System Resilience Research at ORNL in the Context of HPC

Presented by

Christian Engelmann

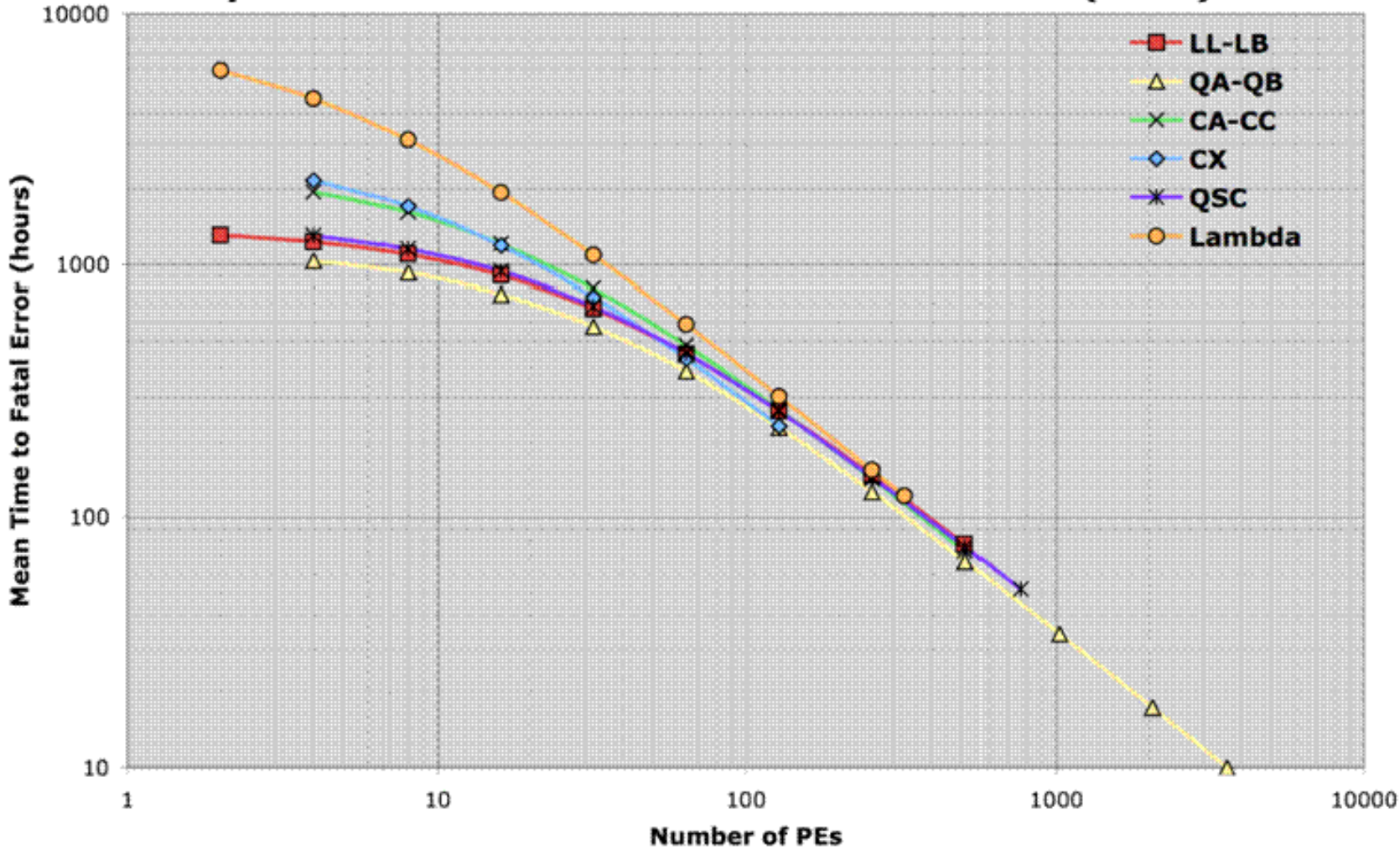
Computer Science Research Group
Computer Science and Mathematics Division

Motivation

- **Large-scale 1 PFlop/s systems have arrived:**
  - **#1: LANL Roadrunner with 129,600 processor cores**
  - **#2: ORNL Jaguar with 150,152 processor cores**
- **Other large-scale systems exist**
  - **LLNL @ 212,992, ANL @ 163,840, TACC @ 62,976**
- **The trend is toward even larger-scale systems**
- **Significant increase in component count and complexity**
- **Expected matching increase in failure frequency**
- **Checkpoint/restart is becoming less and less efficient**

