Lin Li, Xiuyi Zhou, Jun Yang, Victor Puchkarev
University of Pittsburgh

THRESHOT: AN AGGRESSIVE TASK SCHEDULING APPROACH IN CMP THERMAL DESIGN

Outline

- Introduction
- ThresHot Algorithm
- Experiment and Results
- Conclusion

Thermal Management is Critical

- Technology $\downarrow$ $\rightarrow$ Power Density $\uparrow$
- Temperature $\uparrow$
  - Circuit performance $\downarrow$
  - Reliability $\downarrow$ $$MTTF = C \times e^{\frac{E_A}{kT}}$$
  - Thermal runaway $T \uparrow \rightarrow P_{leakage} \uparrow \rightarrow P_{total} \uparrow$
  - Packaging and cooling cost $\uparrow$

Task Scheduling Can Help

- Conventionally
  - Performance throttling, e.g. DVFS
- Our objective
  - Preserve performance – ↓ DVFS
- Rationale
  - Workloads stress processor differently in space and time
- Approach
  - Find a good schedule of workloads to keep temperature low

Task Scheduling Trade-offs

- Pros:
  - No need to change hardware
  - Flexible: scheduling algorithm can be changed in OS
- Cons:
  - Scheduling overhead
  - Lack accurate hardware details

Task Scheduling Algorithms

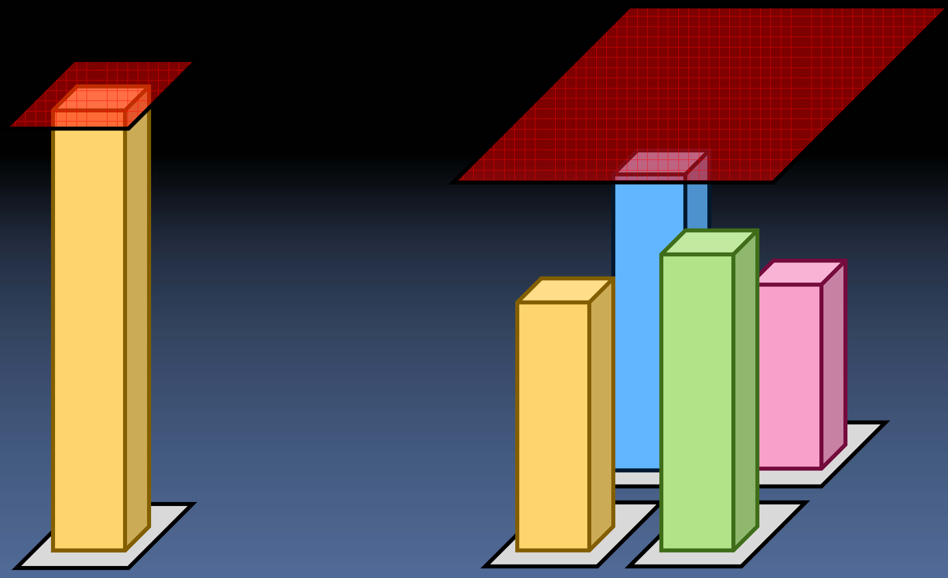
- Objective: Reduce thermal emergencies
  - Improve performance
  - Improve reliability
- Naïve scheduling algorithms
  - Random
  - Round-Robin
  - Power balancing

Outline

- Introduction
- ThresHot Algorithm
- Experiment and Results
- Conclusion

Temperature Slack

- Temporal temperature slack in a single processor
  - Task scheduling can reduce thermal emergencies [Yang et al. ISPASS 2008]
- Spatial and temporal temperature slack in CMP
  - How to schedule tasks to minimize total thermal emergencies?



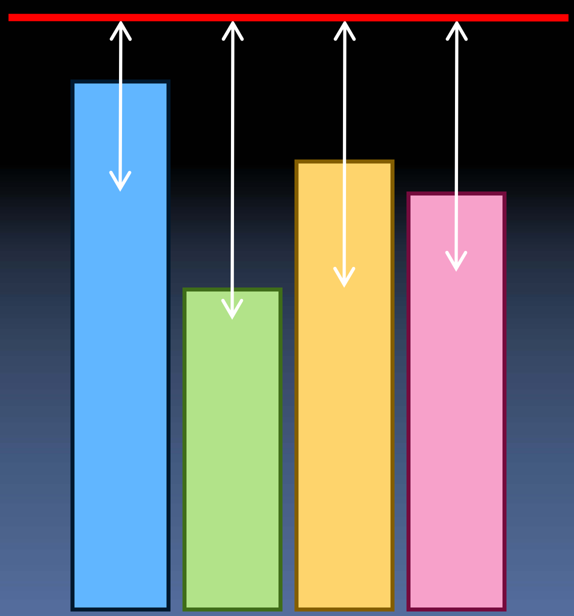
Thermal Model

- Han, Koren, Krishna, "TILTS: A Fast Architectural-Level Transient Thermal Simulation Method," *J. of Low Power Electronics*, 2007.

