# Temperature Aware Load Balancing For Parallel Applications

Osman Sarood
Parallel Programming Lab (PPL)
University of Illinois Urbana Champaign

# Why Energy?

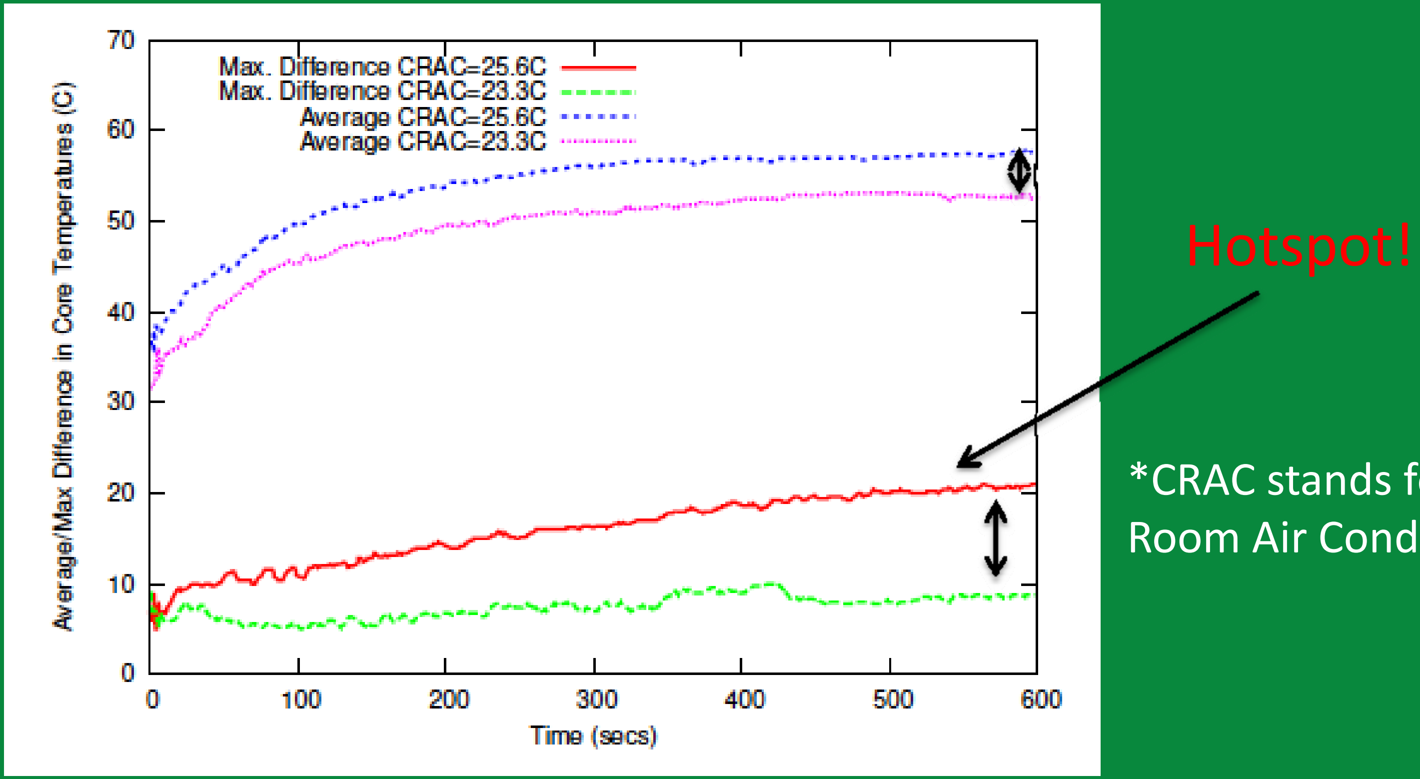
- Data centers consume 2% of US Energy Budget in 2006
- Costed $4.1 billion consumed 59 billion KWh
- The 3-year cost of powering and cooling servers exceeds the cost of purchasing the server hardware
- 2.5X system level power efficiency improvement in last three years (100X needed for exascale)

# Why Cooling?

- Cooling accounts for 50% of total cost
- Most data centers face HotSpots responsible for lower temperatures in machine rooms
- Data center managers can save*:
  - 4% (7%) for every degree F (C)
  - 50% going from 68F(20C )to 80F(26.6C)
- Room temperatures can be increased provided:
  - No Hotposts
  - Cores temperatures don’t get too high

