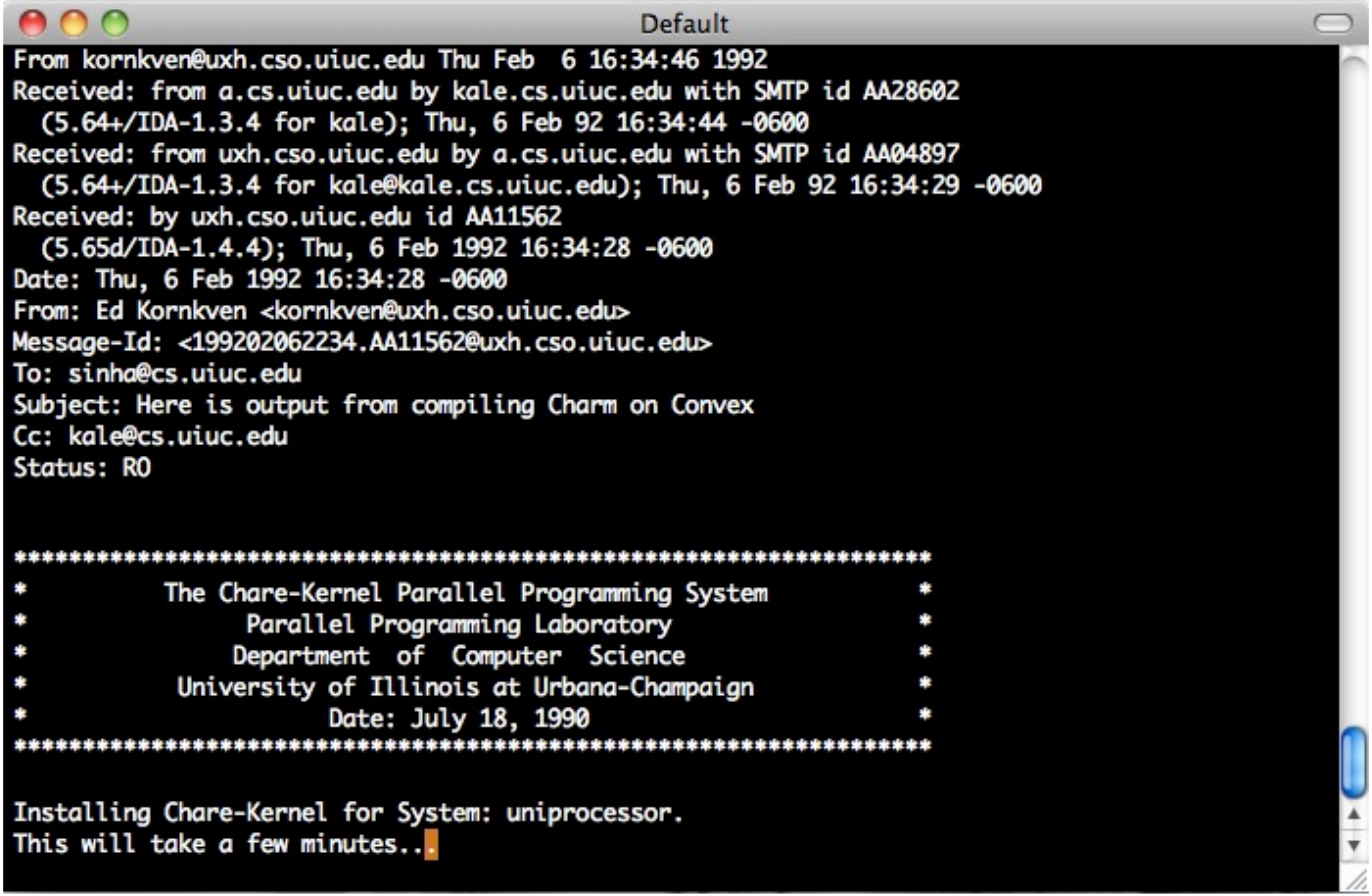
State of Charm++

Laxmikant V. Kale

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

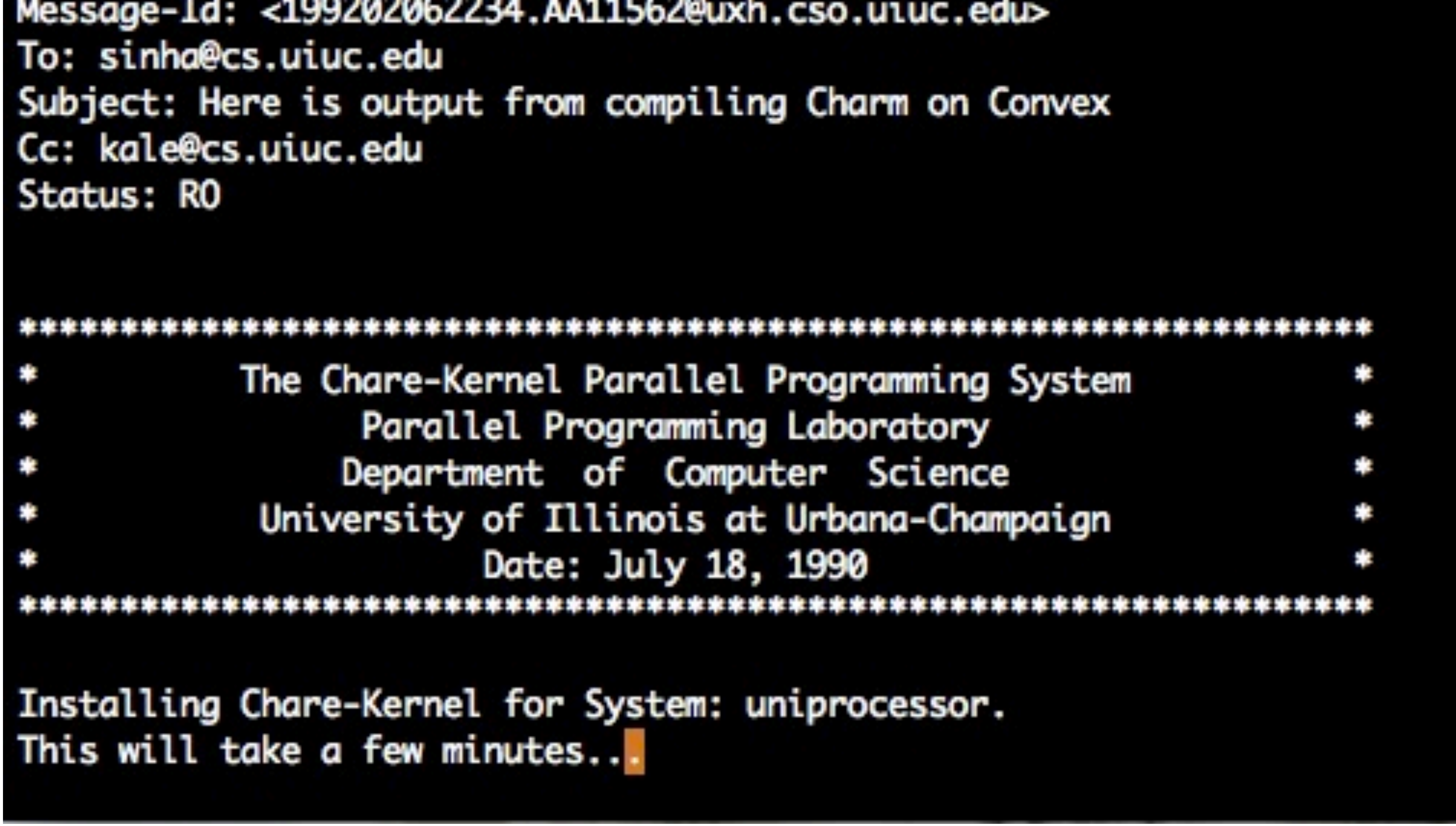
PARALLEL PROGRAMMING LAB
DEPT. OF COMPUTER SCIENCE, UNIVERSITY OF ILLINOIS



```
From kornkven@uxh.cso.uiuc.edu Thu Feb  6 16:34:46 1992
Received: from a.cs.uiuc.edu by kale.cs.uiuc.edu with SMTP id AA28602
  (5.64+/IDA-1.3.4 for kale); Thu, 6 Feb 92 16:34:44 -0600
Received: from uxh.cso.uiuc.edu by a.cs.uiuc.edu with SMTP id AA04897
  (5.64+/IDA-1.3.4 for kale@kale.cs.uiuc.edu); Thu, 6 Feb 92 16:34:29 -0600
Received: by uxh.cso.uiuc.edu id AA11562
  (5.65d/IDA-1.4.4); Thu, 6 Feb 1992 16:34:28 -0600
Date: Thu, 6 Feb 1992 16:34:28 -0600
From: Ed Kornkven <kornkven@uxh.cso.uiuc.edu>
Message-Id: <199202062234.AA11562@uxh.cso.uiuc.edu>
To: sinha@cs.uiuc.edu
Subject: Here is output from compiling Charm on Convex
Cc: kale@cs.uiuc.edu
Status: RO


*******************************************************************
*          The Chare-Kernel Parallel Programming System           *
*                Parallel Programming Laboratory                  *
*               Department  of  Computer  Science                 *
*           University of Illinois at Urbana-Champaign            *
*                      Date: July 18, 1990                        *
*******************************************************************

Installing Chare-Kernel for System: uniprocessor.
This will take a few minutes...
```



```
Message-Id: <199202062234.AA11562@uxh.cso.uiuc.edu>
To: sinha@cs.uiuc.edu
Subject: Here is output from compiling Charm on Convex
Cc: kale@cs.uiuc.edu
Status: RO



*******************************************************************
*          The Chare-Kernel Parallel Programming System          *
*                Parallel Programming Laboratory                  *
*               Department  of  Computer  Science                 *
*           University of Illinois at Urbana-Champaign            *
*                      Date: July 18, 1990                        *
*******************************************************************

Installing Chare-Kernel for System: uniprocessor.
This will take a few minutes...
```

Charm++ is a robust system

- Release 6.3 last month.
  - 19+ major releases over 20 years.
- Autobuild
  - Build and test every night
  - Local machines, major supercomputers, and NMI
  - 300 functional tests, Over 50 system configurations
- Comprehensive set of tools:
  - Projections: Performance visualization
  - Debuggers: freeze-and-inspect, record-replay, ..

