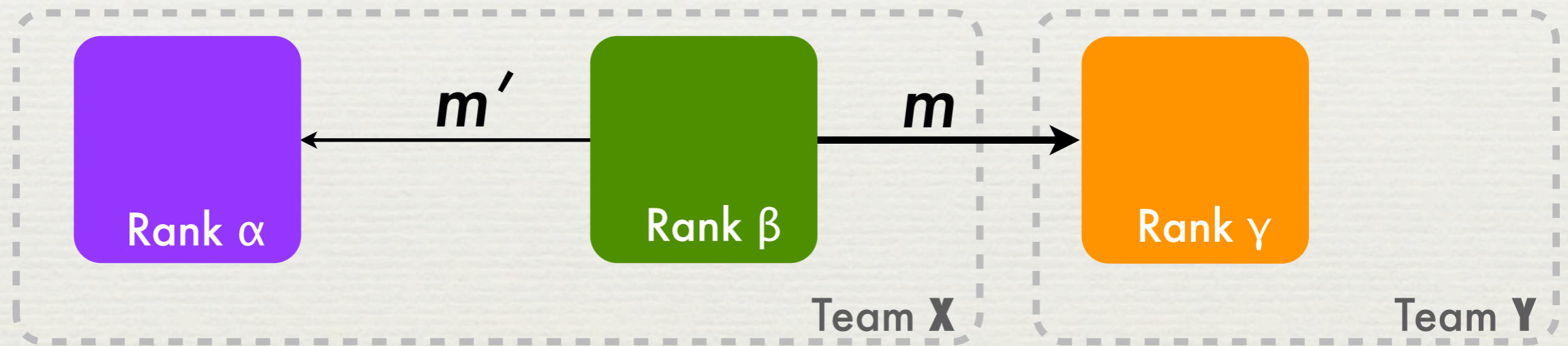
- Only messages crossing team **boundaries** are logged.

Rank α m' Rank β m Rank γ — Team **X** — Team **Y**

Teams

- **Goal**: reduce memory overhead of message log.

- Only messages crossing team **boundaries** are logged.

Teams

- **Goal**: reduce memory overhead of message log.
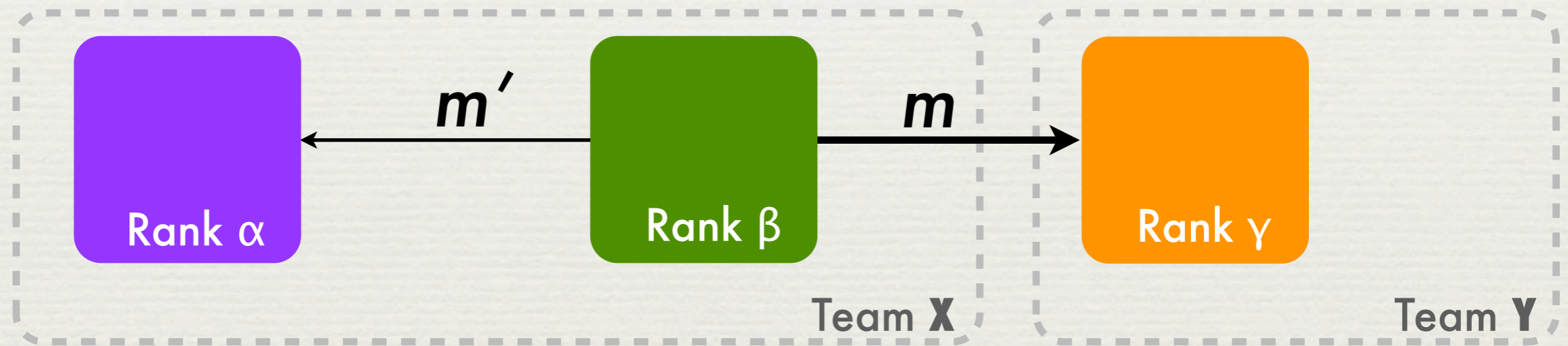
- Only messages crossing team **boundaries** are logged.

m' m

Rank α   Rank β   Rank γ

Team **X**   Team **Y**

Message Logging

1 . . . . **k** . . . . **N**

Team Size
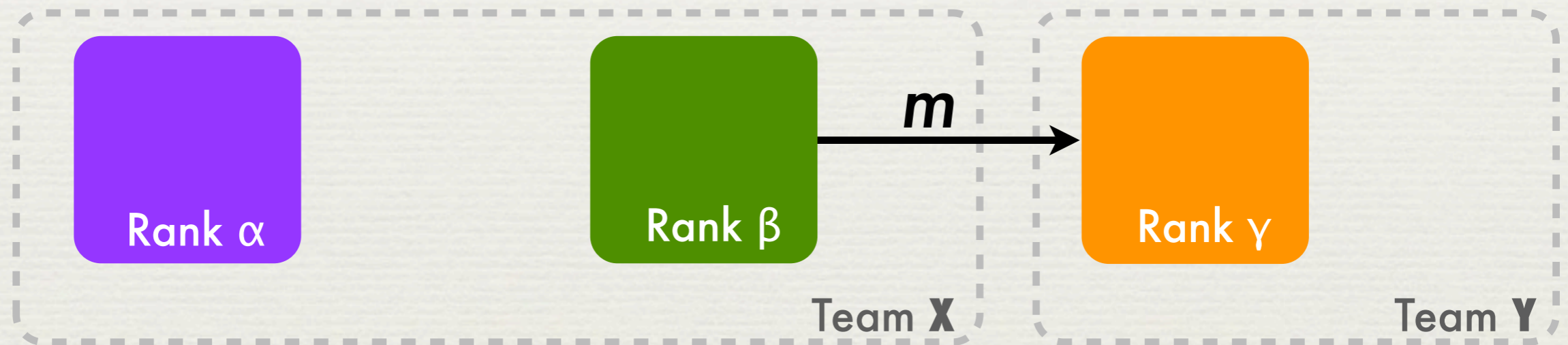
Teams

- **Goal**: reduce memory overhead of message log.

- Only messages crossing team **boundaries** are logged.

Message Logging

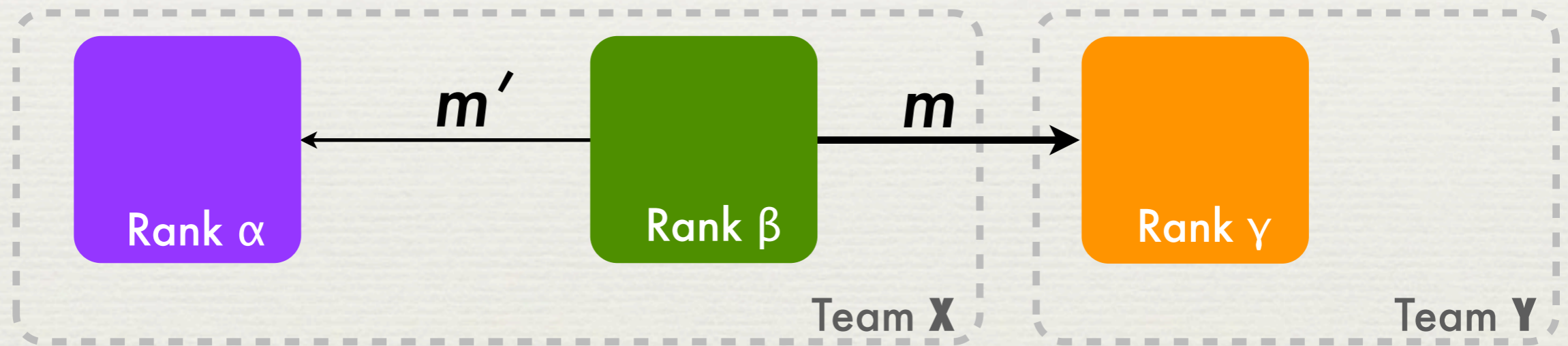**1** .... **k** .... **N**

Team Size

Checkpoint/Restart

Teams

- **Goal**: reduce memory overhead of message log.

- Only messages crossing team **boundaries** are logged.

Rank α $\xleftarrow{\;m'\;}$ Rank β $\xrightarrow{\;m\;}$ Rank γ

Team **X**   Team **Y**

Message Logging   *Checkpoint/Restart*

1 ········· **k** ········· **N**

Team Size

Teams

- **Goal**: reduce memory overhead of message log.

- Only messages crossing team **boundaries** are logged.

Rank α m' Rank β m Rank γ

Team **X**   Team **Y**

Message Logging

1 · **k** · **N**

(lower recovery cost)

Team Size

Checkpoint/Restart

Teams

- **Goal**: reduce memory overhead of message log.

- Only messages crossing team **boundaries** are logged.

