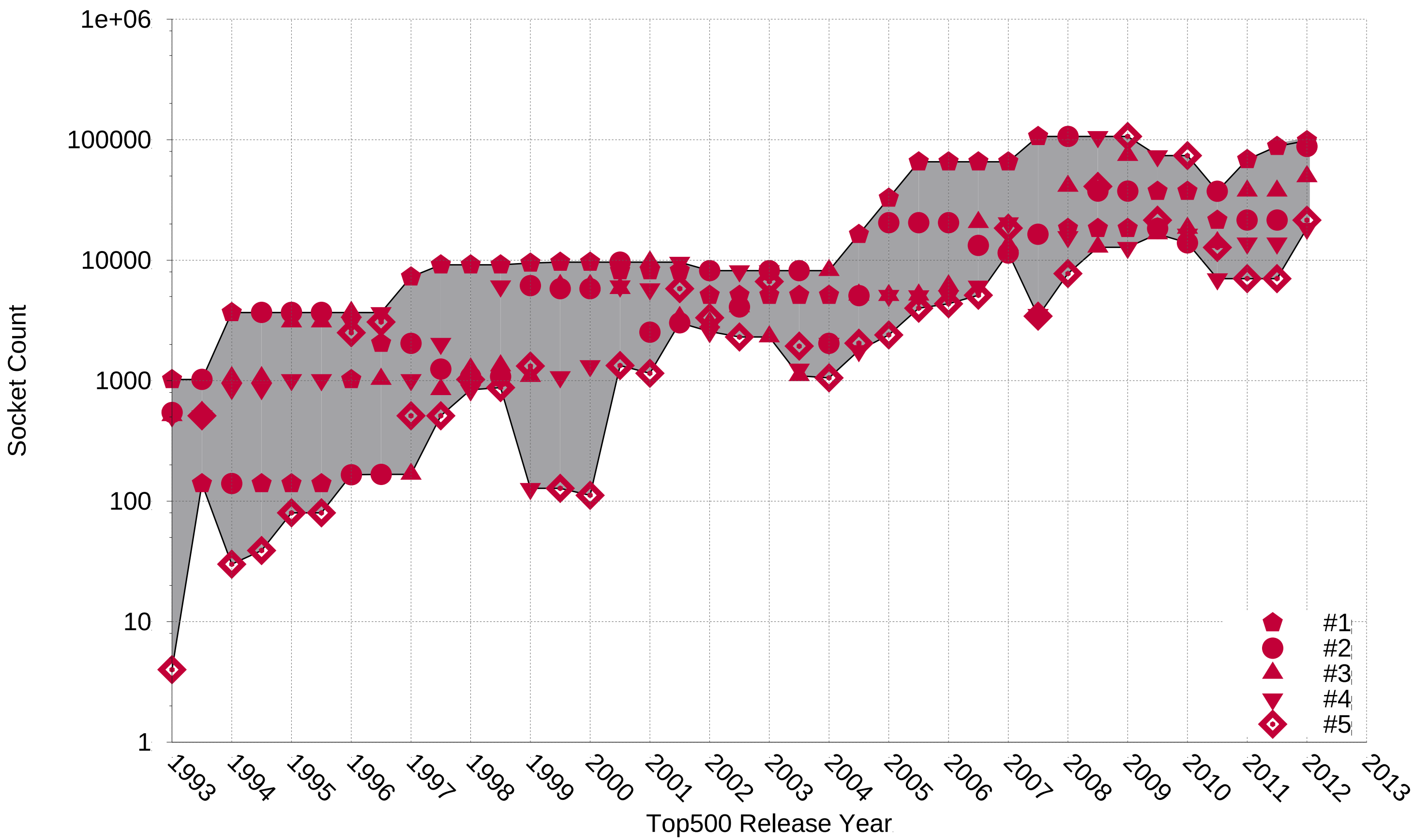
TORSTEN HOEFLER

# Using Simulation to Evaluate the Performance of Resilience Strategies at Scale

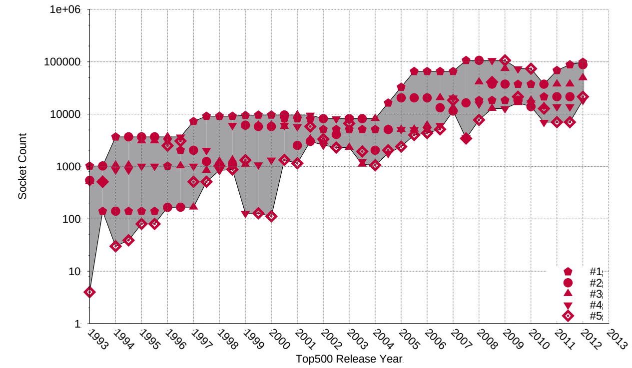
**in collaboration with Scott Levy, Bryan Topp, Dorian Arnold, Kurt B. Ferreira, Patrick Widener, UNM + SNL, Albuquerque, NM, USA**

