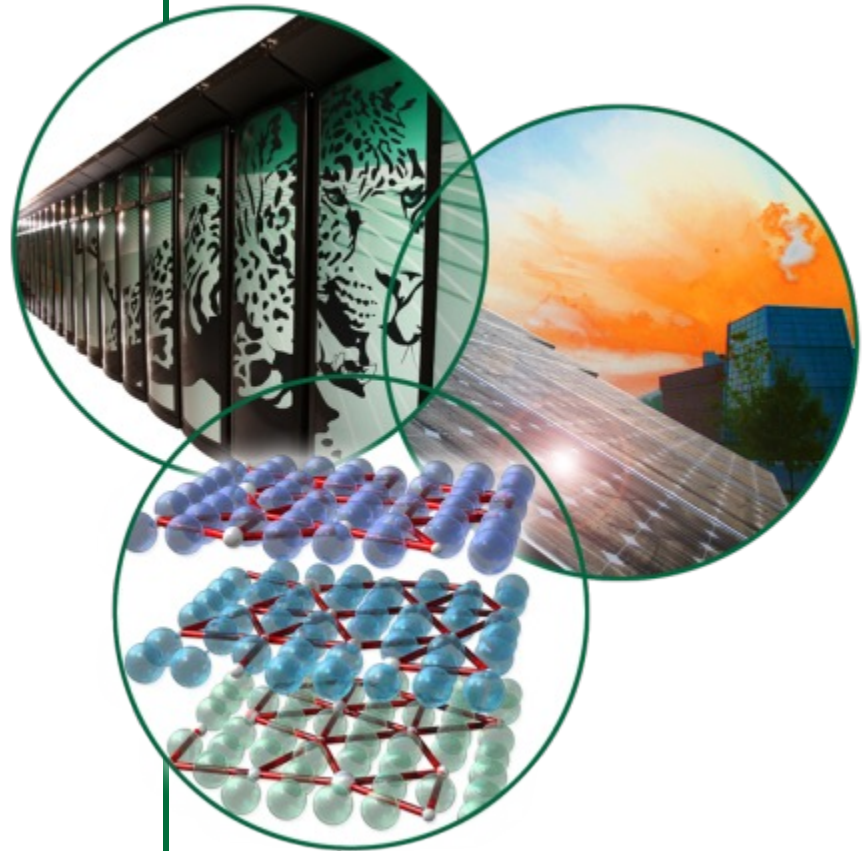


Resilience Challenges and Solutions for Extreme-Scale Supercomputing

Christian Engelmann

R&D Staff Scientist
Computer Science Research Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory



Speaker Introduction

- Research in HPC resilience, HPC performance tools, and HPC system software
 - 2015 DOE Early Career Award (5 years of funding for HPC resilience)
- Oak Ridge National Laboratory, USA
 - 2009-... : R&D Staff Scientist
 - 2012-2015 : System Software Team Lead
 - 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



15 Research Projects in 15 Years

- 2015-....: Resilience Design Patterns: A Structured Approach to Resilience at Extreme Scale
- 2015-....: Catalog: Characterizing Faults, Errors, and Failures in Extreme-Scale Systems
- 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
- \$1.4 billion budget
- 4400+ employees total
- 3,200 research guests annually
- 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

Engineering
Technology
Facility

Systems
Research Team

Old Computational
Sciences Building
(until June 2003)

Joint Institute for
Computational Sciences

Research Support Center
(Cafeteria, Conference, Visitor)

Oak Ridge Leadership Computing Facility

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 redundant cooling
- High-ceiling area for visualization lab
 - 30.5' x 8.5' display wall, 18 displays, 37.3 G pixels



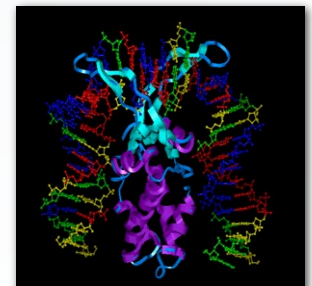
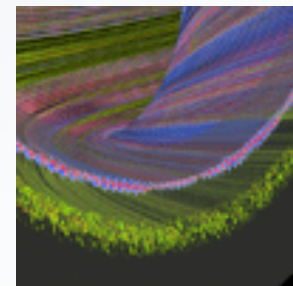
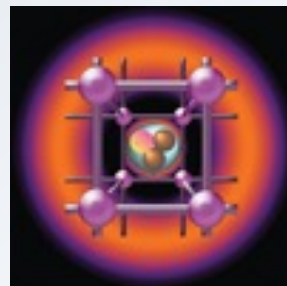
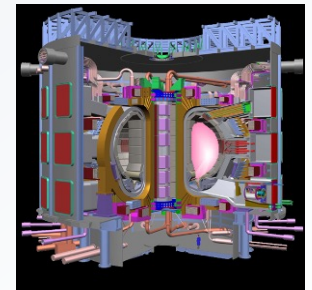
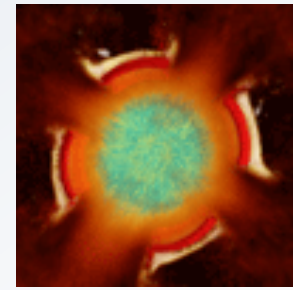
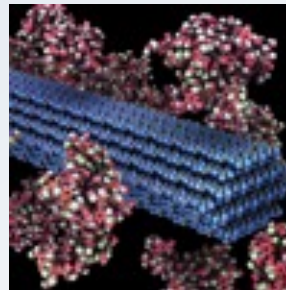
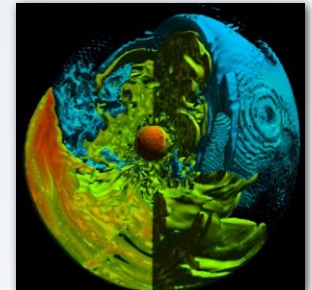
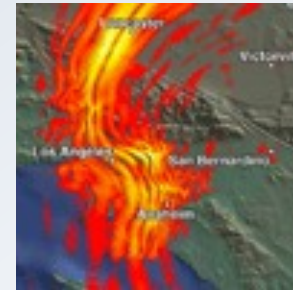
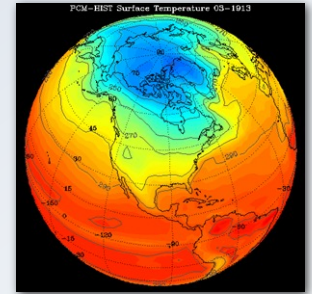
#2 system in the Top 500 List of Supercomputer Sites:

- Titan: Cray XK7, MPP with 18,688 16-core Processors and Nvidia GPUs @ 17,590 Tflops
 - 560,640 total cores, 8.2 MW power consumption



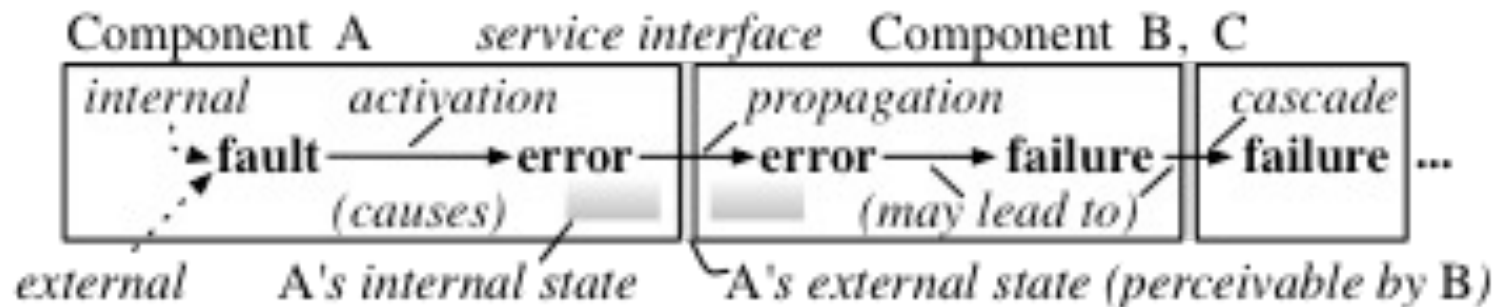
At Forefront in Scientific Computing and Simulation

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



What is resilience in HPC?