Oak Ridge National Laboratory

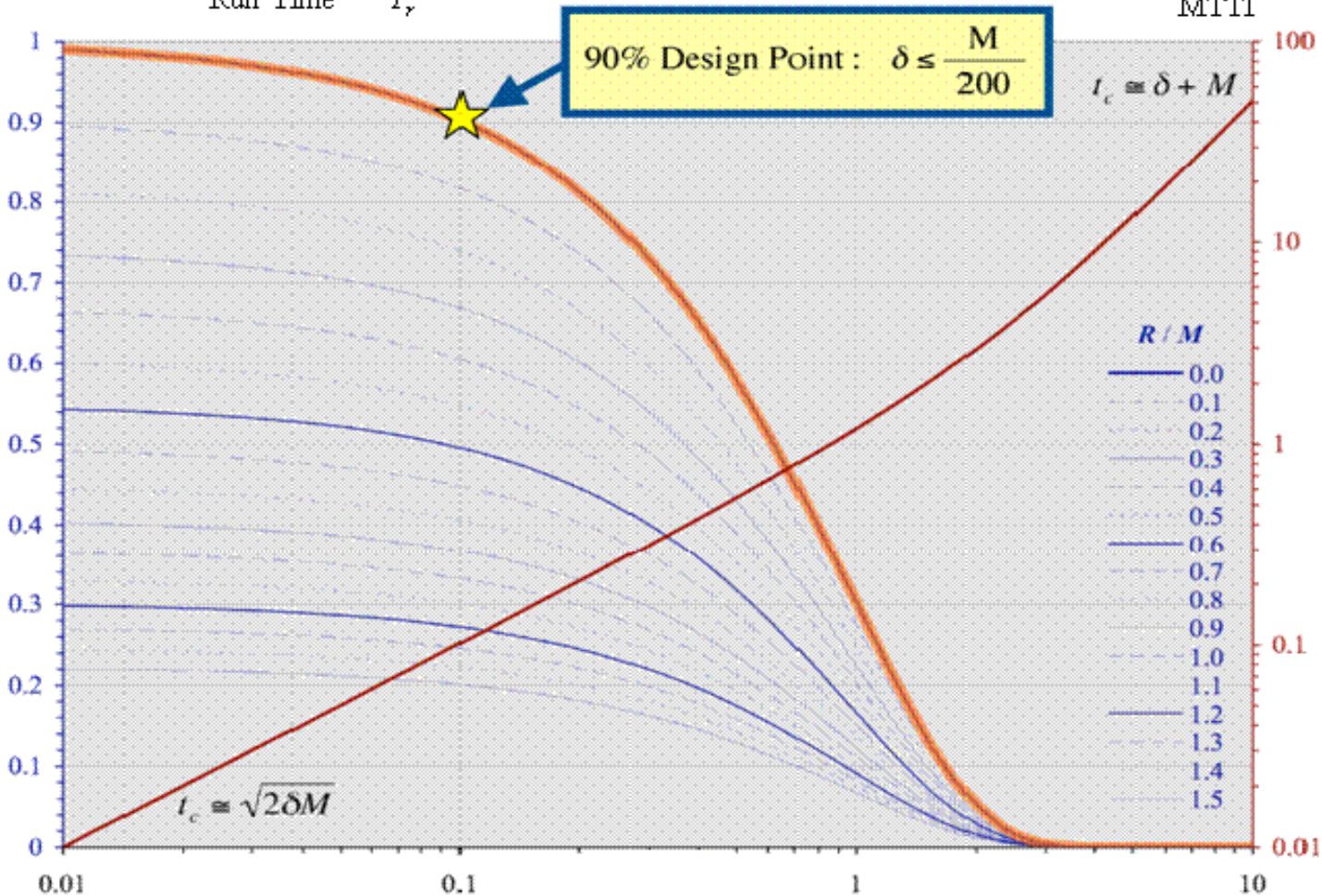
Summary of Application Reliability as Measured from System Data Across 21 Los Alamos Platforms (2006)



J. T. Daly. Methodology and metrics for quantifying application throughput. In Proceedings of the Nuclear Explosives Code Developers Conference (NECDC) 2006, Los Alamos, NM, USA, Oct. 23-27, 2006.

$$\text{Checkpoint Efficiency} = \frac{\text{Solve Time}}{\text{Run Time}} = \frac{T_s}{T_r}$$

$$\frac{\text{Checkpoint Interval}}{\text{MTTI}} = \frac{T_c}{M}$$



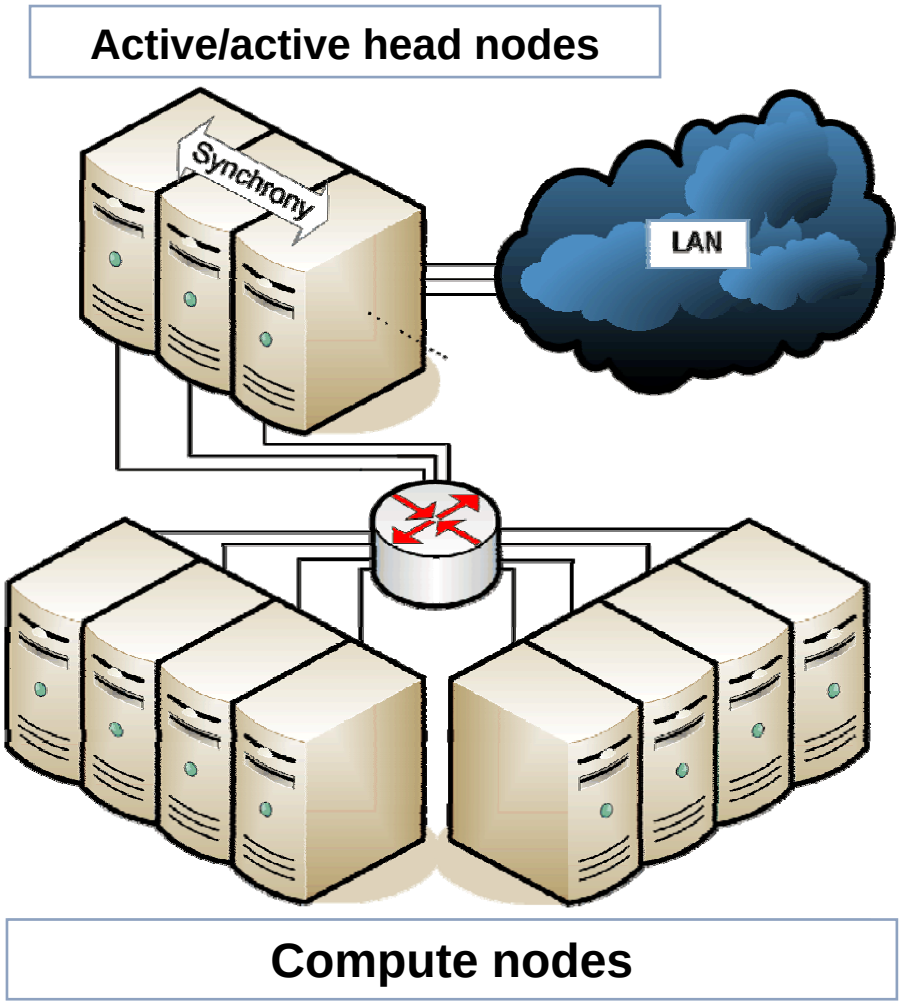
$$\text{Dump Time Parameter} = \sqrt{\frac{2\,\text{Dump Time}}{\text{MTTI}}} = \sqrt{\frac{2\delta}{M}}$$

J. T. Daly. Methodology and metrics for quantifying application throughput. In Proceedings of the Nuclear Explosives Code Developers Conference (NECDC) 2006, Los Alamos, NM, USA, Oct. 23-27, 2006.

