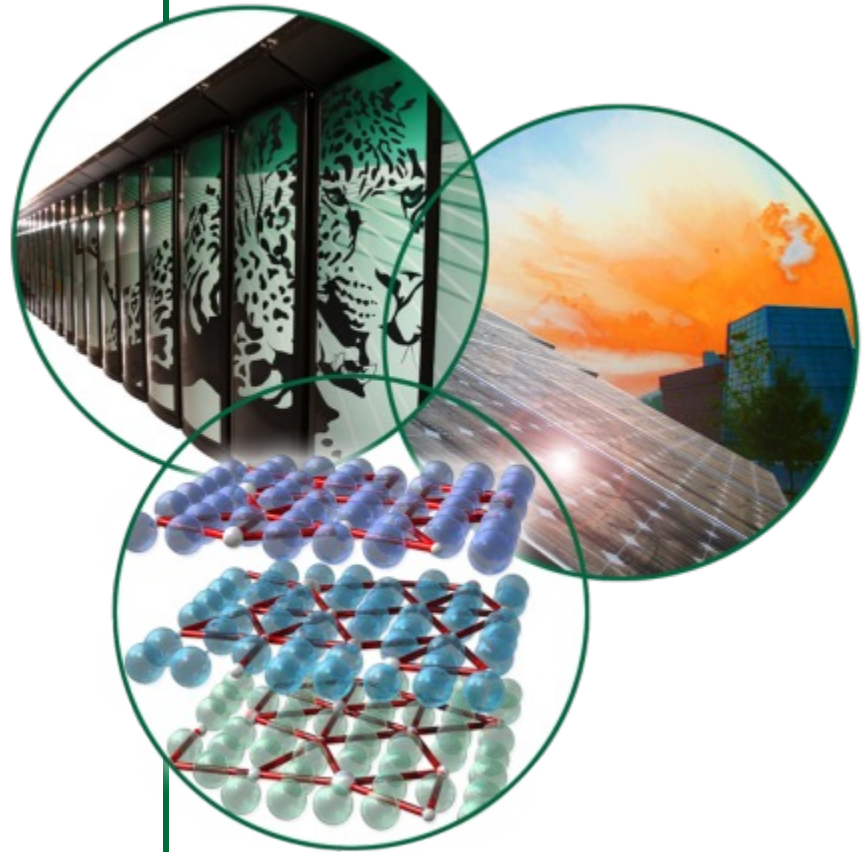


Resilience Challenges and Solutions for Extreme-Scale Supercomputing

Christian Engelmann

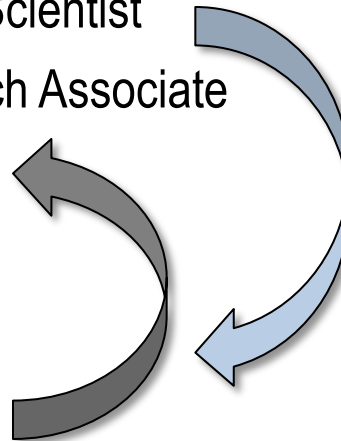
System Software Team Lead
Computer Science Research Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory

Invited Talk, TU Dresden, 9/3/2013



Speaker Introduction

- Research in HPC resilience, HPC performance tools, and HPC system software
 - 10 grants (\$23.5M), 70 papers, 8 MSc students, DOE Resilience Tech Council memb.
- Oak Ridge National Laboratory, USA
 - 2012-... : System Software Team Lead
 - 2009-... : R&D Staff Scientist
 - 2004-2009: Associate R&D Staff Scientist
 - 2001-2004: Post-masters Research Associate
 - 2000-2001: Software Developer
- University of Reading, UK
 - 2008: Ph.D. in Computer Science
 - 2001: M.Sc. in Computer Science
- University of Applied Sciences Berlin, Germany
 - 2001: Dipl.-Ing. (FH) in Computer Systems Engineering



Research Projects

- 2013-....: Hobbes – OS and Runtime Support for Application Composition
- 2013-....: MCREX – Monte Carlo Resilient Exascale Solvers
- 2012-....: Hardware/Software Resilience/Performance HPC Co-Design Tools
- 2011-12: Extreme-scale Algorithms and Software Institute
- 2009-11: Soft-Error Resilience for Future-Generation HPC
- 2008-11: RAS for Petascale High-End Computing and Beyond
- 2008-11: Scalable Algorithms for Petascale Multicore Systems
- 2006-09: Harness Workbench: Portability Across HPC Platforms
- 2006-08: Virtualized System Environments for Petascale HPC
- 2004-07: Modular Linux and Adaptive Runtime Support for HPC
- 2004-06: RAS for Terascale Computing
- 2002-04: Super-Scalable Algorithms for Cellular Architectures
- 2000-05: Harness – Heterogeneous Distributed Computing

Largest Multipurpose Science Laboratory within the U.S. Department of Energy

- Privately managed for US DOE
- \$1.65 billion budget
- 4400+ employees total
 - 1600 scientists and engineers
- 3,000 research guests annually
- 30,000 visitors each year
- Total land area 58mi² (150km²)

- Nation's largest energy laboratory
- Nation's largest science facility:
 - The \$1.4 billion Spallation Neutron Source
- Nation's largest concentration of open source materials research
- Nation's largest open scientific computing facility

ORNL East Campus: Site of World Leading Computing and Computational Sciences

Computational Sciences Building



Research Office Building

System Software Team

Engineering Technology Facility

Old Computational Sciences Building (until June 2003)

Joint Institute for Computational Sciences

Research Support Center (Cafeteria, Conference, Visitor)

National Center for Computational Sciences

40,000 ft² (3700 m²) computer center:

- 36-in (~1m) raised floor, 18 ft (5.5 m) deck-to-deck
- 25 MW of power with 6,600 tons of cooling capacity
- High-ceiling area for visualization lab: 35 MPixel PowerWall



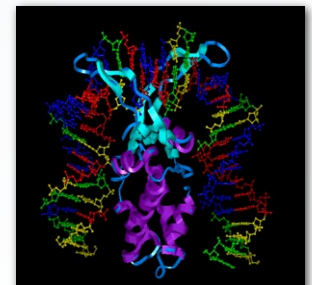
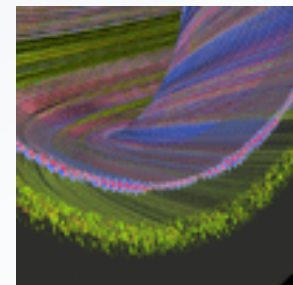
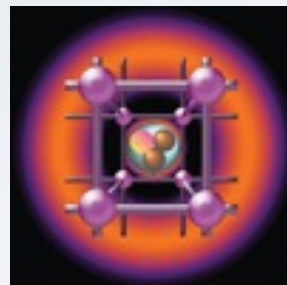
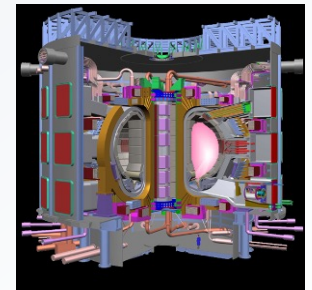
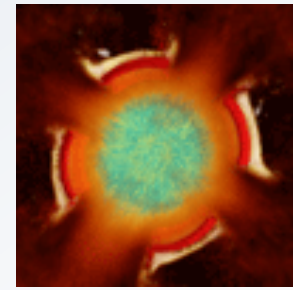
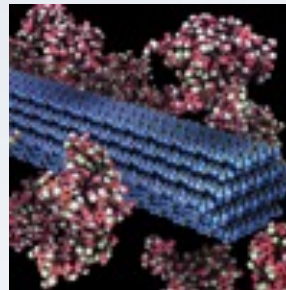
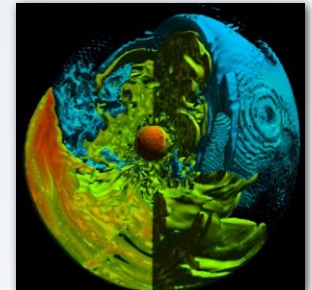
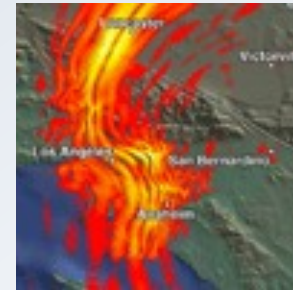
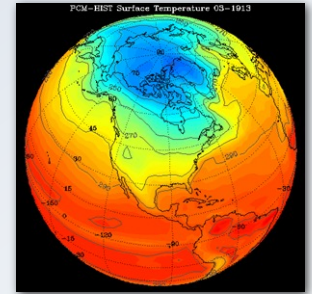
4 systems in the Top 500 List of Supercomputer Sites:

- Titan: 2. Cray XK7, 560,640 cores (AMD CPUs & Nvidia GPUs) @ 17,590 TFlops
- Kraken: 30. Cray XT5, 112,800 cores (AMD CPUs) @ 919 TFlops
- Gaea C2: 48. Cray XE6, 77,824 cores (AMD CPUs) @ 565 TFlops
- Gaea C1: 98. Cray XE6, 41,984 cores (AMD CPUs) @ 565 TFlops



At Forefront in Scientific Computing and Simulation

- Biological sciences
- Chemistry
- Computer science
- Earth science
- Engineering
- Materials science
- Physics



The Road to Exascale

- Current top systems are at ~1-15 PFlops:
 - #1: NUDT, NUDT, 3,120,000 cores, 33.9 PFlops, 62% Eff.
 - #2: ORNL, Cray XK7, 560,640 cores, 17.6 PFlops, 65% Eff.
 - #3: LLNL, IBM BG/Q, 1,572,864 cores, 16.3 PFlops, 81% Eff.
- Need 30-60 times performance increase in the next 9 years
- Major challenges:
 - *Power consumption*: Envelope of ~20MW (drives everything else)
 - *Programmability*: Accelerators and PIM-like architectures
 - *Performance*: Extreme-scale parallelism (up to 1B)
 - *Data movement*: Complex memory hierarchy, locality
 - *Data management*: Too much data to track and store
 - *Resilience*: Faults will occur continuously