*according to Mark Monroe of Sun Microsystem

# Core Temperatures



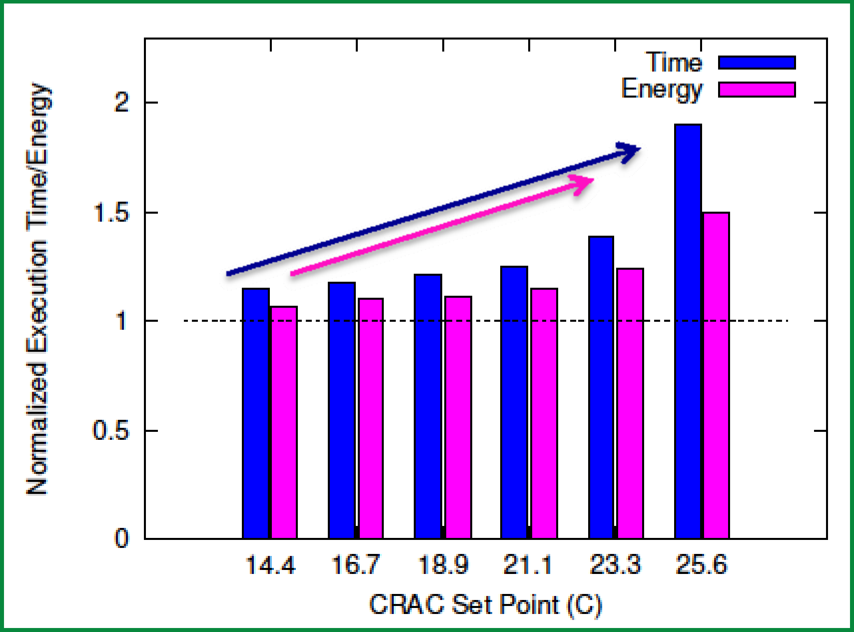

*CRAC stands for Computer Room Air Conditioning

- Reducing cooling results for Wave2D:
  – Difference of 6C in average temperature
  – Difference of 12C in deviation from average

# Constraining Core Temperatures using DVFS

- Periodic check on core temperatures
- Timing penalty grows with a decrease in cooling
- Machine energy increases as well!
- Not useful due to tightly coupled nature of applications

Normalization w.r.t all cores running at maximum frequency without temperature control



# Temperature Aware Load Balancer

- Specify temperature threshold and sampling interval
- Runtime system periodically checks core temperatures
- Scale down/up if temperature exceeds/below maximum threshold at each decision time
- Transfer tasks from slow cores to faster ones

# Charm++

- Object-based over-decomposition
  - Helpful for refinement load balancing
- Migrateable objects
  - Mandatory for our scheme to work
- Time logging for all objects
  - Central to load balancing decisions

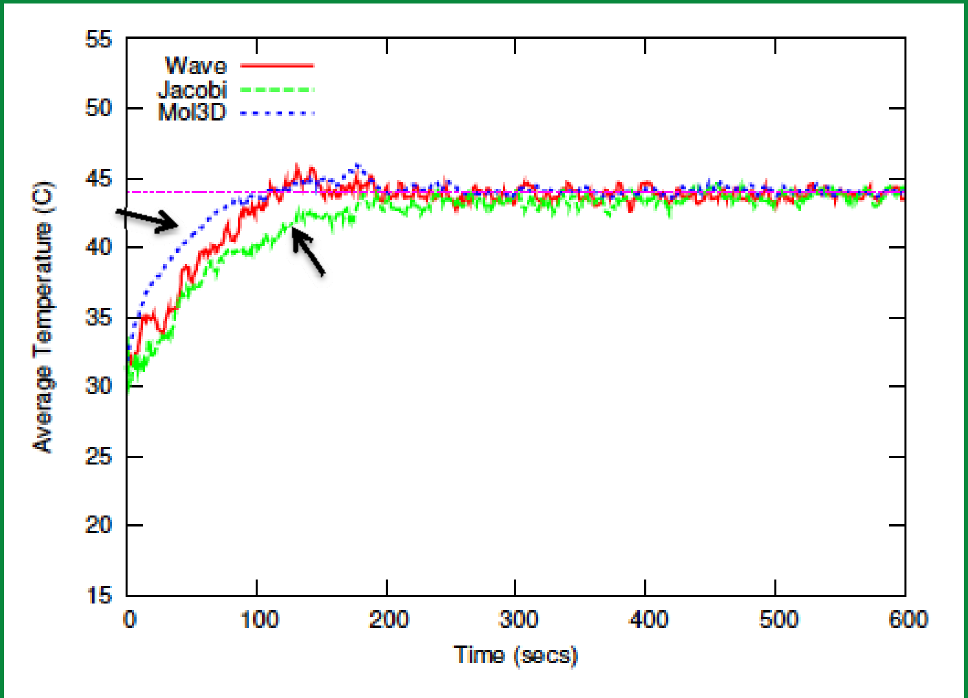
# Experimental Setup

- 128 cores (32 nodes), 10 different frequency levels (1.2GHz – 2.4GHz)
- Direct power measurement
- Dedicated CRAC
- Power estimation based on

$$P_{ac} = f_{ac} * c_{air} * (T_{hot} + T_{ac})$$

- Applications: Jacobi2D, Mol3D, and Wave2D
  - Different power profiles
- Max threshold: 44C

# Average Core Temperatures in Check

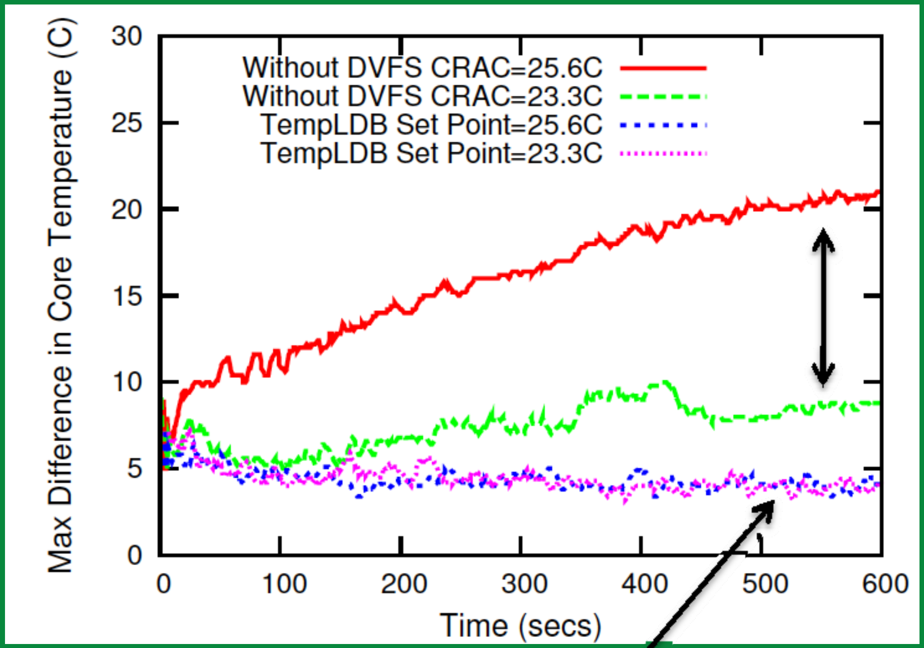



- Avg. core temperature within 1-2 C of threshold
- Can handle applications having different temperature gradients