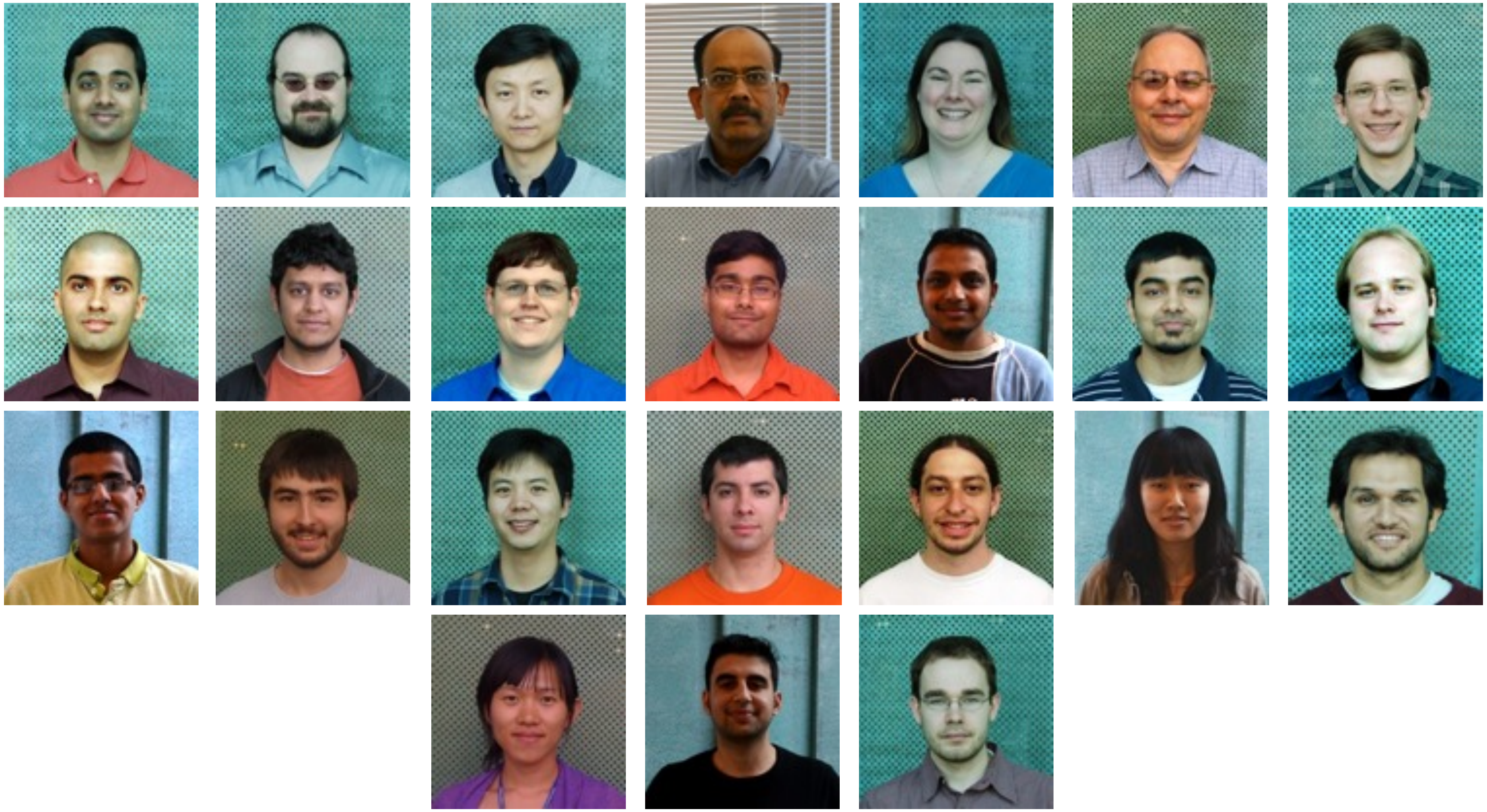
Some statistics for Charm++

Popularity
- 2900 distinct direct source downloads
- 1400 distinct direct binary downloads
- 40,000+ downloads for NAMD

Lines of code:
350,000 lines in Charm++ itself,
~30,000 in Projections,
~10,000 : the debugger.

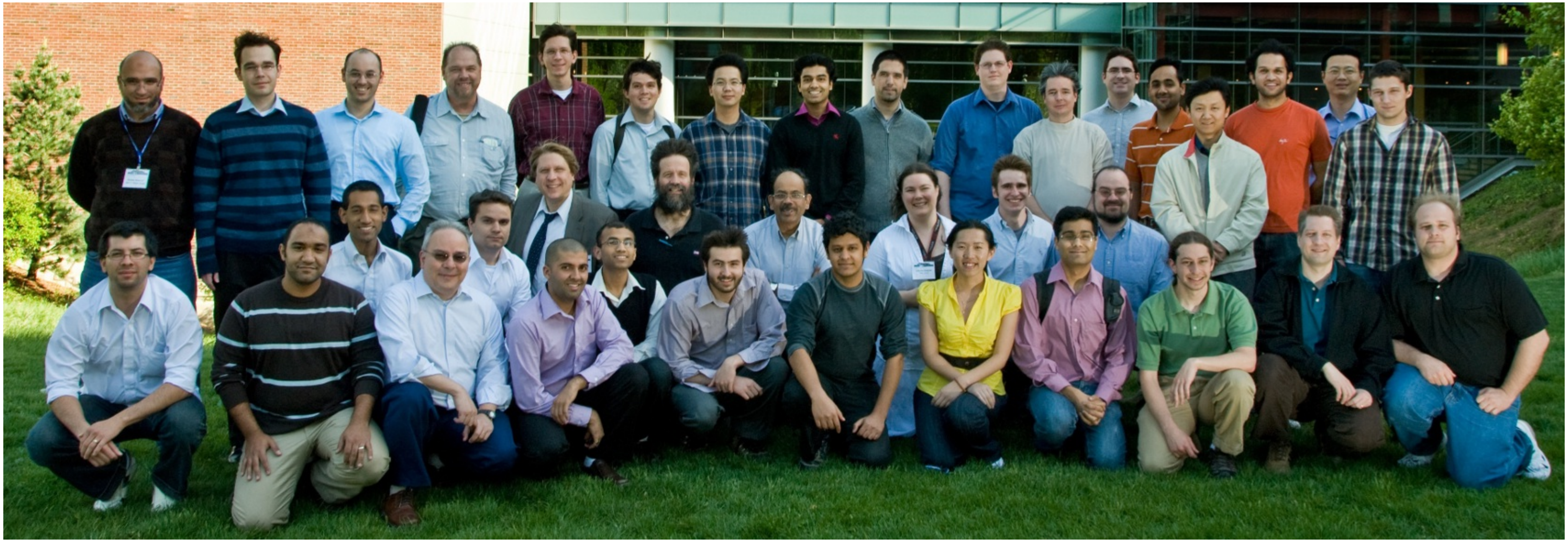
generated using David A.Wheeler's 'SLOCCount'.

Parallel Programming Laboratory



Parallel Programming Laboratory

PPL with Collaborators



A glance at history

- 1987: Chare kernel arose from parallel Prolog work
    - Dynamic load balancing for state-space search, Prolog, ...
- 1992: Charm++
- 1994: position paper on application oriented yet CS centered research
    - NAMD: 1994, 1996
- Charm++ in almost current form: 1996-1998
    - Chare arrays
    - Measurement-based dynamic load balancing
- 1997: Rocket center: a trigger for AMPI
- 2001: Era of ITRs:
    - Quantum chemistry collaboration: OpenAtom
    - Computational astronomy collaboration: ChaNGa
- 2008: Multicore meets Petaflop/s, Blue Waters
- 2010: Collaborations, BigSim, scalability

