A `Cool' Way of Improving the Reliability of HPC Machines

Osman Sarood, Esteban Meneses, and Laxmikant Kale

Parallel Programming Laboratory (PPL)
University of Illinois Urbana Champaign

Fault tolerance in present day supercomputers

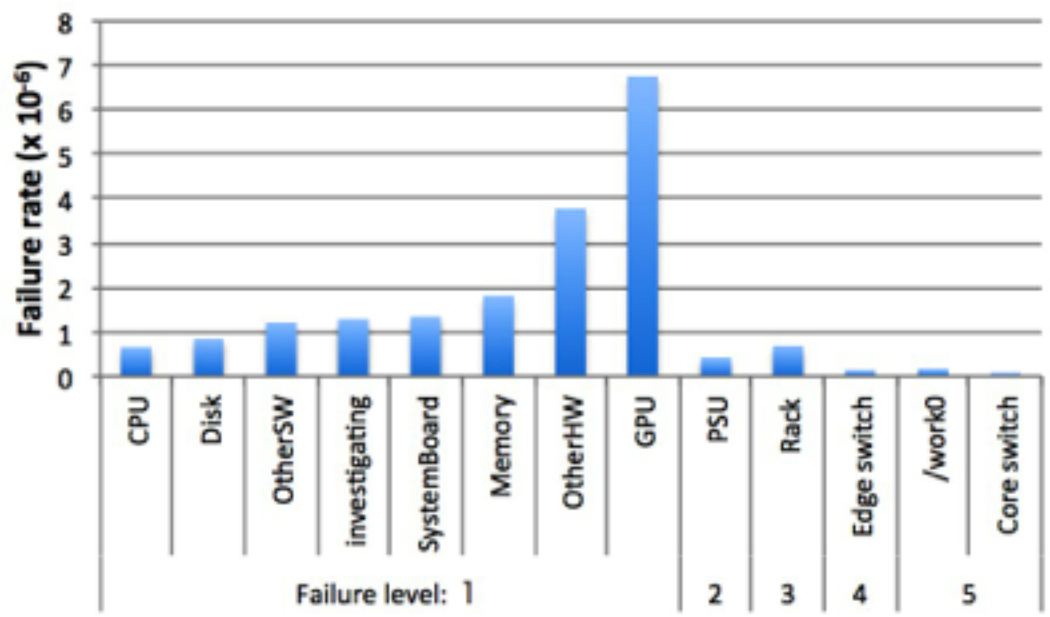
- Energy, power and reliability!
- Earlier studies point to per socket Mean Time Between Failures (MTBF) of 5 years - 50 years
- More than 20% of computing resources are wasted due to failures and recovery in a large HPC center[1]
- Less than 11% machine efficiency for exascale machine with 200,000 sockets[2]

1. Ricardo Bianchini et. al., System Resilience at Extreme Scale, White paper
2. Kurt Ferreira et. al., Evaluating the Viability of Process Replication Reliability for Exascale Systems, Supercomputing'11

Tsubame2.0 Failure Data[4]

Component	MTBF
Core Switch	65.1 days
Rack	86.9 days
Edge Switch	17.4 days
PSU	28.9 days
Compute Node	15.8 hours

- Tsubame2.0 failure rates
  - Compute failures are much frequent
  - High failure rate due to increased temperatures



4. Kento Sato et. al., Design and Modeling of a Non-Blocking Checkpointing System, Supercomputing'12

CPU Temperature and MTBF

- 10 degree rule: MTBF halves (failure rate doubles) for every 10C increase in temperature[3]
- MTBF (m) can be modeled as:

$$m = A * e^{-b*T}$$

where 'A', 'b' are constants and 'T' is processor temperature

- A single failure can cause the entire machine to fail, hence MTBF for the entire machine (M) is defined as:

