Scalable Fault Tolerance with Charm++

Esteban Meneses
Gengbin Zheng
Celso L. Mendes
Laxmikant V. Kalé

PARALLEL PROGRAMMING LAB
DEPT OF COMPUTER SCIENCE, UNIVERSITY OF ILLINOIS
PPL UIUC

Contents

- Fault Tolerance Techniques in Charm++
- Recent Developments
- Future Work

A problem hard to ignore

Installed	System	Processors	SMTBF
2000	ASCI White	8,192	40.0 h
2001	PSC Lemieux	3,016	9.7 h
2002	NERSC Seaborg	6,656	351.0 h
2002	ASCI Q	8,192	6.5 h
2003	Google	15,000	1.2 h
2006	Blue Gene/L	131,072	147.8 h

Extract taken from *High-End Computing Resilience* [1]

We will live with failures

2484 separate node crashes on *Jaguar* during 537 days period (Aug-22-2008 to Feb-10-2010)

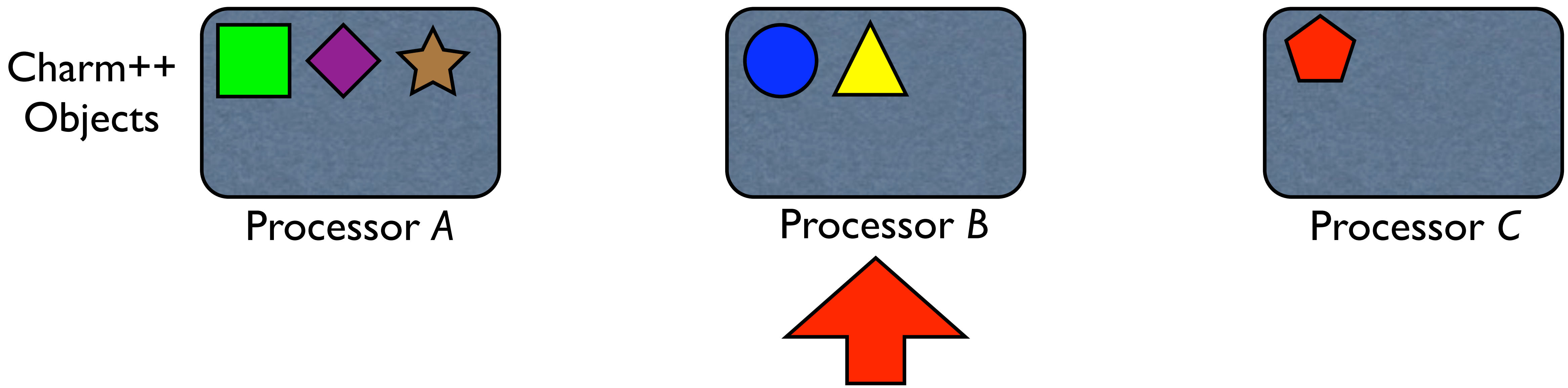
4.62 failures per day

What about *Sequoia* with 1.6 million cores or an exascale machine with 100 million cores?

Overview of Charm++ Fault Tolerant Techniques

Proactive Fault Tolerance

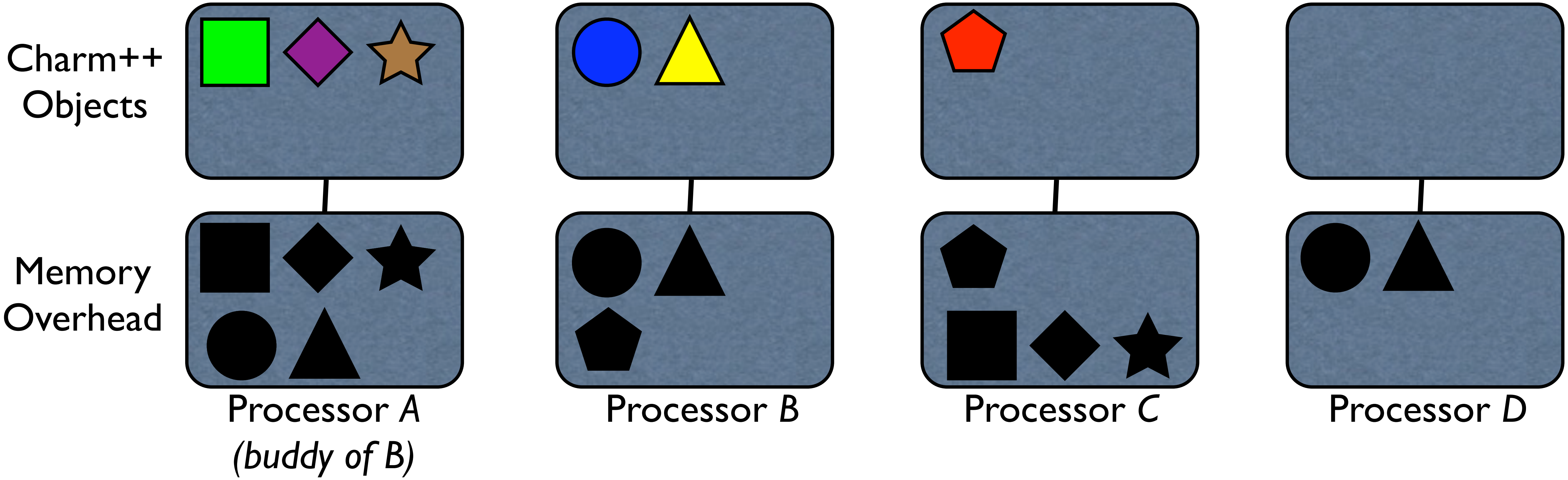
- Use knowledge about **impending** faults.
- **Evacuate** objects from processors that may fail soon.



