SAND2011-4789P

Structural Simulation Toolkit

Arun Rodrigues

JOWOG 2011

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

View of the Simulation Problem

Scale.....

Many Cores + Memory	X	Many Many Nodes	X	Many Many Many Threads

Multiple Audiences.....

Network Processor System	X	Application writers purchasers designers	X	system procurement algorithm co-design architecture research language research	X	present systems future systems

Complexity.....

Multi-Physics Apps Informatics Apps	X	Communication Libraries Run-Times OS Effects	X	Existing Languages New Languages

Constraints.....

Performance Cost	Power Reliability	Cooling Usability	Risk Size

NNSA National Nuclear Security Administration

Sandia National Laboratories

SST Simulation Project Overview

Goals

- Become the standard architectural simulation framework for HPC
- Be able to evaluate future systems on DOE workloads
- Use supercomputers to design supercomputers

Status

- Current Release (2.1) at code.google.com/p/sst-simulator/
- Includes parallel simulation core, configuration, power models, basic network and processor models, and interface to detailed memory model

Technical Approach

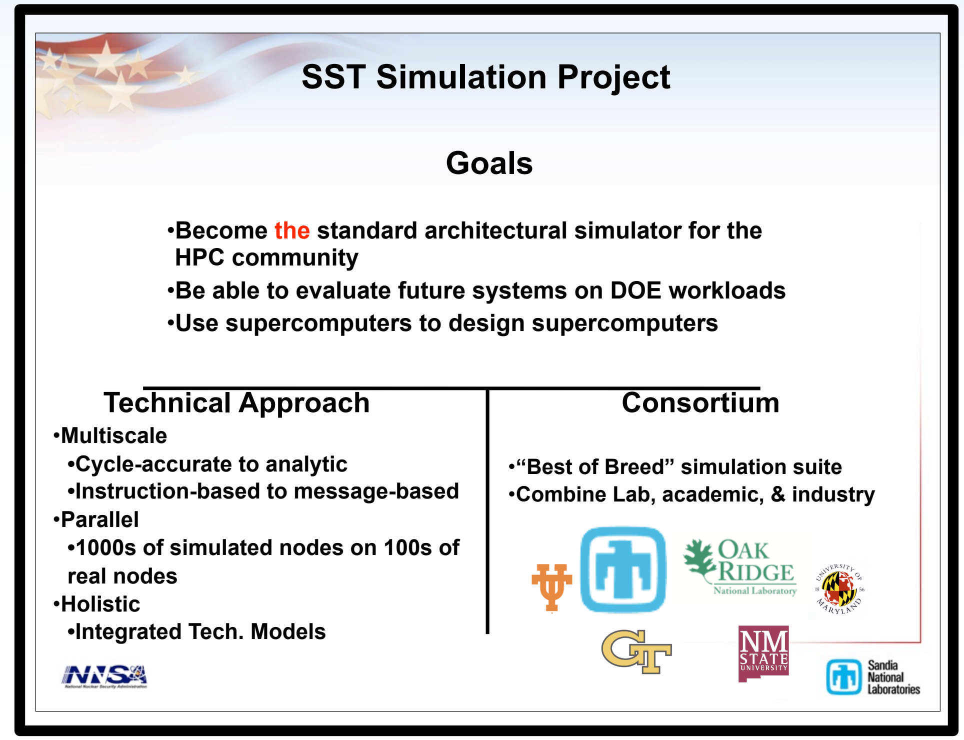
- Parallel
  - Parallel Discrete Event core with conservative optimization over MPI
- Holistic
  - Integrated Tech. Models for power
  - McPAT, Sim-Panalyzer
- Multiscale
  - Detailed and simple models for processor, network, and memory
- Open
  - Open Core, non viral, modular

Consortium

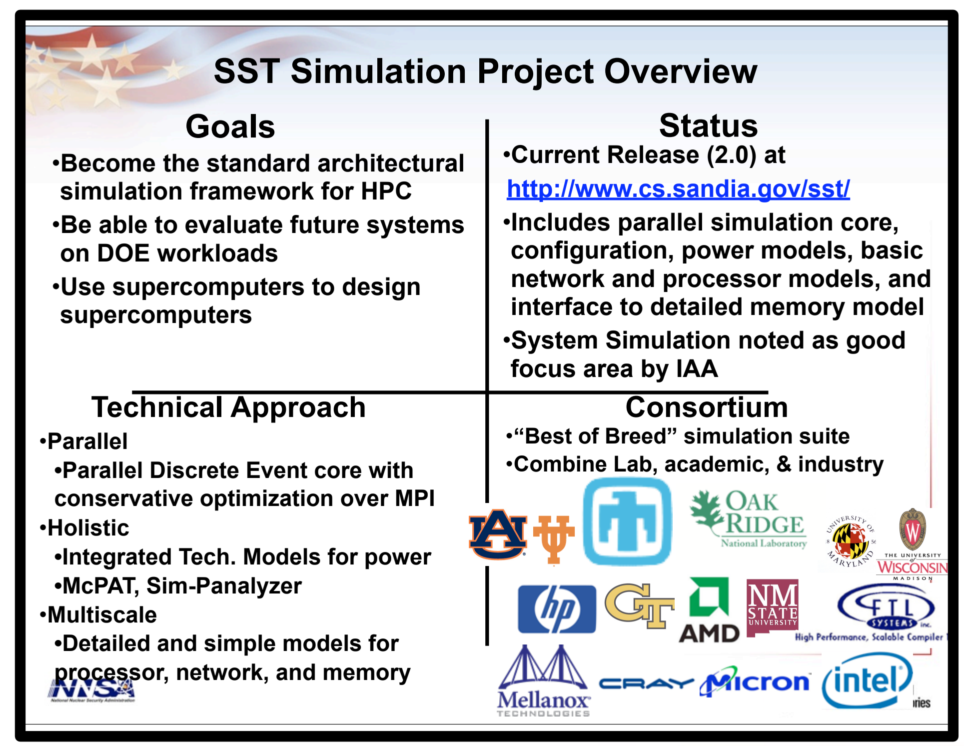
- "Best of Breed" simulation suite
- Combine Lab, academic, & industry

Progress

- Since 2010...
- Improved Parallel Core
  - Better time handling
- Build System
  - Improved
  - Documented!
- More Technology Models
  - Reliability
  - Thermal Effects (Hotspot)
- Components
  - M5/GeM5
  - State Machines
  - Disk Sim
  - eBOBSim



2009 JOWOG SST Overview



2010 JOWOG SST Overview

Parallel Implementation

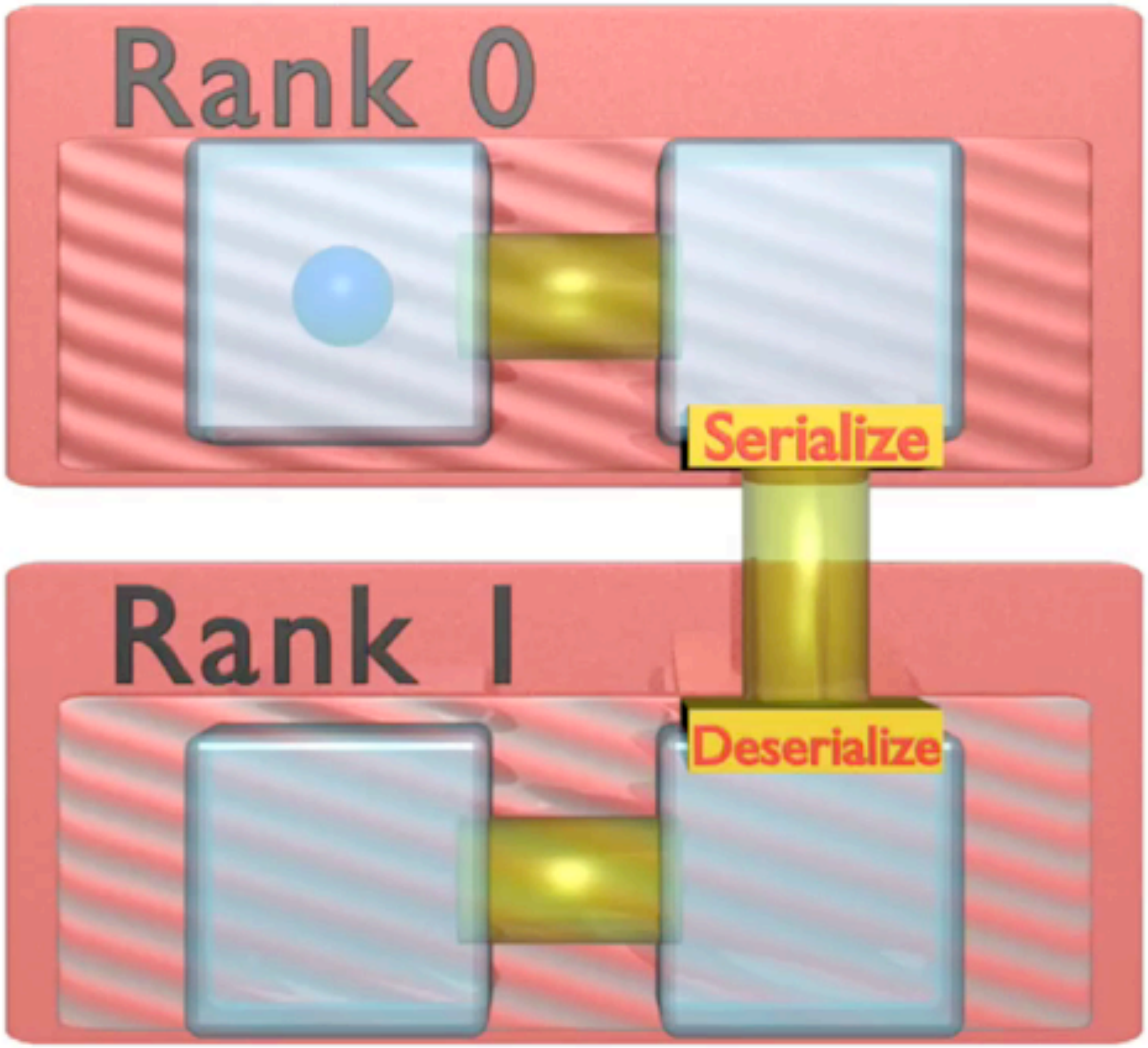
- Implemented over MPI
- Configuration, partitioning, initialization handled by core
- Conservative, distance-based optimization

