What is stopping us from getting to exascale computing and what should we do about it?

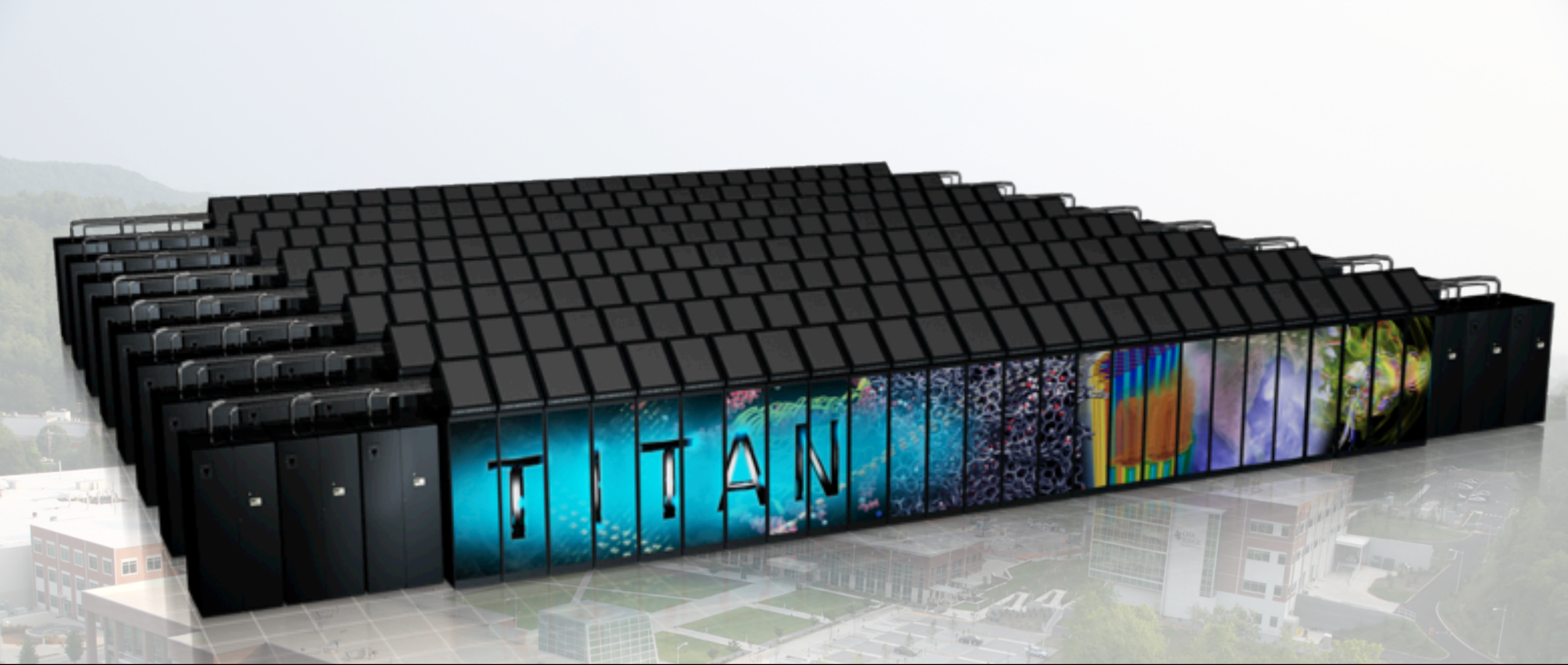
Devesh Tiwari
Northeastern University
tiwari@northeastern.edu

Invited Talk at HPC Day 2017
at UMass Dartmouth


WHAT YOU SEE HERE
WHAT YOU DO HERE
WHAT YOU HEAR HERE
WHEN YOU LEAVE HERE
LET IT STAY HERE


OAK RIDGE

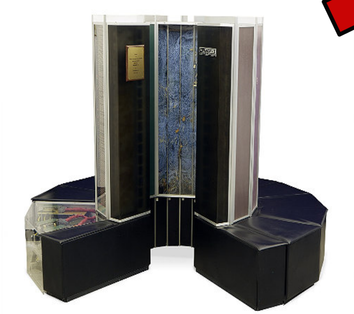

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT BATTELLE
FOR U.S. DEPARTMENT OF ENERGY


TITAN


OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT BATTELLE
FOR U.S. DEPARTMENT OF ENERGY

Change


EXIT 26
AMBRIDGE - SOMERVILLE
STORROW DRIVE
EXIT 25
NO. STATION
GOV'T. CTR.
CHARLESTOWN



1970s
Cray-1
160Mflops

1990s
ASCI Red
1-3Tflops

2010s
Titan
18Pflops

But…

HOME COMPUTE STORE CONNECT CONTROL CODE ANALYZE HPC ENTERPRISE

EXASCALE TIMELINE PUSHED TO 2023: WHAT'S MISSING IN SUPERCOMPUTING?

April 27, 2016 Nicole Hemsoth

The roadmap to build and deploy an exascale computer has extended over the last few years–and more than once.

Ini
cap
CO
lab
acc
rec
20
the efforts toward exascale computing
key challenges ahead. While the petafl
fastest systems are yielding tremendo

EXASCALE COMPUTING PROJECT

HPC wire

Since 1987 - Covering the Fastest Computers in the World and the People Who Run Them

- Home
- Technologies

US Moves Exascale Goalpost, Targets 2021 Delivery

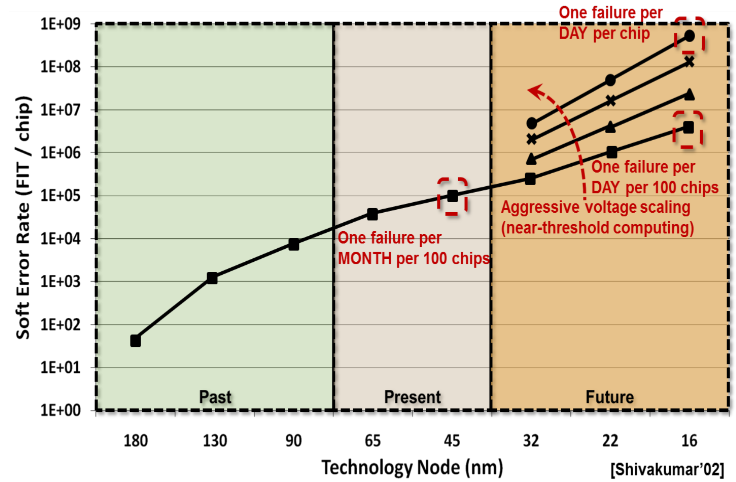
By Tiffany Trader

December 12, 2016

During SC16, Exascale Computing Project Director Paul Messina hinted at an accelerated timeline for reaching exascale in the US and now we have official confirmation from Dr. Messina that the US is contracting its exascale timeline by one year.

Under the updated plan, the US will still be fielding at least two exascale machines in the next seven years. One of those machines remains on the original timeline – targeting delivery for 2022 and acceptance in 2023. However, the other machine is now on track for delivery in 2021 and acceptance in 2022. Further the intention of the DOE is that that first machine will employ a novel architecture.

System Reliability Challenge for Exascale

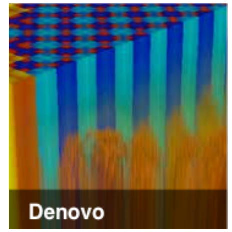


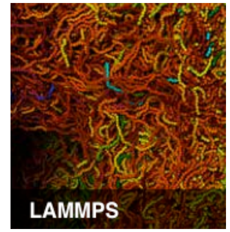
Large-scale scientific applications are going to face severe resilience challenge at exascale!

- "Top Ten Exascale Research Challenges", DOE ASCAC Subcommittee Report, Feb. 2014

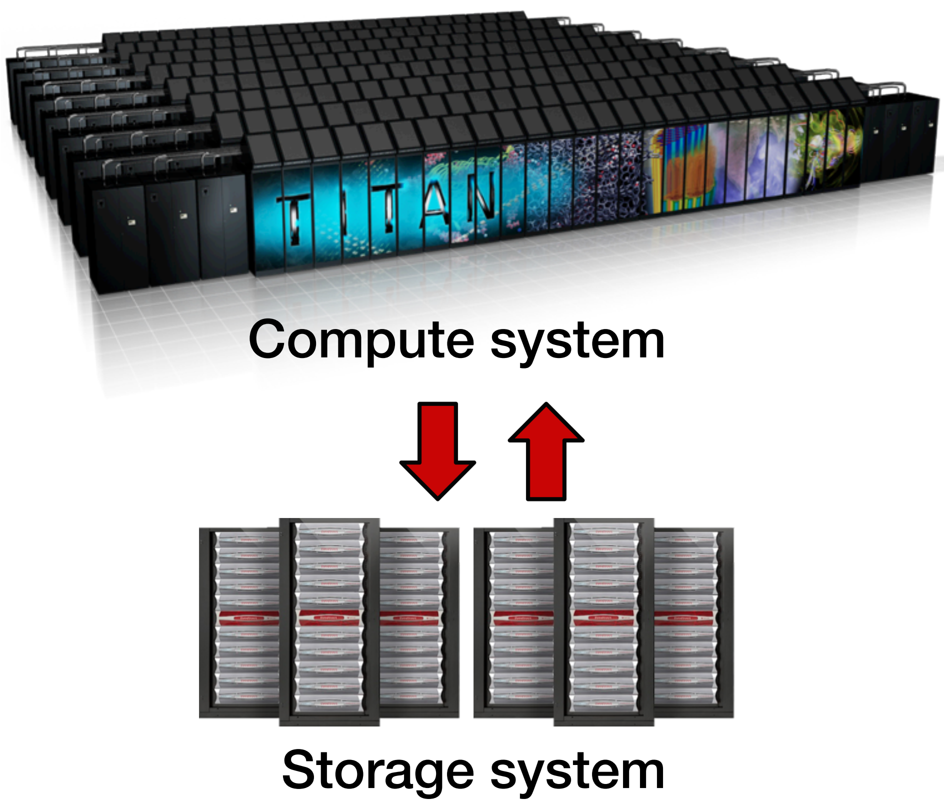
Long-running, large-scale scientific applications are interrupted by failures on HPC systems.