PPL Mission and Approach

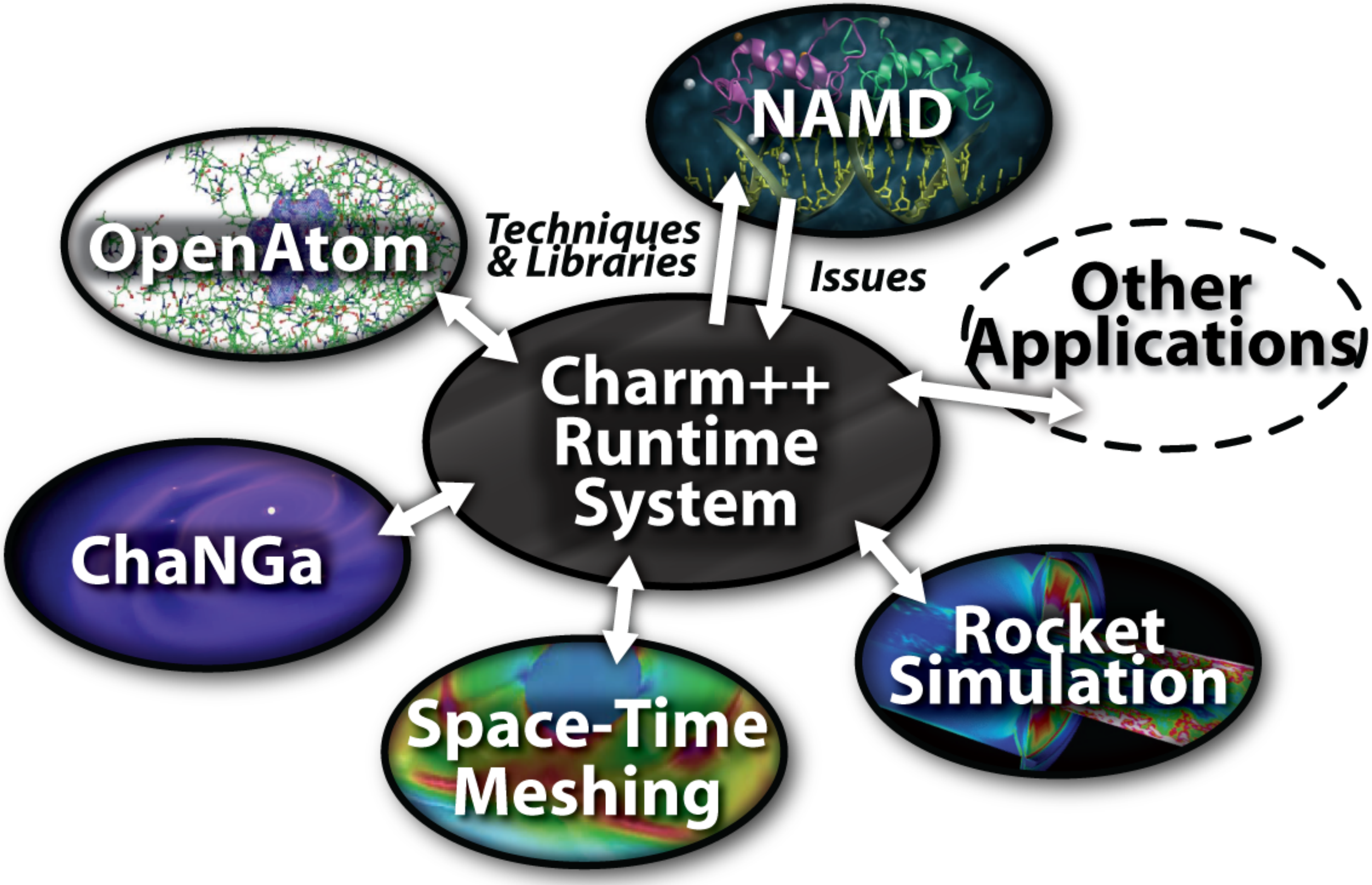
- To enhance performance and productivity in programming complex parallel applications
  - Performance: scalability to hundreds of thousands of processors
  - Productivity: of human programmers
  - Complex: irregular structure, dynamic variations
- Approach: application oriented yet CS centered research
  - Developing enabling technology for a wide collection of apps
  - Develop, use and test it in the context of real applications

Our guiding principles

- No magic: parallelizing compilers have achieved close to technical perfection but are not enough
  - Sequential programs obscure too much information
- Seek an optimal division of labor between the system and the programmer
- Design abstractions based solidly on use-cases

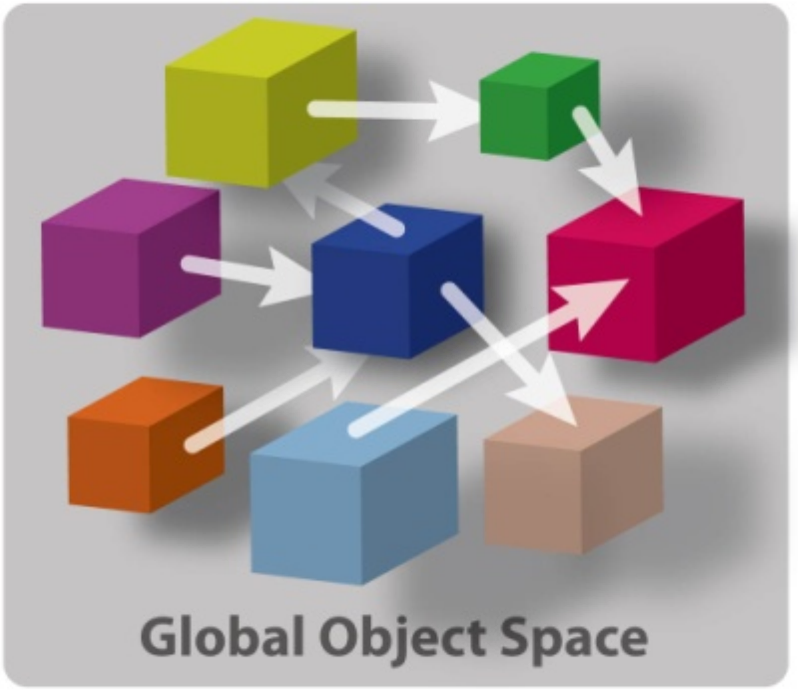
L.V. Kale, "Application Oriented and Computer Science Centered HPCC Research", Developing a Computer Science Agenda for High-Performance Computing, New York, NY, USA, 1994, ACM Press, pp. 98-105.

Charm++ and CSE Applications

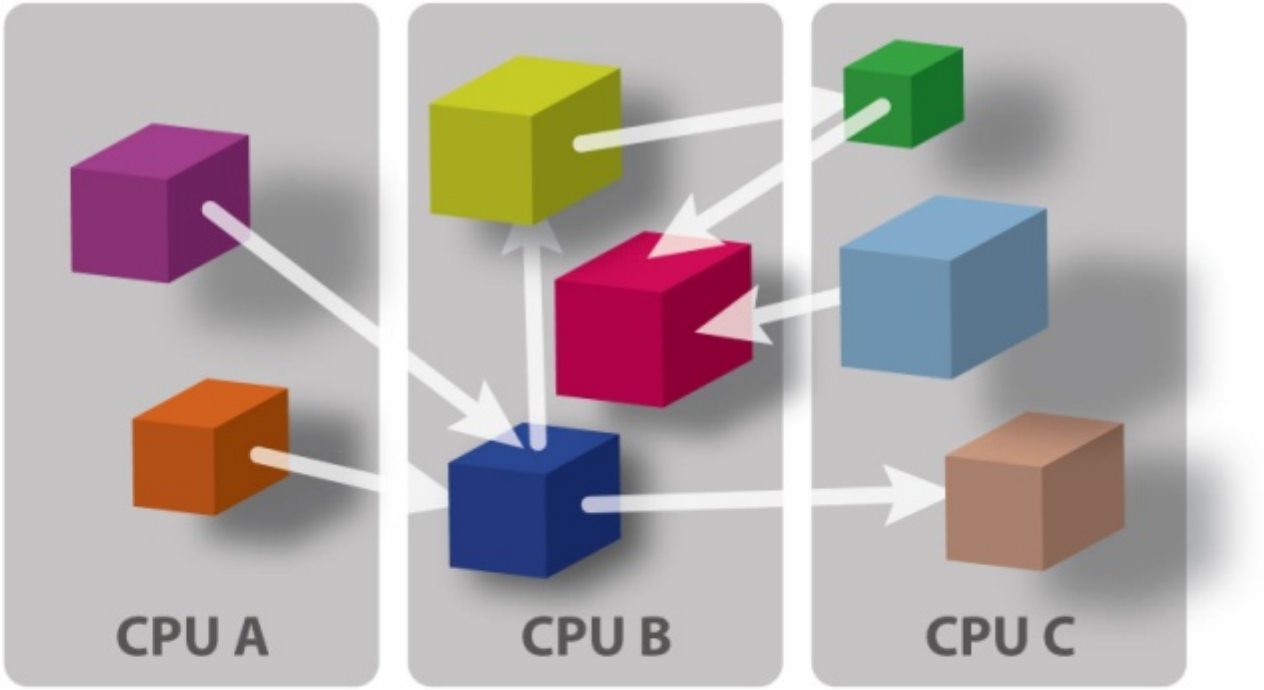


Migratable objects

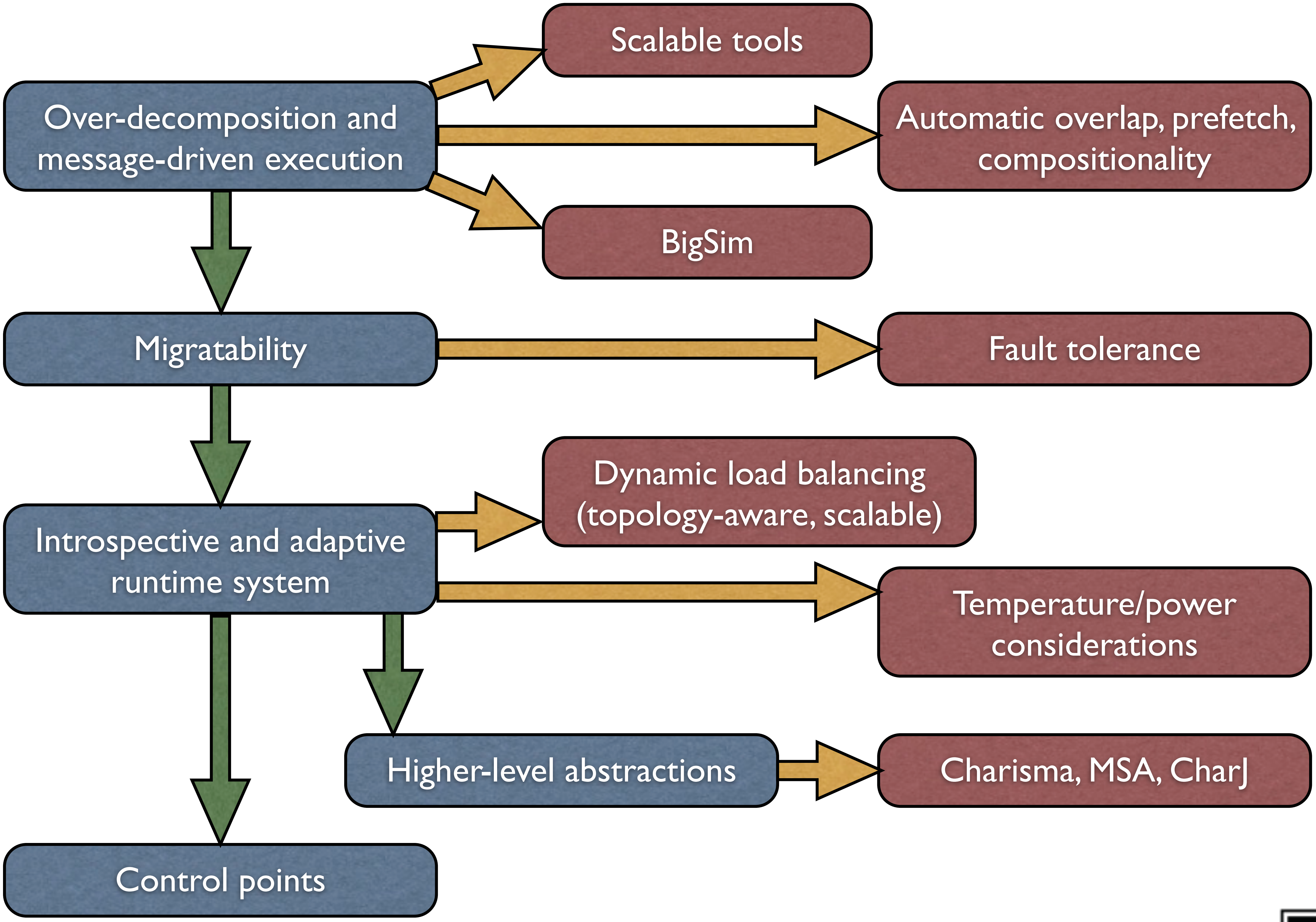
- Programmer: Overdecomposition into virtual processors
- Runtime: Assign VPs to processors
- Enables adaptive runtime strategies
- Implementations: Charm++, AMPI