Parallel Implementation

- Implemented over MPI
- Configuration, partitioning, initialization handled by core
- Conservative, distance-based optimization

Message Handling

- SST core transparently handles message delivery
- Detects if destination is local or remote
- Local messages delivered to local queues
- Remote messages stored for later serialization and remove delivery
  - Boost Serialization Library used for message serialization
  - MPI used for transfer
- Ranks synchronize based on partitioning



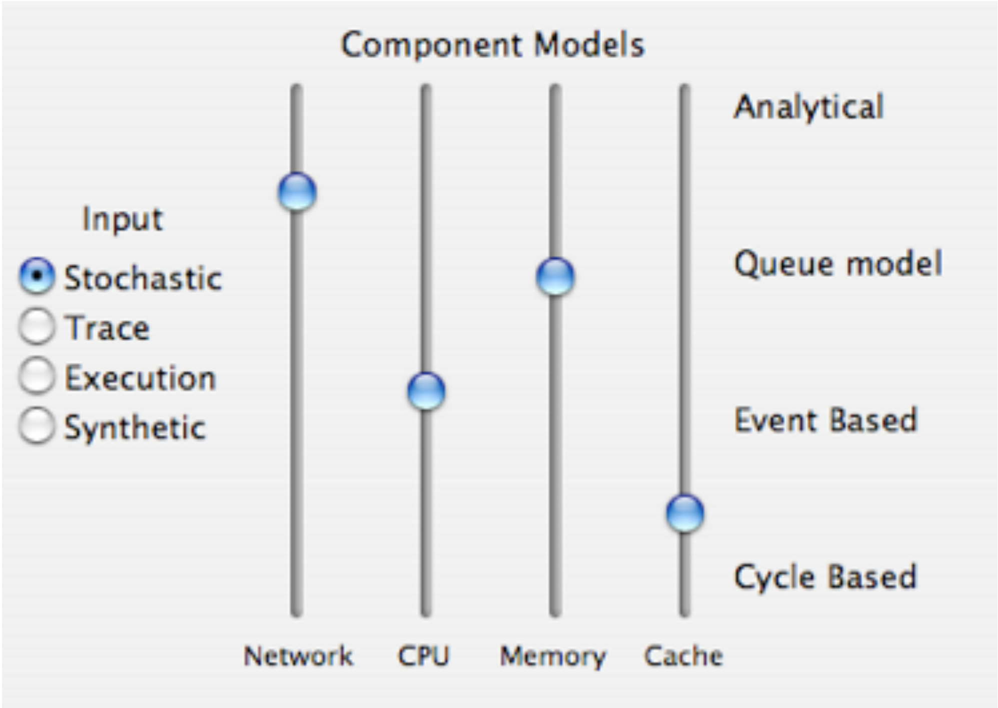
National Nuclear Security Administration

Sandia National Laboratories

Multi-Scale

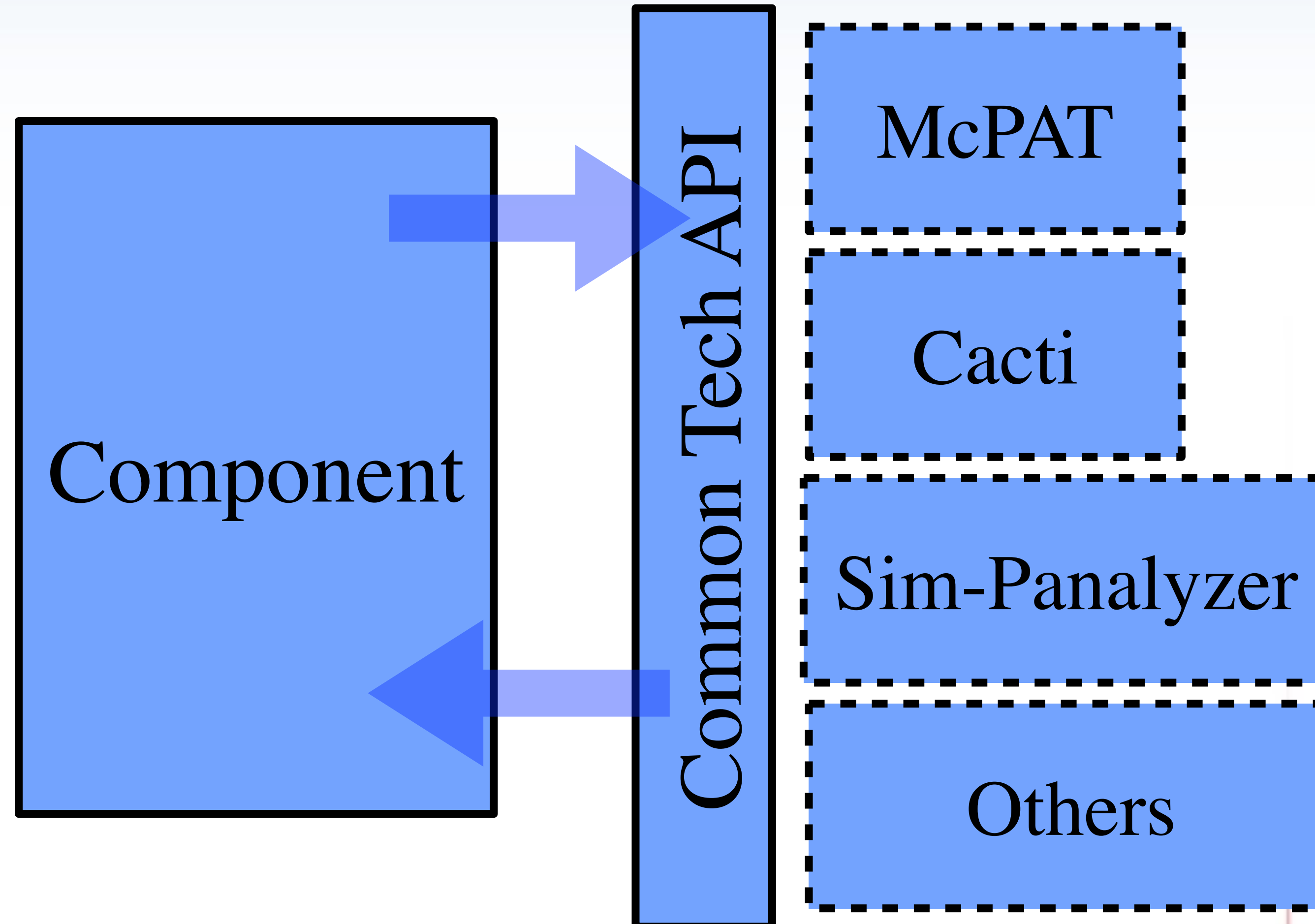
- Goal: Enable tradeoffs between accuracy, flexibility, and simulation speed
  - No single "right" way to simulate
  - Support multiple audiences
- High- & Low-level interfaces
  - Allows multiple input types
  - Allows multiple input sources
    - Traces, stochastic, state-machines, execution...

	High-Level	Low-Level
Detail	Message	Instruction
Fundamental Objects	Message, Compute block, Process	Instruction, Thread
Static Generation	MPI Traces, MA Traces	Instruction Trace
Dynamic Generation	State Machine	Execution



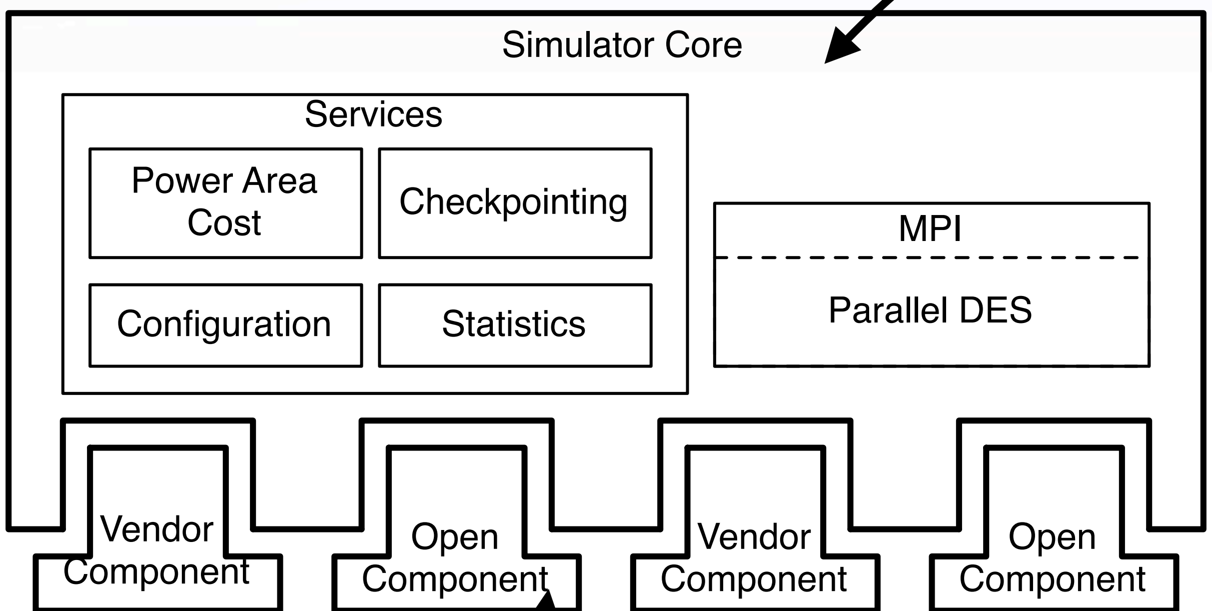
Multiscale Parameters

Holistic Simulation



- Design space includes much more than simple performance
- Create common interface to multiple technology libraries
  - Power/Energy
  - Area/Timing estimation
- Make it easier for components to model technology parameters