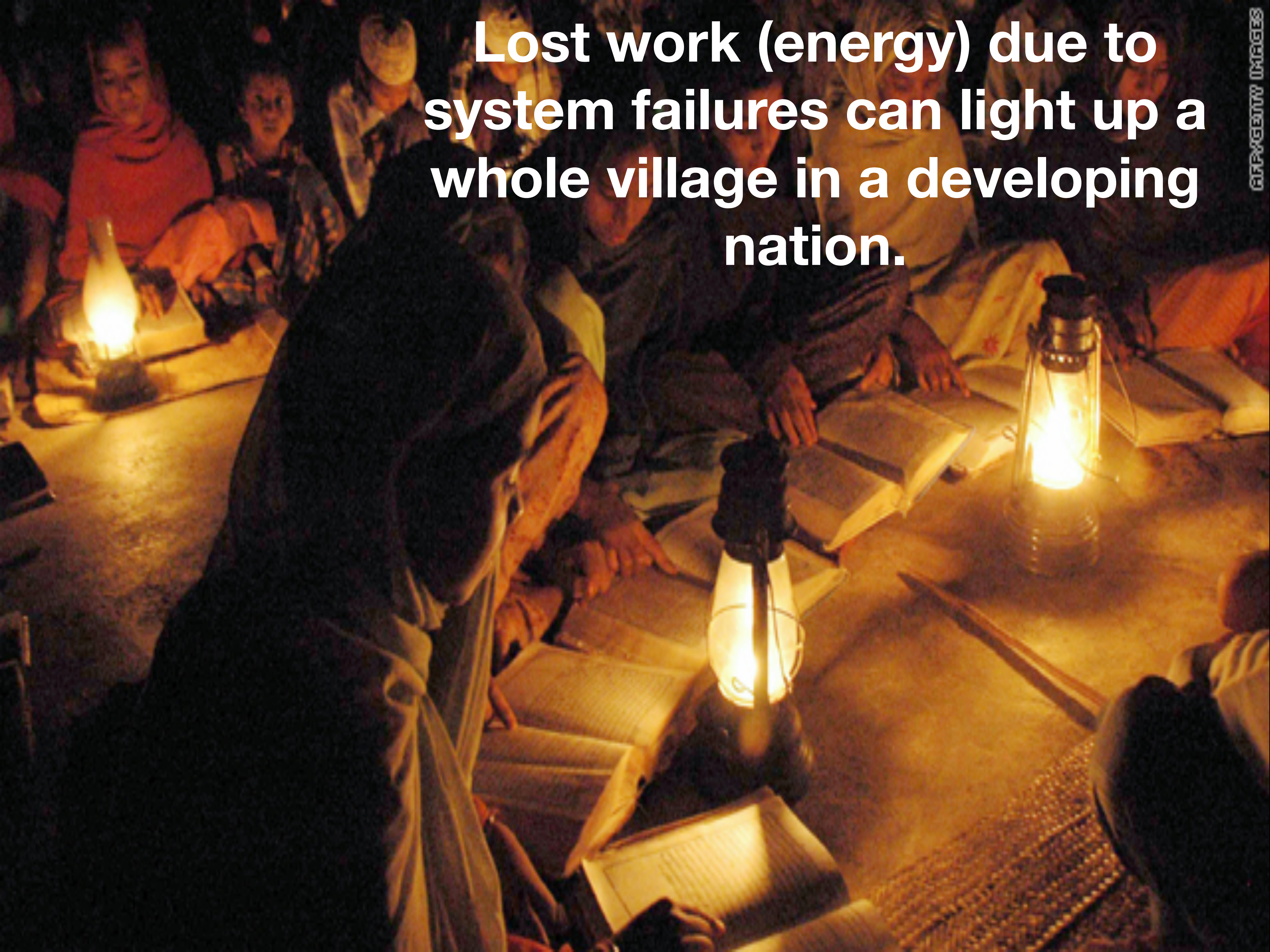
At exascale, an application is expected to be interrupted every couple of hours.





Astrophysics, climate modeling, combustion and fusion applications periodically write checkpoints to permanent storage system, and recover from the last checkpoint in case of a failure.

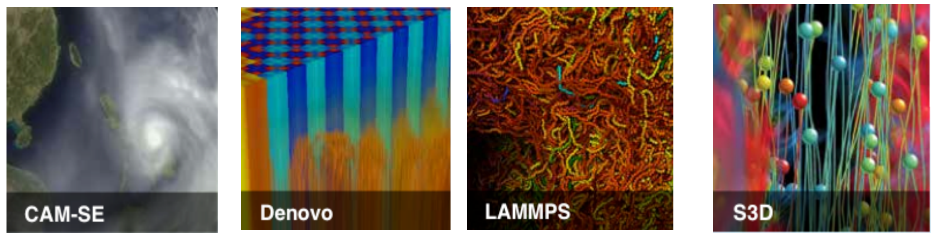



Lost work (energy) due to system failures can light up a whole village in a developing nation.
AFP/GETTY IMAGES

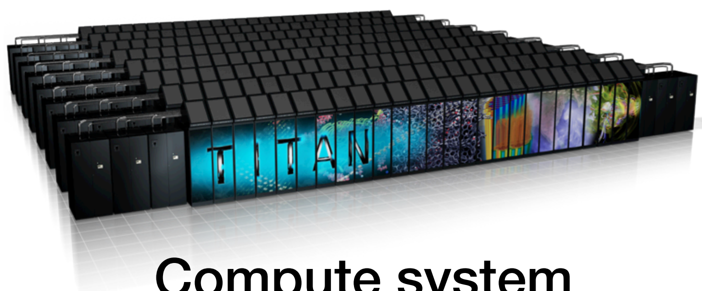

High operating
cost (and pain)

Energy-Efficiency and Data Movement Challenge at Exascale

Data Production For Reliability



Already excessive I/O overhead due to checkpoint/restart

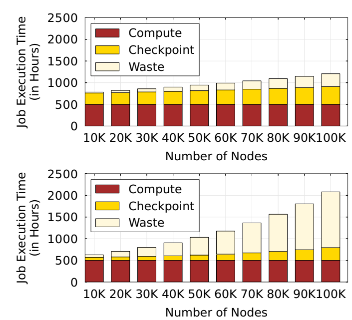


Compute system

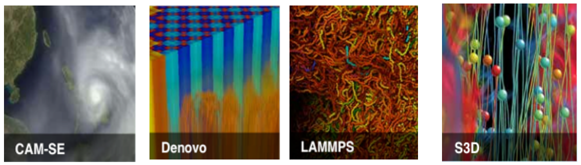
At exascale, applications may spend up to 60% of execution time in checkpoint/restart

Domain	Application	Checkpoint data size
Astrophysics	CHIMERA	160 TB
Astrophysics	VULCUN/2D	0.83 GB
Climate	POP	26 GB
Combustion	S3D	5 TB
Fusion	GTC	20 TB
Fusion	GYRO	50 GB

Storage system

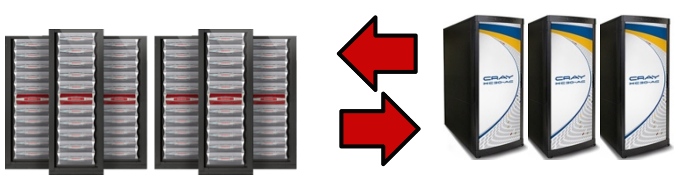


Data Production For Post Analysis



Data-intensive applications

Compute system

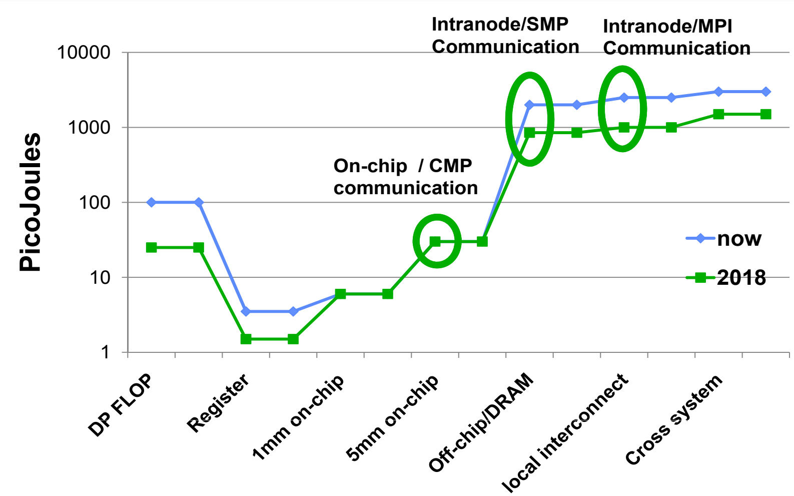


Storage system

Analysis cluster

GTC (a plasma physics application) produces 30TB data per hour at-scale on the Titan supercomputer

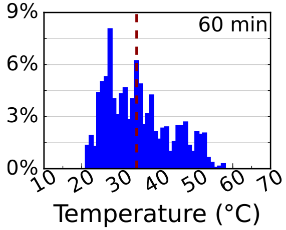
On the Mira supercomputer, few applications in the material science domain spend 30%-85% of execution time in I/O at-scale.