# Hotspot Avoidance

- Without our scheme max. difference:
  - Increases over time
  - Increases with CRAC set point
- With our scheme
  - Max. temperature decreases with time
  - Insensitive to CRAC set point
- Our scheme avoids Hotspots

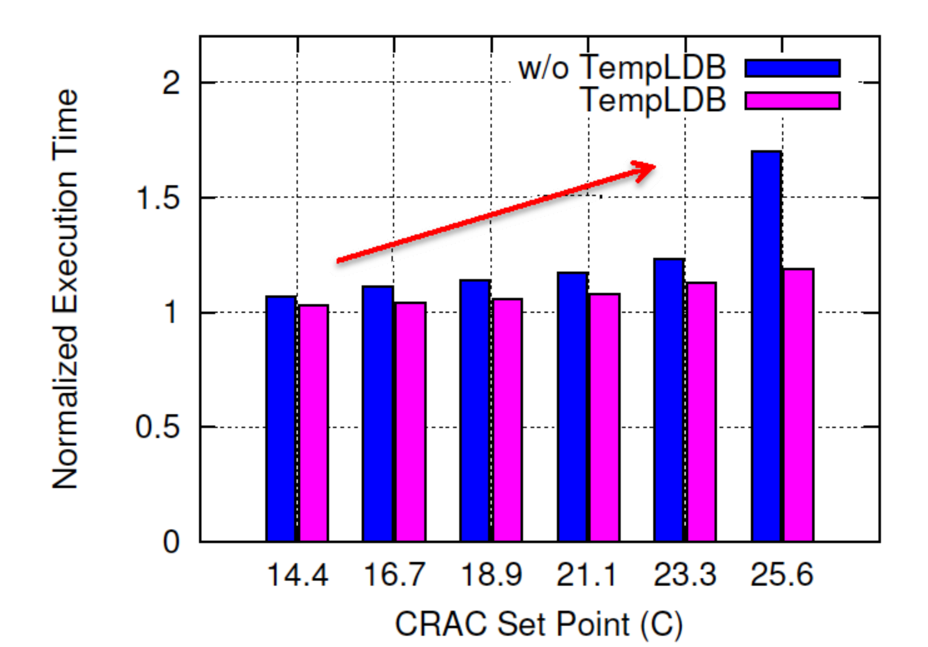
Wave2D on 128 Cores




**Hot Spots Avoided!**

# Timing Penalty

Jacobi2D on 128 Core



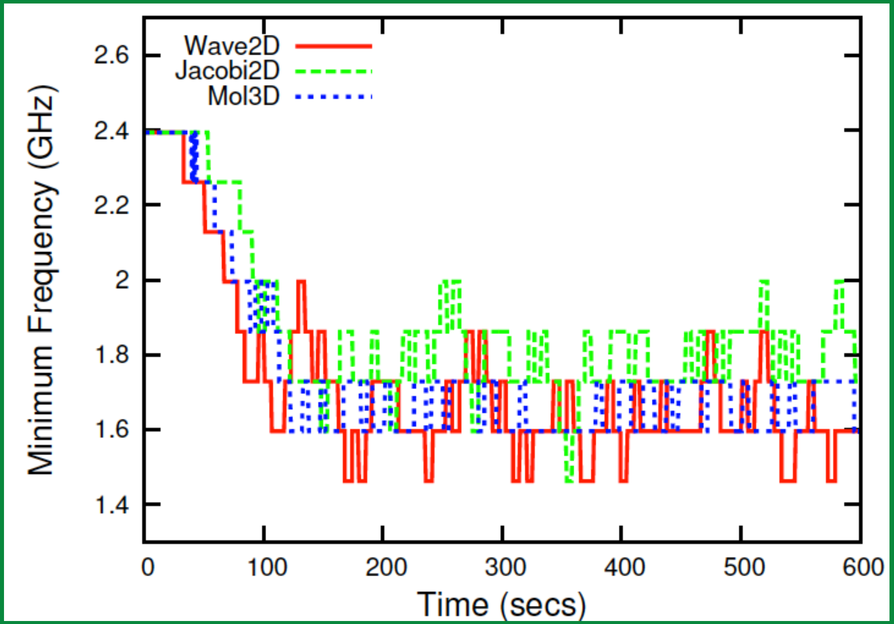

- Our load balancer performs better
- Decrease in cooling, increases:
  - Timing penalty
  - Advantage of our scheme

# Processor Timelines for Wave2D


- Shows processor utilization during execution time (green and pink correspond to computations)
- Execution time dependent on slowest core
- One core can cause timing penalty/slowdown