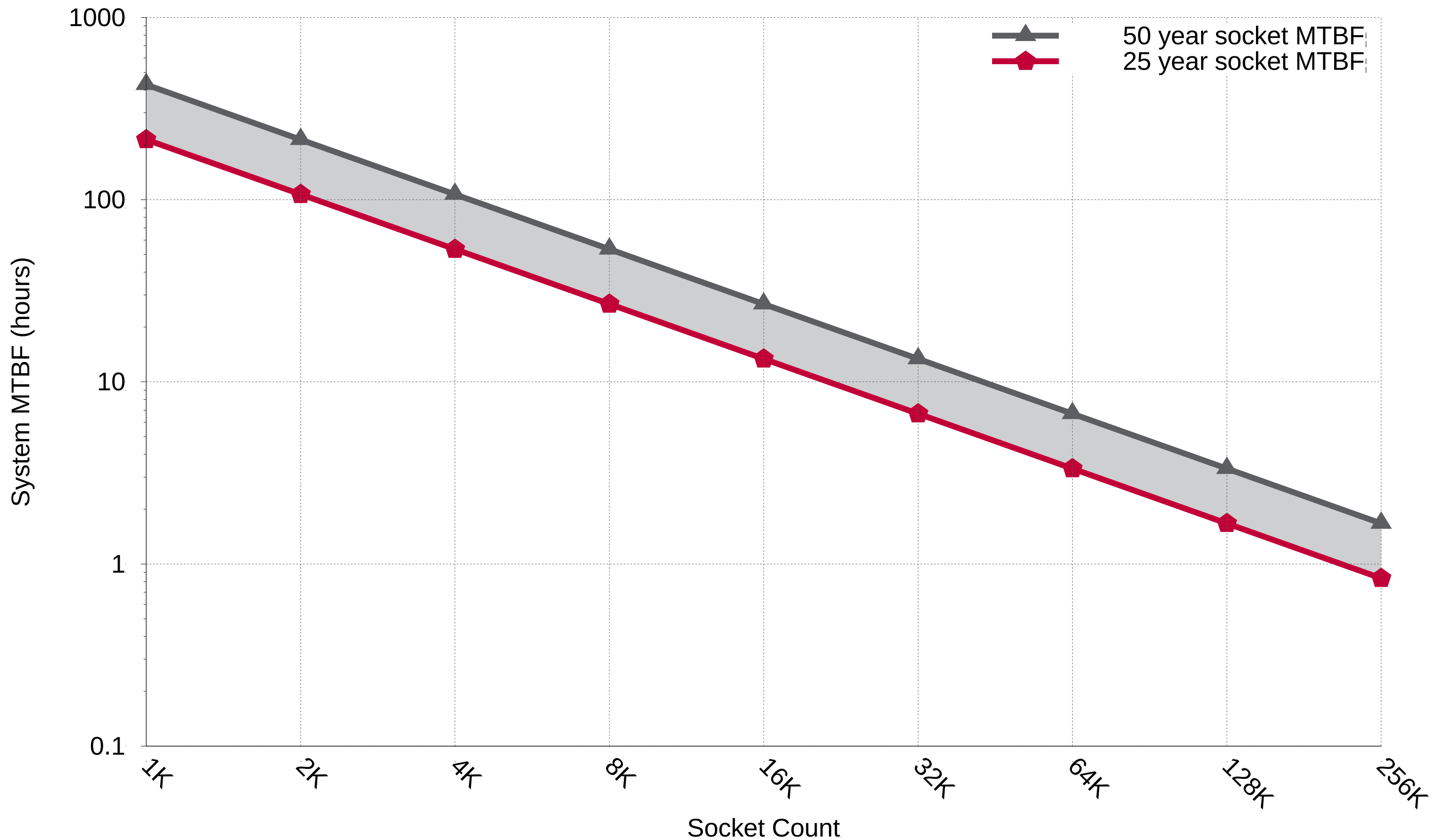
# Resilience Matters

- ... because scale matters
- Scientific workloads demand larger, more powerful systems