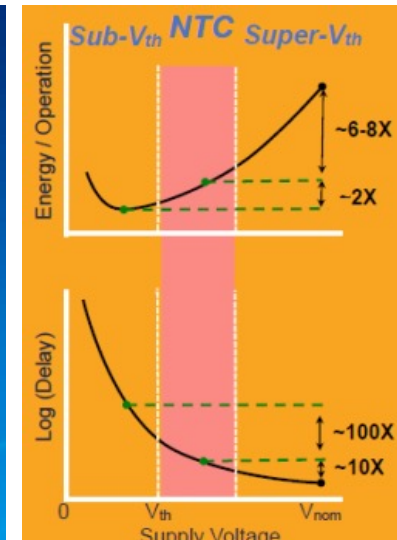
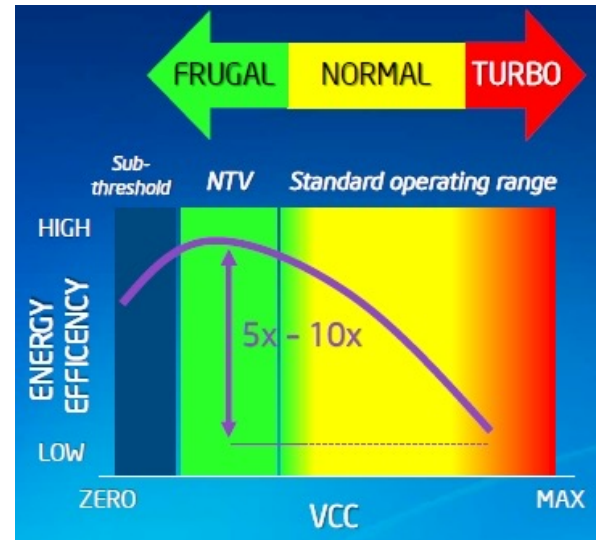
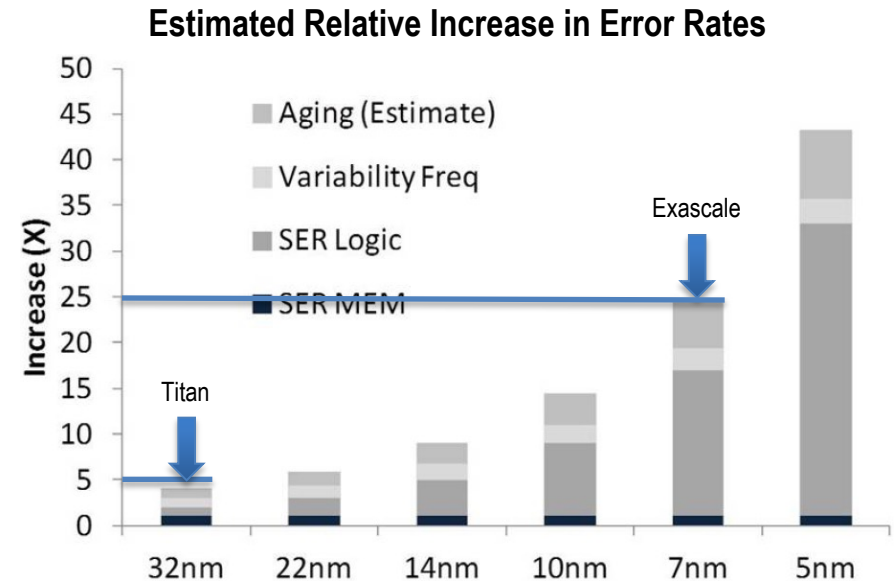
Discussed Exascale Road Map (2011)

Many design factors are driven by the power ceiling of 20MW

<i>Systems</i>	<i>2009</i>	<i>2012</i>	<i>2016</i>	<i>2020</i>
System peak	2 Peta	20 Peta	100-200 Peta	1 Exa
System memory	0.3 PB	1.6 PB	5 PB	10 PB
Node performance	125 GF	200GF	200-400 GF	1-10TF
Node memory BW	25 GB/s	40 GB/s	100 GB/s	200-400 GB/s
Node concurrency	12	32	O(100)	O(1000)
Interconnect BW	1.5 GB/s	22 GB/s	25 GB/s	50 GB/s
System size (nodes)	18,700	100,000	500,000	O(million)
Total concurrency	225,000	3,200,000	O(50,000,000)	O(billion)
Storage	15 PB	30 PB	150 PB	300 PB
IO	0.2 TB/s	2 TB/s	10 TB/s	20 TB/s
MTTI	1-4 days	5-19 hours	50-230 min	22-120 min
Power	6 MW	~10MW	~10 MW	~20 MW

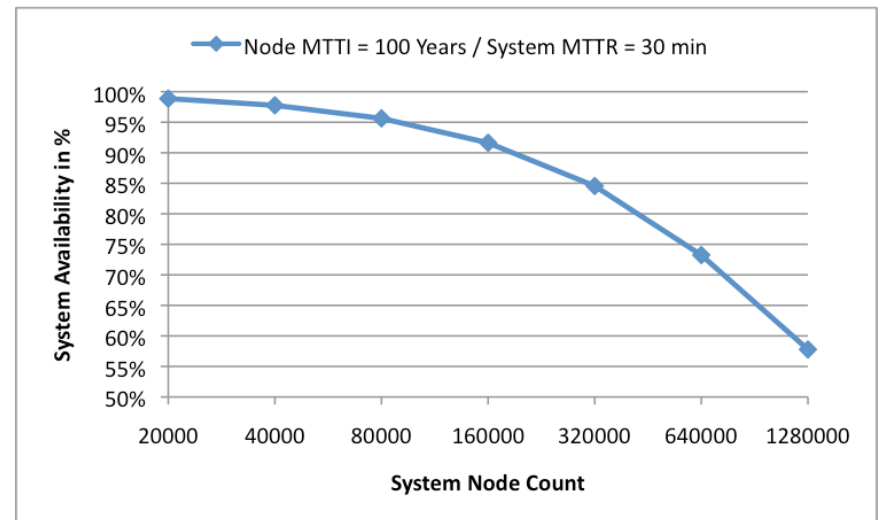
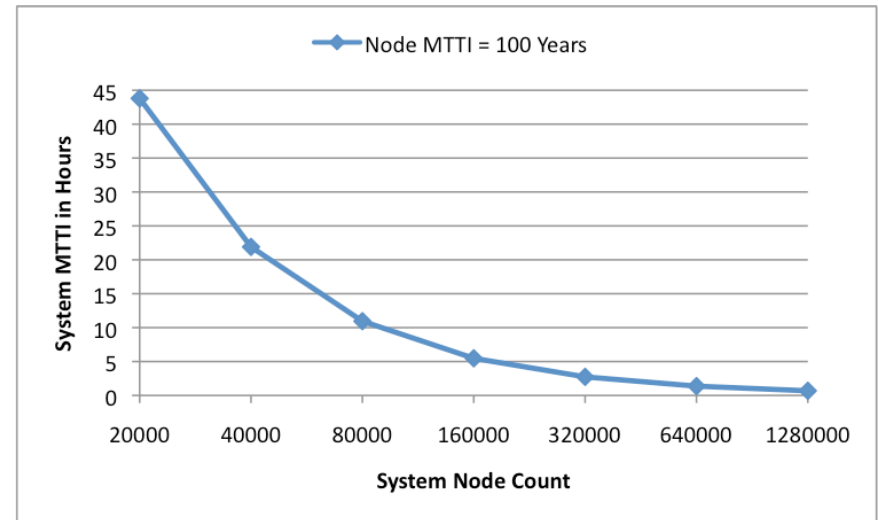
Problem #1: Less Reliable Components

- 7 nm process technology for exascale in 2020
 - Unknown aging effects
 - Increased variability
 - Increased soft error vulnerability
- Near-threshold voltage to achieve energy efficiency
 - Increased soft error vulnerability
 - Decreased noise immunity