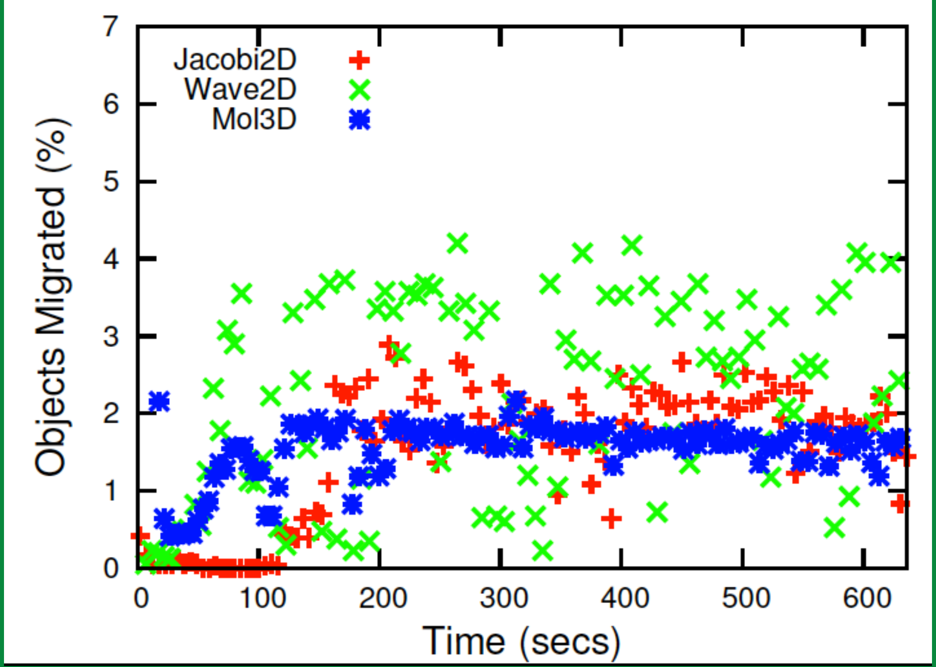
# Minimum Frequency (No TempLB)

- Frequency of slowest core for (CRAC 23.3C)
- Wave2D and Mol3D
  - Lower minimum frequencies
  - Higher timing penalties




| Application | Time Penalty(%) |
|---|---|
| Wave | 38 |
| Mol3D | 28 |
| Jacobi2D | 23 |

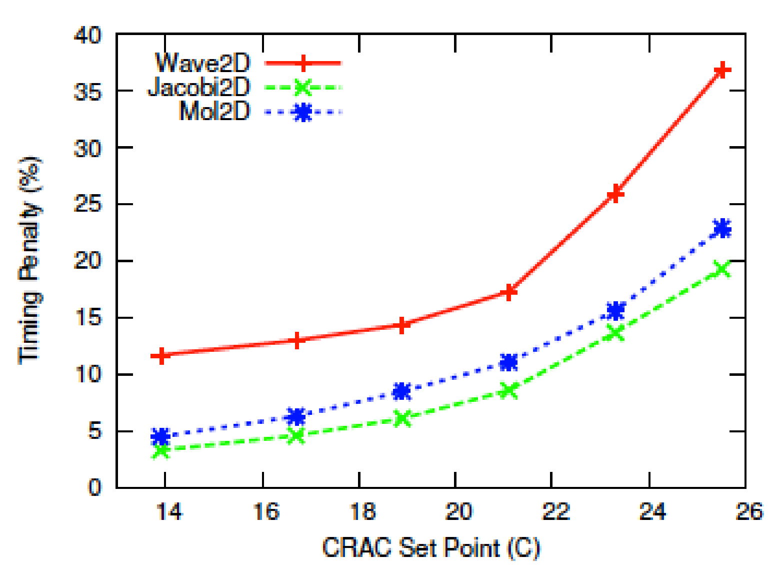
# Timing Overhead




- Dependent on:
  - How frequently temperatures checked
  - How many migrations
- Wave2D has the highest migration percentage

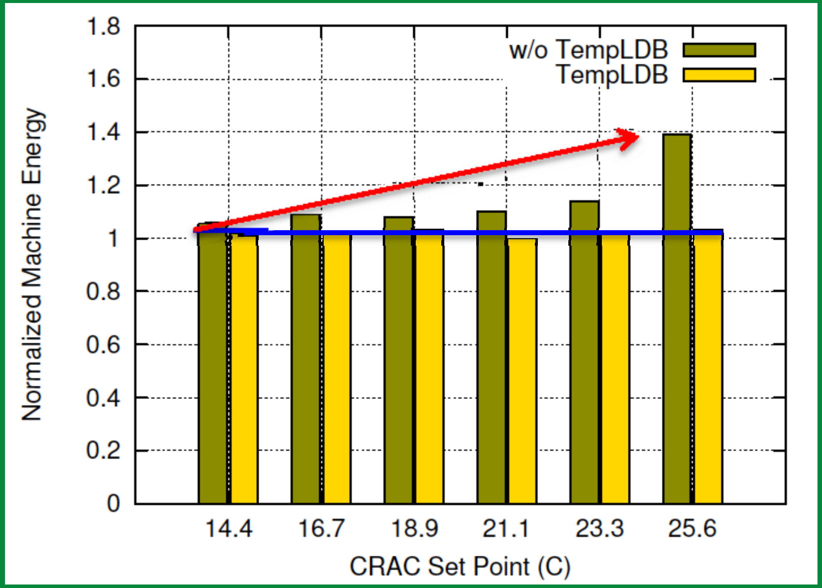
# Timing Penalty and CRAC Set Point

- Slope: timing penalty (secs) per 1C increase in CRAC set point
- Correlation between Timing penalty and MFLOP/s




| Application | MFLOP/s |
|---|---|
| Wave | 292 |
| Mol3D | 252 |
| Jacobi2D | 240 |