$$M = \frac{1}{\sum_{n=1}^{N} \frac{1}{m_n}}$$

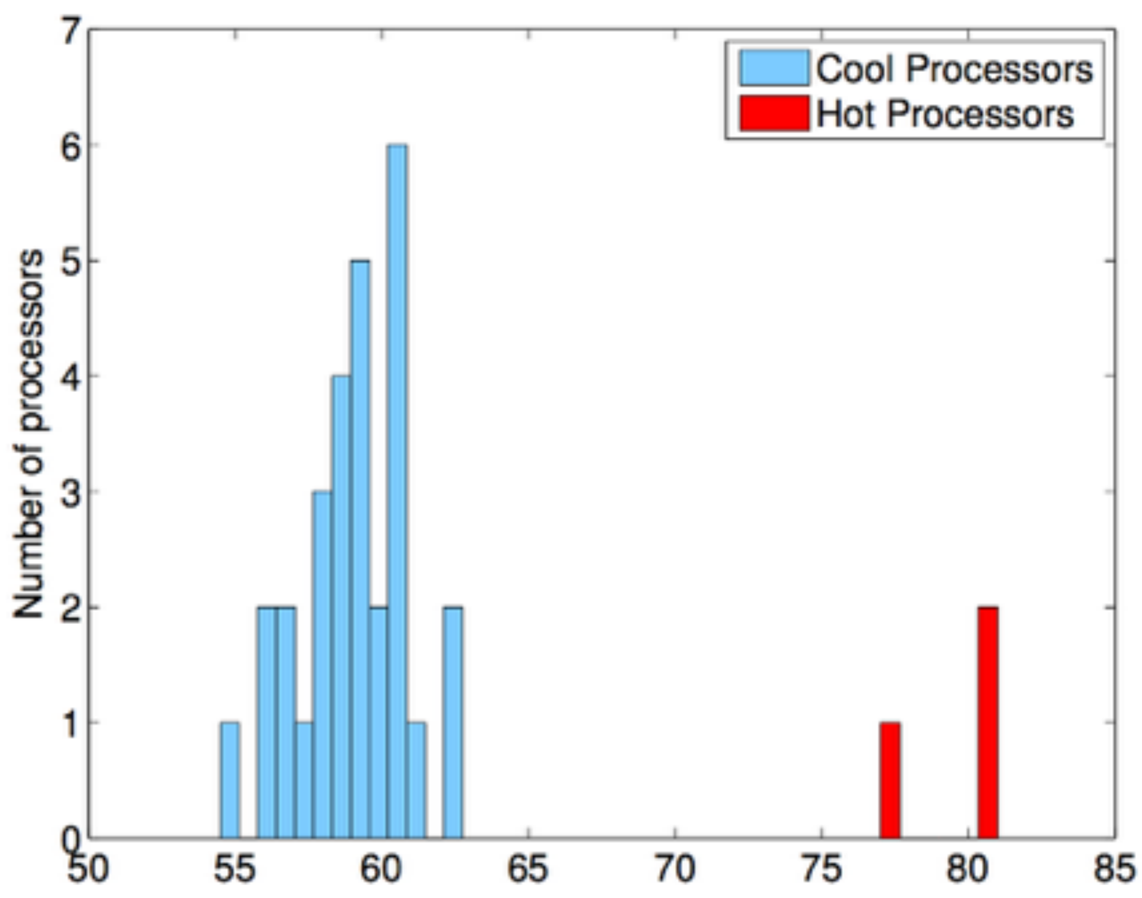
3. Wu-Chun Feng, Making a Case for Efficient Supercomputing, New York, NY, USA

Related Work

- Most earlier research focusses on improving fault tolerance protocol (*dealing efficiently with faults*)
- Our work focusses on reducing the MTBF (*reducing the occurrence of faults*)
- Our work can be combined with any fault tolerance protocol