Reduce Memory Overhead
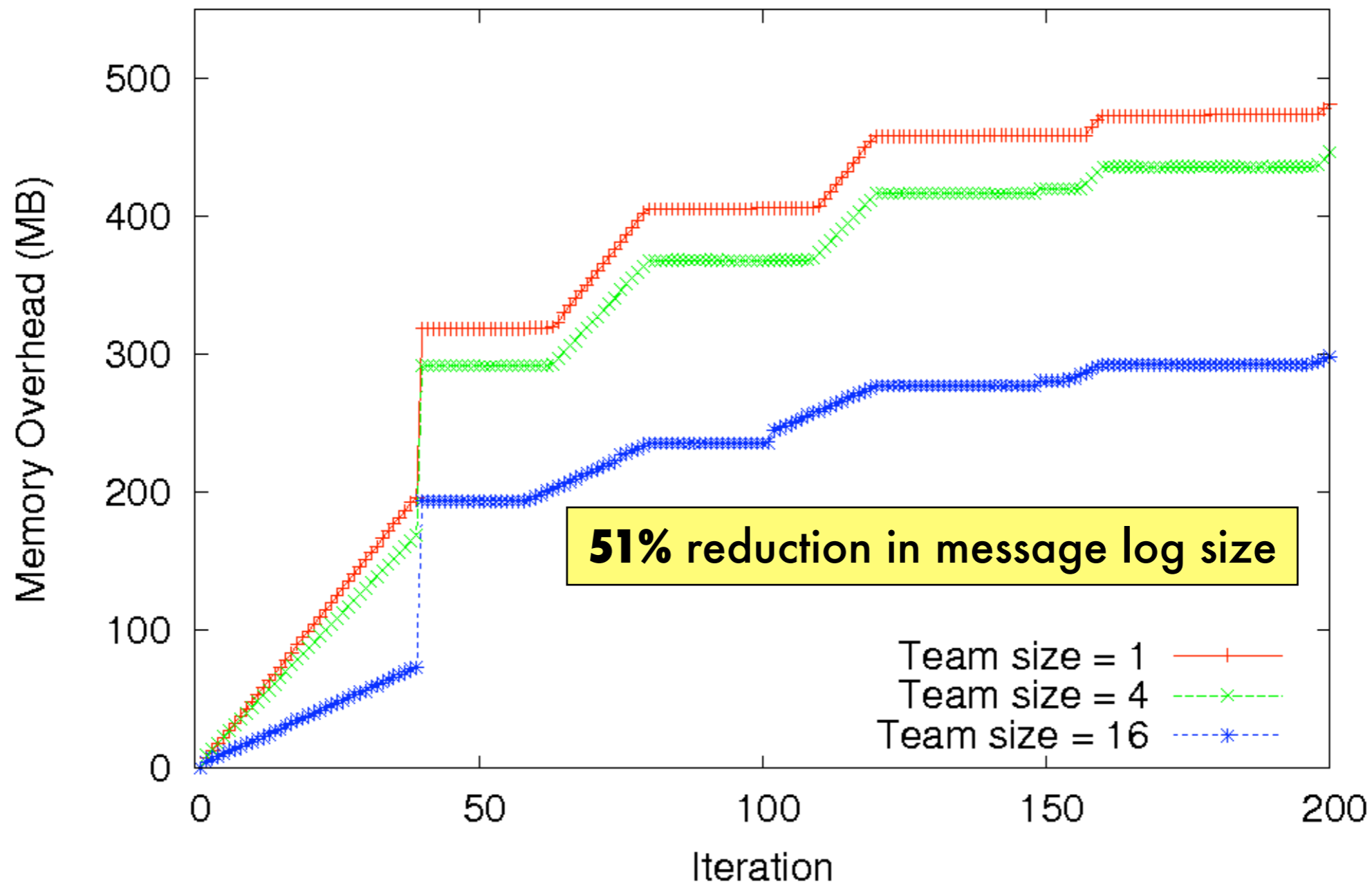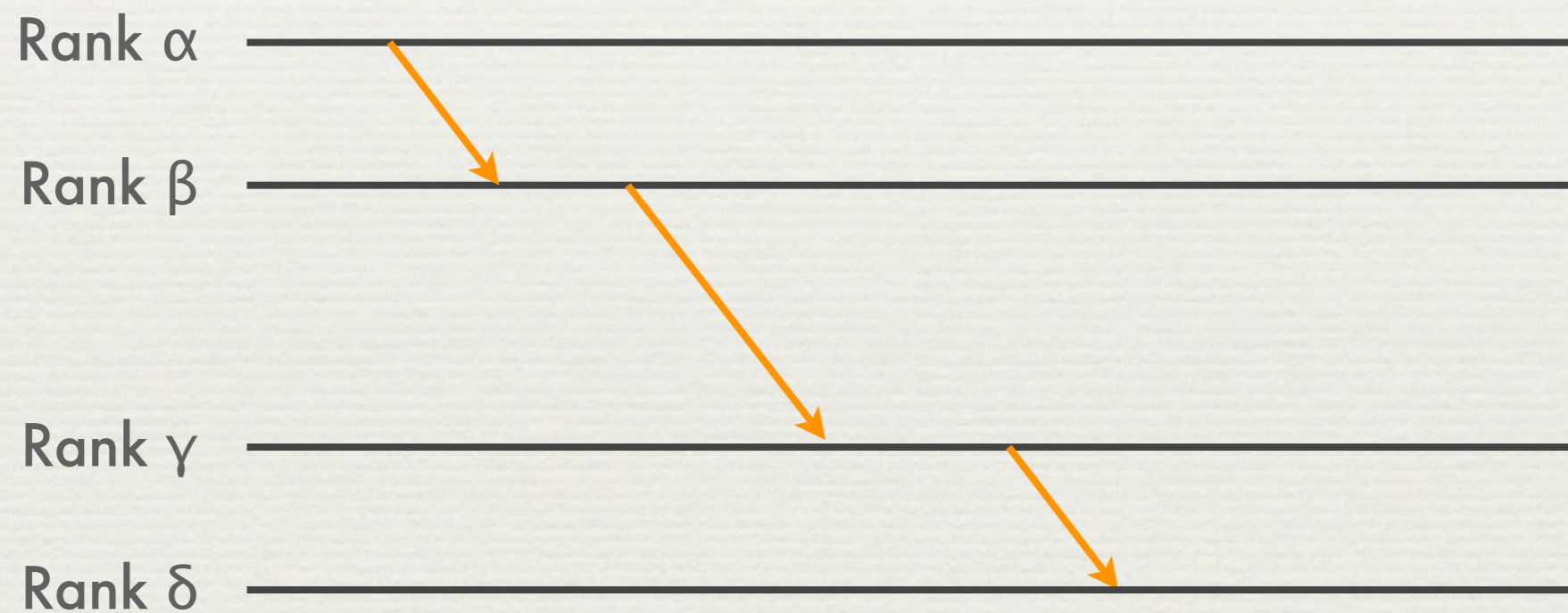
Jacobi (Abe, p=256, n=1536, b=64)

Reduce Memory Overhead

Jacobi (Abe, p=256, n=1536, b=64)