Sayantan Chakravorty, Celso L. Mendes, Laxmikant V. Kale, **Proactive Fault Tolerance in MPI Applications via Task Migration**, *In Proceedings of HIPC 2006, LNCS volume 4297, page 485*

Checkpoint/Restart

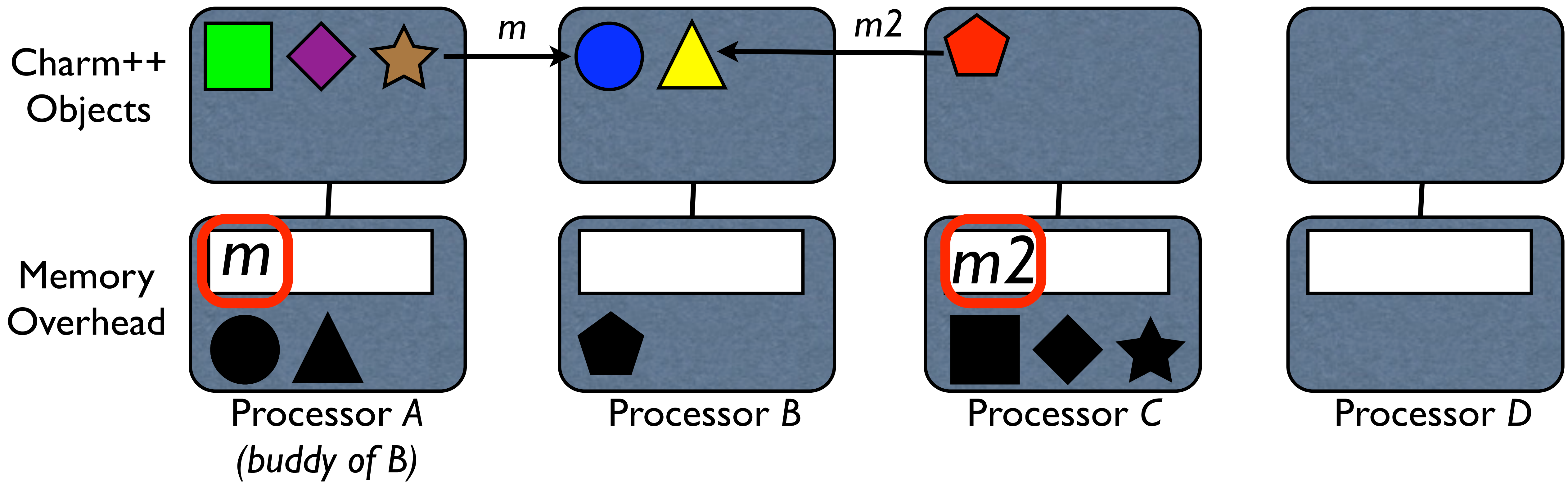
- **Double** in-memory checkpoint.
- **Synchronized** checkpoint.



Gengbin Zheng, Lixia Shi, Laxmikant V. Kale, **FTC-Charm++: An In-Memory Checkpoint-Based Fault Tolerant Runtime for Charm++ and MPI**, *Cluster 2004*

Message Logging

- Every message is stored in the **sender** log.
- **Pessimistic**: messages and determinants have to be stored before delivery.



Sayantan Chakravorty, Laxmikant V. Kale, **A Fault Tolerance Protocol with Fast Fault Recovery**, *Proceedings of the 21st International Parallel and Distributed Processing Symposium, 2007, Long Beach California*

Comparison

(Reactive Approaches)