- ***Parallel application correctness and execution efficiency, in spite of frequent faults, errors, and failures.***
- Faults happen everywhere in the system, e.g.:
 - Radiation-induced bit flips in memory, hardware wearing out and bugs
- Whether a fault results in an error and an eventual failure depends on employed resilience mechanisms, e.g.:
 - Fault masking, memory error correction and failure recovery with checkpoint/restart



Why do we need resilience in HPC?

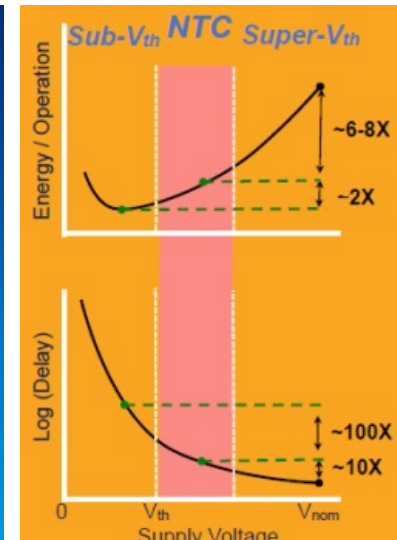
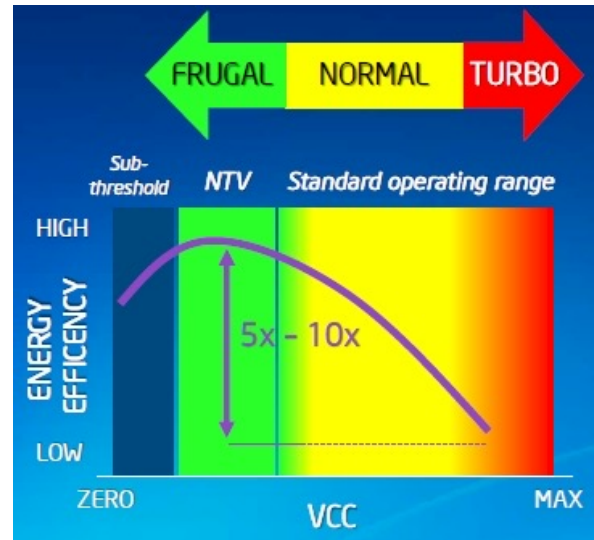
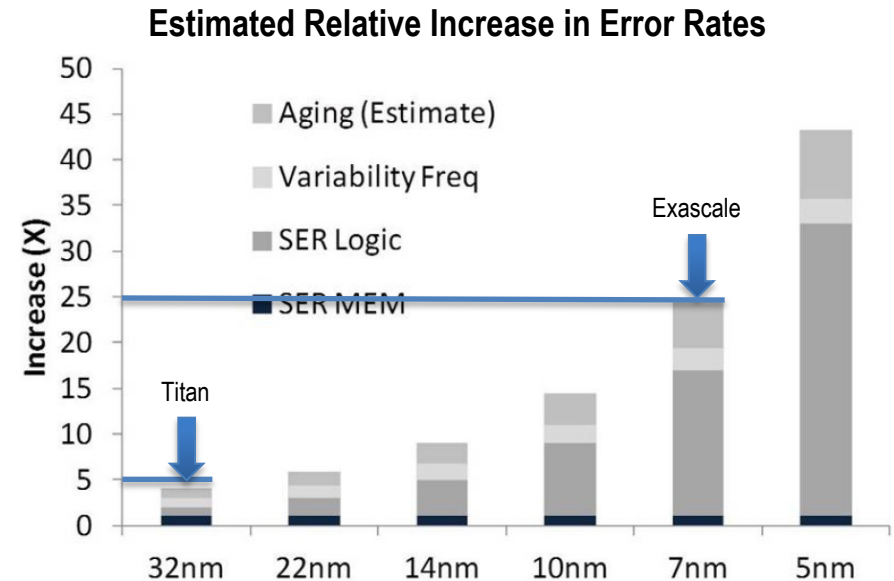
- Correctness
 - Computing the wrong result has consequences for policy decisions, e.g.:
 - Climate and weather simulation codes impact national/local decisions
 - Erroneous computation does not advance science, e.g.:
 - Materials science applications investigate unknown structures/materials
 - Some applications require bit-wise reproducibility for regulatory compliance, e.g.:
 - Nuclear reactor simulation codes
- Execution efficiency
 - Computational resources are expensive and should not be wasted
- Current state of practice
 - Error detection and correction in hardware
 - Failure detection in software
 - Checkpoint/restart in software

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

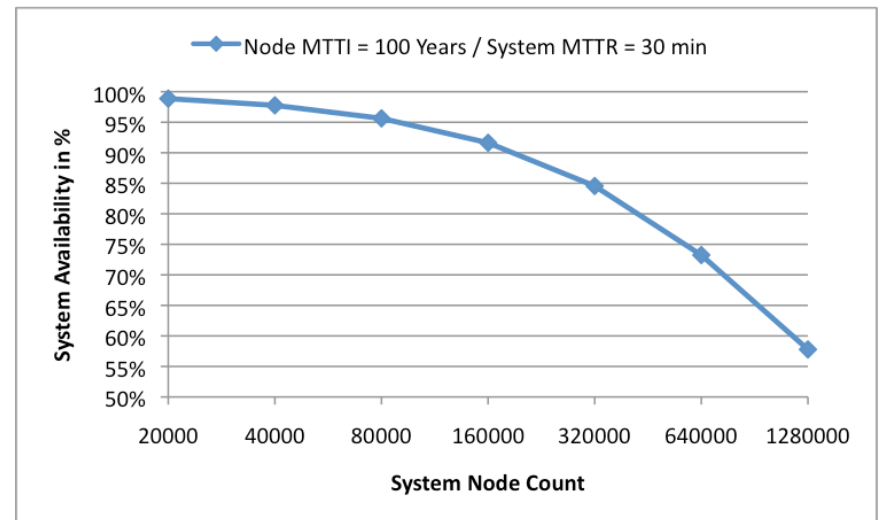
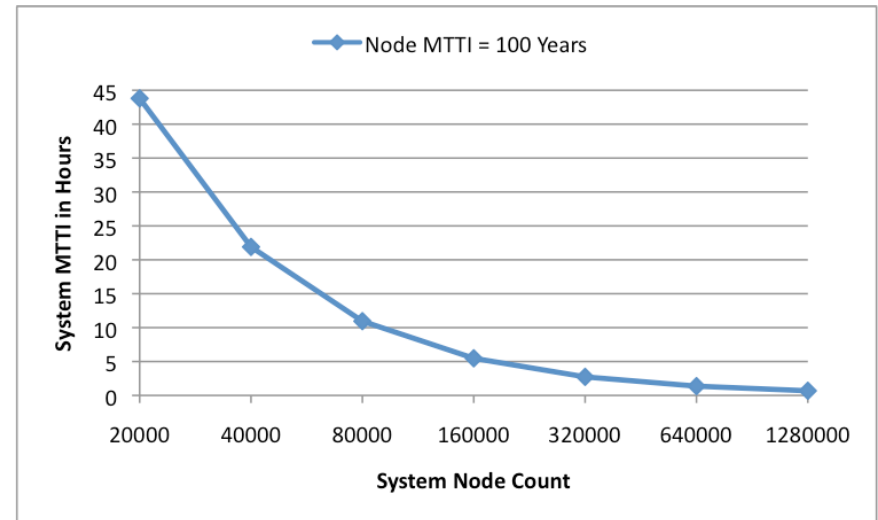
Problem #1: Less Reliable Components