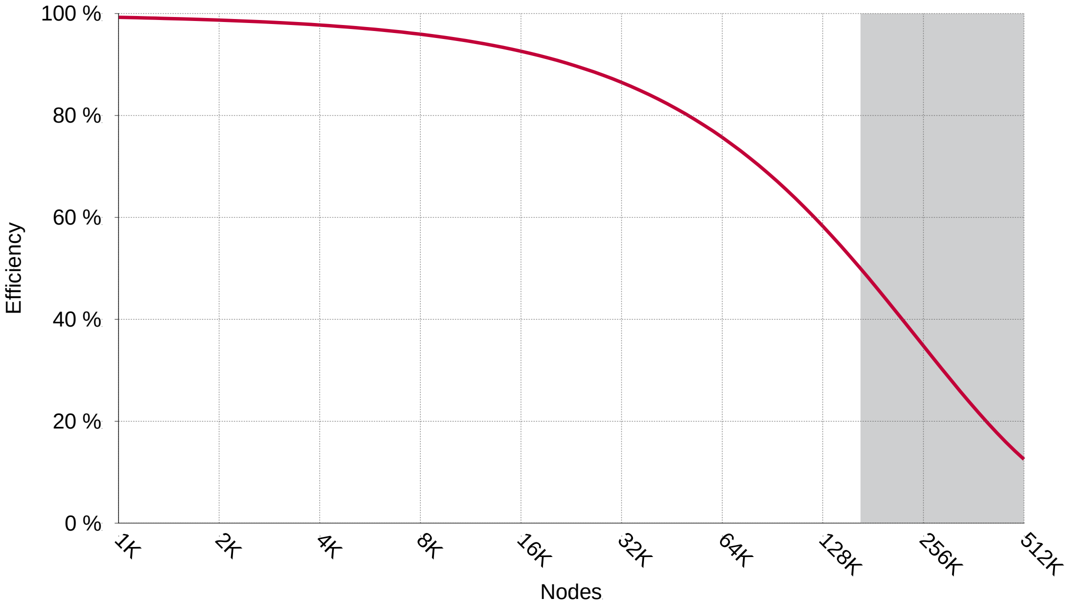
# Bigger Systems = More Failure



# Coordinated Checkpoint/Restart May Not Scale

- Dominant approach to handling failure is coordinated checkpoint/restart
- May be prohibitively expensive for very large systems




- Many alternatives have been proposed

# Evaluating resilience at scale is difficult

- **Small-scale testing**
  - cannot account for the impact of scale
  - lacks advanced hardware features

- **Analytic models**
  - good models exist for coordinated checkpointing
  - ... but non-existent for novel resilience techniques

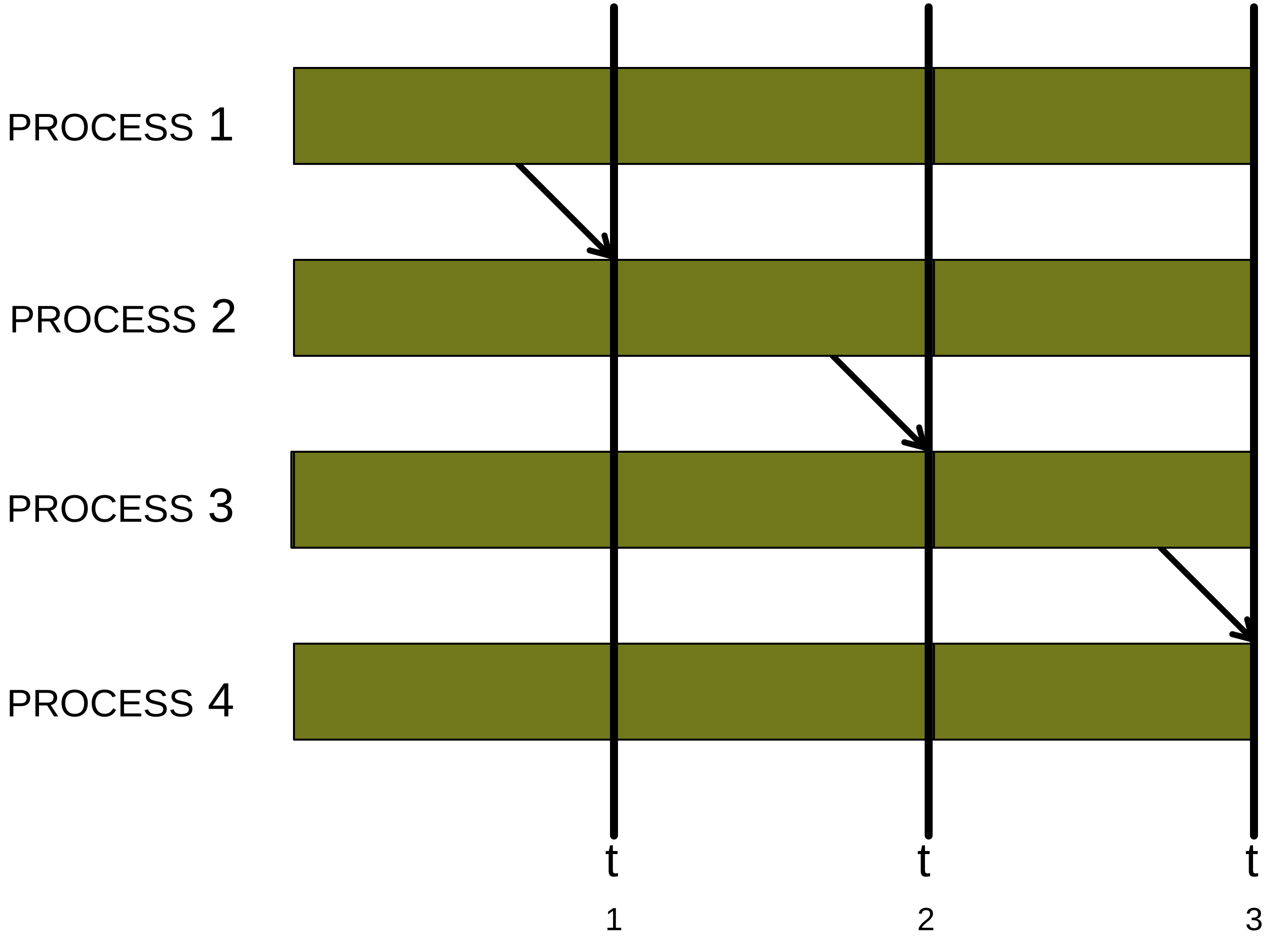
- **Use simulation!**
  - Key observations:
    1) *Resilience is composed of coarse-grained operations; cycle-accurate simulation may be unnecessary*
    2) *Simulation can be expensive; identify those characteristics that are necessary for accuracy*