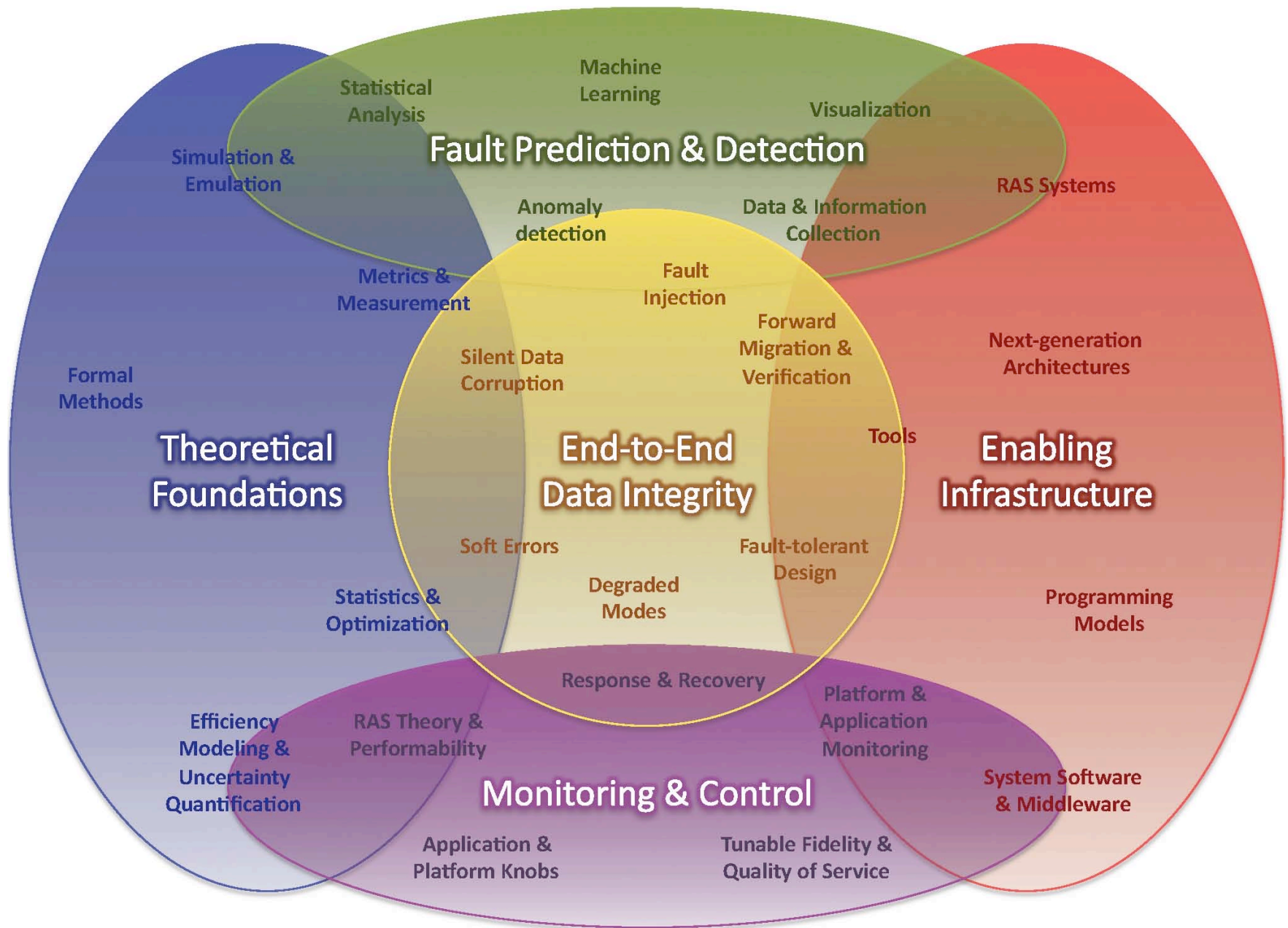


Problem #2: More Components

- System MTTI decreases as component count increases
- MTTI stayed roughly the same for the last 10 years
 - ~40x more cores/nodes
- Node reliability has been improved accordingly
 - Removed disks and fans
 - Improved system software
- No room to improve node reliability (see Problem #1)

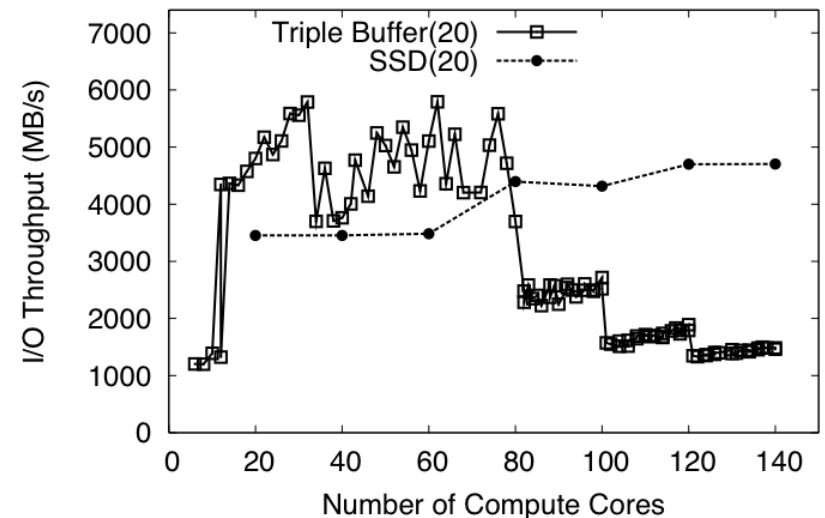
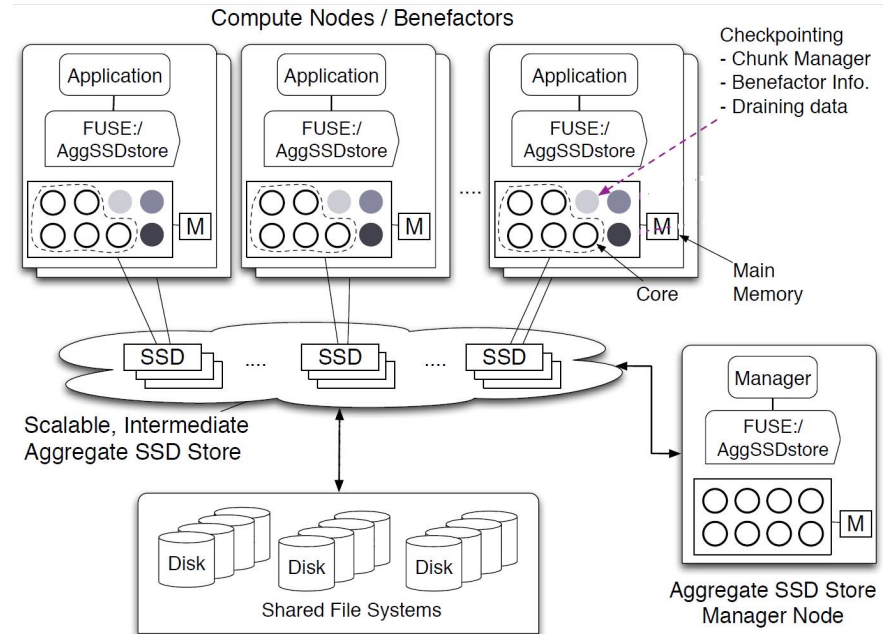


Key Areas of HPC Resilience Research



Advanced Checkpoint Storage Architecture

- Nodes offer available node-local SSD or memory space
- Manager process maintains metadata for each client
- FUSE client offers mount point w/o POSIX locking
- Data striped across nodes
- Tested on system with 160 cores using 140 as clients
- Solution helps maintaining I/O throughput



Full/Incremental Checkpointing with BLCR

- Hybrid checkpointing:
1 full and k incremental
(part of BLCR distribution)
- Tracks dirty memory pages
- Full: Saves all pages
- Incremental: Appends dirty pages to checkpoint file
- Recovery: Scans file in reverse sequence
- Optimal at 1 full / 9 incr.

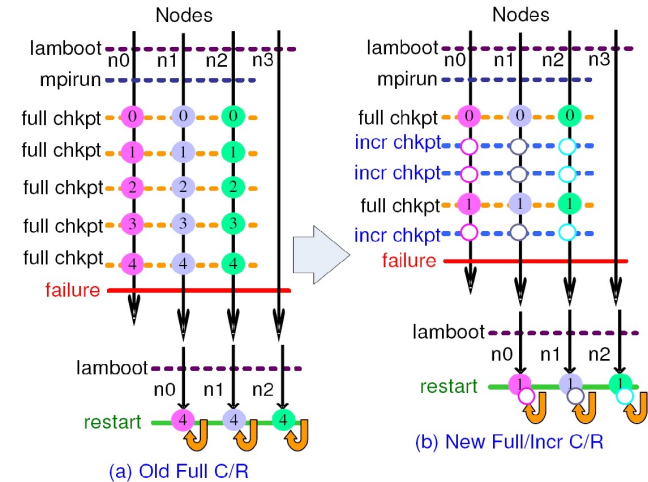
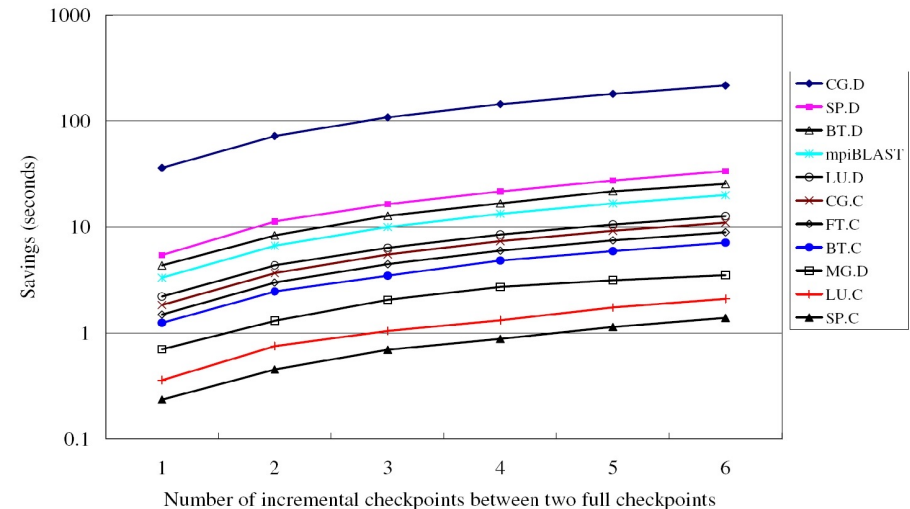


Fig. 1: Hybrid Full/Incremental C/R Mechanism vs. Full C/R



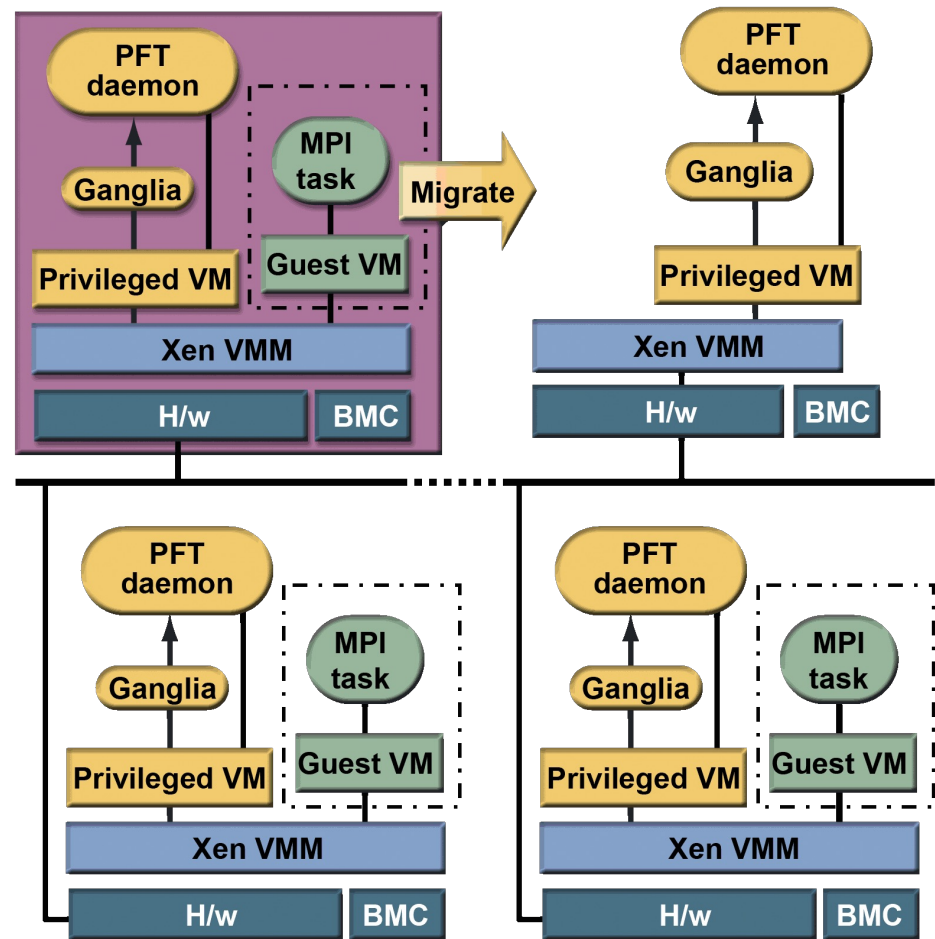
Proactive Fault Tolerance using Migration