- 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

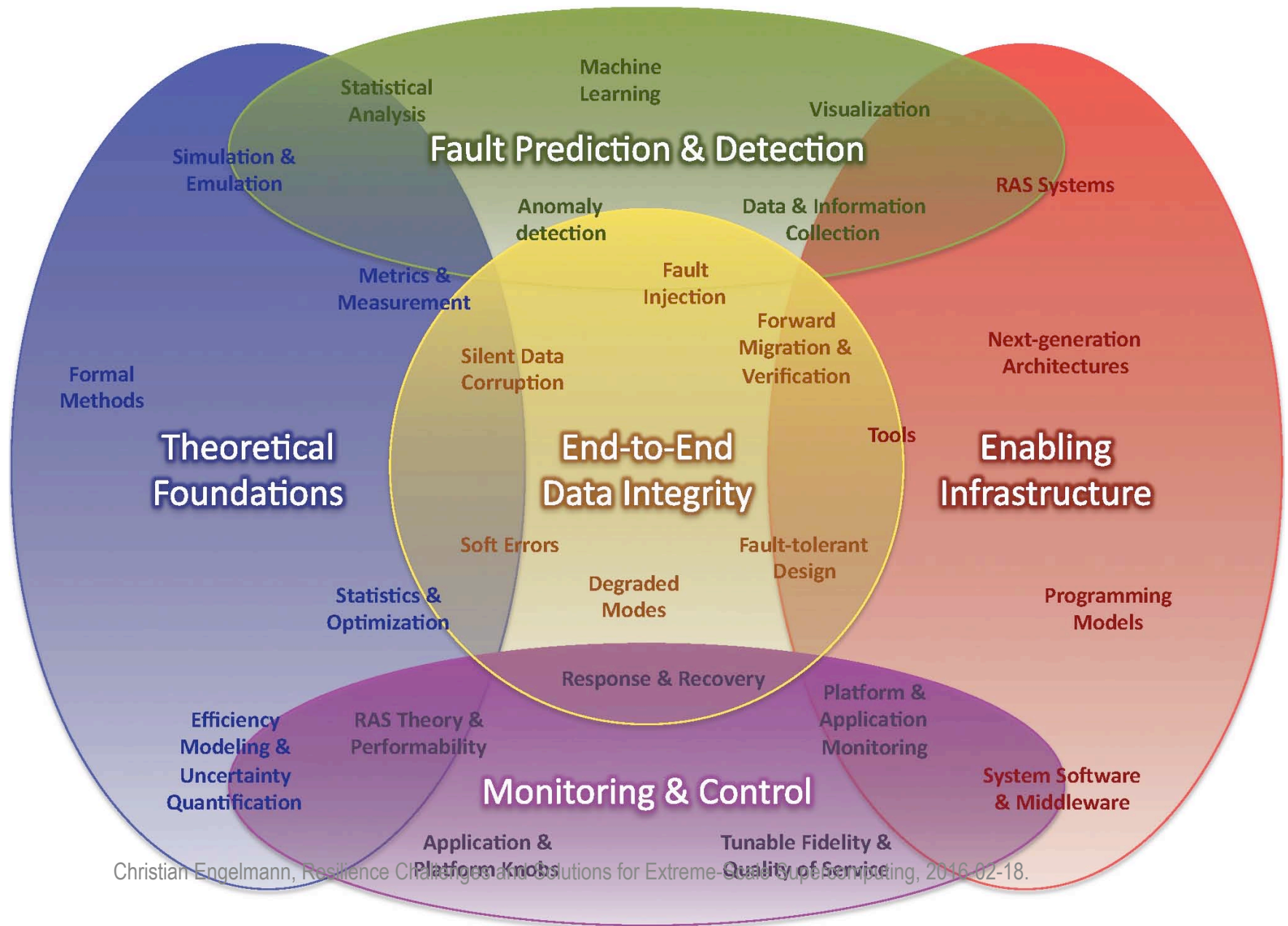
- System MTTI decreases as component count increases
- MTTI stayed roughly the same for the last 10 years
 - ~40x more cores/nodes
- Node reliability has been improved accordingly
 - Removed disks and fans
 - Improved system software
- No room to improve node reliability (see Problem #1)



Why is Resilience a Challenge?

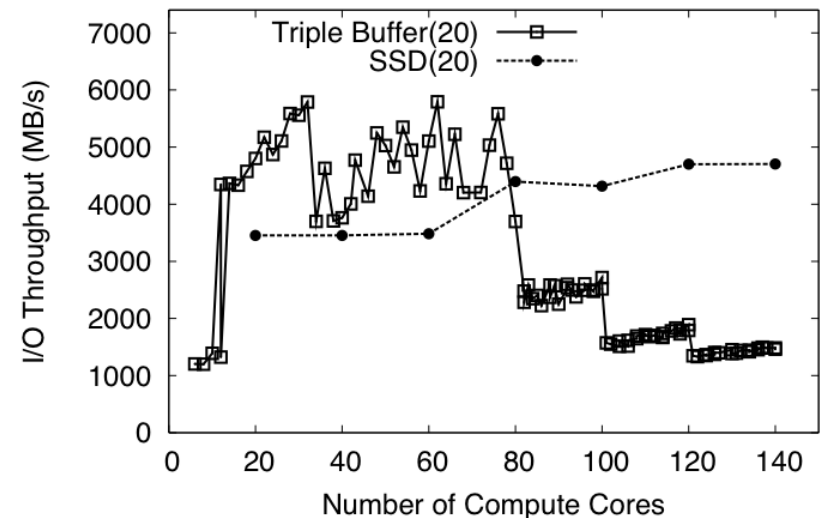
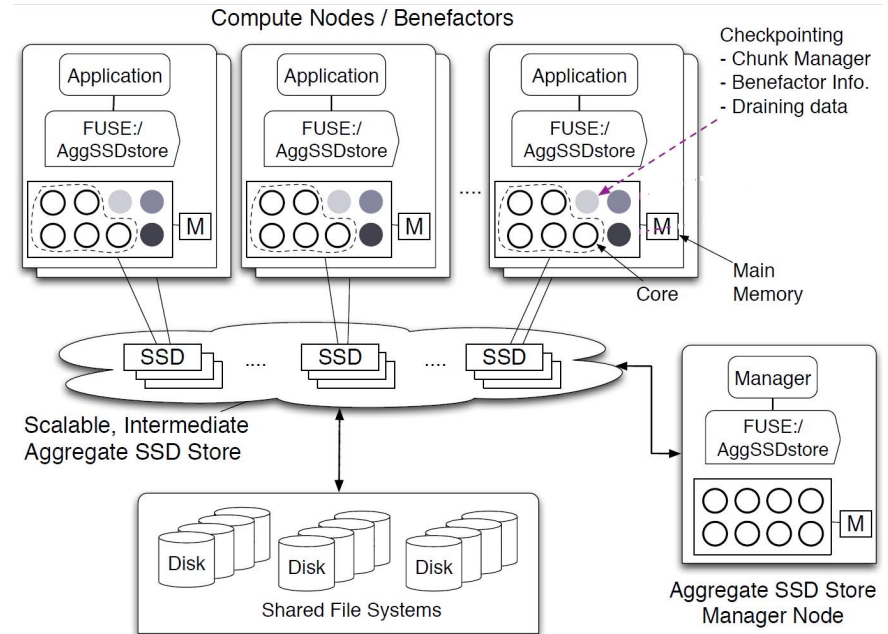
- It is really a cost problem
 - This isn't NASA's Moon Program (\$25.4 billion in 1973 Dollars - \$170 billion in 2005 Dollars - over an 11-year period)
- The challenge is to build a reliable system within a given cost budget that achieves the expected performance

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.

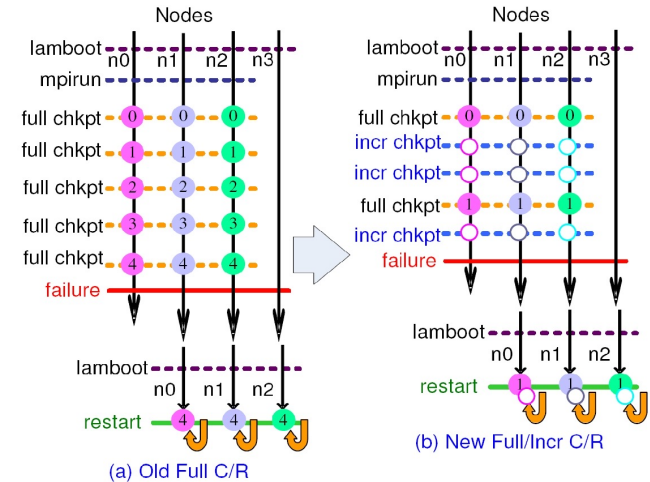
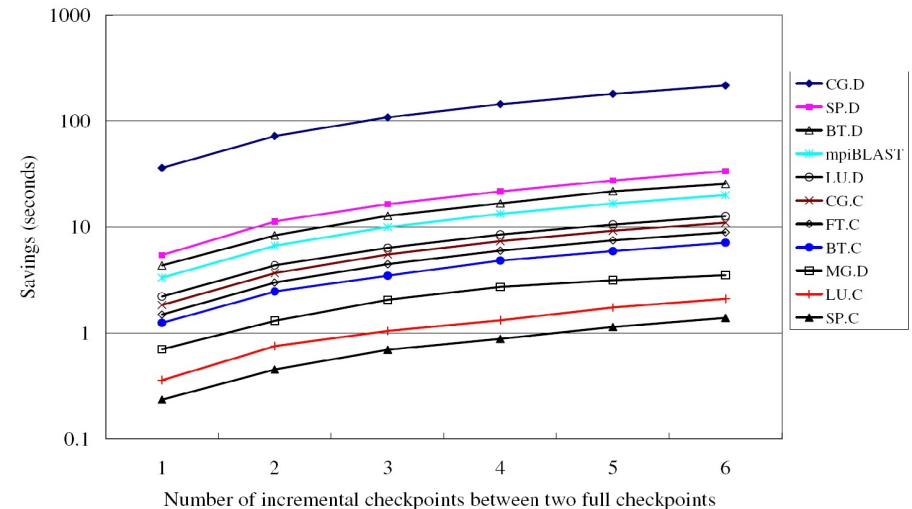