Open Simulator Framework



- Simulator Core will provide...
  - Power, Area, Cost modeling
  - Checkpointing
  - Configuration
  - Parallel Component-Based Discrete Event Simulation
- Components
  - Ships with basic set of open components
  - Industry can plug in their own models
    - Under no obligation to share
- Open Source (BSD-like) license
- SVN hosted on Google Code

Sandia National Laboratories

Component Validation

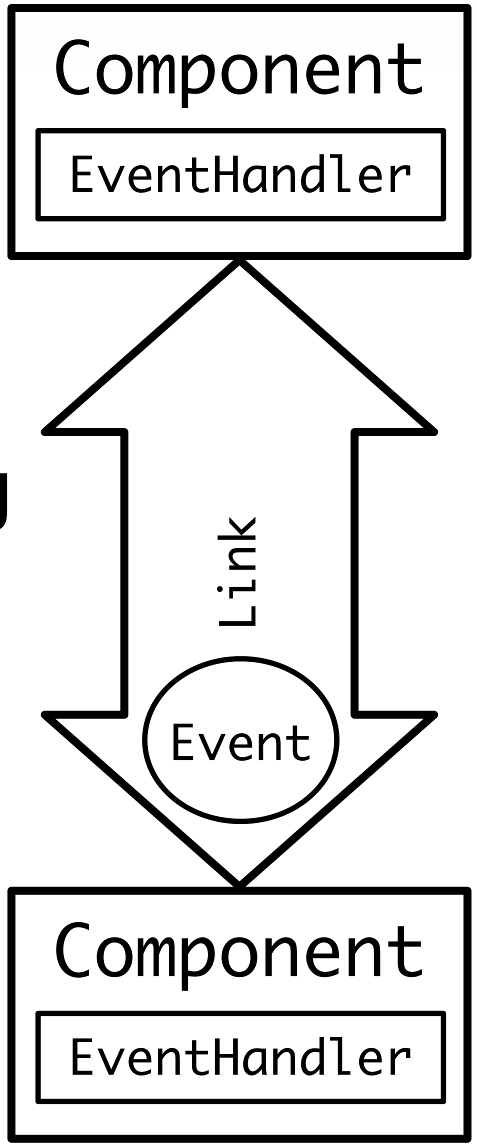
- Strategy: component validation in parallel with system-level validation
- Current components validated at different levels, with different methodologies
- Validation in isolation

- What is needed
  - Uniform validation methodology (apps)
  - System (multi-component) level validation

Component	Method	Error
DRAMSim	RTL Level validation against Micron	Cycle
Generic Proc	Simplescalar SPEC92 Validation	~5%
NMSU	Comparison vs. existing processors on SPEC	<7%
RS Network	Latency/BW against SeaStar 1.2, 2.1	<5%
MacSim	Comparison vs. Existing GPUs	Ongoing <10% expected
Zesto	Comparison vs several processors, benchmarks	4-5%
McPAT	Comparisons against existing processors	10-23%

Key Objects & Interfaces

- **Goal: Simplicity**
- **Objects**
  - SST::Component**: A model of a hardware component**
  - SST::Link**: A connection between two components**
  - SST::Event**: A discrete Event**
  - SST::EventHandler**: Function to handle an incoming event or clock tick**
- **Events**
  - SST::Component::ConfigureLink()**: Registers a link and (optionally) handler**
  - SST::Link::Recv()**: Pull an event from a link**
  - SST::Link::Send()**: Send an event down a link**
  - SST::Component::registerClock()**: Register a clock and handler**

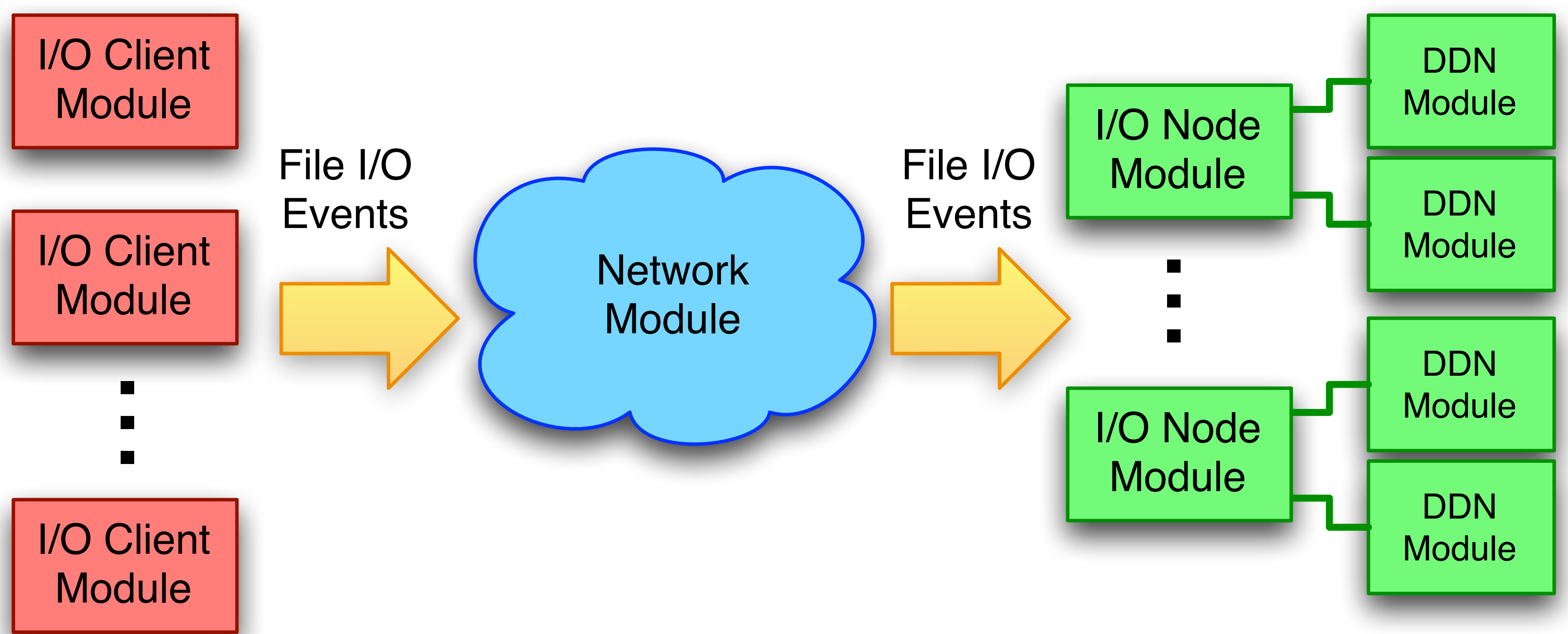


New Capabilities

Disk & I/O Modeling

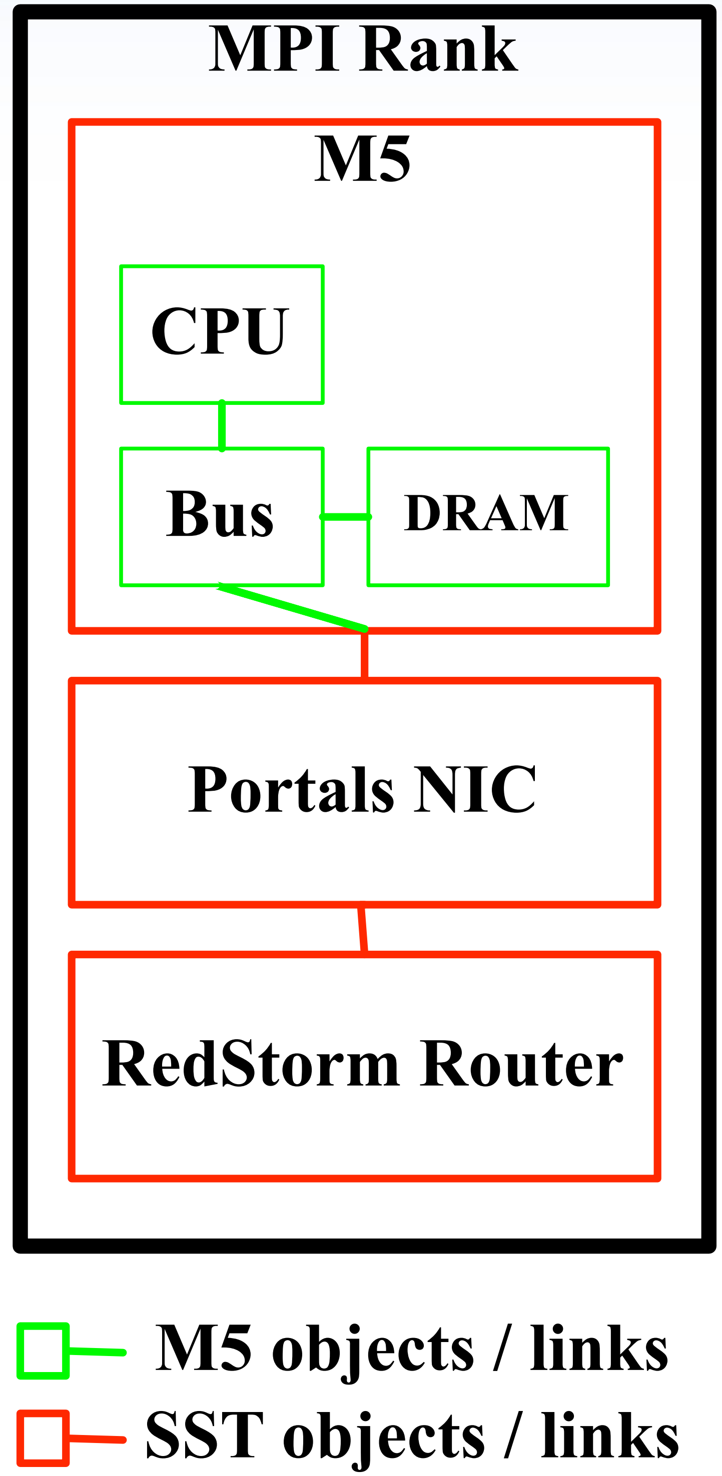
- Goal:Create a file system simulator that can simulate real-world application IO on HPC machines, including simulating various disk models and controllers, IO nodes, networking, metadata servers, and clients

