Performance Evaluation of Adaptive MPI

Chao Huang[1], Gengbin Zheng[1], Sameer Kumar[2], Laxmikant Kale[1]

[1] University of Illinois at Urbana-Champaign
[2] IBM T. J. Watson Research Center

Motivation

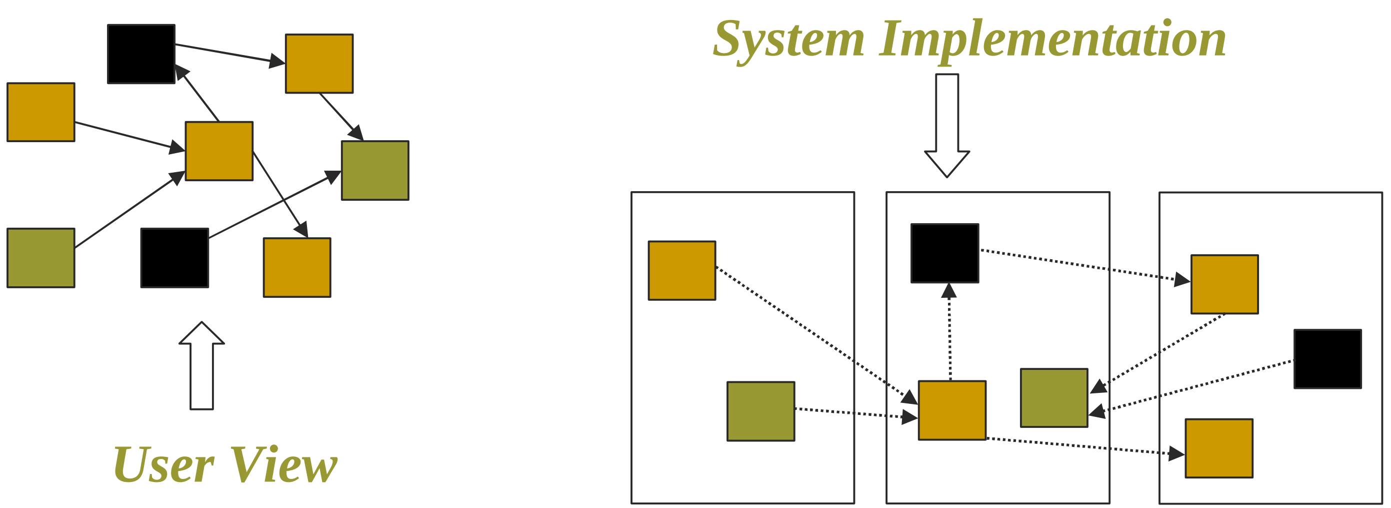
- Challenges
  - Applications with dynamic nature
    - Shifting workload, adaptive refinement, etc
  - Traditional MPI implementations
    - Limited support for such dynamic applications
- Adaptive MPI
  - Virtual processes (VPs) via migratable objects
  - Powerful run-time system that offers various novel features and performance benefits

Outline

- Motivation
- Design and Implementation
- Features and Benefits
  - Adaptive Overlapping
  - Automatic Load Balancing
  - Communication Optimizations
  - Flexibility and Overhead
- Conclusion