Fig. 1: Hybrid Full/Incremental C/R Mechanism vs. Full C/R



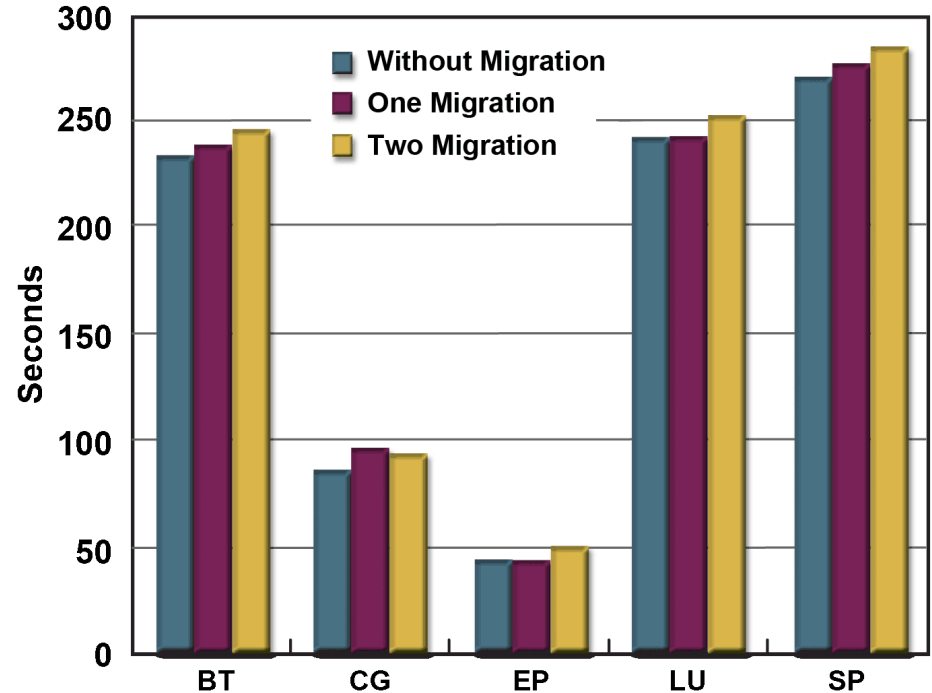
Proactive Fault Tolerance using Migration

- 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

- 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

Single migration overhead

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

Single migration duration

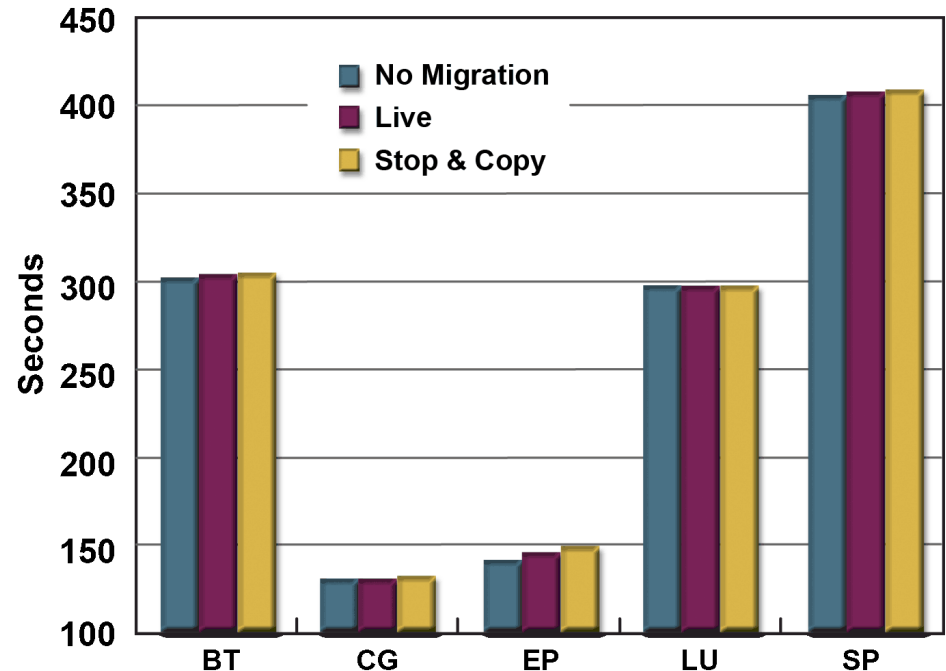
- Stop & copy : 1.0-1.9s
- Live : 2.6-6.5s

Application downtime

- Stop & copy > Live

Node eviction time

- Stop & copy < Live



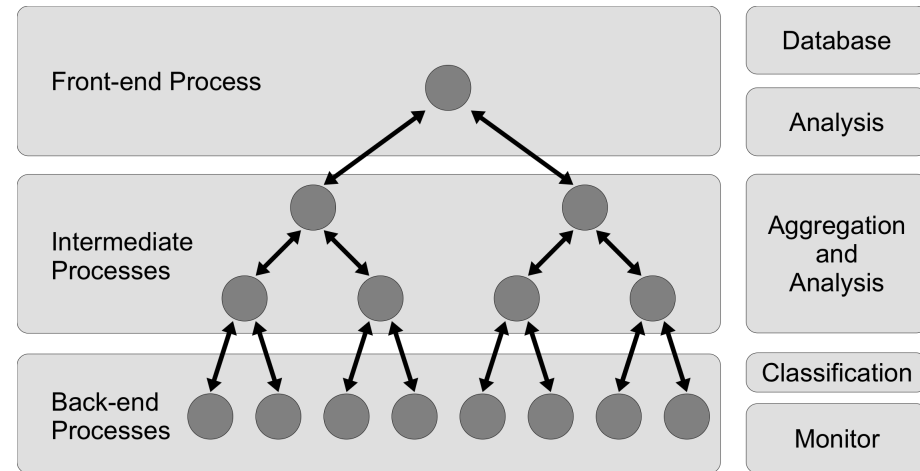
NPB runs on 16-node dual-core dual-processor Linux cluster at NCSU with AMD Opteron and Gigabit Ethernet