# Key contributions so far

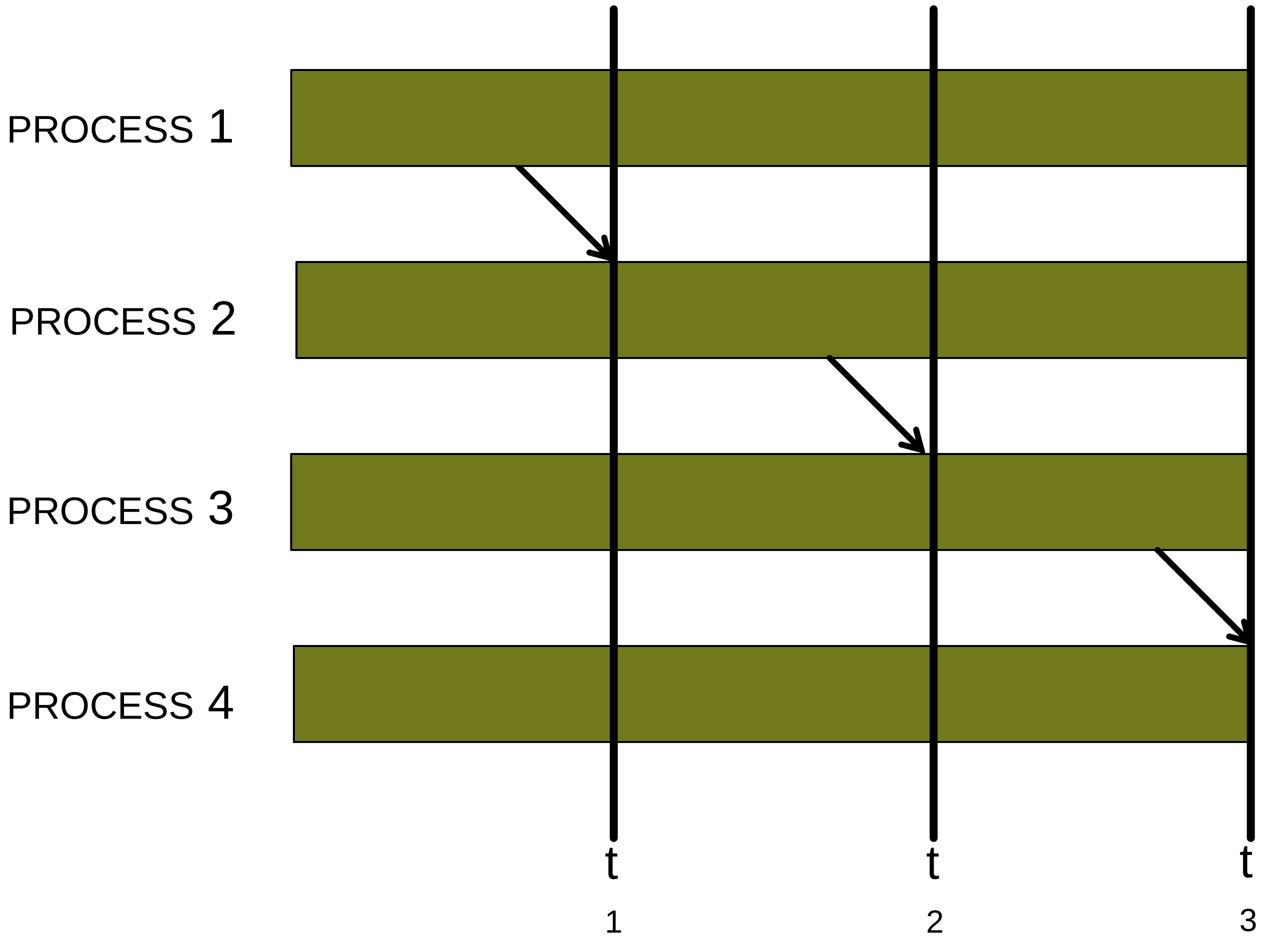
- **Simulation is a powerful technique for examining resilience techniques at scale [1]**
- **Accurate simulation is possible using a small number of coarse-grained platform and application characteristics [1]**
- **Modeling resilience events as CPU detours enables efficient simulation [1]**
- **Overheads of uncoordinated checkpointing [2]**
- **Selection schemes for uCR vs. cCR [2]**