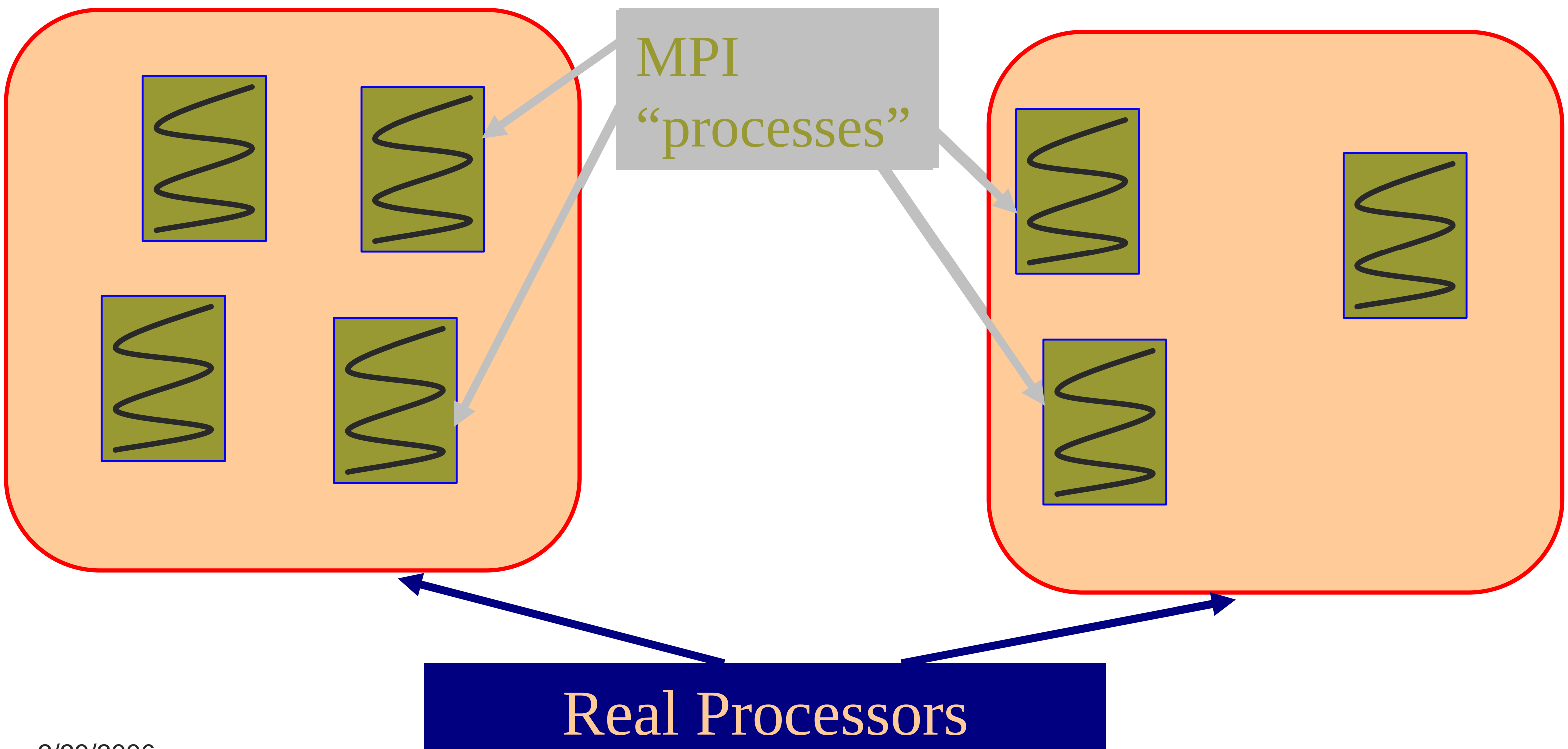
Processor Virtualization

- Basic idea of processor virtualization
  - User specifies interaction between objects (VPs)
  - RTS maps VPs onto physical processors
  - Typically, number of VPs >> P, to allow for various optimizations



AMPI: MPI with Virtualization

- Each AMPI virtual process is implemented by a user-level *thread* embedded in a migratable *object*

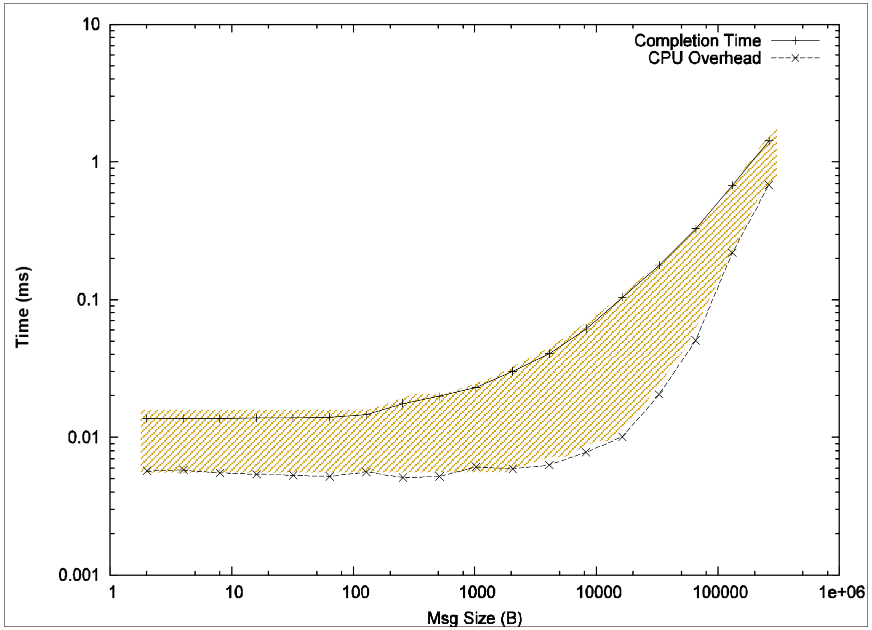


Outline

- Motivation
- Design and Implementation
- Features and Benefits
  - Adaptive Overlapping
  - Automatic Load Balancing
  - Communication Optimizations
  - Flexibility and Overhead
- Conclusion