- Relies on a feedback-loop control mechanism
 - Application health is constantly monitored and analyzed
 - Application is reallocated to avoid failures
 - Closed-loop control similar to dynamic load balancing
- Real-time control problem
 - Need to act in time to avoid imminent failures
- No 100% coverage
 - Not all failures can be anticipated



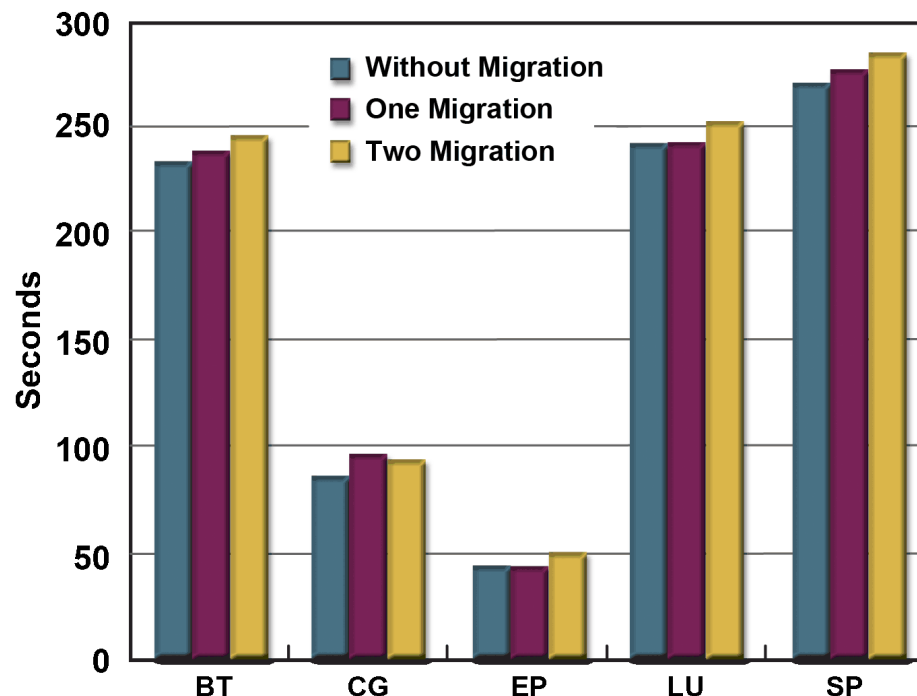
VM-level Migration with Xen (1/2)

- System setup
 - Xen VMM on entire system
 - Host OS for management
 - Guest OS for computation
 - Spares without Guest OS
 - Monitoring in Host OS
 - Decentralized scheduler/load balancer w/ Ganglia
- Deteriorating node health
 - Ganglia threshold trigger
 - Migrate guest OS to spare
 - Utilize Xen migration



VM-level Migration with Xen (2/2)

- Single migration overhead
 - Live : 0.5-5.0%
- Double migration overhead
 - Live : 2.0-8.0%
- Migration duration
 - Stop & copy : 13-14s
 - Live : 14-24s
- Application downtime
 - Stop & copy > Live
- Node eviction time
 - Stop & copy < Live



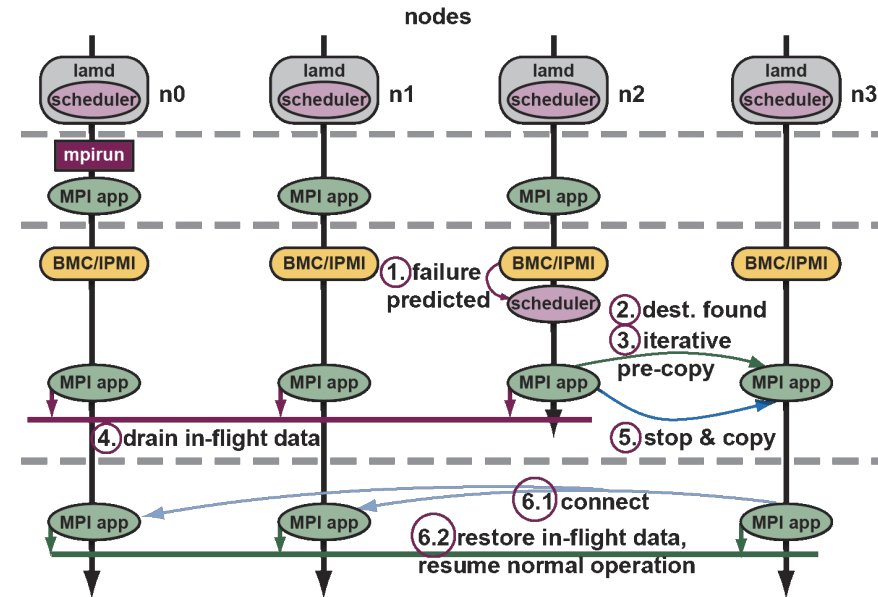
NPB runs on 16-node dual-core dual-processor Linux cluster at NCSU with AMD Opteron and Gigabit Ethernet



NC STATE UNIVERSITY

Process-Level Migration with BLCR (1/2)

- LAM/MPI with Berkeley Lab Checkpoint/Restart (BLCR)
- Per-node health monitoring
- New decentralized scheduler/ load balancer in LAM
- New process migration facility in BLCR (stop© and live)
- Deteriorating node health
 - Simple threshold trigger
 - Migrate process to spare



Process-Level Migration with BLCR (2/2)

Single migration overhead

- Stop & copy : 0.09-6.00%
- Live : 0.08-2.98%

Single migration duration

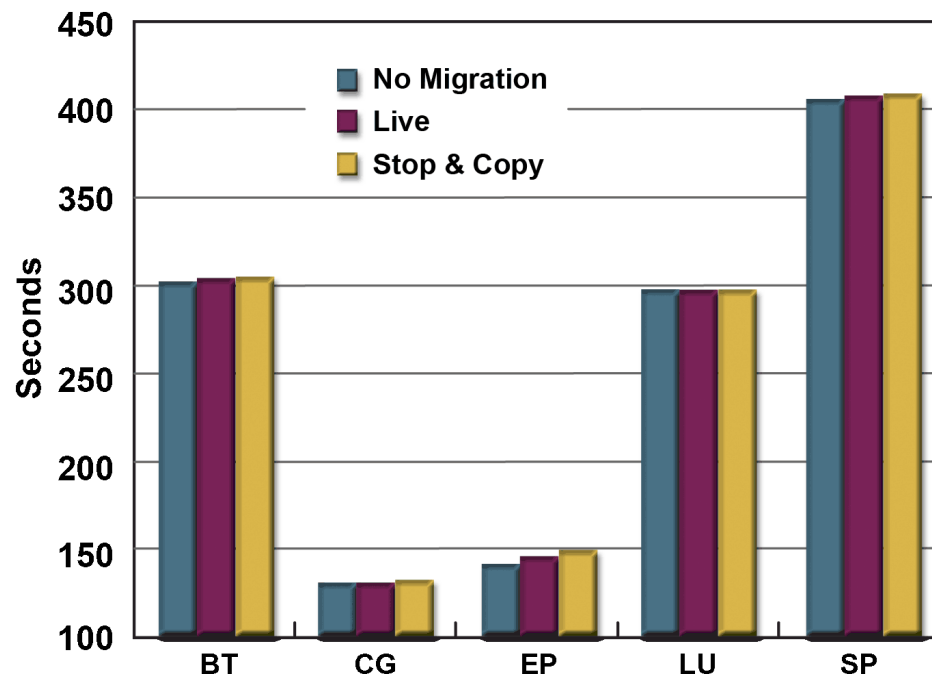
- Stop & copy : 1.0-1.9s
- Live : 2.6-6.5s

Application downtime

- Stop & copy > Live

Node eviction time

- Stop & copy < Live



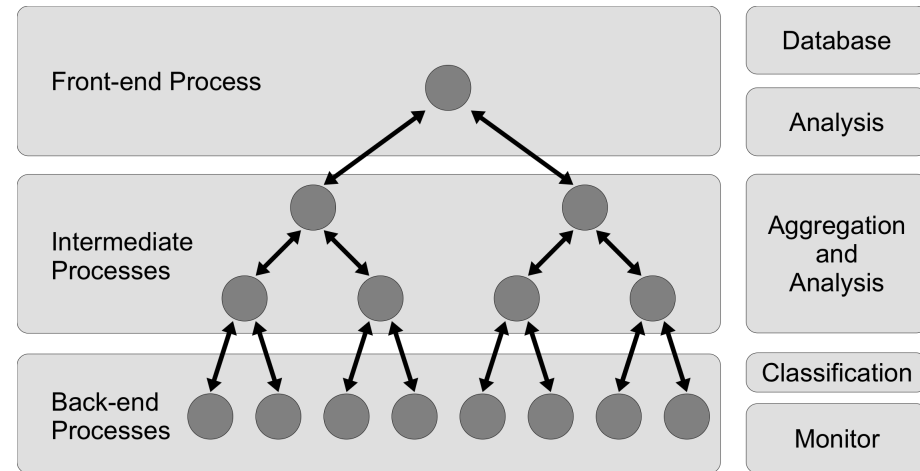
NPB runs on 16-node dual-core dual-processor Linux cluster at NCSU with AMD Opteron and Gigabit Ethernet