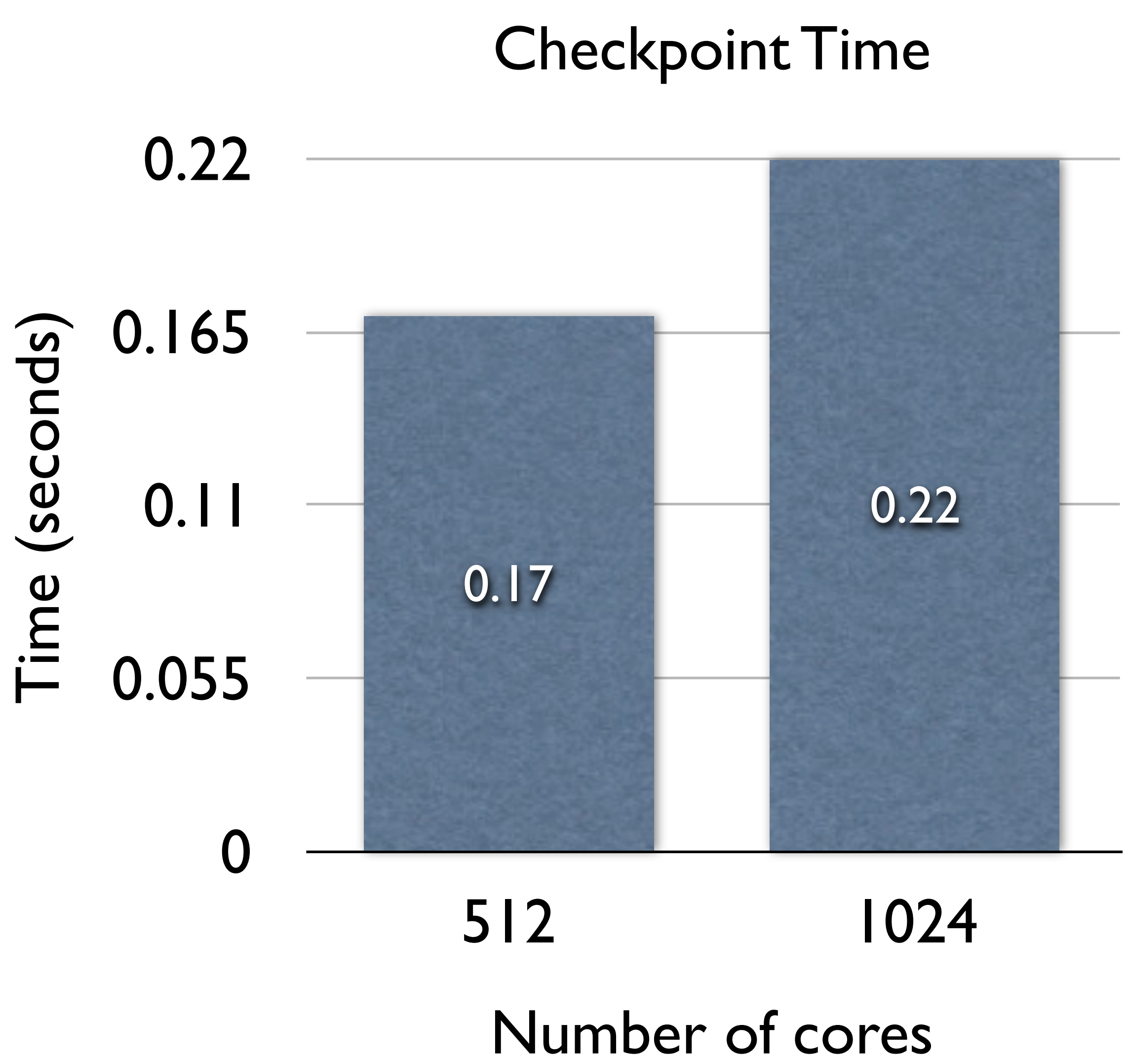
Technique	Memory Overhead	Communication Overhead	Recovery Time
Checkpoint/ Restart	☻	☺	☹
Message Logging	☹	☹	☺

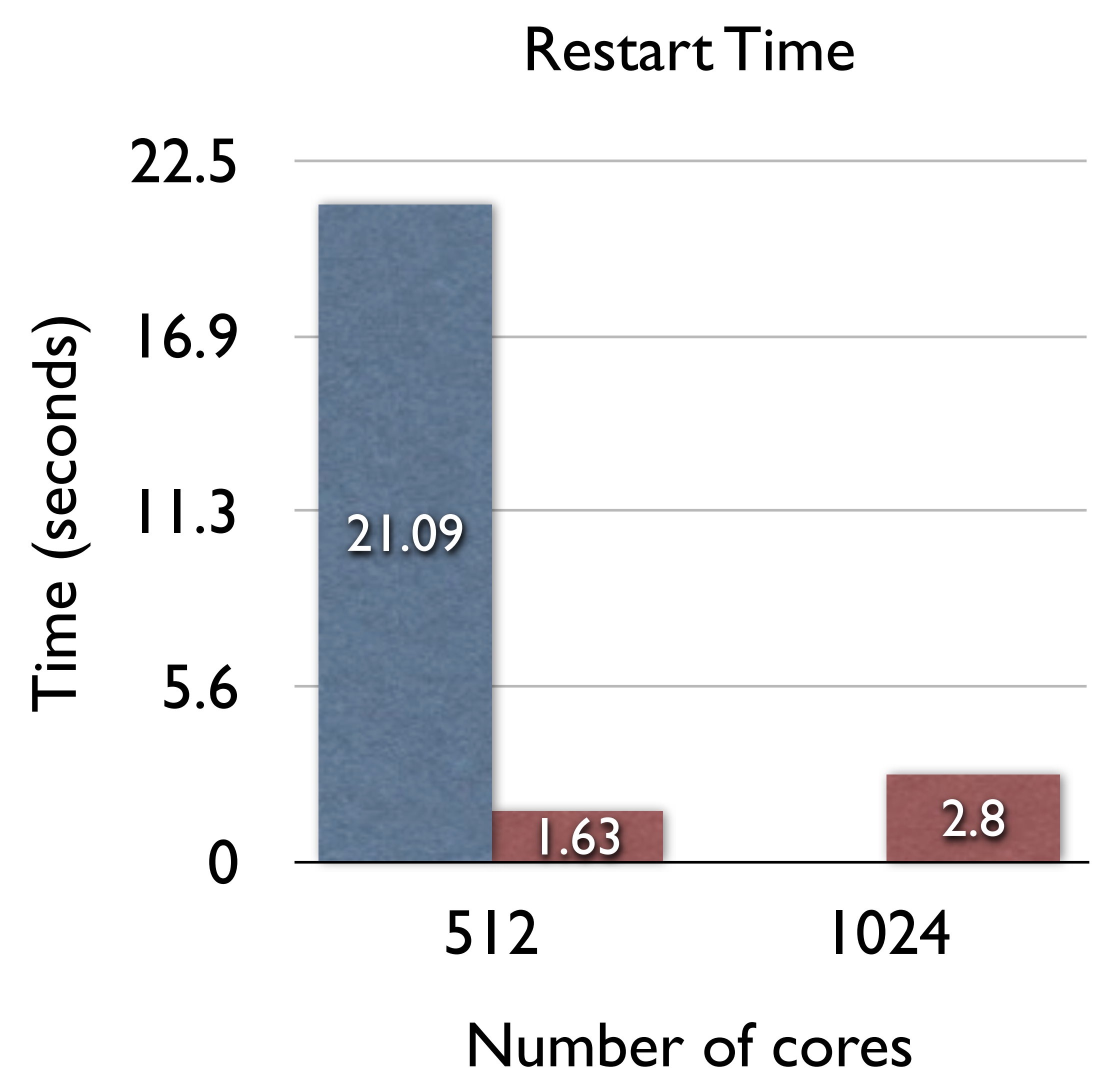
Recent Developments

Checkpoint/Restart Optimization

- Discard old messages to resume progress as soon as possible.
- Improve quiescence detection.
- Combine message to update home location of objects.