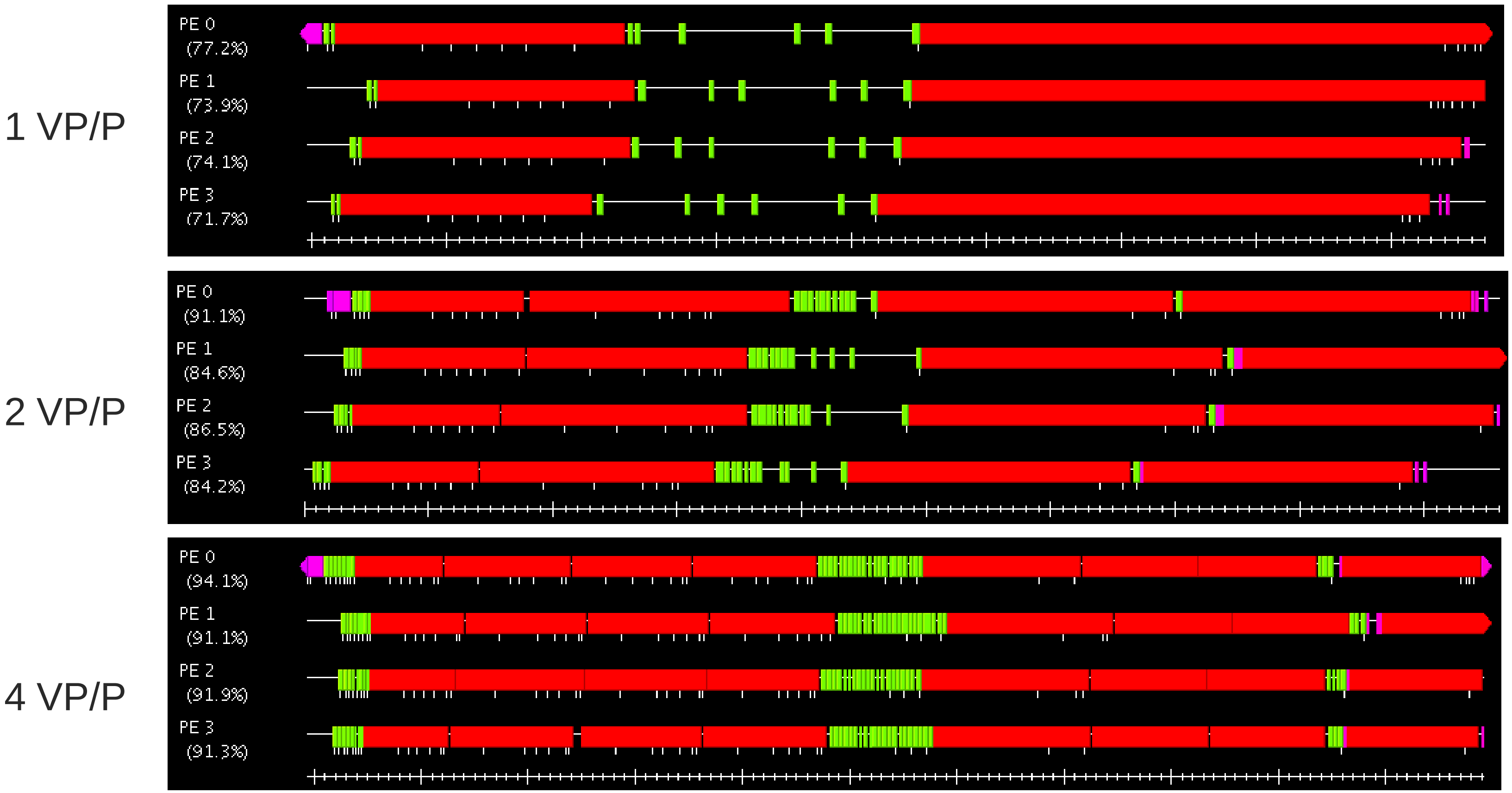
Adaptive Overlap

- Problem: Gap between completion time and CPU overhead
- Solution: Overlap between communication and computation