NC STATE UNIVERSITY

Scalable System Monitoring using MRNet

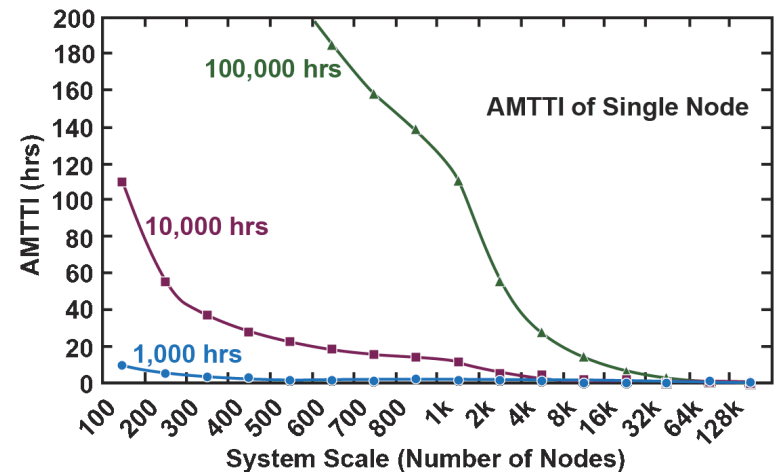
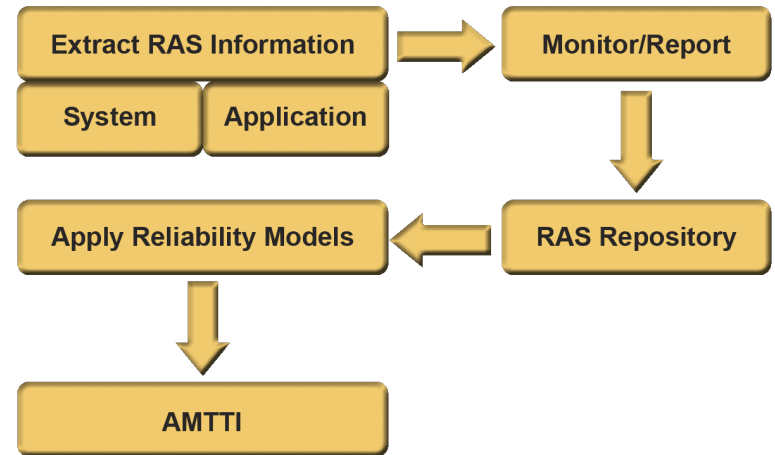
- Aggregation of metrics
- Tree-based overlay network
- Fan-in for metric data
- Fan-out for management
- Classification of data on back-end nodes
- In-flight processing on intermediate nodes
- Collection and storing on front-end node



- 8-year old 32-node cluster with collecting 18 metrics every 30s
- 1 MB of data in 4 hours ($\approx 56x$ less than Ganglia)

Modeling the Reliability of HPC Systems

- Application MTTI estimation
 - Monitoring & recording of application & system health
 - Reliability analysis on data
 - Adaptation of checkpoint interval to system health
- Finding failure patterns
 - Recording of interrupts
 - Reliability analysis on recent and historical data
- Systems: Blue Gene/L@LLNL and Jaguar@ORNL



Combining Checkpoint/Restart with Prediction-based Migration

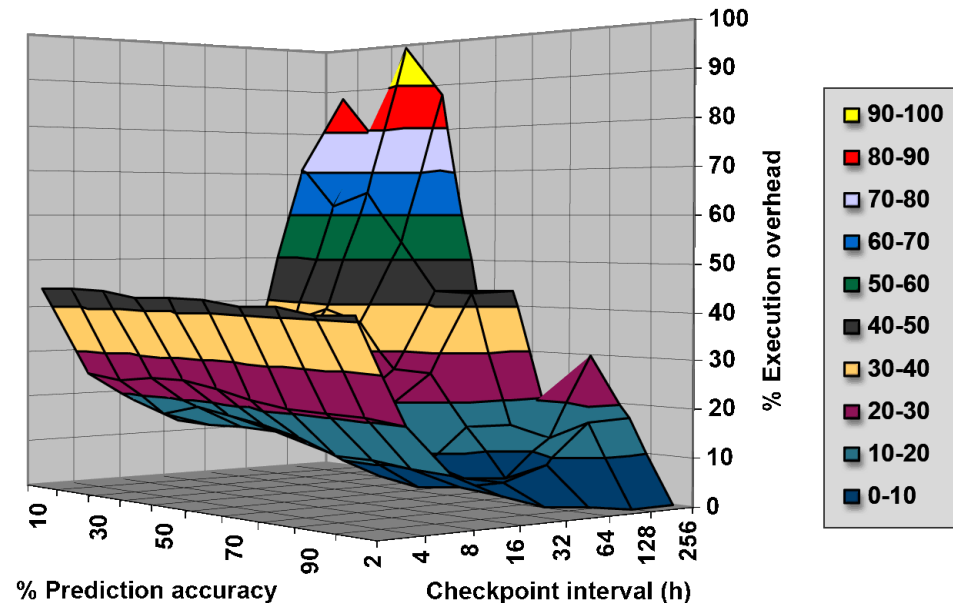
- Optimum for the given logs:
 - Prediction accuracy > 60%
 - Checkpoint interval 16-32h
- Results for higher accuracies and very low intervals are worse than only proactive or only reactive

Number of processes	125
Active/Spare nodes	125/12
Checkpoint overhead	50min
Migration overhead	1 min

*Simulation based on ASCI White logs
(nodes 1-125 and 500-512)*

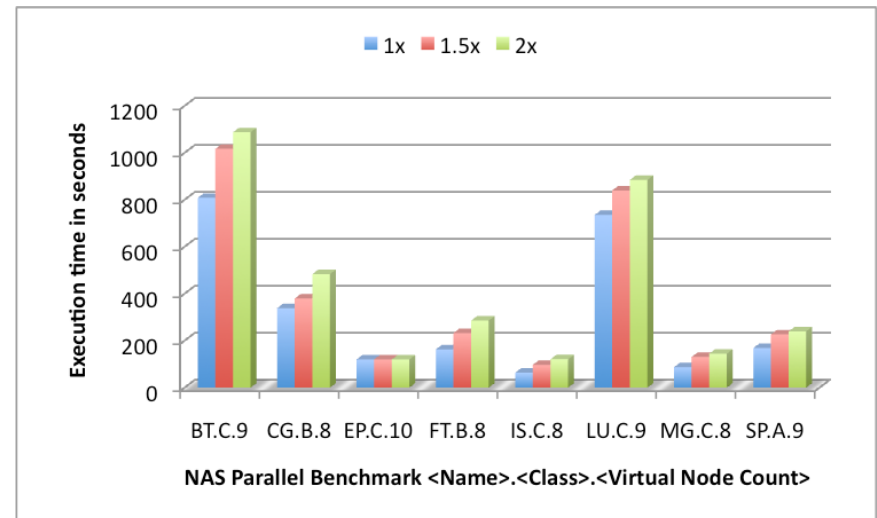
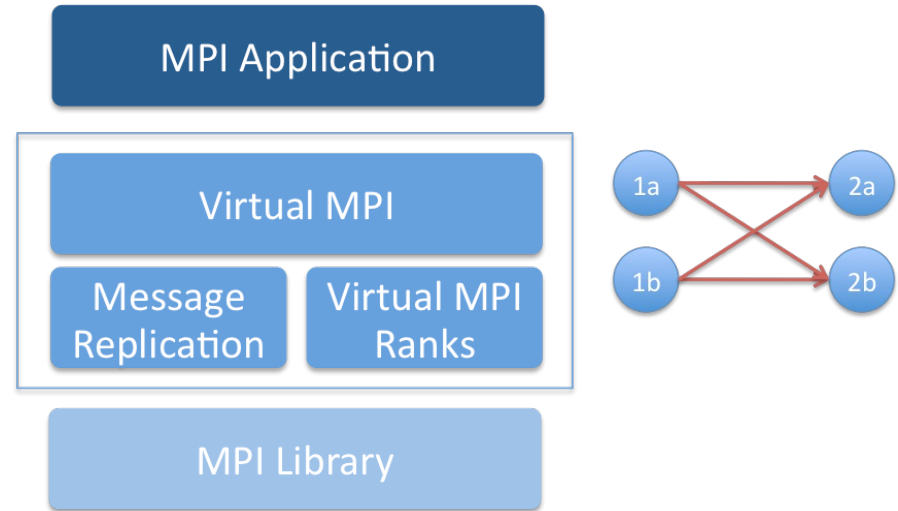
Christian Engelmann, Resilience Challenges and Solutions for Extreme-Scale Supercomputing, 9/3/2013.

