

BigSim

In a nutshell

Parallel Programming Laboratory
University of Illinois at Urbana-Champaign

BigSim Motivation

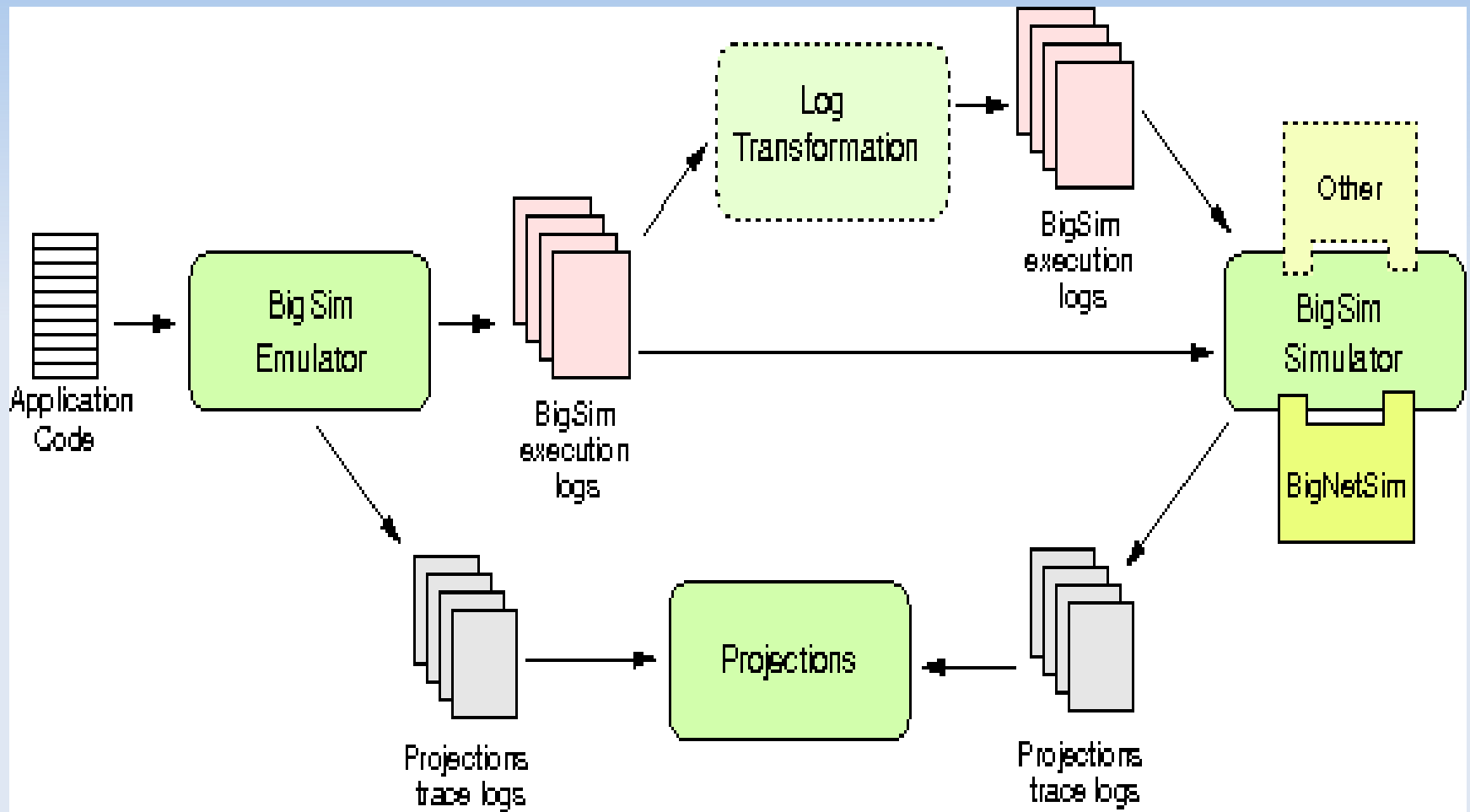
- Big machines are here, even larger machines are coming.
- How will you scale your application up a few orders of magnitude?
- How can you be ready to run when the machines become available?

