Designing Power-Aware Collective Communication Algorithms for InfiniBand Clusters

Krishna Kandalla, Emilio P. Mancini, Sayantan Sur, and Dhabaleswar. K. Panda

Department of Computer Science & Engineering, The Ohio State University

Presented by: Hari Subramoni

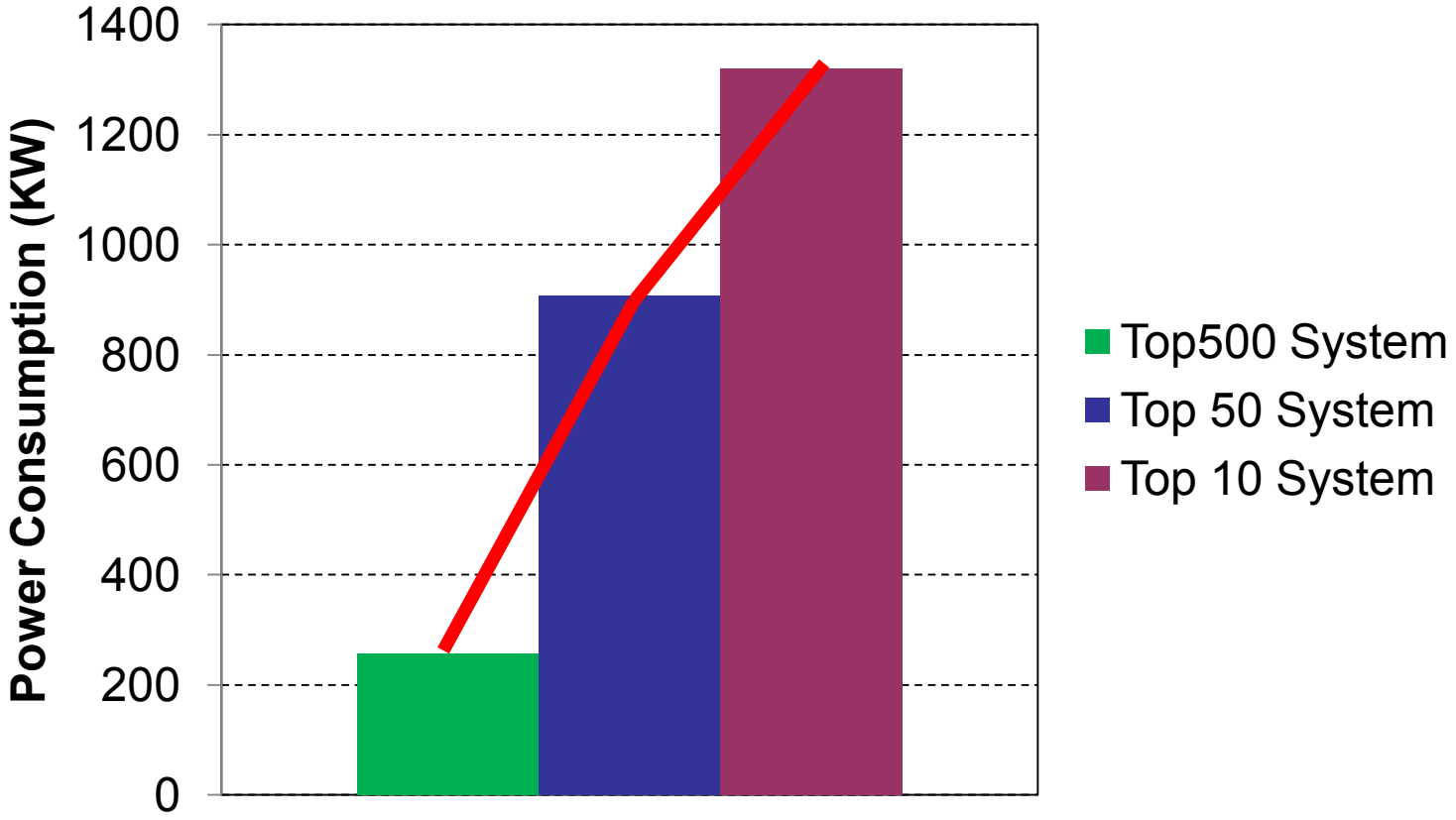
Outline

- Introduction and Background
- Motivation
- Problem Statement
- Designing “Power-Aware” Collective Algorithms
- Experimental Evaluation
- Conclusions and Future Work

Introduction

- Scientific research being driven by multi-cores and high-speed networks
- Supercomputers typically comprise of hundreds of thousands of compute cores
- The power consumed by such systems has also increased sharply
- “Power” is deemed as one of the major challenges in designing next generation exascale systems

Average Power Consumption of Current Generation Supercomputers



- Based on the power measurements of LINPACK benchmark, as reported on the Top500 site

Introduction

- Modern architectures offer fine grained schemes for saving power during idle periods – *Frequency, Voltage Scaling(DVFS)* and *CPU Throttling*

- Power conservation techniques are typically associated with a "delay" leading to performance overheads

- **Broad Challenge:** *Is it possible to design software and middleware stacks in a power-aware manner to minimize the overall system power consumption, with negligible performance overheads?*

Power Saving Innovations in Current Generation Processors and Networks

- Processors (Intel “Nehalem”)
  - DVFS
    - Dynamically scale the Frequency and Voltage of the CPU
    - Core-level DVFS
    - CPU Frequency range [1.6 – 2.4] GHz
  - CPU Throttling
    - Insert brief idle periods to save power
    - Socket-level CPU Throttling
    - Multiple Throttling states (T0 – T7)
      - T0 - 100% CPU activity
      - T7 - 12% CPU activity
- Networks (InfiniBand)
  - Allow offloading significant parts of communication
  - Offer “polling” and “blocking” modes of message progression

State-Of-The-Art in Power Conservation Techniques

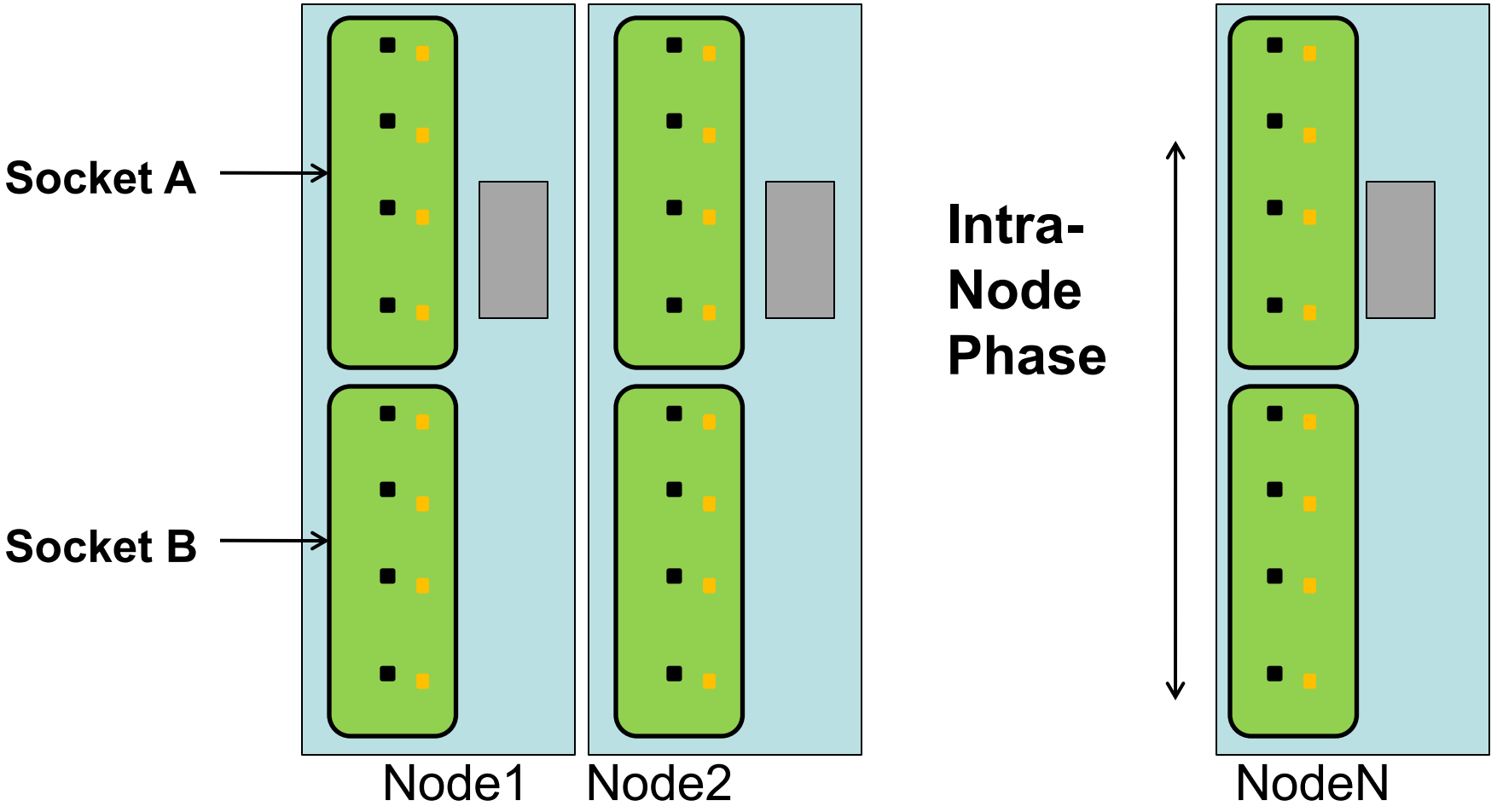
- Where can we conserve power?
  - *Communication phases*

- Cameron et al, have proposed and demonstrated the utility of *PowerPack*
- Lowenthal et al - dynamically detect communication phases and scale the CPU frequency to save power
- Liu et al, studied power consumption with RDMA based networks comparing “blocking” and “polling” modes