Results





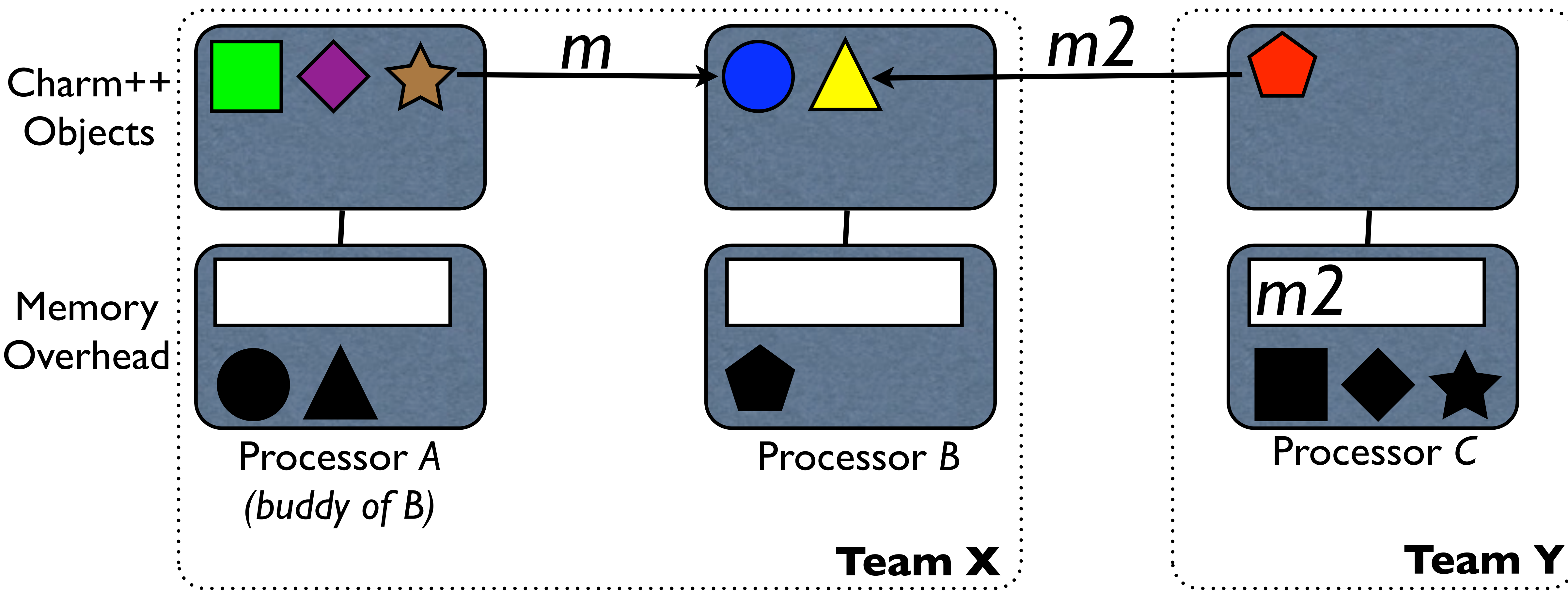
Application: Molecular3D (APOA1 ~100K atoms)
Data Size: 624 KB per core (512 cores), 351 KB per core (1024 cores)

Message Logging Optimization

- Memory overhead reduction:
  - Team-based approach.
- Latency overhead reduction:
  - Causal protocol.

