System Software Research for Extreme-Scale Computing

SAND2010-1702P

LCF Seminar

March 22, 2010

Ron Oldfield
Sandia National Laboratories

Team Members

Ron Brightwell
Kurt Ferreira
Rolf Riesen
Jim Laros
Sue Kelly
Todd Kordenbrock

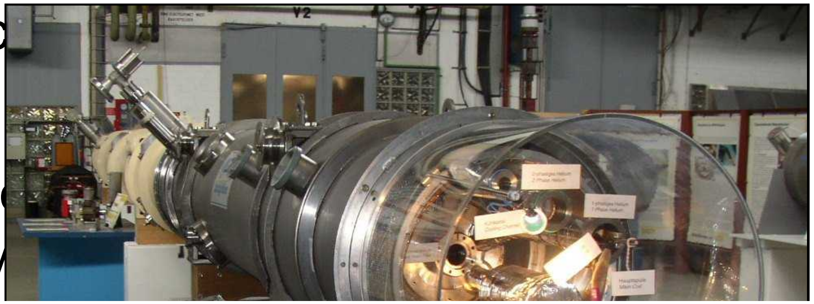
Sandia National Laboratories
Projects supported by ASC and LDRD programs

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.

Our Traditional View of Capability Systems

"MPP systems are the particle accelerators of the computing world" [Bill Camp]

– IBM RoadRunner: 12,900 Cell, 12,960 cores



Supercomputers should be thought of as highly specialized instruments for scientific discovery

(© Sandia Corporation)

(Courtesy of Wikipedia)

Sandia National Laboratories

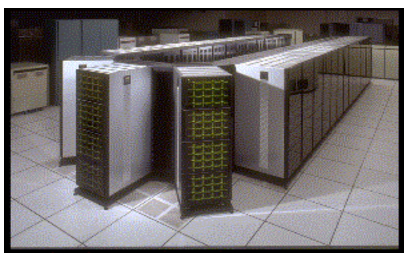
Sandia Has a Long History in MPP Architectures and System Software

1990

nCUBE2

- Sandia’s first large MPP
- Achieved Gflops performance on applications

1993



Paragon

- Tens of users
- First periods processing MPP
- World record performance
- Routine 3D simulations
- SUNMOS lightweight kernel

1997

ASCI Red

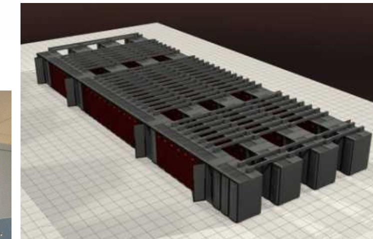
- Production MPP
- Hundreds of users
- Red & Black partitions
- Improved interconnect
- High-fidelity coupled 3-D physics
- Puma/Cougar lightweight kernel

1999

Cplant

- Commodity-based supercomputer
- Hundreds of users
- Enhanced simulation capacity
- Linux-based OS licensed for commercialization

2004



Red Storm

- 41 Tflops
- Custom interconnect
- Purpose built RAS
- Highly balanced and scalable
- Catamount lightweight kernel

Sandia National Laboratories

Outline of Our Plans for the INCITE Allocation

INCITE provides platforms necessary to continue research in system software

- Research Activities
  - Lightweight Kernel OS and Virtualization
  - Resilience
  - Scalable I/O
  - Power Efficiency and Utilization
  - Debugging
- System Requirements
- Our first set of experiments

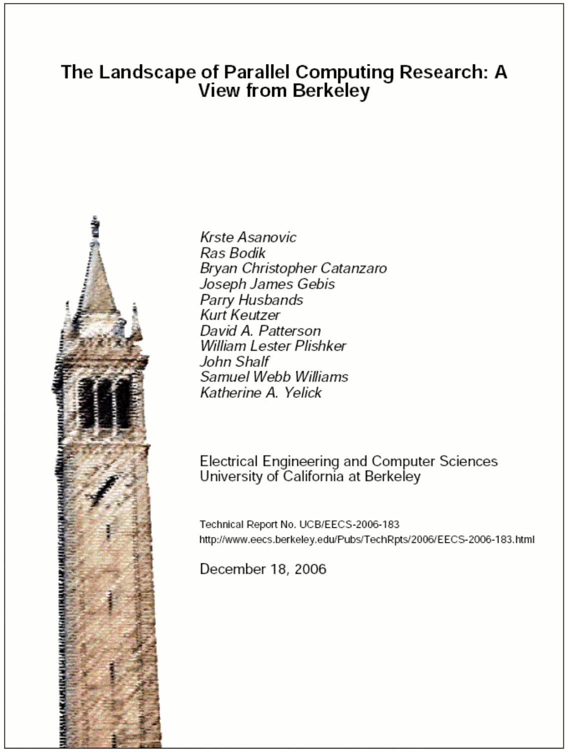
LDRD LABORATORY DIRECTED RESEARCH & DEVELOPMENT

ASC™

Sandia National Laboratories

Drivers for LWK Compute Node OS

- Practical advantages
  - Low OS noise
  - Performance – tuned for scalability
  - Determinism – inverted resource management
  - Reliability
- Research advantages
  - Small and simple
  - Freedom to innovate (see “Berkeley View”)
    - Multi-core
    - Virtualization
  - Focused on capability systems
  - Much simpler to create LWK than mainstream OS



The Landscape of Parallel Computing Research: A View from Berkeley

Krste Asanovic
Ras Bodik
Bryan Christopher Catanzaro
Joseph James Gebis
Parry Husbands
Kurt Keutzer
David A. Patterson
William Lester Plishker
John Shalf
Samuel Webb Williams
Katherine A. Yelick

Electrical Engineering and Computer Sciences
University of California at Berkeley