NC STATE UNIVERSITY

Scalable System Monitoring using MRNet

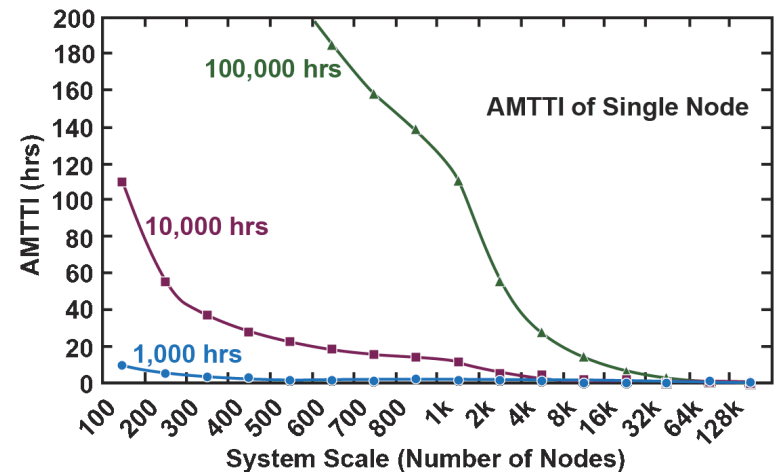
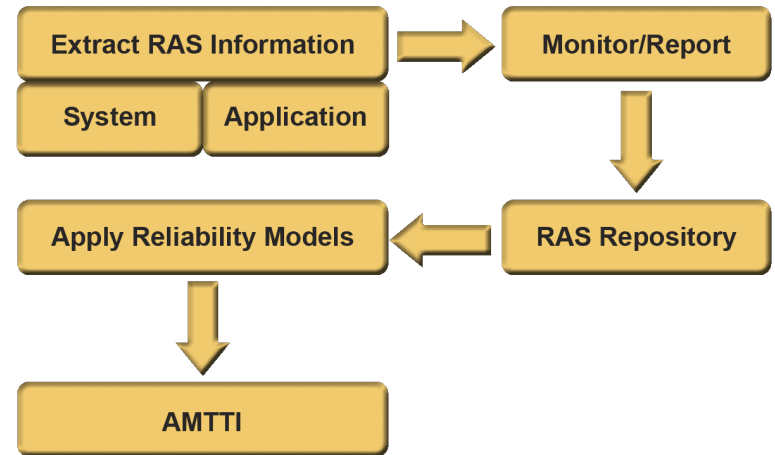
- Aggregation of metrics
- Tree-based overlay network
- Fan-in for metric data
- Fan-out for management
- Classification of data on back-end nodes
- In-flight processing on intermediate nodes
- Collection and storing on front-end node



- 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

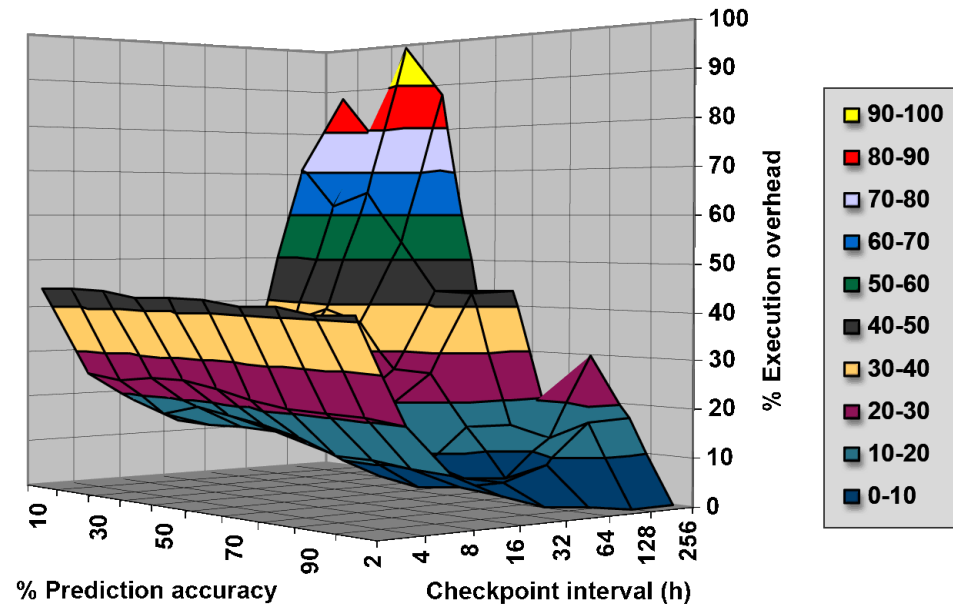
- Optimum for the given logs:
 - Prediction accuracy > 60%
 - Checkpoint interval 16-32h
- Results for higher accuracies and very low intervals are worse than only proactive or only reactive

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)*

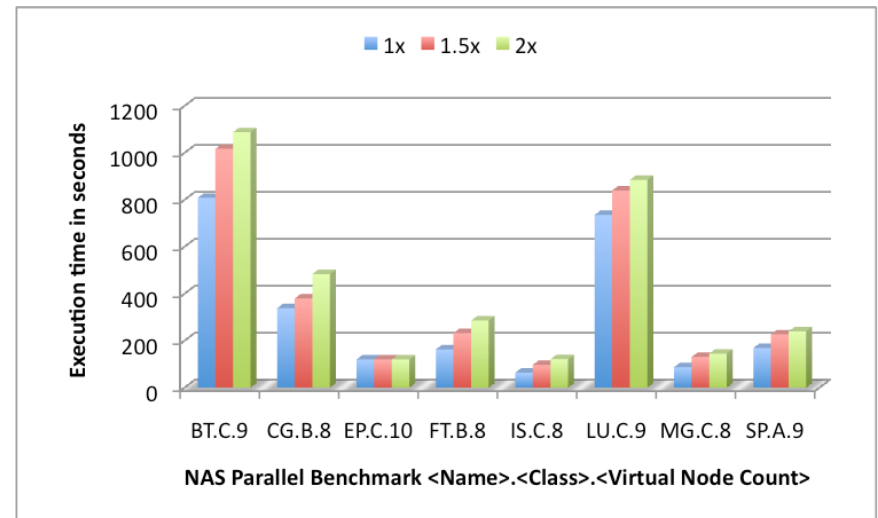
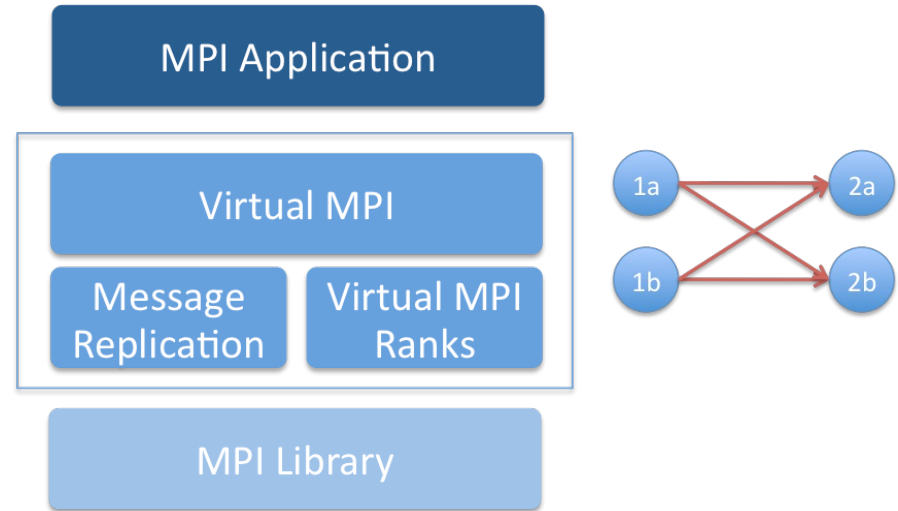
Christian Engelmann, Resilience Challenges and Solutions for Extreme-Scale Supercomputing, 2016-02-18.