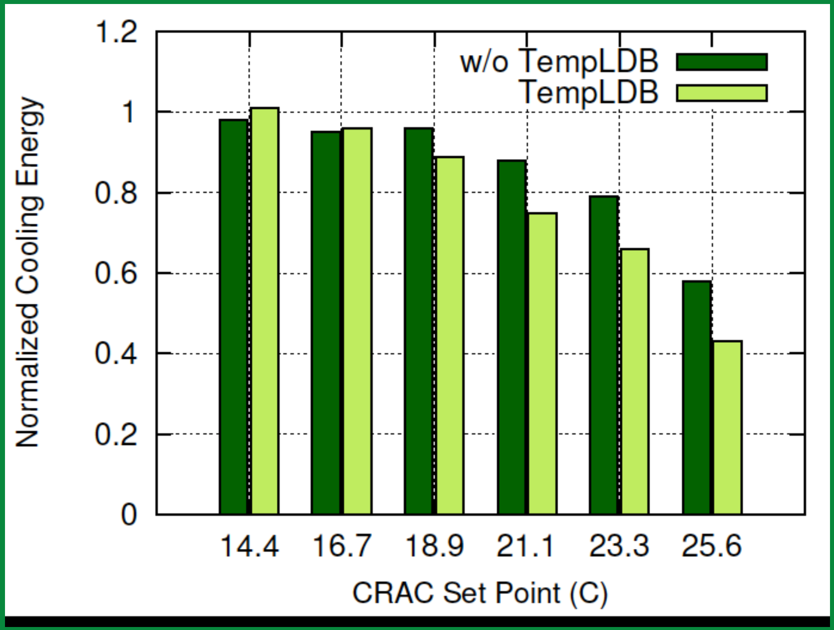
# Machine Energy Consumption




Mol3D on 128 Cores

- Our scheme consistently saves machine power in comparison to `w/o TempLB'.
- High idle power coupled with timing penalty doesn't allow machine energy savings.

# Cooling Energy Consumption

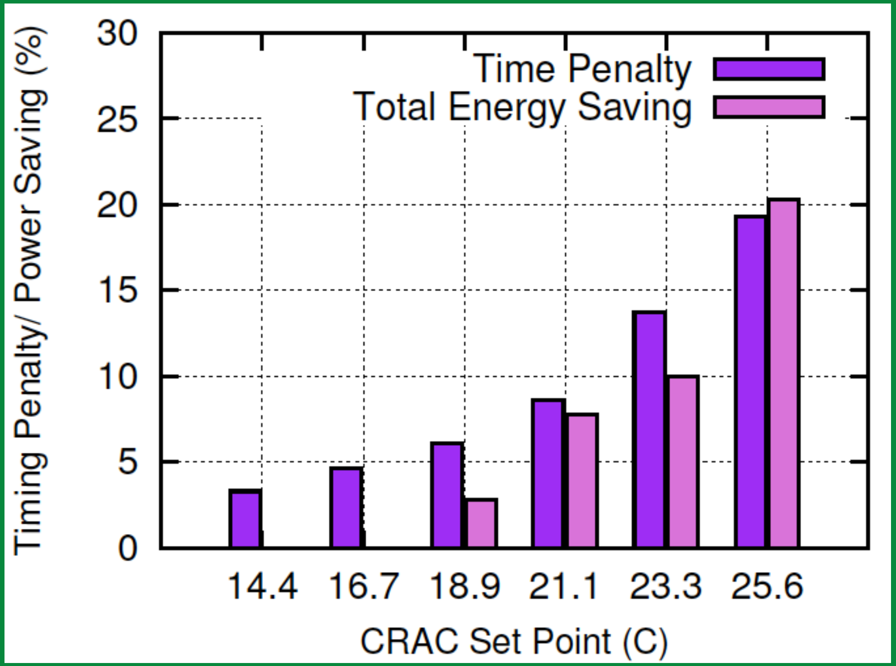



$$c_{norm} = \frac{T_{hot}^{LB} - T_{ac}^{LB}}{T_{hot}^{base} - T_{ac}^{base}} * t_{norm}^{LB}$$