51% reduction in message log size

Team size = 1
Team size = 4
Team size = 16

Bound Cascading Rollback

Bound Cascading Rollback

Bound Cascading Rollback

Rank α

Rank β

Team **X**

Rank γ

Rank δ

F

Bound Cascading Rollback

Rank α

Rank β

Team **X**

Rank γ

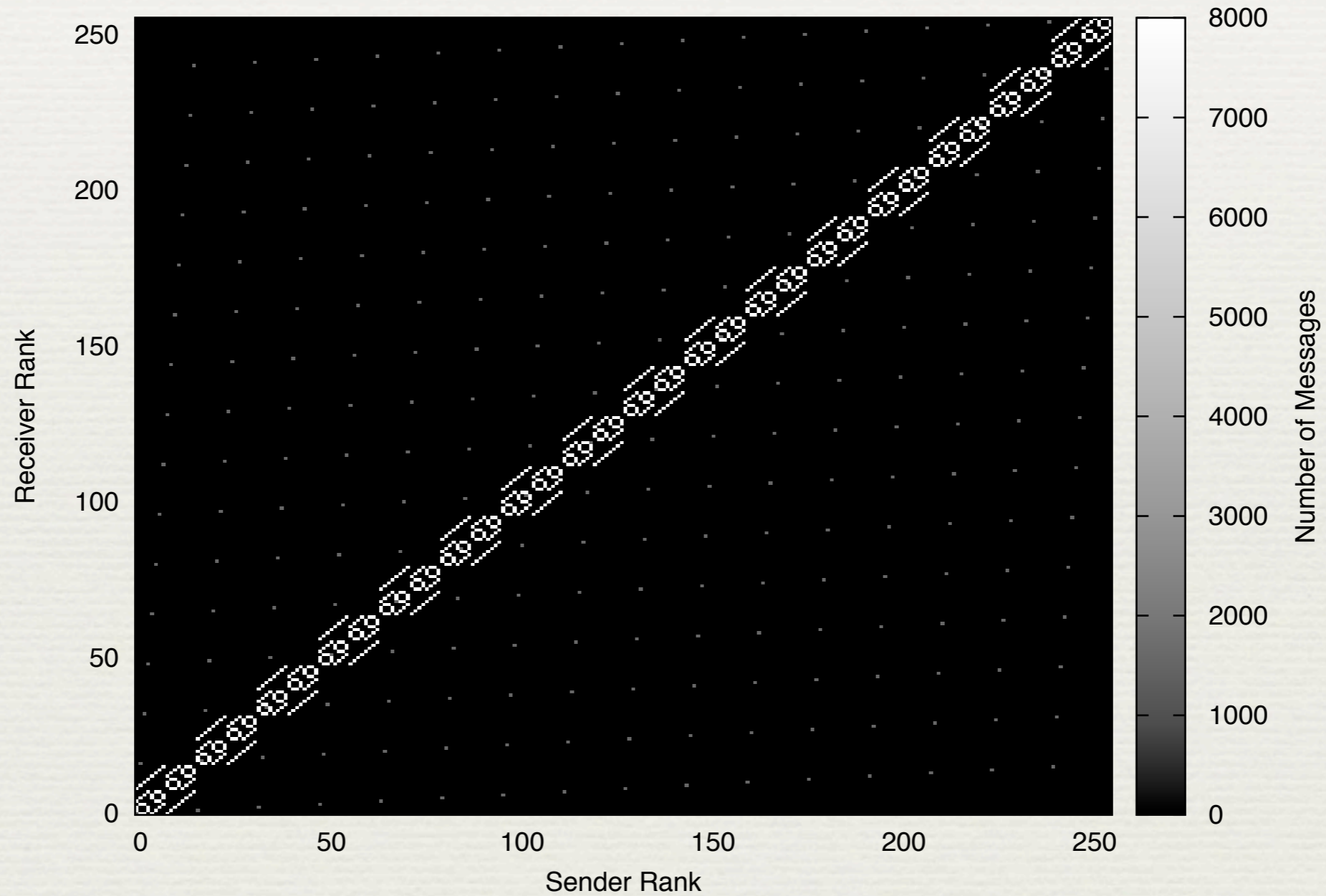Rank δ — F

Team **Y**
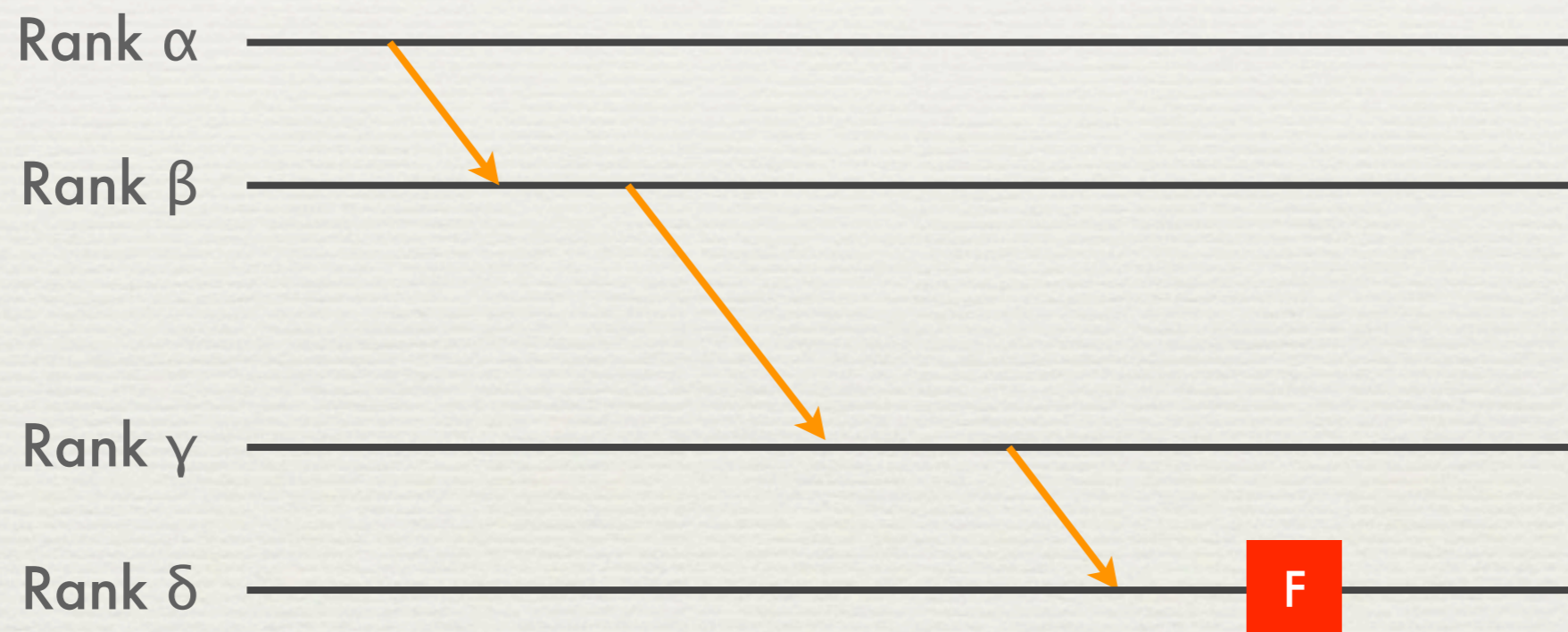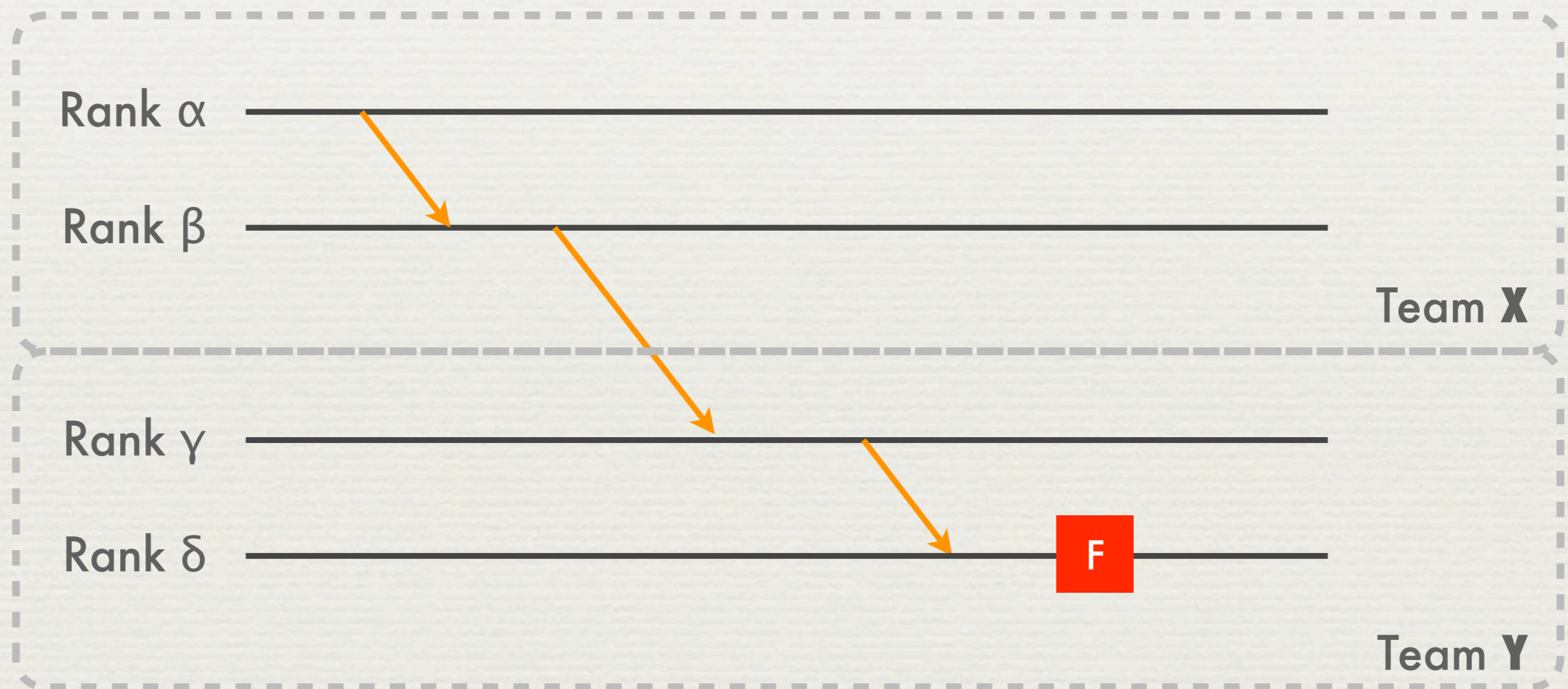
Bound Cascading Rollback

How to split the ranks to minimize the **communication volume**?