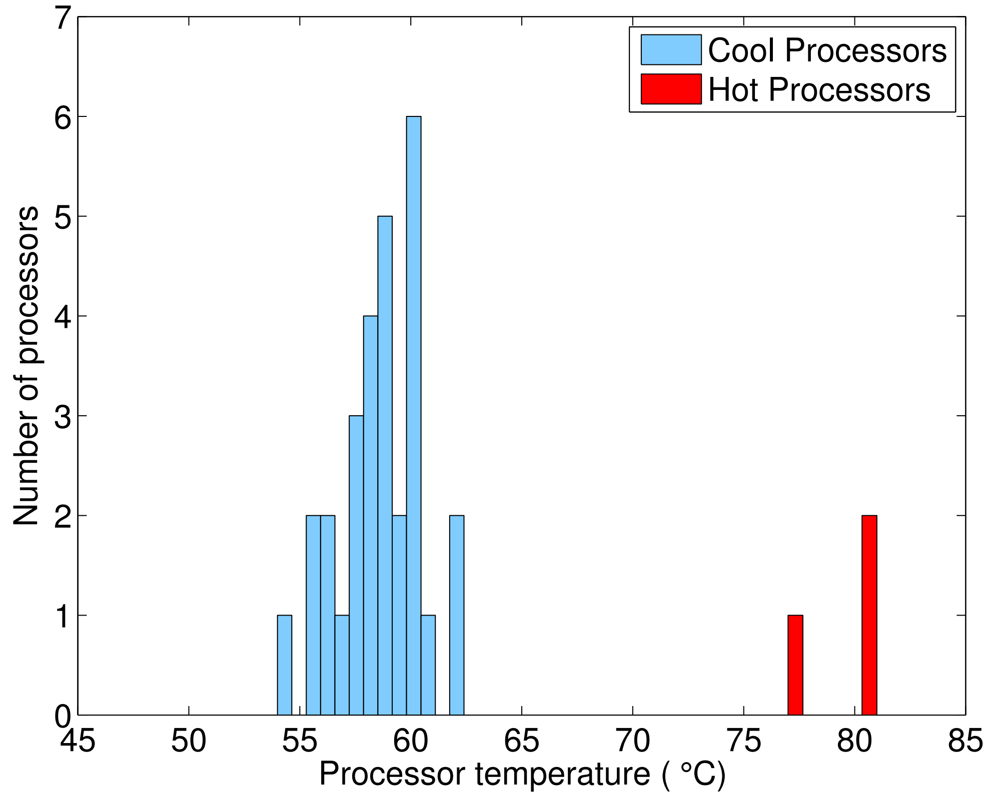
Distribution of Processor Temperature

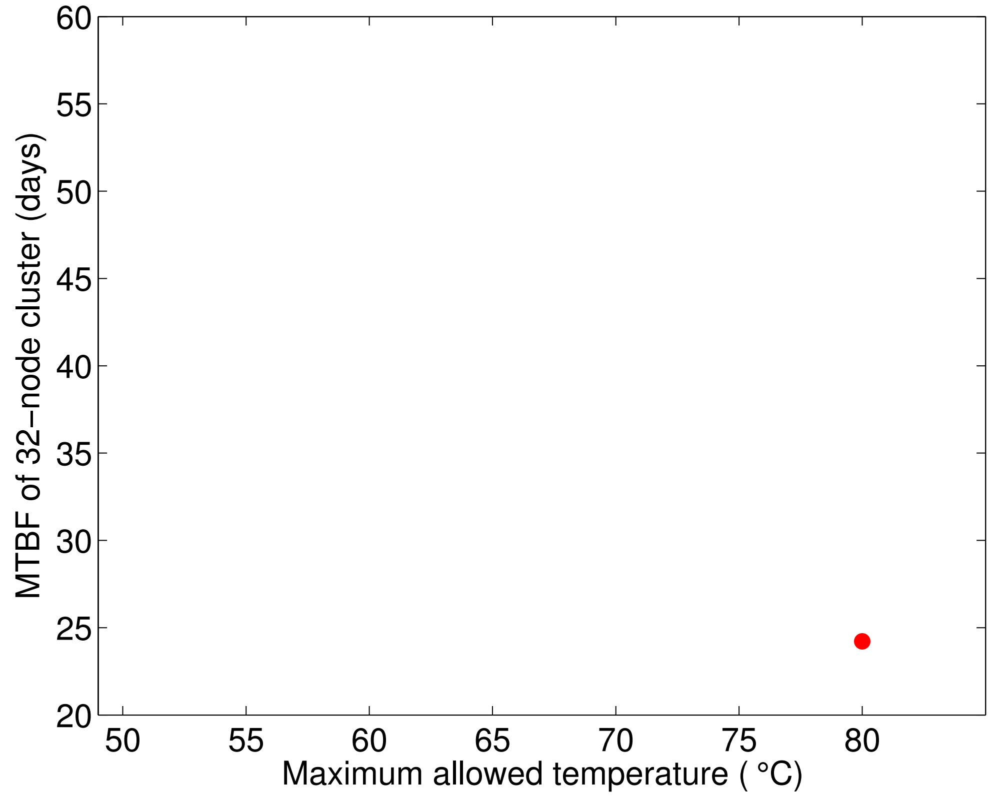
- 5-point stencil application (Wave2D from Charm++ suite)
- 32 node cluster (single socket, Intel Xeon X3430)
- Cool processor mean: 59C, std deviation: 2.17C



Estimated MTBF - No Temperature Restraint

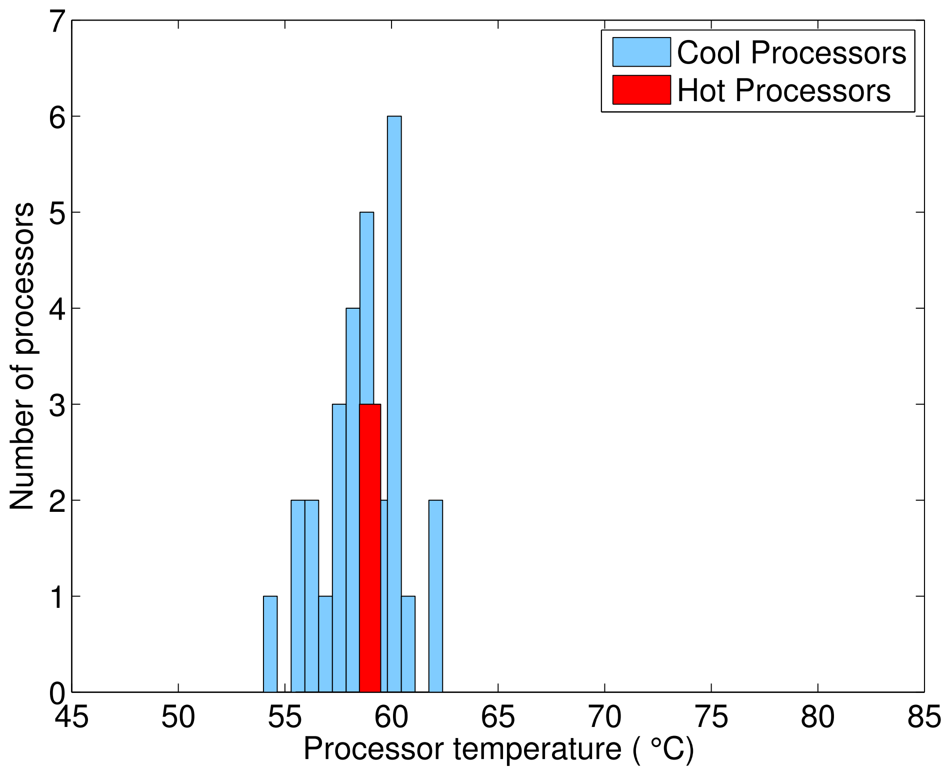
- Using observed max temperature data and per-socket MTBF of 10 years (cool processor mean: 59C, std deviation: 2.17C)
- Formula for M: $m = 160 * e^{-0.069T}$ $M = \frac{1}{\sum_{n=1}^{N} \frac{1}{m_n}}$