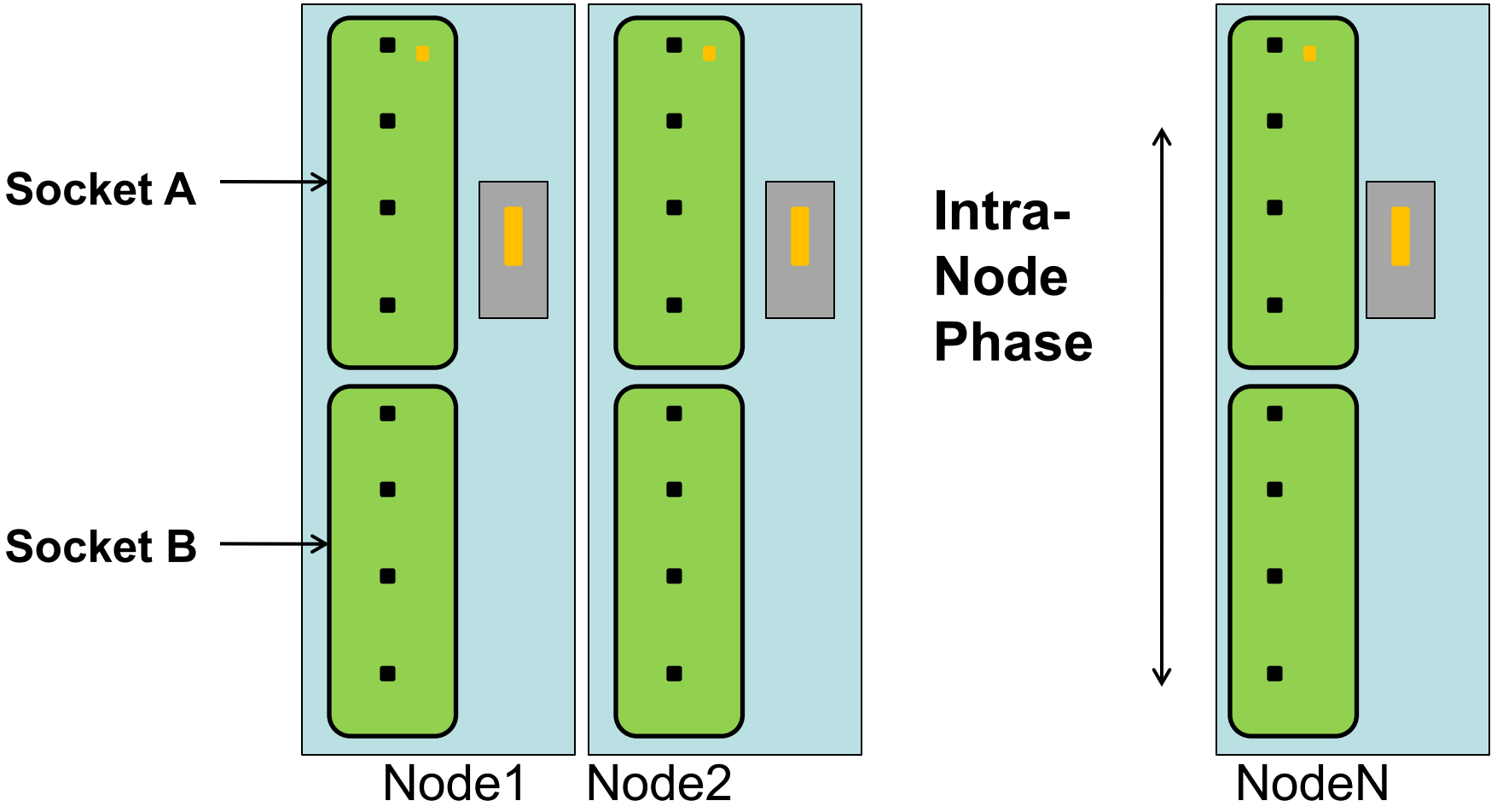
Collective Communication in MVAPICH2

- Scientific parallel applications spend a considerable amount of time in collective communication operations
- Multi-core aware and network topology-aware algorithms optimize the communication costs
- Current power saving methods treat communication phases as a "*black-box*" and use DVFS
- *Is it possible to re-design collective communication algorithms to deliver fine-grained power savings with very little performance overheads?*

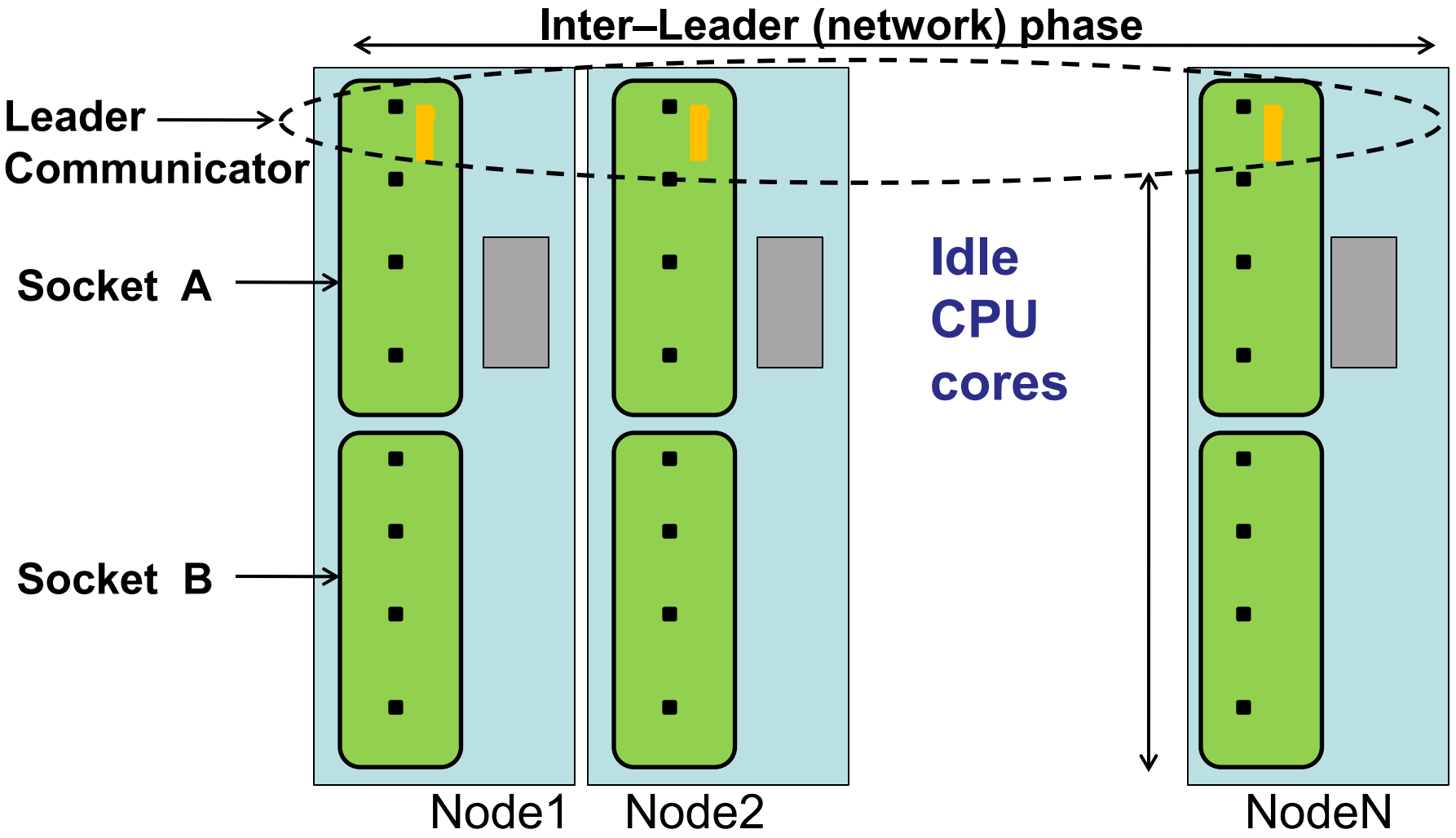
Multi-Core Aware Shared-Memory Based Collective Algorithms in MVAPICH2



Multi-Core Aware Shared-Memory Based Collective Algorithms in MVAPICH2



Multi-Core Aware Shared-Memory Based Collective Algorithms in MVAPICH2



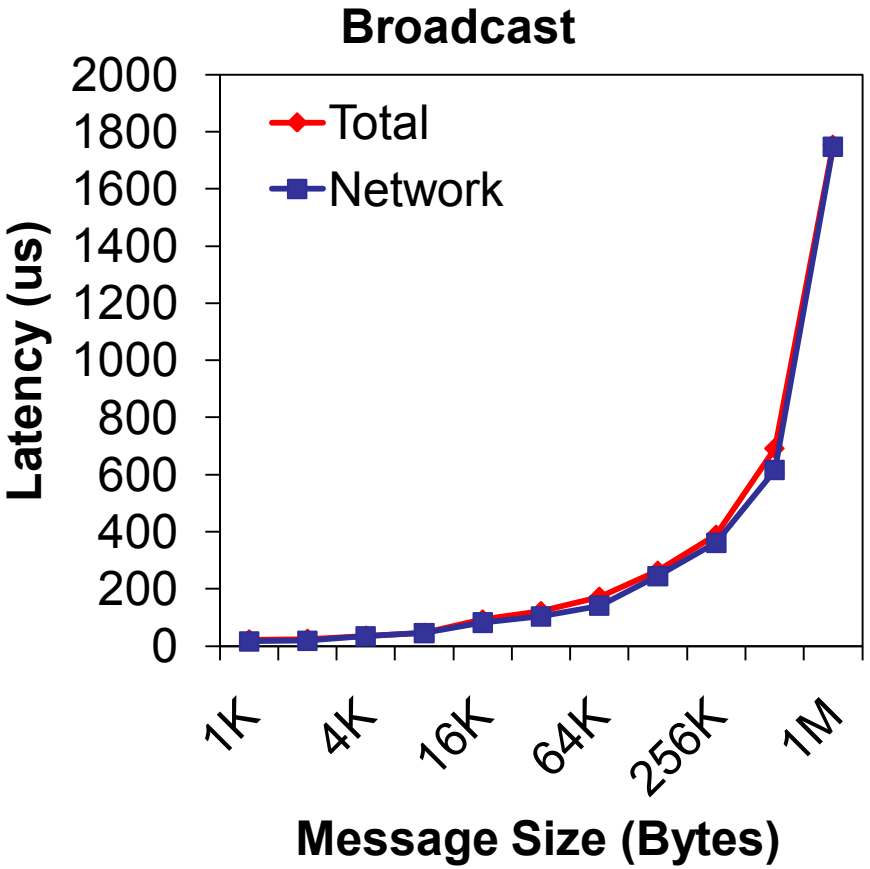
Multi-Core Aware Shared-Memory Based Collective Algorithms in MVAPICH2

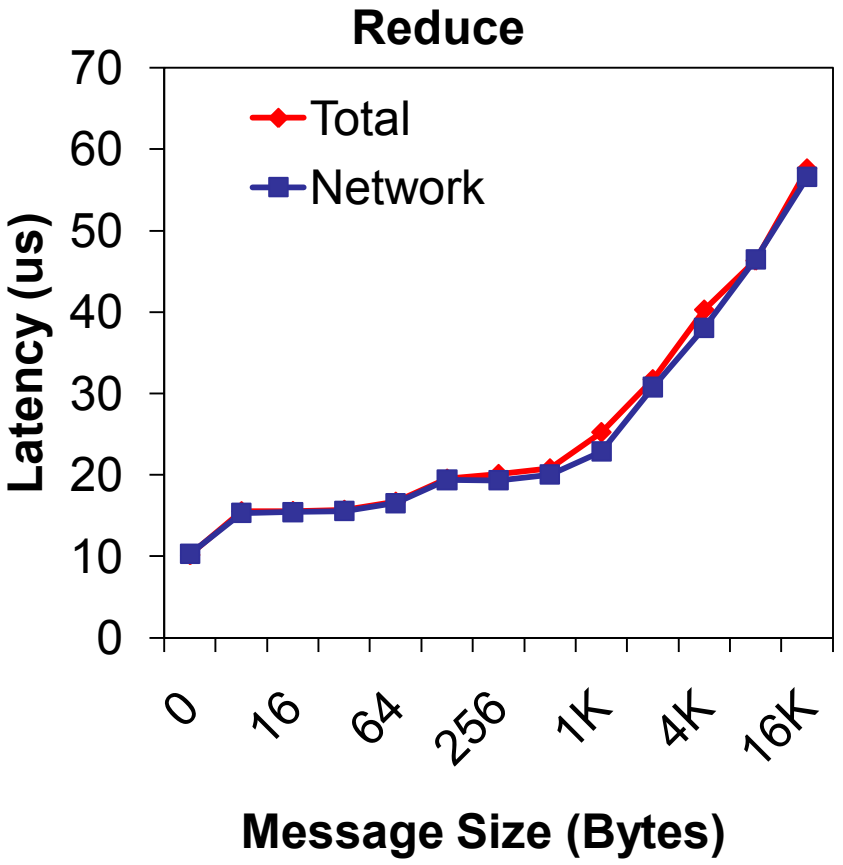
- Shared memory algorithms significantly improves performance
- Many cores remain idle during collective operations
- *Not power efficient*

Outline

- Introduction and Background
- Motivation
- Problem Statement
- Designing “Power-Aware” Collective Algorithms
- Experimental Evaluation
- Conclusions and Future Work