Jacobi2D on 128 Cores

- Both schemes save energy (TempLDB better)
- Our scheme saves upto 57%

# Timing Penalty/ Total Energy Savings




Jacobi2D on 128 Cores

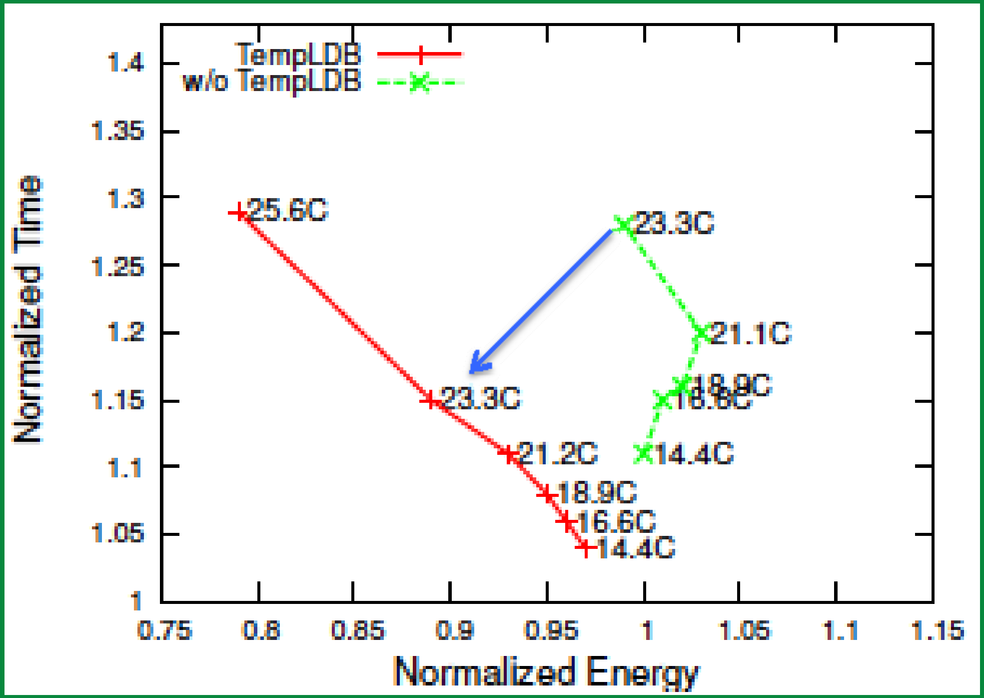
- Mol3D and Jacobi2D show good energy savings
- Wave2D not appropriate for energy savings?

# Temperature range instead of Threshold

- Temperature Range: 44C – 49C
- Scale down if core temperature > upper limit
- Scale up if core temperature < lower limit

| CRAC Set Point | Timing Penalty: Range (%) | Timing Penalty: Threshold(%) | Power Saving: Range (%) | Power Saving: Threshold (%) |
|---|---|---|---|---|
| 23.3 | 3 | 15 | 19 | 11 |
| 25.6 | 12 | 22 | 23 | 20 |

# Energy Vs Execution Time




Mol3D on 128 Cores

Normalization w.r.t all cores running at maximum frequency without temperature control

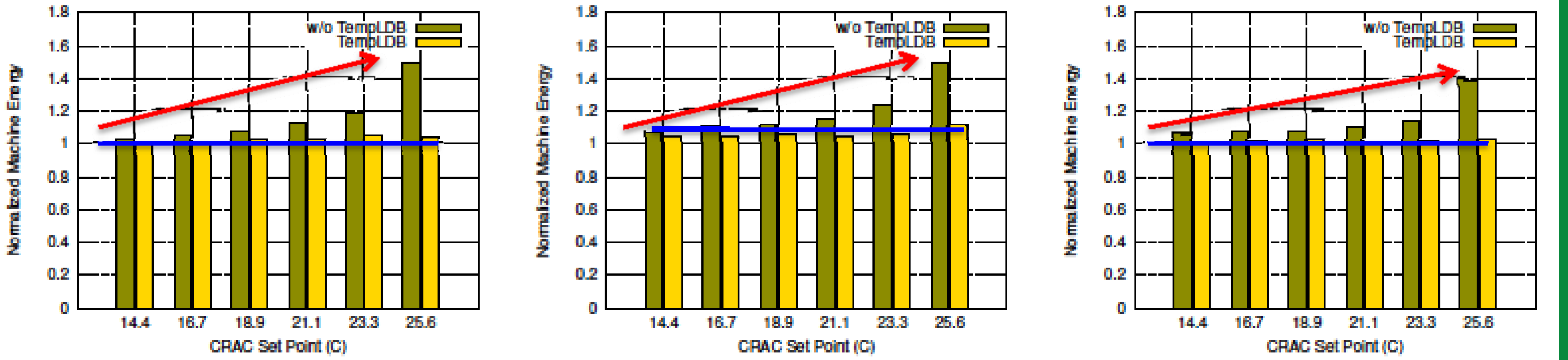
- Our scheme brings green line to red line
  - Moving left: saving total energy
  - Moving down: saving execution time penalty
- Slope: timing penalty (secs) per joule saved in energy

# Contributions

- Stabilizing core temperatures
- Avoiding Hotspot
- Minimize timing penalty/ slowdown
- Minimize Cooling costs
  - Saved 48% cooling moving from 18.9C – 25.6C