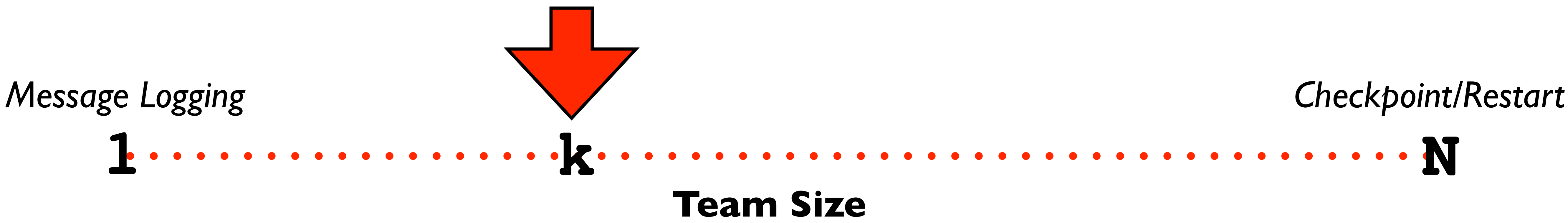
Team-based Approach

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



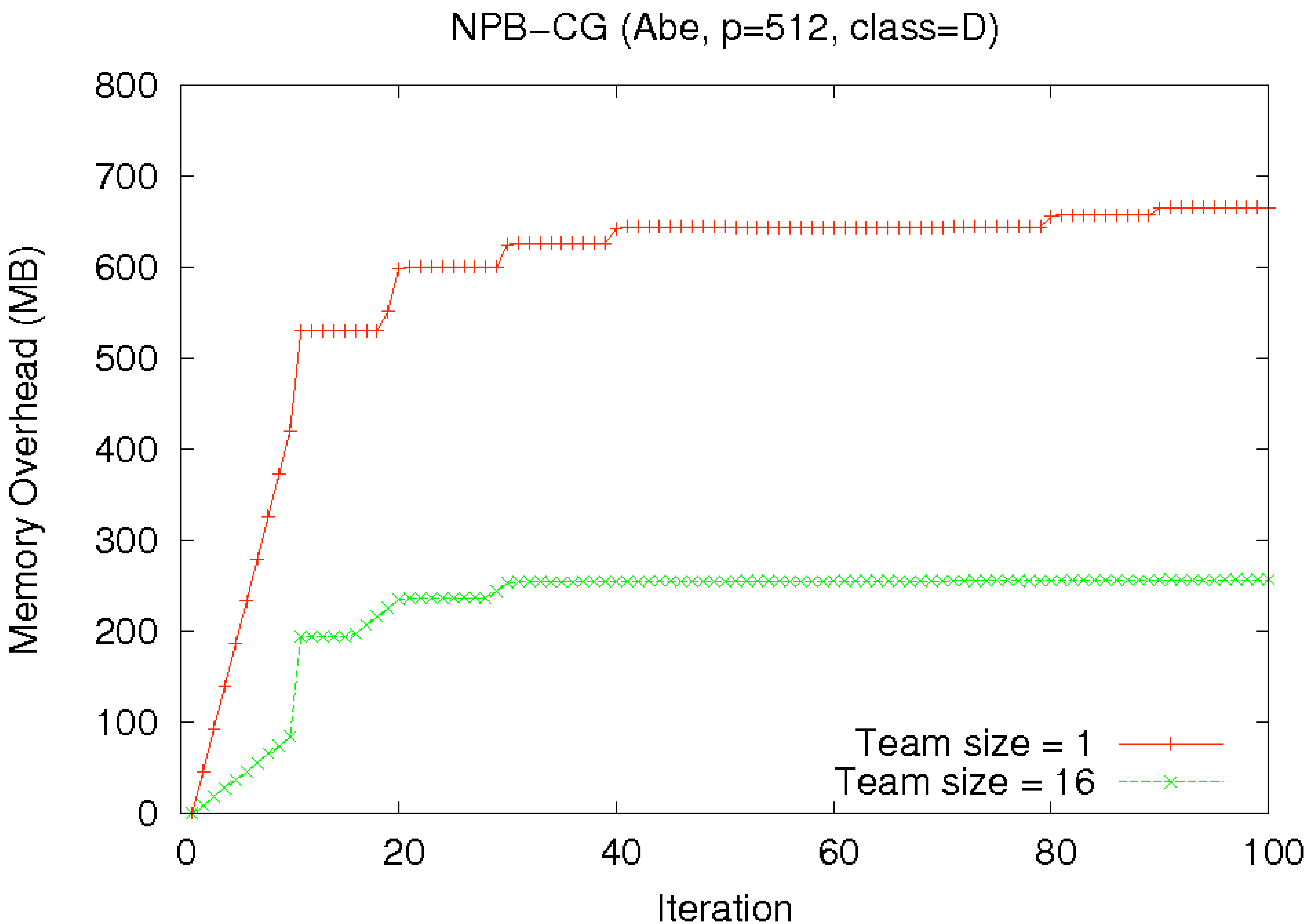
Processor Teams

- Each team acts as a recovery **unit**:
  - All members must checkpoint in a coordinated fashion.
  - If one member fails, the whole team rolls back.