MPI Collective Performance
Total Vs Network Time

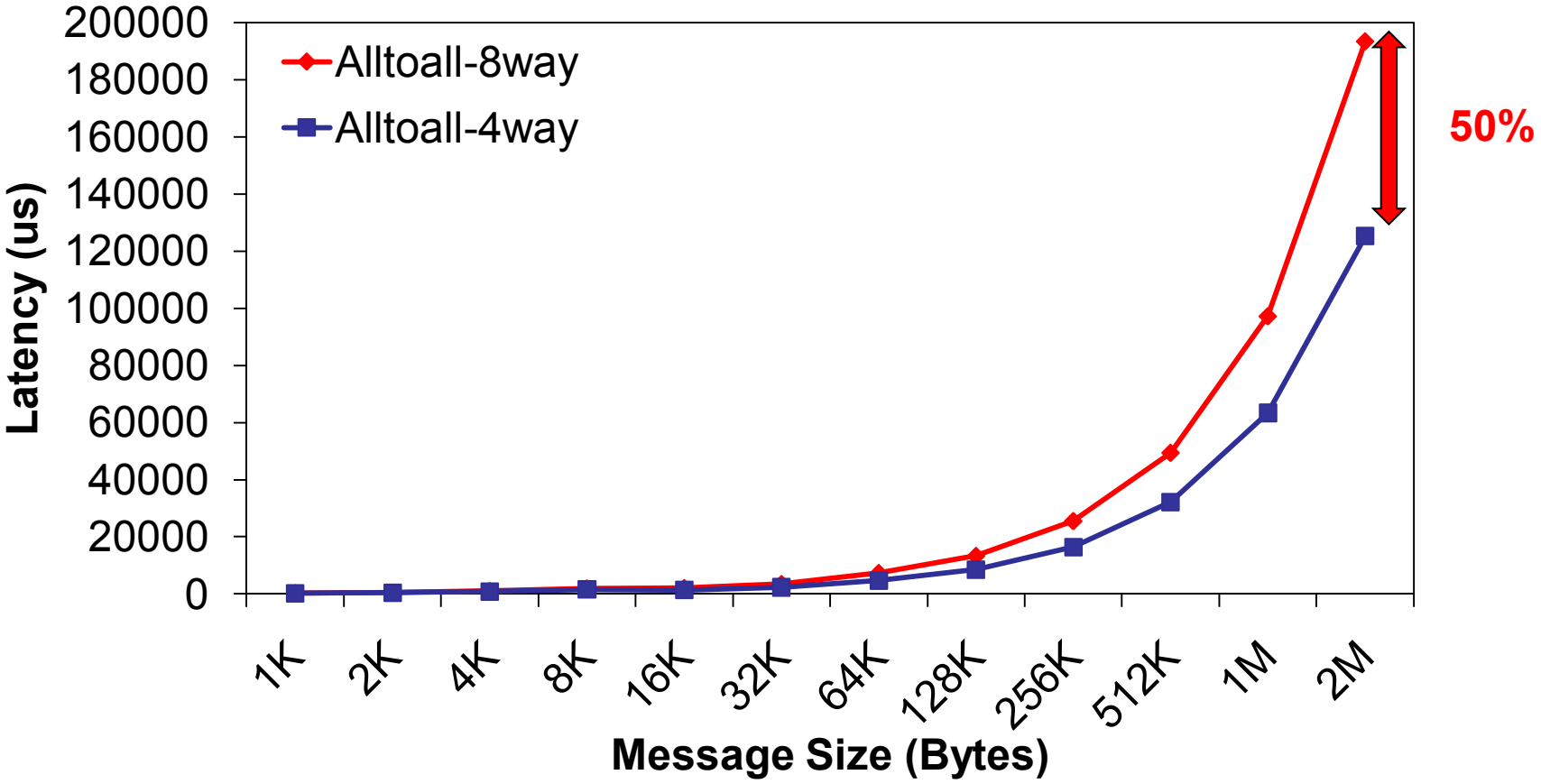




MPI Collective Performance Total Vs Network Time

- Network phase dominate total transfer time
- Non-leader cores idle – waste of power!

MPI Collective Performance
Impact of Network Congestion



- Performance in 8-way (4 nodes) configuration 50% worse than the 4-way (8-nodes) with the same system size

MPI Collective Performance
Impact of Network Congestion

Performance hit when more cores simultaneously involved in network transfer

Outline

- Introduction and Background
- Motivation
- Problem Statement
- Designing “Power-Aware” Collective Algorithms
- Experimental Evaluation
- Conclusions and Future Work

Problem Statement

- Existing designs use DVFS to save power without considering the nature of the collective algorithms
- Modern architectures allow DVFS and CPU Throttling operations to be performed within a few micro-seconds
- *Can we re-design collective communication algorithms in a power-aware manner taking into consideration the nature of the collective operation?*
- *What is the impact on performance?*
- *What architectural features can allow for more power savings with smaller performance overheads?*

Outline

- Introduction and Background
- Motivation
- Problem Statement
- Designing “Power-Aware” Collective Algorithms
- Experimental Evaluation
- Conclusions and Future Work

Design Space of Power-Aware Algorithms

Default (No Power Savings): Run each core at peak frequency/throttling state.

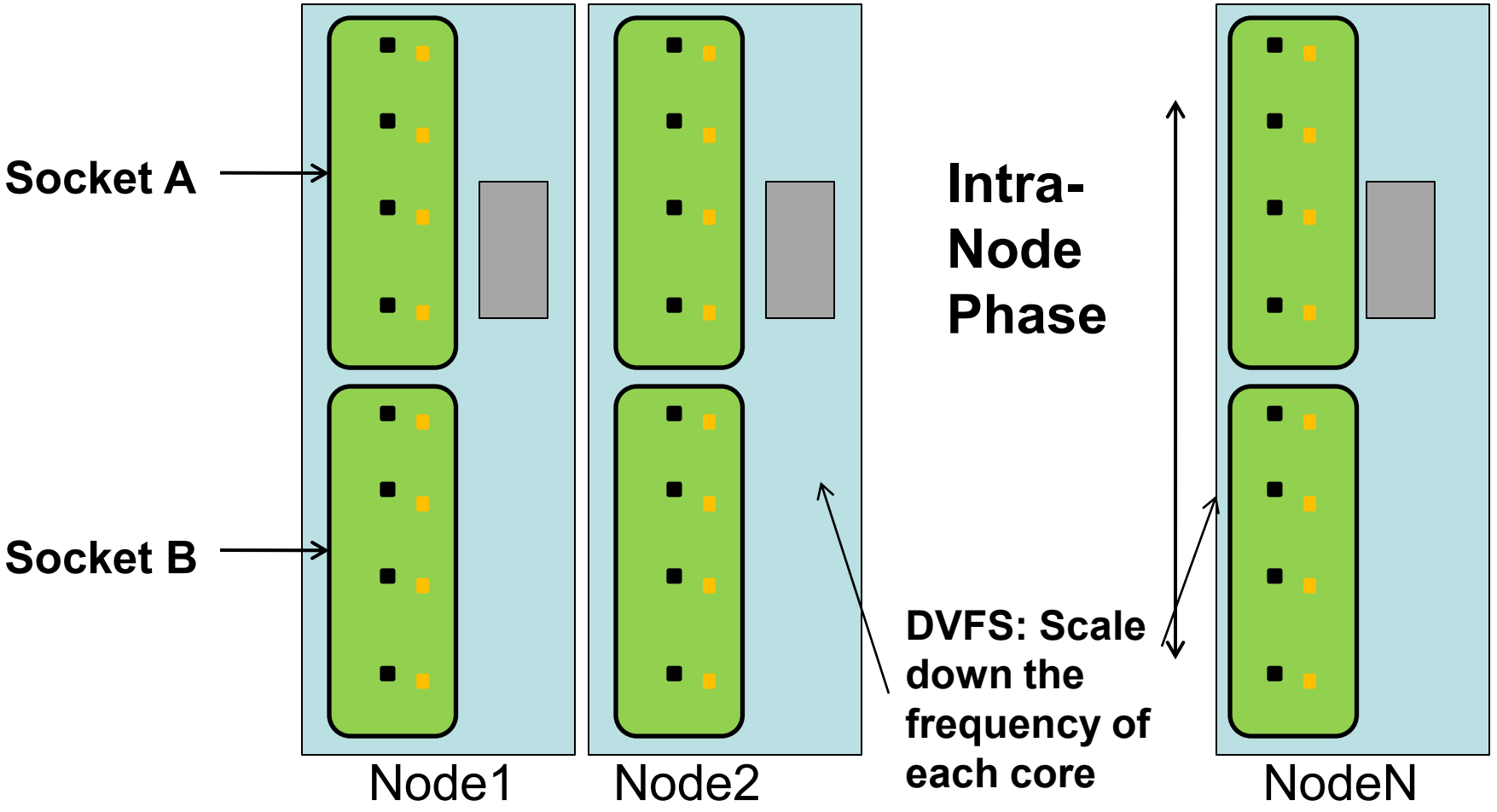
Frequency Scaling Only: Dynamically detect communication phases. Treat them as a black-box and scale the CPU frequency

Proposed: Consider the communication characteristics of different collectives, intelligently use both DVFS and CPU Throttling to deliver fine-grained power-savings

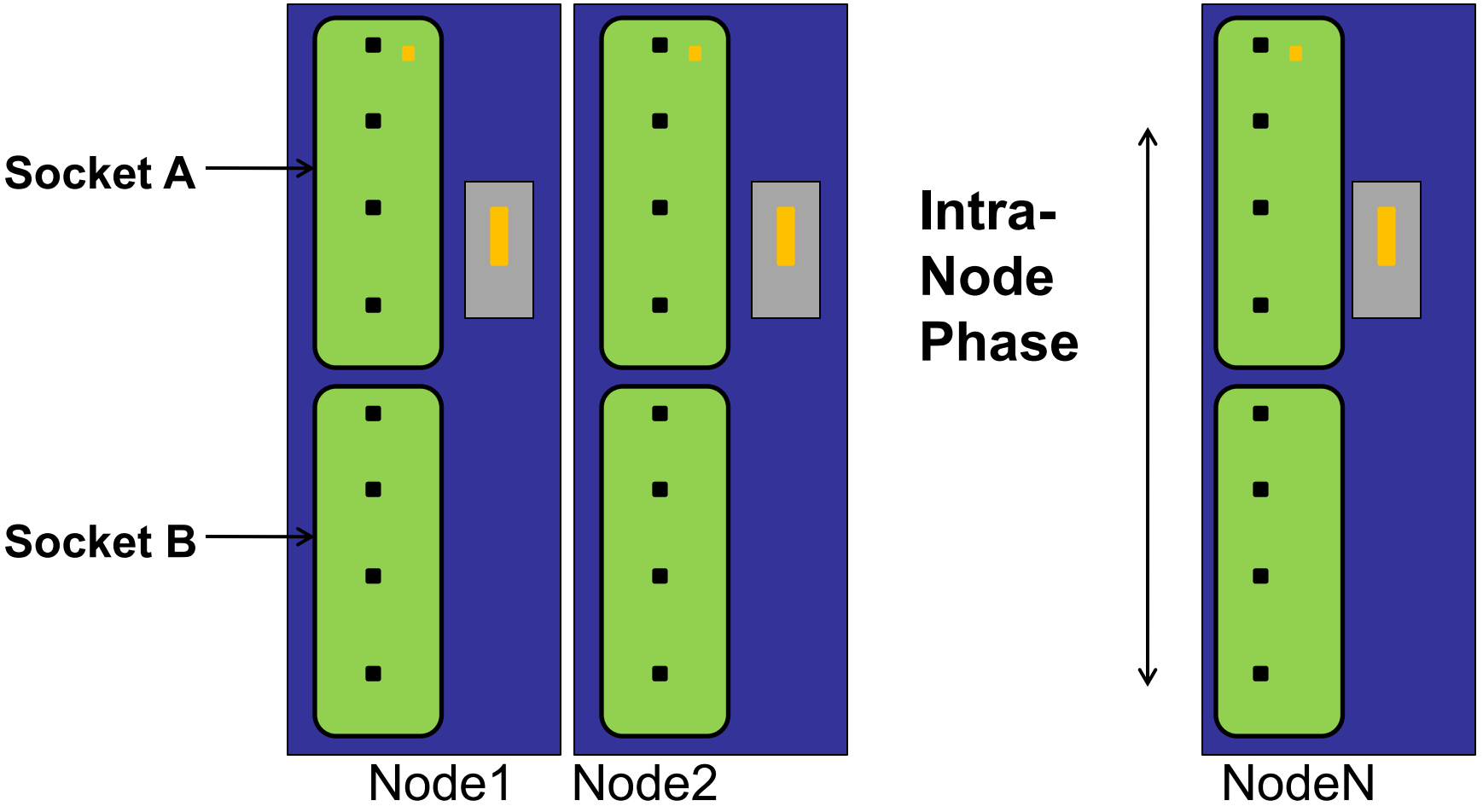
Proposed Approach

- Apply DVFS to scale down the frequency of all cores to the lowest level at the start of the collective operation
- Look for opportunities within the algorithms to use CPU Throttling to specific sets of cores of save more power
- Reset the state of the cores to the normal power settings while exiting collective operations