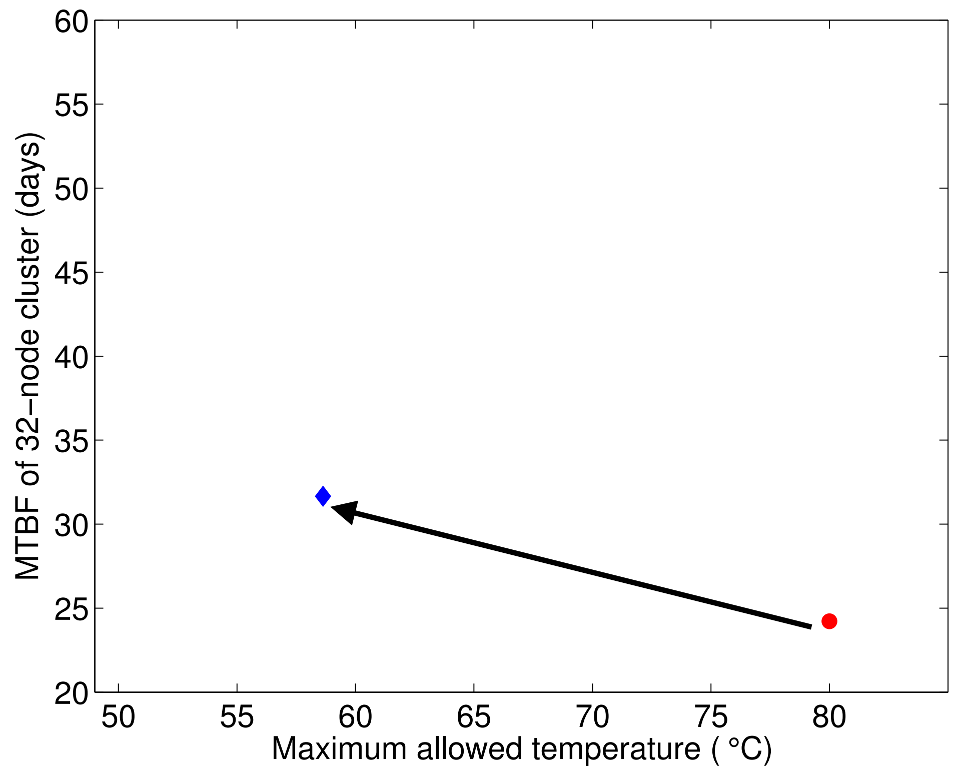




Estimated MTBF - Removing Hot Spot

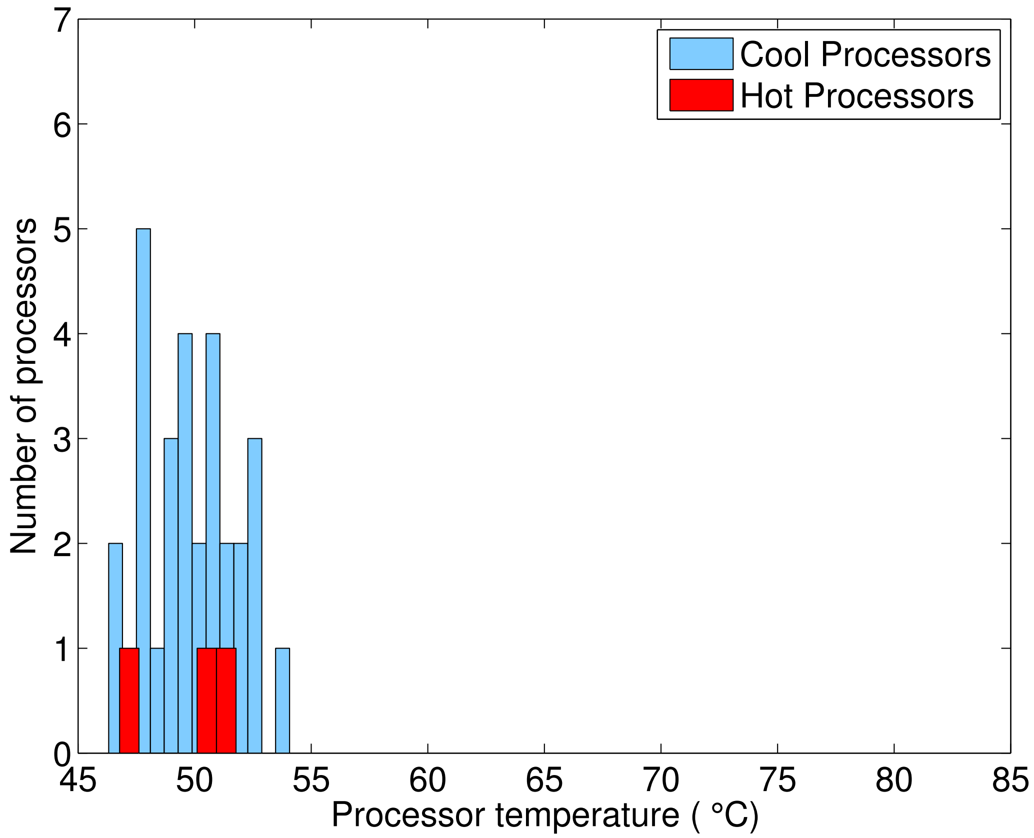
- Using measured max temperature data for cool processors and 59C (same as average temperature for cool processors) for hot processors