Research and development activities

- **<u>Efficient redundancy strategies</u> for HPC head/service nodes for high availability and high performance of critical services**
- **<u>Reactive fault tolerance</u> for HPC compute nodes utilizing the job pause approach and checkpoint placement adaptation**
- **<u>Proactive fault tolerance</u> using preemptive migration of computation away from compute nodes that are about to fail**
- **<u>Reliability analysis</u> for identifying pre-fault indicators, predicting failures, and modeling and monitoring reliability**
- **<u>Holistic fault tolerance</u> through combination of adaptive proactive and reactive fault tolerance mechanisms**

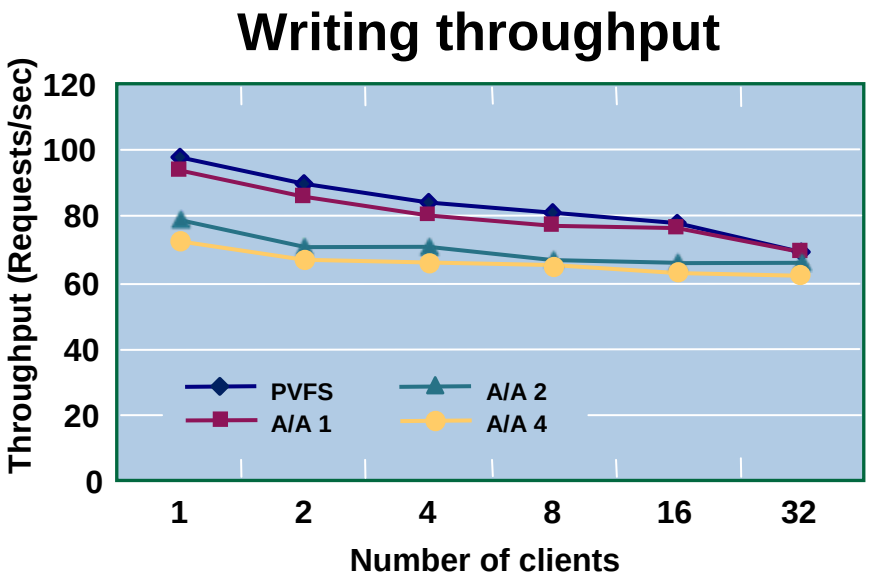
Symmetric active/active redundancy

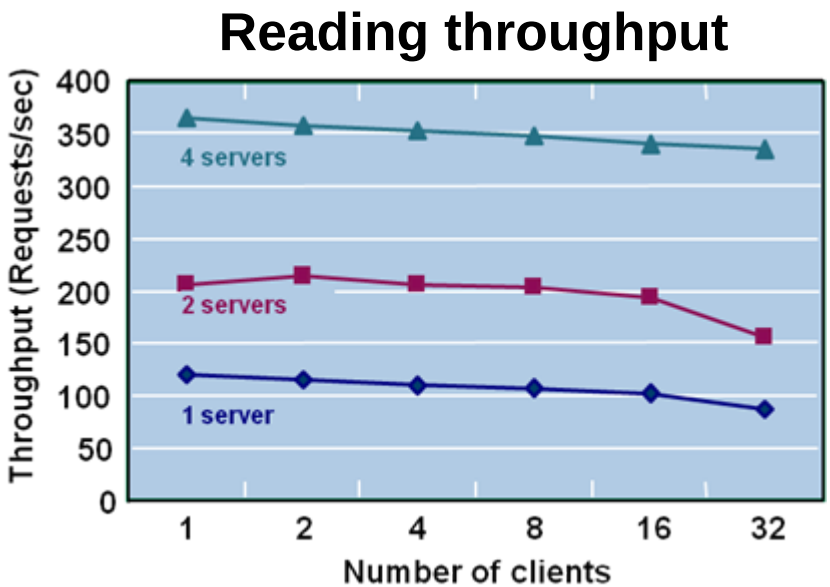


- Many active head nodes
- Workload distribution
- Symmetric replication between head nodes
- Continuous service
- Always up to date
- No fail-over necessary
- No restore-over necessary
- Virtual synchrony model
- Complex algorithms

➡ Prototypes for Torque and Parallel Virtual File System metadata server

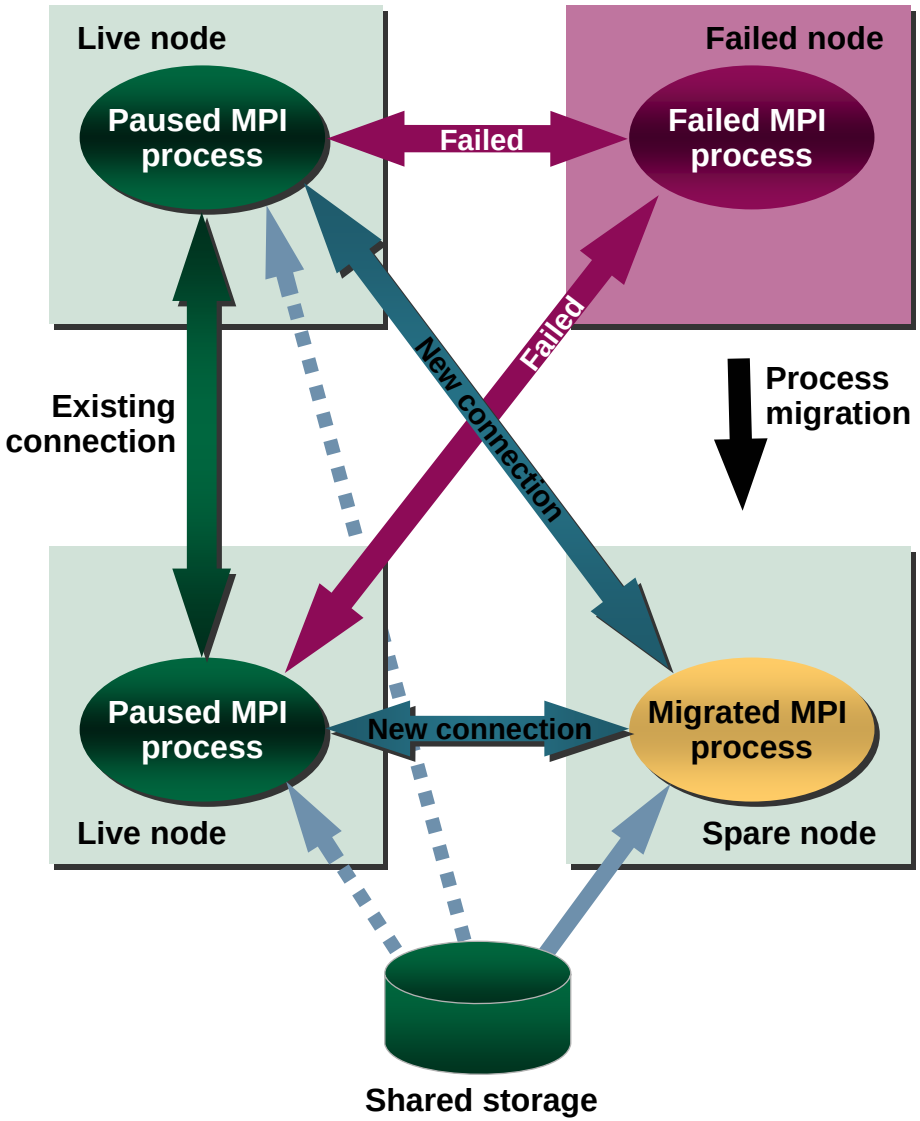
Symmetric active/active Parallel Virtual File System metadata server