User view



System view



Over-decomposition and message-driven execution

Scalable tools

Automatic overlap, prefetch, compositionality

BigSim

Migratability

Fault tolerance

Introspective and adaptive runtime system

Dynamic load balancing (topology-aware, scalable)

Temperature/power considerations

Higher-level abstractions

Charisma, MSA, CharJ

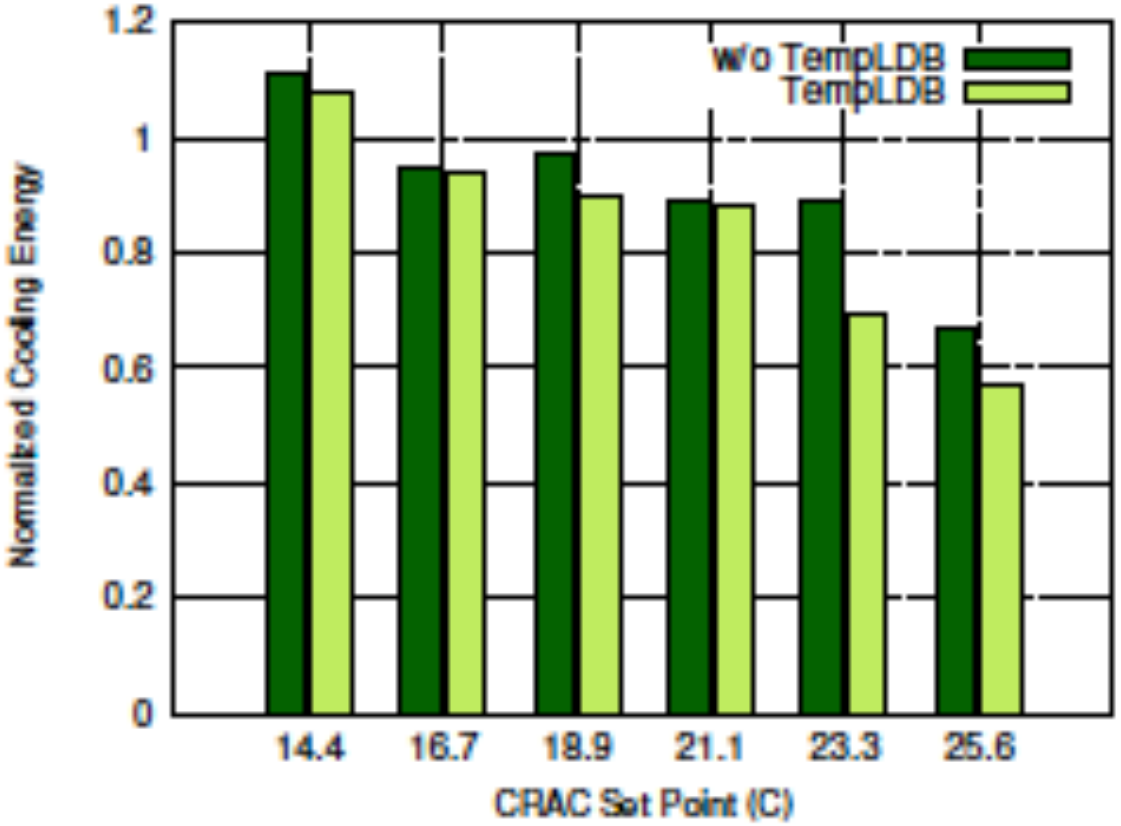
Control points

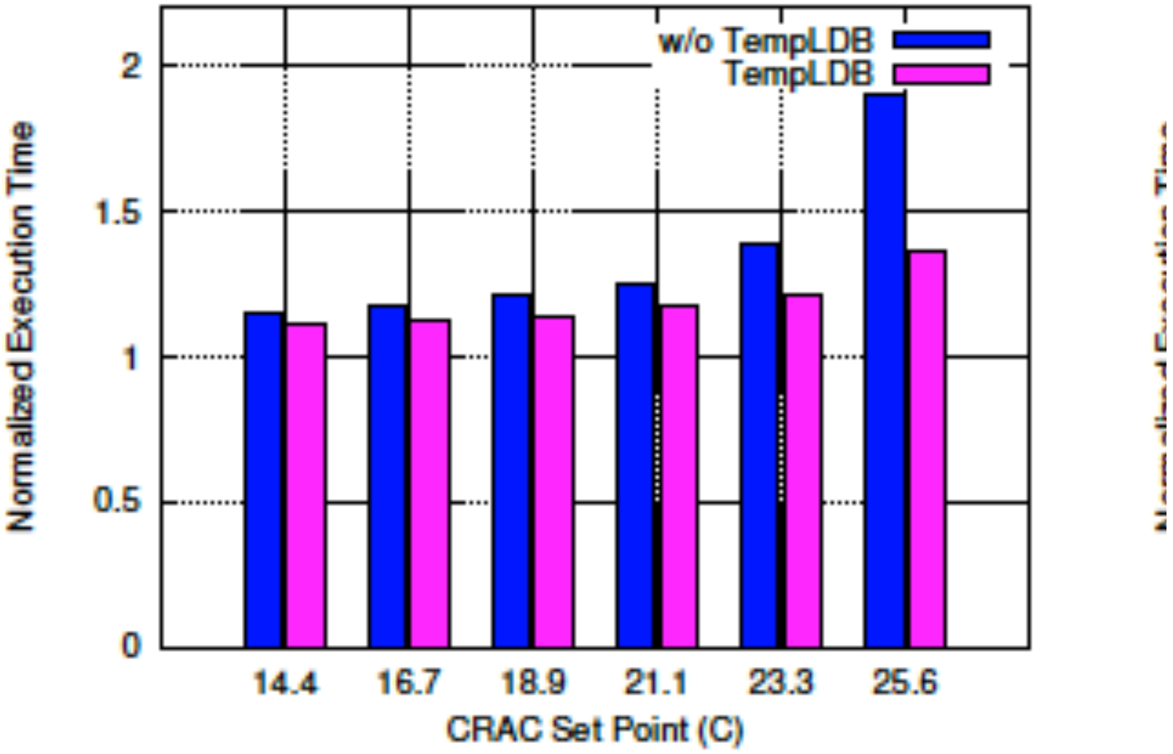
Highlights of recent results

Temperature-aware load balancing

- One objective: to save cooling energy
- Set CRAC thermostat at a higher temperature
- but control core temperature using DVFS
- This leads to load imbalances, but these can be handled via object migrations

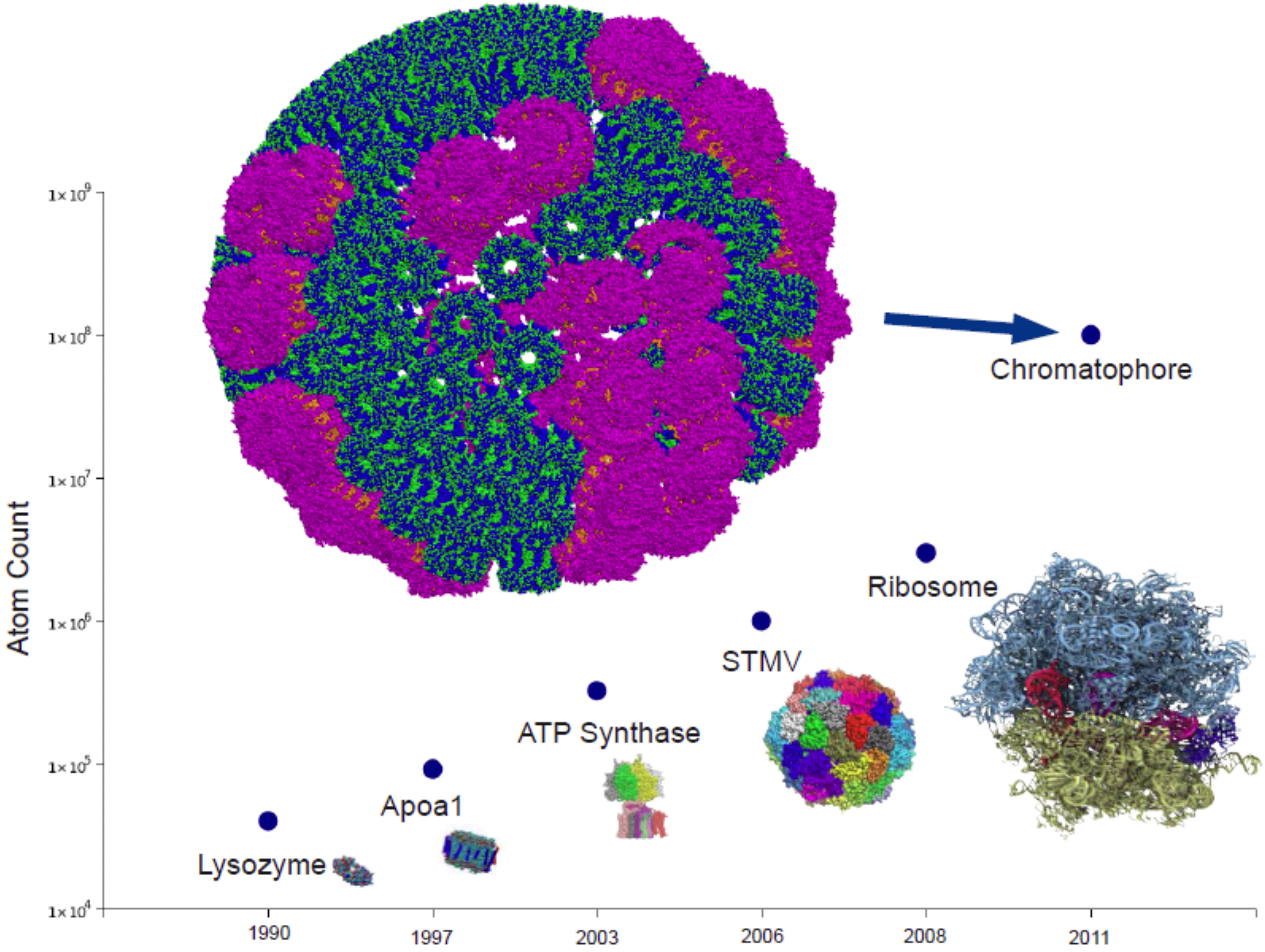
Energy Savings with minimal timing penalty