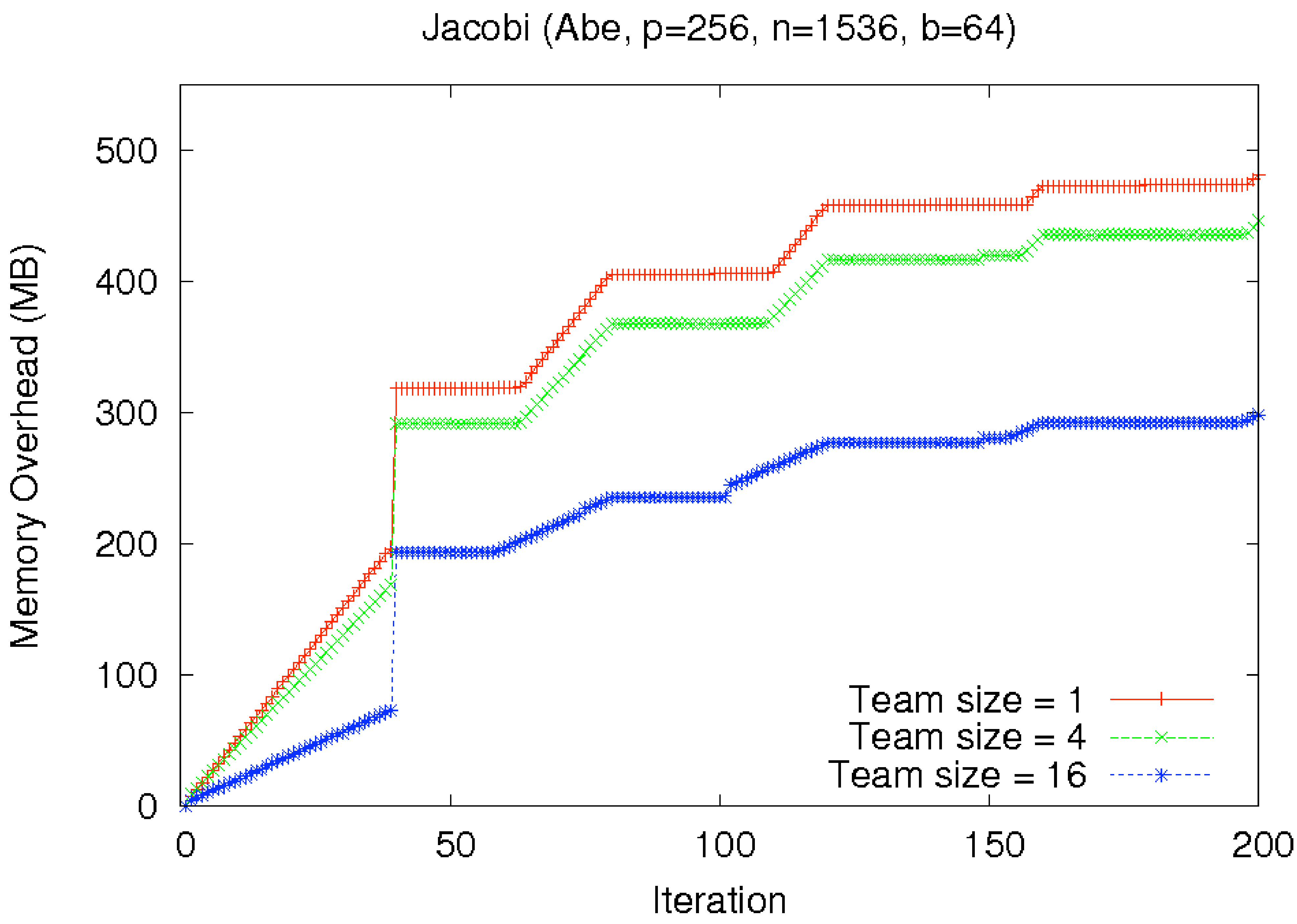


Esteban Meneses, Celso L. Mendes and Laxmikant V. Kale, **Team-based Message Logging: Preliminary Results**, *3rd Workshop on Resiliency in High Performance Computing (Resilience) in Clusters, Clouds, and Grids (CCGRID 2010)*