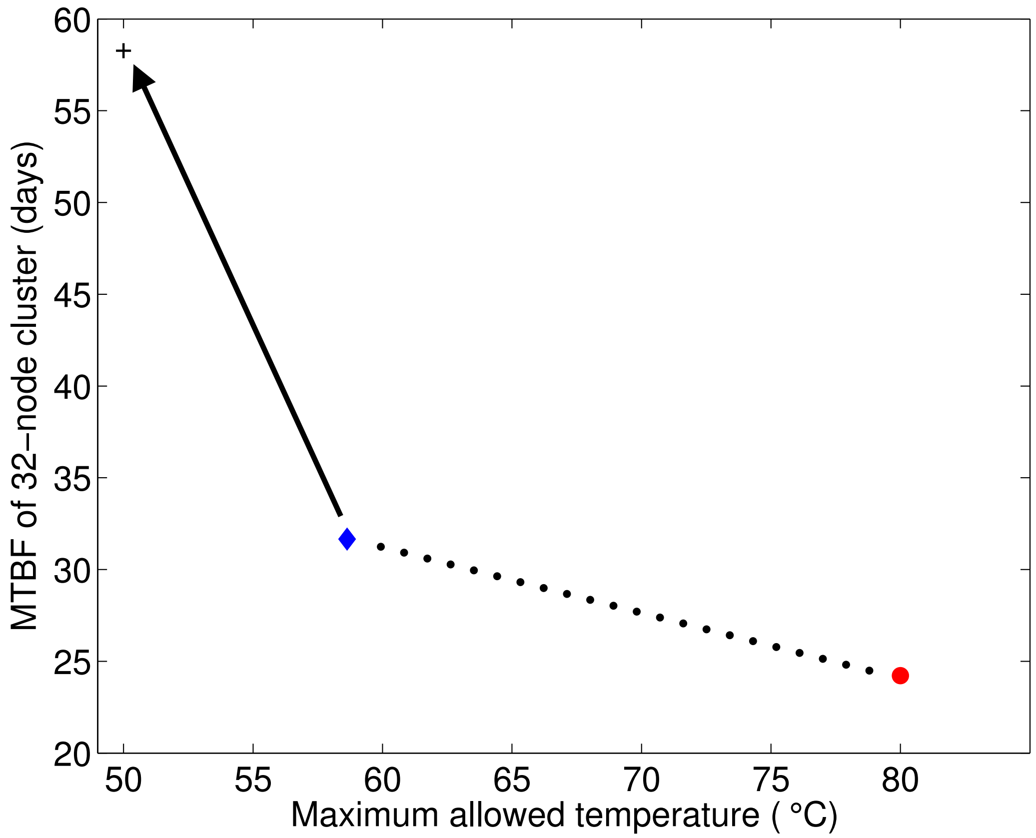




Estimated MTBF - Temperature Restraint

- Using randomly generated temperature data with mean: 50C and std deviation: 2.17C (same as cool processors from the experiment)





Recap

- Restraining temperature can improve the estimated MTBF of our 32-node cluster
    - Originally (No temperature control): 24 days
    - Removing hot spots: 32 days
    - Restraining temperature (mean 50C): 58 days
- How can we restrain processor temperature?
    - Dynamic Voltage and Frequency Scaling (DVFS)[5]?

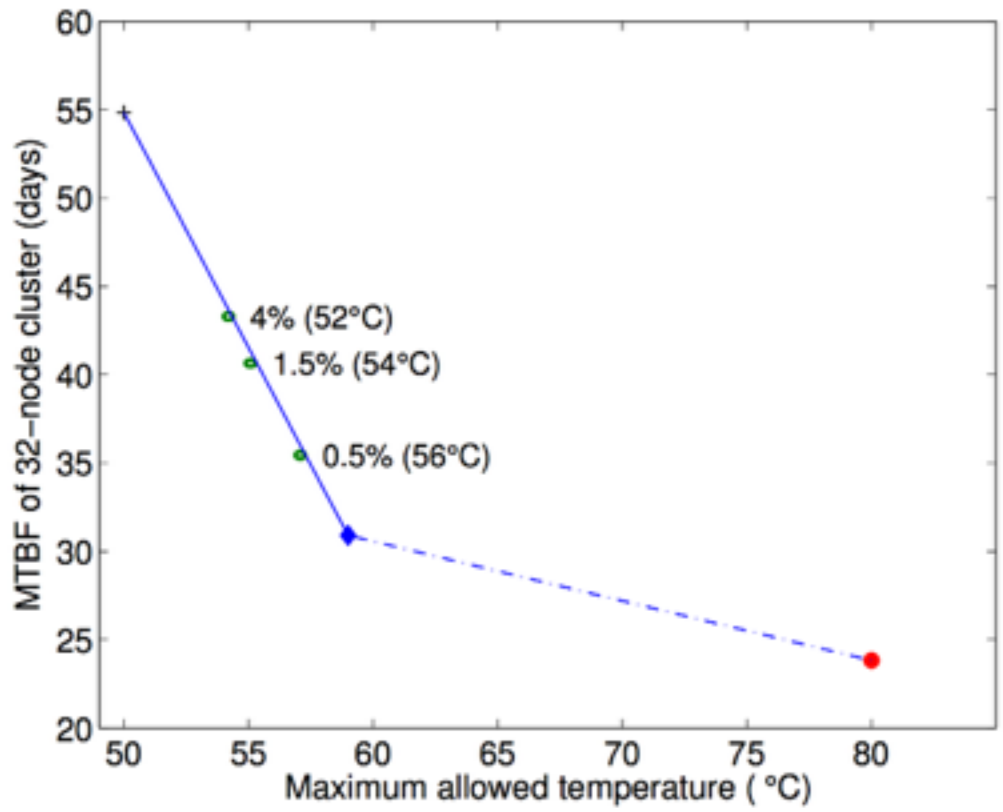
5. Reduces both voltage and frequency which reduces power consumption resulting in temperature to fall

Restraining Processor Temperature

- Extension of ‘Cool Load Balancer’ from SC’11
- Specify temperature threshold and sampling interval
- Runtime system periodically checks processor temperature
- Scale down/up frequency (by one level) if temperature exceeds/below maximum threshold at each decision time
- Transfer tasks from slow processors to faster ones
- Extended by making it communication aware (details in paper):
    - Select objects (for migration) based on the amount of communication it does with other processors

Estimated MTBF For Different Temperature Restraints

- Restraining temperature to 56C, 54C, and 52C for Wave2D application
- Each label shows the execution time penalty (in %) and the corresponding average temperature for all the processors (in C)