Static Clustering

Amina Guermouche
Thomas Ropars
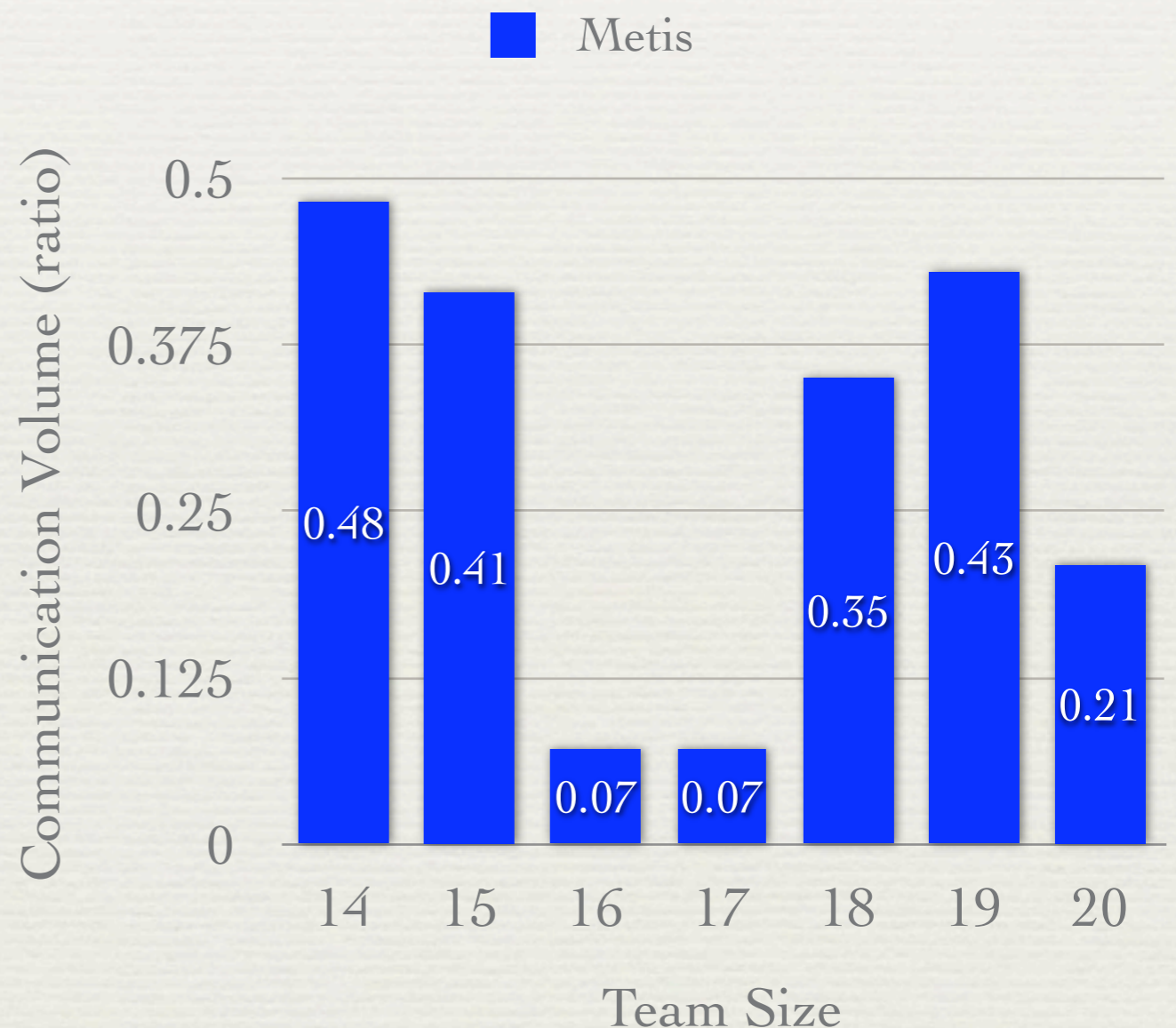Prof. Franck Cappello
(INRIA)
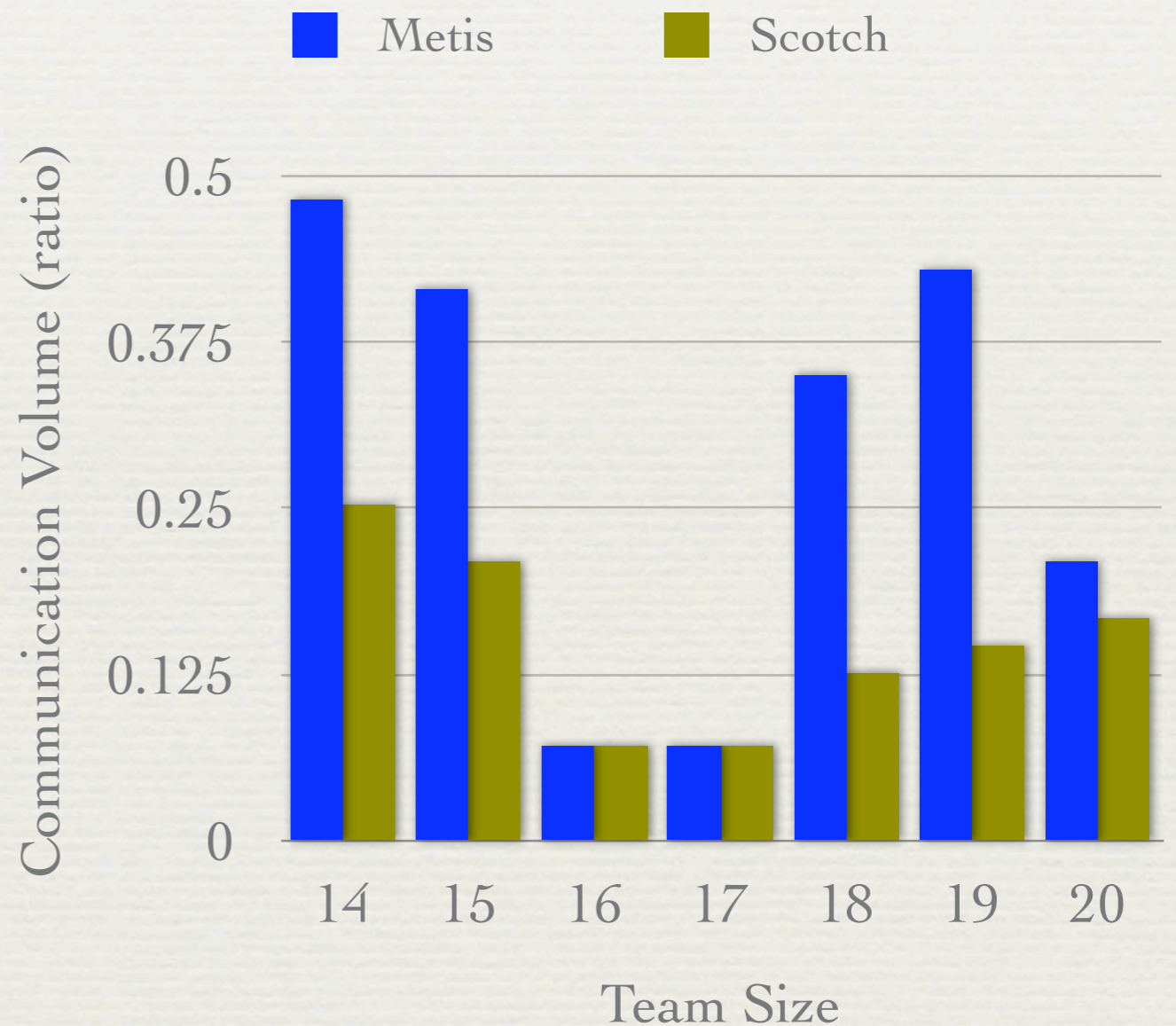
Communication Pattern

Communication Pattern (NPB CG.D.256)

Team Size

Constraint: maximum team size (t).

Graph partitioning techniques with k clusters: $k = \lceil N/t \rceil$.

Example: t=20.

Team Size

- **Constraint**: maximum team size (t).

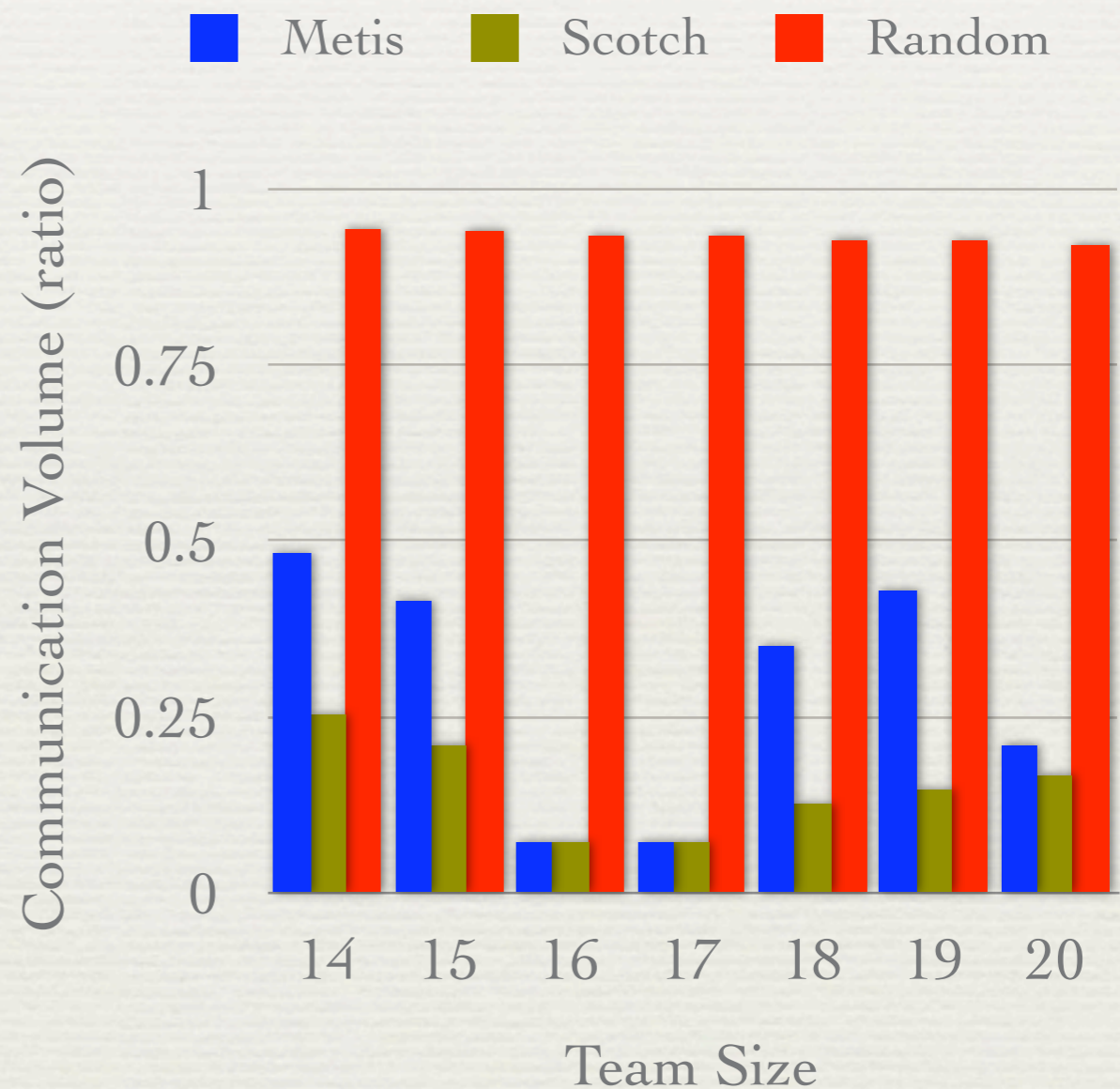- **Graph partitioning** techniques with k clusters: $k=\lceil N/t \rceil$.