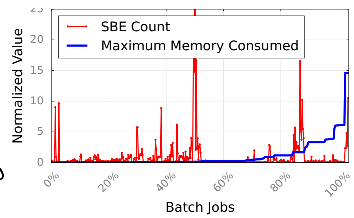


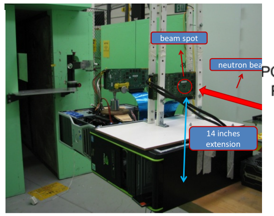
Source: Kogge et el., ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems, 2008

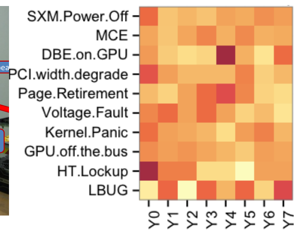
Brief Overview of My Research Addressing These Challenges

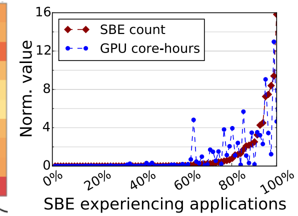
Improving operational efficiency and reliability of data-center scale systems





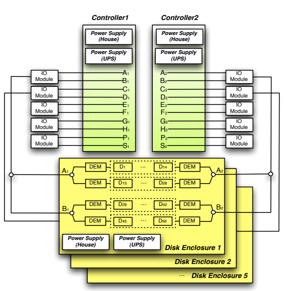


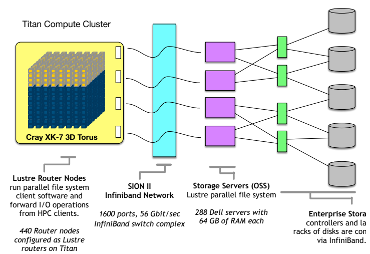


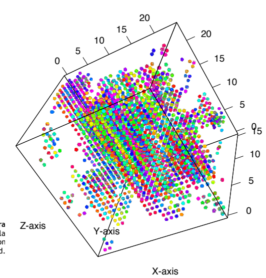


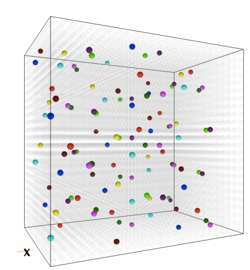
HPCA 16, DSN 15, HPCA 15, SC 15 (a), DSN 14, LCTES 15, IPDPS 16

Large-scale data storage systems provisioning, reliability, and performance



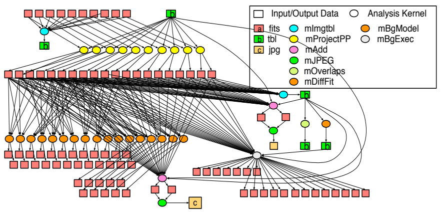


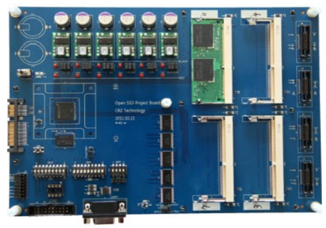


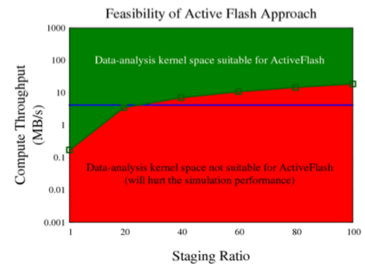


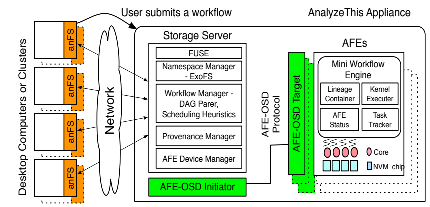
SC 15 (b), SC 14, ICPADS 14, CUG 14

Energy-efficient in-situ data analysis on SSDs for extreme-scale machines



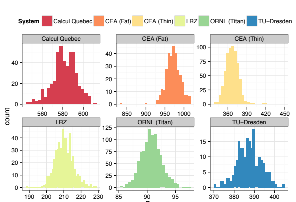


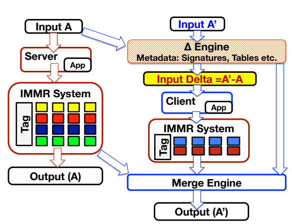


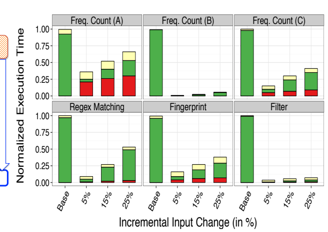


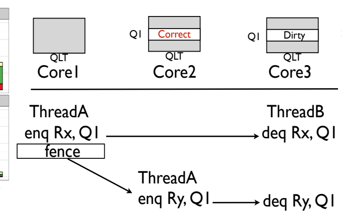
SC 15 (c), USENIX FAST 13, USENIX HotPower 12

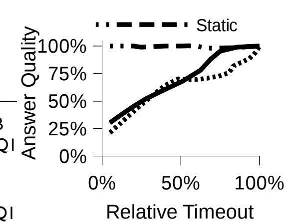
Analytical models and methods for better performance and managing answer quality of data analytics workloads







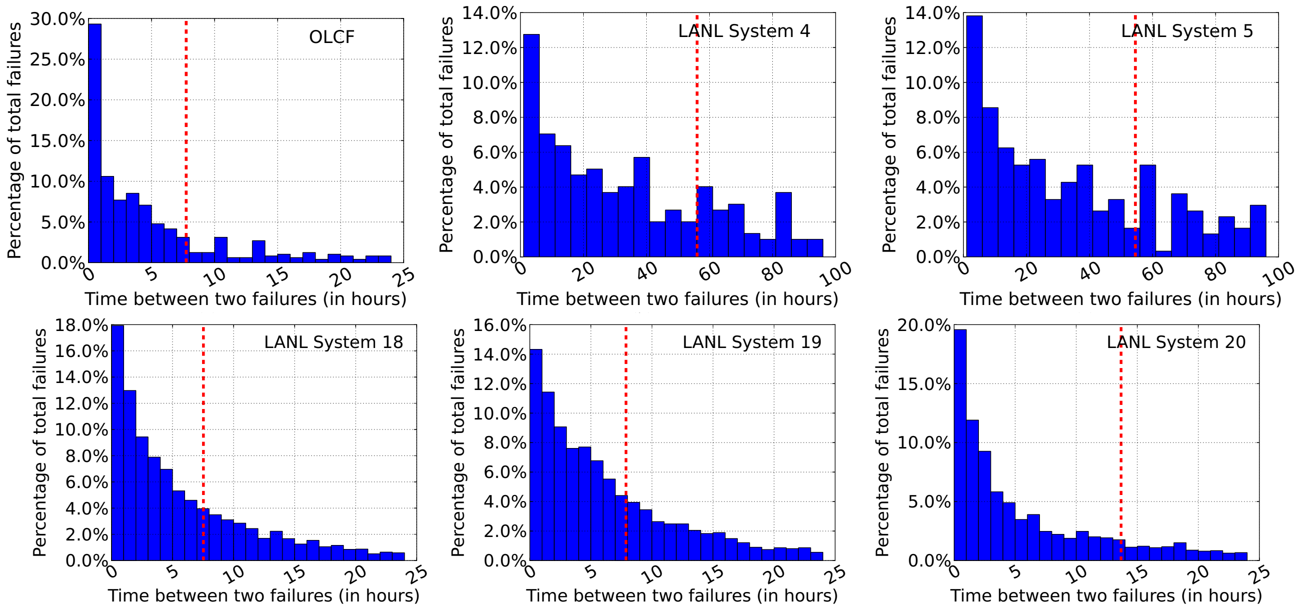




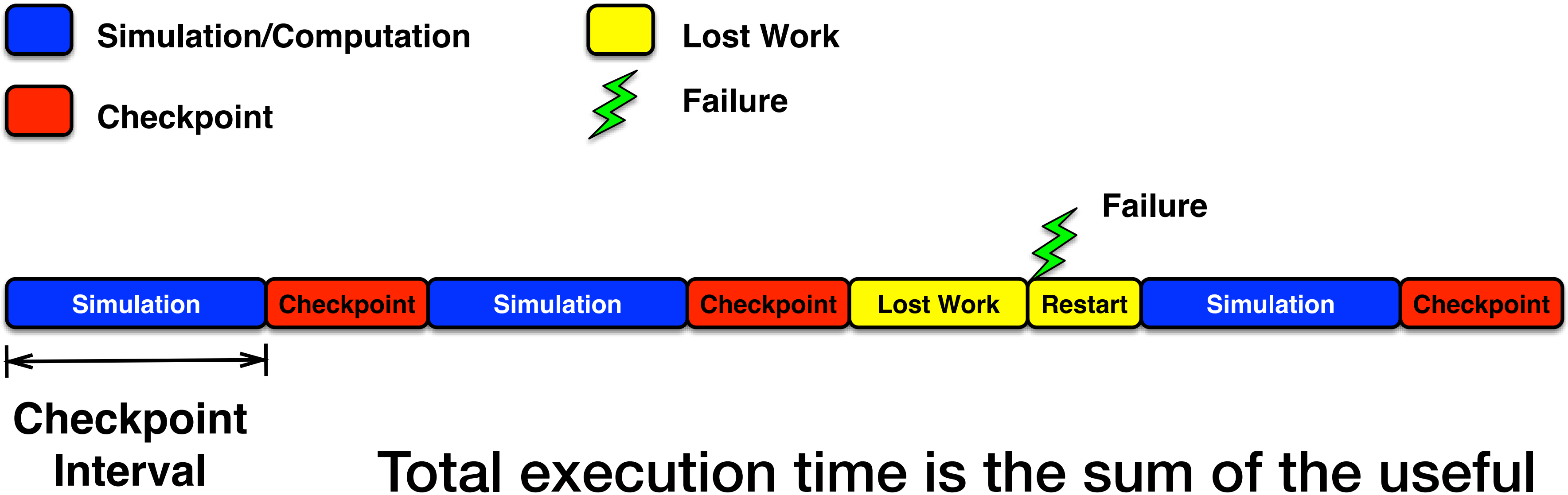
HPCA 11, IPDPS 14 & 12 & 10, ICAC 15, SC 15 (d)