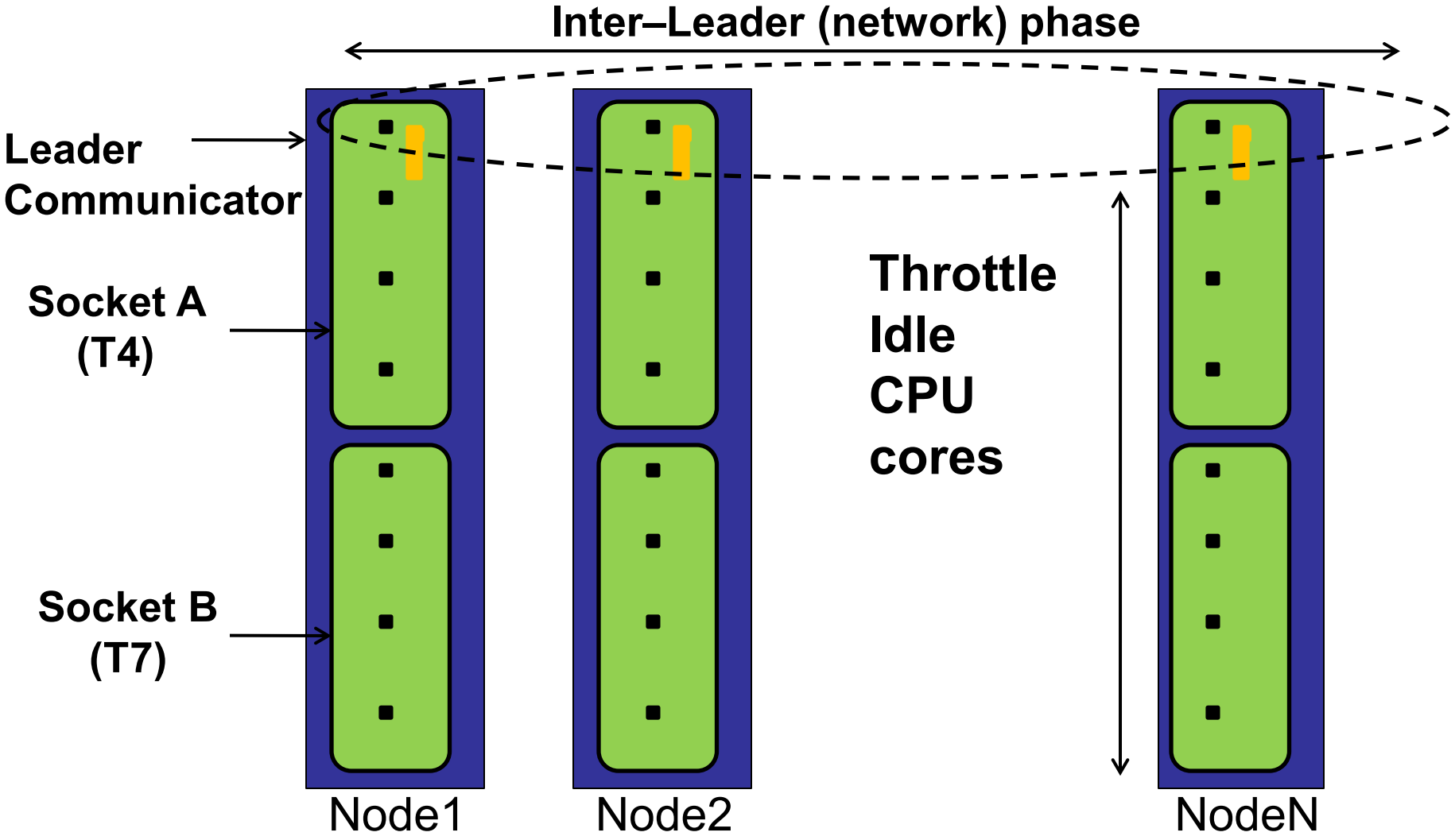
Proposed Power- Aware Shared-Memory Based Collective Algorithms in MVAPICH2



Proposed Power- Aware Shared-Memory Based Collective Algorithms in MVAPICH2



Proposed Power- Aware Shared-Memory Based Collective Algorithms in MVAPICH2

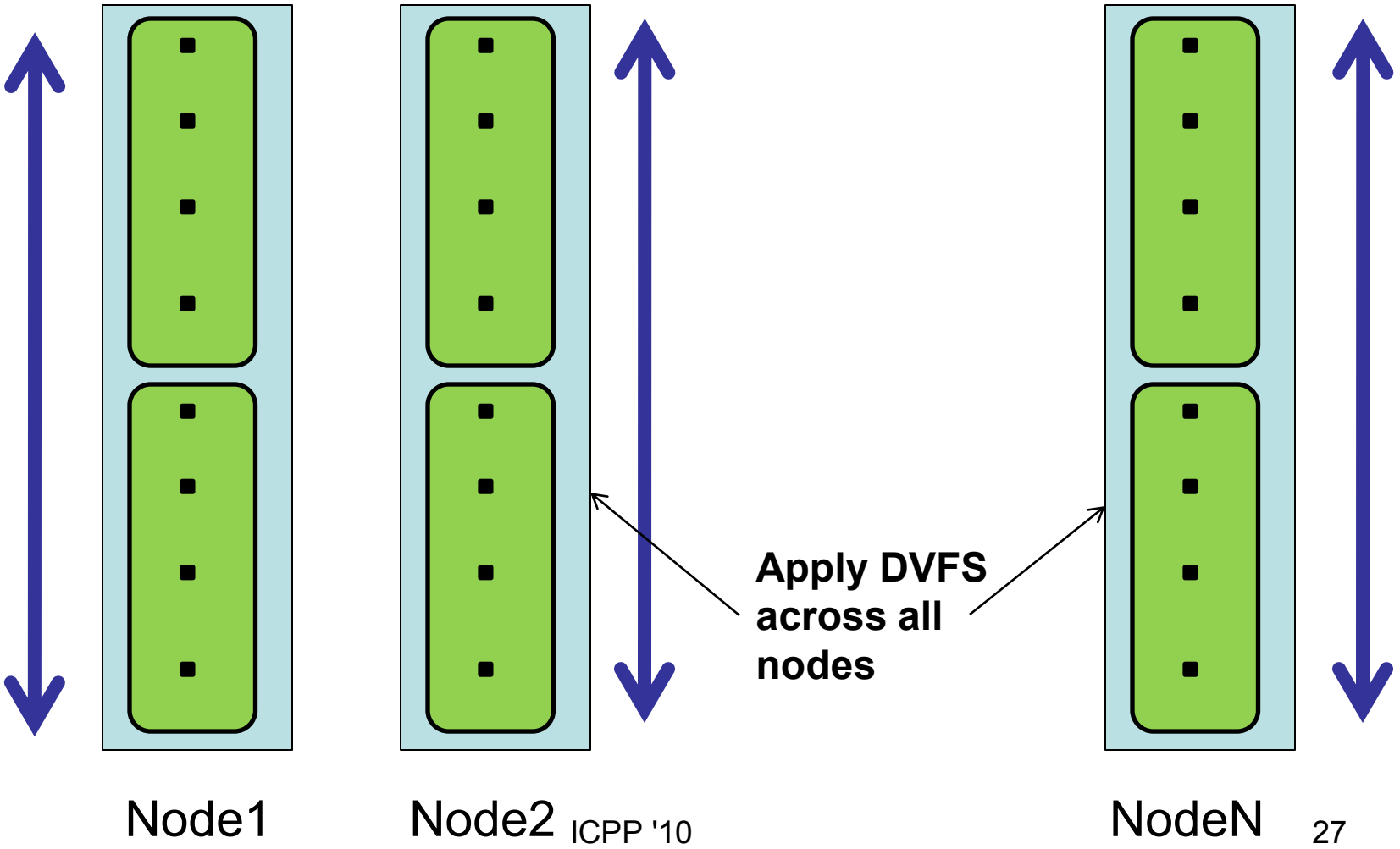


Proposed Power- Aware Shared-Memory Based Collective Algorithms in MVAPICH2

- Core-level CPU throttling would allow leaving leader core at high power state
- Lesser performance overhead ☺

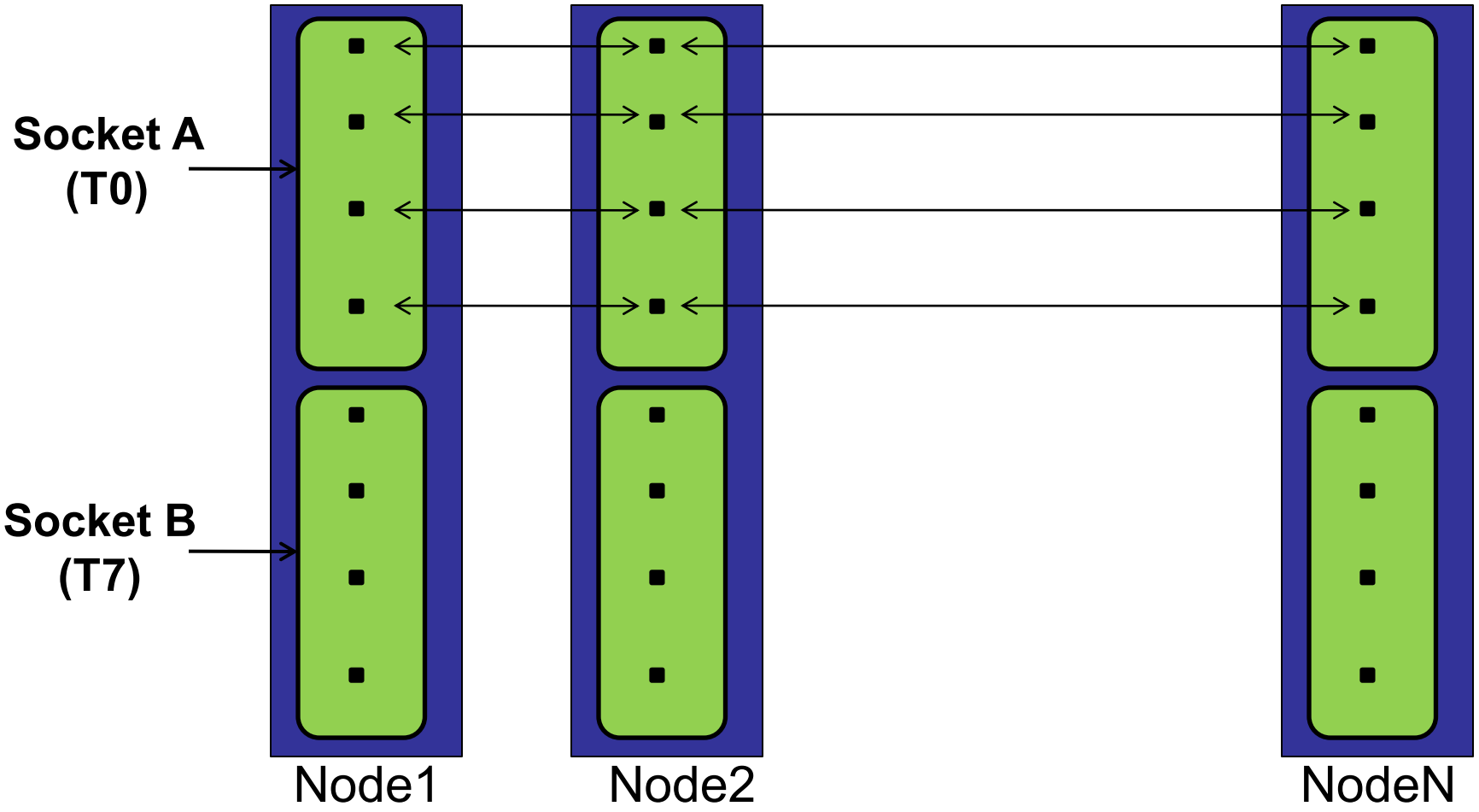
Proposed Power- Aware Alltoall Algorithms (Phase 1)