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

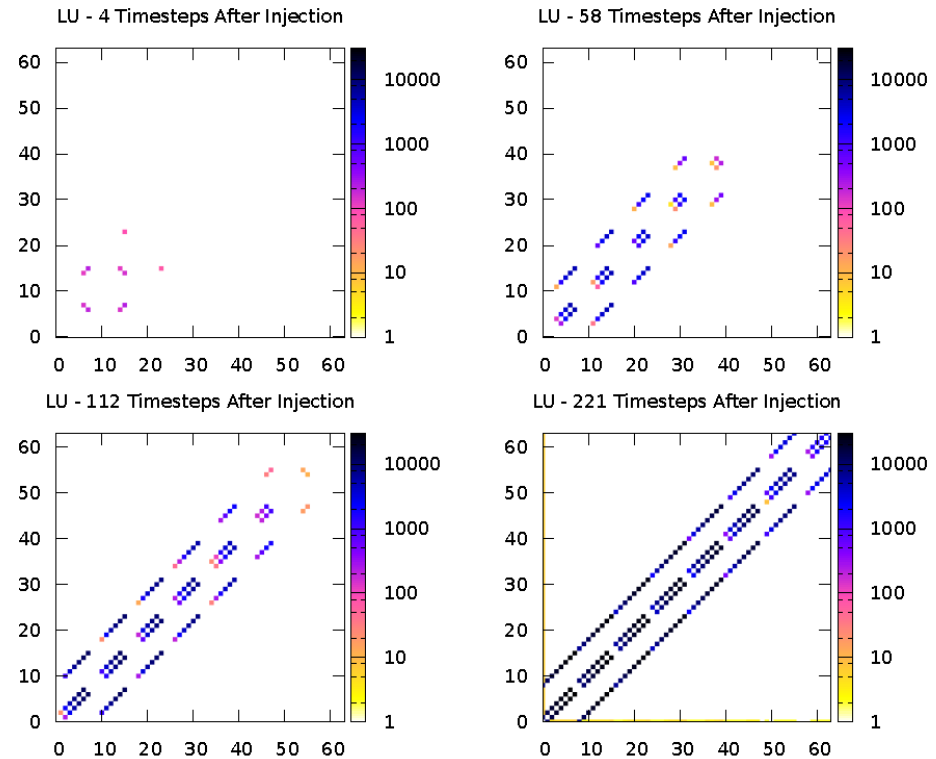


Fig. 3. NPB LU Corrupted Communication Patterns

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

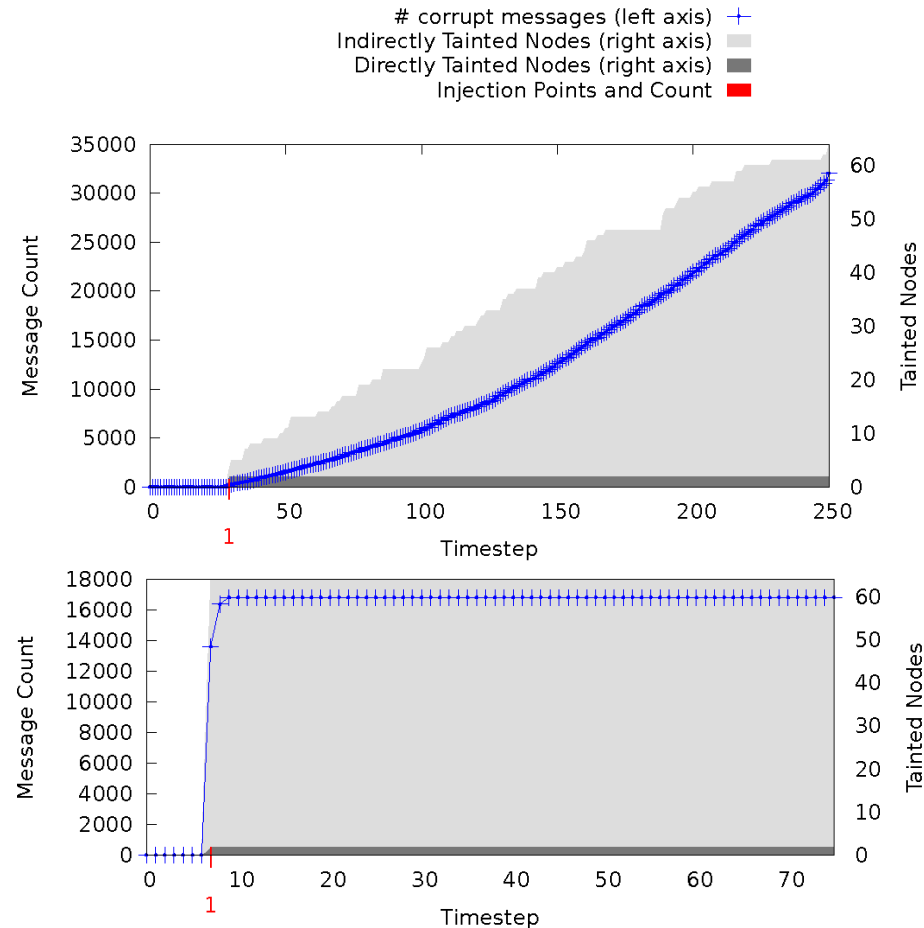


Fig. 8. NPB CG Overview of Corrupt Nodes and Messages

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

- Investigating timing errors in 180 nm 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