Completion time and CPU overhead of 2-way ping-pong program on Turing (Apple G5) Cluster

Adaptive Overlap



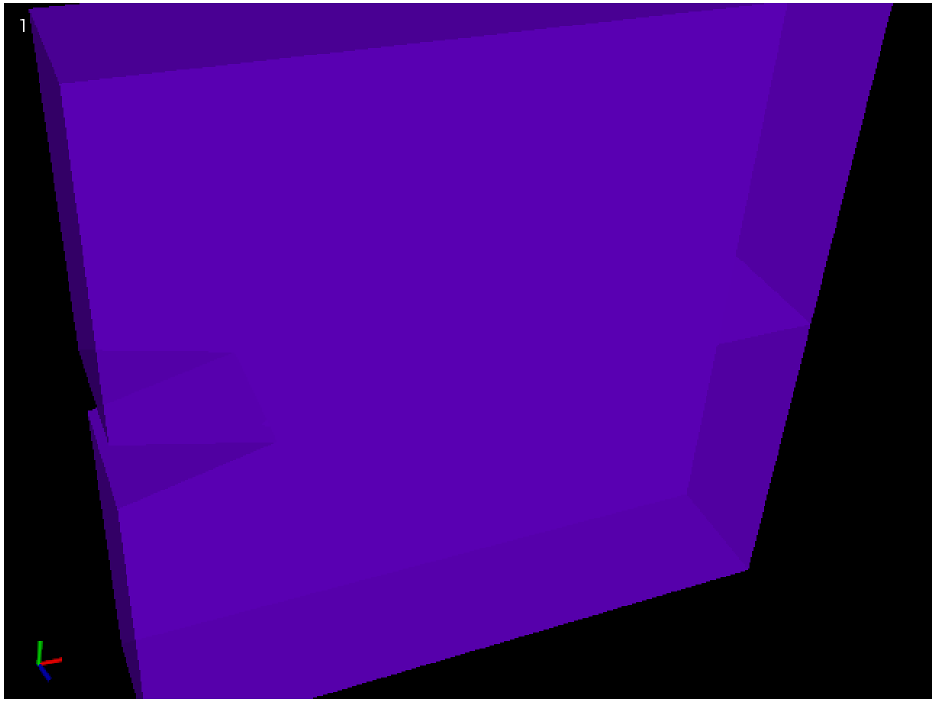
Timeline of 3D stencil calculation with different VP/P

Automatic Load Balancing

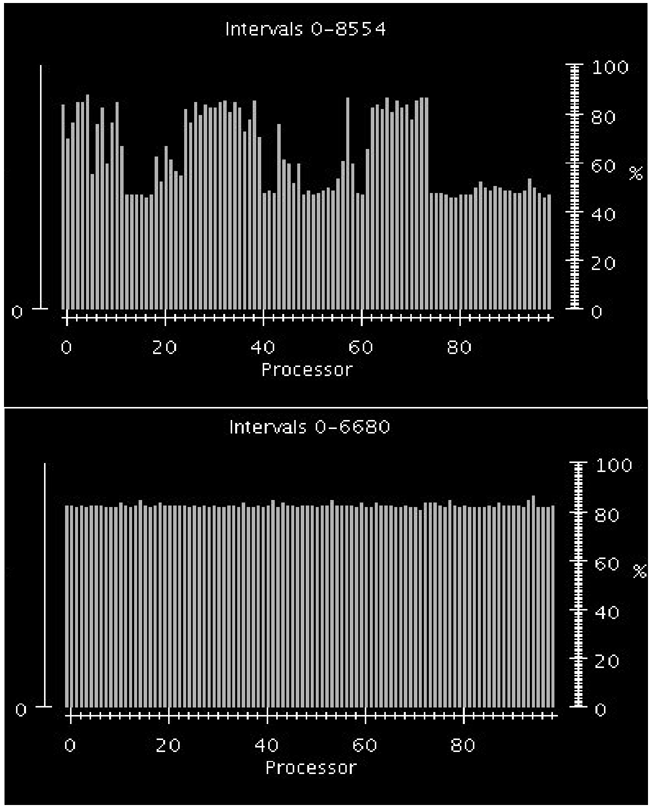
- Challenge
  - Dynamically varying applications
  - Load imbalance impacts overall performance
- Solution
  - Measurement-based load balancing
    - Scientific applications are typically iteration-based
    - The *principle of persistence*
    - RTS collects CPU and network usage of VPs
  - Load balancing by migrating threads (VPs)
    - Threads can be packed and shipped as needed
  - Different variations of load balancing strategies