- **Example**: t=20.

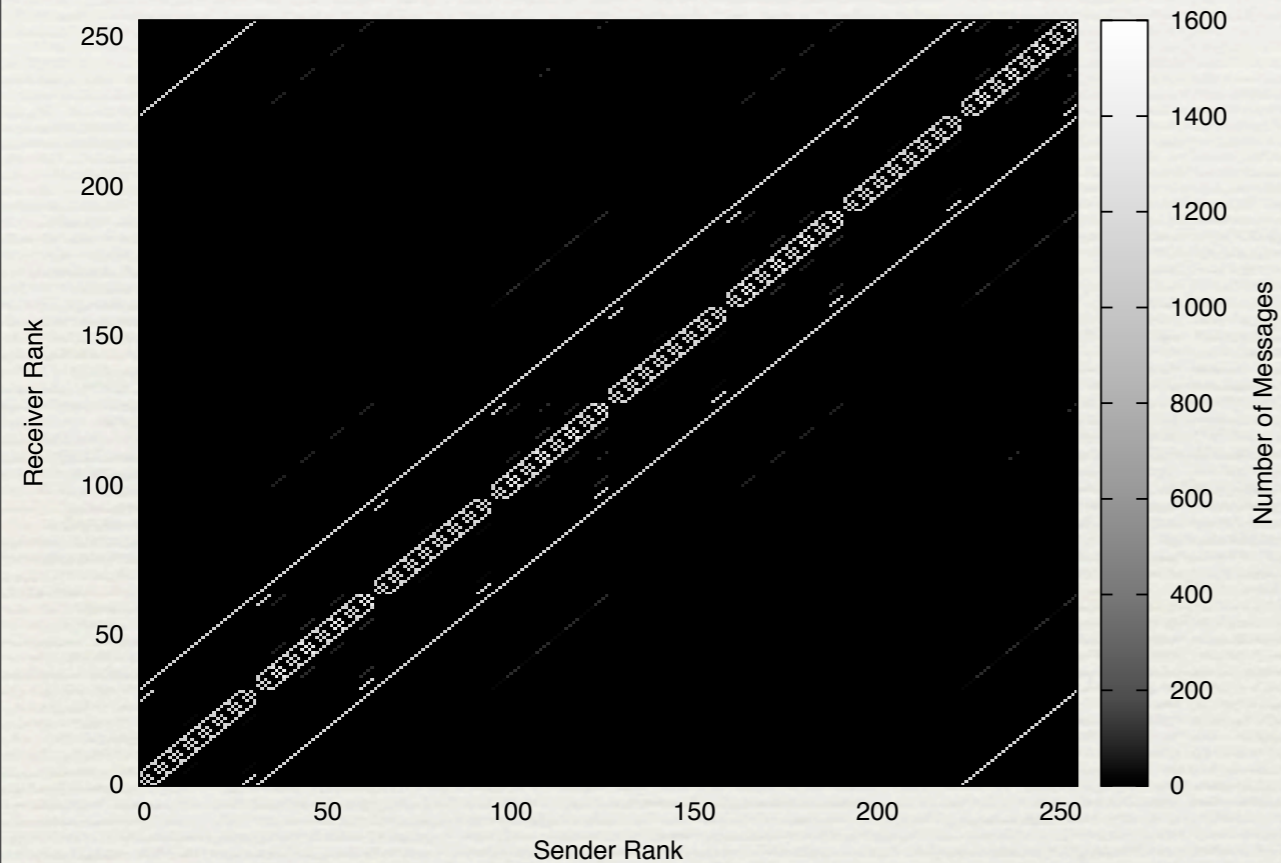# Team Size

- **Constraint**: maximum team size (t).

- **Graph partitioning** techniques with k clusters: $k = \lceil N/t \rceil$.

- **Example**: t=20.

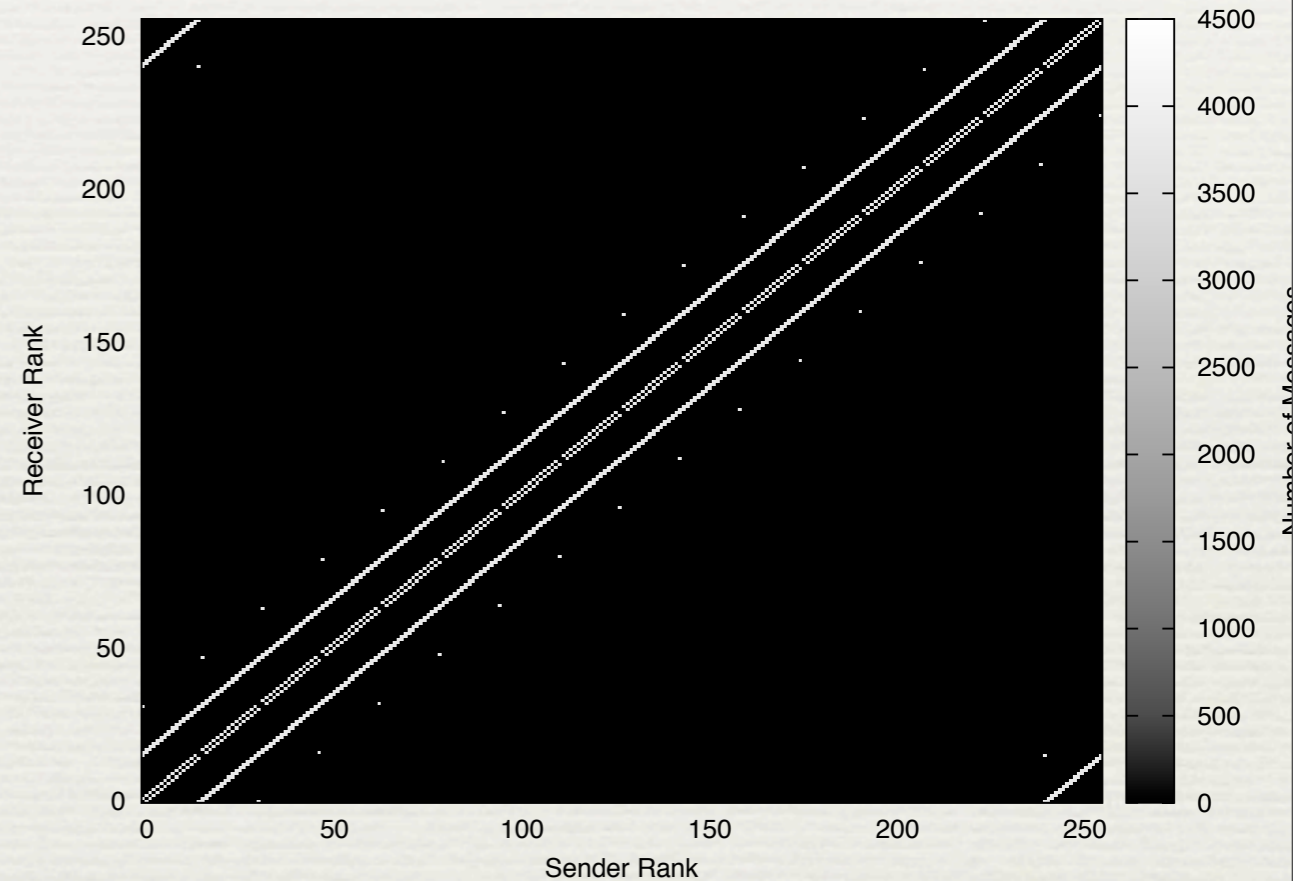# Benchmarks

Communication Pattern (NPB MG.D.256)

Communication Pattern (NPB BT.D.256)