Use BigSim

BigSim Objective

- Aim at developing techniques and methods to facilitate the development of efficient peta-scale applications on very large parallel machines.
- Based on performance prediction via simulation
- Performance analysis to do bottleneck discovery

Workflow Overview



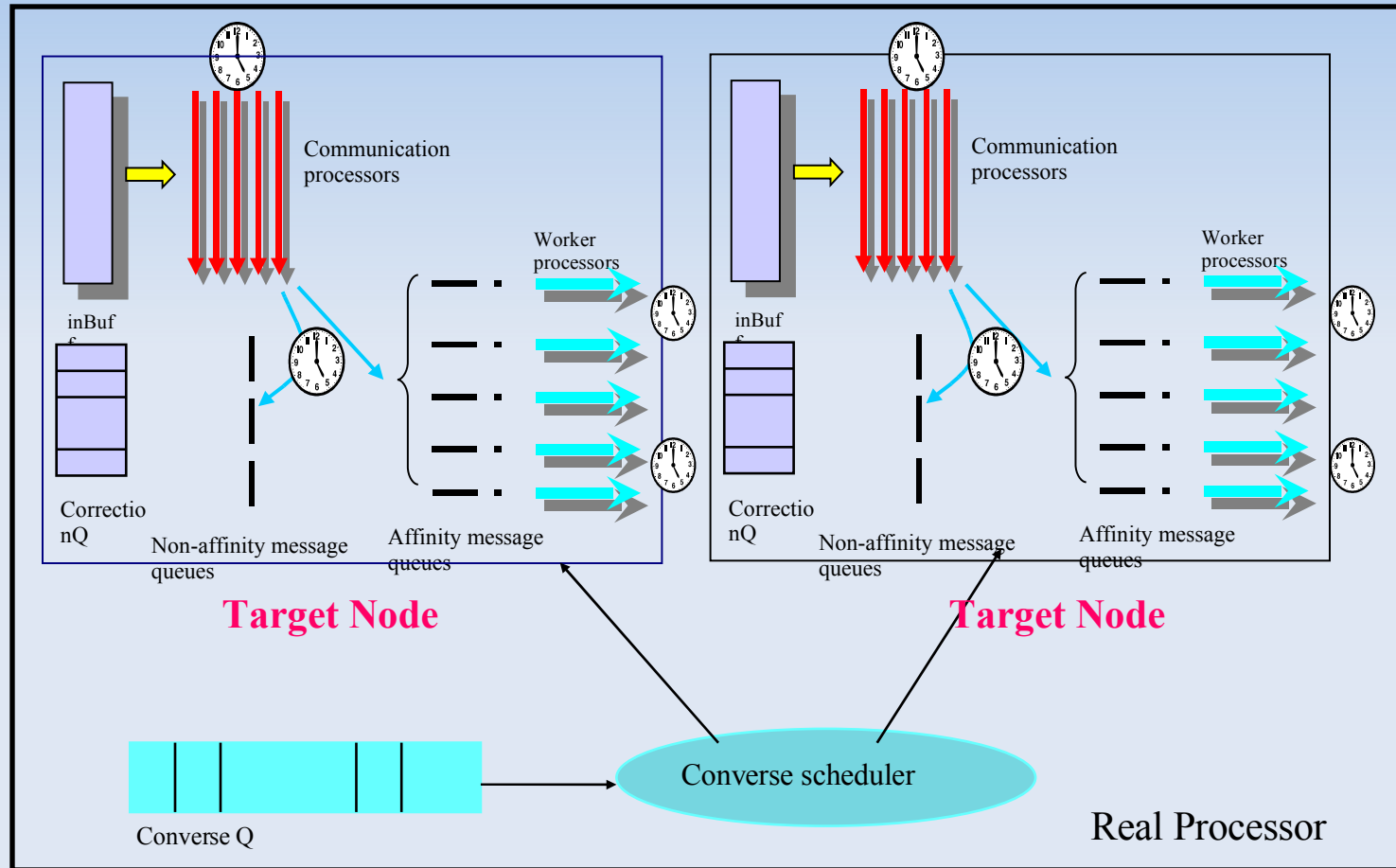
Simulation-based Performance Prediction

- With focus on Charm++ and AMPI programming models
- Performance prediction is based on Parallel Discrete Event Simulation (PDES)
- Simulation is challenging, aims at different levels of fidelity
 - Processor prediction
 - Network prediction
- Two approaches
 - Direct execution (online mode)
 - Trace-driven (post-mortem mode)

Emulator

- Emulate full machine on existing parallel machines
 - Actually run a parallel program with multi-million way parallelism
- Started with mimicking Blue Gene/C low level API
- Machine layer abstraction
 - Many multiprocessor (SMP) nodes connected via message passing

BigSim Emulator: functional view



BigSim Charm++/AMPI

- Need high level programming language such as Charm++/AMPI
- Charm++/AMPI implemented on top of BigSim emulator, using it as another machine layer
- Support frameworks and libraries
 - Load balancing framework
 - Communication optimization library (comlib)
 - FEM
 - Multiphase Shared Array (MSA)

Performance Prediction

- How to predict performance?
 - Different levels of fidelity
 - Processor model:
 - User supplied timing expression
 - Interpolated timing from sequential simulator
 - Wallclock time
 - Performance counters
 - Interpolation from instruction level simulation
 - Network model:
 - Simple latency-based network model
 - Contention-based network simulation