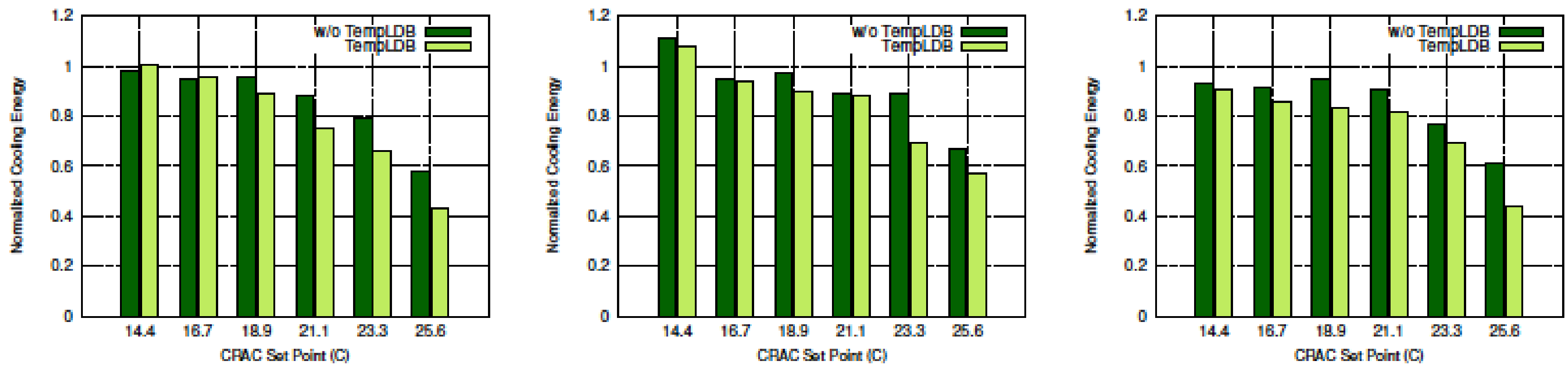
# Questions

# Machine Energy Consumption



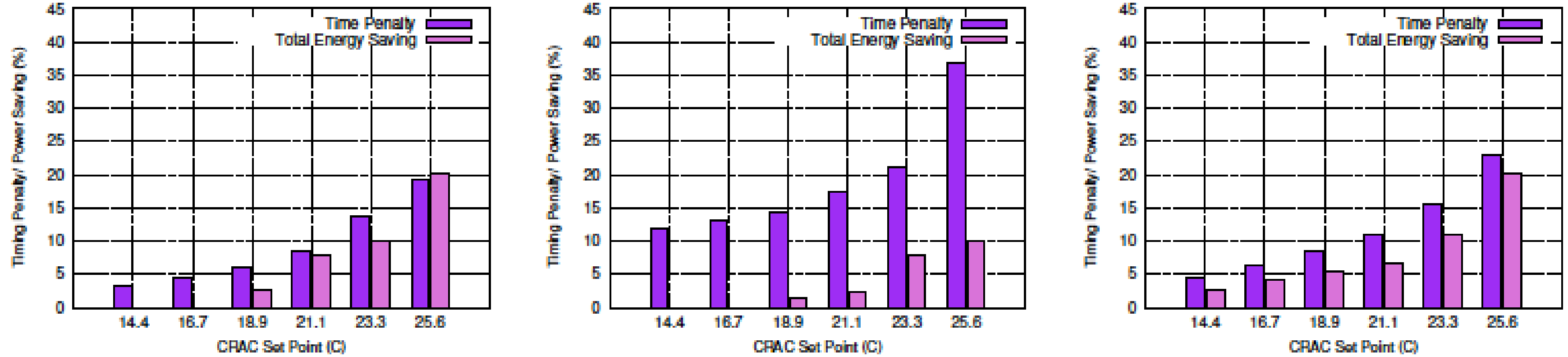

(a) Jacobi2D (b) Wave2D (c) Mol3D

# Cooling Energy Savings



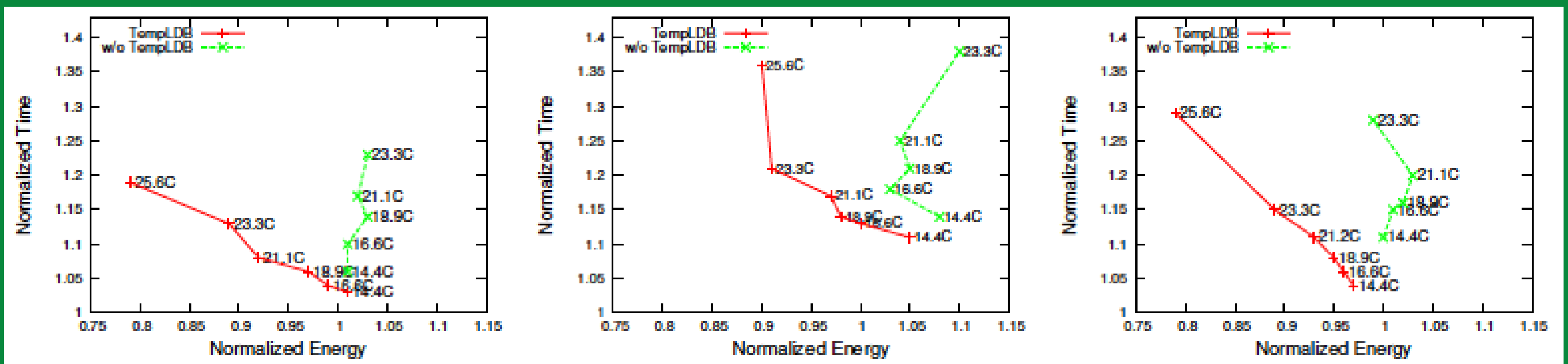

(a) Jacobi2D (b) Wave2D (c) Mol3D

# Timing Penalty/ Total Energy Savings




(a) Jacobi2D (b) Wave2D (c) Mol3D

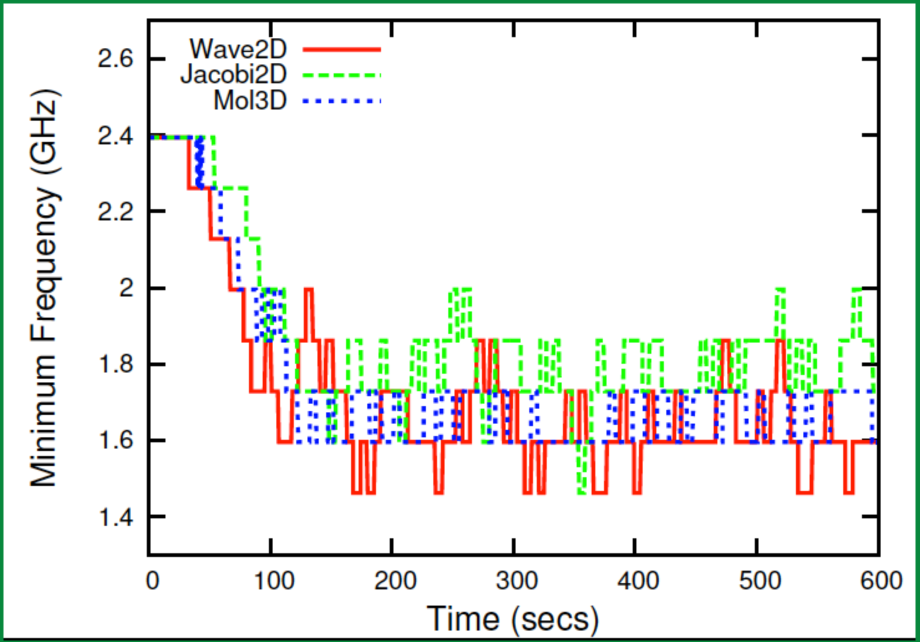
# Energy Vs Execution Time




(a) Jacobi2D (b) Wave2D (c) Mol3D

# Average Frequency

- Mol3D's average frequency lower than Jacobi even with less power/CPU utilization
- High total power for Jacobi
  - Greater number of DRAM accesses
  - Overall large memory footprint
- High MFLOP/s for Mol3D considering low CPU utilization
  - Data readily available in L1+L2




Performance counters for one core

| Counter Type | Jacobi2D | Mol3D | Wave2D |
|---|---|---|---|
| Execution Time (secs) | 474 | 473 | 469 |
| MFLOP/s | 240 | 252 | 292 |
| Traffic L1-L2 (MB/s) | 995 | 10,500 | 3,044 |
| Traffic L2-DRAM (MB/s) | 539 | 97 | 577 |
| Cache misses to DRAM (billions) | 4 | 0.72 | 4.22 |
| CPU Utilization (%) | 87 | 83 | 93 |
| Power (W) | 2472 | 2353 | 2558 |