Exploiting Temporal Locality in System Failures for Mitigating I/O Overhead

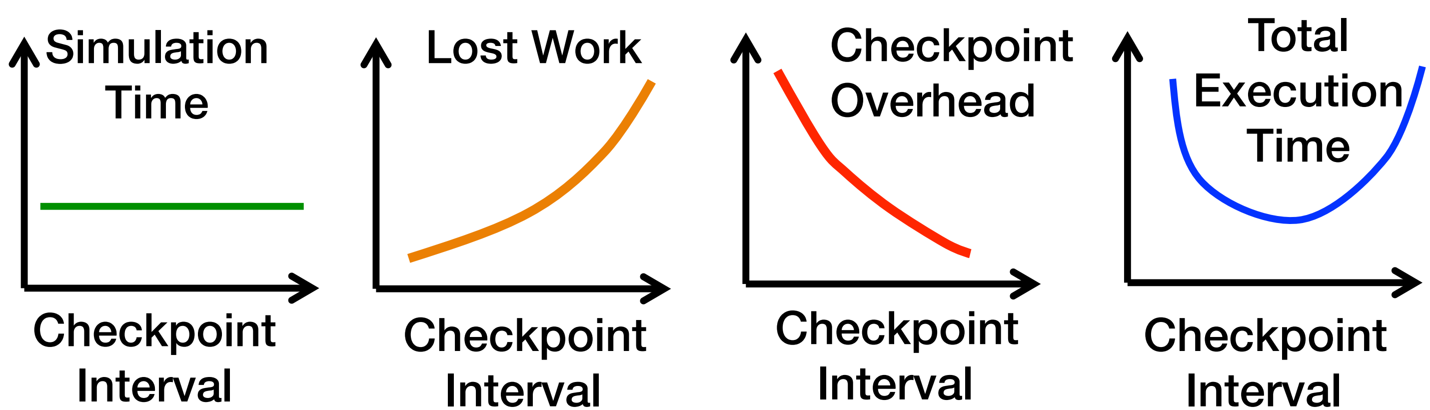
Lazy Checkpointing



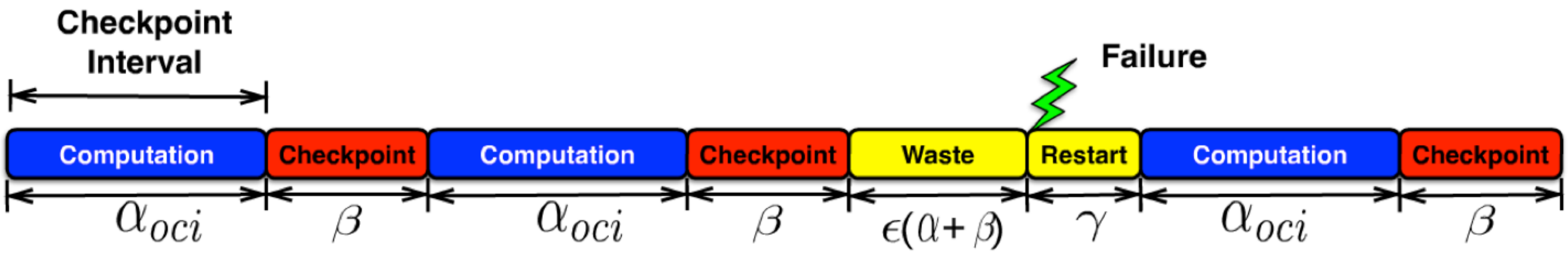
A large fraction of failures occur much before the MTBF for many HPC systems.



Total execution time is the sum of the useful computation/simulation time, the checkpointing overhead and the lost work



Optimal Checkpointing Interval (OCI)



$$T_{total} = T_{compute} + T_{checkpoint} + T_{waste}$$

$$T_{compute} = S\alpha$$

$$T_{checkpoint} = (S-1)\beta$$
$$= \left(\frac{T_{compute}}{\alpha} - 1\right)\beta$$

$$T_{waste} = N_f(\epsilon(\alpha+\beta)+\gamma)$$

$$T_{total} = T_{compute} + \left(\frac{T_{compute}}{\alpha} - 1\right)\beta + \frac{T_{compute}}{\alpha}\left(e^{\frac{\alpha+\beta}{M}} - 1\right)(\epsilon(\alpha+\beta)+\gamma)$$

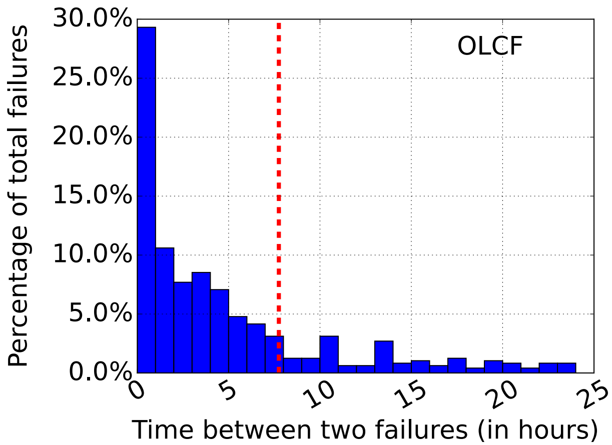
$$\frac{1}{M}\left(\epsilon - \frac{\epsilon\beta^2}{\alpha_{oci}^2} - \frac{\epsilon\gamma}{\alpha_{oci}^2}\right) - \frac{\beta}{\alpha_{oci}^2} = 0$$

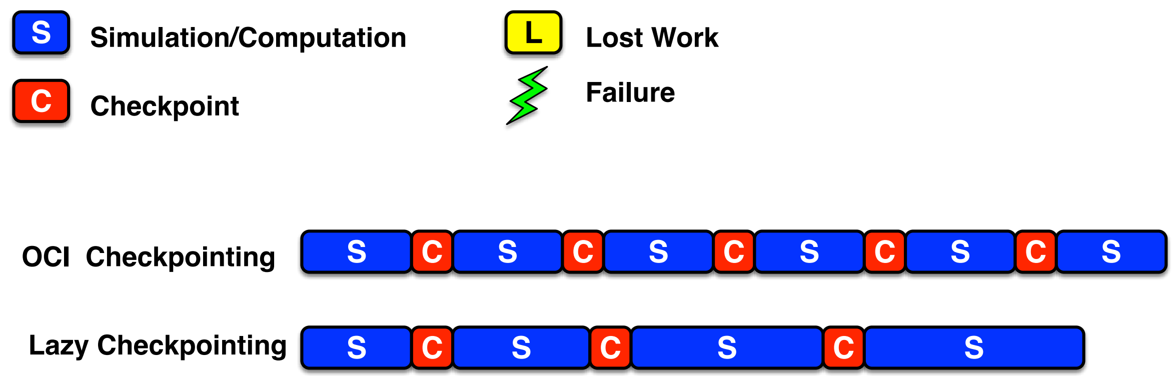
α_{oci}	**Optimal Checkpoint Interval (OCI)**
β	**Time-to-checkpoint**
ϵ	**Average lost work fraction**
γ	**Restart overhead**
M	**Mean Time Between Failure (MTBF)**

$$\alpha_{oci} = \sqrt{\beta^2 + \frac{\beta\gamma}{\epsilon} + \frac{M\beta}{\epsilon}}$$

Assuming exponential distribution of system failures

Lazy Checkpointing
Basic Idea and Intuition

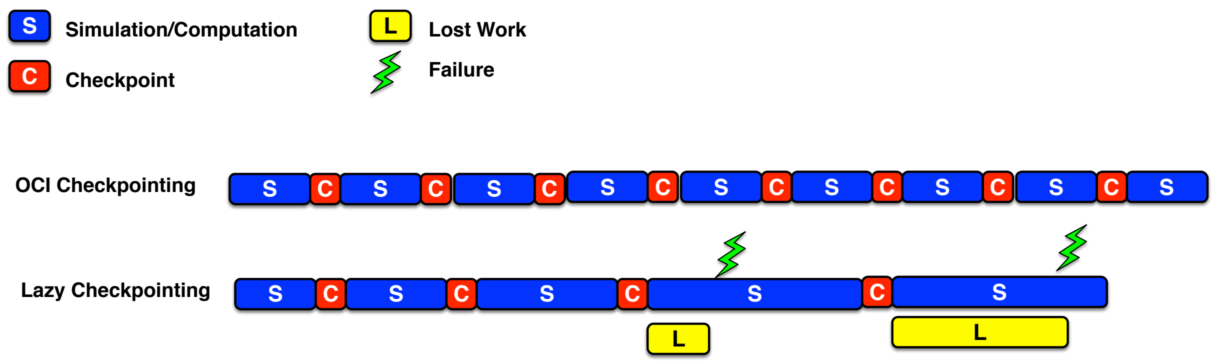


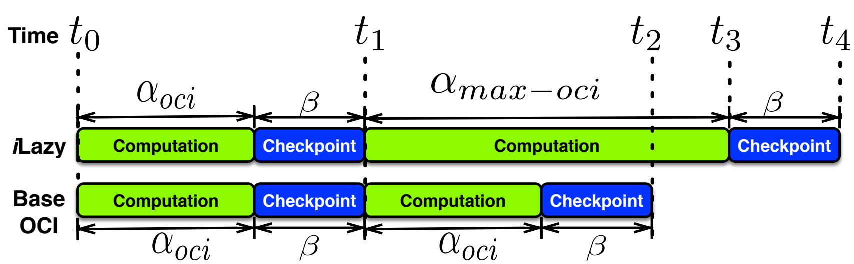


Temporal locality in failures can be exploited by becoming “increasingly” lazy in taking checkpoints.