NPB-MG

NPB-BT

Graph Properties

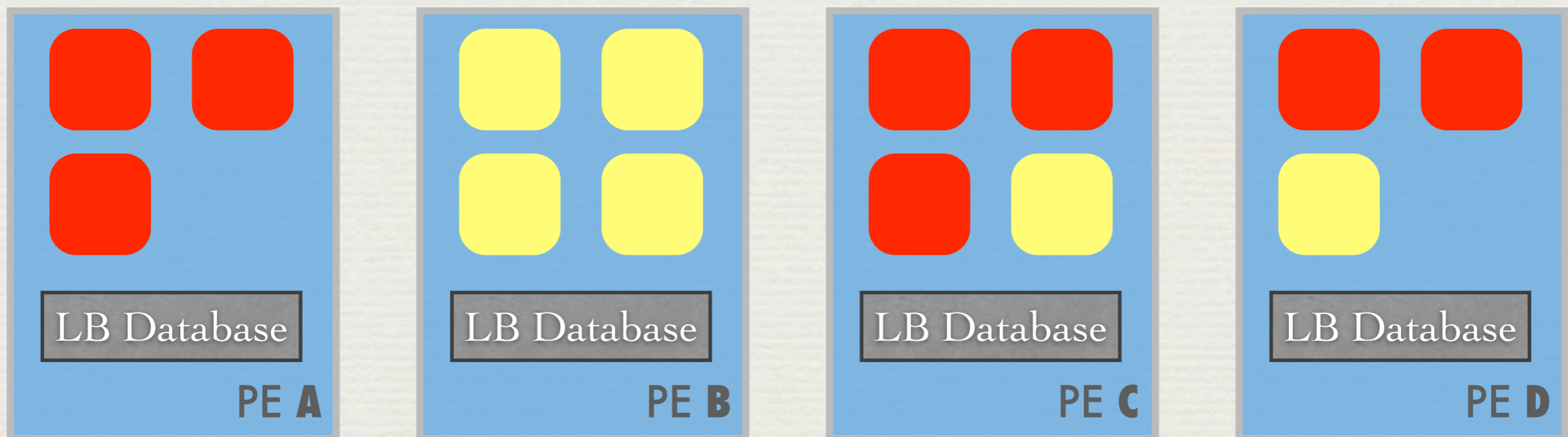| Program	Average Path Length	Clustering Coefficient	Communication Volume (ratio)		
			Metis	Scotch	Random
NPB-CG (t=16)	4.49	0	0.07	0.07	0.93
NPB-MG (t=32)	3.82	0.09	0.27	-	0.87
NPB-BT (t=16)	6.24	0.40	0.35	0.33	0.93

Dynamic Clustering

Load Balancing in Charm++

- **Migratable** objects, asynchronous method invocation.

- **Measurement-based** load balancing: collects computation load and communication structure.

LB Database — PE **A**

LB Database — PE **B**

LB Database — PE **C**

LB Database — PE **D**
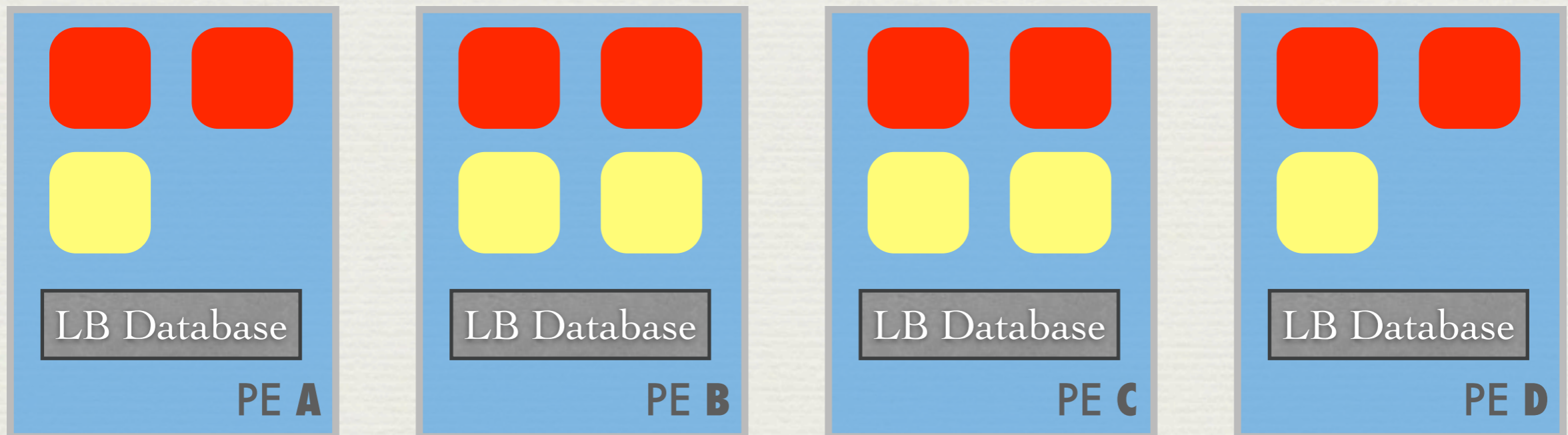
Load Balancing in Charm++

- ✦ **Migratable** objects, asynchronous method invocation.

- ✦ **Measurement-based** load balancing: collects computation load and communication structure.

Load Balancing in Charm++
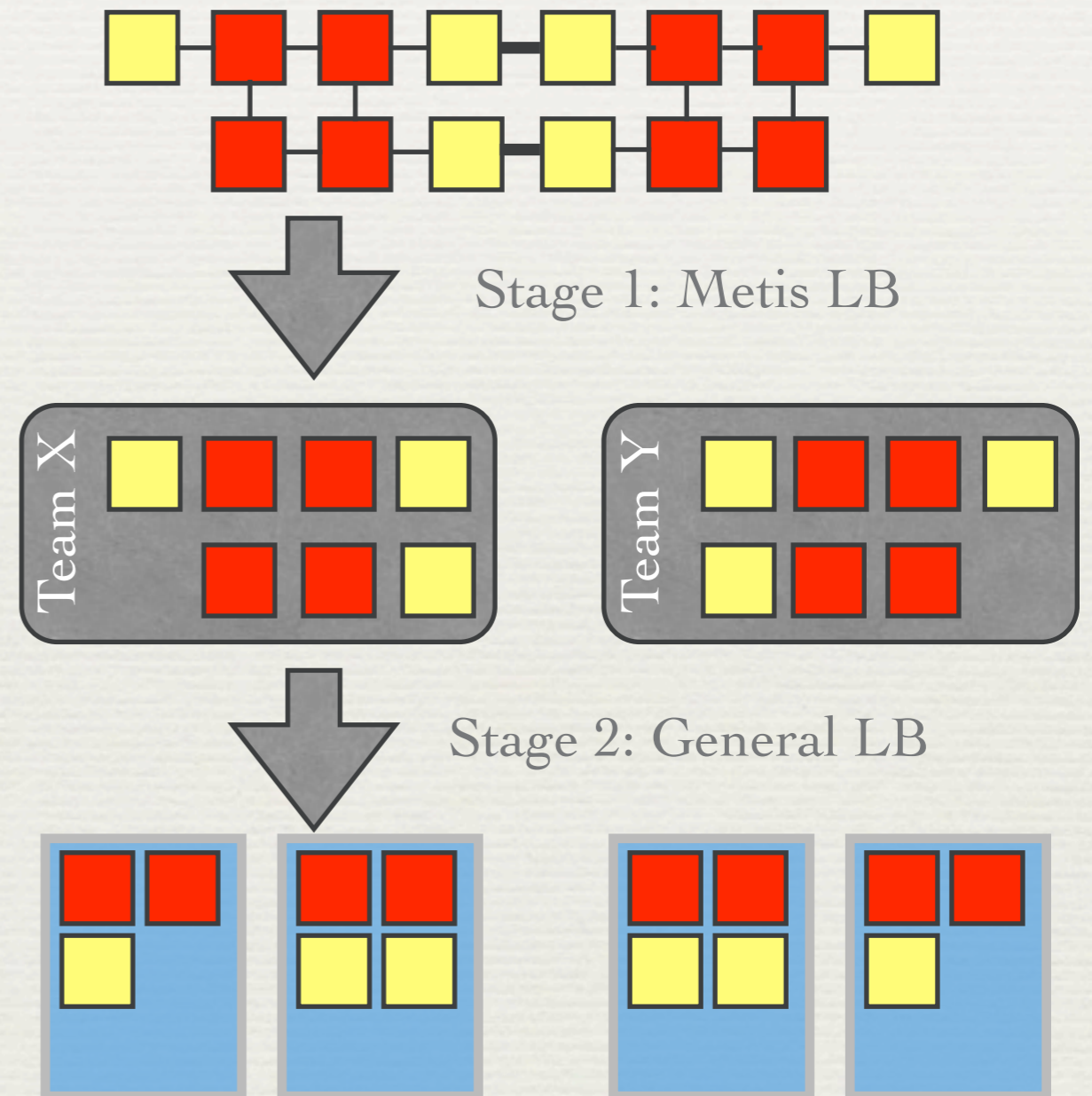
- **Migratable** objects, asynchronous method invocation.

- **Measurement-based** load balancing: collects computation load and communication structure.