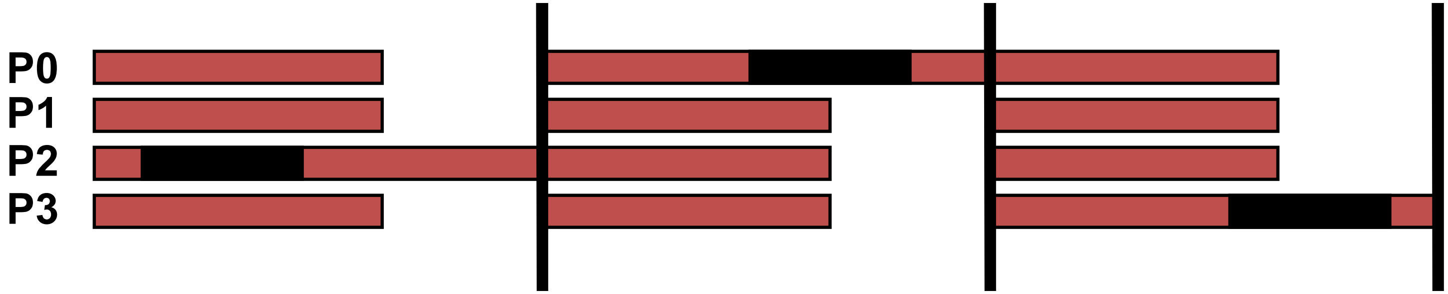
Technical Report No. UCB/EECS-2006-183
http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html

December 18, 2006

LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

We Know OS Noise Matters



- Impact of noise increases with scale (basic probability)
- Multi-core increases load on OS
- Idle noise measurements distort reality
  - Not asking OS to do anything
  - Micro-benchmark != real application

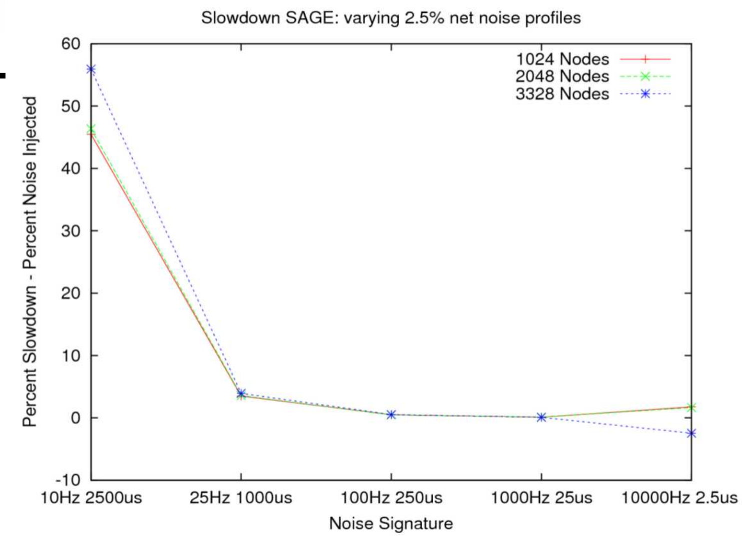
See “The Case of the Missing Supercomputer Performance”, Petrini, et al.

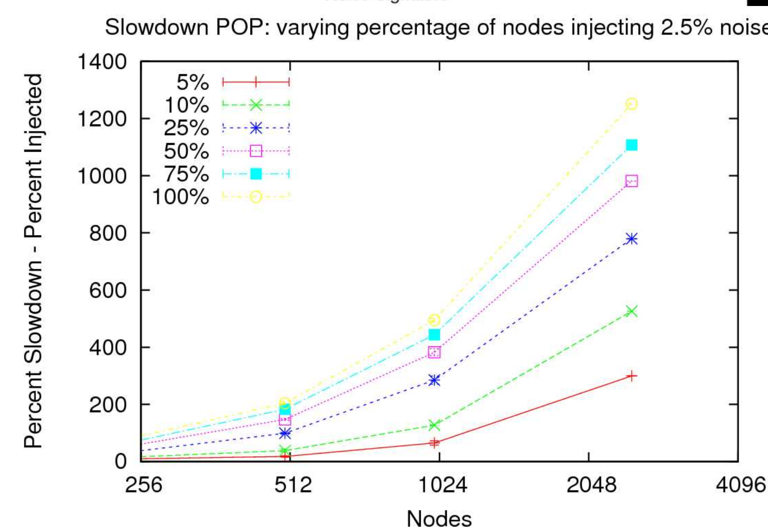
ASC™

Sandia National Laboratories

Impact of OS Noise on Applications

- Built a kernel-level noise injection framework in Catamount for synthetic noise injection
- Parameters for noise injection
  - Frequency and duation of noise
  - Set of participating nodes
  - Randomization method for noise patterns
- Results
  - Importance of how is noise injected (frequency vs duration)
  - Distribution of “noisy” nodes
  - Application characteristics likely amplify or absorb noise
- Continuing Work with INCITE
  - More applications at scale
  - Other noise sources: network and memory management