$$T(n) = AT(n-1) + BP(n-1)$$

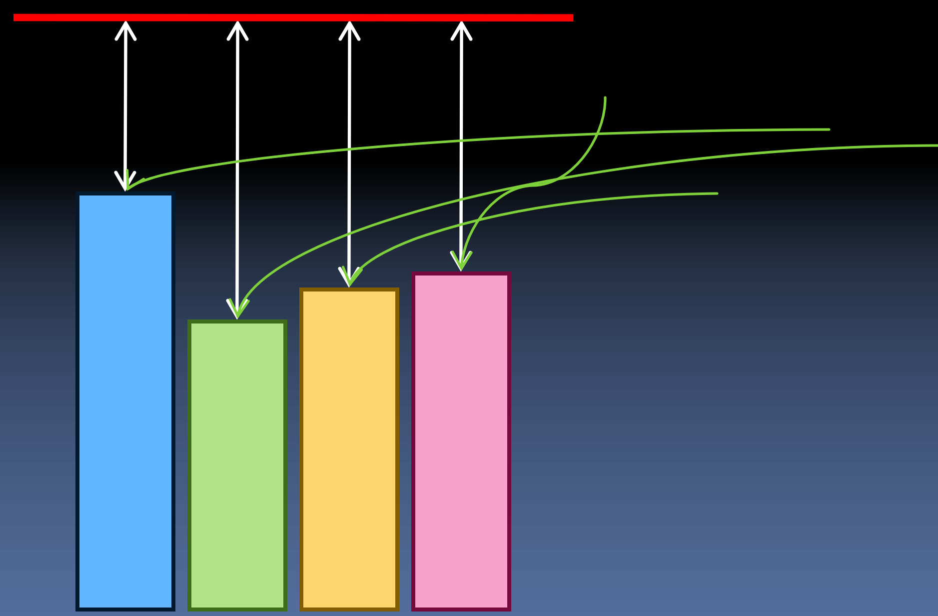
Understanding the Model 1

- AT(*n-1*) describes the temperature drop at time *n*, if there is no power
  - Available temperature slacks formed