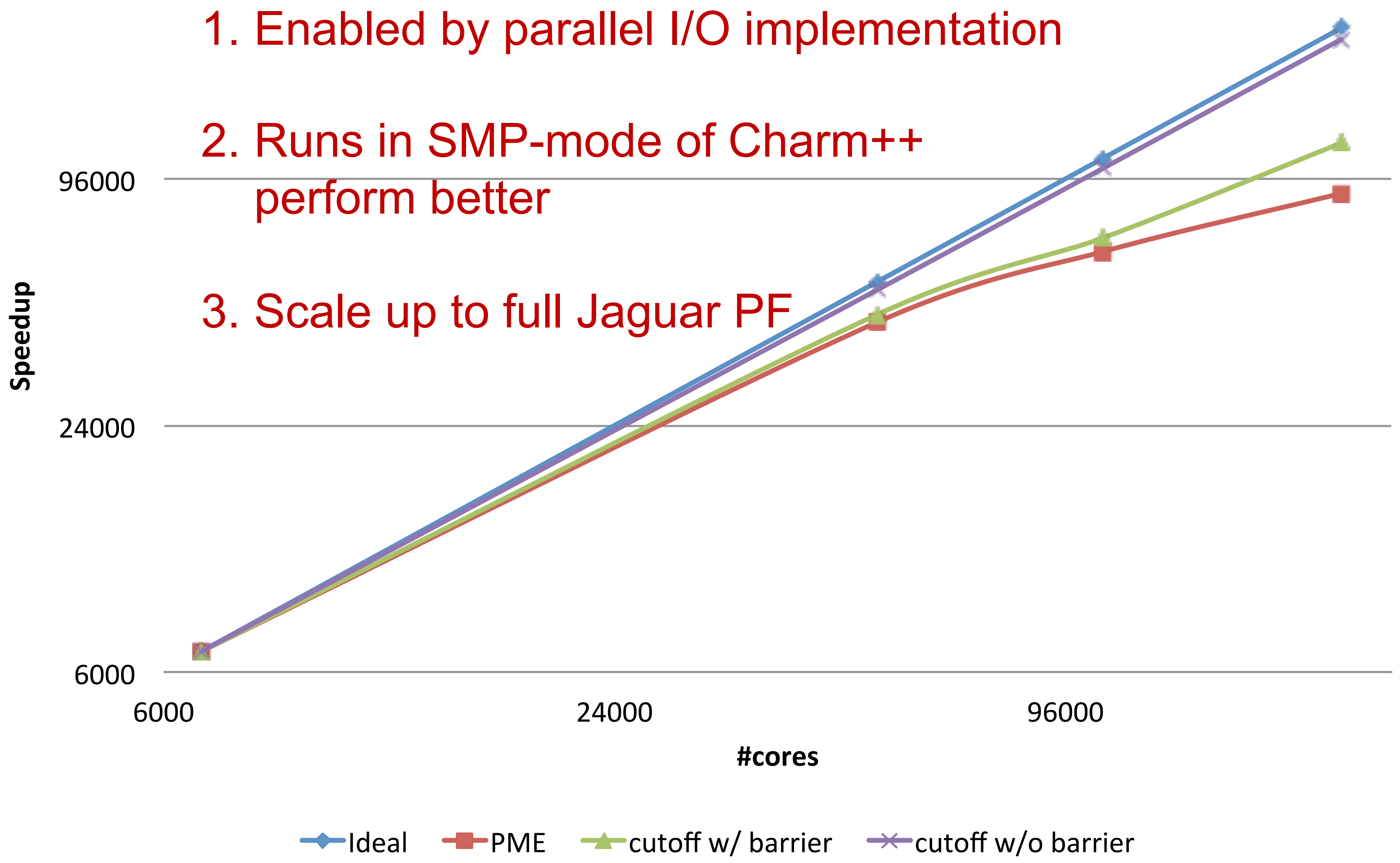


(b) Wave2D

Evolution of Biomolecular System Size

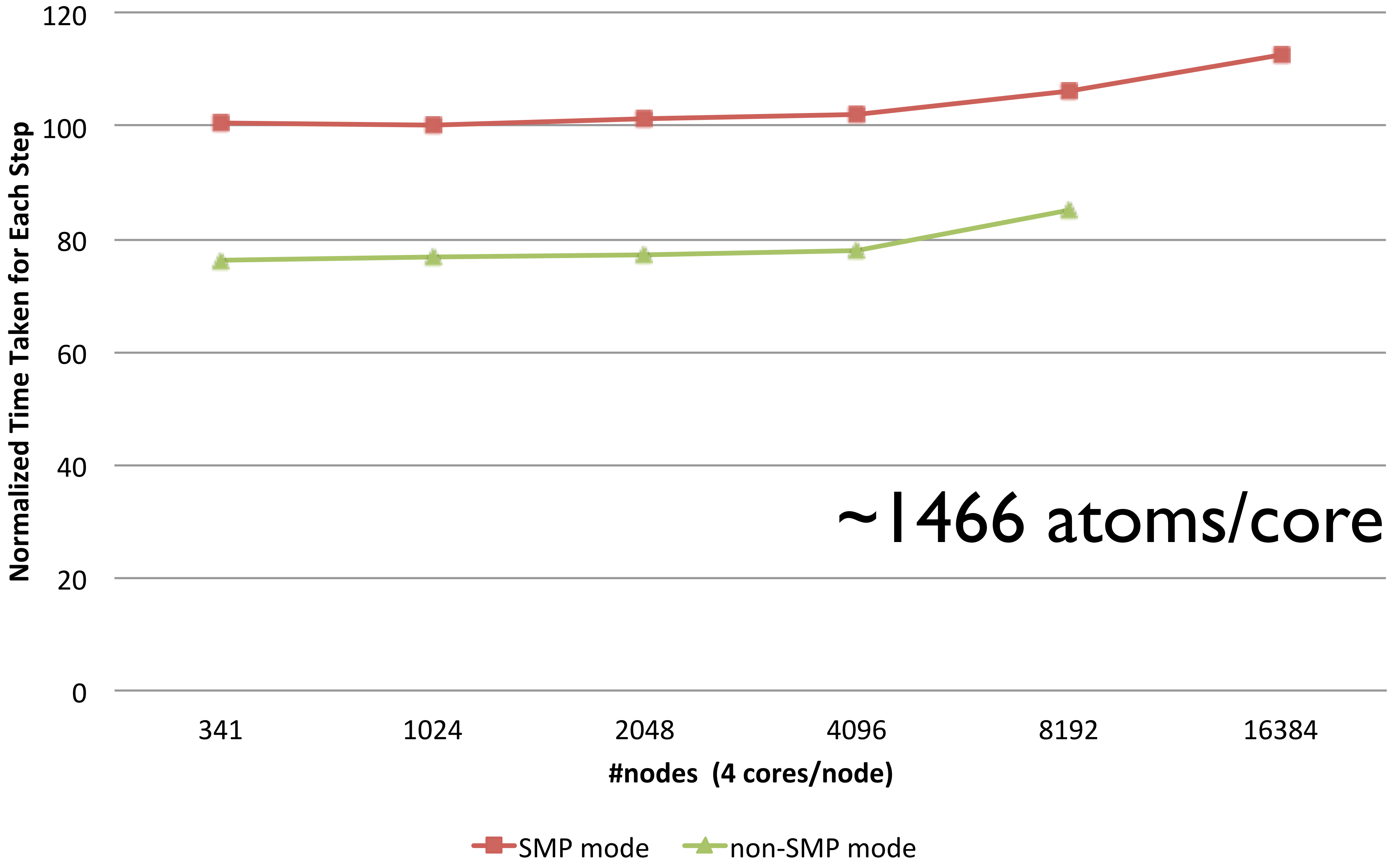


100 million atoms on Jaguar



1. Enabled by parallel I/O implementation
2. Runs in SMP-mode of Charm++ perform better
3. Scale up to full Jaguar PF

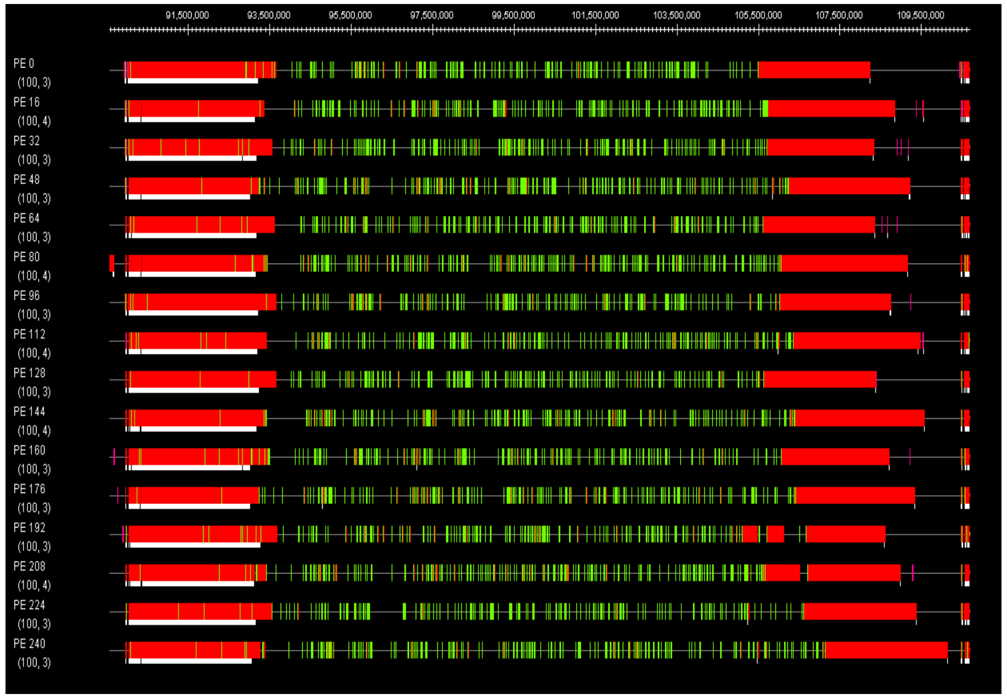
Weak scaling on Intrepid