Timing penalty calculated based on the run where all processors run at maximum frequency

Takeaways

- By restraining processor temperature one can *select* an MTBF (within a range)
- An execution time penalty associated with temperature restraint for *selecting* the MTBF
- Different applications would:
  - have different temperature profiles
  - result in different MTBF
  - have different execution time penalty for temperature control

Any Benefits of Temperature Restraint?

$$T = T_{Solve} + T_{Checkpoint} + T_{Recover} + T_{Restart}$$

- Execution time (T): sum of useful work, check pointing time, recovery time and restart time
- Temperature restraint:
  - decreases MTBF which in turn decreases check pointing, recovery, and restart times
  - increases time taken by useful work

Performance Model

Symbol	Description
T	Total execution time
W	Useful work
τ	Check point period
δ	check point time
R	Restart time
μ	slowdown

$$T = T_{Solve} + T_{Checkpoint} + T_{Recover} + T_{Restart}$$

$$T = W + \left(\frac{W}{\tau} - 1\right)\delta + \frac{T}{M}\left(\frac{\tau + \delta}{2}\right) + \frac{T}{M}R$$

$$T = W\mu + \left(\frac{W\mu}{\tau} - 1\right)\delta + \frac{T}{M}\left(\frac{\tau + \delta}{2}\right) + \frac{T}{M}R$$

Model Validation

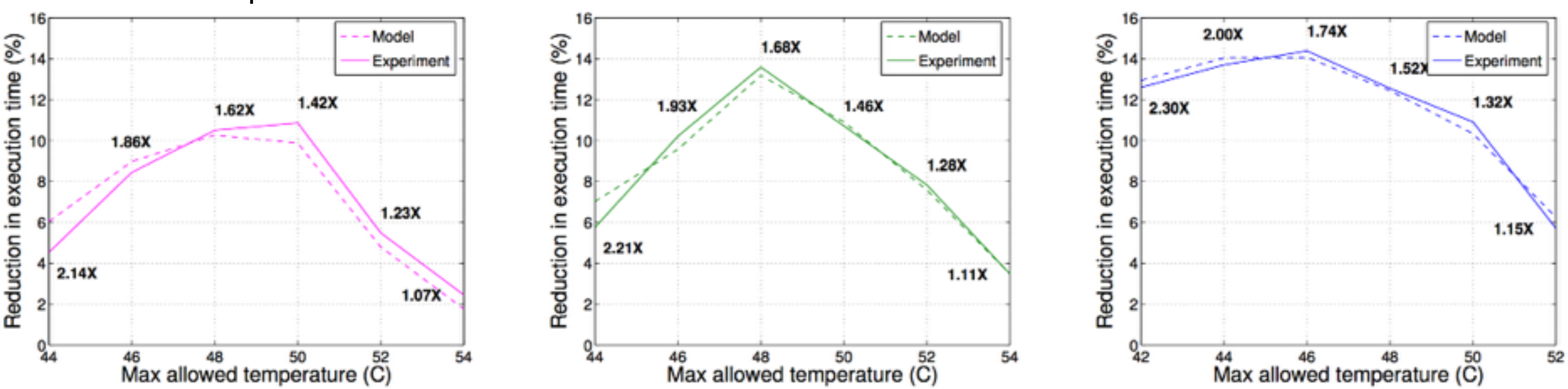
- Experiments on a 32-node cluster that supports DVFS (1.2 Ghz - 2.4 Ghz)
- To emulate the number of failures in a 700K socket machine, we utilize a scaled down value of MTBF (4 hours per socket)
- Inject random faults based on estimated MTBF values using ‘kill -9’ command
- Three applications:
  - Jacobi2D: 5 point-stencil
  - LULESH: Livermore Unstructured Lagrangian Explicit Shock Hydrodynamics
  - Wave2D: finite difference for pressure propagation

Model Validation

- Baseline experiments:
  - Without temperature restraint
  - MTBF based on actual temperature data from experiment
- Temperature restrained experiments:
  - MTBF calculated using the max allowed temperature
- τ calculated using Daly's formula $\tau = \sqrt{2\delta M} - \delta$

Reduction in Execution Time

- Each experiment was longer than 1 hour having at least 40 faults
- Execution time includes useful work, checkpointing, restart and recovery times
- Inverted-U curve points towards a tradeoff between timing penalty and improvement in MTBF