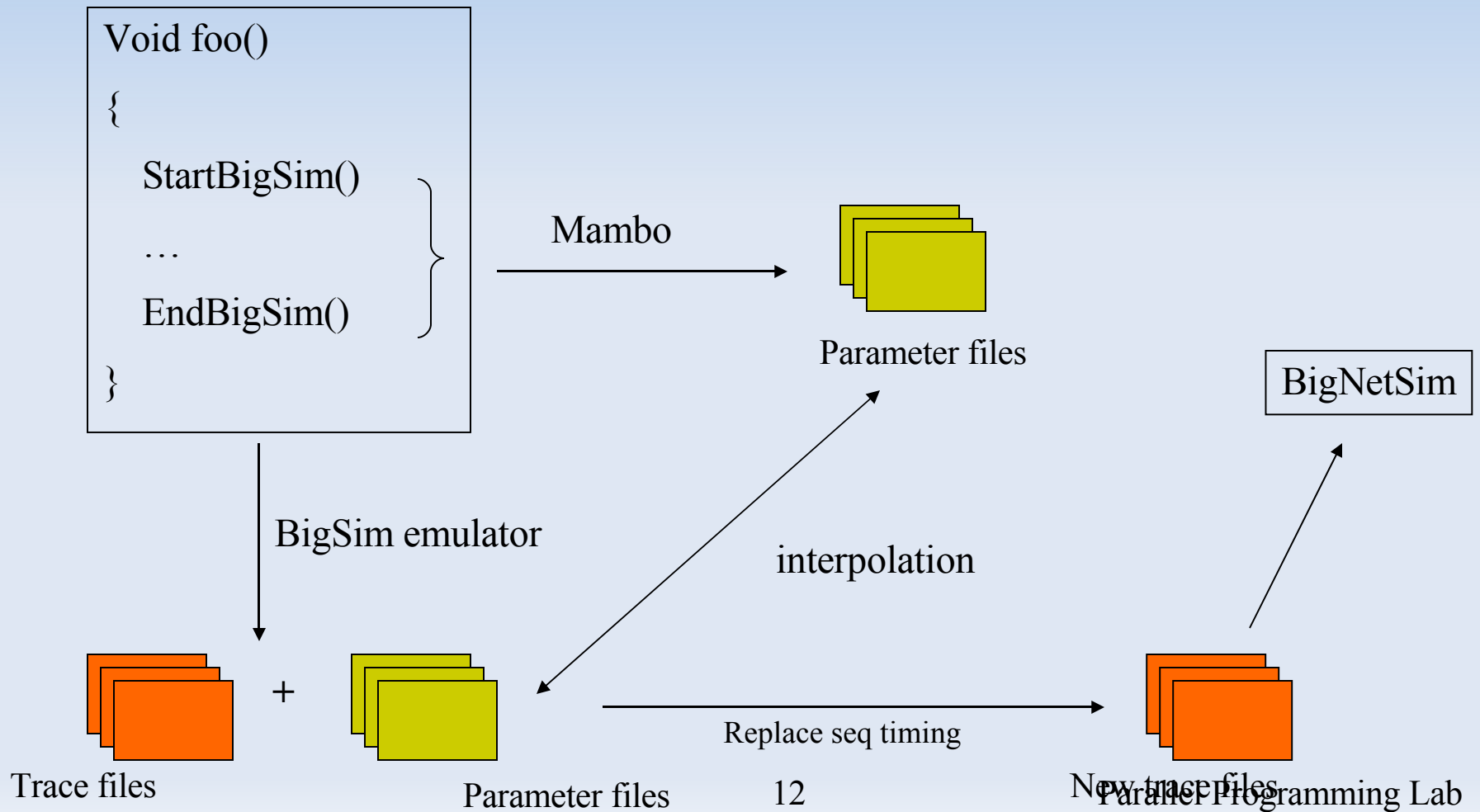
How to Ensure Simulation Accuracy

- The idea:
 - Take advantage of **inherent determinacy** of an application
 - Don't need rollback - same user function then is executed only once
 - In case of out of order delivery, only timestamps of events are adjusted
 - Integrate results from other simulators
 - Use Post-mortem simulation for correction and accuracy improvement

Integration with Mambo simulation

- What we have:
 - BigSim emulator produce trace files
 - BigNetSim simulates parallel performance
 - IBM Mambo cycle accurate simulator predicts execution time for sequential blocks of code
- To integrate
 - We use Mambo predicted time to replace the sequential timing in trace files

Mambo Simulator Workflow



Postmortem Simulation

- Run application once, get trace logs
- Apply log transformations
- Run simulation with logs for a variety of network configurations
- Implemented on POSE simulation framework

Output

- Completion time for trace run
- Turn on `-tproj` to get simple updated trace of network performance
- POSE trace for projections output
 - limited value to end user
- Projections output displaying user events in simulation time

BigNetSim Modules

- Build your own interconnect
 - Architecture
 - Bluegene
 - KaryNmesh
 - OB Output Buffered
 - IB Input Buffered
 - Virtual Channel selection
 - Topology
 - 3D Mesh
 - FatTree
 - Torus
 - HyperCube
 - Routing
 - Adaptive
 - Static
 - Hybrid
 - Fat Tree + Torus