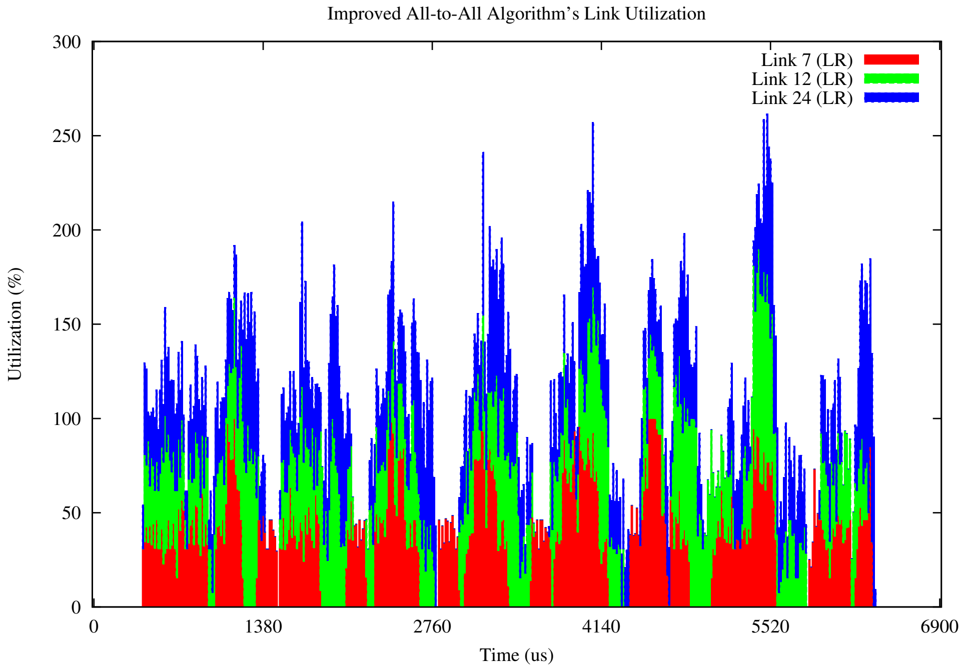


~1466 atoms/core

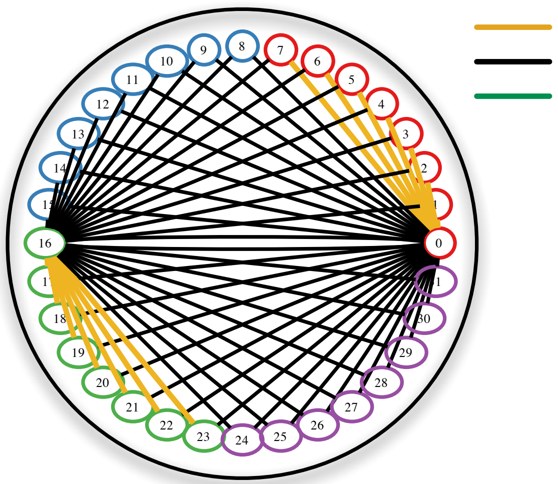
BigSim

- Detailed network model for Blue Waters
- Development to run at large scale efficiently
- Many different outputs
    - Link statistics, projections, user prints …
- Studies: Topology-Mapping, System Noise, Collective Optimization