| Memory Footprint(% of memory) | 8.1 | 2.4 | 8.0 |

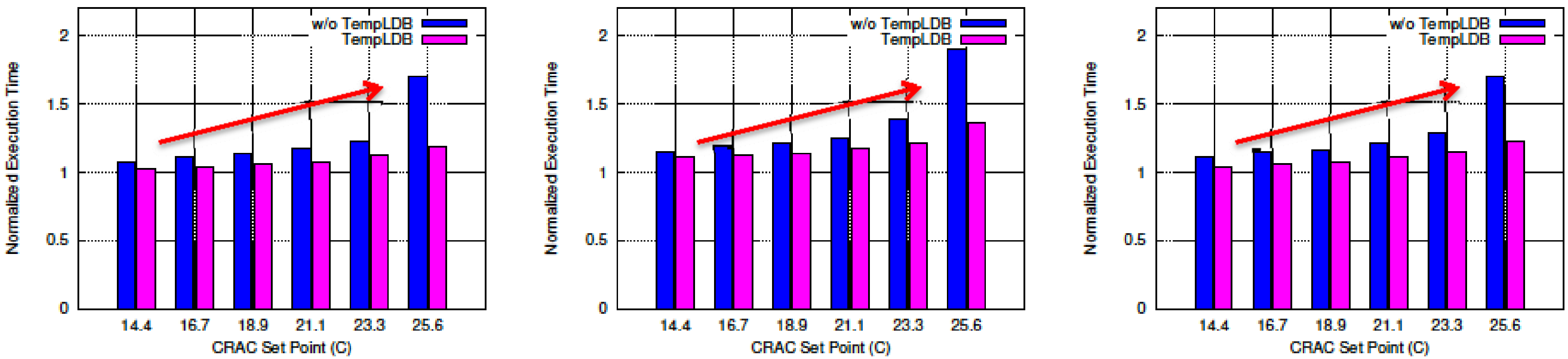
# Cooling Energy Savings

| | Mol3D | | Jacobi | | Wave2D | |
|---|---|---|---|---|---|---|
| | TempLB | No TempLB | TempLB | No TempLB | TempLB | No TempLB |
| 57 | 0.83008303 | 0.92662993 | 0.9535189 | 0.9825494 | 1.08436118 | 1.10835359 |
| 62 | 0.91371415 | 0.91630064 | 1.02676076 | 0.94843724 | 0.94017829 | 0.94566919 |
| 66 | 0.87636352 | 0.95424306 | 0.89537819 | 0.95016447 | 0.89775714 | 0.9712767 |
| 70 | 0.84932621 | 0.90341406 | 0.75526485 | 0.87756626 | 0.87606527 | 0.88519665 |
| 74 | 0.69208677 | 0.76990626 | 0.66527495 | 0.79335206 | 0.71963512 | 0.89981607 |

# Timing Penalty

| | Mol3D | | Jacobi | | Wave2D | |
|---|---|---|---|---|---|---|
| 57 | TempLDB | w/o TempLDB | TempLDB | w/o TempLDB | TempLDB | w/o TempLDB |
| 57 | 1.04 | 1.11 | 1.03 | 1.06 | 1.11 | 1.14 |
| 62 | 1.06 | 1.15 | 1.04 | 1.10 | 1.13 | 1.18 |
| 66 | 1.08 | 1.16 | 1.06 | 1.14 | 1.14 | 1.21 |
| 70 | 1.11 | 1.20 | 1.08 | 1.17 | 1.17 | 1.25 |
| 74 | 1.15 | 1.28 | 1.13 | 1.23 | 1.26 | 1.38 |
| 78 | 1.22 | 1.70 | 1.19 | 1.80 | 1.36 | 1.90 |

# Timing Penalty




(a) Jacobi2D (b) Wave2D (c) Mol3D