"Bounding" the Checkpointing Interval

Key is to balance the trade-off between reduction in checkpointing overhead and possible increase in the waste work





$$\text{performance gain} = \beta e^{-(\frac{t_3}{\lambda})^k}$$

$$\text{performance loss} = (\alpha_{max-oci} - \alpha_{oci})(e^{-(\frac{t_2}{\lambda})^k} - e^{-(\frac{t_4}{\lambda})^k})$$

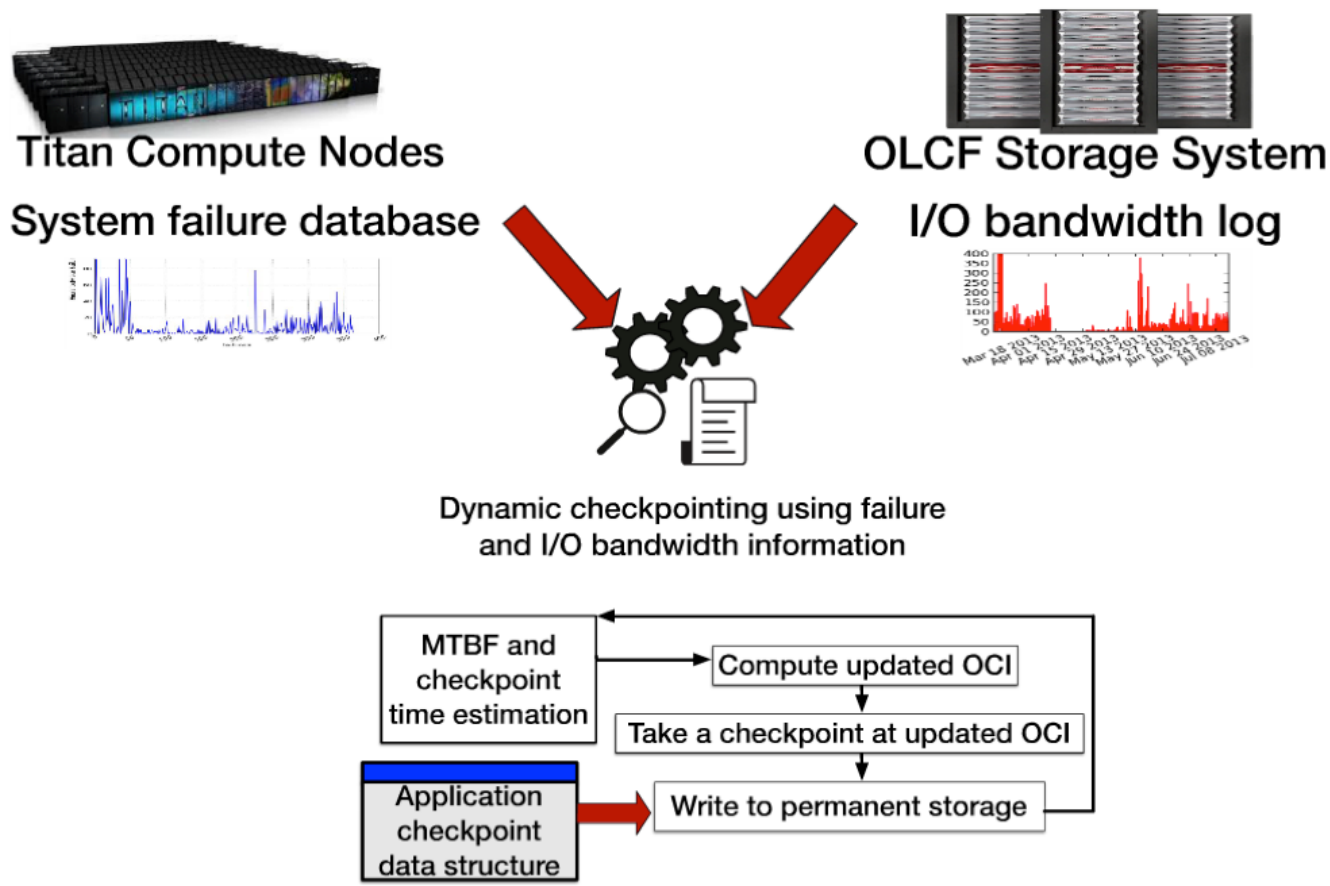
$$= (\alpha_{max-oci} - \alpha_{oci})(e^{-(\frac{2(\alpha_{oci}+\beta)}{\lambda})^k} - e^{-(\frac{\alpha_{max-oci}+\alpha_{oci}+2\beta}{\lambda})^k})$$

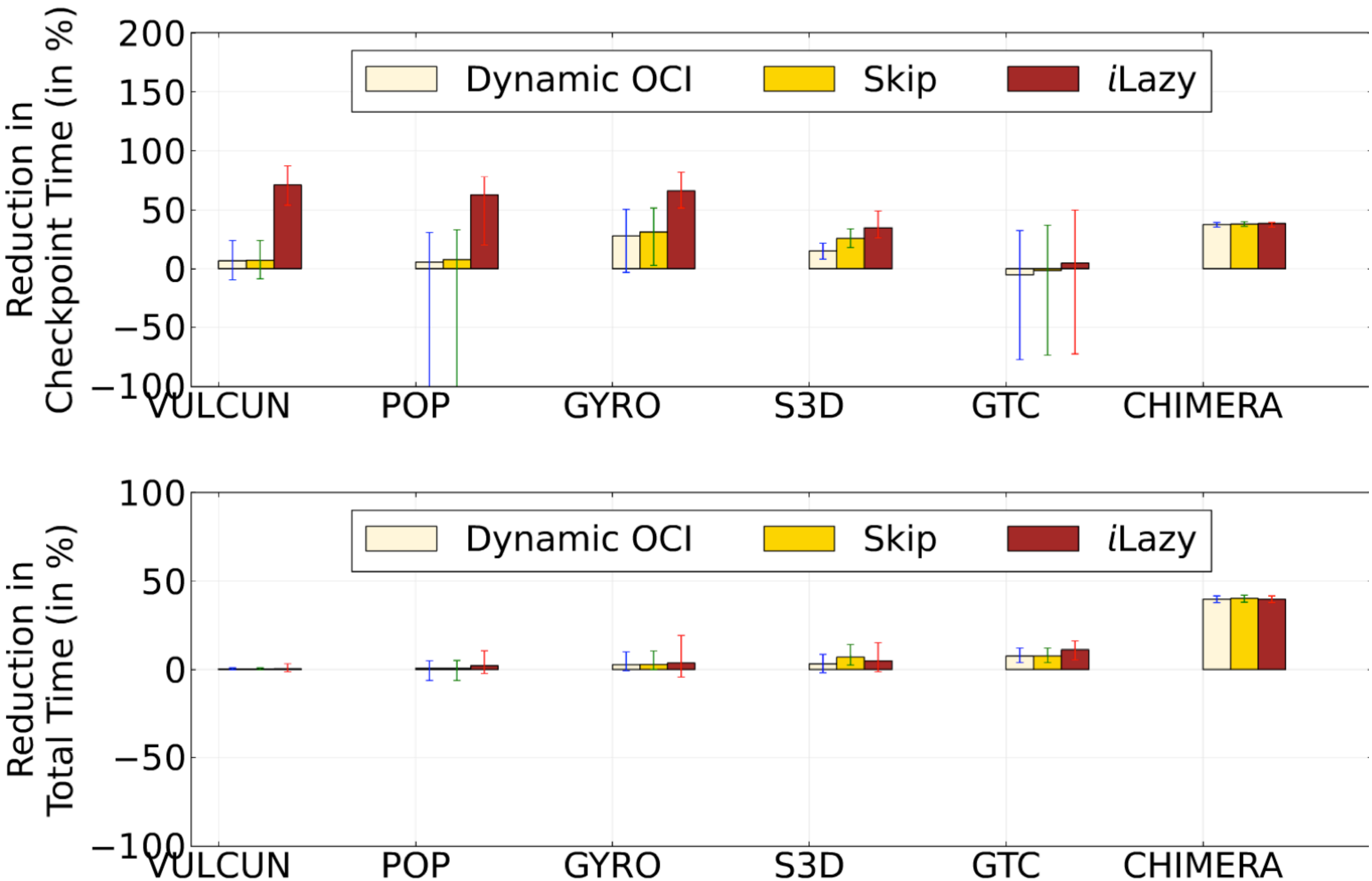
$$\beta e^{-(\frac{\alpha_{max-oci}+\alpha_{oci}+\beta}{\lambda})^k} = (\alpha_{max-oci} - \alpha_{oci}) e^{-(\frac{2(\alpha_{oci}+\beta)}{\lambda})^k} - (\alpha_{max-oci} - \alpha_{oci}) e^{-(\frac{\alpha_{max-oci}+\alpha_{oci}+2\beta}{\lambda})^k}$$

Devesh Tiwari, Saurabh Gupta, Sudharshan Vazhkudai, "Lazy Checkpointing: Exploiting Temporal Locality in Failures to Mitigate Checkpointing Overheads on Extreme-Scale Systems", Proceedings of the 44th Annual IEEE/IFIP Int'l Conference on Dependable Systems and Networks (DSN), 2014.

Lazy Checkpointing Prototype

Evaluated using real failure and I/O bandwidth logs



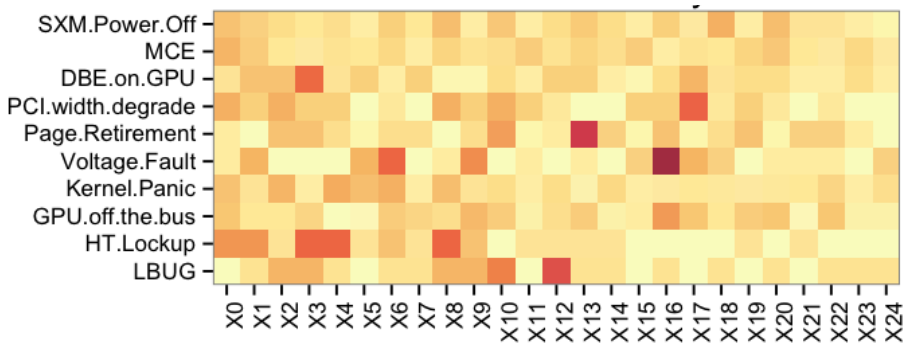


Reduction in
Checkpoint Time (in %)
200
150
100
50
0
−50
−100
Dynamic OCI
Skip
iLazy
VULCUN
POP
GYRO
S3D
GTC
CHIMERA
Reduction in
Total Time (in %)
100
50
0
−50
−100
Dynamic OCI
Skip
iLazy
VULCUN
POP
GYRO
S3D
GTC
CHIMERA

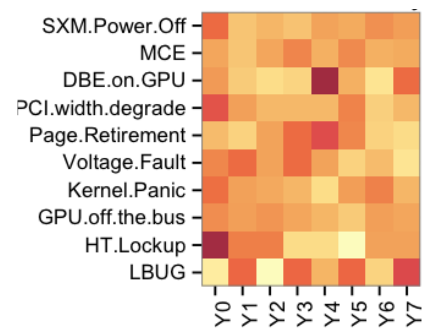
Exploiting Spatial Locality for Improved Reliability