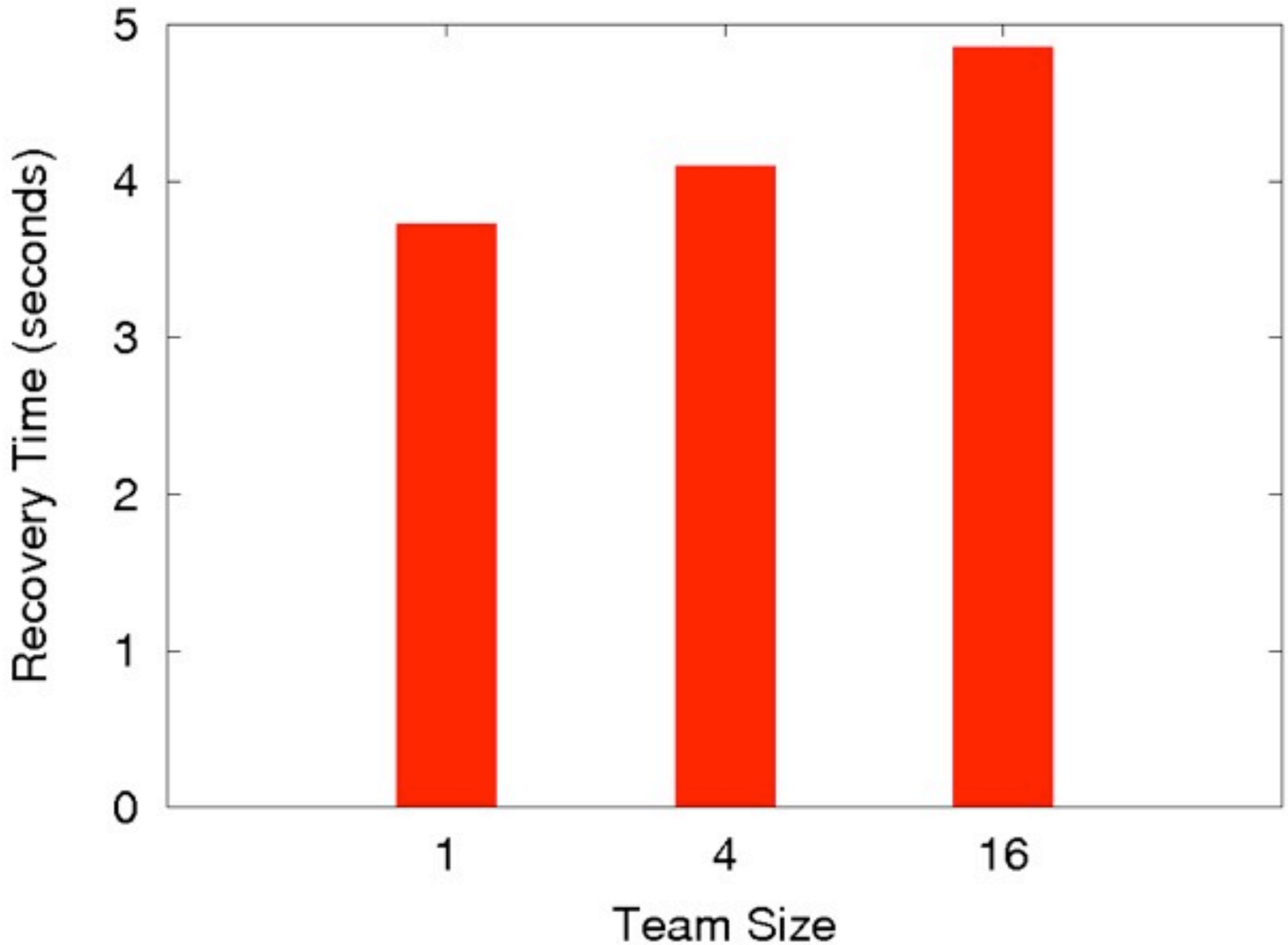
Results


NPB−CG (Abe, p=512, class=D)
Memory Overhead (MB)
800
700
600
500
400
300
200
100
0
0
20
40
60
80
100
Iteration
Team size = 1
Team size = 16

Results (cont.)



Recovery Time

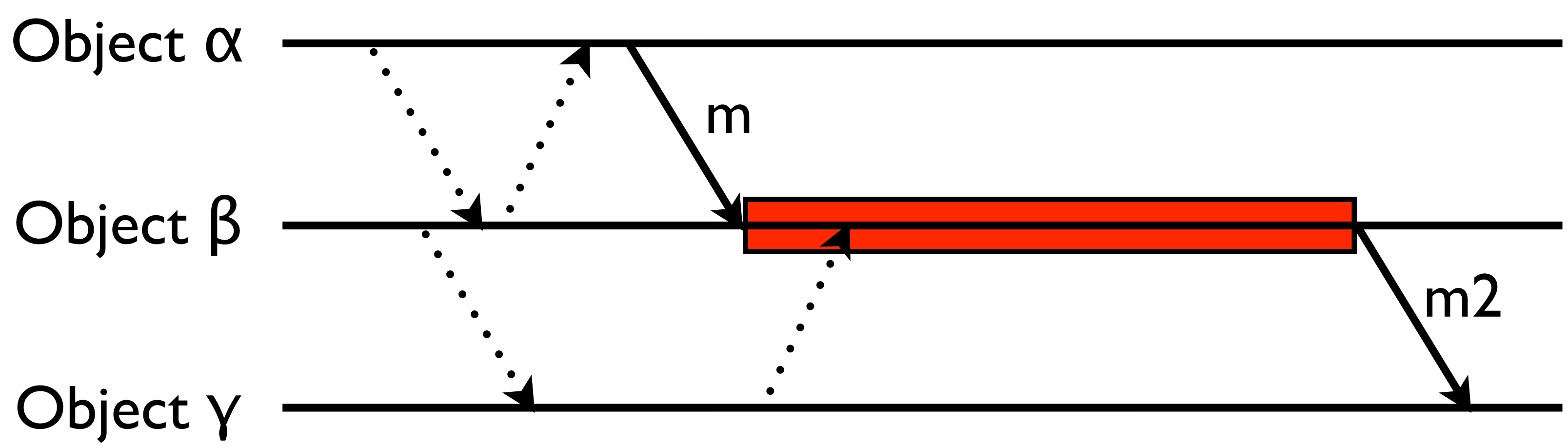


Further Developments

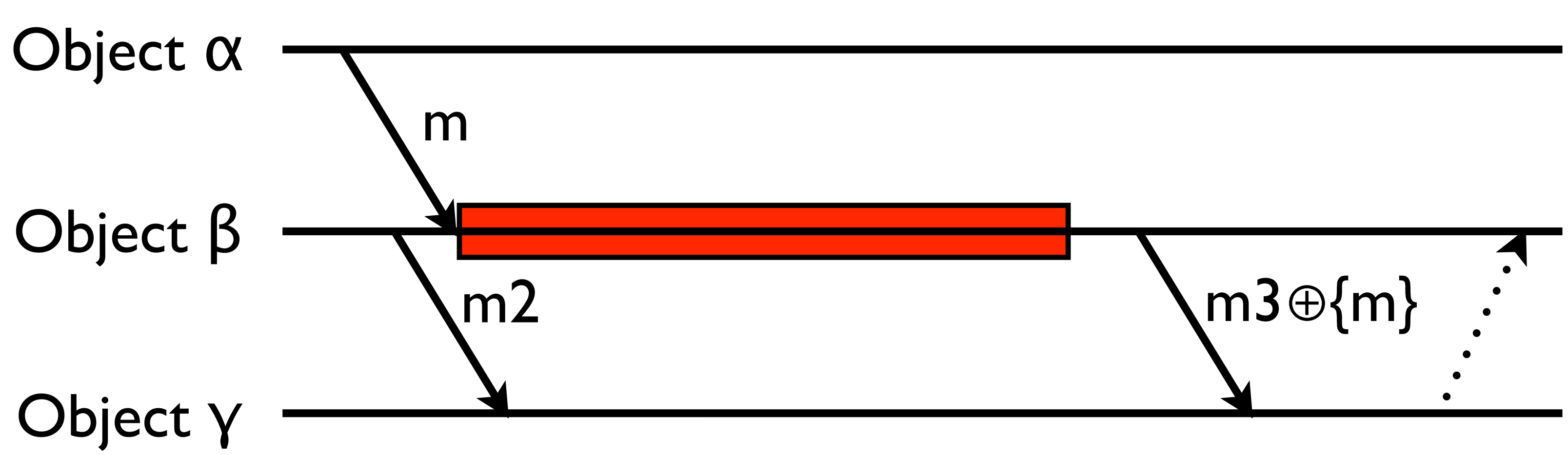
- Highly **connected** objects should belong to the same team.
  - Exploit communication graph, dynamic groups, team-aware load balancer.
- Teams can address some **correlated** failures.
- Applicable to other message-logging **protocols**.