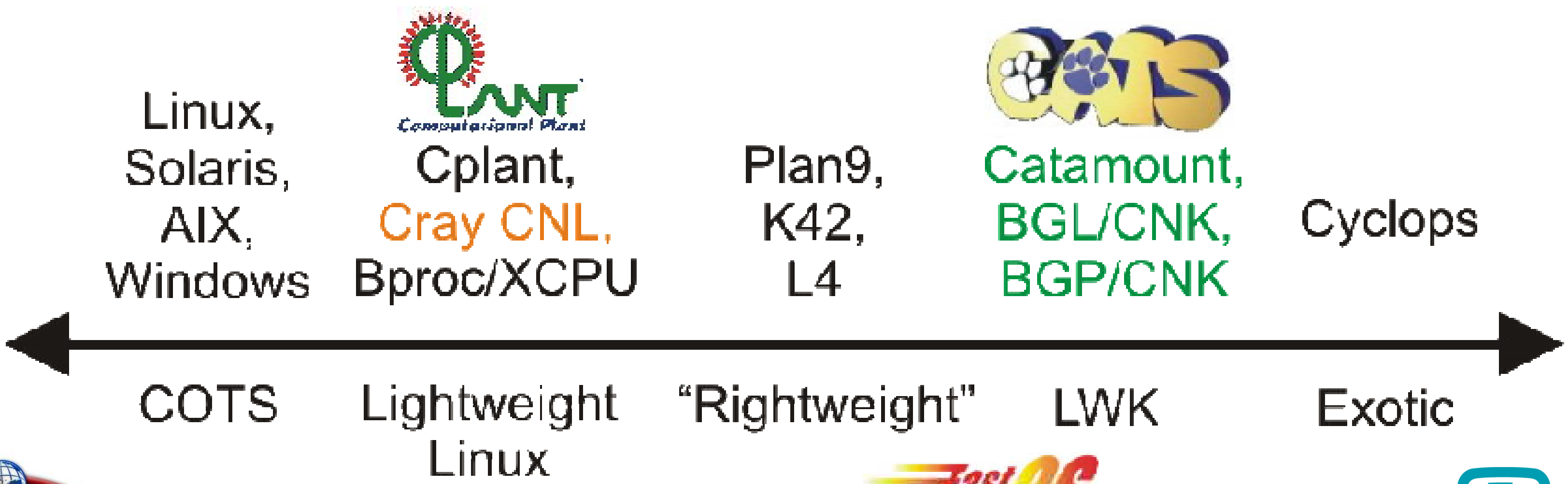




Project Kitten

Our Vehicle for OS Research

- **Creating modern open-source LWK platform**
  - Multi-core becoming MPP on a chip, requires innovation
  - Leverage hardware virtualization for flexibility
- **Retain scalability and determinism of Catamount**
- **Better match user and vendor expectations**
- **Available from *http://software.sandia.gov/trac/kitten***



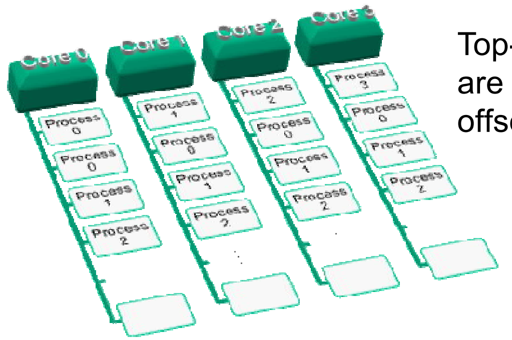
LDRD
LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

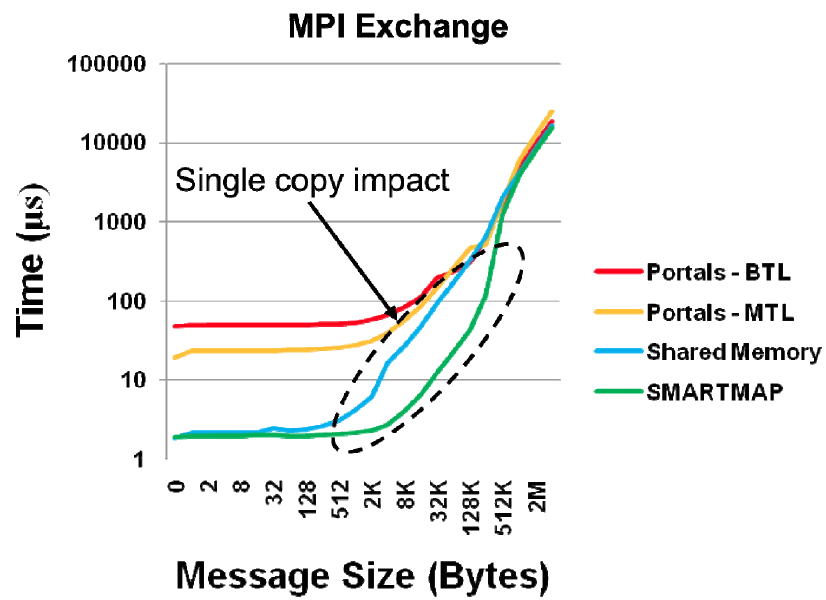
Kitten Supports SMARTMAP

Simple Mapping of Address Region Tables for Multi-core Aware Programming

- Direct access to shared memory
  - Access to “remote” data by flipping bits in the virtual address
- Each process still has a separate virtual address space
  - Everything is “private” and everything is “shared”
  - Processes can be threads
- Allows MPI to eliminate all extraneous memory-to-memory copies
  - Single-copy MPI messages
  - No extra copying for non-contiguous datatypes
  - In-place and threaded collective operations
- Not just for MPI
  - Emulate POSIX shared memory regions
  - One-sided PGAS operations
  - Can be used by applications directly
  - Leverages lightweight kernel page table layout



Top-level page table slots are used to create a fixed-offset virtual address space

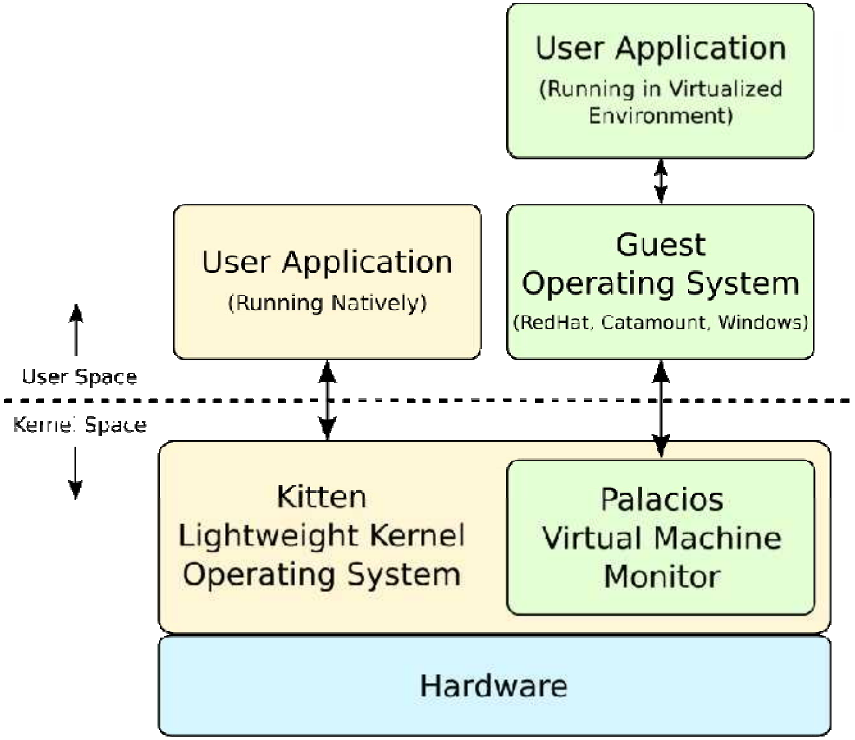


LDRD LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

Kitten and Palacios for Virtualization

- Palacios is a VMM from Northwestern
- For end-user flexibility
  - Provide full functionality OS functionality
  - Run commodity Oses
  - Dynamic selection of compute-node OS
  - Convenient packaging
- For research
  - X-Stack development and large-scale test
  - Add capabilities to guest OS without modifying it
  - VM migration/resilience
  - Instrumentation and debugging