- Current Capabilities
  - Can run lua scripts or tau traces of real applications and simulate a single disk
  - Has been verified against known benchmarks and does simulate single disks



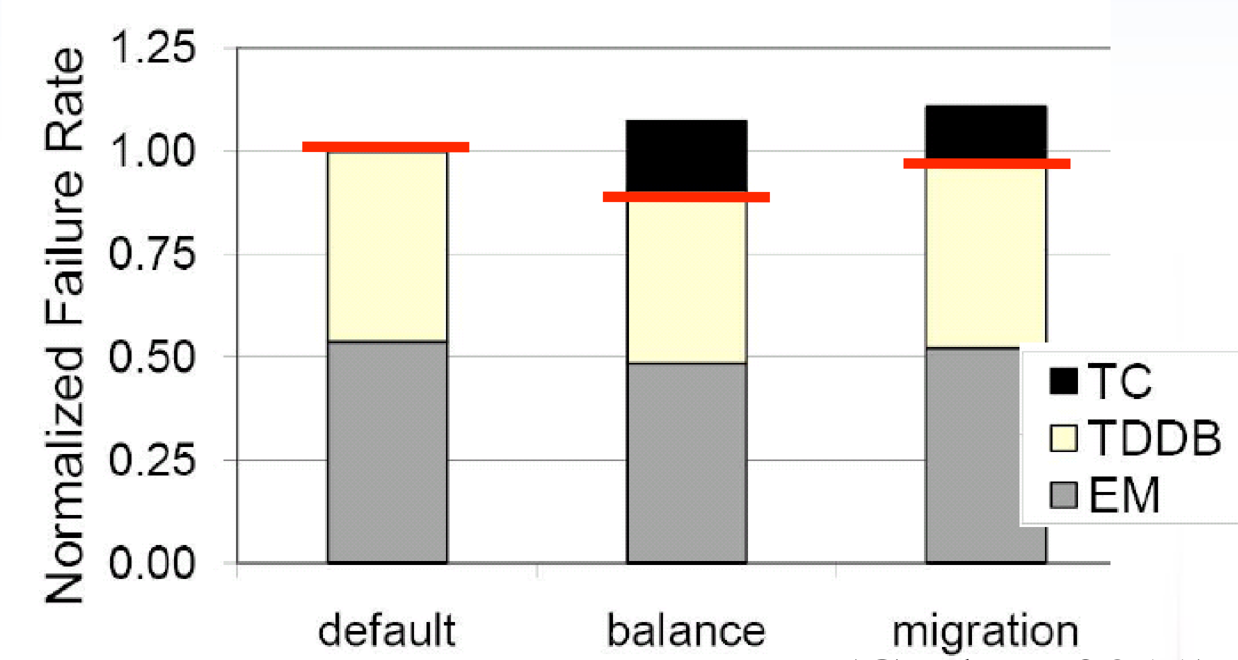
M5

- M5: Modular platform for computer system architecture research, encompassing system-level architecture as well as processor microarchitecture.
- Provides detailed, full-system CPU models for x86, ARM, SPARC, Alpha
- Integrated at SST Component, allows interaction with SST models, and parallel execution
- Currently tested up to 256 nodes.



Case Study: Reliability vs. Power
Hidden cost of DVFS

- Dynamic voltage/frequency Scaling reduces power
- →Reduces temperature
- →Causes thermal cycling
- →Reduces reliability

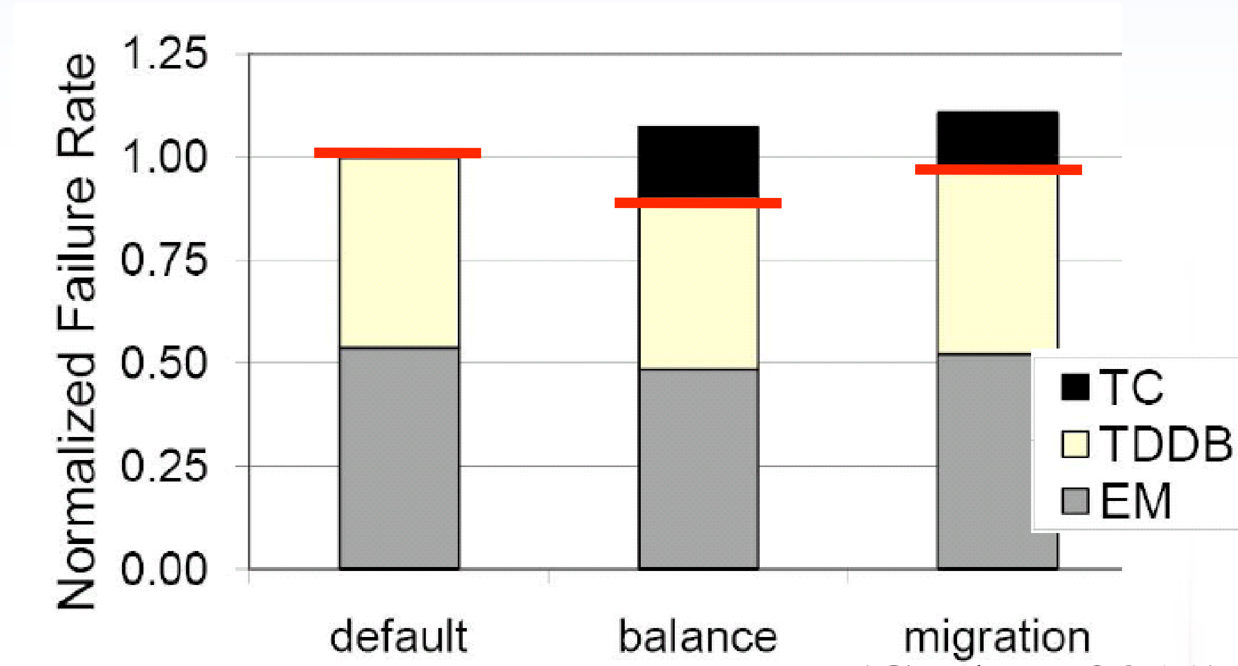


(Coskun 2011)

- Need
  - –Algorithms to balance temperature, lower power, & maintain performance
  - –Arch: Sensors and feedback
  - –Runtime: Scheduler changes
  - –App: Awareness

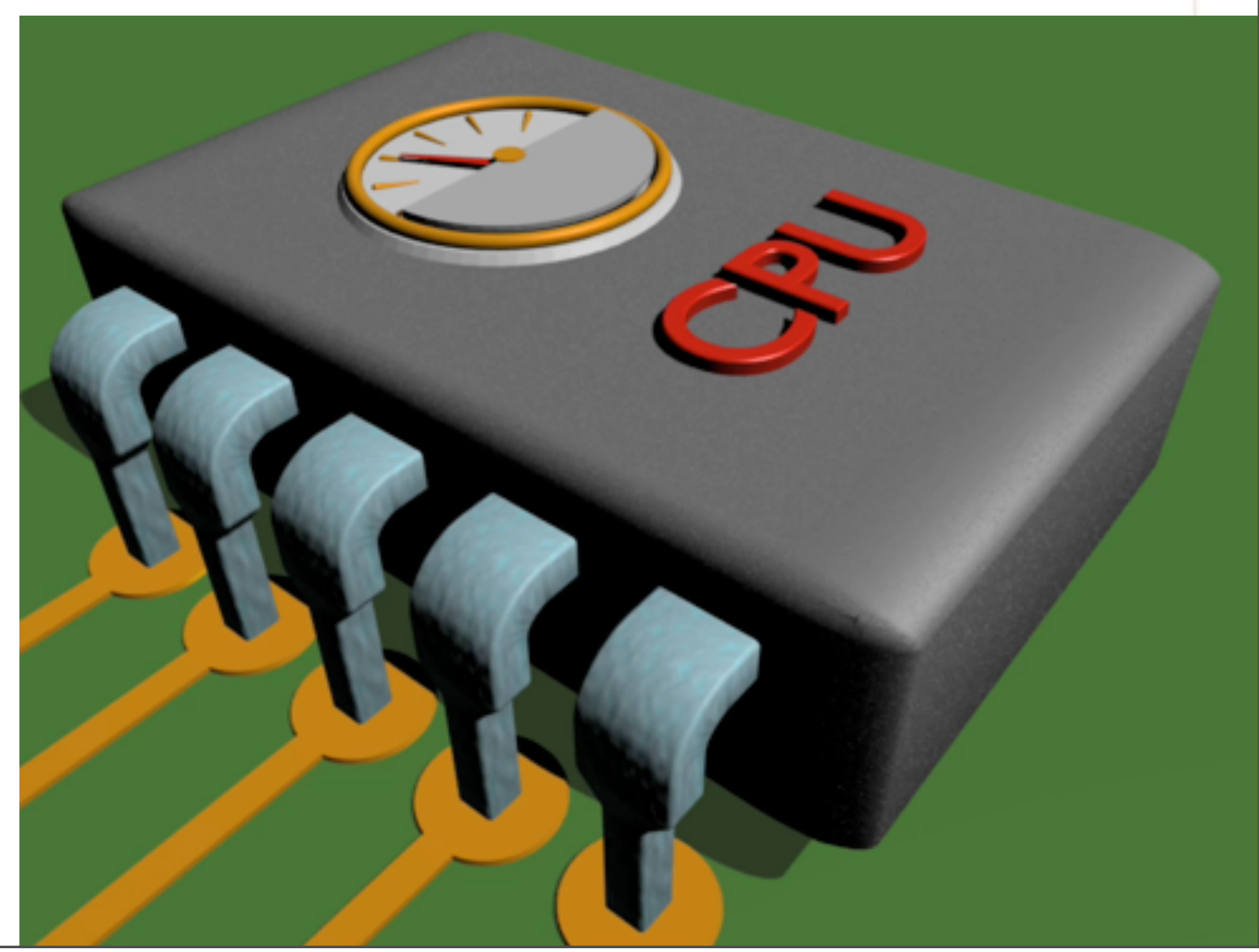
Case Study: Reliability vs. Power
Hidden cost of DVFS