Quarantine Technique

Failure type

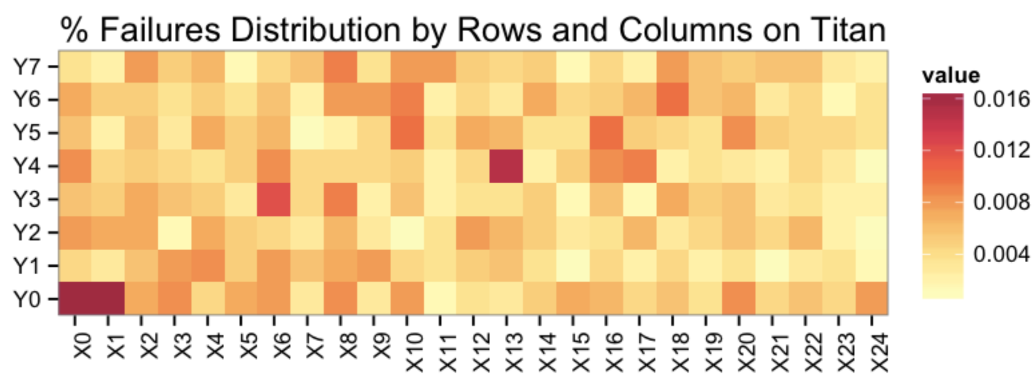


Cabinet columns

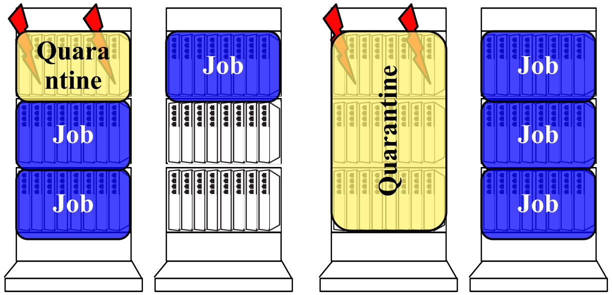


Cabinet rows

Cabinet rows

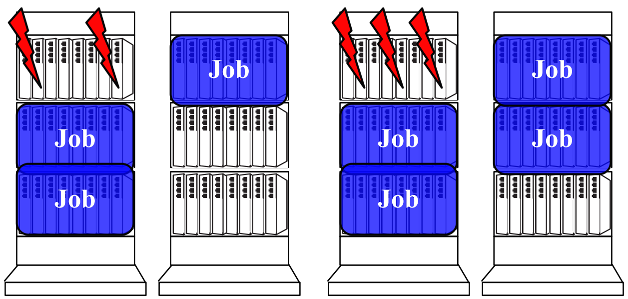


Quarantine: Design Challenges



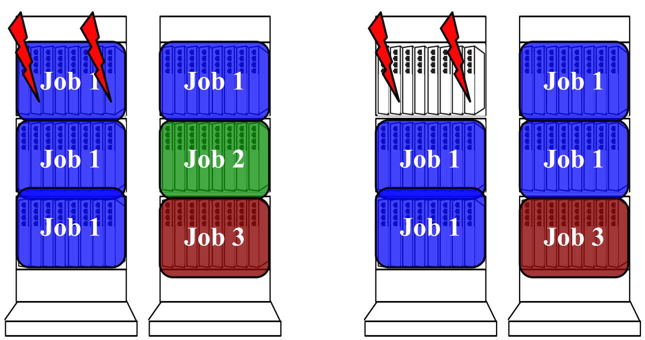
Quarantine Granularity

Fraction of avoided system failures versus compute resource waste



Quarantine Time Duration

Diminishing returns on the number of avoided failures

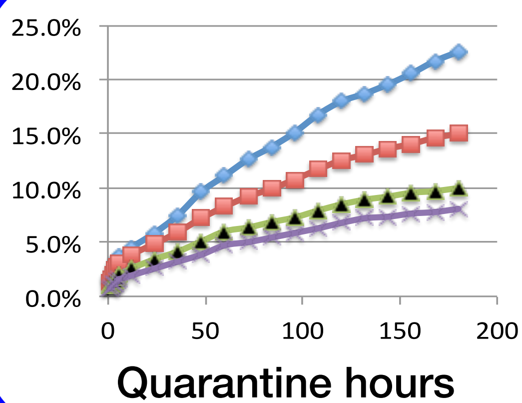


System Utilization vs. Reliability

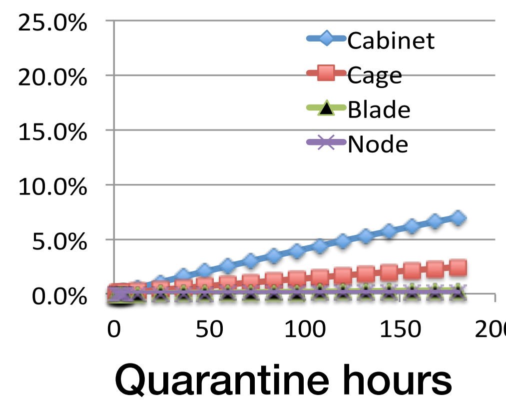
Trading-off lower system utilization for improved reliability

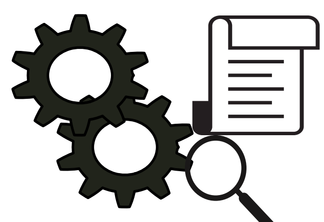
Quarantine Technique: In Action

System Reliability
Fraction of failures avoided



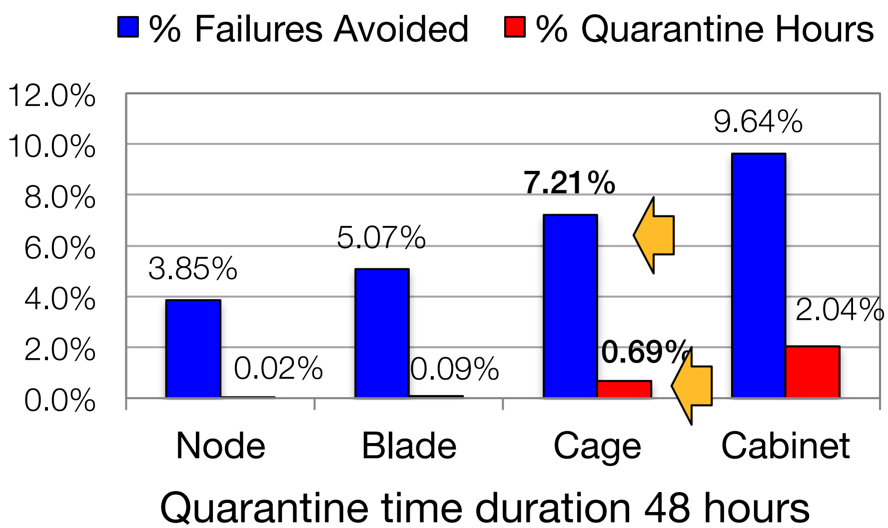
System Utilization
Quarantine node hours