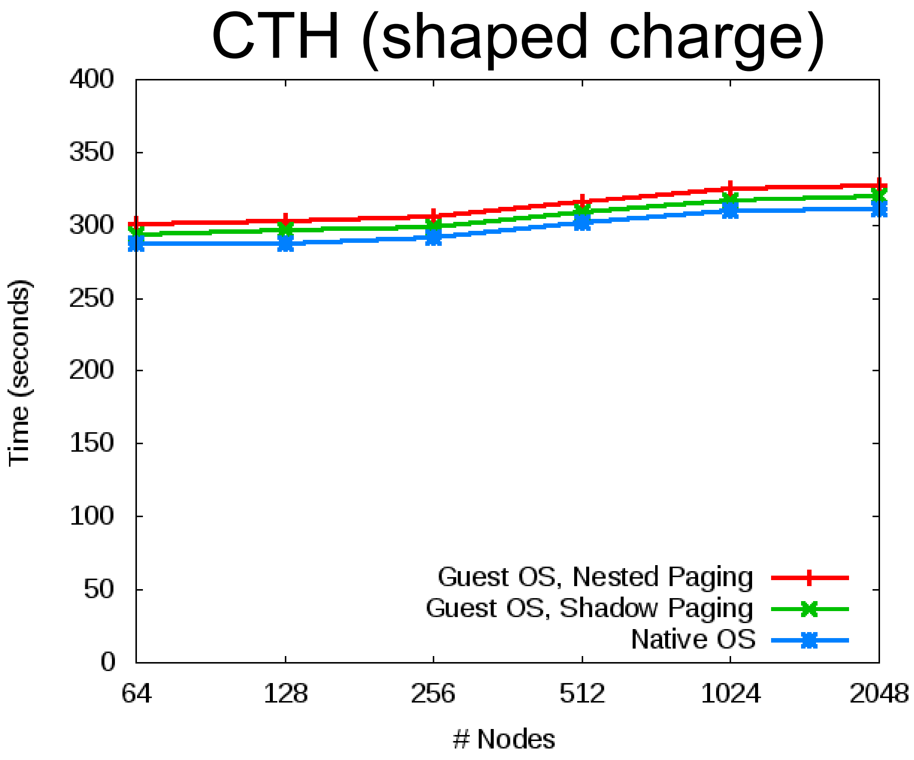


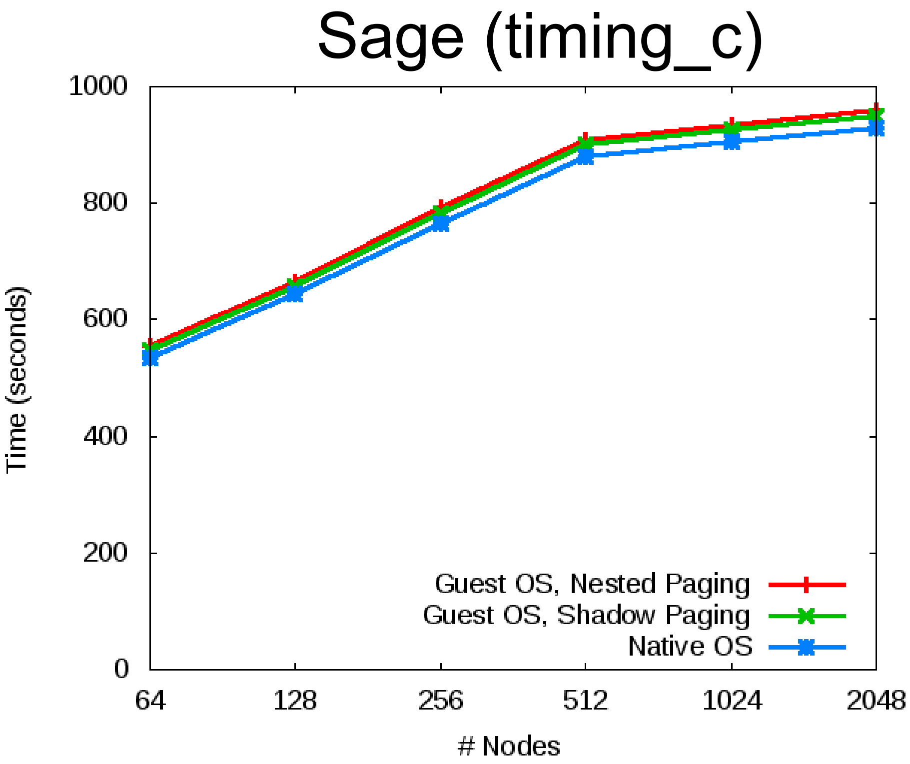
Kitten homepage: https://software.sandia.gov/trac/kitten
Palacios homepage: http://www.v3vee.org/palacios/

LDRD LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

Large-scale Virtualization Experiments on Red Storm





< 5% virtualization overhead for all cases tested

LDRD
LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

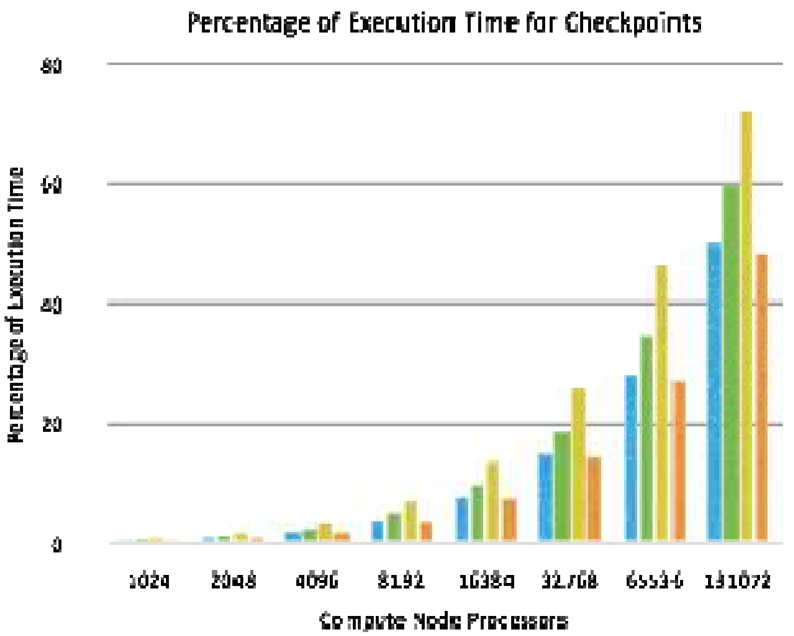
Resilience!?... Its an I/O Problem

Most of our I/O is for resilience

- Application-directed checkpoints are the primary protection against faults
- Frequency of checkpoint is based on probability of failure
- Probability of failure is based on application size.