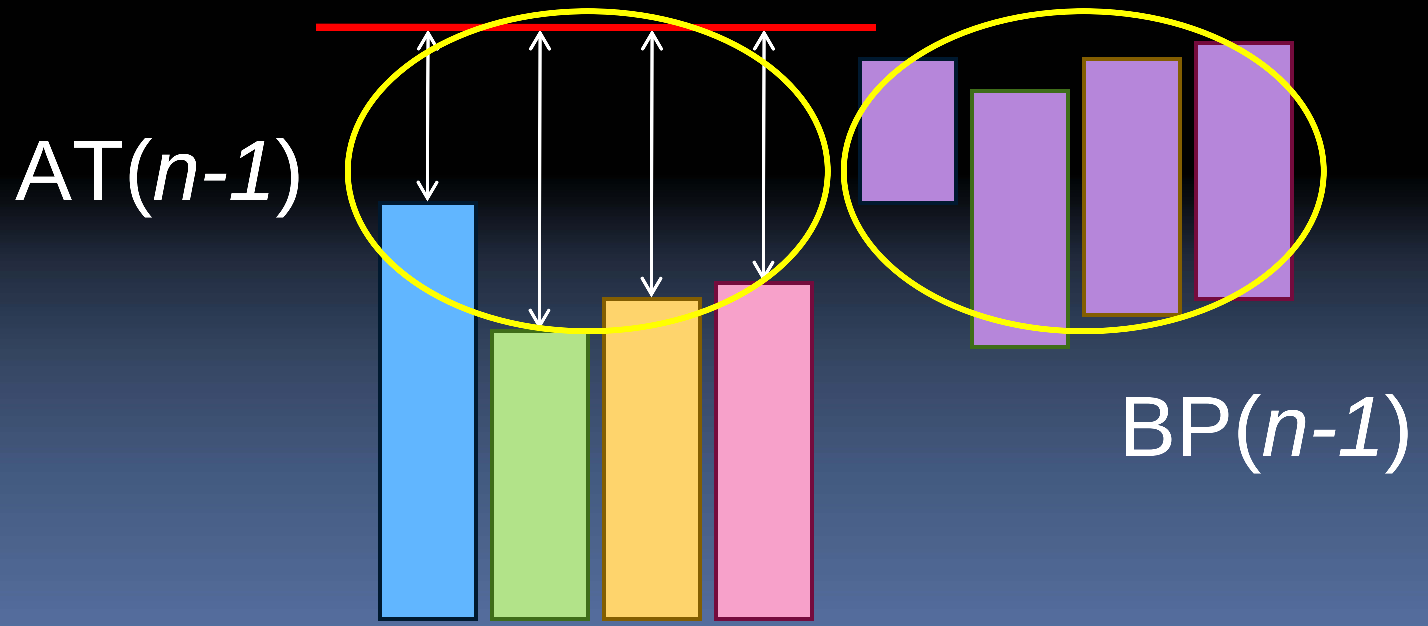
Understanding the Model 2

- BP(*n-1*) describes the temperature increase due to injected power of different tasks
- Task scheduling is to find a mapping between these increases and the thermal slacks.



Fast Temperature Calculation

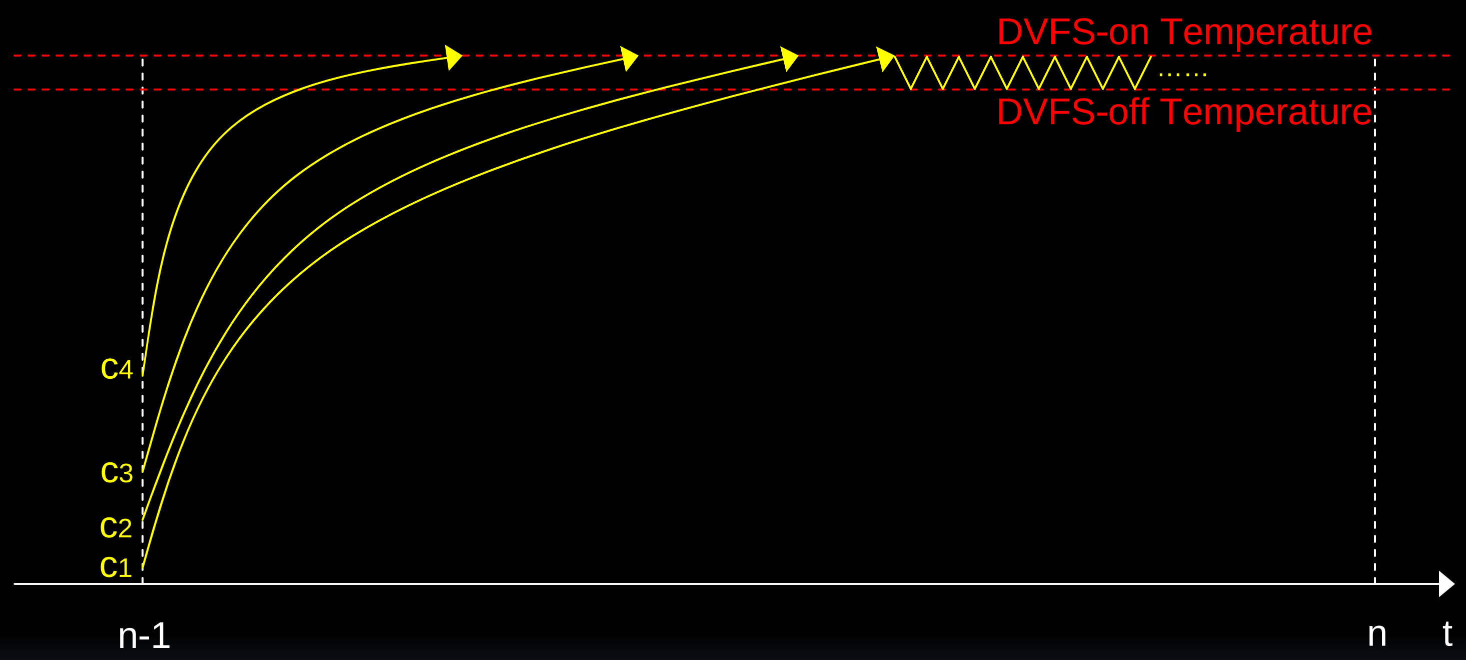
- Temperature rises due to power hardly change from core to core
- Calculate AT(n-1) and BP(n-1) once → T(n) for all possible schedule