[1] Levy, et al. "Using Simulation to Evaluate the Performance of Resilience Strategies at Scale", PMBS workshop, SC13
[2] Ferreira, et al. "Understanding the Effects of Communication and Coordination on Checkpointing at Scale", to appear at SC14

# Example: Coordinated C/R



# Example: Uncoordinated C/R



# Simulating Application CR

- **Application trace:**
  - COMPUTATION TIME: time spent outside of communication
  - COMMUNICATION GRAPH: which processes communicate
  - DEPENDENCIES: partial ordering of communication and computation
- **Machine characteristics:**
  - CHECKPOINT TIME: time taken away from the application for checkpointing activities

    *Coordination, checkpoint computation, checkpoint commit*
  - CHECKPOINT INTERVAL: time between checkpoints
  - FAILURE CHARACTERIZATION: a description of when failures occur (e.g., a probability distribution)
  - REPAIR TIME: time that must elapse following a failure before the hardware resources are available
  - RECOVERY MODEL: description of time between restoration of hardware and meaningful forward progress

# Where is the collaboration?

- **Switzerland has the simulator**
  - Based on LogGOPS (a descendent of LogP) [1]
  - Provides many of the features that we require
  - Composed of three components

    *a trace collector*

    *a schedule generator*

    *discrete-event simulator*

- **Sandia/UNM have the FT toolchain**
  - Protocols and models (libsilopsis)
  - Applications, experience

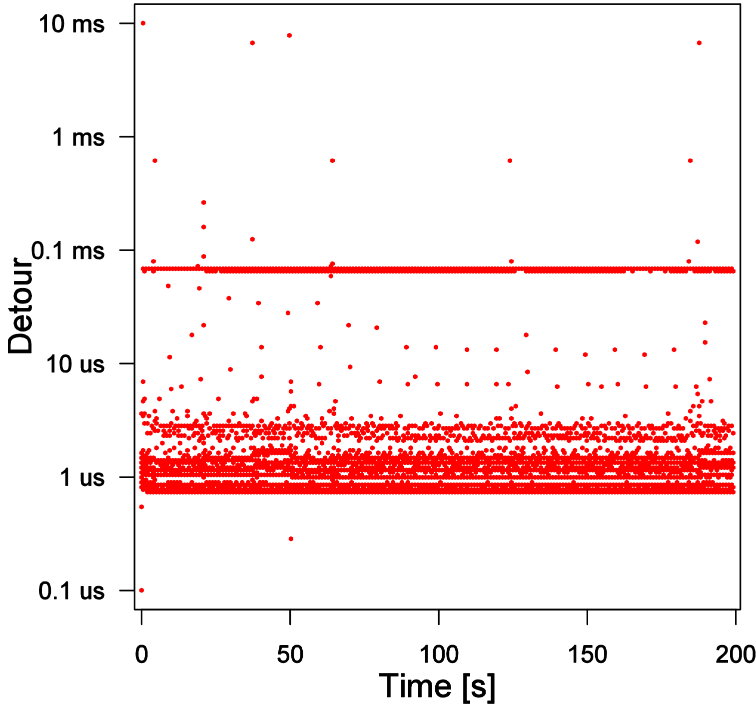
[1] Hoefler et al. "LogGOPSim - Simulating Large-Scale Applications in the LogGOPS Model", LSAP/HPDC 2010

# Simulating Fault Tolerance with LogGOPSim

- **Key insight: fault tolerance can be modeled as CPU detours [1]**

- **Because of LogGOPSim's history it has a convenient interface for CPU detours [2]**

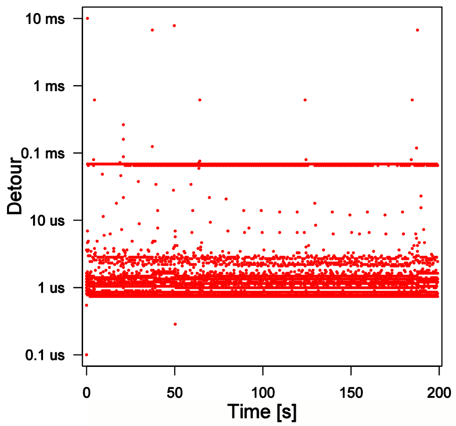
- **libsolipsis: generates CPU detours for a particular application and fault tolerance mechanism**
  - for example: $T_{detour} = T_{coord} + T_{ckpt} + T_{commit}$