Our resilience efforts reduce I/O

- System-influence on how/when to chkpt
- **Viability of incremental checkpoints,**
- Diskless checkpoints,
- **Partial-redundant computation**



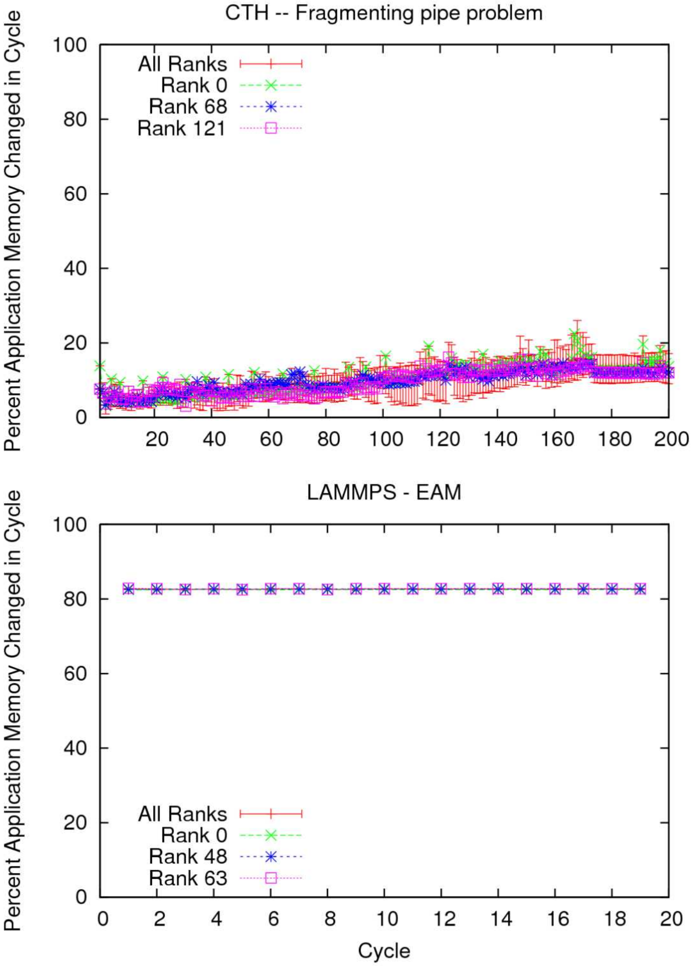
Oldfield et al. Modeling the impact of checkpoints on next-generations systems. In *Proceedings of the 24th IEEE MSST*, Sept. 2007

LDRD LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

The Case For/Against Incremental Checkpointing

- Lightweight lib to identify modified memory
  - Page-table trickery identifies modified pages
  - Crypto-hash (MD5) identifies modified blocks
  - No app changes required; user/system specifies interval to collect memory statistic
  - User & kernel-space version for Catamount. CNL user-space version in testing.
- Results
  - Runtime overhead < 10%
  - CTH: modified memory within 8% of app
  - LAMMPS: modified memory 4x larger than checkpoint

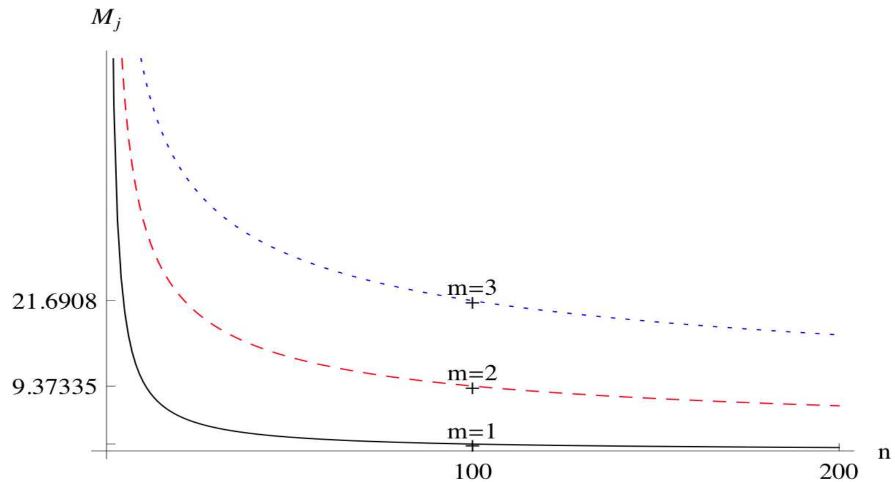


LDRD
LABORATORY DIRECTED RESEARCH & DEVELOPMENT

Sandia National Laboratories

Exploring Redundant Computation

- Motivation
  - Overhead of checkpoint unacceptable
  - Increase MTTI
  - Reduce defensive I/O
  - Hypothesis: at large scale, overhead of redundant computation is less than checkpoint/restart
- rMPI library
  - Between application and MPI
  - Replicates ranks 0..n
  - Checkpoint still required (just not as often)
  - rMPI almost a full MPI implementation
    - MPI_Wtime, MPI_Probe, … need to return same answer for both nodes
    - Message order and other MPI semantics must be preserved