Complete all the intra-node exchanges

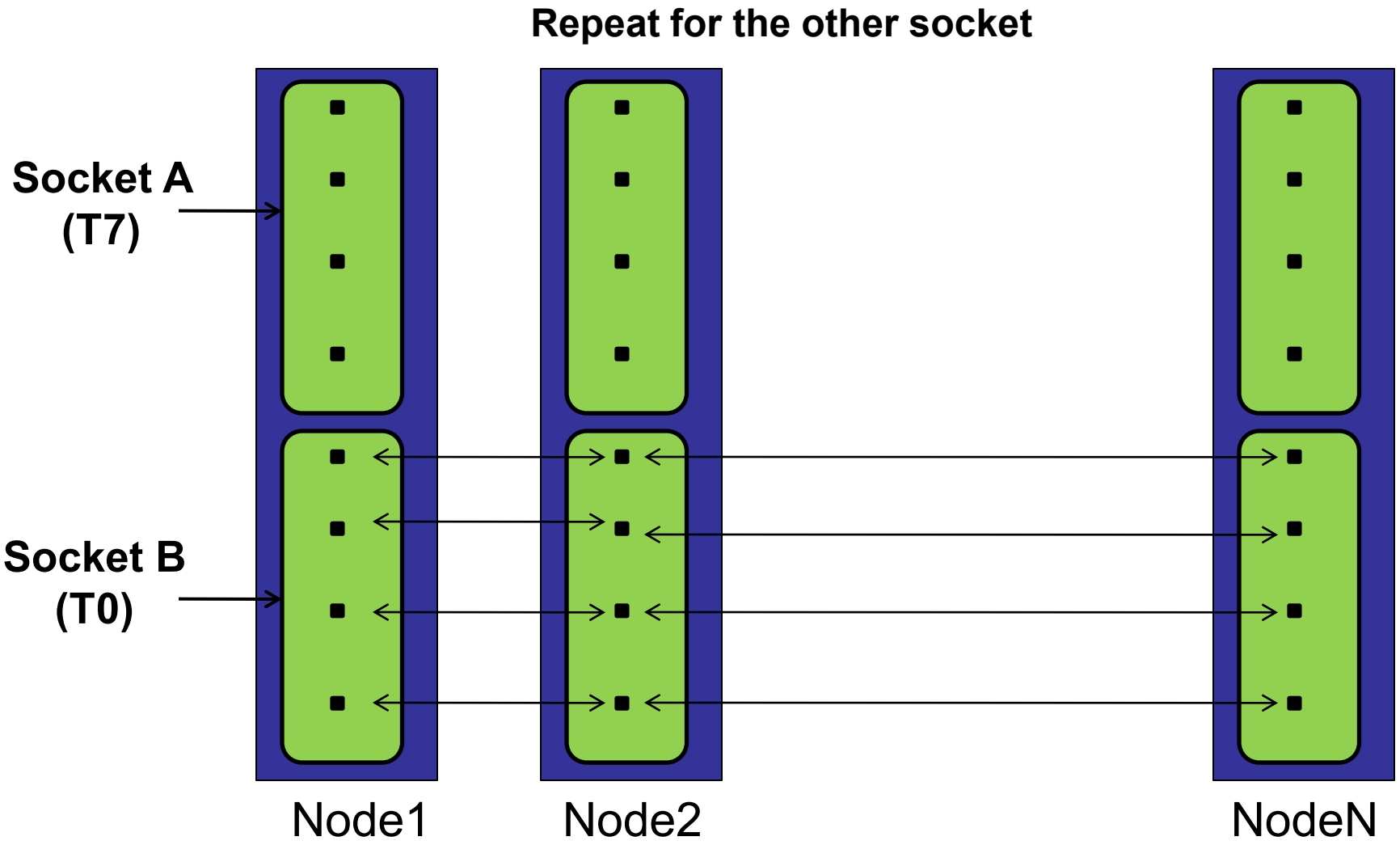


Proposed Power- Aware Alltoall Algorithms (Phase 2)

Active socket remains at T0
Passive socket powered down to T7

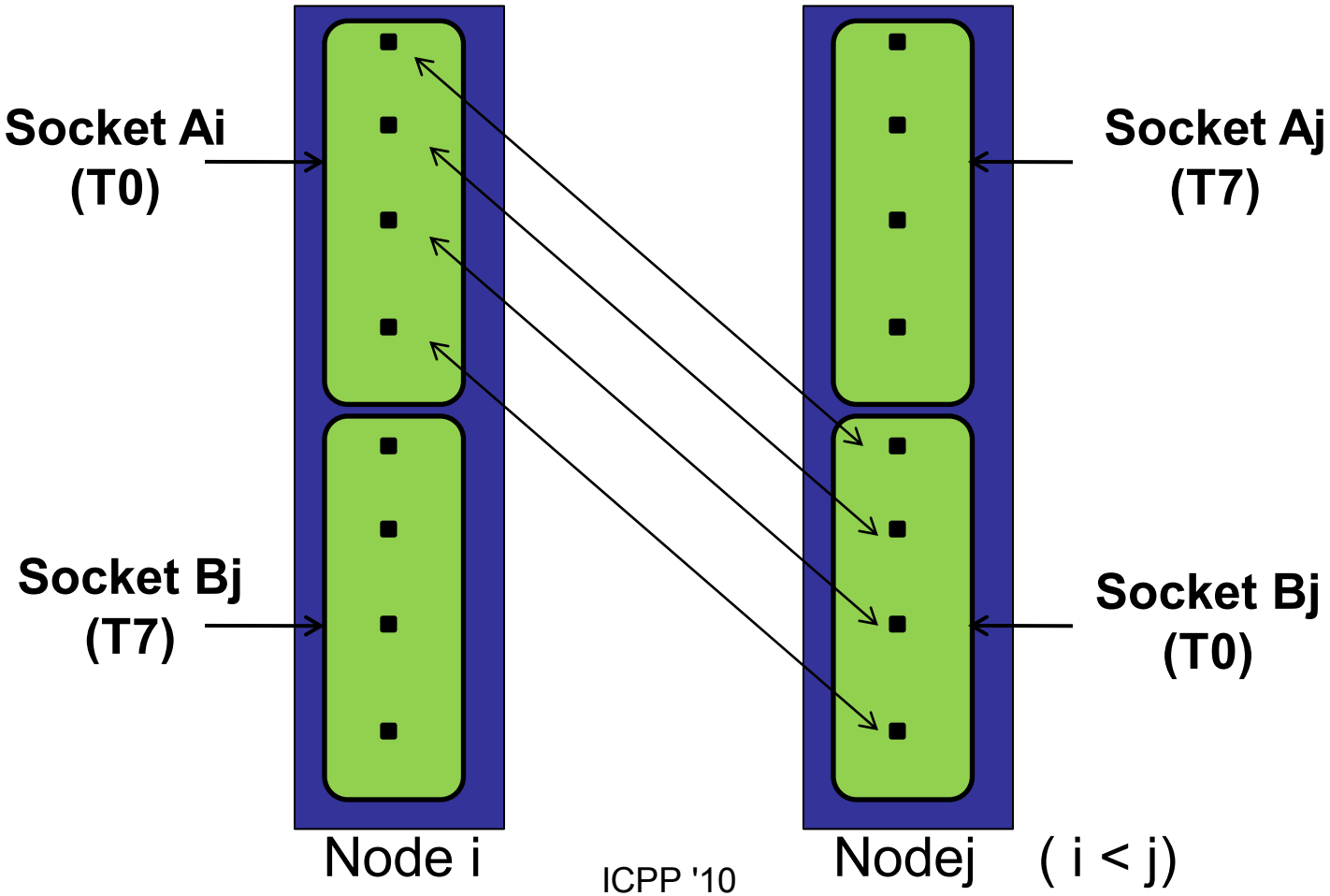


Proposed Power- Aware Alltoall Algorithms (Phase 3)

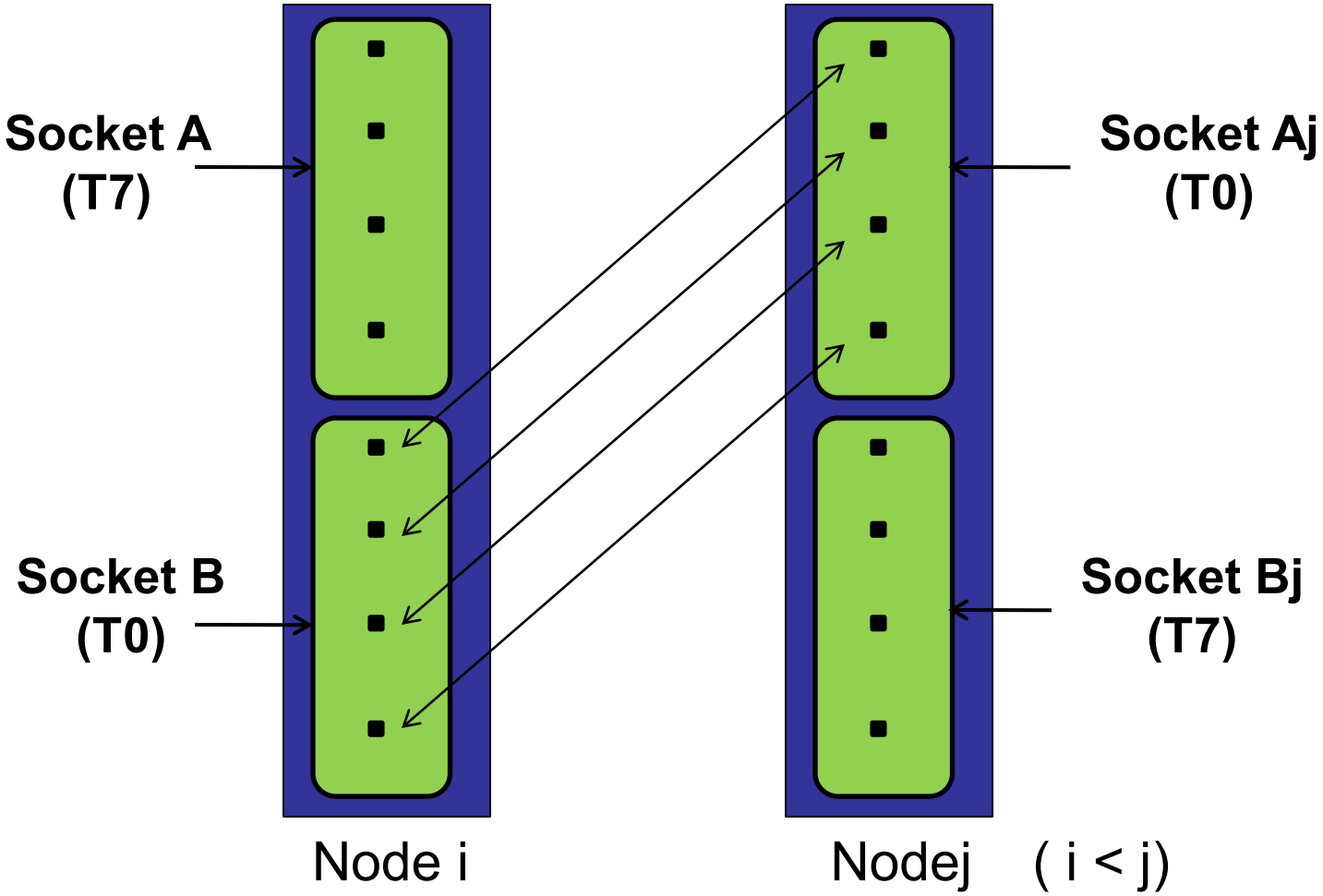


Proposed Power- Aware Alltoall Algorithms (Phase 4(a))