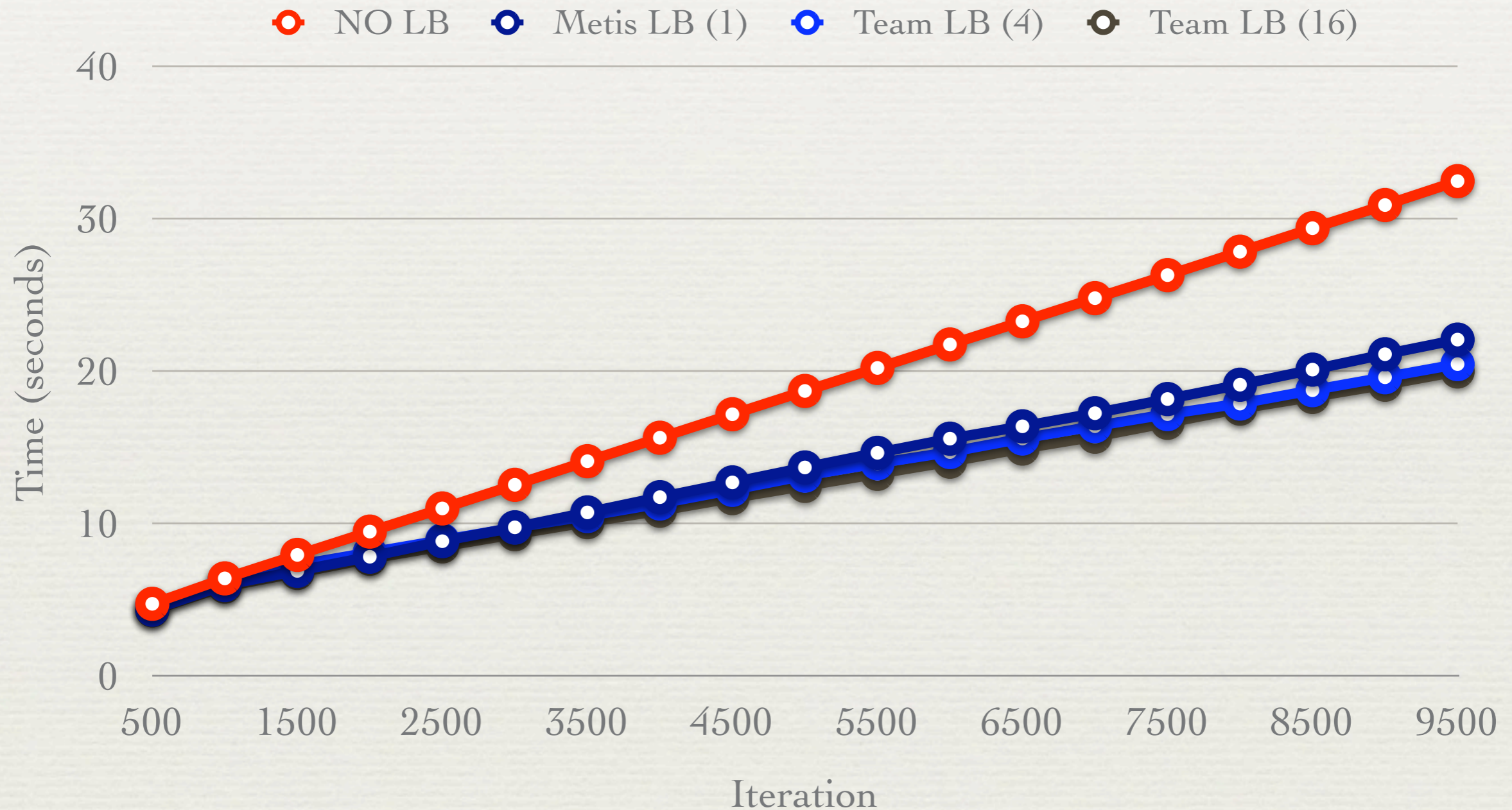# Team Load Balancer

- Divides (evenly) the objects into teams while minimizing **communication volume**.

- Team LB (t), t is the team size (number of PEs).

- **Two stage** process:

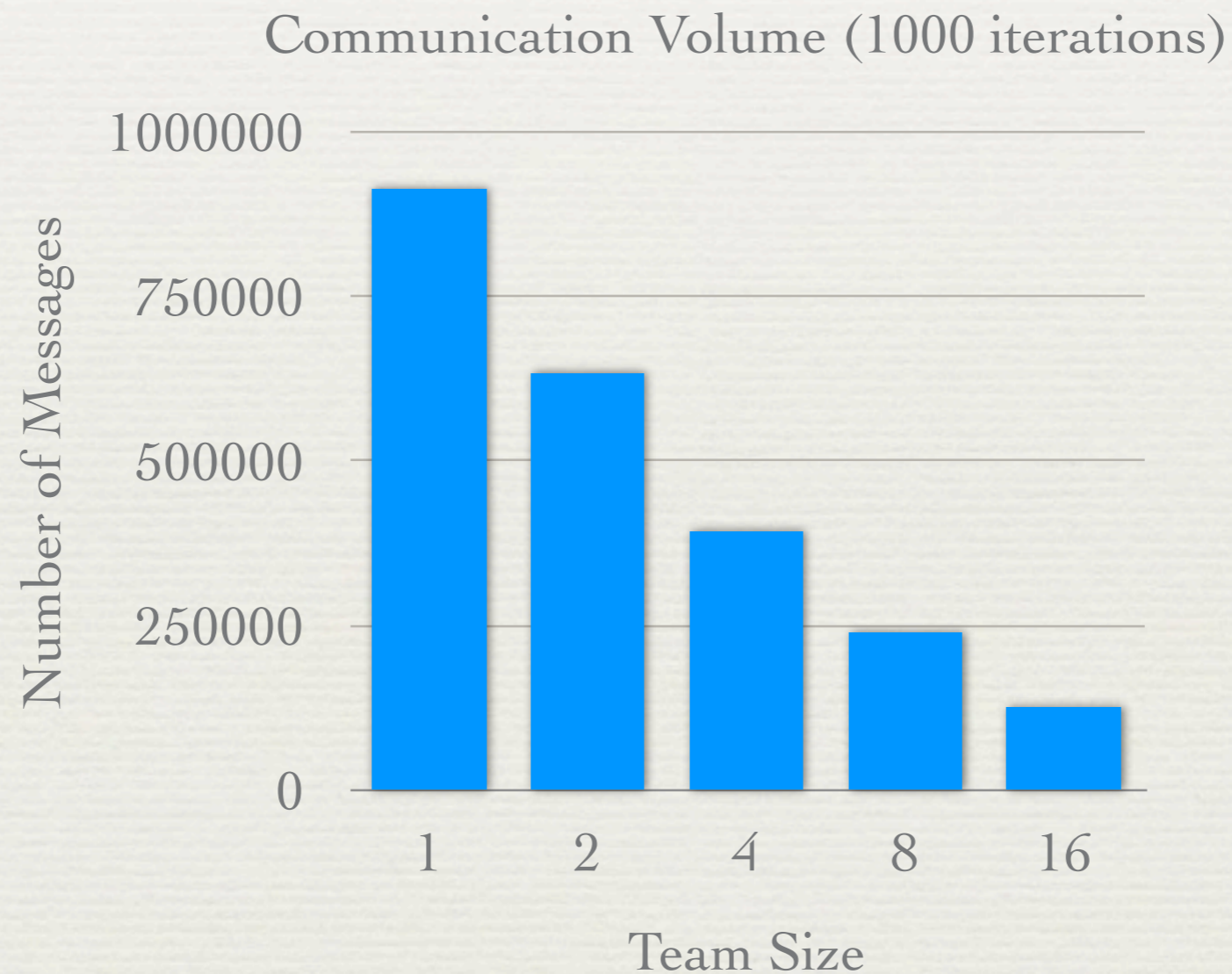 - Divide objects into teams.

  - Load balance each team.

Stage 1: Metis LB

Team X   Team Y

Stage 2: General LB

Reducing Execution Time

Legend: NO LB | Metis LB (1) | Team LB (4) | Team LB (16)

Y-axis: Time (seconds) — 0, 10, 20, 30, 40

X-axis: Iteration — 500, 1500, 2500, 3500, 4500, 5500, 6500, 7500, 8500, 9500

Reducing Message Log Size

Communication Volume (1000 iterations)

Conclusions

- Graph partitioning techniques are a promising alternative to cluster parallel applications.

- Message logging protocols benefit from team partitioning:

    - Reduce message log size.
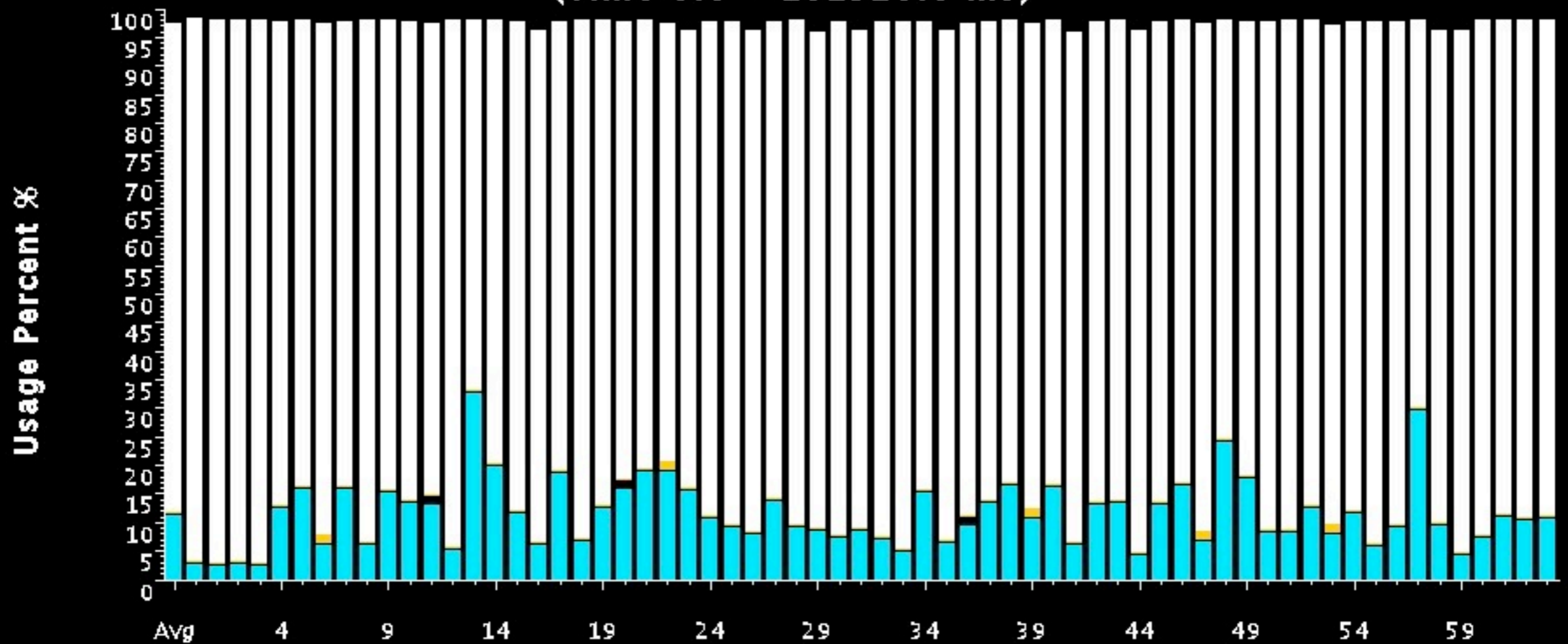
    - Avoid cascading rollback.