Scalable I/O Services

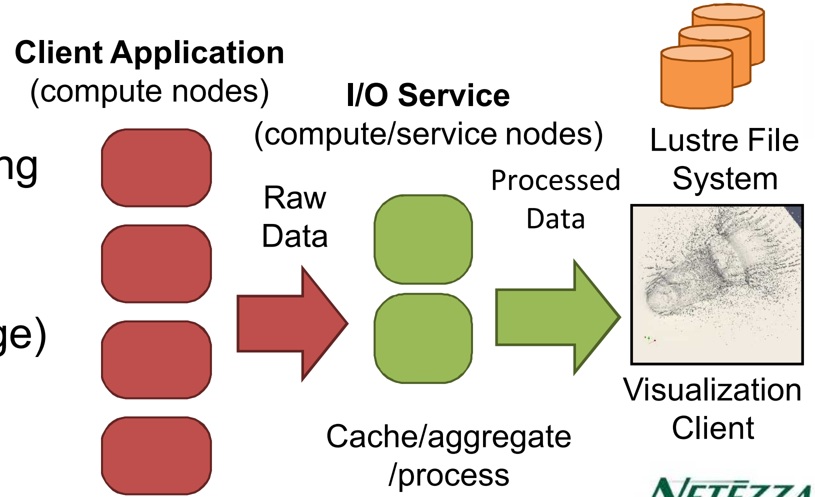
Even our I/O research is about reducing I/O

Purpose

- Leverage available compute/service node resources for I/O caching and data processing

Application-Level I/O Services

- Lightweight File System (authr, authn, storage)
- PnetCDF caching service
- SQL Proxy (for NGC)
- Sparse-matrix visualization (for NGC) CTH Particle tracking



NETEZZA

LexisNexis

INCITE Plans

- PnetCDF caching
- Investigate placement issues
- ADIOS I/O services for fusion, climate, combustion apps on Jaguar

Sandia National Laboratories

Scalable I/O Services

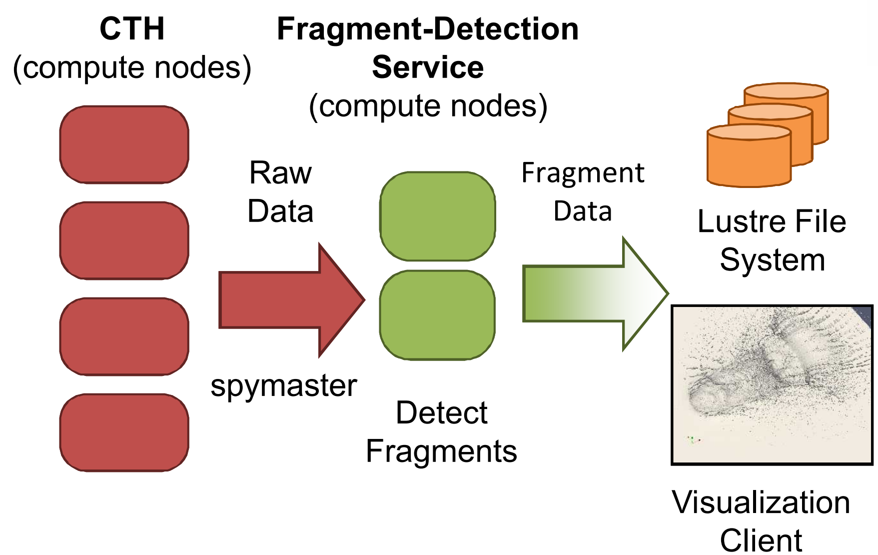
CTH Fragment Detection

Motivation

- Fragment detection requires data from every time step (I/O intensive)
- Detection process takes 30% of time-step calculation (scaling issues)
- Integrating detection software with CTH is intrusive on developer

CTH fragment detection service

- Extra compute nodes provide in-line processing (overlap fragment detection with time step calculation)
- Only output fragments to storage (reduce I/O)
- Non-intrusive
  - Looks like normal I/O (spymaster interface)
  - Can be configured out-of-band



Fragment detection service provides on-the-fly data analysis with no modifications to CTH.

Status

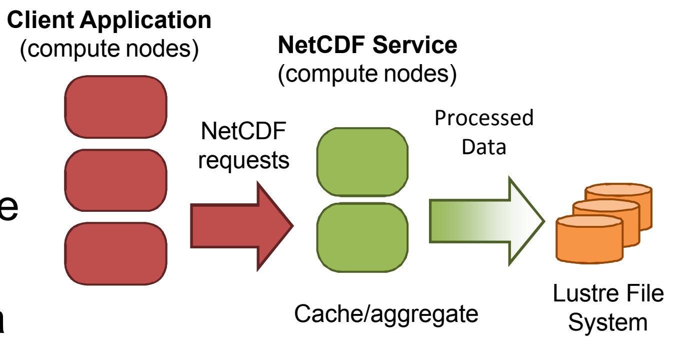
- Developing client/server stubs for spymaster
- Developing Paraview proxy service

Scalable I/O Services

NetCDF I/O Cache

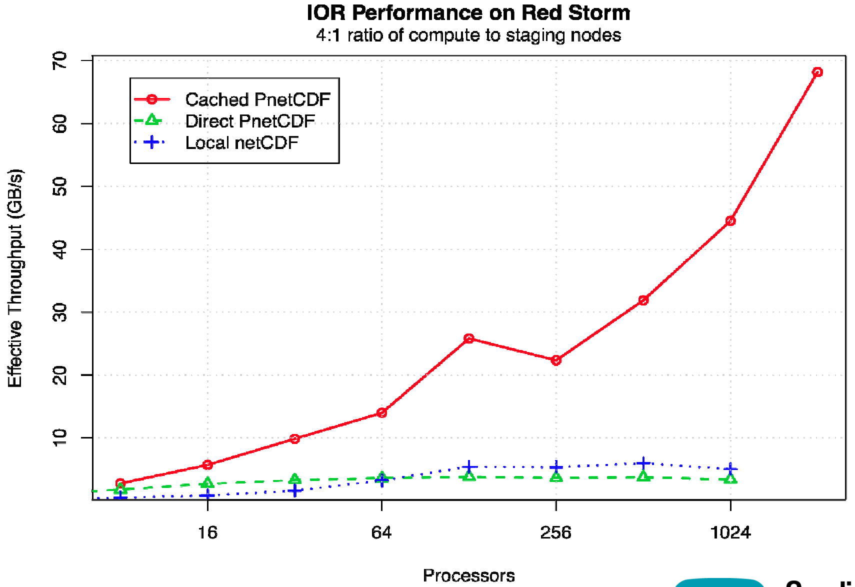
Motivation

- Synchronous I/O libraries require app to wait until data is on storage device
- Not enough cache on compute nodes to handle "**I/O bursts**"
- NetCDF is basis of important I/O libs at Sandia (Exodus)