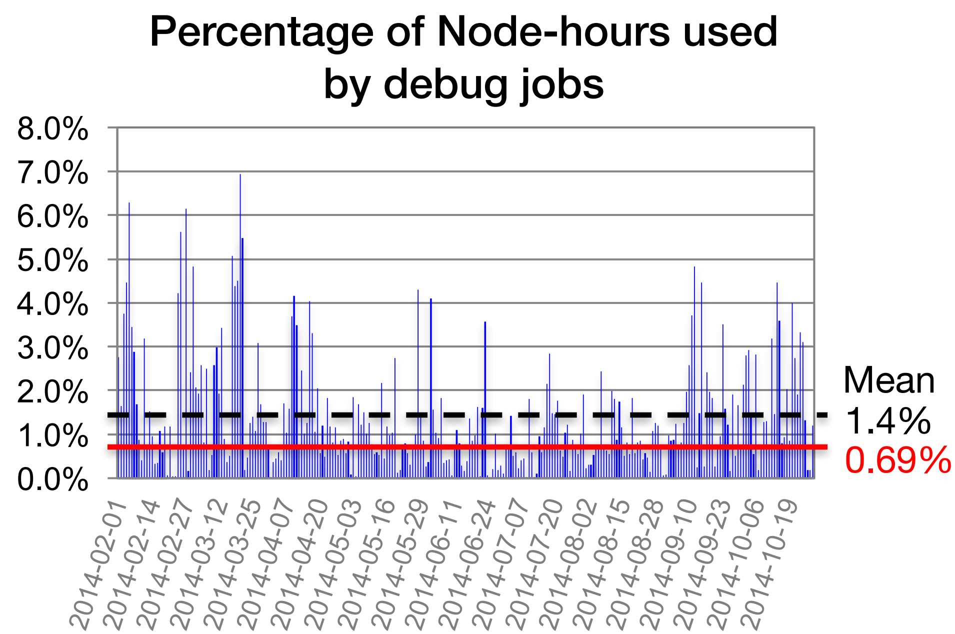




Feedback to the job scheduler

Ensuring High System Utilization





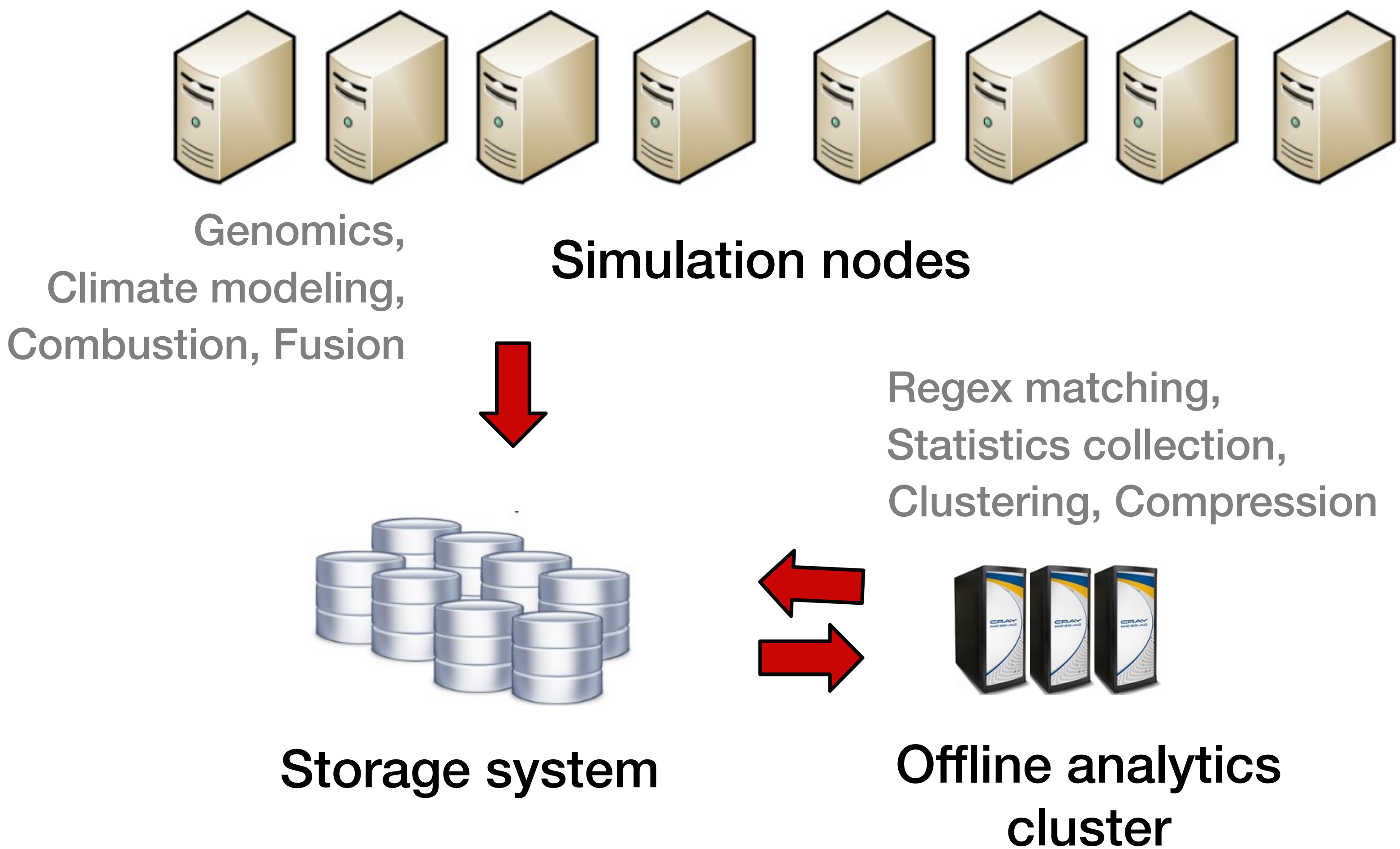
Significant fraction of failures can be avoided from interrupting production applications

Debug or non-production jobs can be scheduled on quarantine nodes

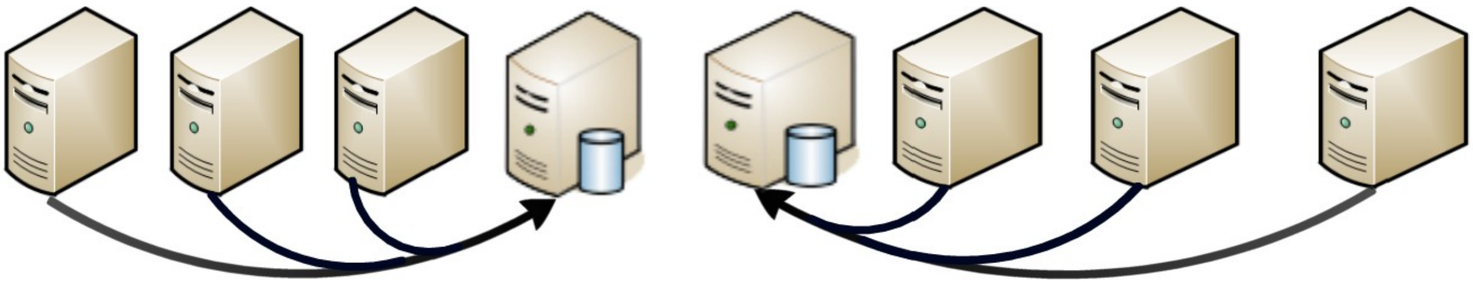
In-Situ Data Analysis via Active Computation on Emerging Storage Devices

Active Flash

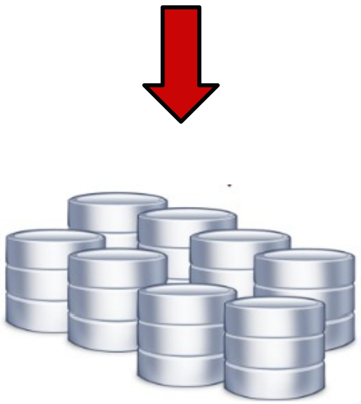
Traditional Scientific Data Analysis via Offline Cluster



In-situ Data Analysis via Active Computation on SSDs

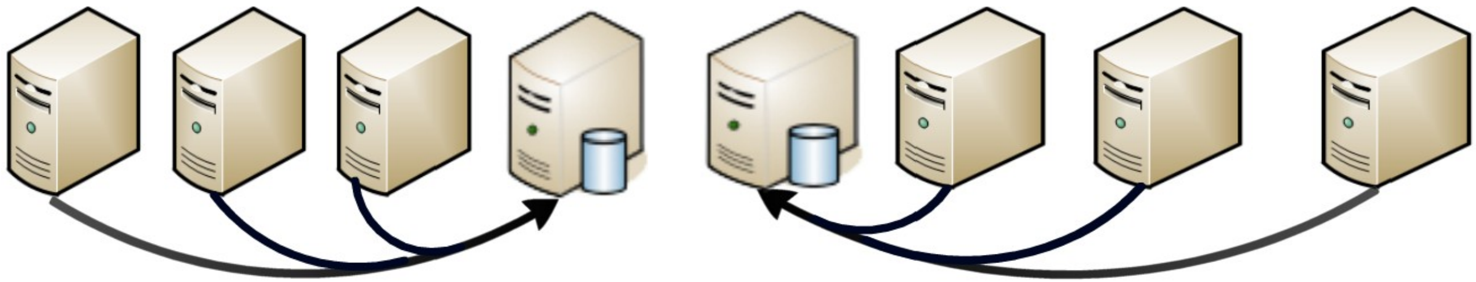


Scientific data analysis performed on SSD controllers concurrently without hurting simulation performance

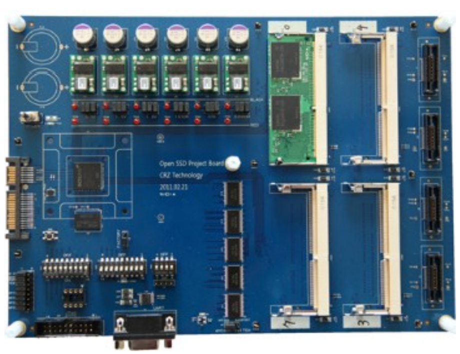


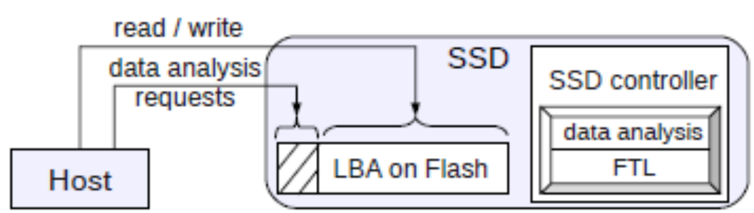
Storage system

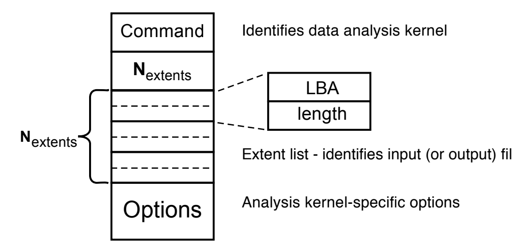
In-situ Data Analysis via Active Computation on SSDs



Enabled by increasingly multi-core controllers in SSDs

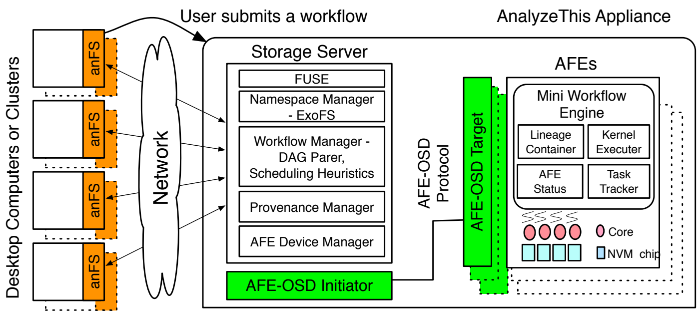


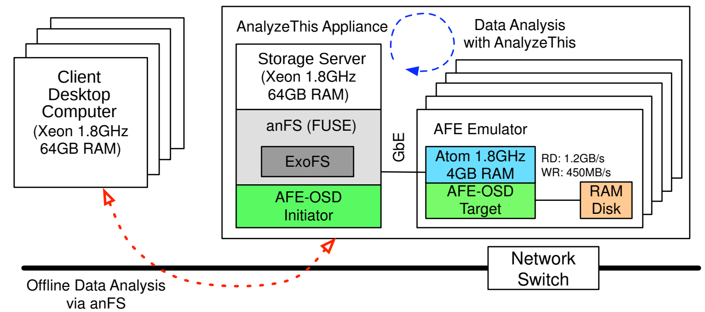




Feasibility of the approach demonstrated by prototype implementation on OpenSSD platform, but…