Execution overhead for various checkpoint intervals and different prediction accuracy



Process-level Redundancy atop MPI for Soft Error Resilience

- Transparent redundant execution of MPI apps. (redMPI)
- Interposition library between MPI and the app.
- App. runs with $r * m$ ranks:
 - r ranks visible to the app.
 - m is the replication degree
- Fault model is fail-stop
- All messages are replicated
- File I/O is unified or replicated



Using Redundancy for Soft Error Injection

- Study propagation of silent data corruption at runtime
- Taint one replica and use the other one as live control
- Disable error correction by the redundant MPI (redMPI)
- Compare MPI messages and record mismatch
- Obtain some sense of silent data corruption vulnerability
 - Corruption patterns
 - Propagation speed

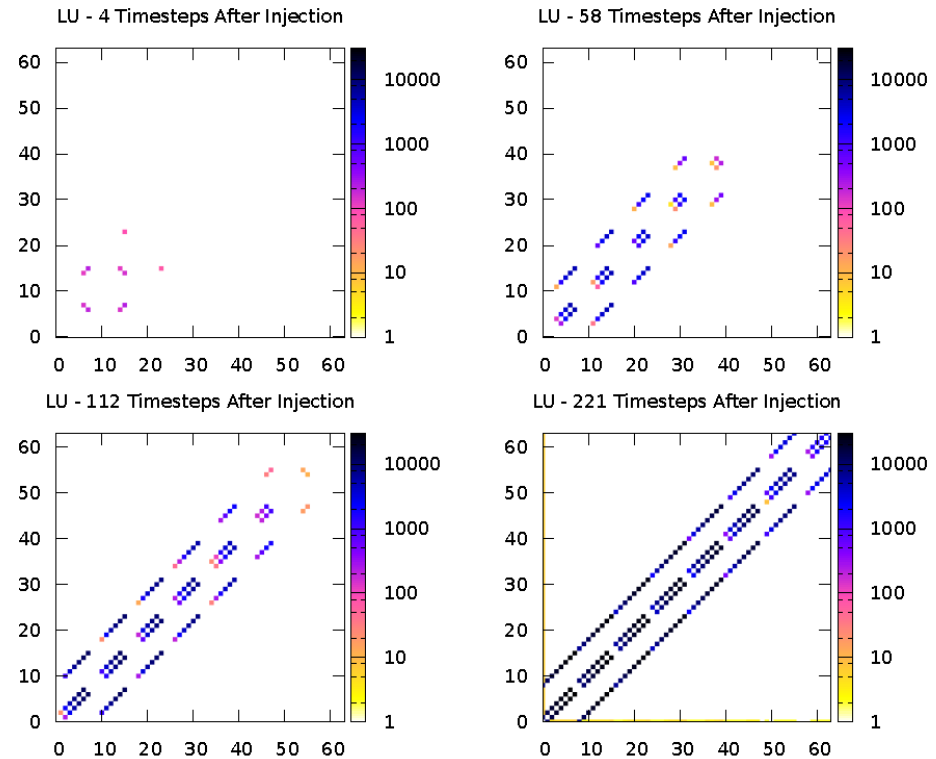


Fig. 3. NPB LU Corrupted Communication Patterns

Using Redundancy for Soft Error Injection

- Study propagation of silent data corruption at runtime
- Taint one replica and use the other one as live control
- Disable error correction by the redundant MPI (redMPI)
- Compare MPI messages and record mismatch
- Obtain some sense of silent data corruption vulnerability
 - Corruption patterns
 - Propagation speed

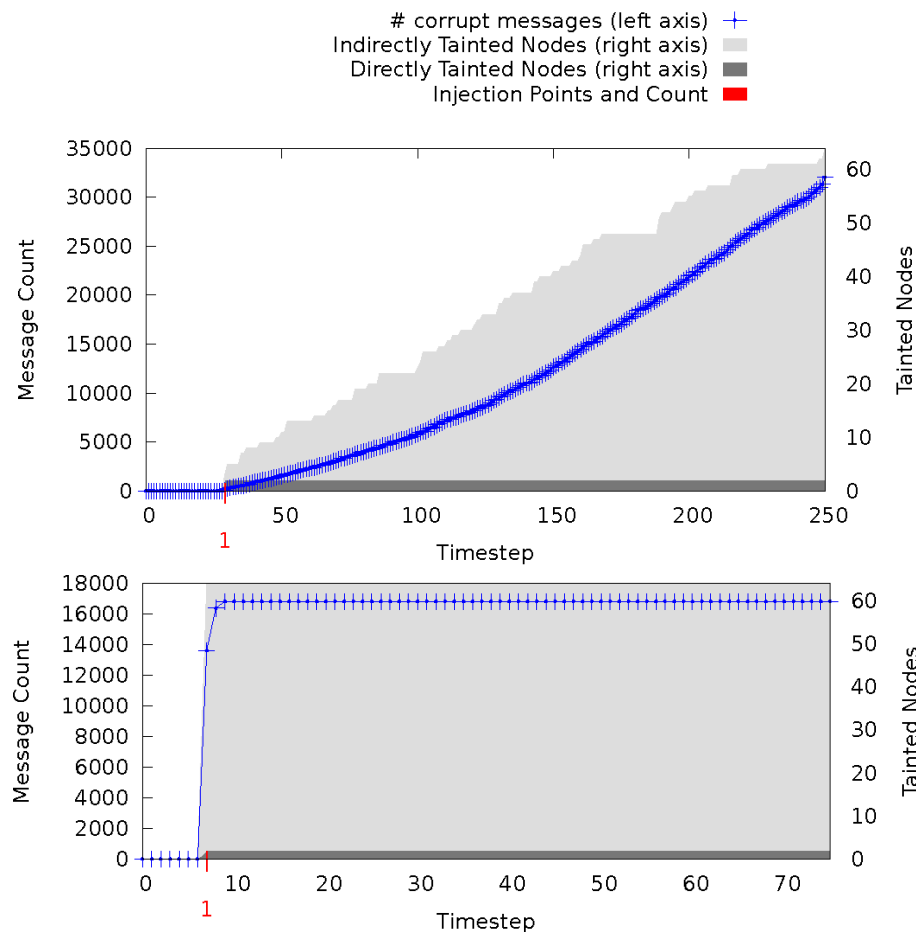


Fig. 8. NPB CG Overview of Corrupt Nodes and Messages

Radiation-Induced Soft Errors in CMOS Logic: A Case Study using Simulation

- Investigating timing errors in 180nm CMOS logic:
 - Reaching correct output state, but too late
- Methodology:
 - Design logic and identify vulnerabilities
 - Design logic layout and identify vulnerable areas
 - Simulation of logic with injected errors (SPICE)
- FIT rate for timing errors higher than for state errors

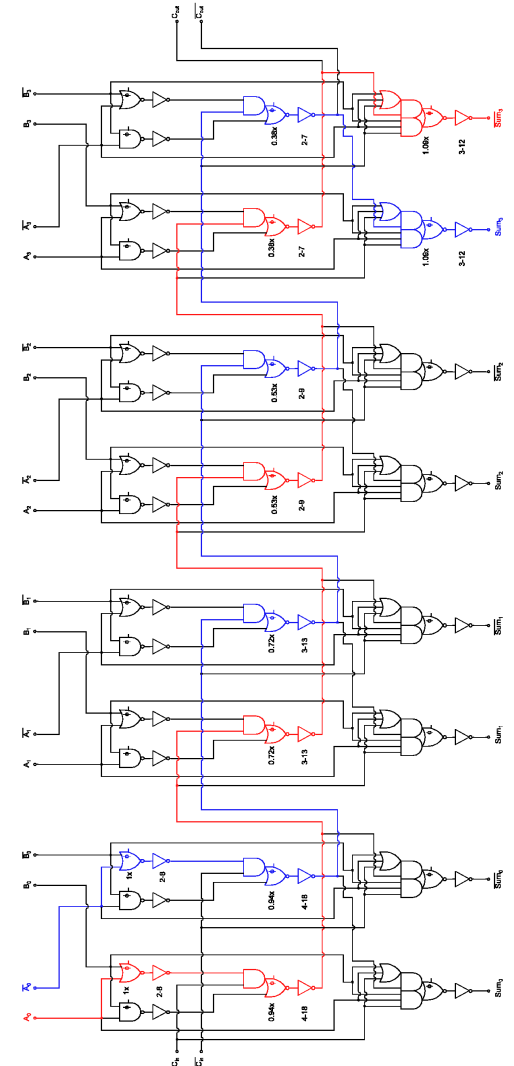


figure 6: Domino Logic Realization of a 4-bit Ripple Carry Adder

Radiation-Induced Soft Errors in CMOS Logic: A Case Study using Simulation

- Investigating timing errors in 180nm CMOS logic:
 - Reaching correct output state, but too late
- Methodology:
 - Design logic and identify vulnerabilities
 - Design logic layout and identify vulnerable areas
 - Simulation of logic with injected errors (SPICE)
- FIT rate for timing errors higher than for state errors

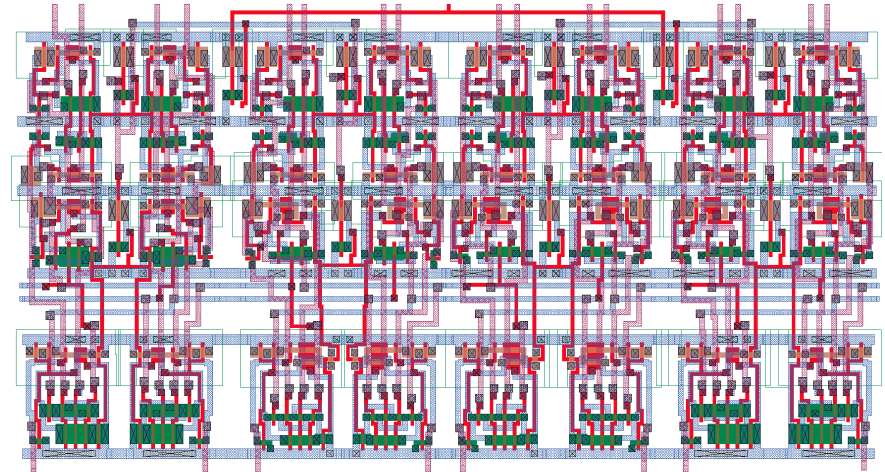


Figure 7: 4-bit Ripple Carry Adder Layout in the 0.18um TSMC process

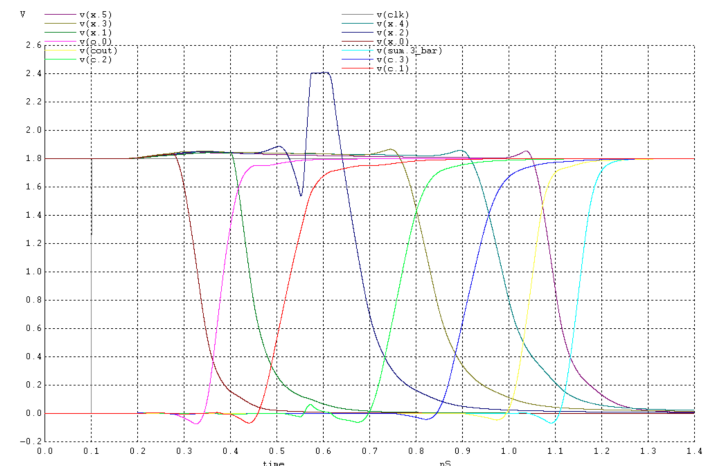


Figure 9: Voltages at the output nodes of gates on the sensitized path after an ion strike at the second a021 gate