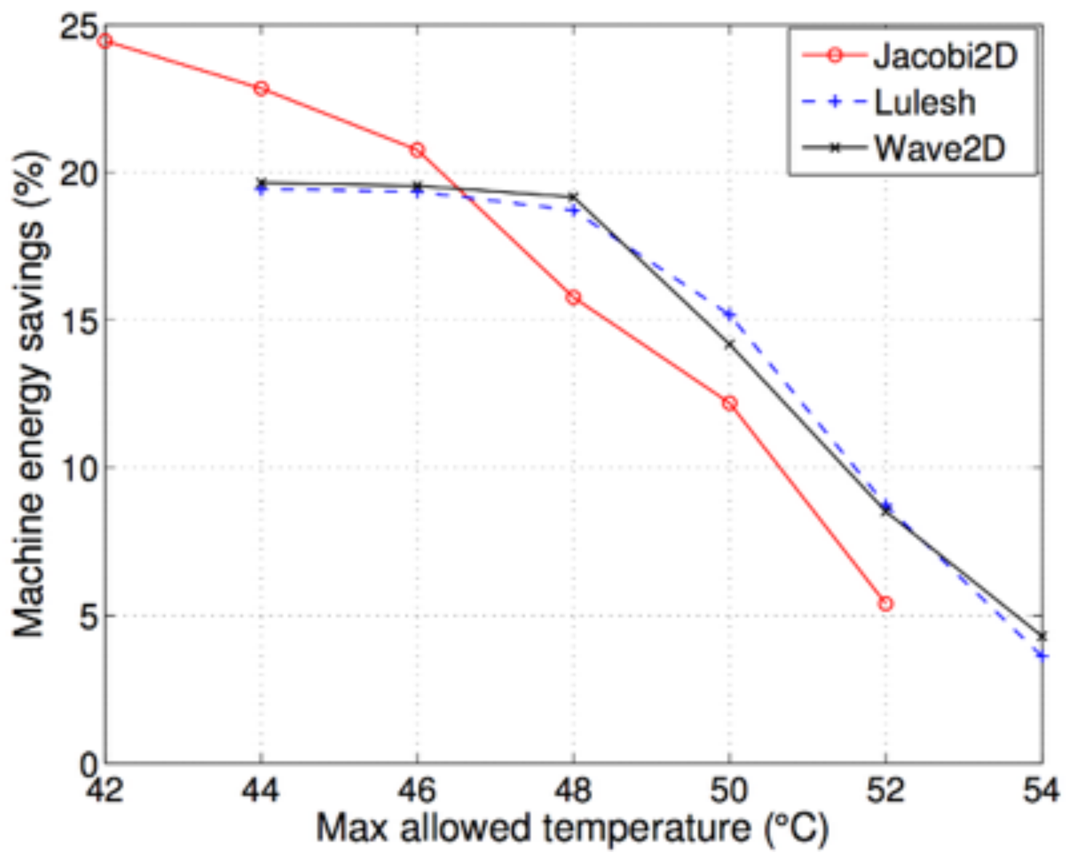
(a) Lulesh (b) Wave2D (c) Jacobi2D

Reduction in time calculated compared to baseline case with no temperature control

Savings in Machine Energy Consumption

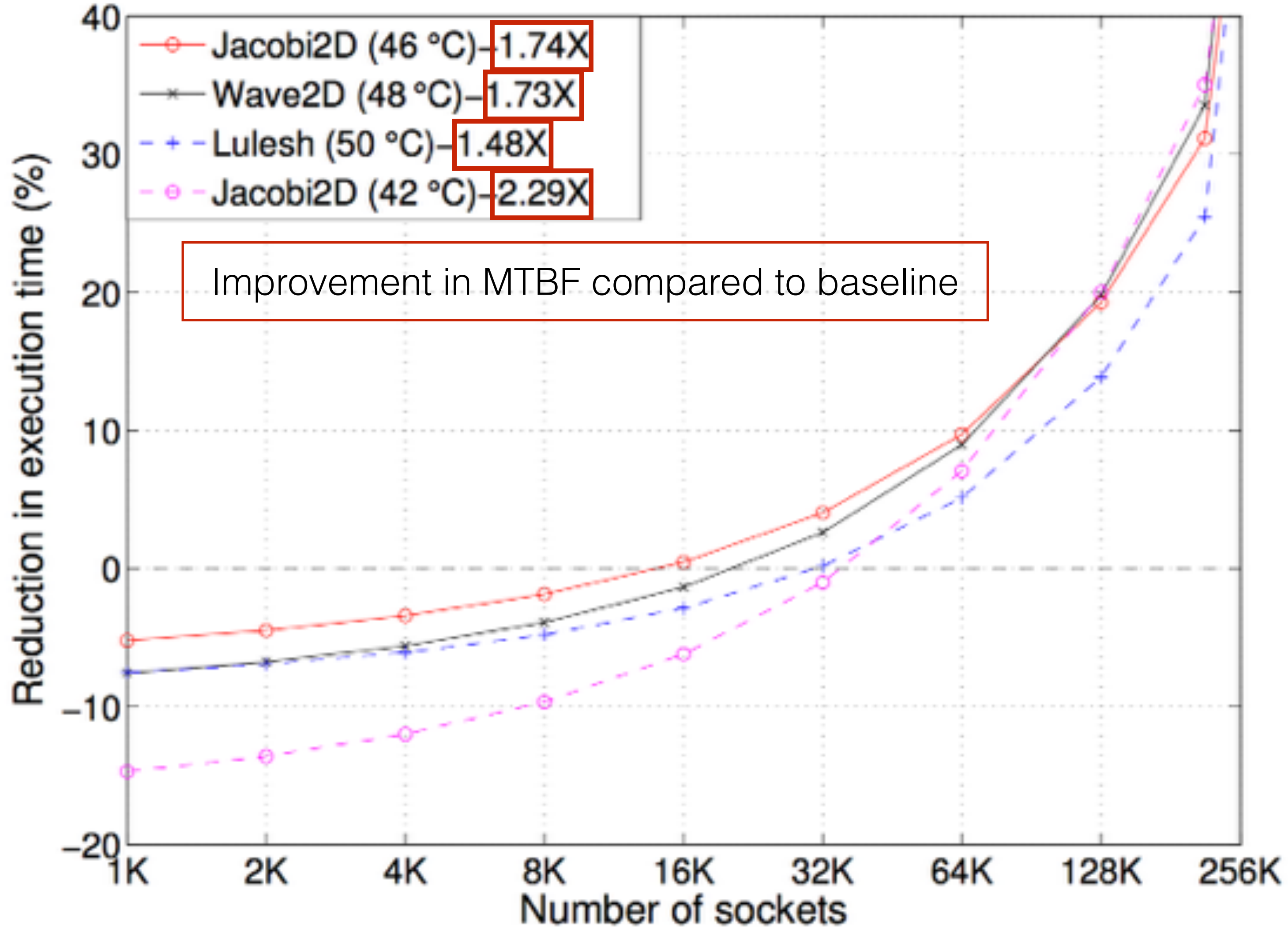
- Actual measurements based on power meters installed in the PDU
- Reasons for energy reduction:
  - Lower power consumption due to DVFS
  - Reduction in execution time due to improved MTBF