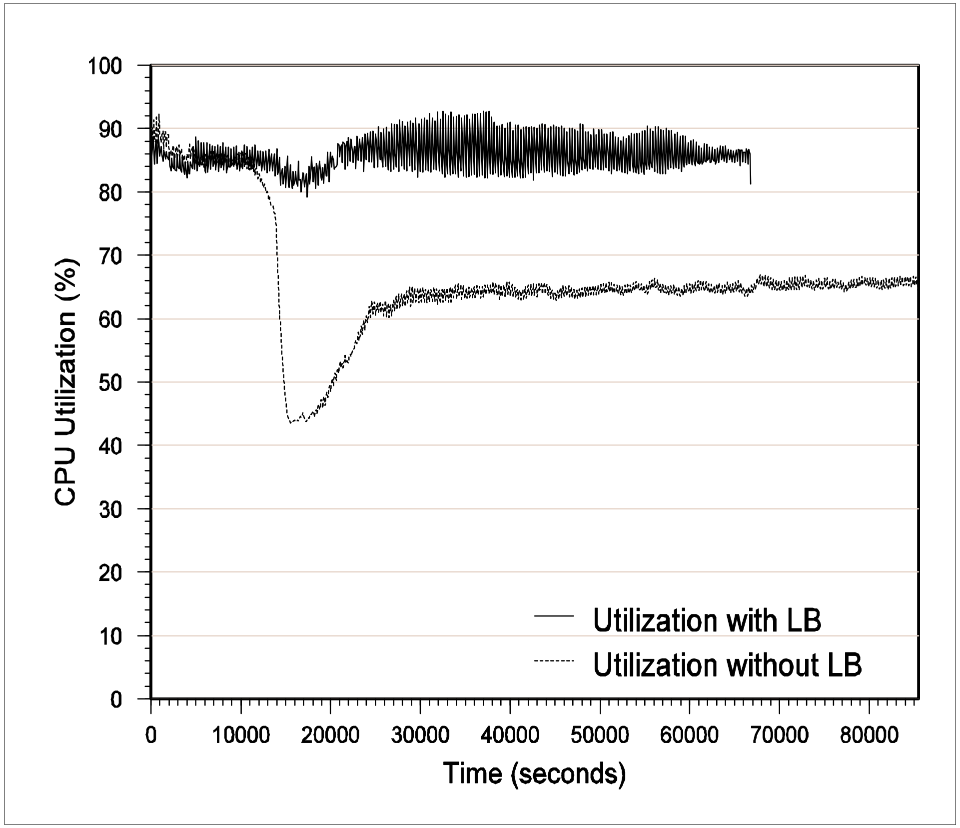
Automatic Load Balancing

- Application: Fractography3D
  - Models fracture propagation in material



Automatic Load Balancing





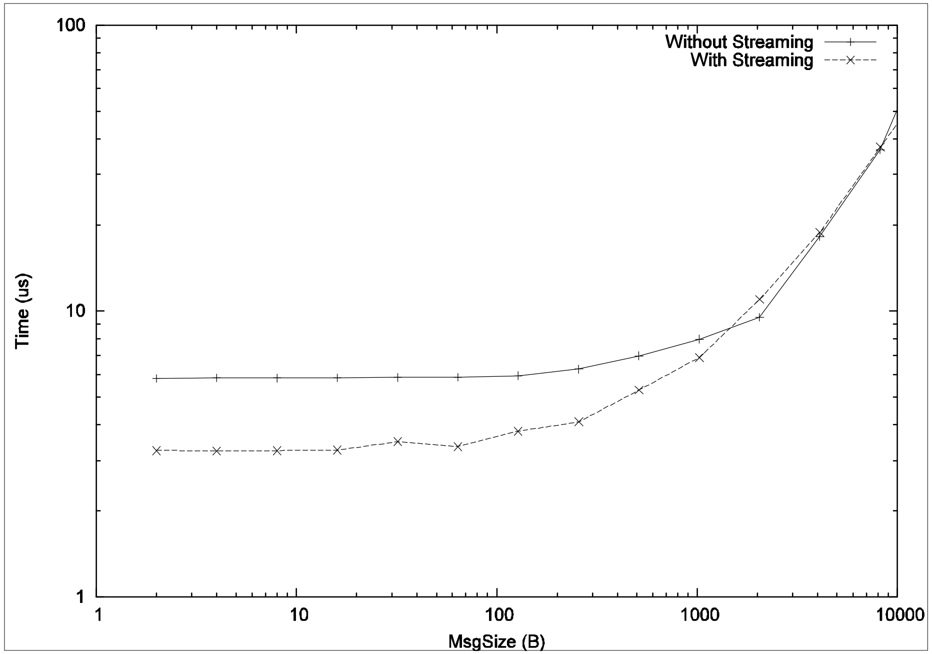
CPU utilization of Fractography3D without vs. with load balancing