Acknowledgements

- **Oak Ridge National Laboratory (ORNL):** S. L. Scott, C. Engelmann, G. Vallée, T. Naughton, C. Wang, A. Tikotekar, G. Ostrouchov, S. Vazhkudai, Y. Kim, G. Shipman, S. Boehm, I. Jones, and F. Lauer
- **Sandia National Laboratory:** K. Ferreira and Ron Brightwell
- **Louisiana Tech University:** C. Leangsuksun, N. Naksinehaboon, R. Nassar, and M. Paun
- **North Carolina State University:** F. Mueller, A. Nagarajan, J. Varma, X. Ma, F. Meng, and J. Elliott
- **Virginia Tech University:** M. Li, A. Butt
- **North Carolina A&T State:** A. Walker
- **Funding sources:** DoE OASCR, U.S. NSF, and ORNL/SNL LDRD



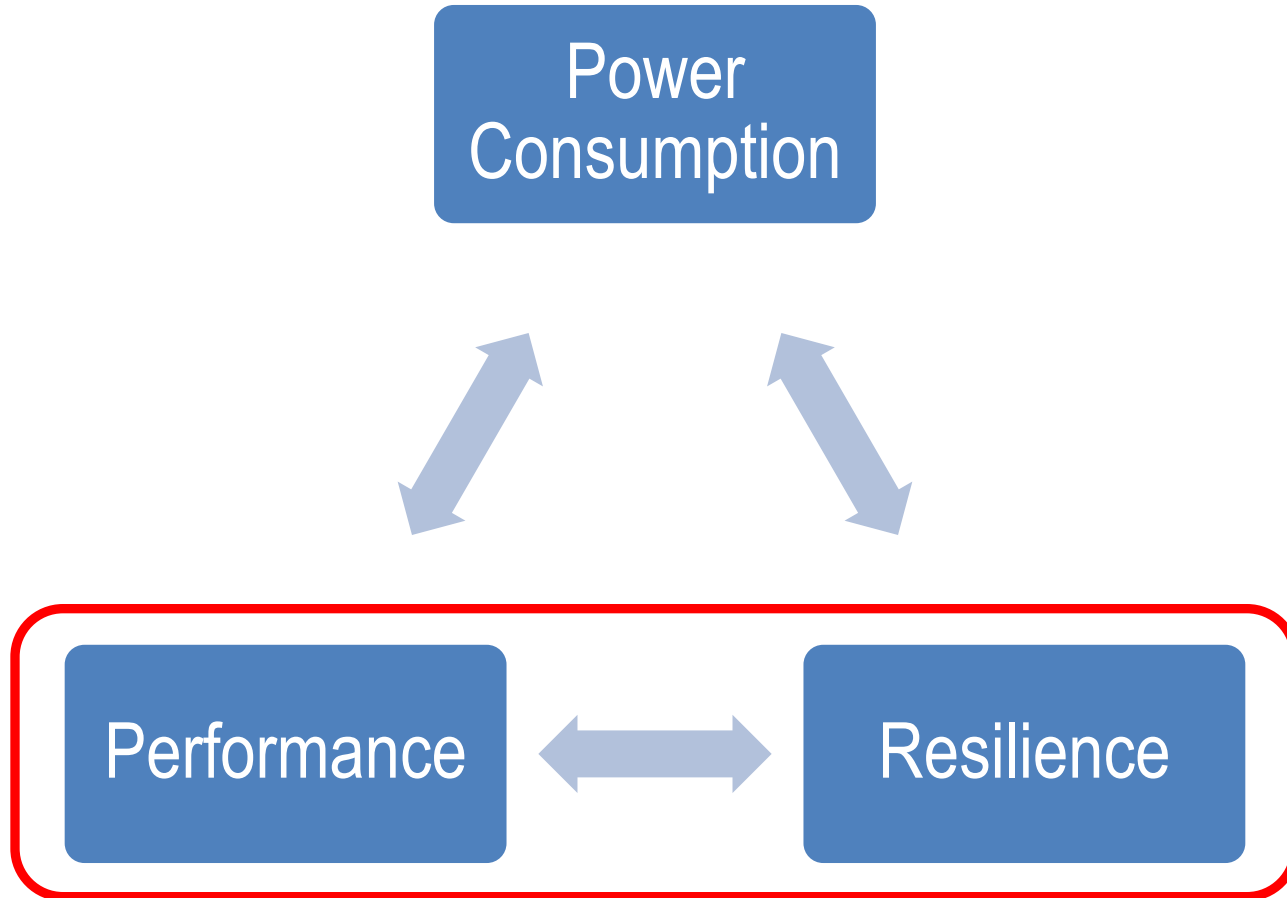
NC STATE UNIVERSITY



LOUISIANA TECH
UNIVERSITY®



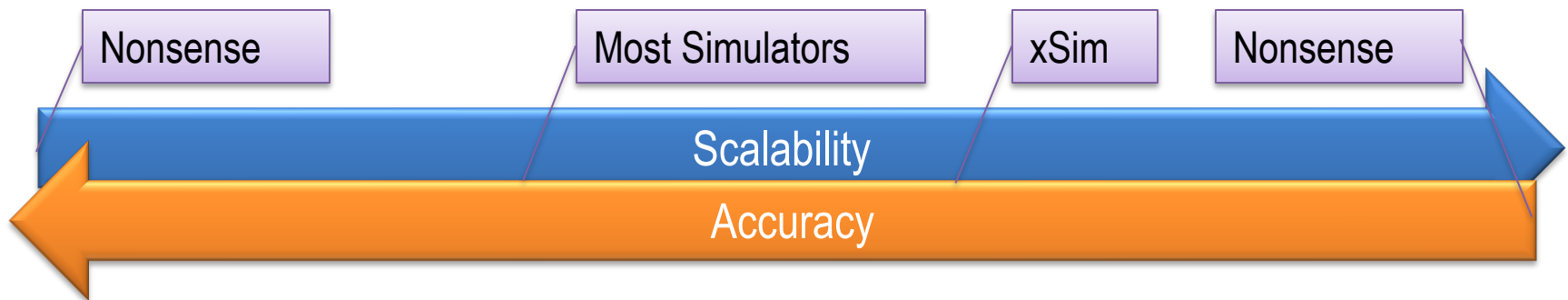
HPC System Hardware/Software Co-Design: Optimizing Trade-offs



Examples: ECC memory, checkpoint storage, data redundancy, computational redundancy, algorithmic resilience

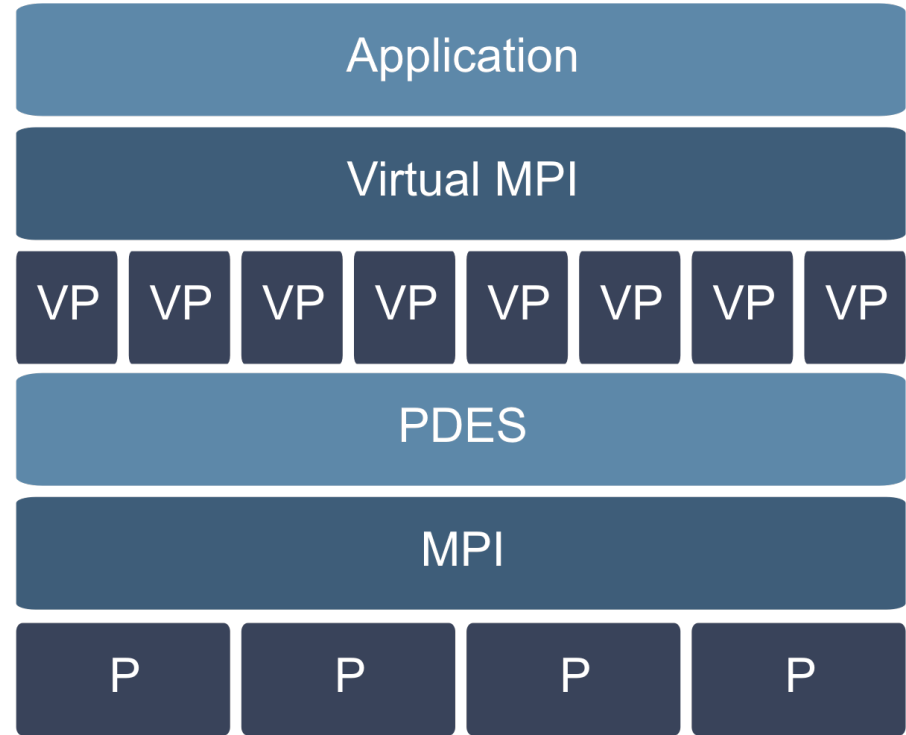
xSim: The Extreme-Scale Simulator

- Execution of real applications, algorithms or their models atop a simulated HPC environment for:
 - Performance evaluation, including identification of resource contention and underutilization issues
 - Investigation at extreme scale, beyond the capabilities of existing simulation efforts
- xSim: A highly scalable solution that trades off accuracy