Nodes	Availability	Est. annual downtime
1	98.58%	5d, 4h, 21m
2	99.97%	1h, 45m
3	99.9997%	1m, 30s

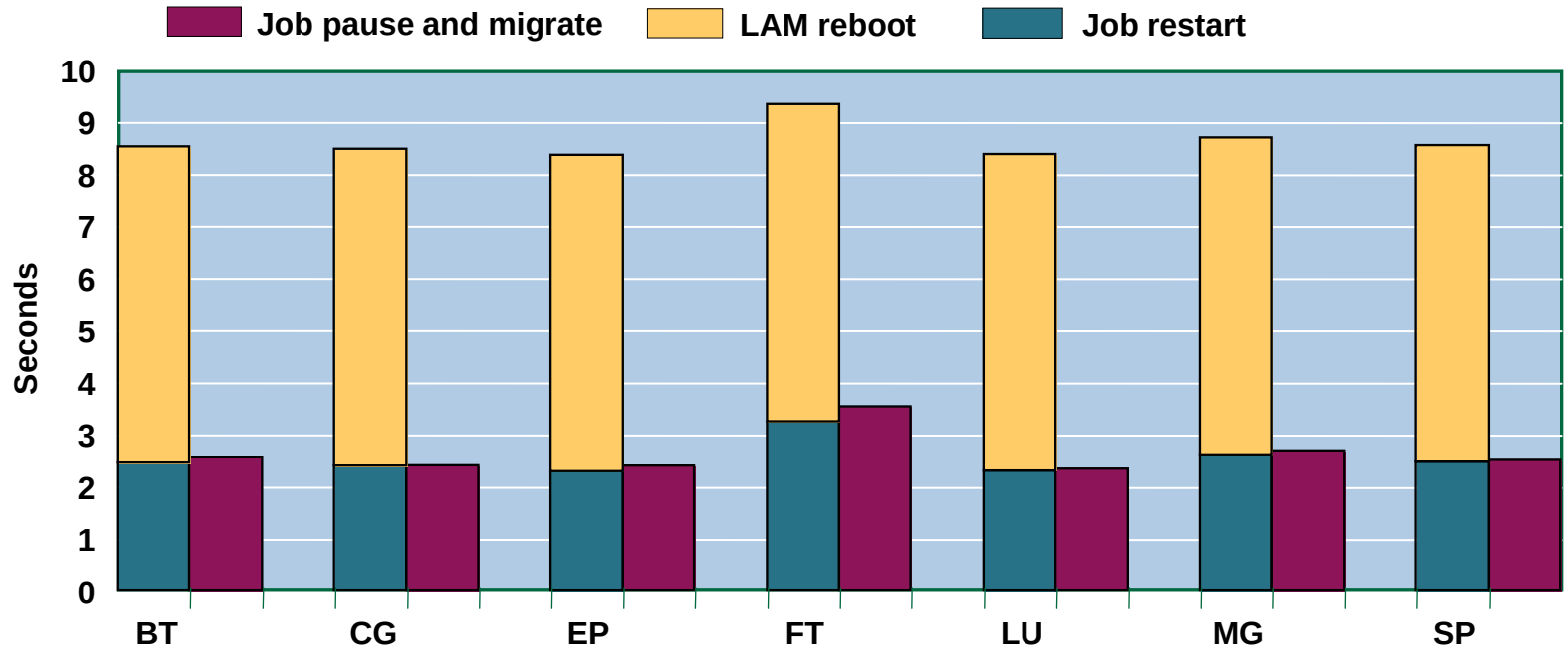
Reactive fault tolerance for HPC with LAM/MPI+BLCR job-pause mechanism



- **Operational nodes: Pause**
  - **BLCR reuses existing processes**
  - **LAM/MPI reuses existing connections**
  - **Restore partial process state from checkpoint**
- **Failed nodes: Migrate**
  - **Restart process on new node from checkpoint**
  - **Reconnect with paused processes**
- **Scalable MPI membership management for low overhead**

➡ Efficient, transparent, and automatic failure recovery

LAM/MPI+BLCR job pause performance



- **3.4% overhead over job restart, but**
  - **No LAM reboot overhead**
  - **Transparent continuation of execution**
- **No requeue penalty**
- **Less staging overhead**

Reactive vs. Proactive Fault Tolerance

- **Reactive fault tolerance**
  - **Keeps parallel applications alive through recovery from experienced failures**
  - **Employed mechanisms react to failures**
  - **Examples: Checkpoint/restart, message logging/replay**
- **Proactive fault tolerance**
  - **Keeps parallel applications alive by avoiding failures through preventative measures**
  - **Employed mechanisms anticipate failures**
  - **Example: Preemptive migration**