Communication Optimizations

- AMPI run-time has capability of
  - Observing communication patterns
  - Applying communication optimizations accordingly
  - Switching between communication algorithms automatically
- Examples
  - Streaming strategy for point-to-point communication
  - Collectives optimizations

Streaming Strategy

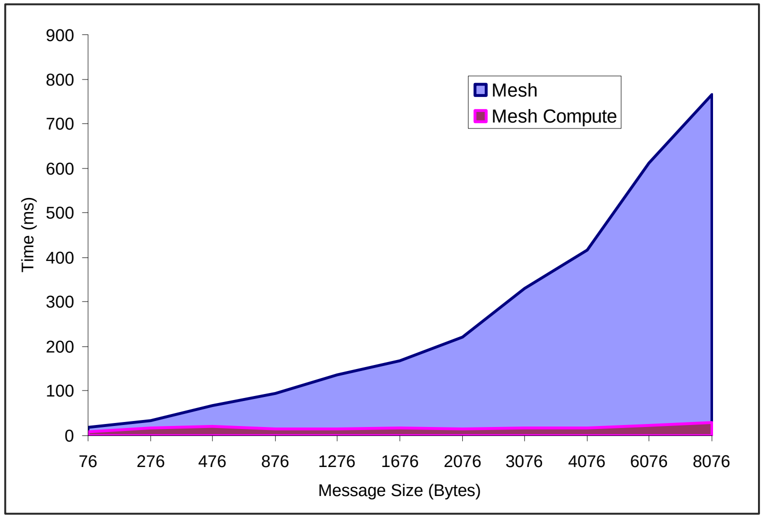
- Combining short messages to reduce per-message overhead



Streaming strategy for point-to-point communication on NCSA IA-64 Cluster

Optimizing Collectives

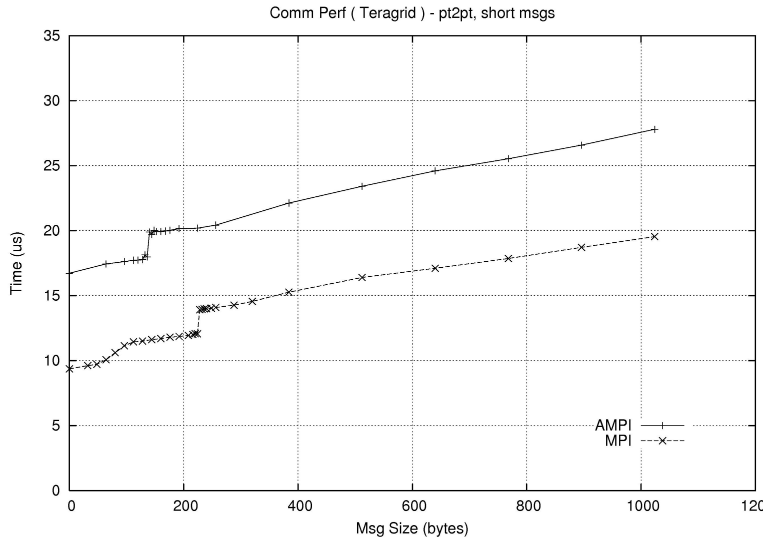
- A number of optimization are developed to improve collective communication performance
- Asynchronous collective interface allows higher CPU utilization for collectives
  - Computation is only a small proportion of the elapsed time