Future Work

* Scalable tool to collect communication information in MPI (collectives, notion of time).

* Evaluate more applications to inspect their clustering properties.

* Integration of clustering algorithms into parallel frameworks.

Q&A

Thank you!

LB Test

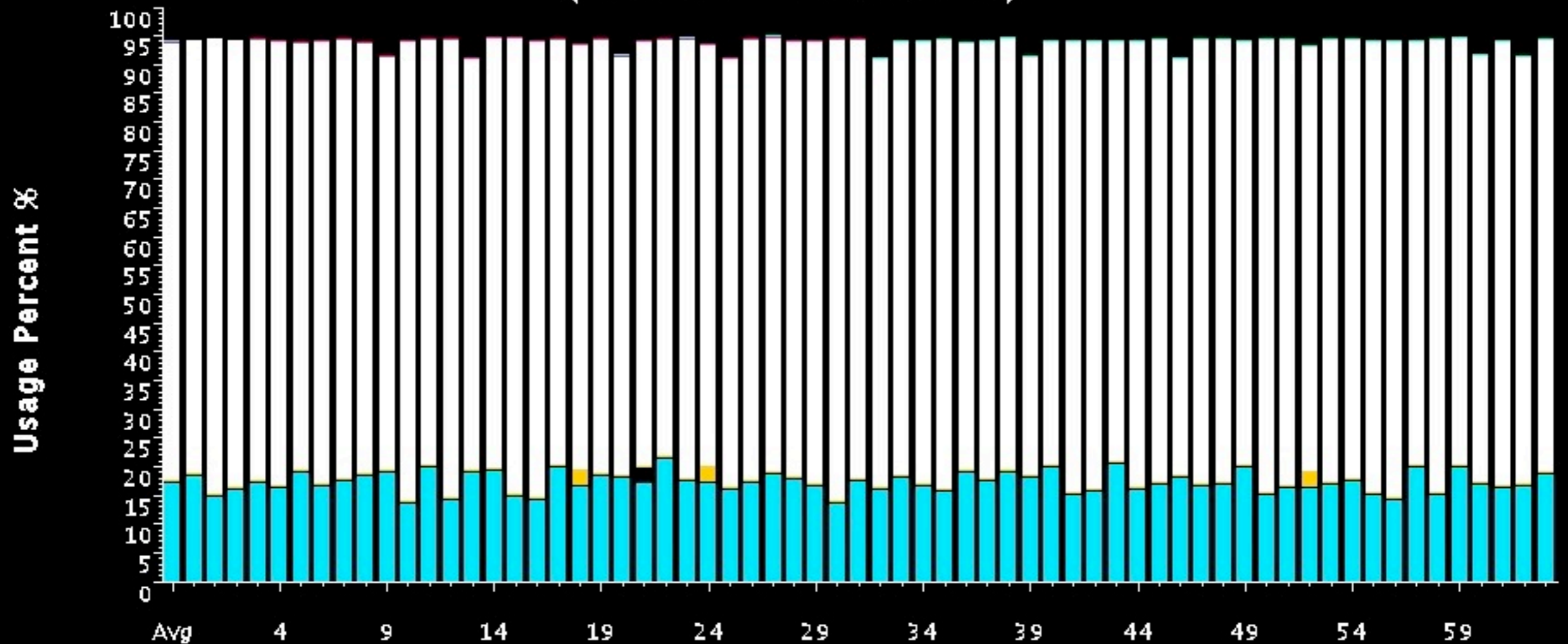Profile of Usage for Processors 0–63
(Time 0.0 ~ 101520.0 ms)

Objects: 256; Cores: 64; Topology: 2D mesh

Load Balance (Metis LB)

Profile of Usage for Processors 0–63
(Time 0.0 ~ 64498.0 ms)

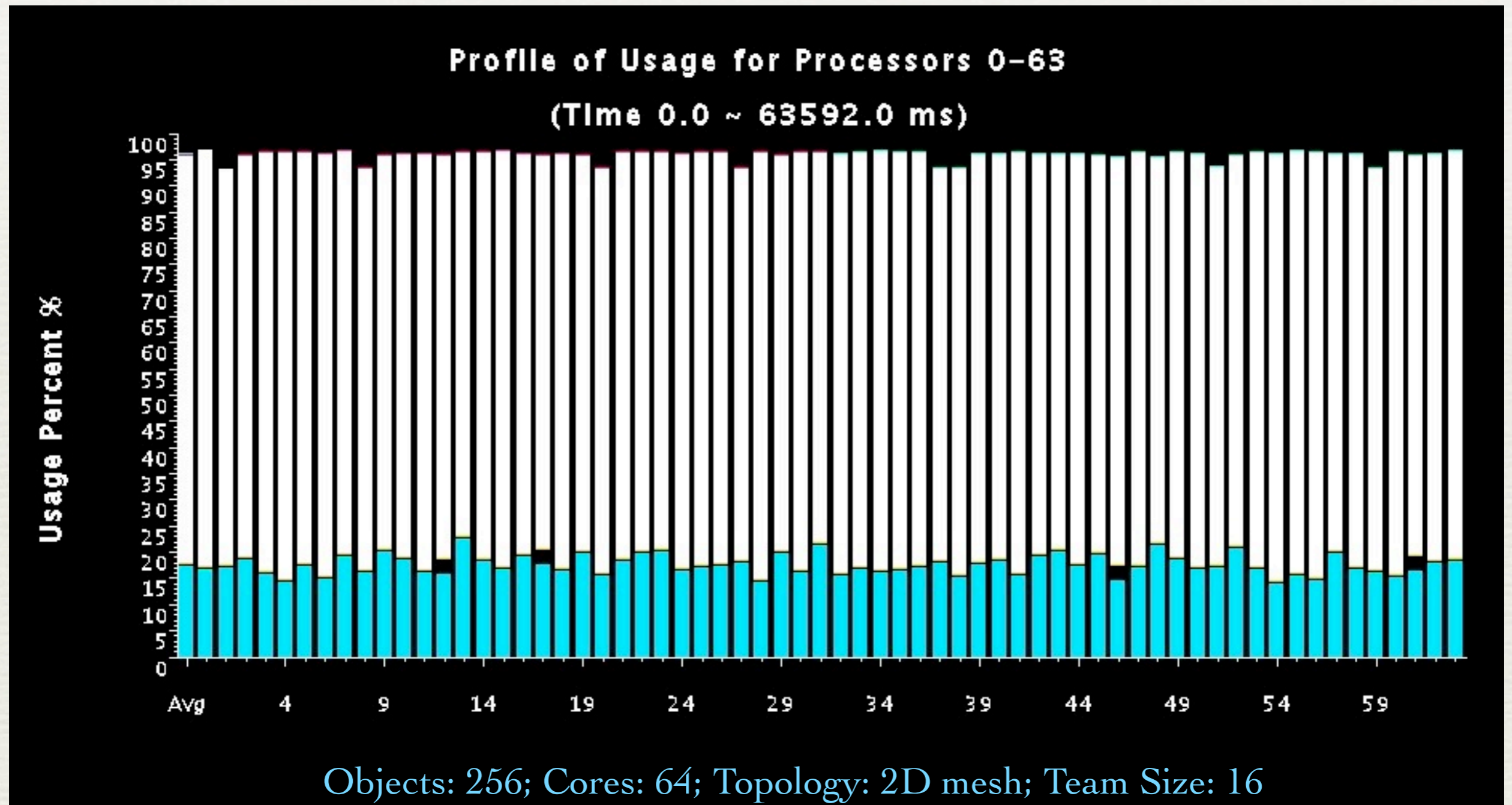Objects: 256; Cores: 64; Topology: 2D mesh

Load Balance (Team LB)

Profile of Usage for Processors 0–63 (Time 0.0 ~ 63592.0 ms)

Objects: 256; Cores: 64; Topology: 2D mesh; Team Size: 16

4th Workshop INRIA-Illinois Joint Laboratory on Petascale Computing