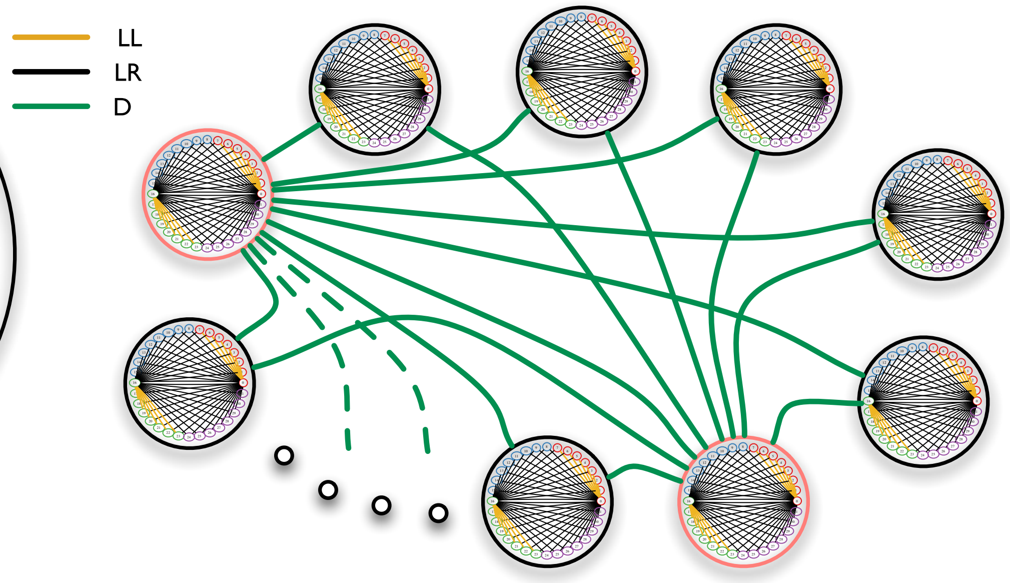




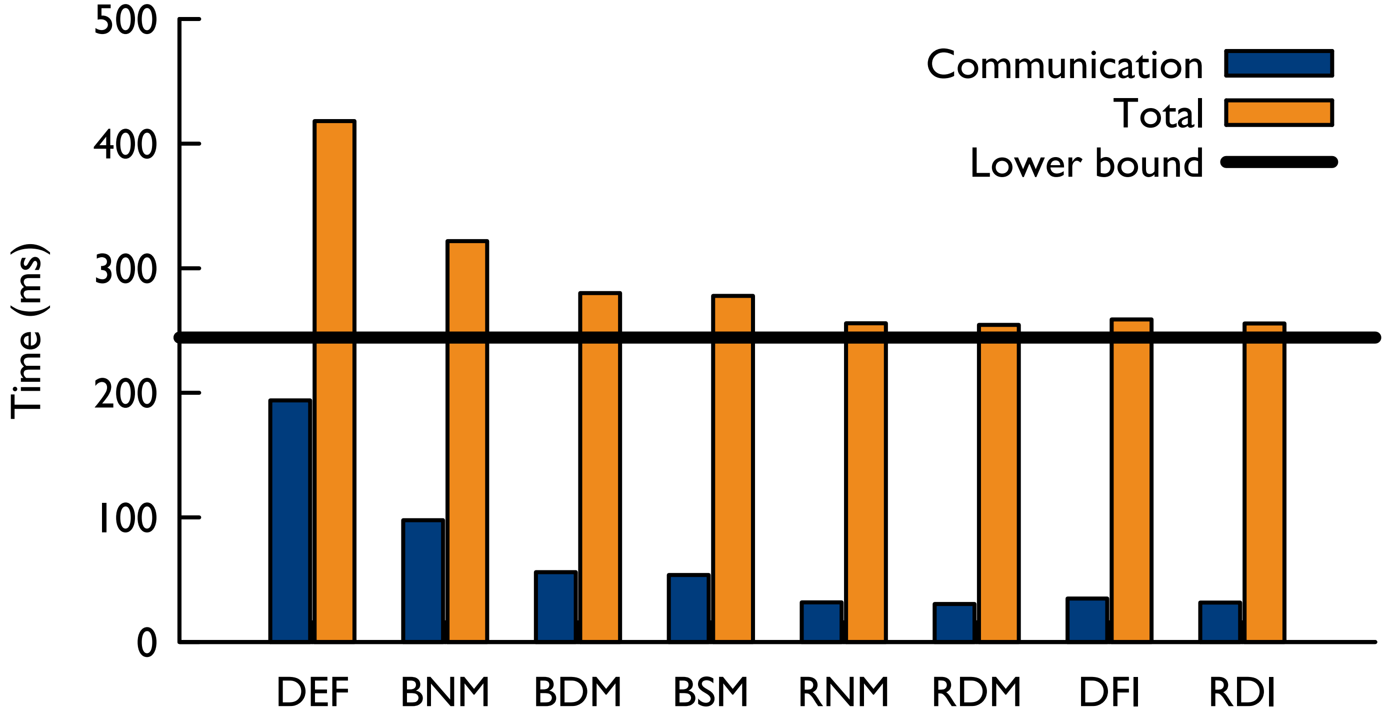
Mapping for Blue Waters



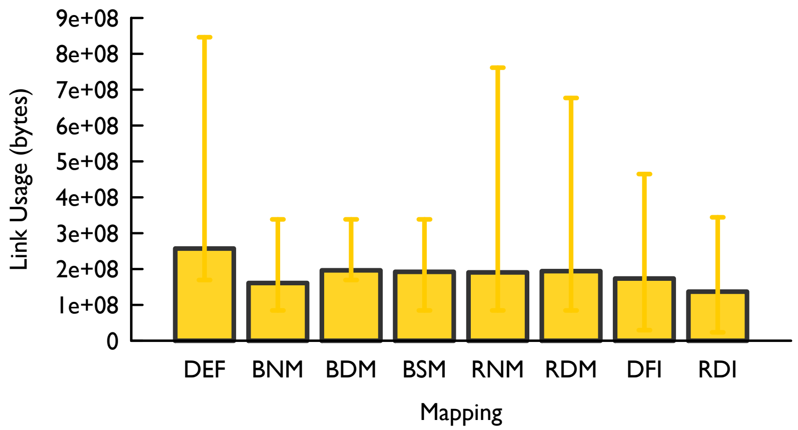
One supernode in the PERCS topology



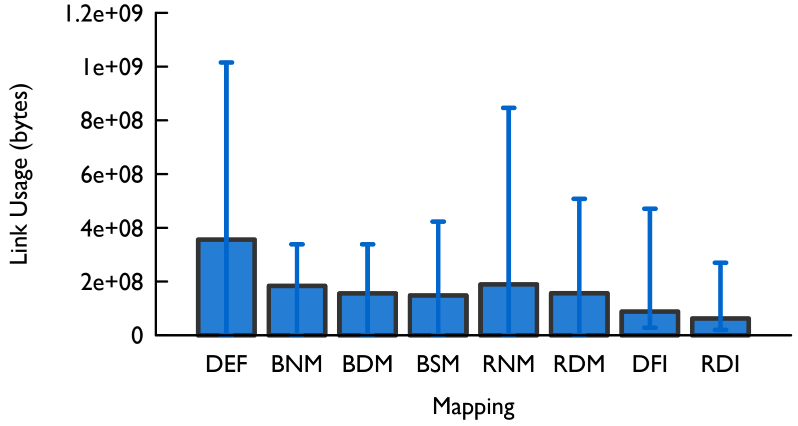
Performance of 4D Stencil on 64 supernodes



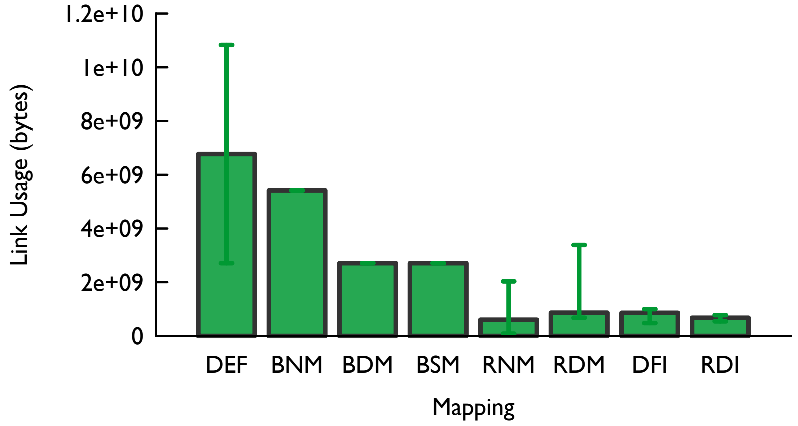
4D Stencil on 64 supernodes (LL links)



4D Stencil on 64 supernodes (LR links)

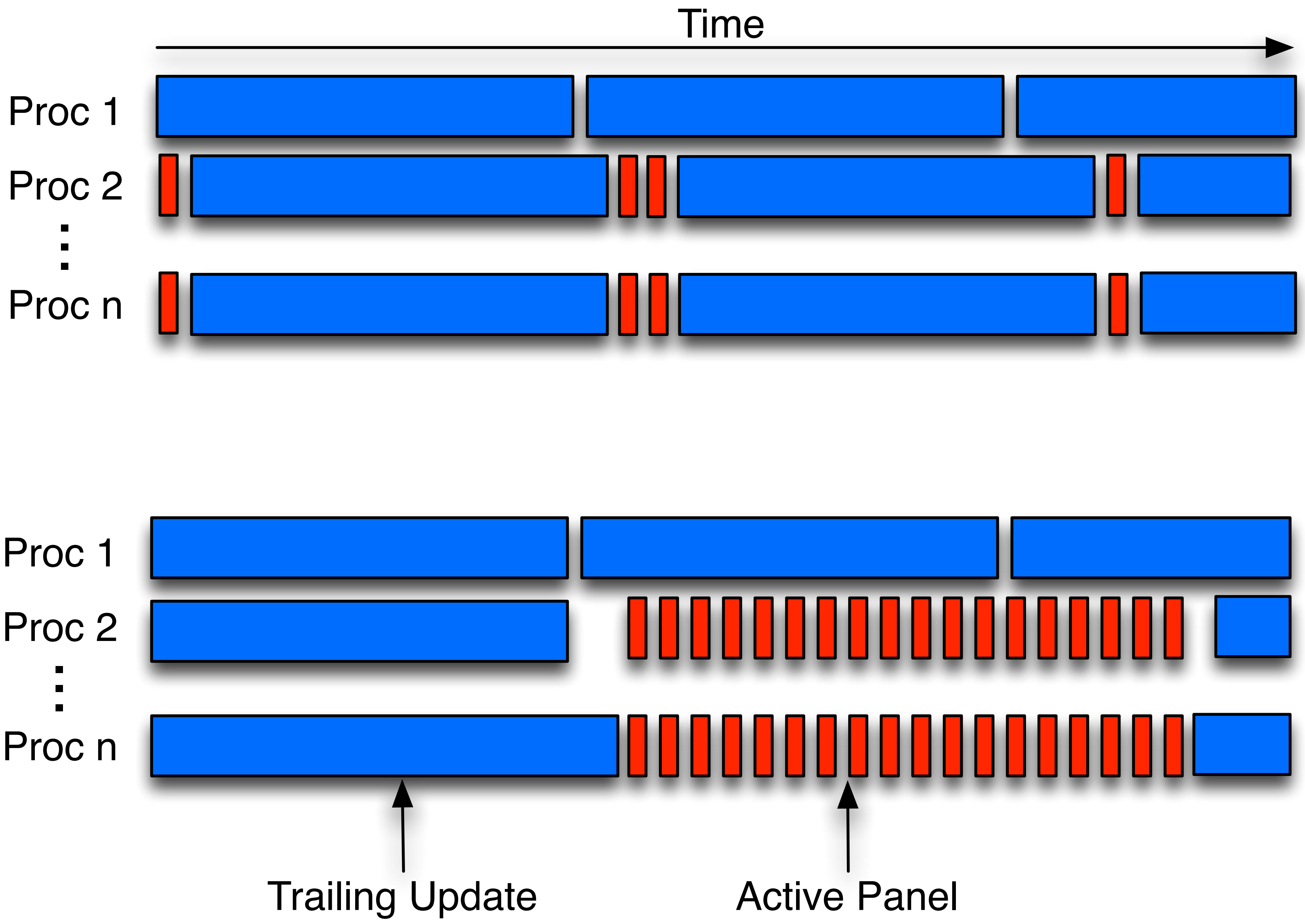


4D Stencil on 64 supernodes (D links)