- **Dynamic voltage/frequency Scaling reduces power**
- **→Reduces temperature**
- **→Causes thermal cycling**
- **→Reduces reliability**



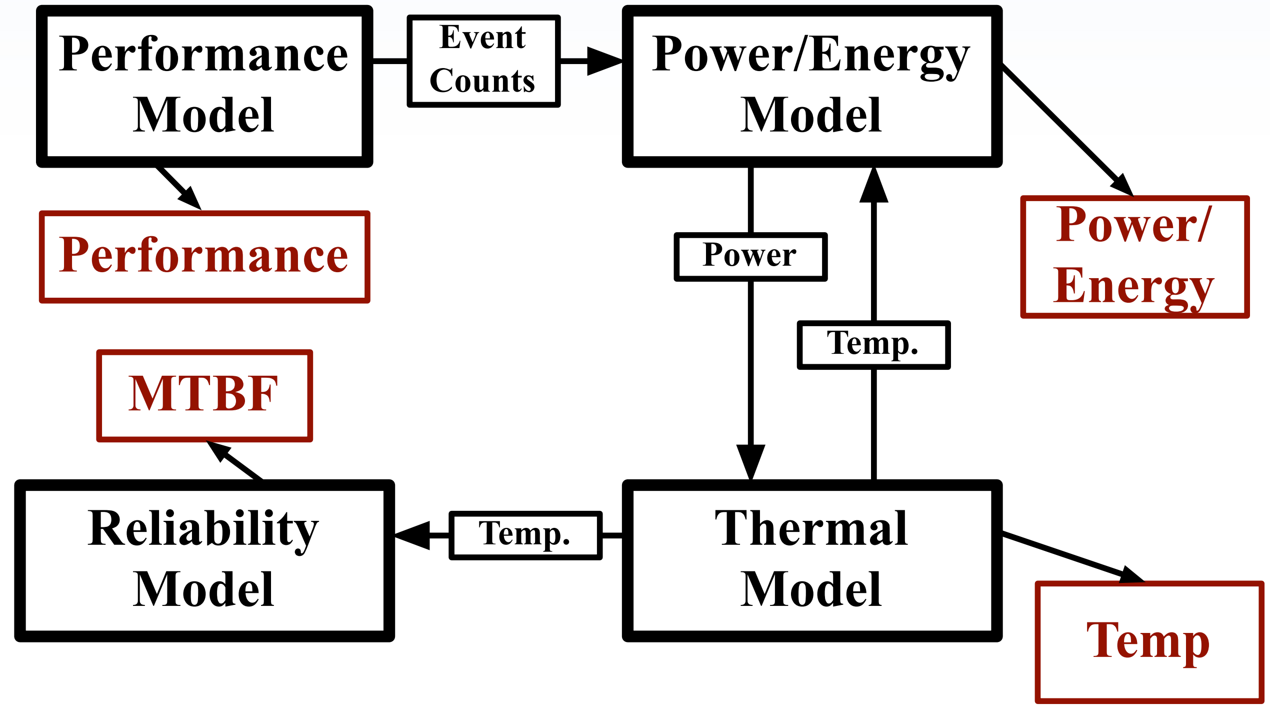
(Coskun 2011)

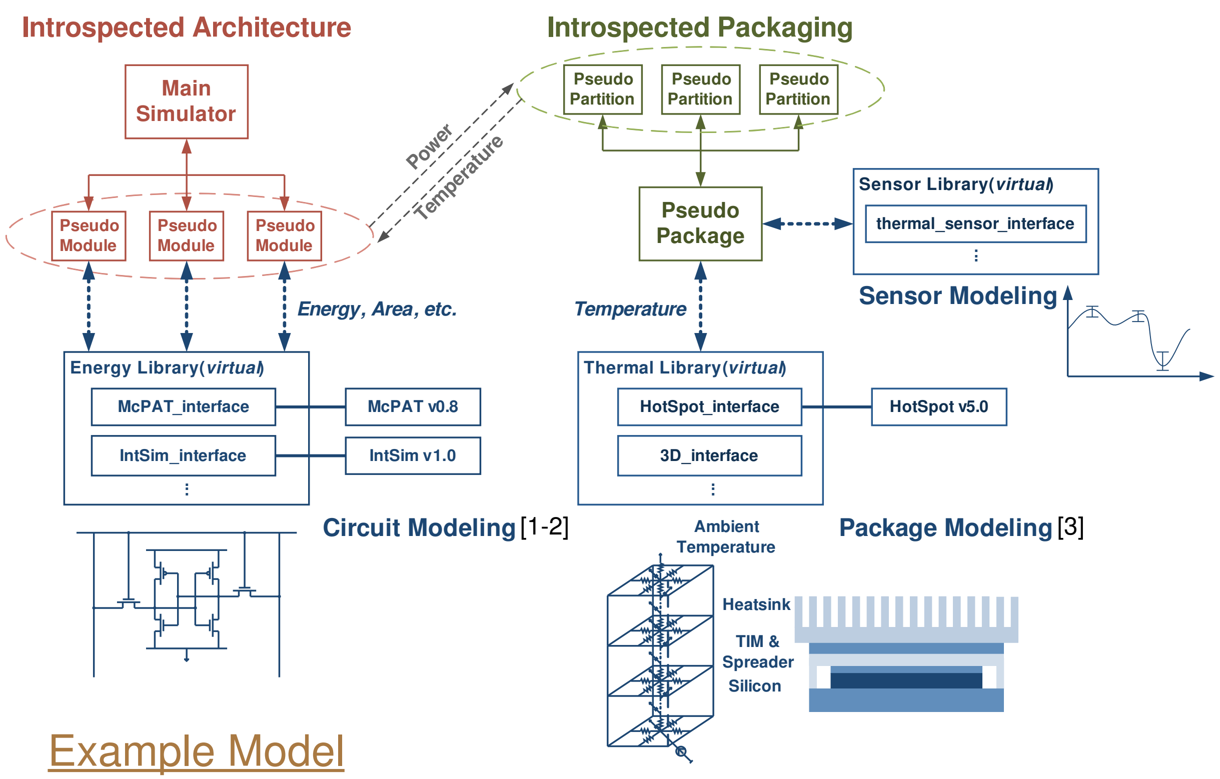
- **Need**
  - **–Algorithms to balance temperature, lower power, & maintain performance**
  - **–Arch: Sensors and feedback**
  - **–Runtime: Scheduler changes**
  - **–App: Awareness**



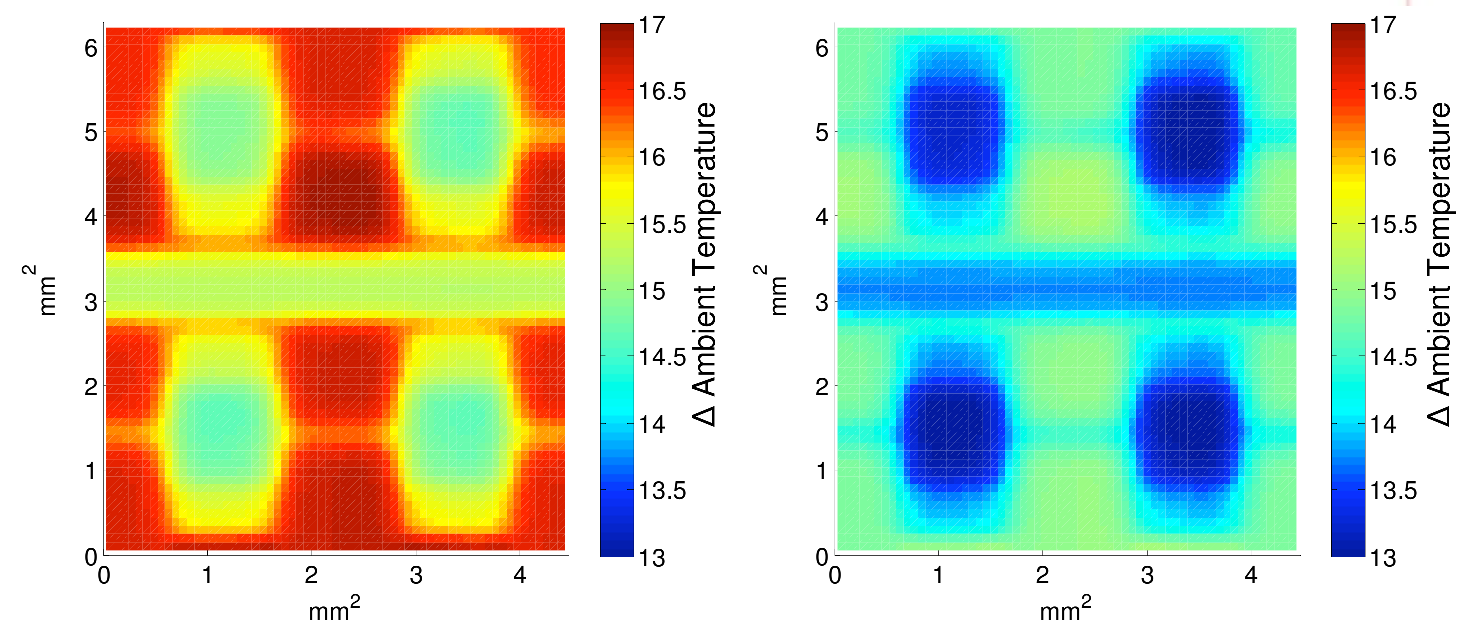
Thermal Modeling

- Unified infrastructure to model temperature
- Area Model provides partitioning information
- Thermal model provides temperature information to help calculate leakage