Machine energy savings calculated compared to the case with no temperature control



Predictions for Larger Machines

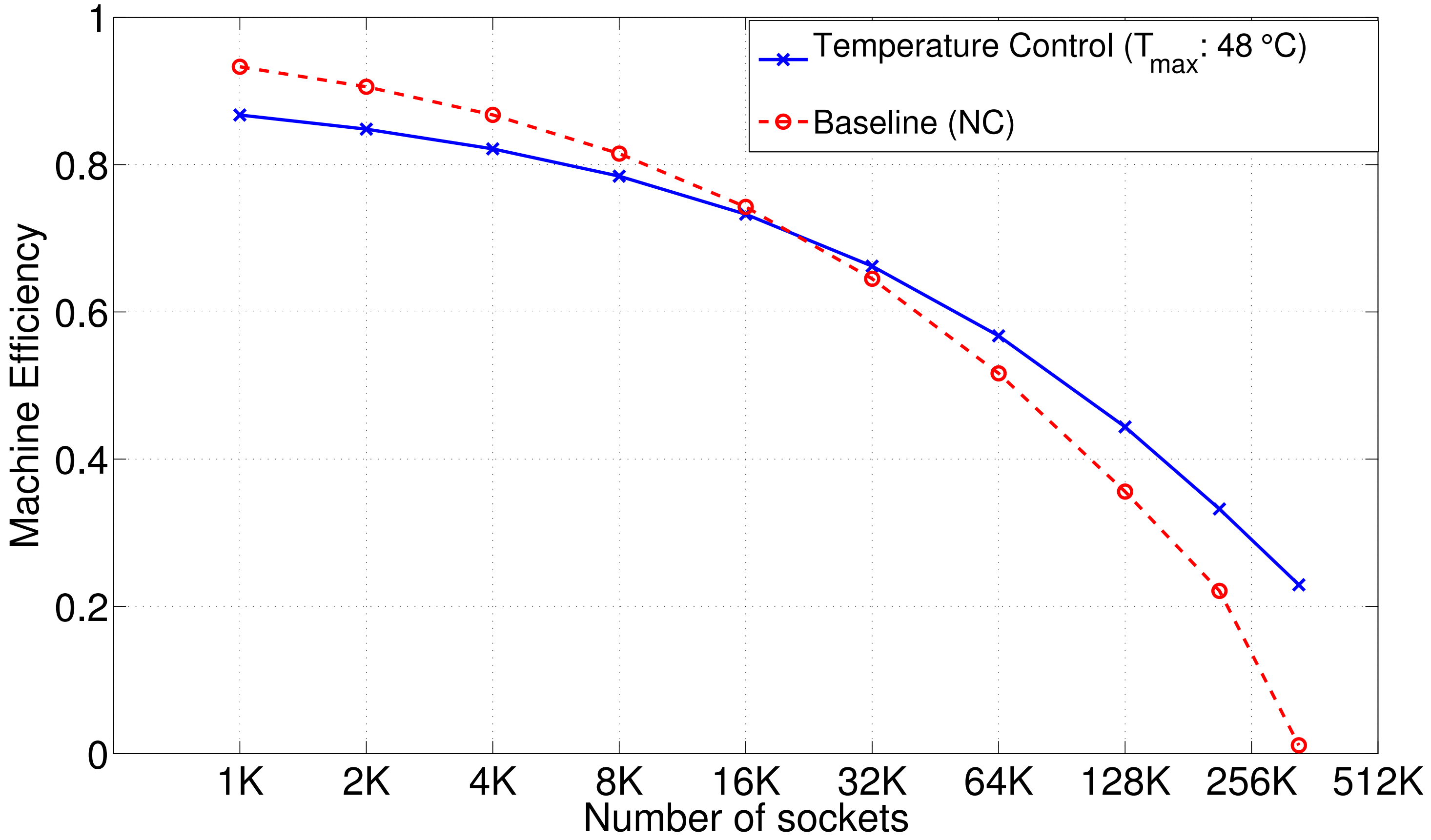
- Per-socket MTBF of 10 years
- Optimum temperature thresholds



Improvement in Machine Efficiency

- Our scheme improves utilization beyond 20K sockets compared to baseline
- For 340K socket machine:
  - Baseline: Efficiency < 1% (un operational)
  - Our scheme: Efficiency ~ 21%

Machine Efficiency: Ratio of time spent doing useful work when running a single application



Summary

- Restraining processor temperature
  - improves MTBF
  - increases time to do useful work
- Our scheme reduces execution time significantly compared to baseline for larger machines (>100K sockets)
- Our scheme can enable a 350K socket machine to operate with 21% efficiency even with current MTBF numbers

Future Work

- Evaluating the benefits of temperature restraint for other fault tolerance protocols e.g., message logging and parallel recovery
- Estimate the impact of thermal throttling on MTBF of the entire machine

- Questions
- Defending my PhD dissertation and looking for jobs
  - email: osmansarood@gmail.com
  - cell: 217-819-9492
- Other Charm++ events: http://charm.cs.uiuc.edu/sc13