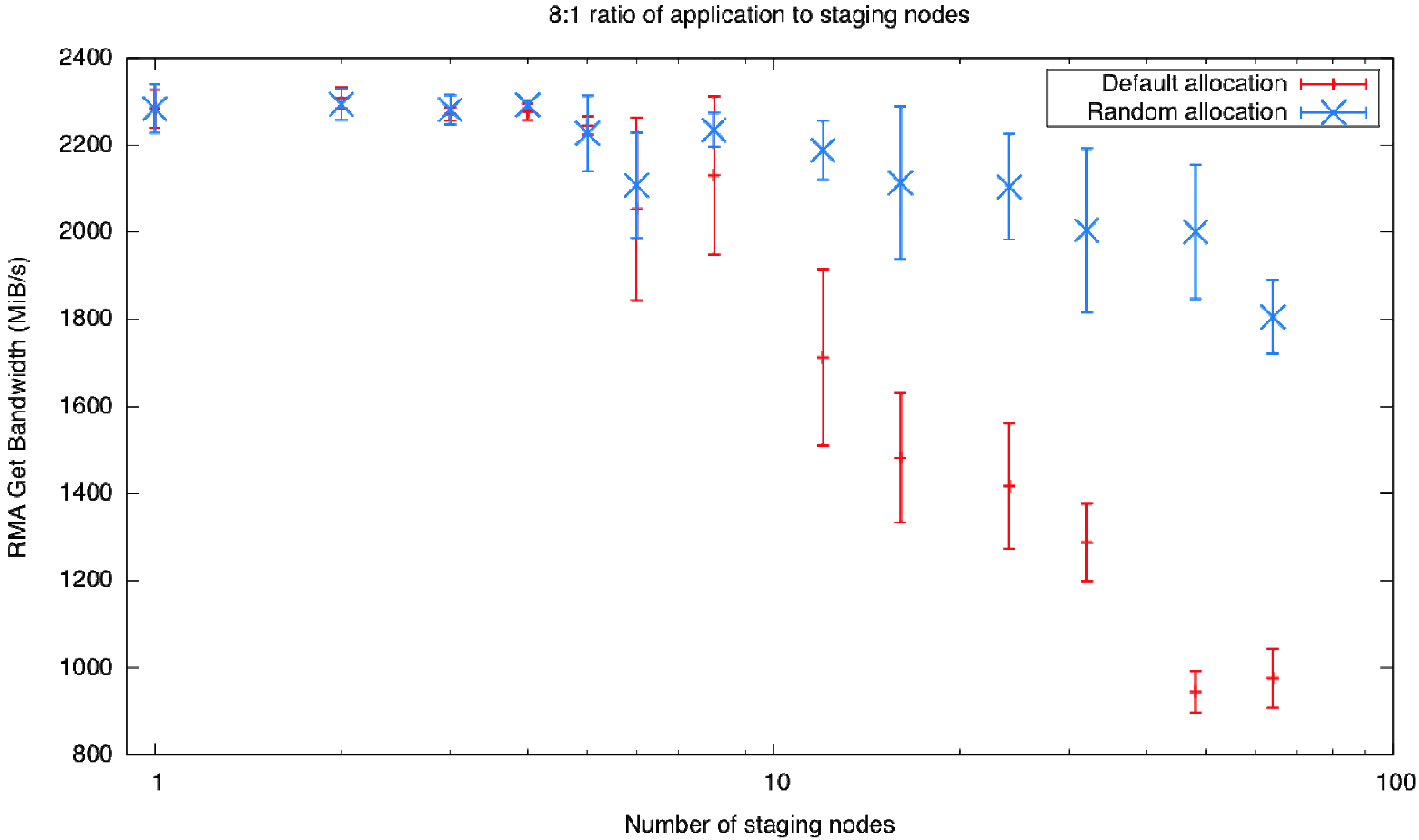
NetCDF Caching Service

- Service aggregates/caches data and pushes data to storage
- Async I/O allows overlap of I/O and computation



Sandia National Laboratories

Placement Issues for I/O Services



Sandia National Laboratories

Application Power and Frequency Analysis

Motivation

- Power is one of or the most important considerations in fielding current and next generation HPC systems.
- HPC application power use and factors impacting this use are not well studied.
- Power saving techniques used in commodity operating systems will greatly impact HPC application performance.

Modifications to RAS and Catamount to support power savings

- RAS
  - Added instrumentation and collection capabilities to RAS
- Catamount
  - Power savings during OS idle, per core
  - OS-level frequency scaling capability
  - User space library interface to frequency scaling
  - MPI profiling layer instrumentation

Sandia National Laboratories

Power Frequency and Analysis

Phase 1

Based on previous power analysis studies

– Laros et.al. *"Topics on Measuring Real Power Usage on High Performance Computing Platforms"*

Analyze performance vs. power efficiency (at scale)