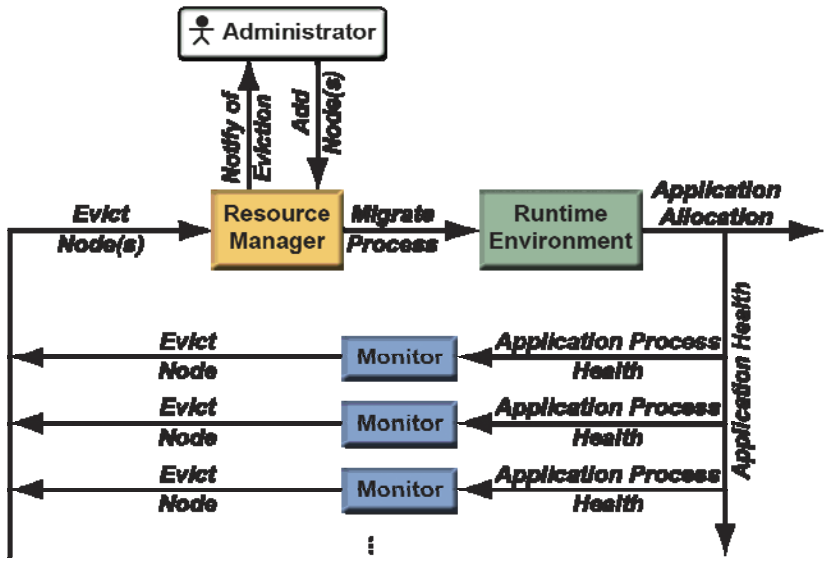
Oak Ridge National Laboratory

Proactive Fault Tolerance using Preemptive Migration

- **Relies on a feedback-loop control mechanism**
  - **Application health is constantly monitored and analyzed**
  - **Application is reallocated to improve its health and 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, such as random bit flips**

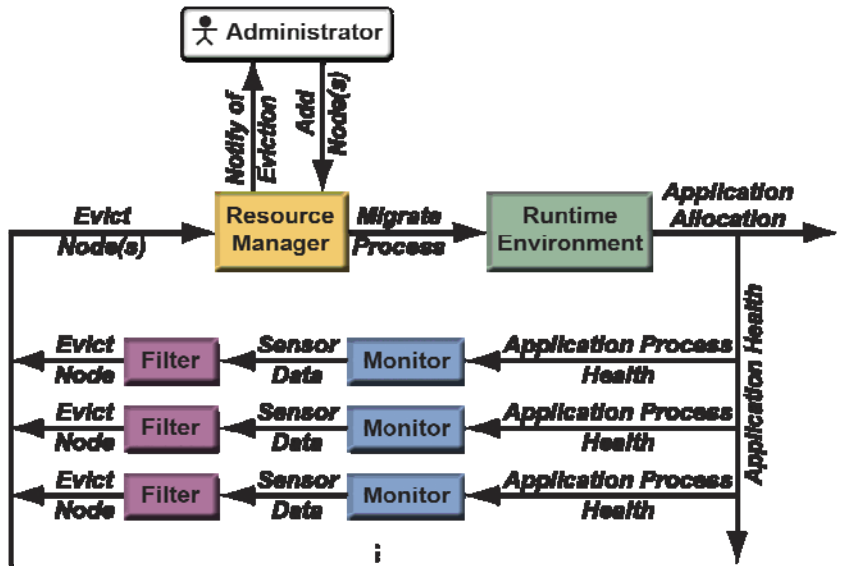
Type 1 Feedback-Loop Control Architecture

- **Alert-driven coverage**
  - **Basic failures**
- **No evaluation of application health history or context**
  - **Prone to false positives**
  - **Prone to false negatives**
  - **Prone to miss real-time window**
  - **Prone to decrease application heath through migration**
  - **No correlation of health context or history**



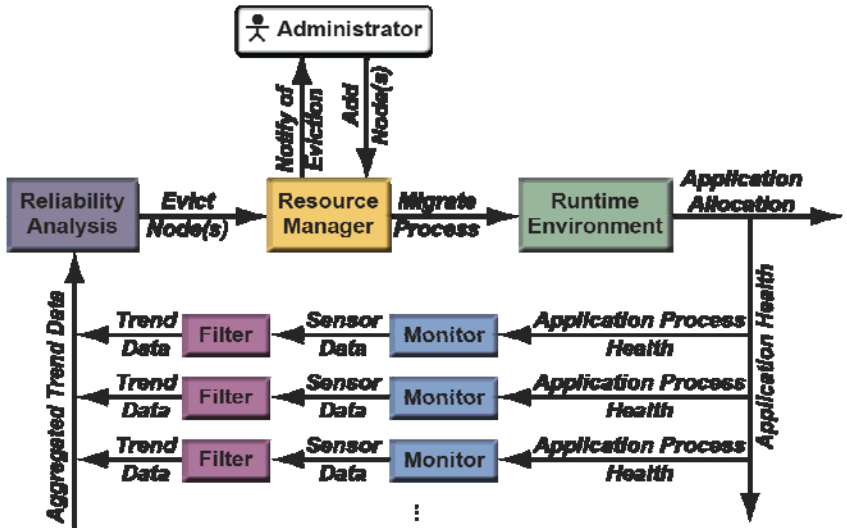
Type 2 Feedback-Loop Control Architecture

- **Trend-driven coverage**
  - **Basic failures**
  - **Less false positives/negatives**
- **No evaluation of application reliability**
  - **Prone to miss real-time window**
  - **Prone to decrease application heath through migration**
  - **No correlation of health context or history**



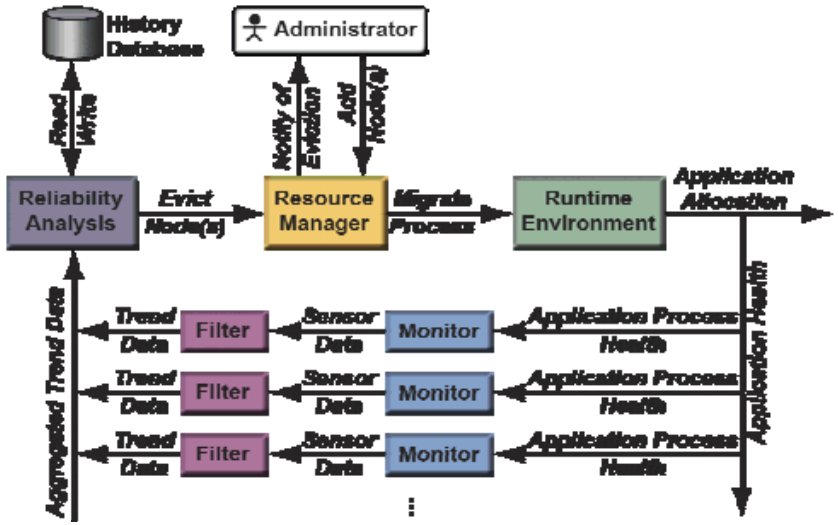
Type 3 Feedback-Loop Control Architecture