Choose a pair of nodes i and j (i < j)
Schedule communication and throttle sockets in the following manner



Proposed Power- Aware Alltoall Algorithms (Phase 4(b))



Proposed Power- Aware Alltoall Algorithms (Phase 4(b))

- Only half the cores communicate at one time
- CPU throttling leads to better power savings

Outline

- Introduction and Background
- Motivation
- Problem Statement
- Designing “Power-Aware” Collective Algorithms
- Experimental Evaluation
- Conclusions and Future Work

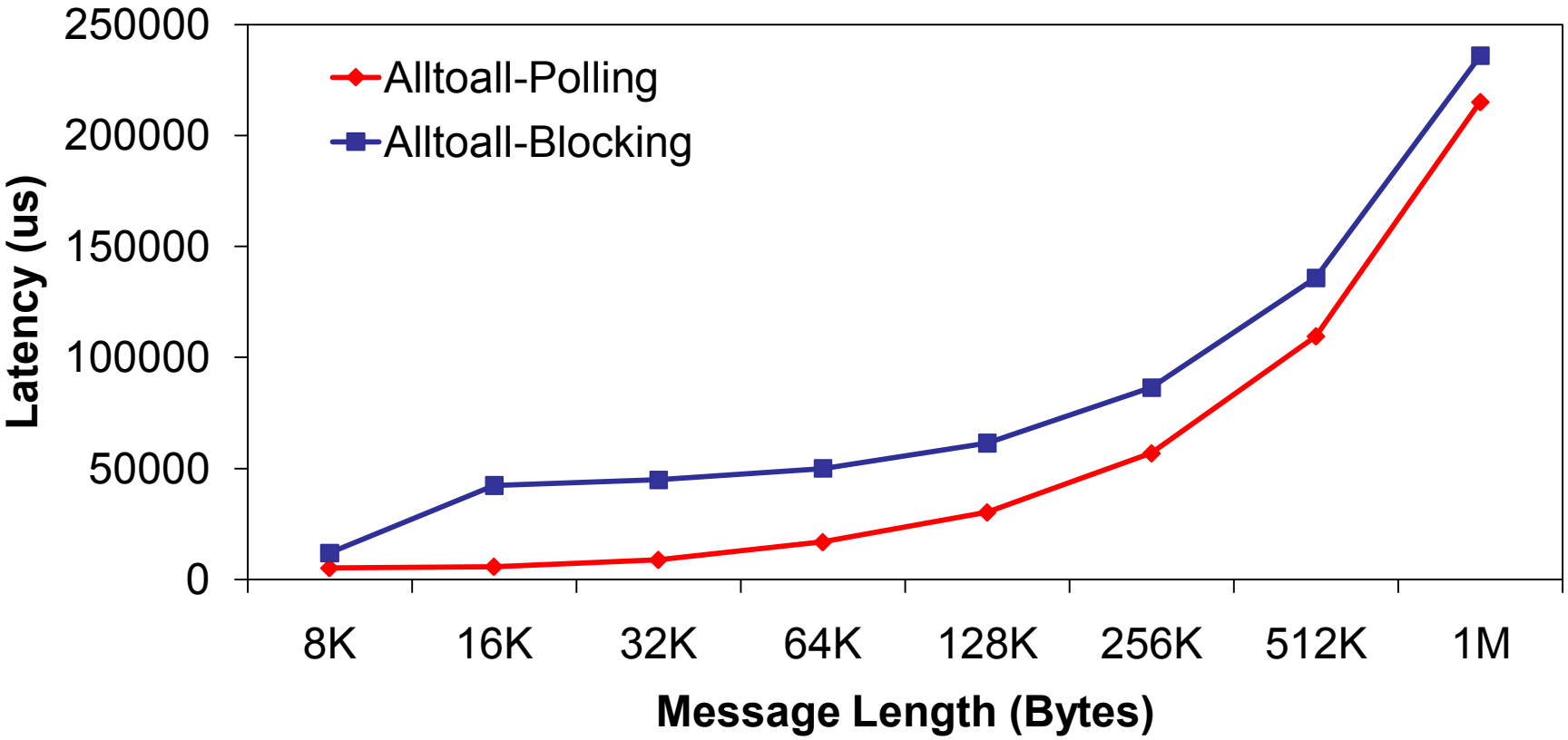
Experimental Setup

- Compute platforms
  - Intel Nehalem
    - Intel Xeon E5530 Dual quad-core processors operating at 2.40 GHz
    - 12GB RAM, 8MB cache
    - PCIe 2.0 interface
- Network Equipments
  - MT26428 QDR ConnectX HCAs
  - 36-port Mellanox QDR switch used to connect all the nodes
- Red Hat Enterprise Linux Server release 5.3 (Tikanga)
- OFED-1.4.2
- MASTECH MS2205 Clamp Power Meter to measure instantaneous power consumption

Experimental Setup (Cont)

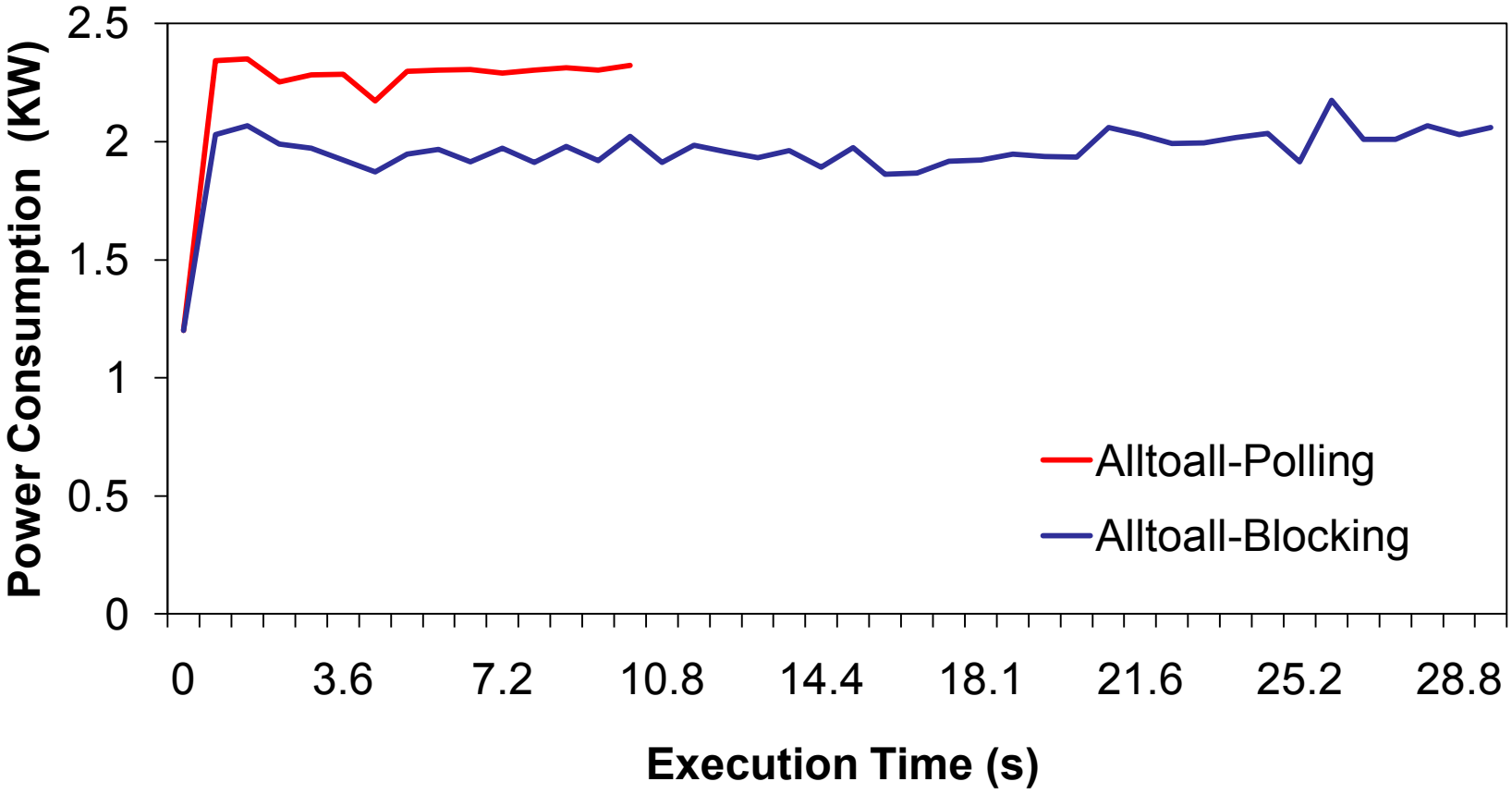
- MVAPICH2 – A High Performance MPI implementation over InfiniBand and other RDMA networks (v1.5.1)
  - http://mvapich.cse.ohio-state.edu/
  - Used by more than 1255 organizations world-wide
- Benchmarks
  - Micro-Benchmarks: OSU Microbenchmark Suite
  - Application Benchmarks: CPMD and NAS
- Estimated power savings with applications based on observations with micro-benchmarks.

Alltoall Latency with InfiniBand Blocking and Polling Progression Modes



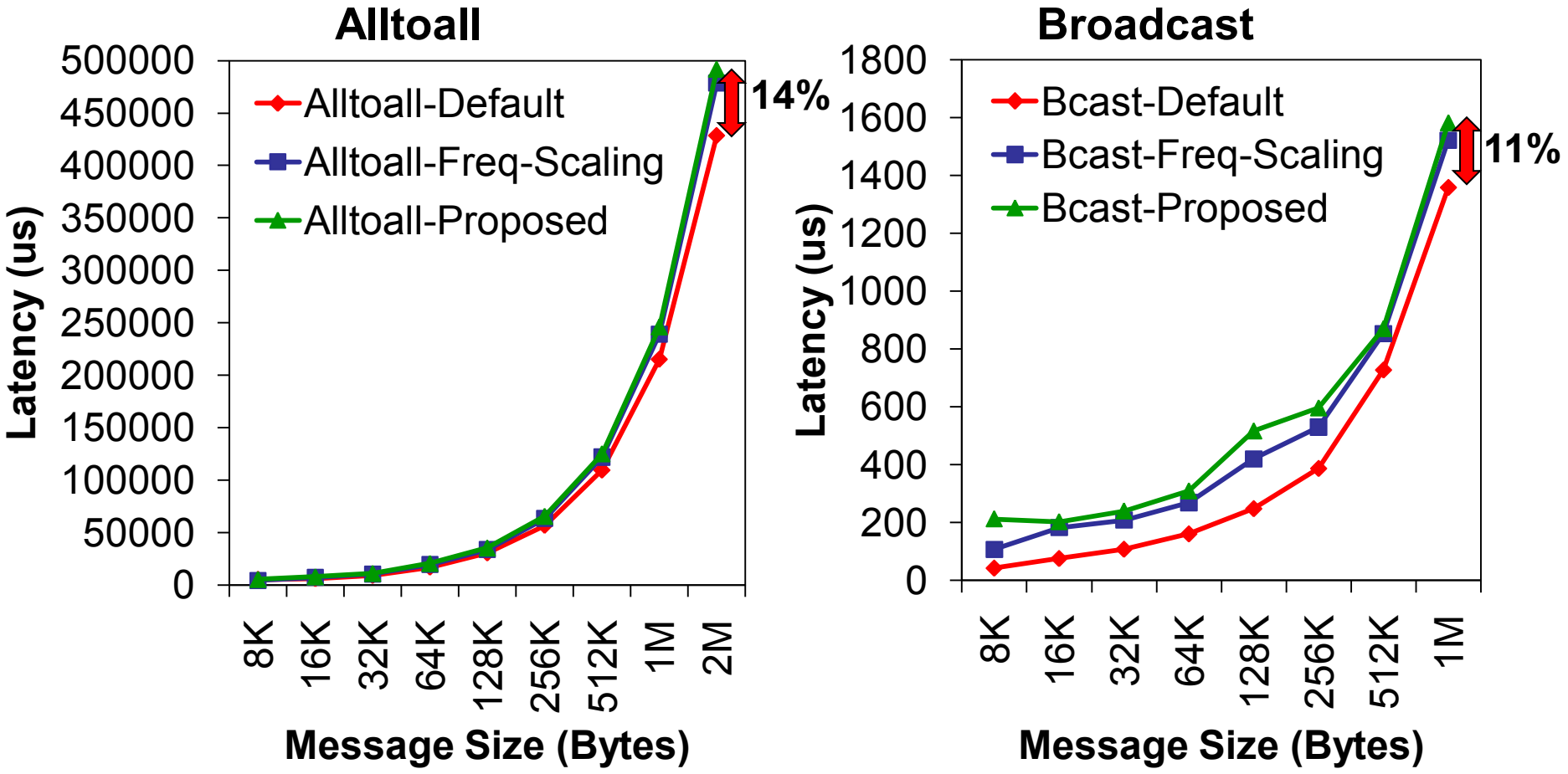
Blocking mode performs worse than polling mode for Alltoall