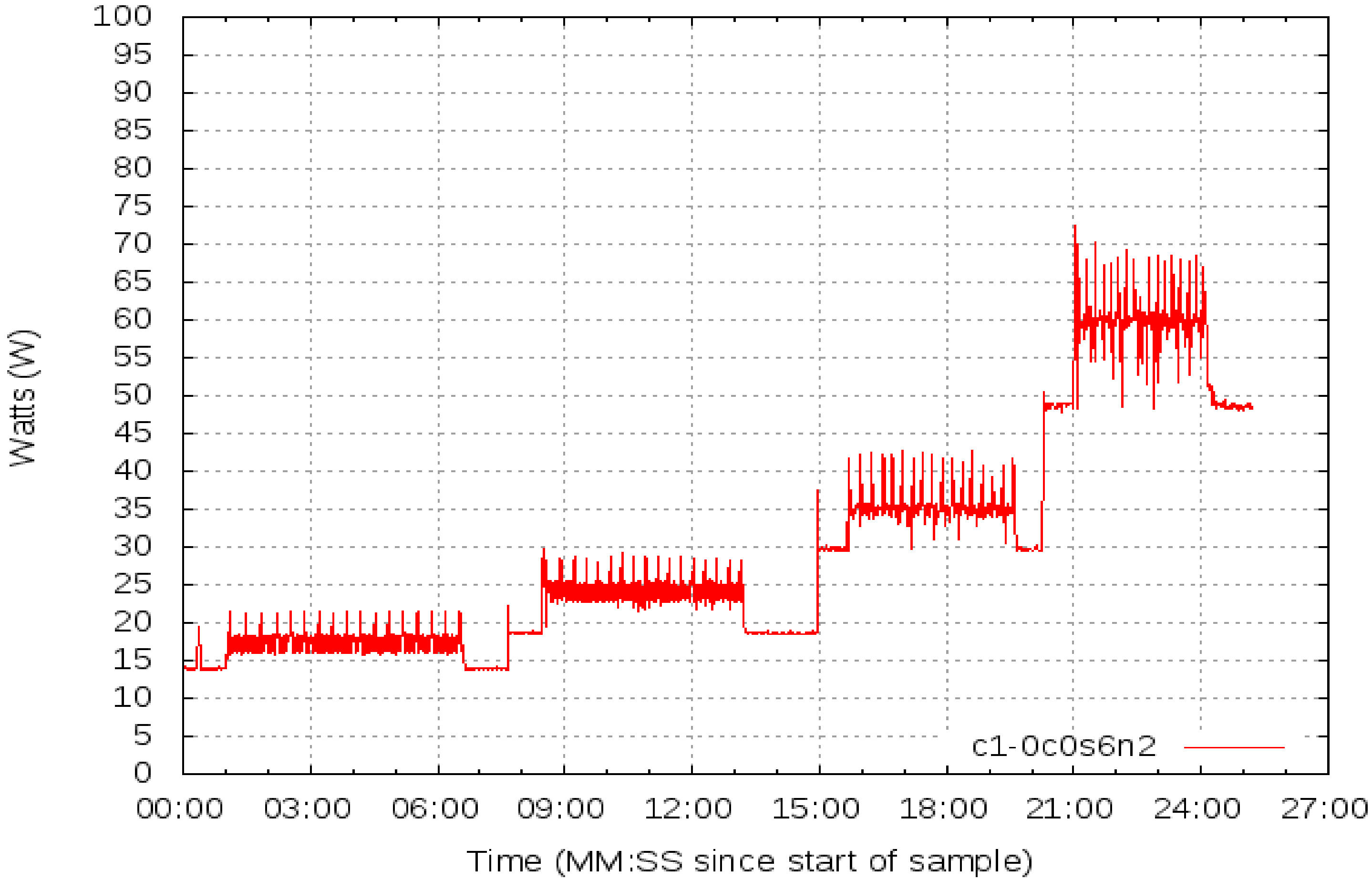
– **STATIC** frequency modification during application run-time.

– Procedure

- Execute application suite using a range of Pstates defining both frequency and input voltage of CPU.
- Collect power usage during runs and analyze total energy use vs. application run-time

Power Frequency Analysis: LAAMPs

Small scale results of multiple LAAMPs runs



Power Frequency and Analysis

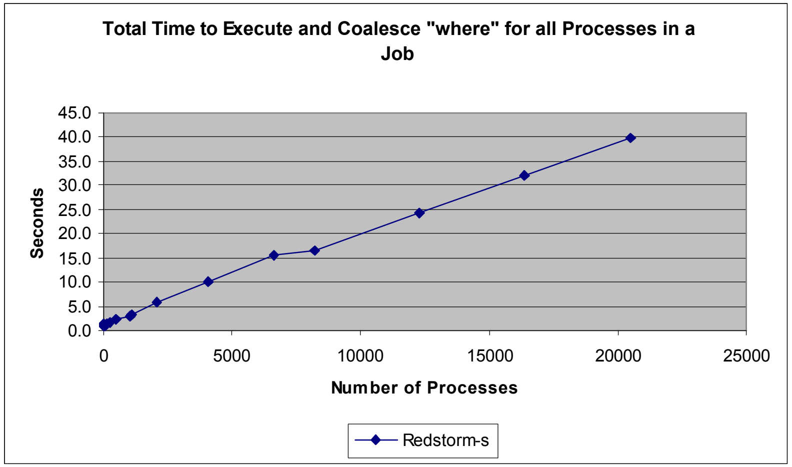
Phase 2

- Analyze performance vs. power efficiency (at scale)
  - **DYNAMIC** frequency modification during application run-time
  - DYNAMIC frequency modification defined as deterministic frequency change driven by application characteristics. Pstate change during MPI barrier for example
- Phase 3 testing, if necessary, will be based on Phase 1 and 2 analysis
- Additionally, power data will be collected during a range of other systems software testing accomplished as part of this overall project

Sandia National Laboratories

Debugging: Fast_where

- A simple utility
  - Command line interface
  - 375 SLOC written in /bin/sh; runs at user level
  - No system software changes; no special daemons
- A "fast" implementation of the "where" function found in traditional debuggers
  - Implicitly verifies health of all nodes in job
- Originally written to address specific operational needs on Red Storm
- Summarizes the results by active function
  - which processes (e.g. ranks) are in each function
  - how many processes are in each function
- Can also request the full stack trace for individual (or range of) processes
- Syntax:
  - `fast_where [-b batch_id] program_exe`
  - other optional arguments are system-specific and are needed if more than one parallel application is running within one batch job



Related Research

- **Stack Trace Analysis Tool (STAT)**
  - **LLNL and U-Wisc collaboration**
- **Multicast Reduction Network (MRNet)**
  - **middleware and network protocol for scalable tools**
  - **used by STAT**
- **Totalview, gdb and other debuggers**

Sandia National Laboratories

Debugging: Research Questions

- What advantages can LWKs offer debugging tools?
- At 20K cores (5000 nodes), fast_where response of 40 seconds is acceptable to the interactive user. What/where is the breaking point for this simple algorithm?
- What additional features or scalability algorithms can be added without compromising the tool’s simple architecture?

Summary and Requests

INCITE provides necessary platforms for scalable system software research

Some of our codes require dedicated access

- OS research (Kitten, Catamount, Noise)
- Power efficiency (need access to RAS)

We need applications from open science community

- For memory characterization, resilience, OS research, I/O research
- Some of our apps are export controlled
- Have GTC, XGC, POP, AMG, LAMMPS,
- Would like to see and Open Science Benchmark Suite

Thanks

- Ron Brightwell, Kevin Pedretti, Rolf Riesen, Jim Laros, Kurt Ferreira, Todd Kordenbrock, Sue Kelly
- ***Don Maxwell*** (for helping set up our dedicated testing)

Sandia National Laboratories

Our First Day

Sandia National Laboratories