- **Reliability-driven coverage**
  - **Basic and correlated failures**
  - **Less false positives/negatives**
  - **Able to maintain real-time window**
  - **Does not decrease application heath through migration**
  - **Correlation of short-term health context and history**
- **No correlation of long-term health context or history**
  - **Unable to match system and application reliability patterns**



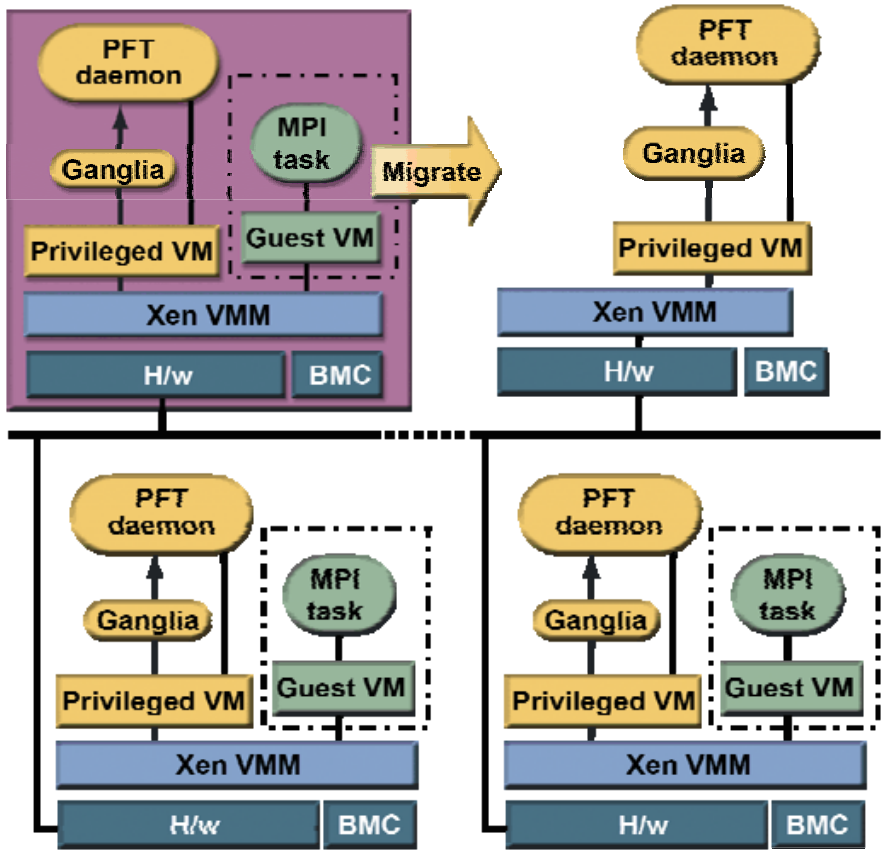
Type 4 Feedback-Loop Control Architecture

- **Reliability-driven coverage of failures and anomalies**
  - **Basic and correlated failures, anomaly detection**
  - **Less prone to false positives**
  - **Less prone to false negatives**
  - **Able to maintain real-time window**
  - **Does not decrease application heath through migration**
  - **Correlation of short and long-term health context & history**



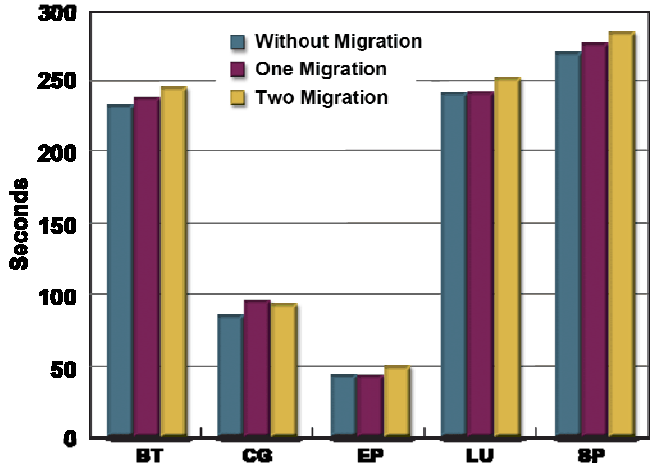
VM-level Preemptive Migration using Xen

- **Type 1 system setup**
  - **Xen VMM on entire system**
  - **Host OS for management**
  - **Guest OS for computation**
  - **Spare nodes without Guest OS**
  - **System monitoring in Host OS**
  - **Decentralized scheduler/load balancer using Ganglia**
- **Deteriorating node health**
  - **Ganglia threshold trigger**
  - **Migrate guest OS to spare**
  - **Utilize Xen's migration facility**