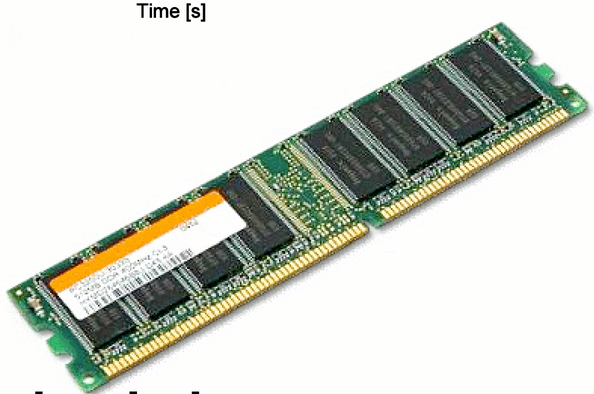

[1] Levy et al. "Using Simulation to Evaluate the Performance of Resilience Strategies at Scale", PMBS workshop, SC13
[2] Hoefler et al. "Characterizing the Influence of System Noise on Large-Scale Applications by Simulation", SC10

# Simulating Platform's Temporal Scale

- **LogGOPSim wasn't built for this purpose**
  - Optimized for massive short simulations

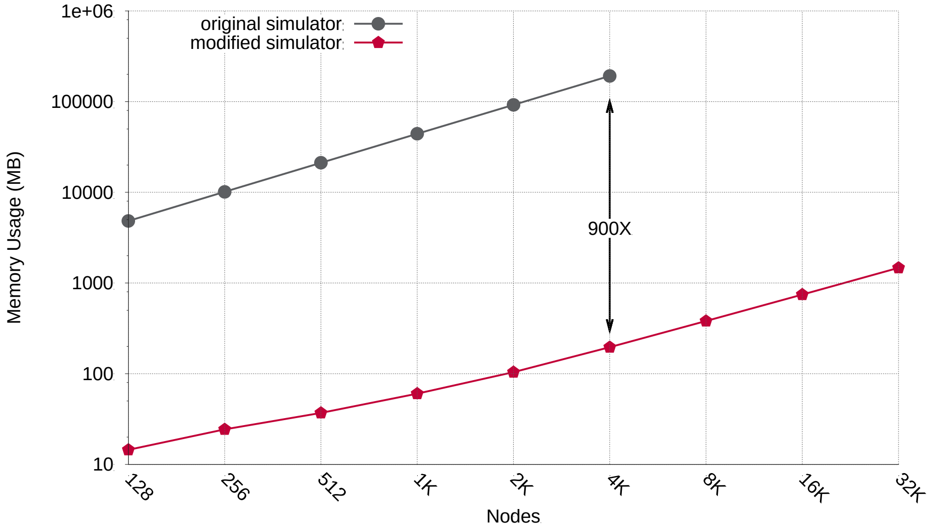



- **Simulated time limited by available memory**
  - Trace extrapolation in memory
  - MTBF in the order of years …

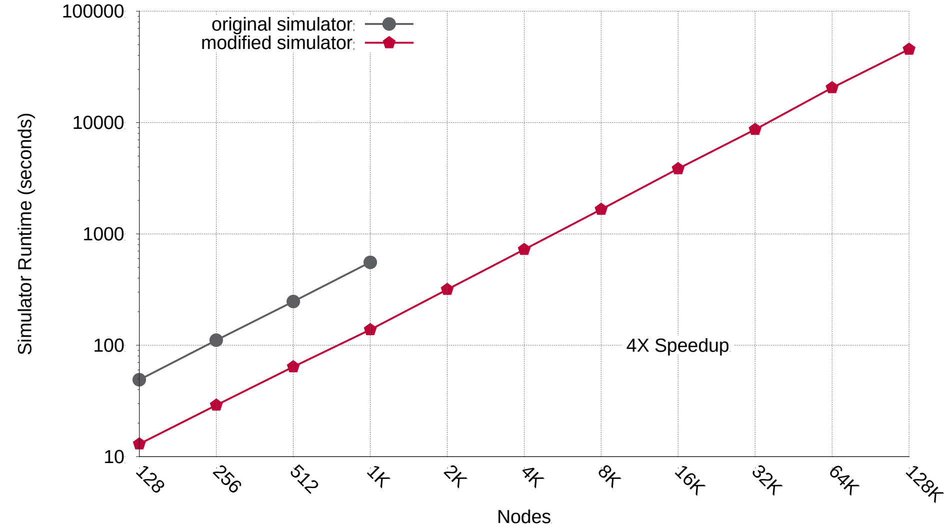


- **Modified trace handling to increase length of simulations**
  - Several additional minor improvements
  - Thanks to S. Levy!