Example Model

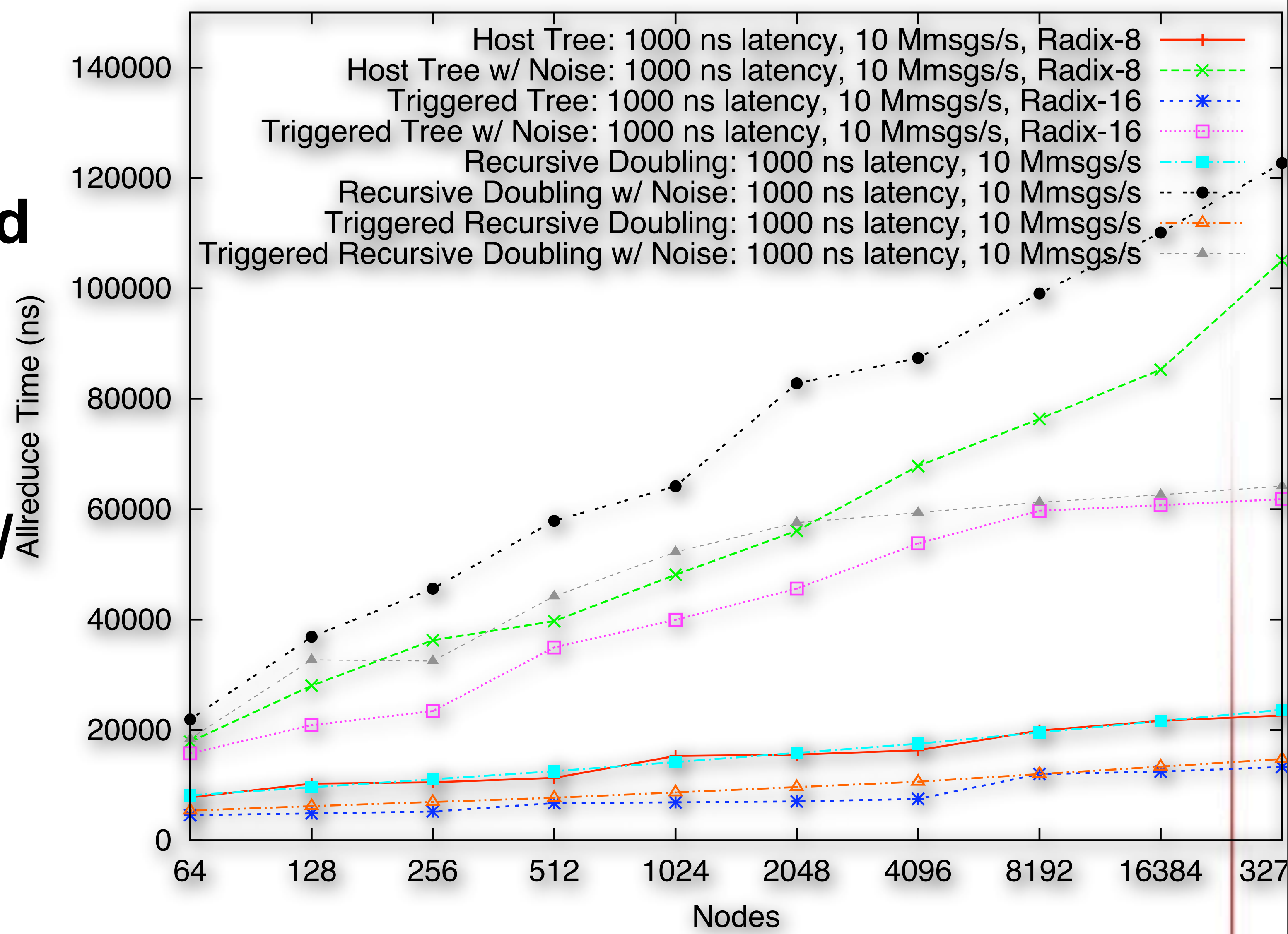


(a) Initial temperature of 313K with leakage feedback

(b) Initial temperature of 313K without leakage feedback

Network State Machines

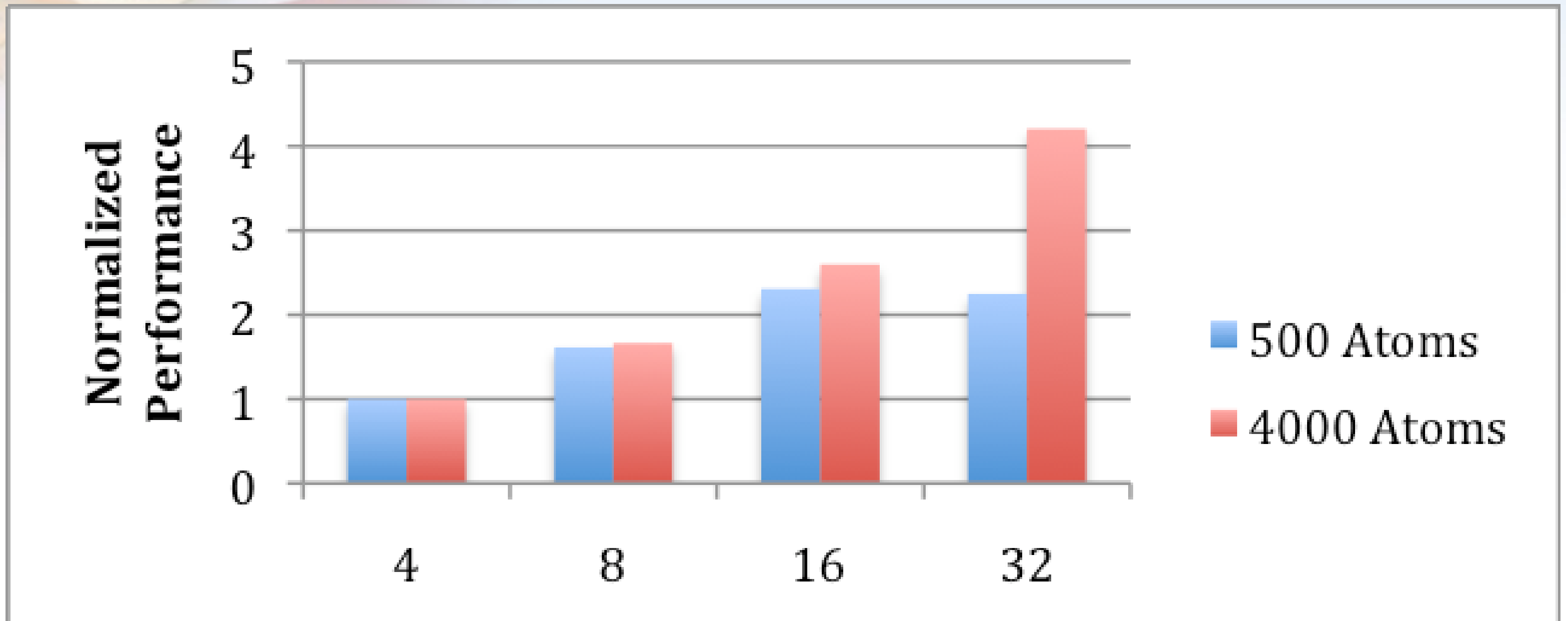
- Interconnect simulation to explore offload of collective operations in the presence of noise
  - Up to 32K nodes simulated
  - Flit-level router based on Red Storm
  - Simplified NIC exposing Portals 4.0 interface
  - State-machine-based CPU w/ different noise injection patterns
- Conclusions
  - Noise impact is very Algorithm dependent (e.g. Recursive Doubling)
  - Offload of collectives can help



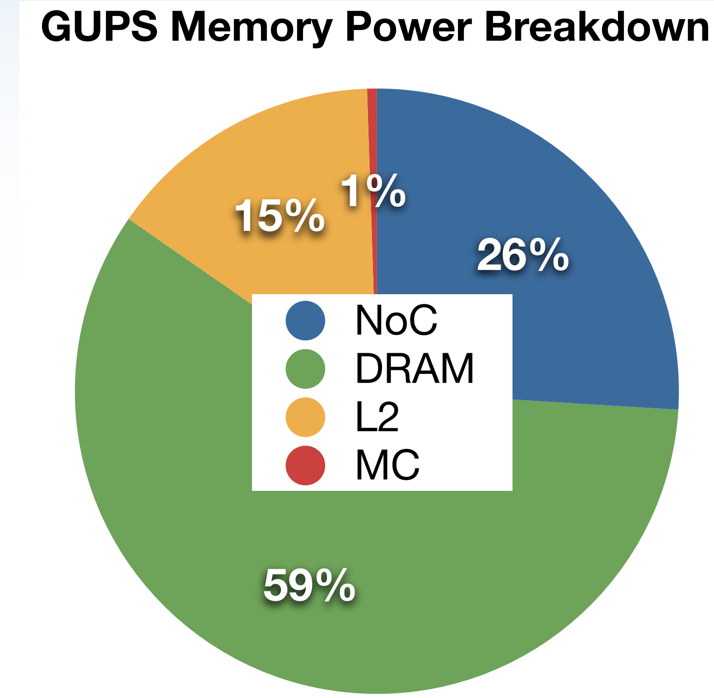
NNSA National Nuclear Security Administration