Traffic Generation

- Generate traffic patterns instead of using trace files

- additional command line parameters

- Pattern
- Frequency

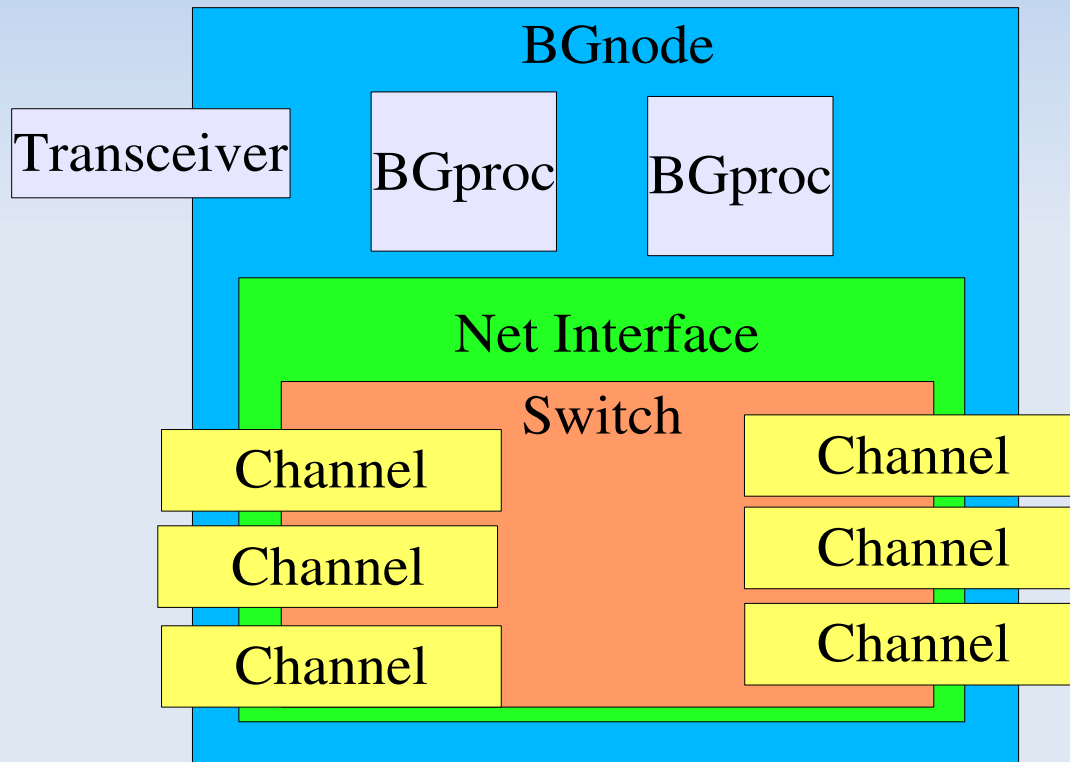
- Pattern

- 1 kshift
- 2 ring
- 3 bittranspose
- 4 bitreversal
- 5 bitcomplement
- 6 poisson