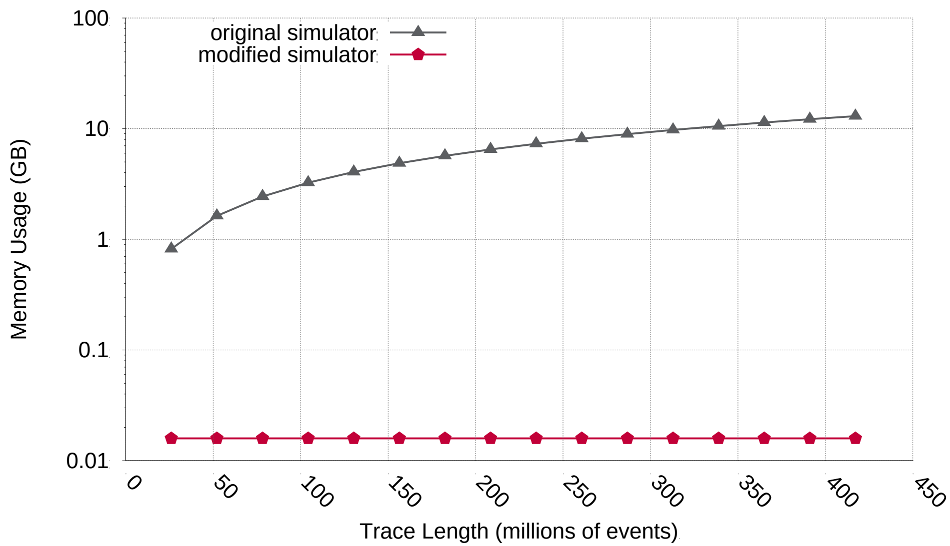
# Benefits of Improved Trace Handling




Reduced Memory Usage




Faster Simulation




Memory Usage Independent of Trace Length

# Validation

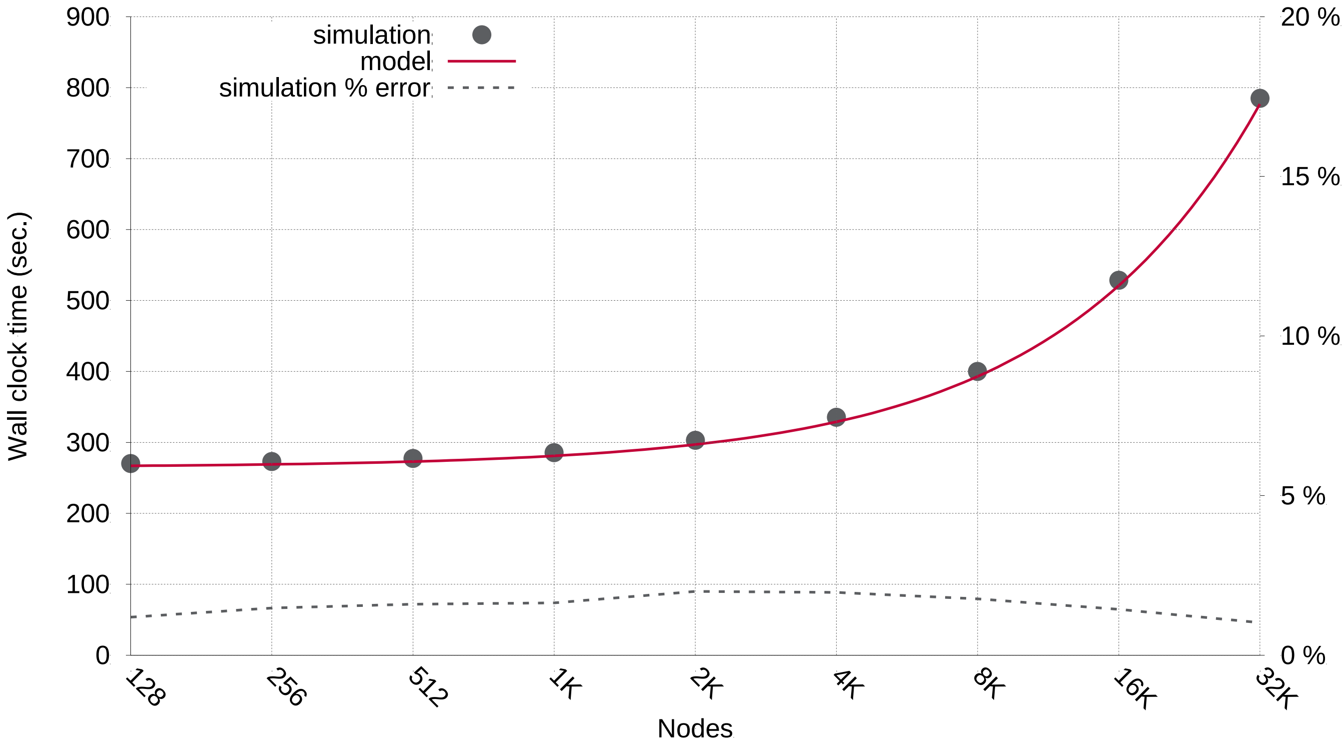
- **Use two important production workloads**
  - CTH: shock physics code
  - LAMMPS: molecular dynamics code