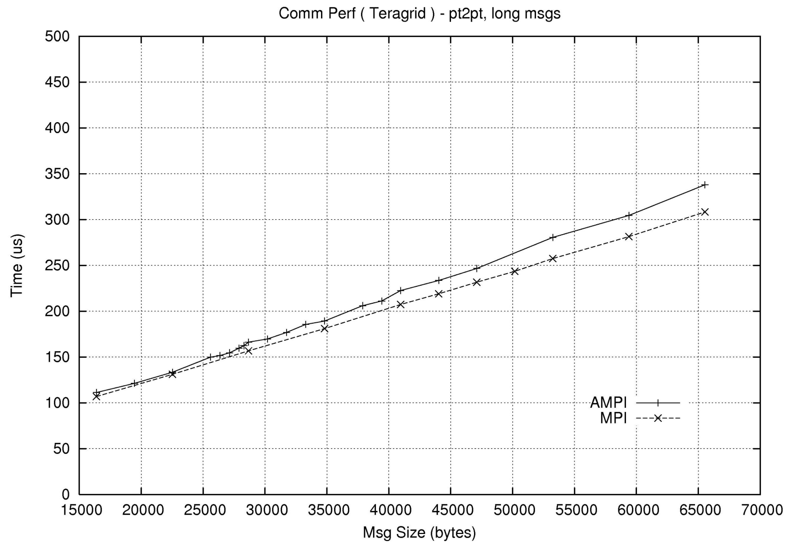


Time breakdown of an all-to-all operation using Mesh library

Virtualization Overhead

- Compared with performance benefits, overhead is very small
  - Usually offset by caching effect alone
- Better performance when features are applied

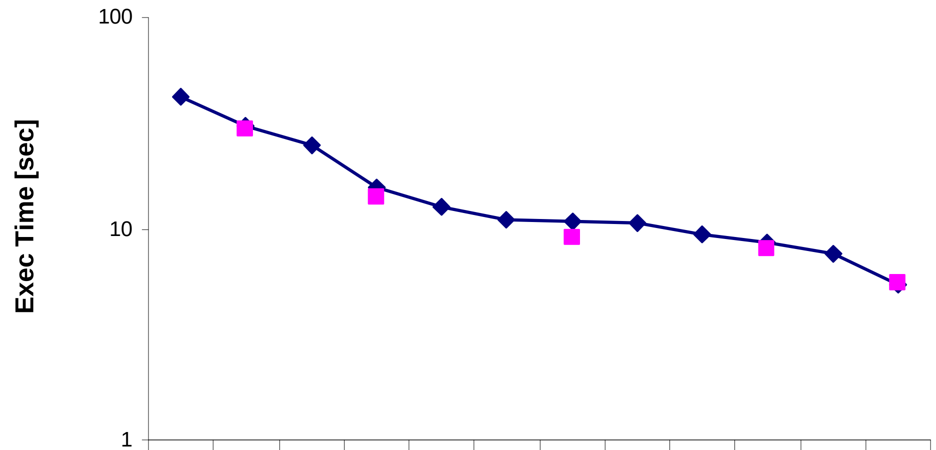




Performance for point-to-point communication on NCSA IA-64 Cluster

Flexibility

- Running on arbitrary number of processors
  - Runs with a specific number of MPI processes
  - Big runs on a few processors



Procs	19	27	33	64	80	105	125	140	175	216	250	512
Adaptive MPI	42.4	30.5	24.6	15.6	12.6	10.9	10.8	10.6	9.39	8.63	7.55	5.46
Native MPI		29.4		14.2			9.12			8.07		5.52

3D stencil calculation of size 240^3 run on Lemieux.

Outline

- Motivation
- Design and Implementation
- Features and Benefits
  - Adaptive Overlapping
  - Automatic Load Balancing
  - Communication Optimizations
  - Flexibility and Overhead
- Conclusion

Conclusion

- Adaptive MPI supports the following benefits
  - Adaptive overlap
  - Automatic load balancing
  - Communication optimizations
  - Flexibility
  - Automatic checkpoint/restart mechanism
  - Shrink/expand
- AMPI is being used in real-world parallel applications and frameworks
  - Rocket simulation at CSAR
  - FEM Framework
- Portable to a variety of HPC platforms

Future Work

- Performance Improvement
  - Reducing overhead
  - Intelligent communication strategy substitution
  - Machine-topology specific load balancing
- Performance Analysis
  - More direct support for AMPI programs

Thank You!

Download of AMPI is available at: http://charm.cs.uiuc.edu/

Parallel Programming Lab at University of Illinois