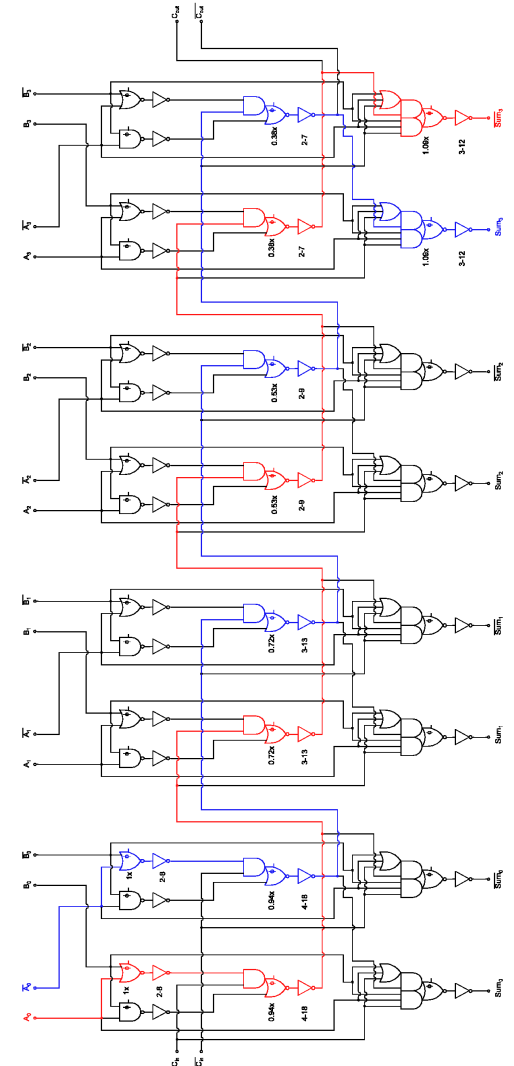


figure 6: Domino Logic Realization of a 4-bit Ripple Carry Adder

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

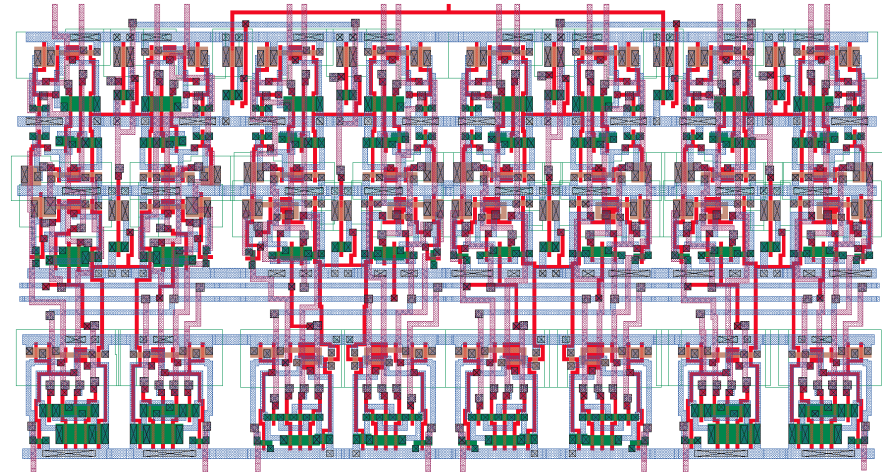


Figure 7: 4-bit Ripple Carry Adder Layout in the 0.18um TSMC process

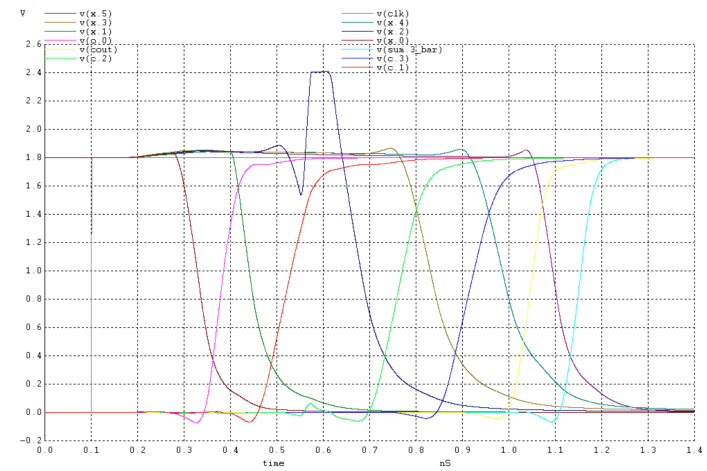
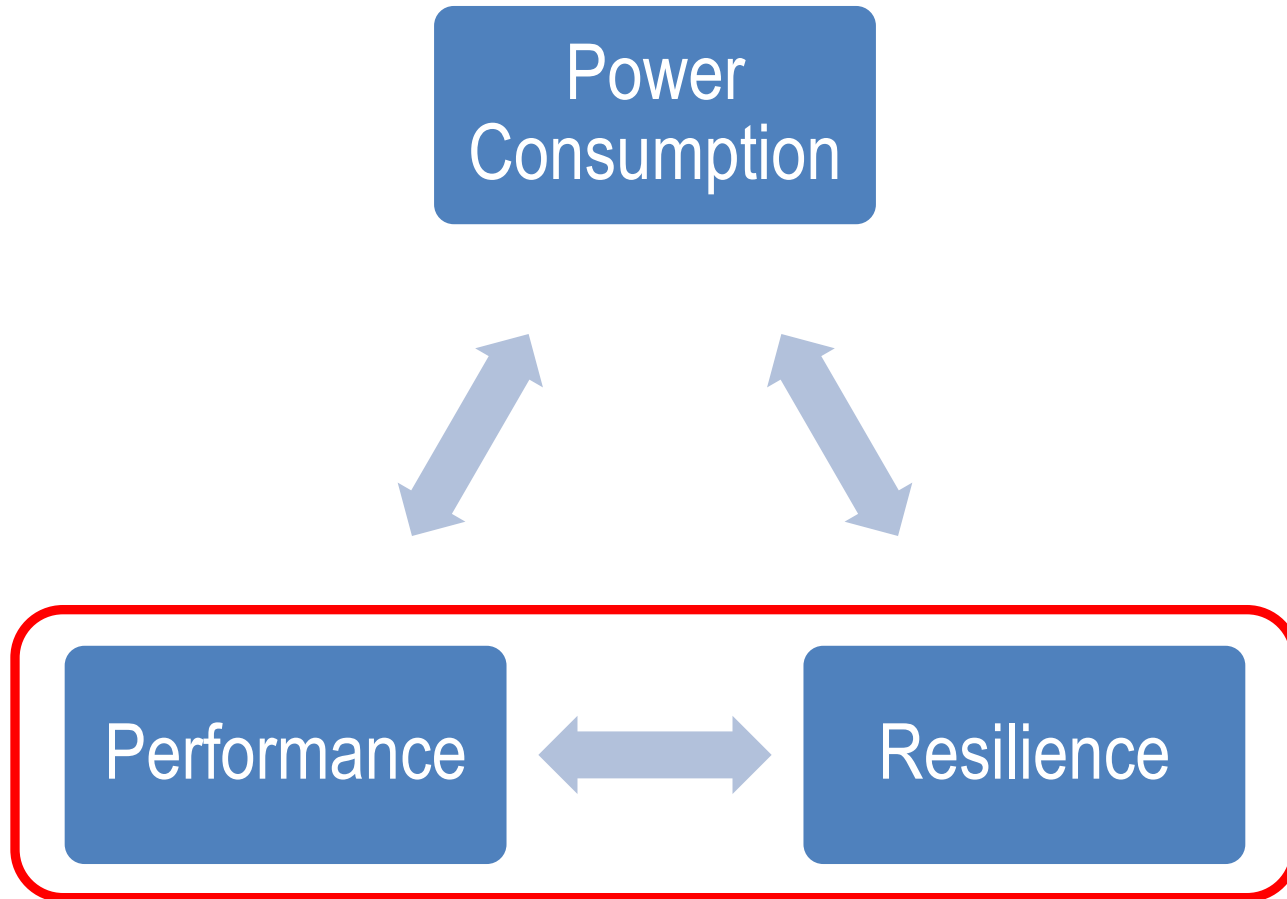


Figure 9: Voltages at the output nodes of gates on the sensitized path after an ion strike at the second a021 gate

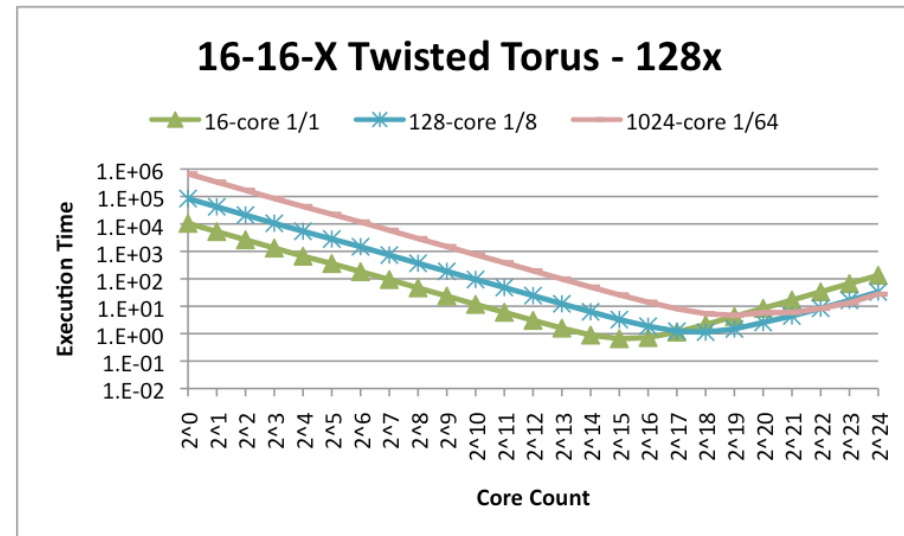
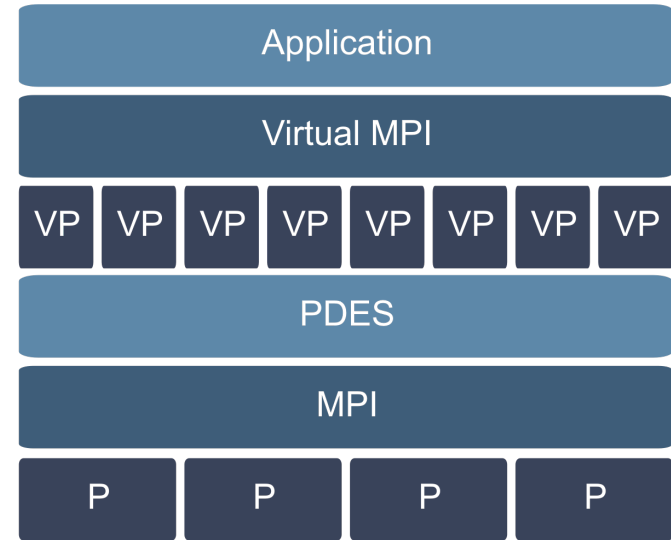
HPC System Hardware/Software Co-Design: Optimizing Trade-offs



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

xSim – The Extreme-Scale Simulator

- Combining highly oversubscribed execution, a virtual MPI, & a time-accurate PDES
 - Execution on native processor
 - Processor and network model
- 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>`

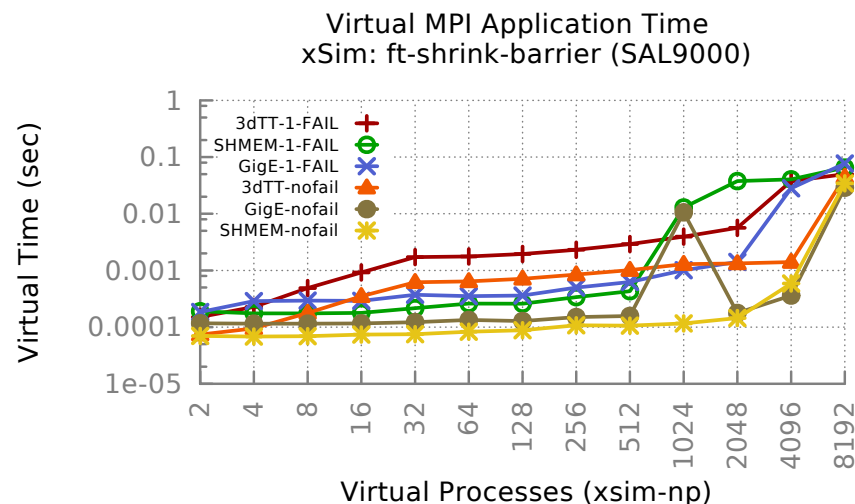


Resilience Simulation Features

- Simulated MPI process failure
 - Injection, propagation and detection in modeled architecture
- Simulated MPI application checkpoint, abort, and restart
 - Support for checkpoint/restart cycles until completion
- Simulated fault tolerant MPI
 - Support for resilient solvers using fault tolerant MPI (ULFM)
- Used in MCREX RX-Solvers project (Trilinos-based solver)