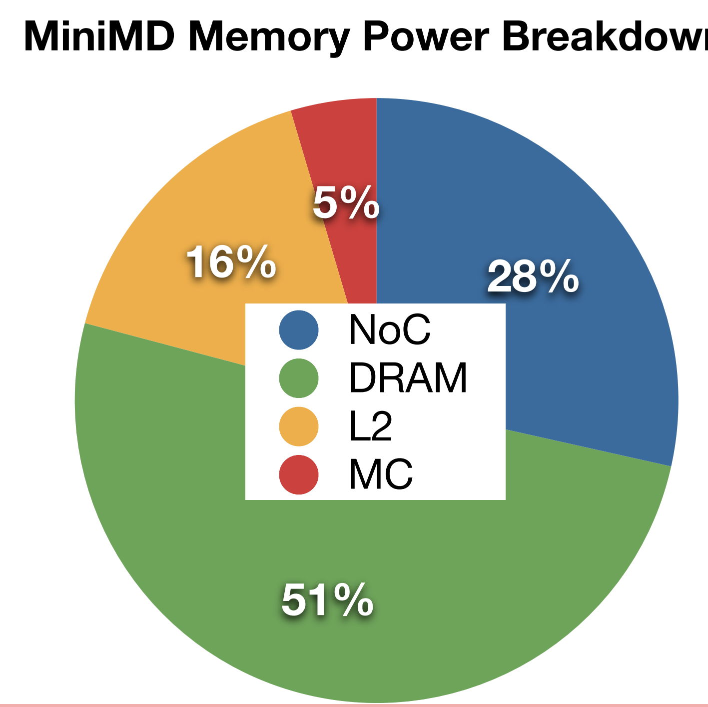
Sandia National Laboratories

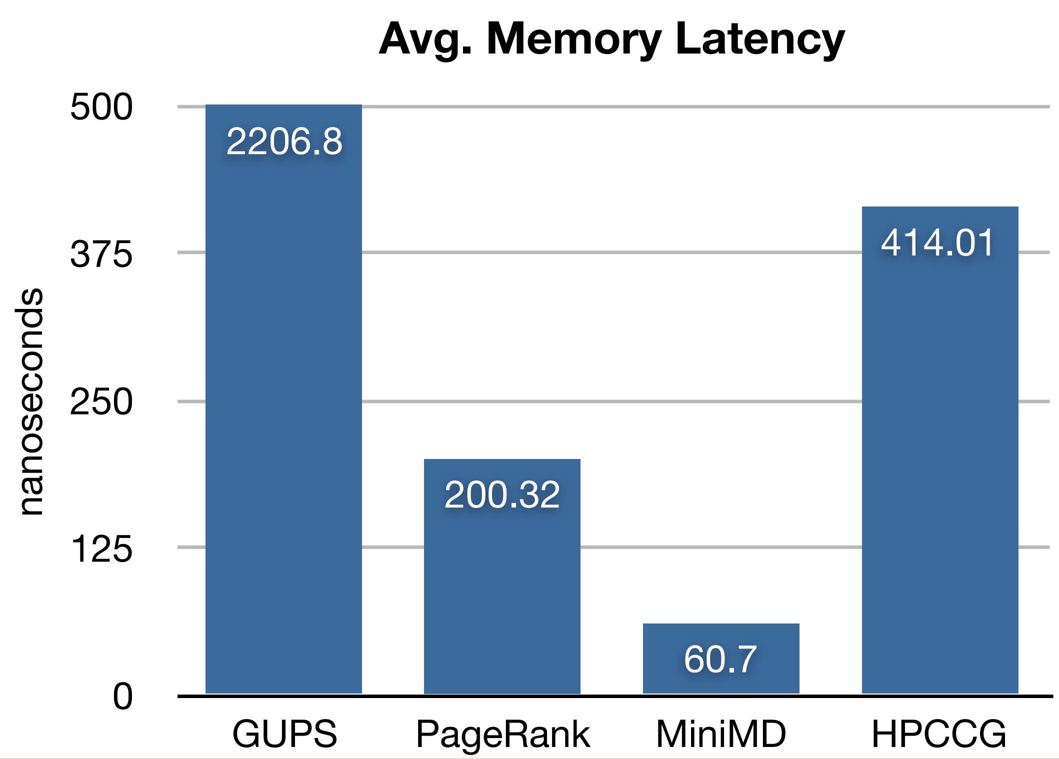
Sample Results & Uses



SST Simulation of MD code shows diminishing returns for threading on small data sets







Detailed component simulation highlights bottlenecks

Power analysis help prioritize technology investments

Current/Future Work

- Memory Models
  - Non-Volatile
  - Stacked
- 'Meso Scale'
  - Disk/IO
  - Allocation & Scheduling
  - Reliability Models/Components
- Si-Photonic network Models
- Mixed Multi-scale Simulation

- **What Can SST Do For YOU?**

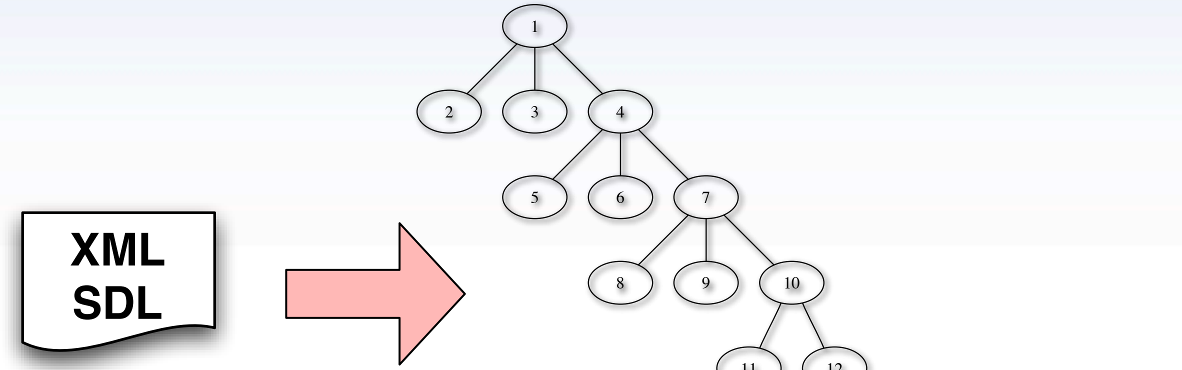
NNSA National Nuclear Security Administration

Sandia National Laboratories

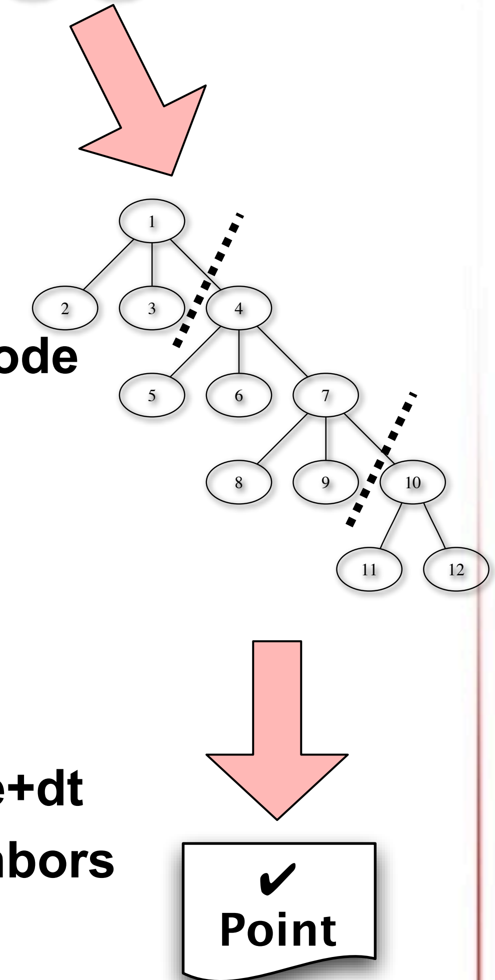
Bonus Slides

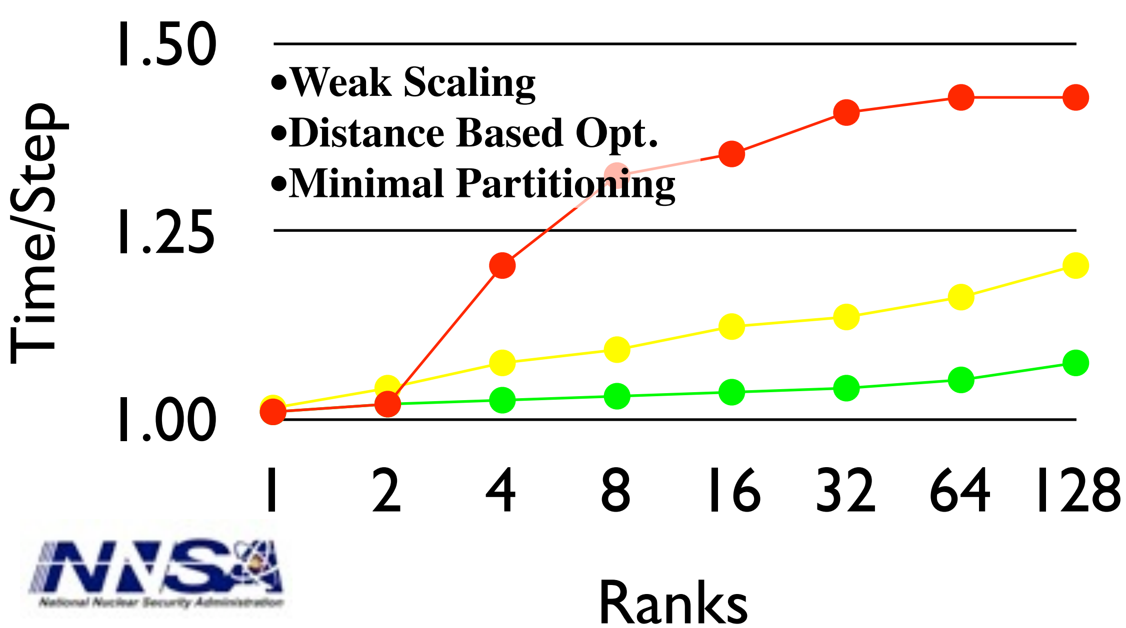
Parallel SST Core

- Core
  - Simulation Startup
  - Component Partitioning
  - Checkpointing
  - Event Passing
- Layered on MPI
  - Could use threads
  - Uses Boost & Zoltan
- Conservative Distance-Based DES algorithm
  - Appears scalable
  - No rollback



- Initial/Setup Mode:
  - 1. Load config file(s)
  - 2. Generate component graph
  - 3. Partition graph
  - 4. Instantiate components on each node
  - 5. Dump initial checkpoint
- Run Mode:
  - 1. Read checkpoint from disk
  - 2. Apply Edits
  - 3. Run Loop
    - a. advance components upto time+dt
    - b. exchange messages with neighbors
    - c. goto (3a)