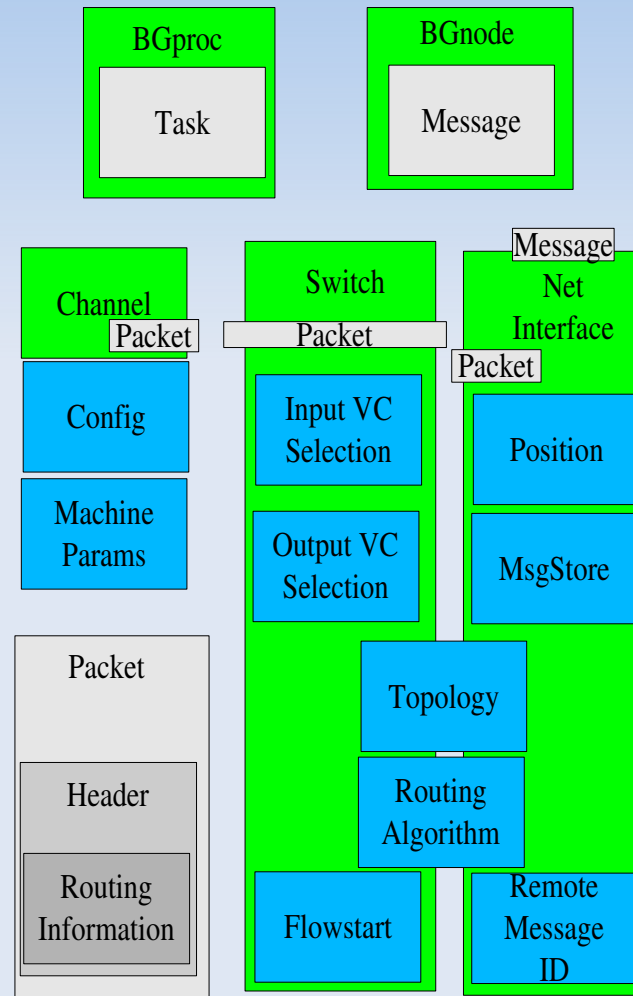
- Frequency

- 0 linear
- 1 uniform
- 2 exponential

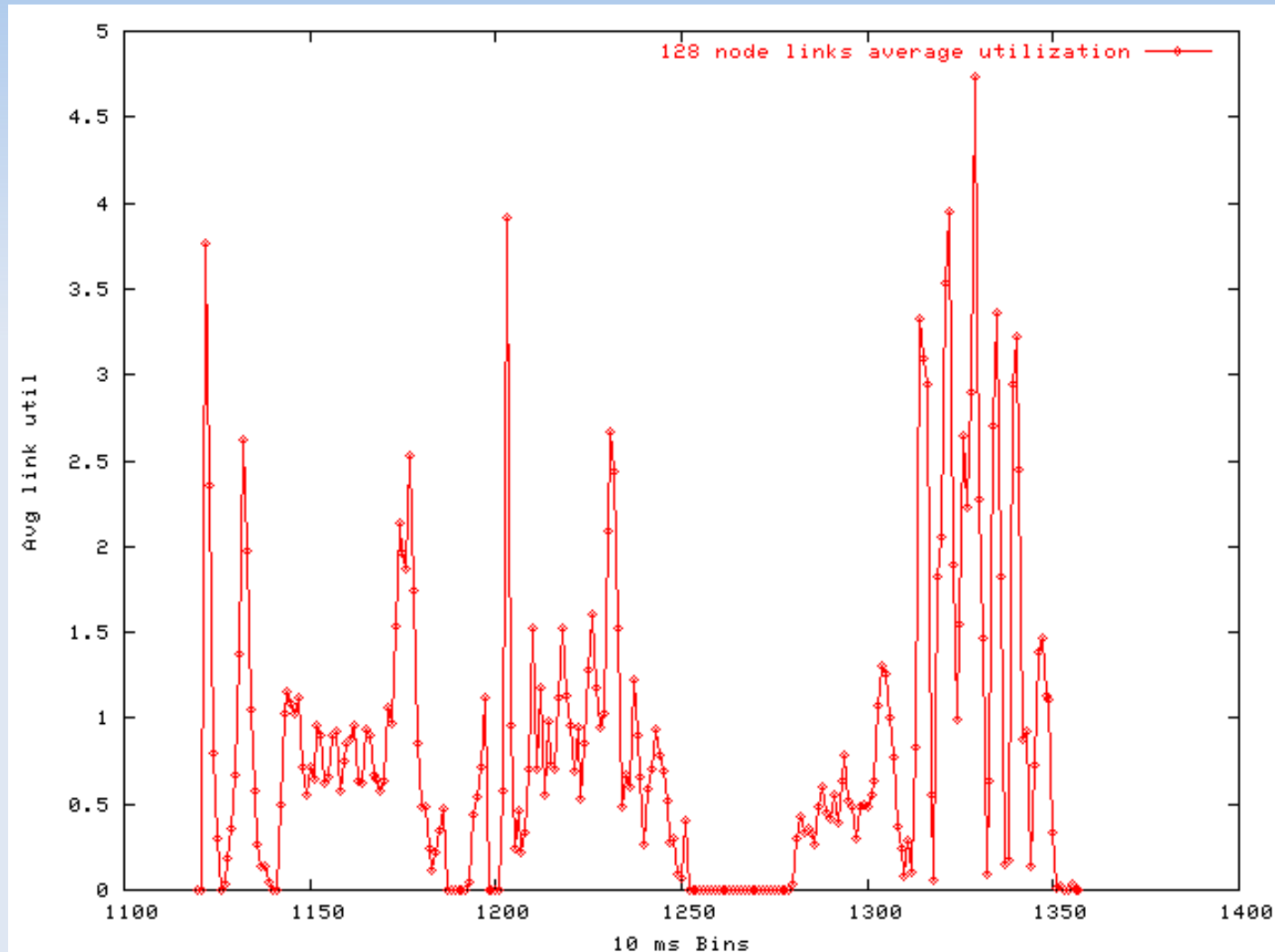
BigNetSim Design