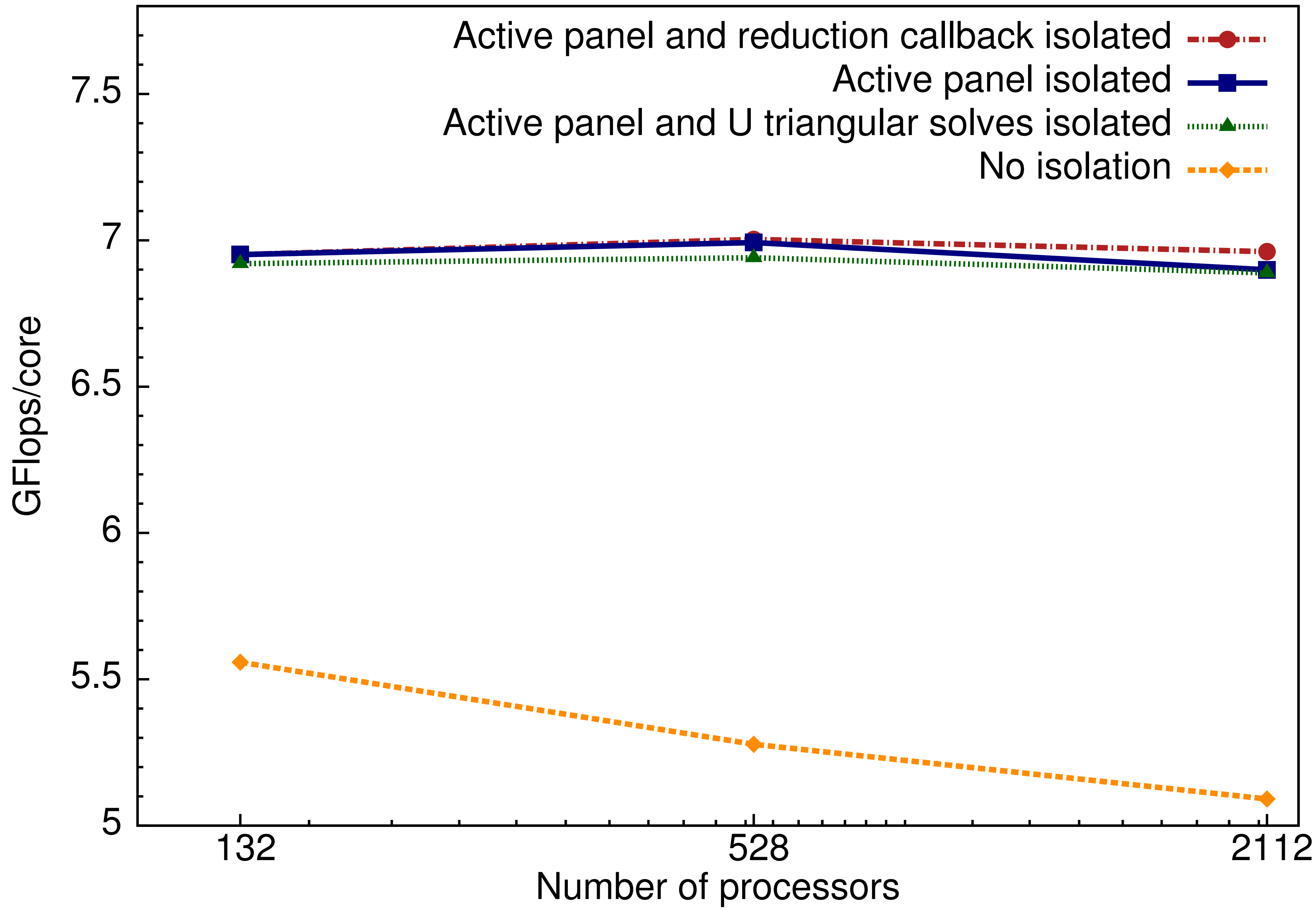


CharmLU Exclusive Scheduling Classes

Balancing Synchrony, Adaptivity, and asynchrony

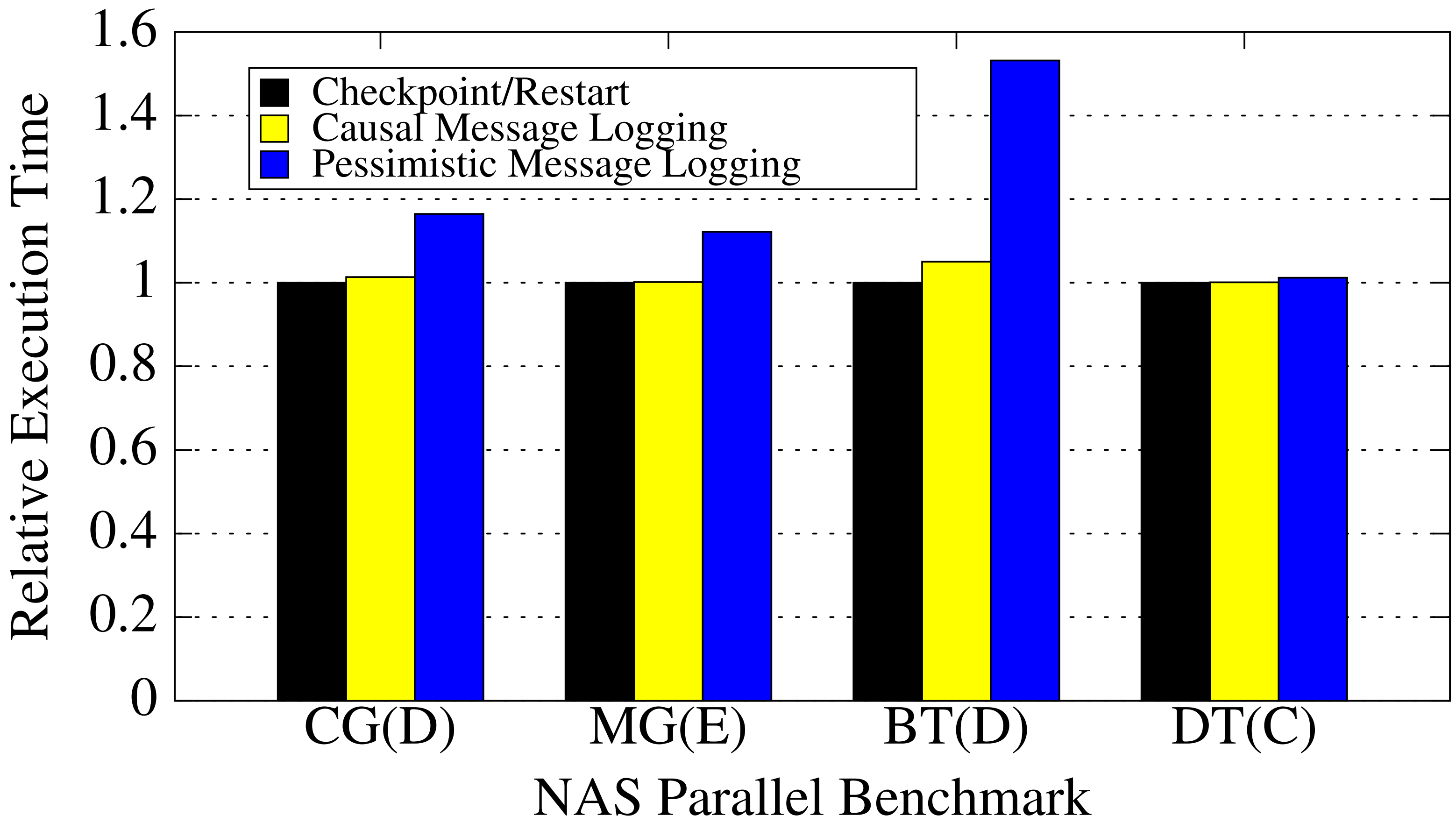


CharmLU Exclusive Scheduling Classes



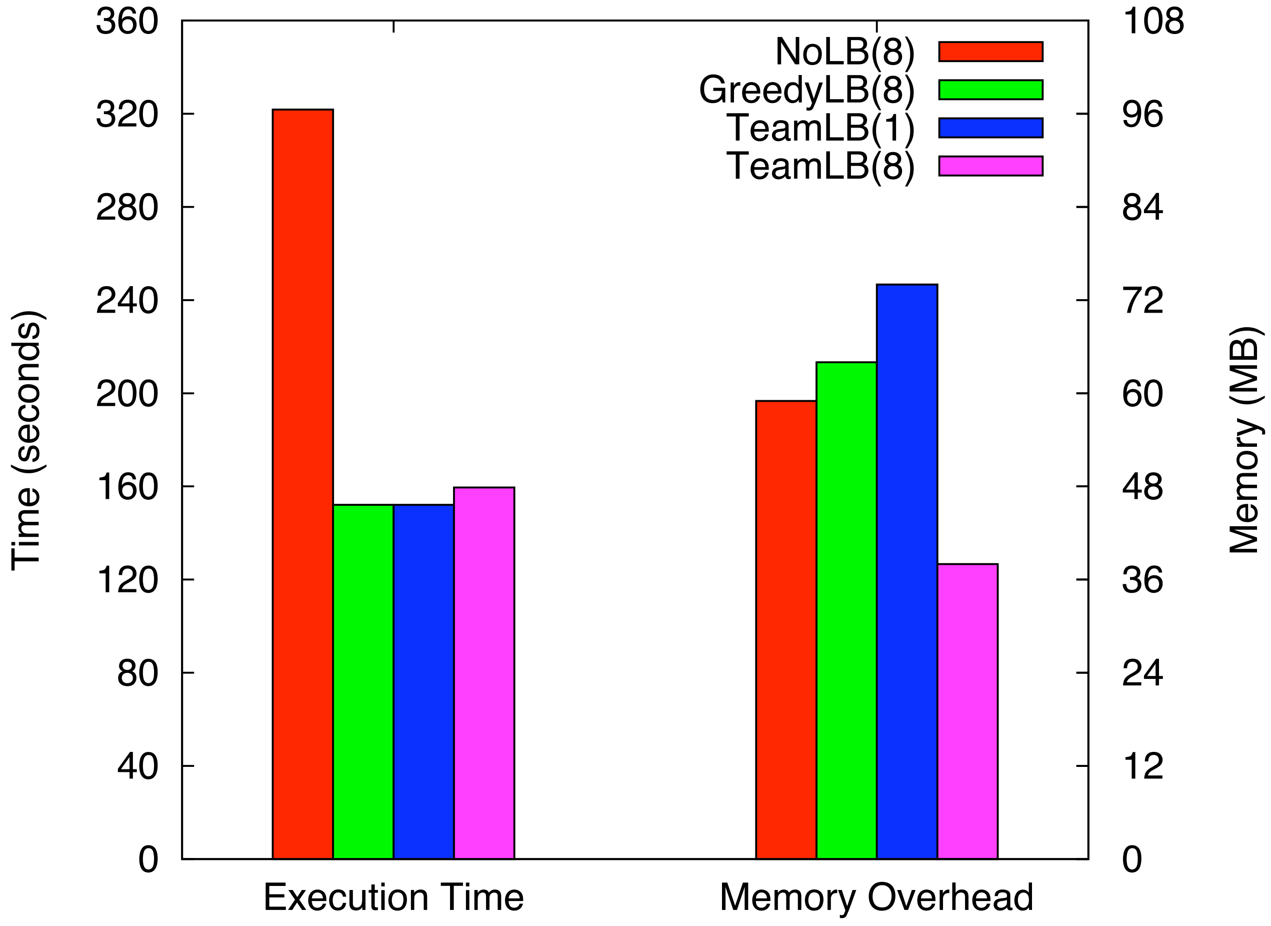
Fault Tolerance: Causal Message Logging

- Consistently better than pessimistic message logging.
- Scaling results up to 1024 cores.



Team-based Load Balancer

- Attains two goals: balance the load and reduce message logging memory overhead.
- Low performance penalty, dramatic memory savings.


Time (seconds)
Memory (MB)
NoLB(8)
GreedyLB(8)
TeamLB(1)
TeamLB(8)
Execution Time
Memory Overhead