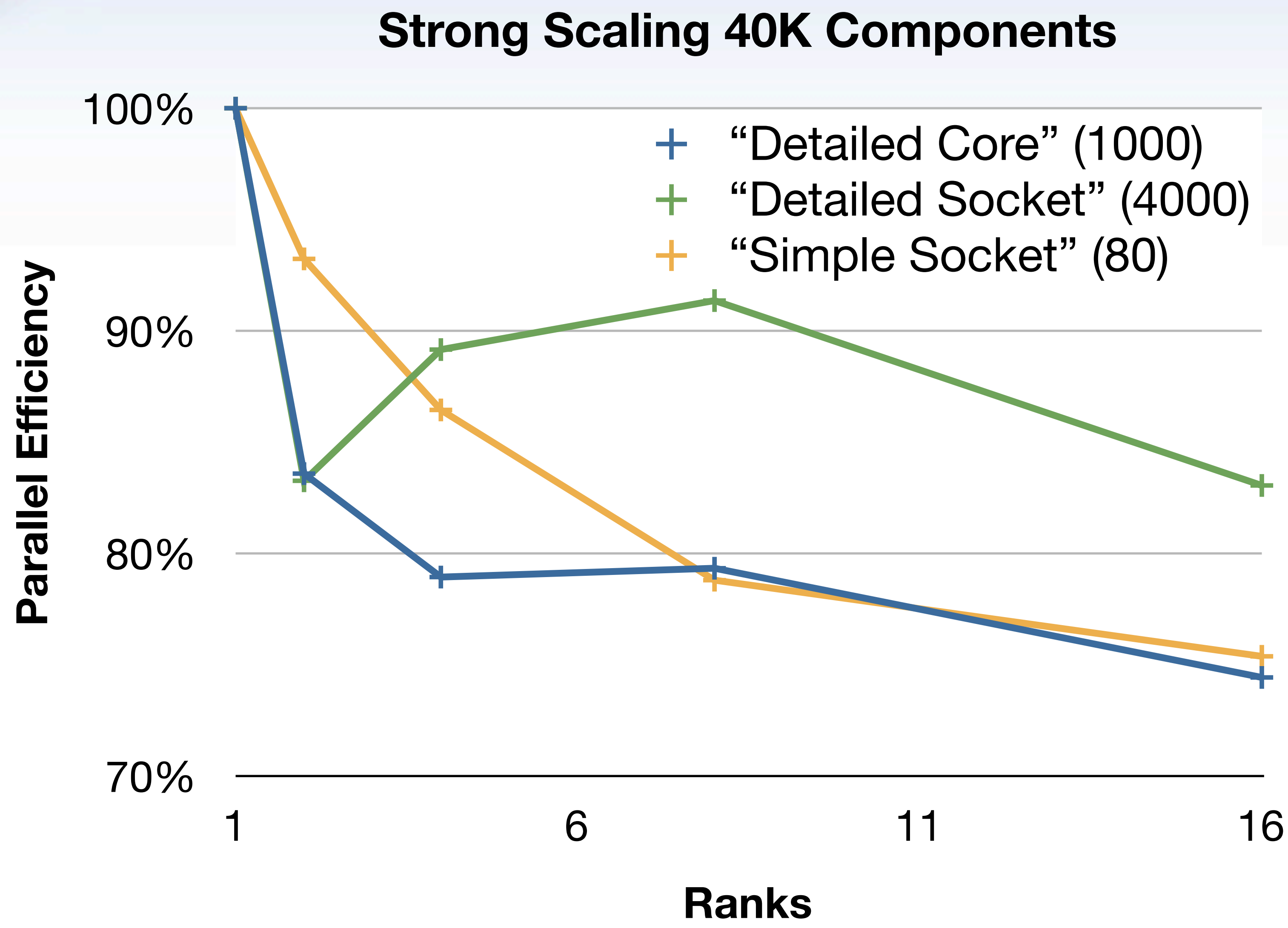
Design Space Exploration Results

- Latest memory technology not always best (DDR2 beats DDR3) due to latency, cost
- For these apps & inputs, fewer memory channels is better
- Better understanding of which configurations are best for a given application

Application	Chan.	Memory	Core	Cache	Energy	Performance	Cost
HPCCG	1	DDR2 25	Small	Small	250	510.7	176.86
HPCCG	1	DDR2 25	Small	Large	253	541.6	195.88
HPCCG	1	DDR3 25	Small	Large	263	566.9	220.20
HPCCG	1	DDR3 15	Small	Large	318	585.4	241.48
MD	1	DDR2 25	Large	Large	1504	105.9	206.14
MD	1	DDR2 25	Small	Small	1106	49.7	176.86
MD	1	DDR2 25	Small	Large	1119	50.7	195.88
MD	1	DDR2 25	Large	Small	1579	102.0	184.40
MD	2	DDR2 25	Large	Large	1480	105.4	213.55
MD	2	DDR2 25	Small	Small	1079	49.6	184.27
MD	2	DDR2 25	Small	Large	1093	50.6	203.29
gups	1	DDR2 25	Large	Small	1777	7.2	184.40
gups	1	DDR2 25	Small	Small	1183	6.9	176.86
gups	2	DDR2 25	Small	Small	1114	6.6	184.27
pagerank	1	DDR2 25	Large	Large	751	162.4	206.14
pagerank	1	DDR2 25	Small	Small	667	49.4	176.86
pagerank	1	DDR2 25	Small	Large	565	64.1	195.88
pagerank	1	DDR2 25	Large	Small	867	126.2	184.40
pagerank	2	DDR2 25	Large	Large	748	151.0	213.55

Pareto Optimal Designs

Early Scaling Results

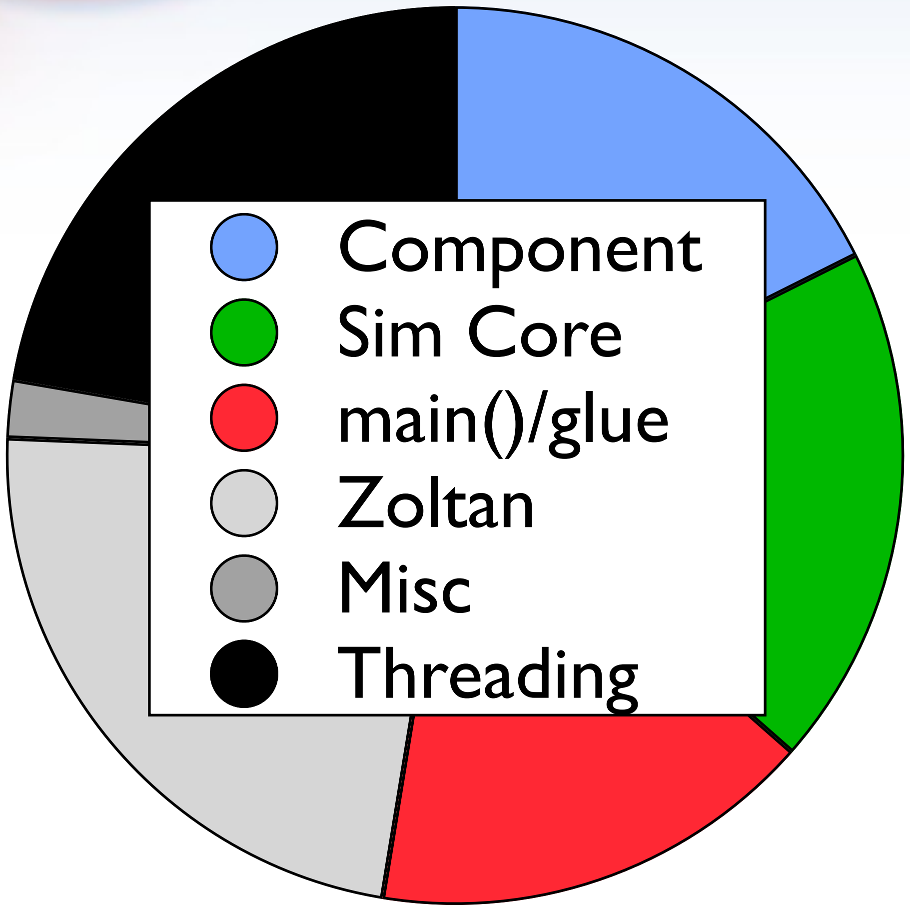


- Early results promising
- 75% parallel efficiency on 16 ranks with worst case layout
- Conservative algorithm appears effective
- Overhead for event serialization manageable (~10-15%)
- Still lots of optimization to be done

HPC Architectural Simulation Workshop

- Audience: Two communities (~50 participants)
  - –Simulation providers (those who write simulations/simulators)
  - –Simulation customers (those who use, or would like to use them)
- Findings
  - –HPC is faced by fundamentally new challenges (Hardware, Software, Scale, Power) and needs new simulation capabilities to confront them
  - –The simulation community has several examples of successful modular frameworks but needs mechanisms to share components
- Consumers
  - –55% currently use simulators "Very Much"
  - –91% would like to use simulators "Very Much"
- Producers
  - –78% believe their simulator would benefit from a common framework "Very Much"; 22% "somewhat"
  - –Only 12% believe "very much" that there would be major issues in integrating with a common framework

Threaded Strawman



- Initial/Setup Mode:
  - 1. Load config file(s)
  - 2. Generate component graph
  - 3. Partition graph
  - 4. Find minimal partition latency (dt)
  - 5. Distribute subgraphs to each node
  - 6. Instantiate components on each node
  - 7. "connect" components
  - 8. Dump initial checkpoint
- Run Mode:
  - 1. Read checkpoint from disk
  - 2. Apply Edits
  - 3. Run Loop
    - a. advance components upto time+dt
    - b. exchange messages with neighbors
    - c. occasionally, checkpoint
    - d. goto (3a)

- Added ~250 LOC
  - Original ~1000
- Each Component & its outgoing links assigned to a single thread
  - Minimal locking
- Three parallel operations
  - Pretic()s
  - HandleEvent()s
  - Queue sorting