VM-level Migration Performance Impact

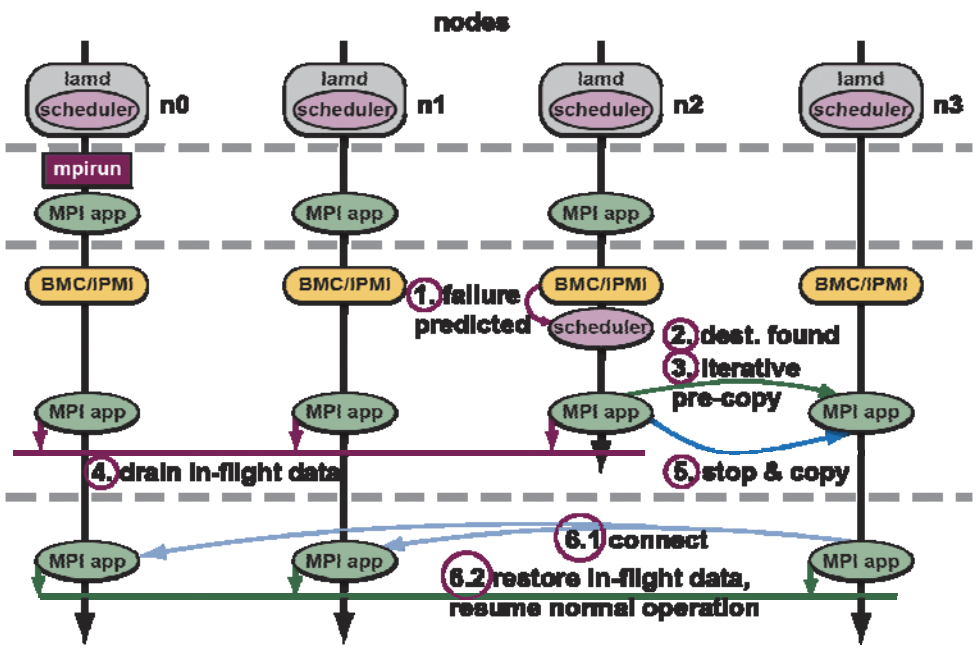
- **Single node migration**
  - **0.5-5% longer run time**
- **Double node migration**
  - **2-8% longer run time**
- **Migration duration**
  - **Stop & copy : 13-14s**
  - **Live : 14-24s**
- **Application downtime**
  - **Stop & copy > Live**



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

Process-Level Preemptive Migration w/ BLCR

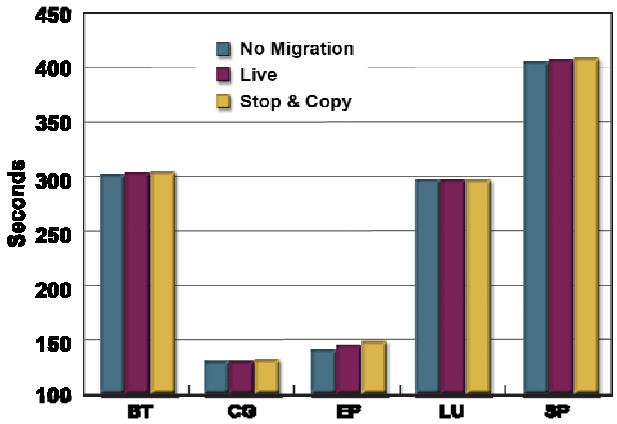
- **Type 1 system setup**
  - **LAM/MPI with Berkeley Lab Checkpoint/Restart (BLCR)**
  - **Per-node health monitoring**
    - **Baseboard management controller (BMC)**
    - **Intelligent platform management interface (IPMI)**
  - **New decentralized scheduler/ load balancer in LAM**
  - **New process migration facility in BLCR (stop&copy and live)**



- **Deteriorating node health**
  - **Simple threshold trigger**
  - **Migrate process to spare**

Process-Level Migration Performance Impact

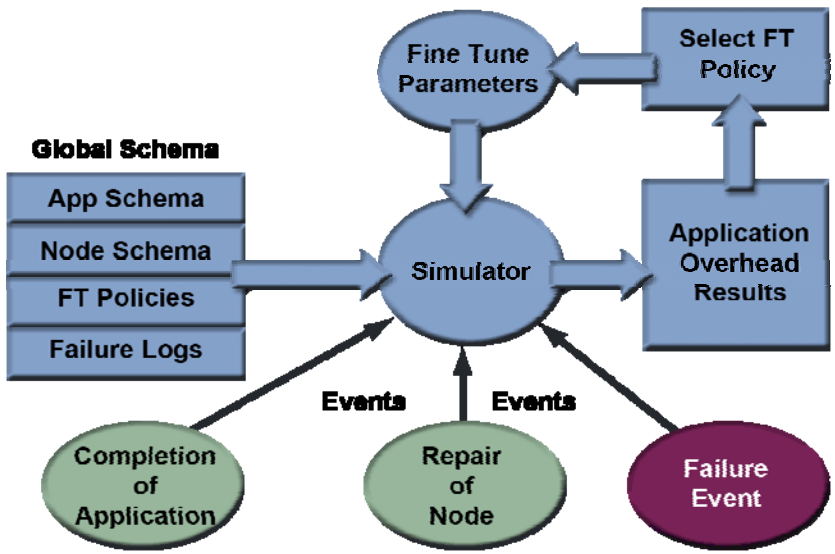
- **Single node migration overhead**
  - **Stop & copy : 0.09-6 %**
  - **Live : 0.08-2.98%**
- **Single node migration duration**
  - **Stop & copy : 1.0-1.9s**
  - **Live : 2.6-6.5s**
- **Application downtime**
  - **Stop & copy > Live**
- **Node eviction time**
  - **Stop & copy < Live**



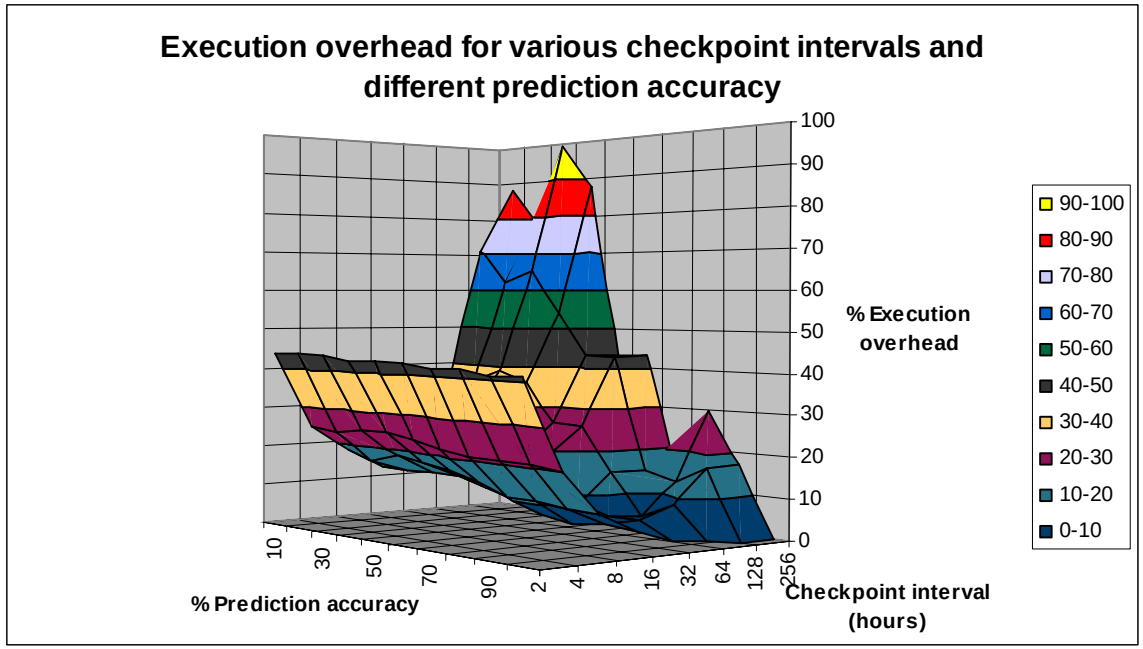
16-node Linux cluster at NCSU with dual core, dual-processor AMD Opteron and Gigabit Ethernet