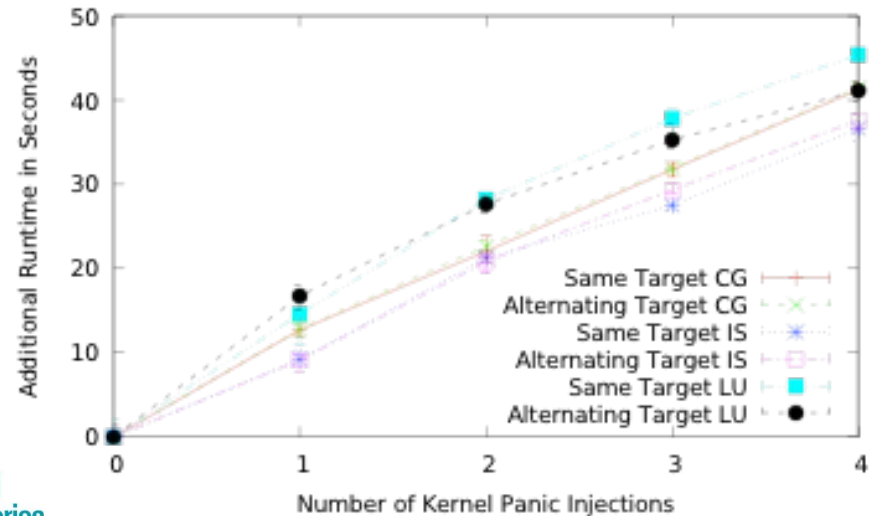
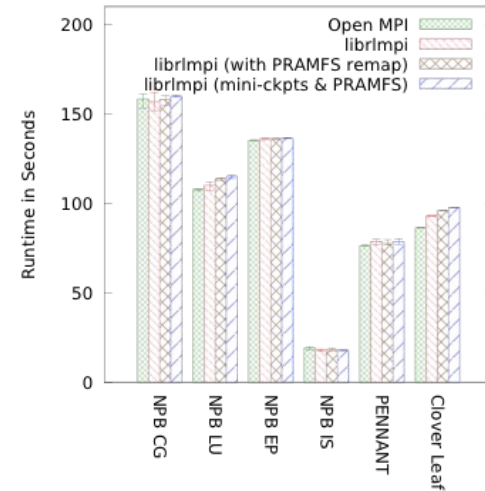
Table 1: Varying the checkpoint interval and system MTTF

$MTTF_S$	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



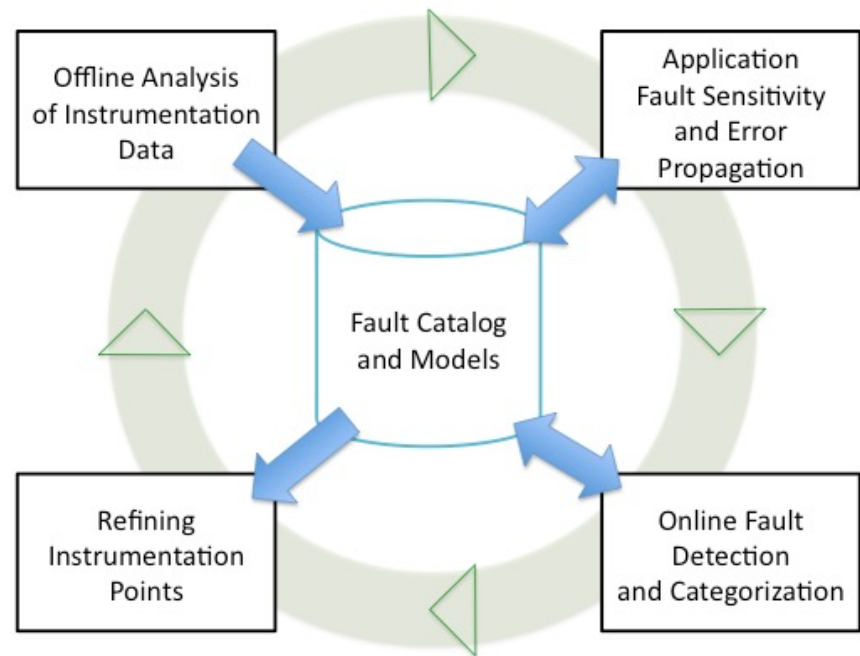
mini-ckpts: Surviving OS Failures in Persistent Memory

- Problem
 - A failure of the OS causes a failure of an otherwise healthy application
- Solution
 - Execute the application in persistent memory (PRAMFS in DRAM) that is able to survive OS failures/reboots
 - Track OS state used by the application and MPI for recovery
 - Rejuvenate (warm reboot) the OS
 - Transparently continue to execute the application without loss of state/progress
- Recent results
 - Prototype implementation supports OpenMP and MPI applications
 - Rejuvenation/recovery takes 3-6 s
 - Failure-free runtime overhead 3-5%



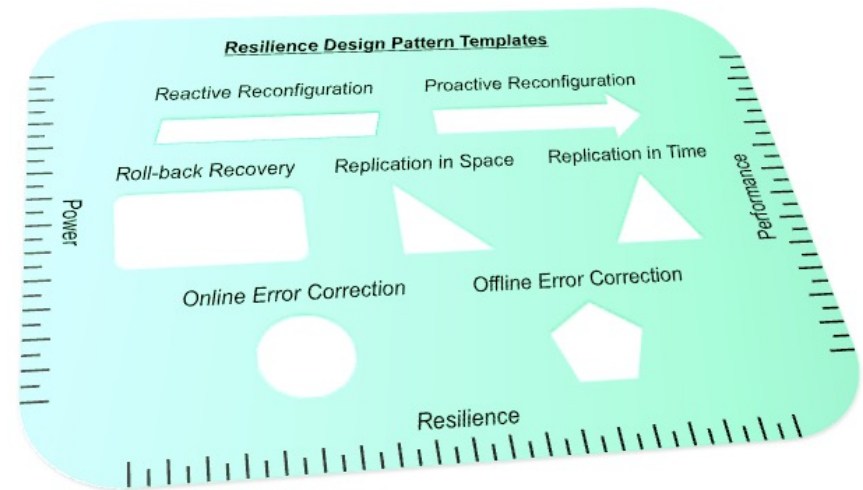
Characterizing Faults, Errors, and Failures in Extreme-scale Systems

- This project identifies, categorizes and models the fault, error and failure properties of DOE systems
- It develops a fault taxonomy, catalog and models that capture current systems and extrapolate this knowledge to exascale



Resilience Design Patterns: A Structured Approach to Resilience at Extreme Scale

- DOE Early Career Award project
- Using a novel resilience design pattern concept, this project:
 - Identifies and evaluates repeatedly occurring resilience problems
 - Coordinates solutions throughout hardware and software stack
- Approach:
 - Develop methods and metrics to evaluate resilience in HPC
 - Establish mechanisms and interfaces for coordination
 - Optimize performance, resilience, and power at design time and runtime



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
 - Performance/resilience/power trade-off is not understood holistically
 - Coordination across the stack is missing
 - End-to-end management of resilience is needed
 - Resilient parallel programming models do not exist
 - Operational resilience policies do not go beyond checkpoint/restart

Acknowledgements

- **Oak Ridge National Laboratory (ORNL):** C. Engelmann, H. Park, D. Tiwari, S. Hukerikar, S. Gupta, S. L. Scott, G. Vallée, T. Naughton, C. Wang, A. Tikotekar, G. Ostrouchov, S. Vazhkudai, Y. Kim, G. Shipman, S. Boehm, I. Jones, and F. Lauer
- **Argonne National Laboratory:** M. Snir, S. Di, and R. Gupta
- **Lawrence Livermore National Laboratory:** *M. Schulz and I. Laguna*
- **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®**



Questions



Christian Engelmann, Resilience Challenges and Solutions for Extreme-Scale Supercomputing, 2016-02-18.