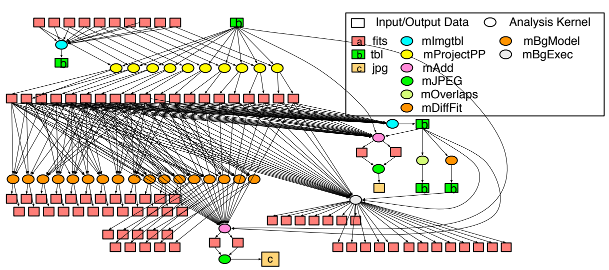
Beyond Active Flash

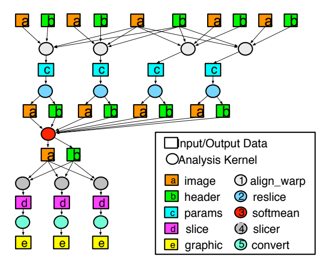


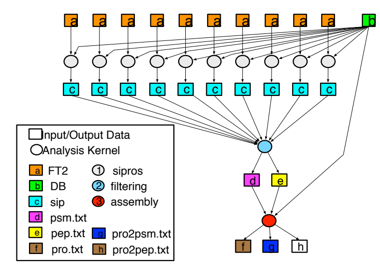


AnalyzeThis architecture

AnalyzeThis testbed set-up



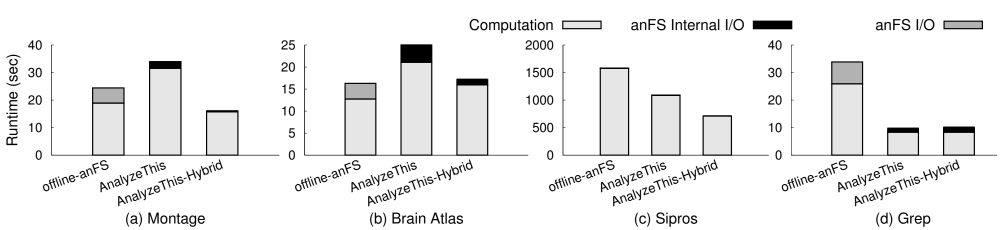


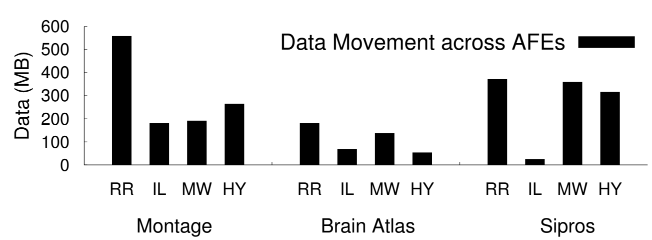


Montage (astronomy)

Brain (neurology)

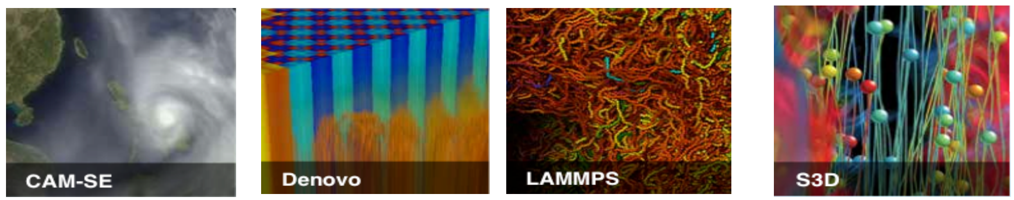
Sipros (DNA)





Exciting Opportunities in Sustainable Exascale Computing

Data-centric Hybrid Systems



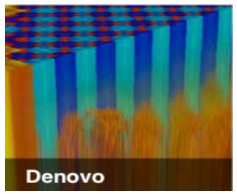
Data-intensive applications

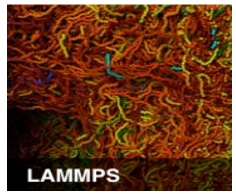
Runtime systems, libraries, system log, job scheduler, resource manager

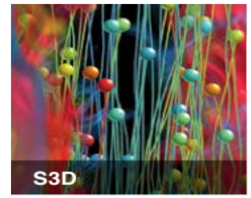
amazon web services EC2

Large-scale compute & storage systems

Intelligently Operating Future Systems







Data-intensive applications

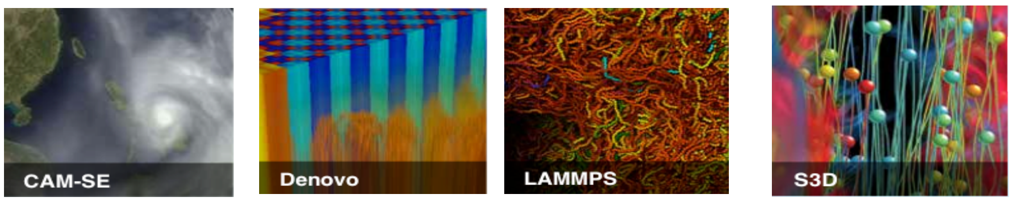
Runtime systems, libraries, system log, job scheduler, resource manager

amazon web services EC2

Large-scale compute & storage systems

Need for effectively managing elastic computing resources (cloud + HPC)

Intelligently Operating Future Systems



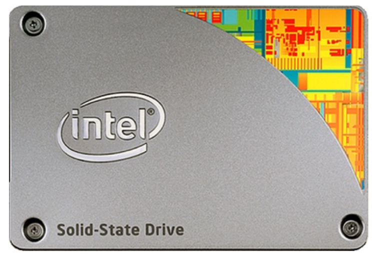
Data-intensive applications

Runtime systems, libraries, system log, job scheduler, resource manager

Large-scale compute & storage systems

Need for intelligent, actionable analytical tools for improving system efficiency (performance, reliability, and cost-efficiency)

Trends and Observation



Heterogeneity in performance, cost, and reliability

Heterogeneity in power-efficiency, programmability, and scalability

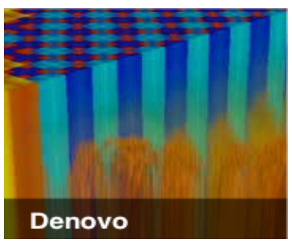
intel inside™
Xeon Phi™

Future devices are likely to be priced according to their reliability and power consumption levels for a given performance level

Focus Areas

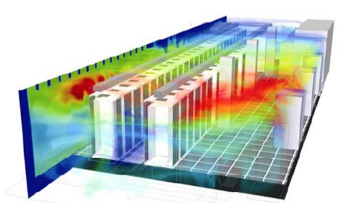
Elastic Computing: Manage "dynamic" heterogeneity efficiently

Actionable analytical tools, and techniques to reduce cost and queue wait time, and better dynamic provisioning of system



Understanding and leveraging application-specific characteristics

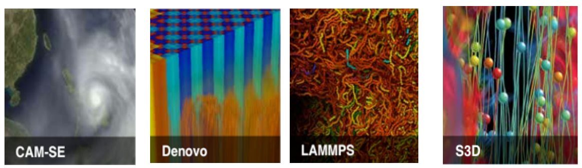
Fault tolerance characteristics, power capping effects on performance, hardware power and resilience knob tuning



Exploiting environmental and power/cooling conditions

Develop new techniques for improved systems performance; power, performance, and reliability trade-offs

In-situ data analytics approach alone will not be sufficient at exascale



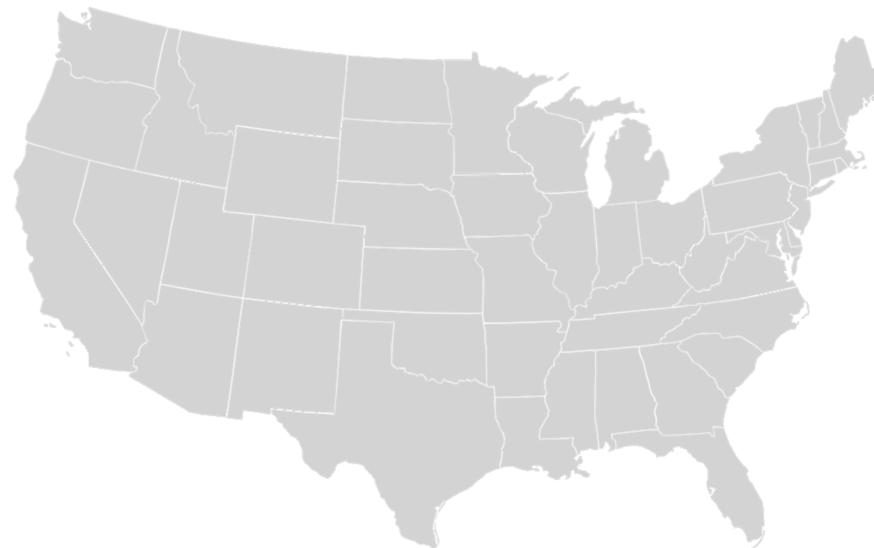
Data-intensive applications

Compute system

SSD-based burst buffer

Storage system

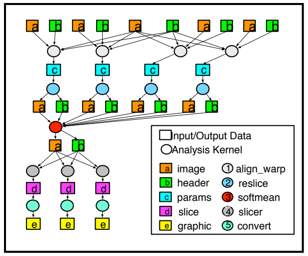
Analysis cluster



Multiple orders of increase in the data production rate for large-scale applications

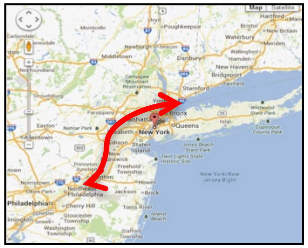
Rapid increase in multi-site data-intensive workflows

Focus Areas



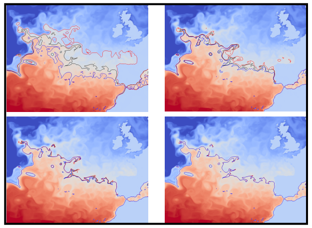
Preserving data provenance

Eliminate computational redundancy (MapReuse)



Opportunistic data analysis on the fly

Statistical and probabilistic sampling



Answer quality trade-offs

Lossy compression and approximate analysis

Thanks!

Devesh Tiwari

tiwari@northeastern.edu