Simulation of Fault Tolerance Policies

- **Evaluation of fault tolerance policies**
  - **Reactive only**
  - **Proactive only**
  - **Reactive/proactive combination**
- **Evaluation of fault tolerance parameters**
  - **Checkpoint interval**
  - **Prediction accuracy**
- **Event-based simulation framework using actual HPC system logs**
- **Customizable simulated environment**
  - **Number of active and spare nodes**
  - **Checkpoint and migration overheads**



Combination of proactive and reactive fault tolerance: Simulation example



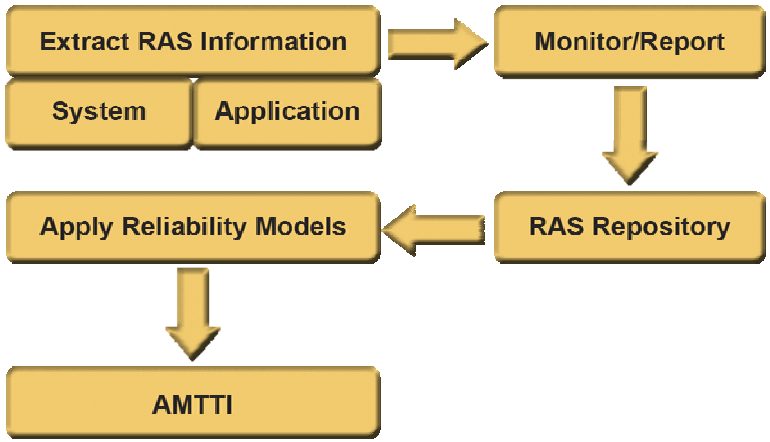
- **Best: Prediction accuracy >60% and checkpoint interval 16-32 hours**
- **Better than only proactive or only reactive**
- **Results for higher prediction accuracies and very low checkpoint intervals are worse than only proactive or only reactive**

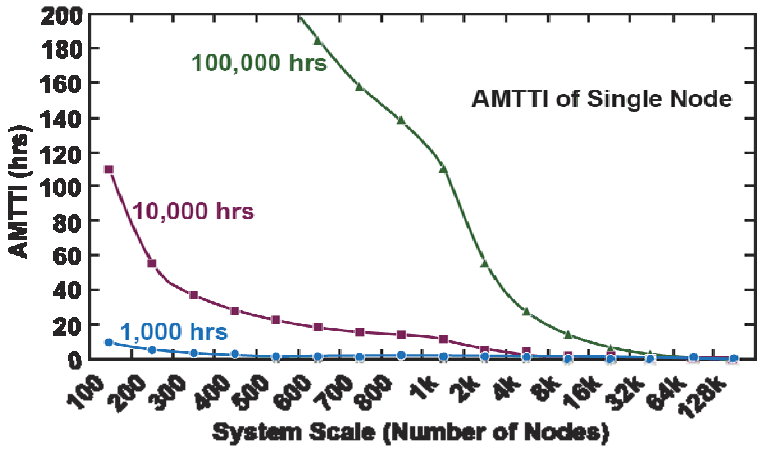
Number of processes	**125**
Active nodes / Spare nodes	**125 / 12**
Checkpoint overhead	**50 min/checkpoint**
Migration overhead	**1 min/migration**

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

Research in Reliability Modeling

- **Type 3 system setup**
  - **Monitoring of application and system health**
  - **Recording of application and system health monitoring data**
  - **Reliability analysis on recorded data**
  - **Application mean-time to interrupt (AMTTI) estimation**
- **Type 4 system setup**
  - **Additional recording of application interrupts**
  - **Reliability analysis on recent and historical data**





Ongoing Work

- **Development of a unified framework for Type 1-4**
  - **Unified interfaces between components**
  - **Extendable RAS engine core interfacing with**
    - **Monitoring data aggregation/filtering component**
    - **Job and resource management service**
    - **Process/VM migration mechanism**
    - **Online/offline reliability modeling**
- **Research in scalable monitoring data aggregation/filtering**
  - **In-flight monitoring data aggregation/filtering**
  - **Scalable, fault tolerant overlay reduction networks**
  - **Fully distributed monitoring data processing (e.g. heat eq.)**
- **Finding the right metrics**

Acknowledgements

- **Investigators at Oak Ridge National Laboratory:**
  - ***Stephen L. Scott [Lead PI]*, Christian Engelmann, Geoffroy Vallée, Thomas Naughton, Anand Tikotekar, George Ostrouchov**
- **Investigators at Louisiana Tech University:**
  - ***Chokchai (Box) Leangsuksun [Lead PI]*, Nichamon Naksinehaboon, Raja Nassar, Mihaela Paun**
- **Investigators at North Carolina State University:**
  - ***Frank Mueller [Lead PI]*, Chao Wang, Arun Nagarajan, Jyothish Varma**
- **Investigators at Tennessee Technological University:**
  - **Xubin (Ben) He, Li Ou, Xin Chen**
- **Funding sources:**
  - **U.S. Department of Energy, National Science Foundation, Oak Ridge National Laboratory**

Contacts regarding HPC RAS research

Stephen L. Scott

Computer Science Research Group
Computer Science and Mathematics Division
(865) 574-3144
scottsl@ornl.gov

Christian Engelmann

Computer Science Research Group
Computer Science and Mathematics Division
(865) 574-3132
engelmannc@ornl.gov

www.fastos.org/ras