- **Compare against:**
  - Model of failure-free coordinated checkpointing
  - Small-scale testing

- **Simulation of failures has been added and validated**

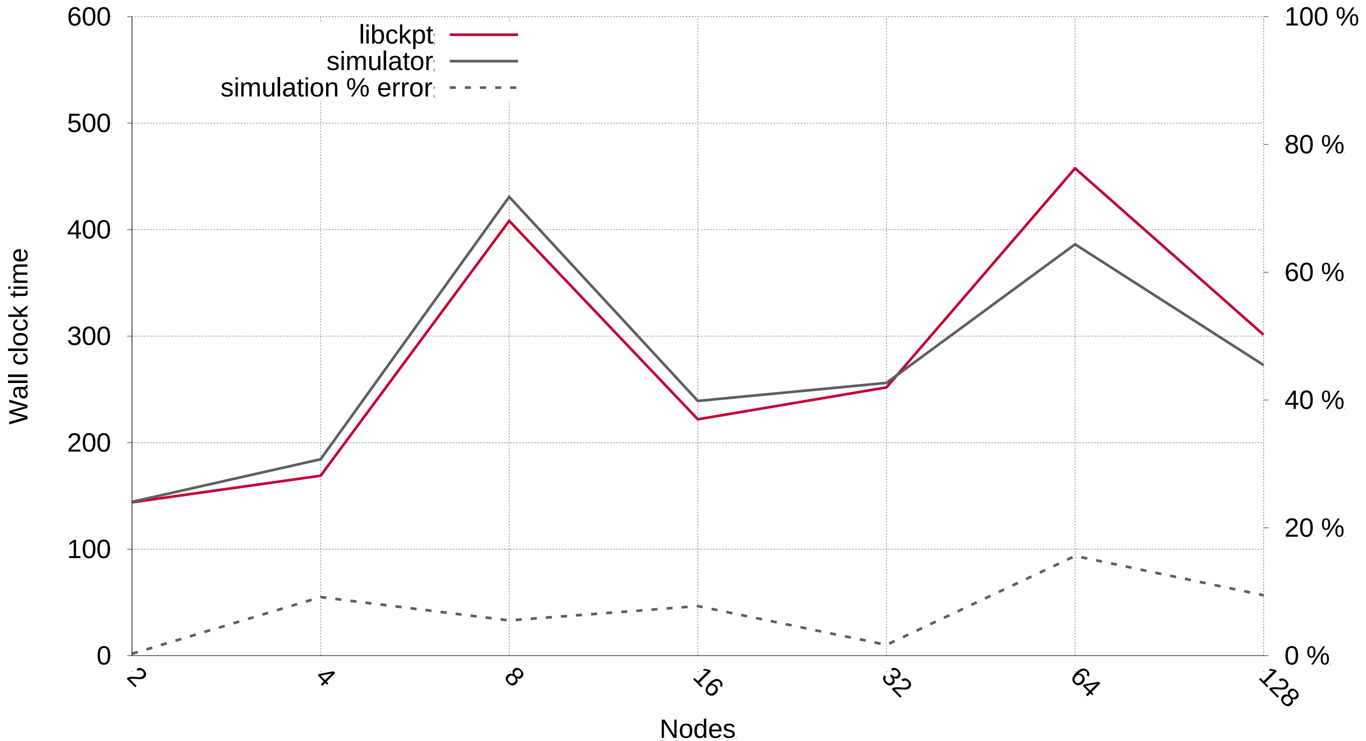
# Validation: analytic model

- **Model of failure-free coordinated checkpointing**
  - LAMMPS within 1%
  - CTH within 3% (see below)



# Validation: small-scale testing

- **Tests with coordinated & uncoordinated checkpointing**
  - LAMMPS within 5%
  - CTH within 16% (coordinated checkpointing results shown)



# Future Work


- **Additional resilience mechanisms:**
  - hierarchical checkpointing
  - process replication
  - communication-induced checkpointing
- **Additional performance improvements (e.g., parallelization)**
- **Explore the performance impact of uncoordinated checkpointing**

# Mode of collaboration

Google+ Hangouts

LaTeX

SUBVERSION

unfunded, getting funding is hard …

# Conclusion

- **Simulation is an effective approach to exploring the performance impact of fault tolerance on extreme-scale systems**
- **Coarse-grained system and application characteristics enable high fidelity simulation of resilience**
- **Our prototype simulator enable further investigation into emerging fault tolerance techniques**
- **ACKNOWLEDGMENTS:**
  - UNIVERSITY OF NEW MEXICO

    *Bryan Topp & Dorian Arnold & Scott Levy*
  - SANDIA NATIONAL LABORATORIES

    *Kurt B. Ferreira & Patrick Widener*

THE UNIVERSITY of NEW MEXICO

Sandia National Laboratories

# Questions?