Alltoall Power Consumption with InfiniBand Blocking and Polling Progression Modes



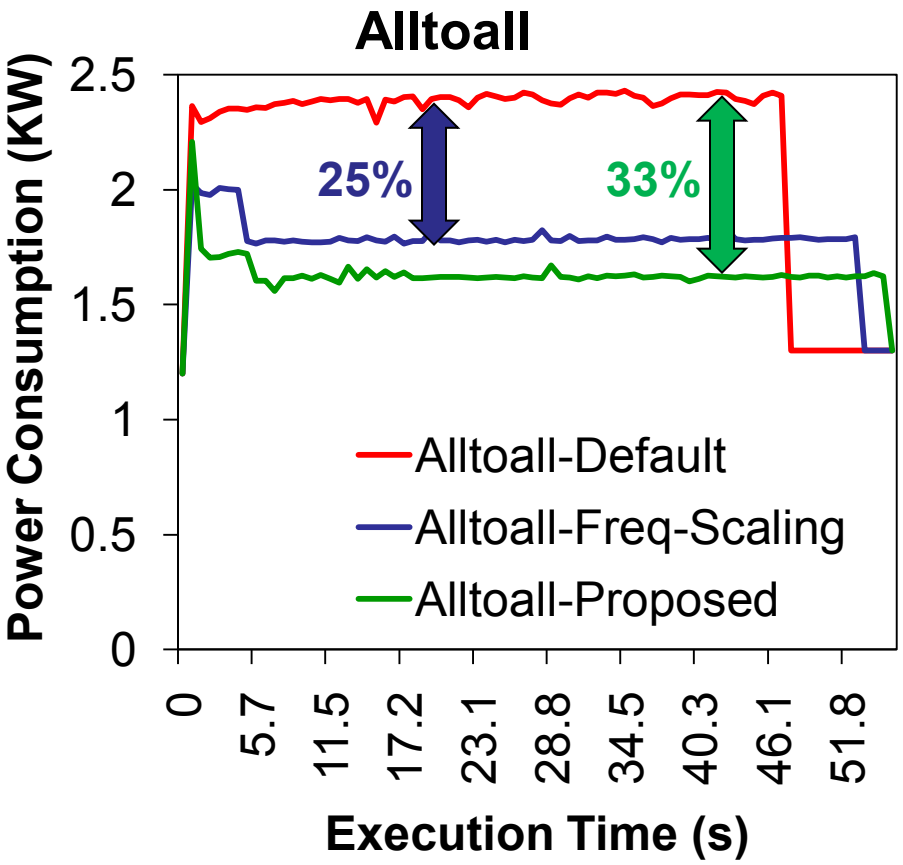
Blocking mode does allow for instantaneous power savings when compared to the polling mode

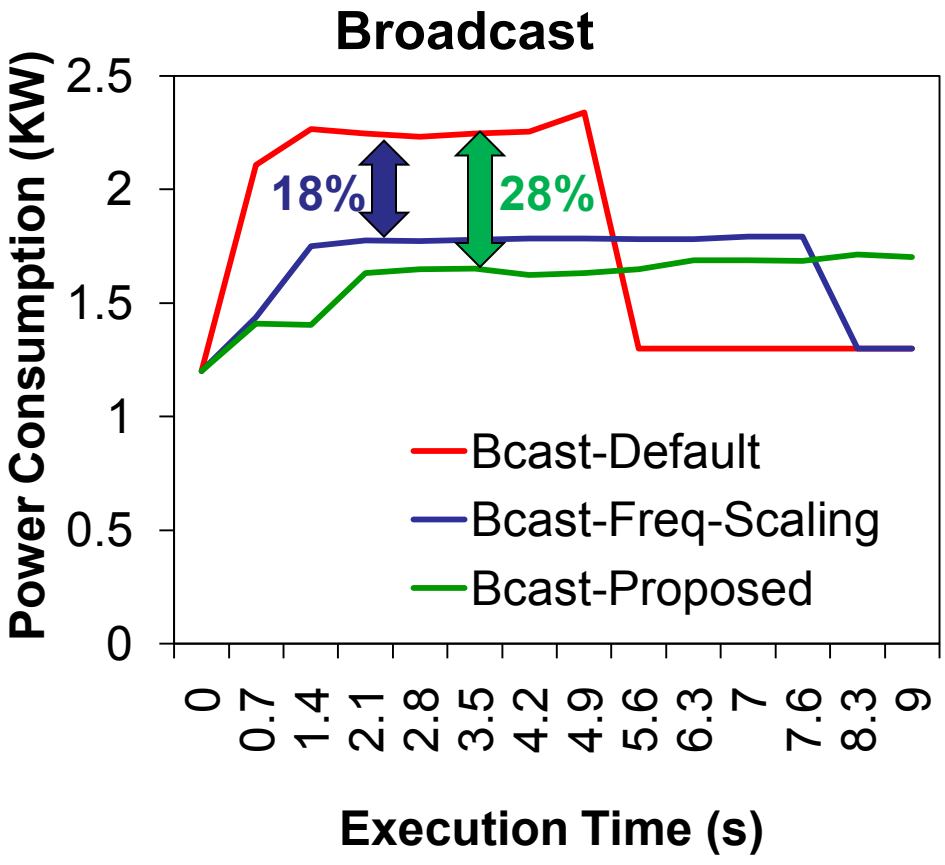
Latency for Collectives with Different Power Conservation Techniques



Additional performance impact due to CPU throttling is minimal

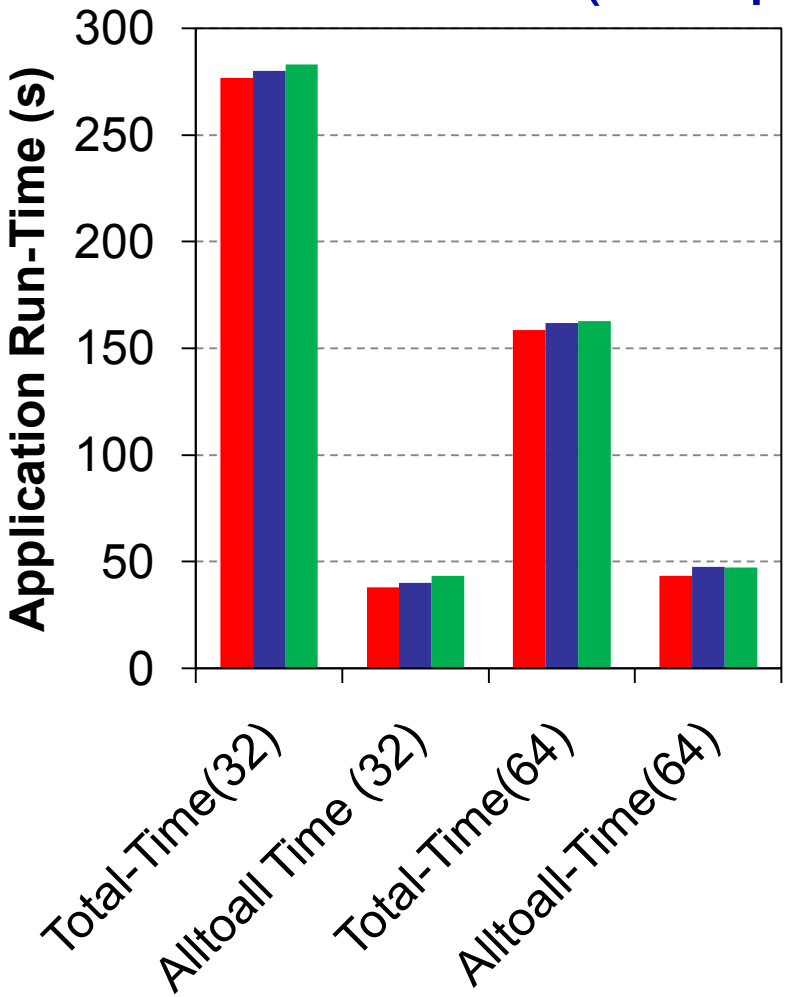
Power Consumption for Collectives with Different Power Conservation Techniques

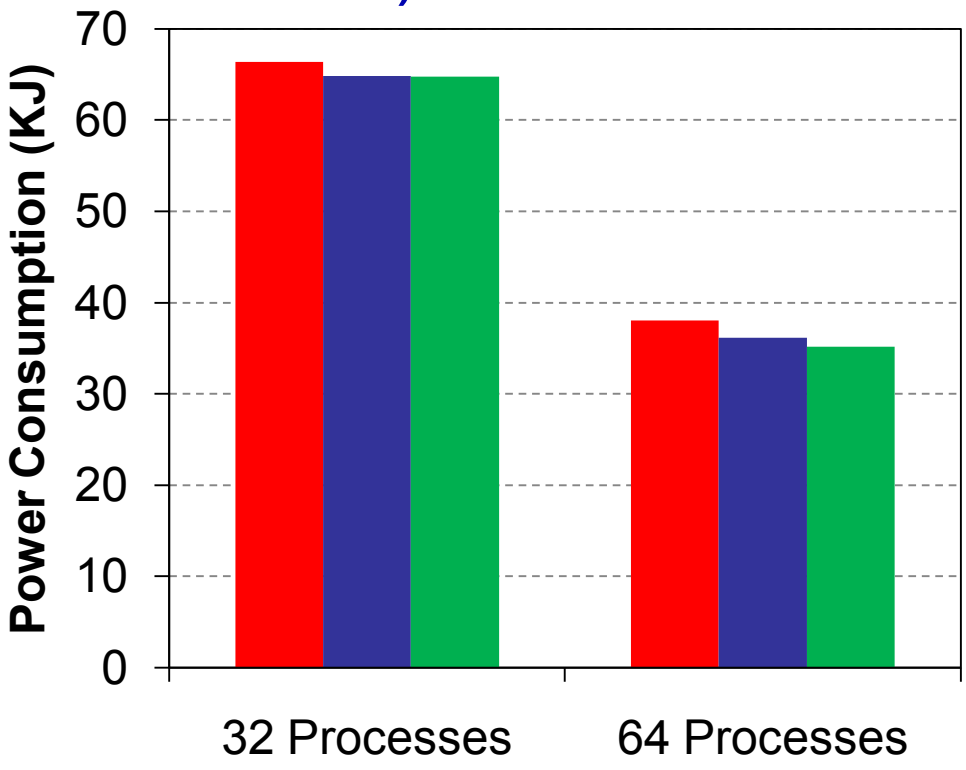




More power savings with proposed approach

CPMD with 32 and 64 Processes (ta-inp-md Data set)