BigNetSim API: Extensibility

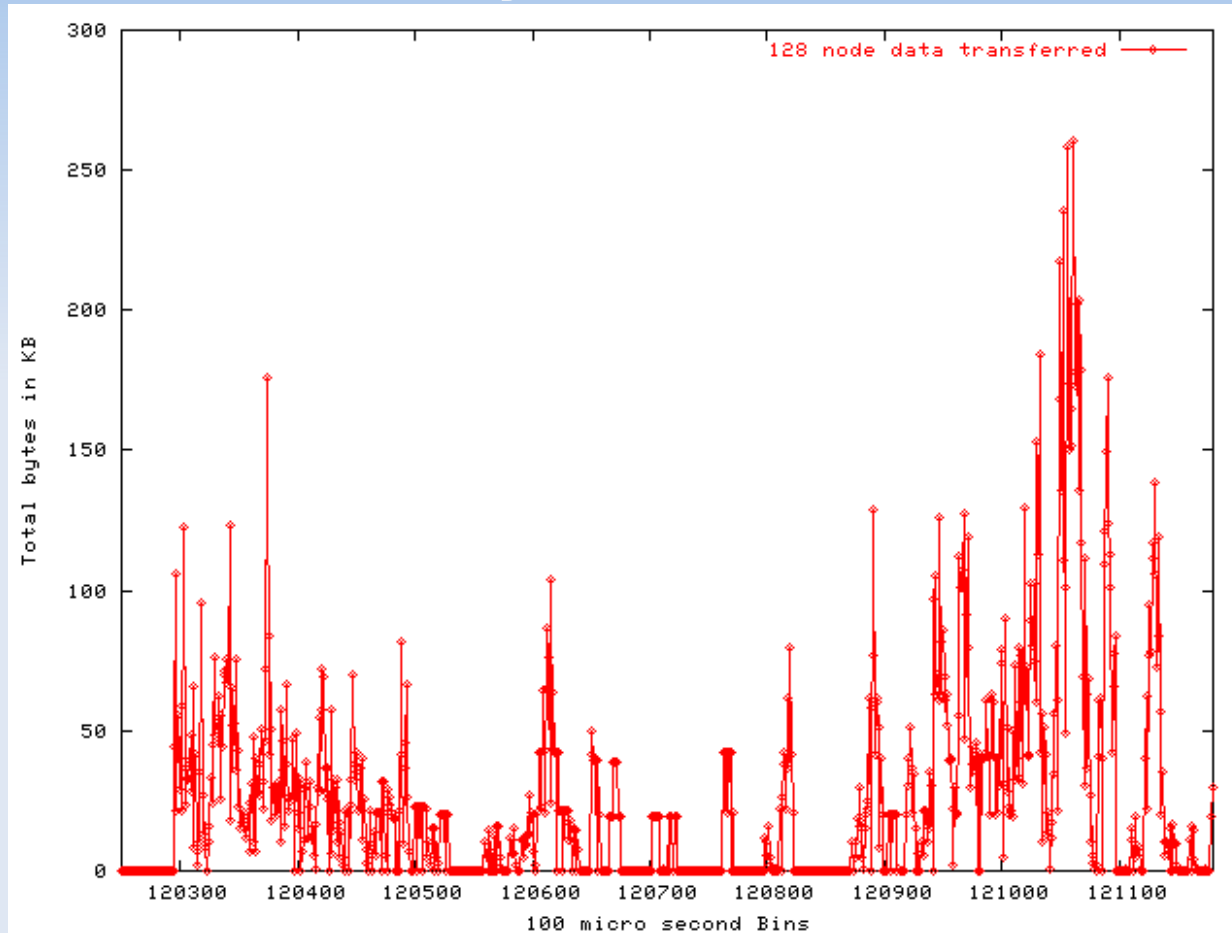


Network Communication Pattern Analysis



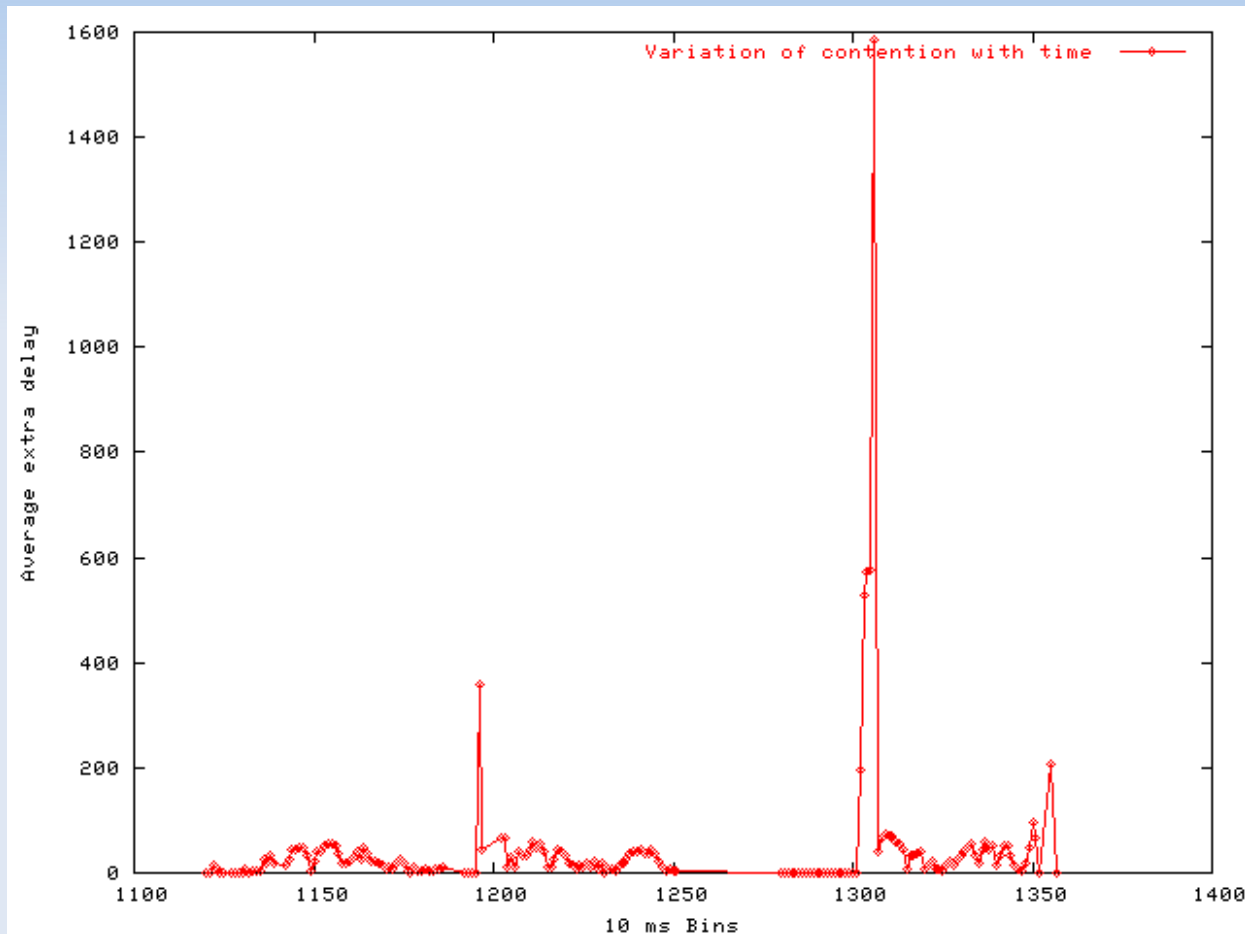
- NAMD with apoa1
- 15 timestep

Network Communication Pattern Analysis



Data transferred (KB) in a single time step

Contention Encountered by Messages





Make bgtest
With 16 processors

Thank You!

Free download of Charm++ and
BigSim at

<http://charm.cs.uiuc.edu>

Send comments to
ppl@charm.cs.uiuc.edu