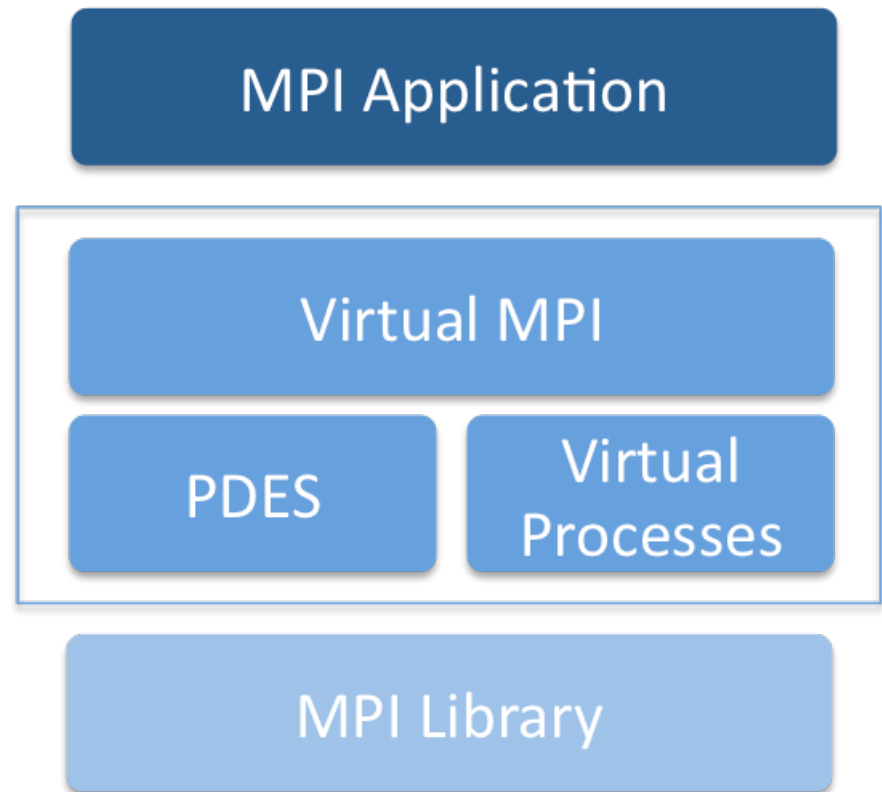
xSim: Technical Approach

- Combining highly oversubscribed execution, a virtual MPI, & a time-accurate PDES
- PDES uses the native MPI and simulates virtual procs.
- The virtual procs. expose a virtual MPI to applications
- Applications run within the context of virtual processors:
 - Global and local virtual time
 - Execution on native processor
 - Processor and network model



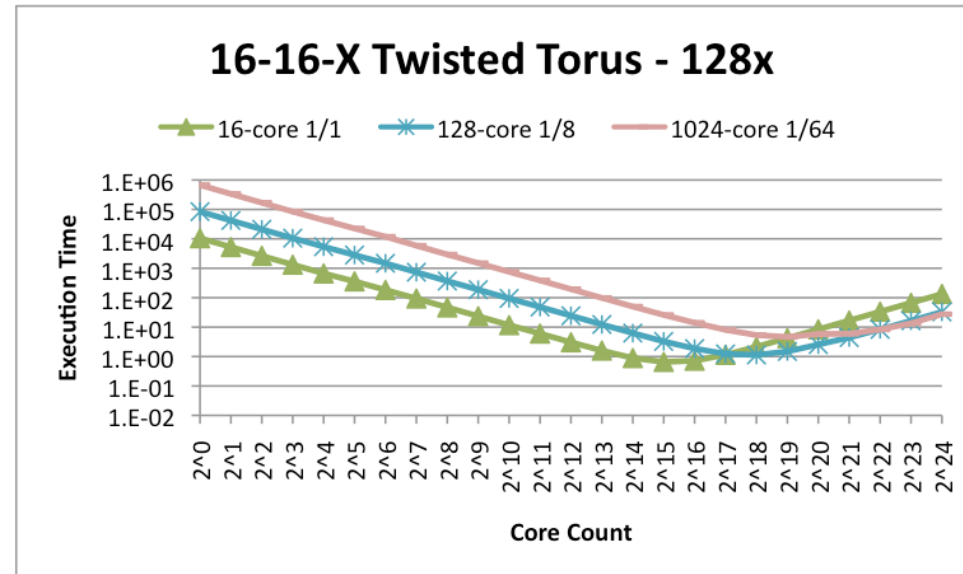
xSim: Design

- The simulator is a library
- Utilizes PMPI to intercept MPI calls and to hide the PDES
- Implemented in C with 2 threads per native process
- Support for C/Fortran MPI
- Easy to use:
 - Compile with xSim header
 - Link with the xSim library
 - Execute: `mpirun -np <np> <application> -xsim-np <vp>`



xSim: Design

- The simulator is a library
- Utilizes PMPI to intercept MPI calls and to hide the PDES
- Implemented in C with 2 threads per native process
- Support for C/Fortran MPI
- Easy to use:
 - Compile with xSim header
 - Link with the xSim library
 - Execute: `mpirun -np <np> <application> -xsim-np <vp>`



Investigating Performance Under Fault

- Parameterized MPI process failure injection/detection
- Parameterized MPI job failure injection/detection and restart
- First co-design toolkit supporting checkpoint/restart:

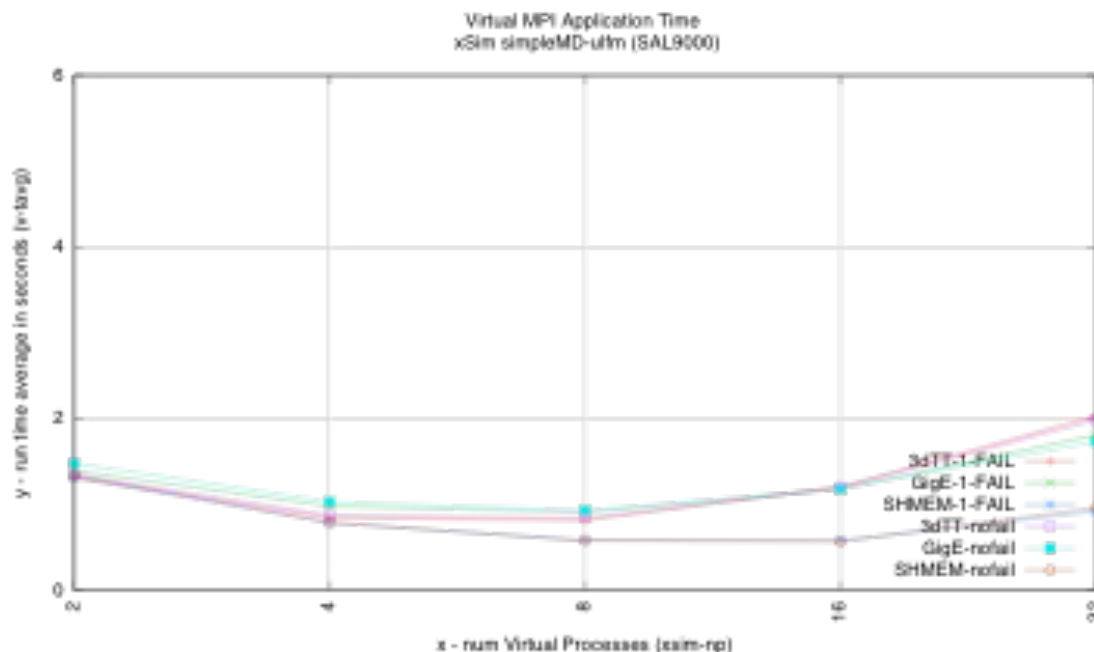
Table 1: Varying the checkpoint interval and system MTTF

$MTTF_S$	C	E_1	E_2	F	$MTTF_A$
—	1,000	5,248 s	—	0	—
6,000 s	500	5,258 s	7,957 s	1	3,978 s
6,000 s	250	6,377 s	7,074 s	1	3,537 s
6,000 s	125	6,601 s	6,750 s	1	3,375 s
3,000 s	500	5,258 s	10,584 s	2	3,528 s
3,000 s	250	6,377 s	8,618 s	2	2,872 s
3,000 s	125	6,601 s	7,948 s	2	2,649 s

*Accepted at the Workshop on Parallel Software Tools and Tool Infrastructures (PSTI)
at the International Conference on Parallel Processing (ICPP) 2013*

Investigating Performance Under Fault

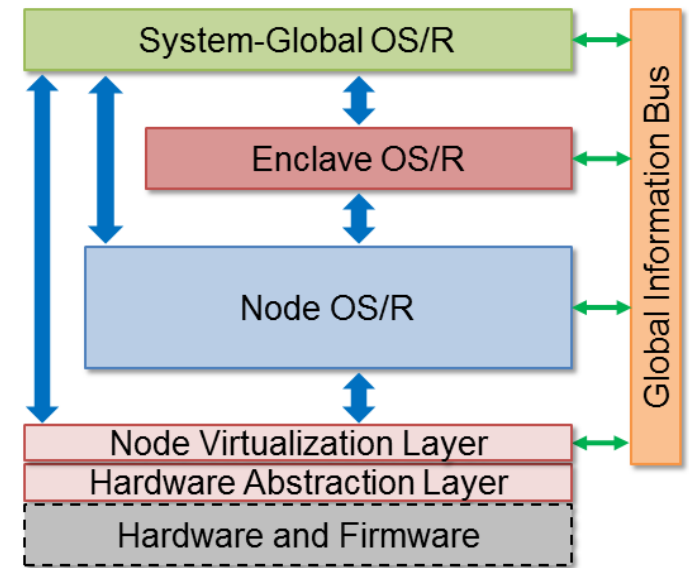
- User-Level Failure Mitigation (ULFM)
 - Enhancements to the MPI standard proposed by the MPI-FTWG
 - Permits application-level handling of MPI process faults
- Implemented ULFM support in xSim



Submitted to the Euromicro International Conference on Parallel, Distributed, and network-based Processing (PDP) 2014

Other Ongoing Work

- Hobbes – OS and Runtime Support for Application Composition
 - Operating system and runtime (OS/R) for extreme-scale HPC
 - Based on Kitten lightweight kernel and Palacios virtual machine monitor
 - Focusing on virtualization, networking, energy/power, software architecture, resilience, programming models, and tools
- MCREX – Monte Carlo Resilient Exascale Solvers
 - Resilient Monte Carlo solvers with natural fault tolerance to hard and soft failures for exascale HPC systems



Conclusion

- Resilience is a critical challenge for extreme-scale HPC
- Solutions have been and are being developed
- There are still many open questions
 - There is no HPC fault model yet
 - The performance/resilience/power trade-off is not understood
 - End-to-end management of resilience is needed
 - Bringing fault tolerance out of the closet

Questions