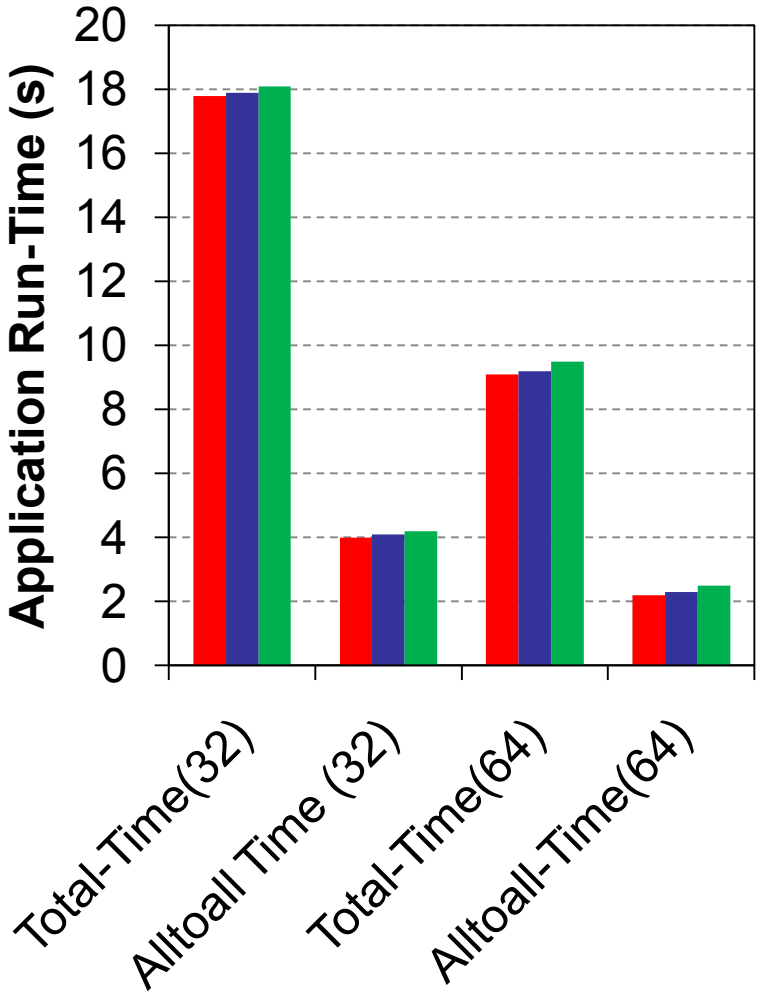


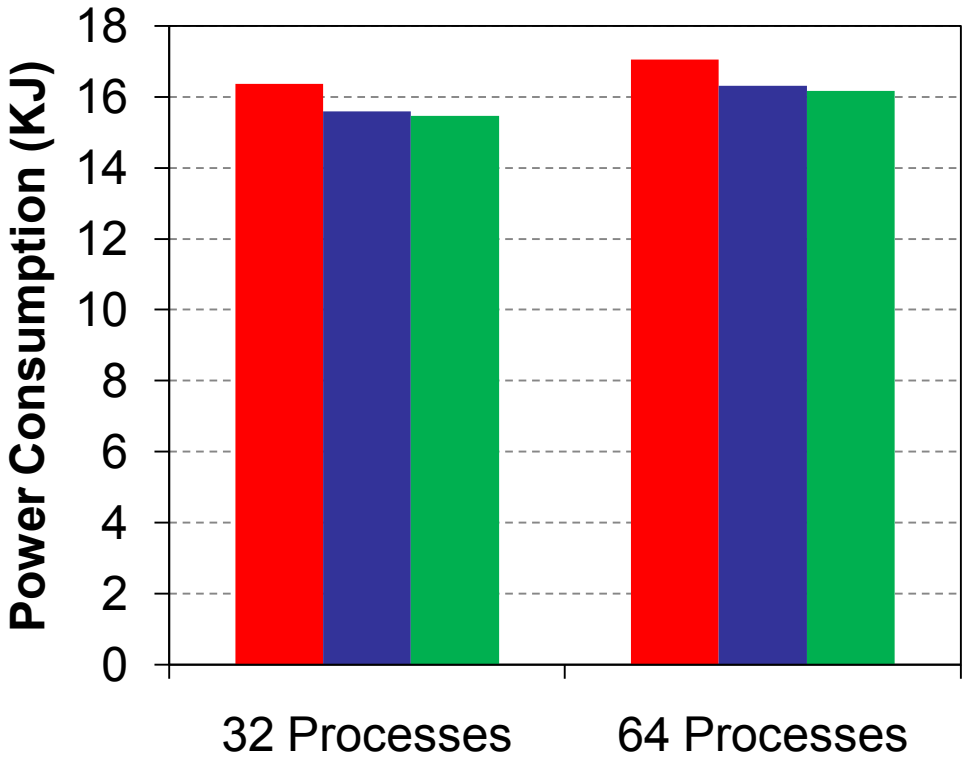


Estimated power savings - 7.7%
Performance degradation - 2.6%

Default Freq-Scaling Proposed Approach

NAS FT (Class C) 32 and 64 Processes





Estimated power savings - 5%
Performance degradation of - 10%

Default Freq-Scaling Proposed Approach

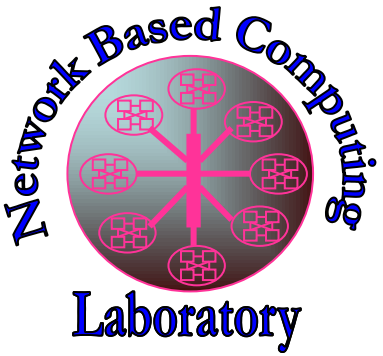
Outline

- Introduction and Background
- Motivation
- Problem Statement
- Designing “Power-Aware” Collective Algorithms
- Experimental Evaluation
- Conclusions and Future Work

Conclusions & Future Work

- Proposed and designed novel power-aware collective algorithms
- Designs deliver fine-grained power savings with little performance overheads
  - 33% savings in instantaneous power consumption with micro-benchmarks
  - Overall estimated energy savings of up to 8% with applications
- Core-level CPU throttling could potentially lead to better power-savings with smaller performance overheads
- Extend these designs to other collective operations and study the potential for saving power at larger scales
- These designs will be made available in future MVAPICH2 releases

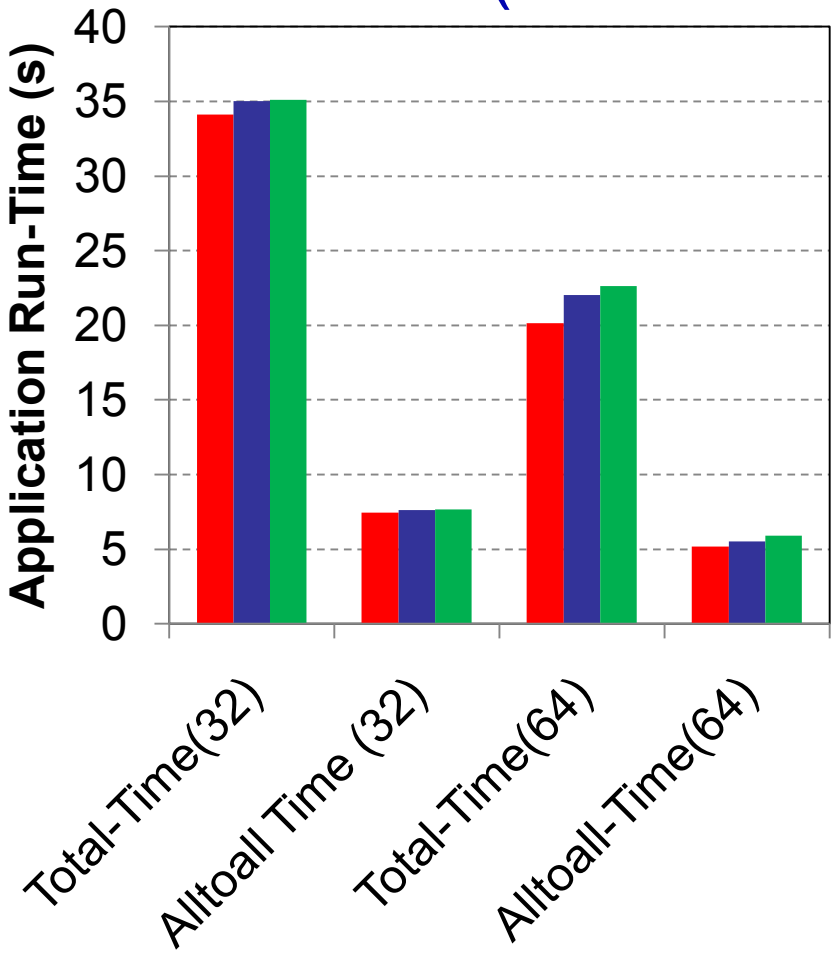
Thank you!

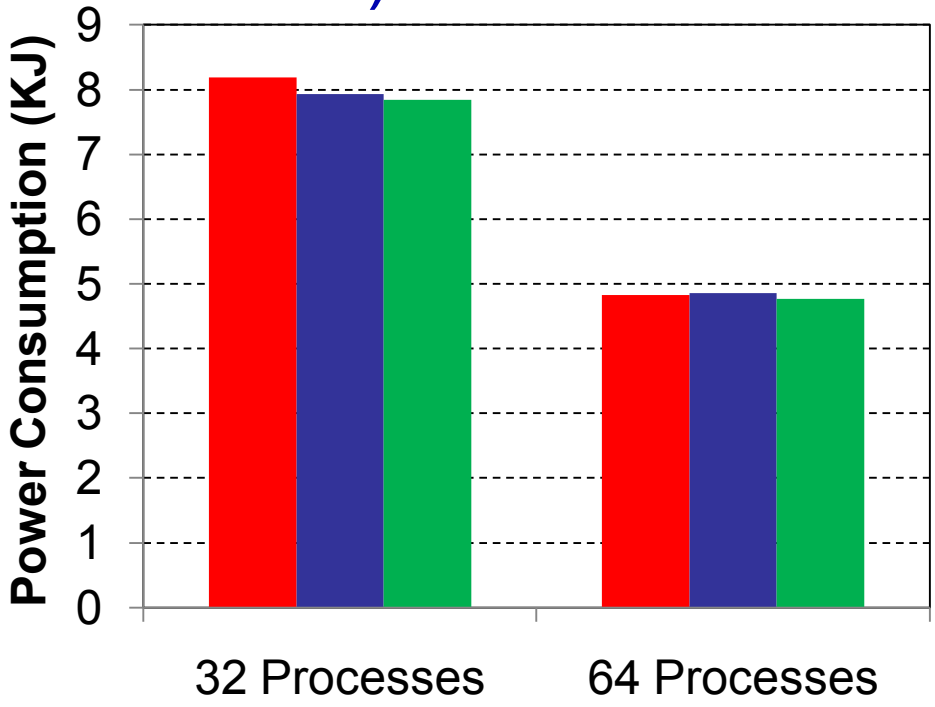


{kandalla, mancini, surs, panda}@cse.ohio-state.edu

Network-Based Computing Laboratory,
Ohio State University

CPMD with 32 and 64 Processes (Wat-32-inp-2 Data set)





About 3% power savings with a performance degradation of 2.8% with 32 processes with the proposed approach

Default Freq-Scaling Proposed Approach