Reducing Latency

Pessimistic Message Logging



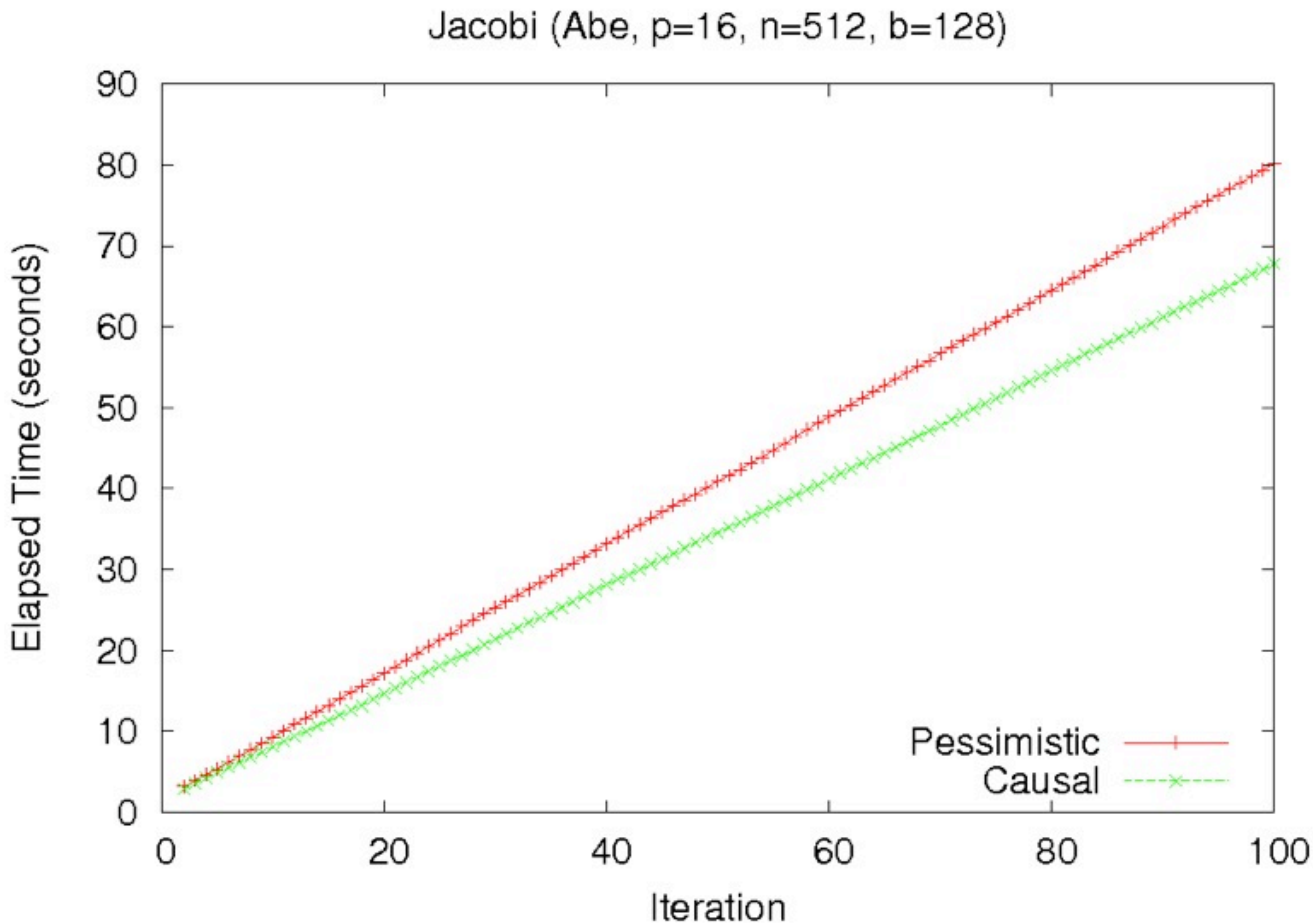
Causal Message Logging



Causal Protocol

- No need to block the **delivery** of a message.
- No need to contact remote processor for a **local** message.
- Metadata is **piggybacked** in application's messages.
- **Recovery** may involve more processors.

Early Results



Future Work

Future Work Roadmap

- Bigger Charm++ **applications**.
- Enhance Proactive Approach with **prediction** schemes.
- Enrich **Team**-based Approach.
  - Smarter team formation.
  - Coupling with load balancer.
- **SMP**-aware fault tolerance.

Acknowledgments

- Department of Energy – FastOS Program.
  - Colony-1 and Colony-2 projects.
- NSF/NCSA
  - Deployment efforts specific for Blue Waters.
- Machine allocation
  - TeraGrid MRAC – NCSA, TACC, ORNL
- Greg Bronevetsky from LLNL.

References

[1] Nathan DeBardeleben, James Laros, John Daly, Stephen Scott, Christian Engelmann and Bill Harrod. **High End Computing Resilience: Analysis of Issues Facing the HEC Community and Path-Forward for Research and Development**.

Q&A

Thank You!