TSM: Temperature Slack Matrix

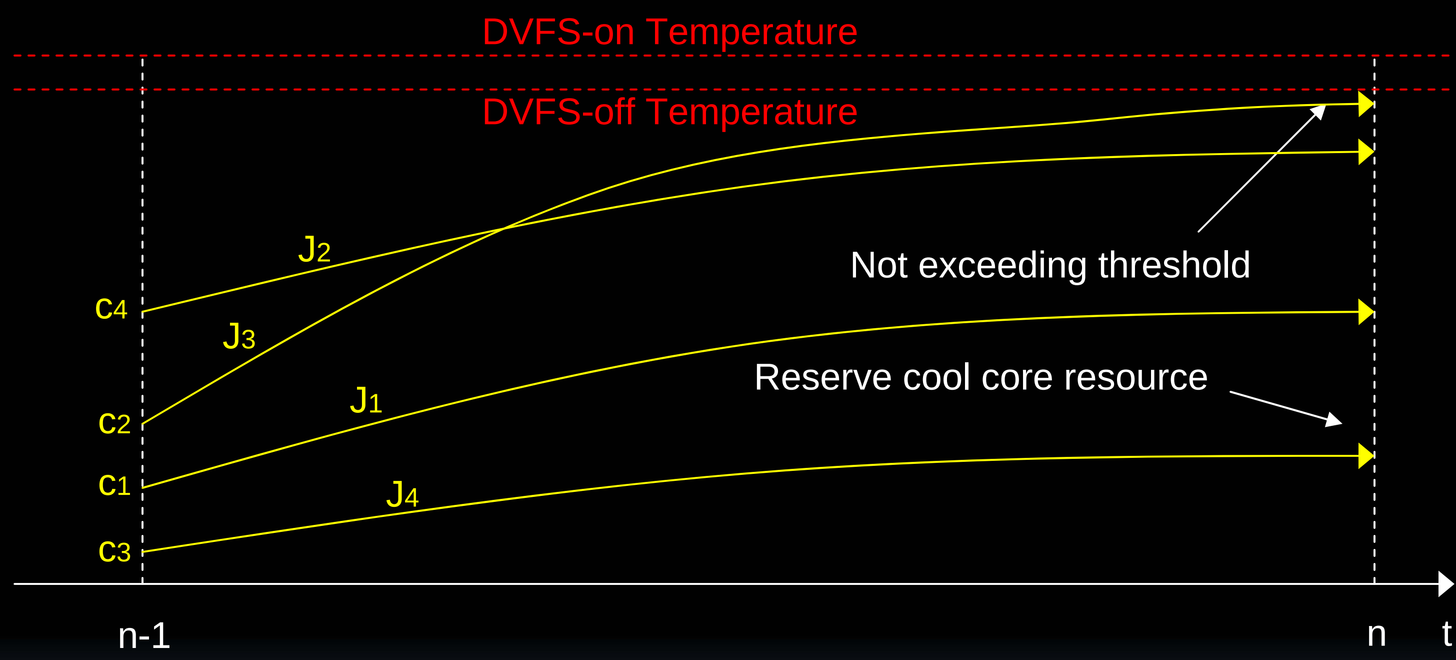
Core \ Tasks	1	2	3	4
1	+			
2			-	
3				
4				

Scheduling Hot Hazard Tasks



- Hot hazard jobs
  - Too hot even on the coolest core
  - Decision: Map it to the coolest core
    - Minimize DVFS penalty in the current scheduling cycle

Scheduling Mild Tasks



- Mild jobs
  - A schedule can be found w/o DVFS
  - Goal is not to average the temperature
  - Rather, reserve cool core resources for hot hazard tasks in the future

ThresHot Scheduling with TSM

Core \ Tasks	1	2	3	4
1	0.415	8.973	-7.617	12.322
2	0.773	9.285	-7.259	12.635
3	0.524	10.158	-7.507	16.503
4	0.857	9.407	-7.175	12.975
$\sum$	2.569	37.823	-29.558	54.435

Outline

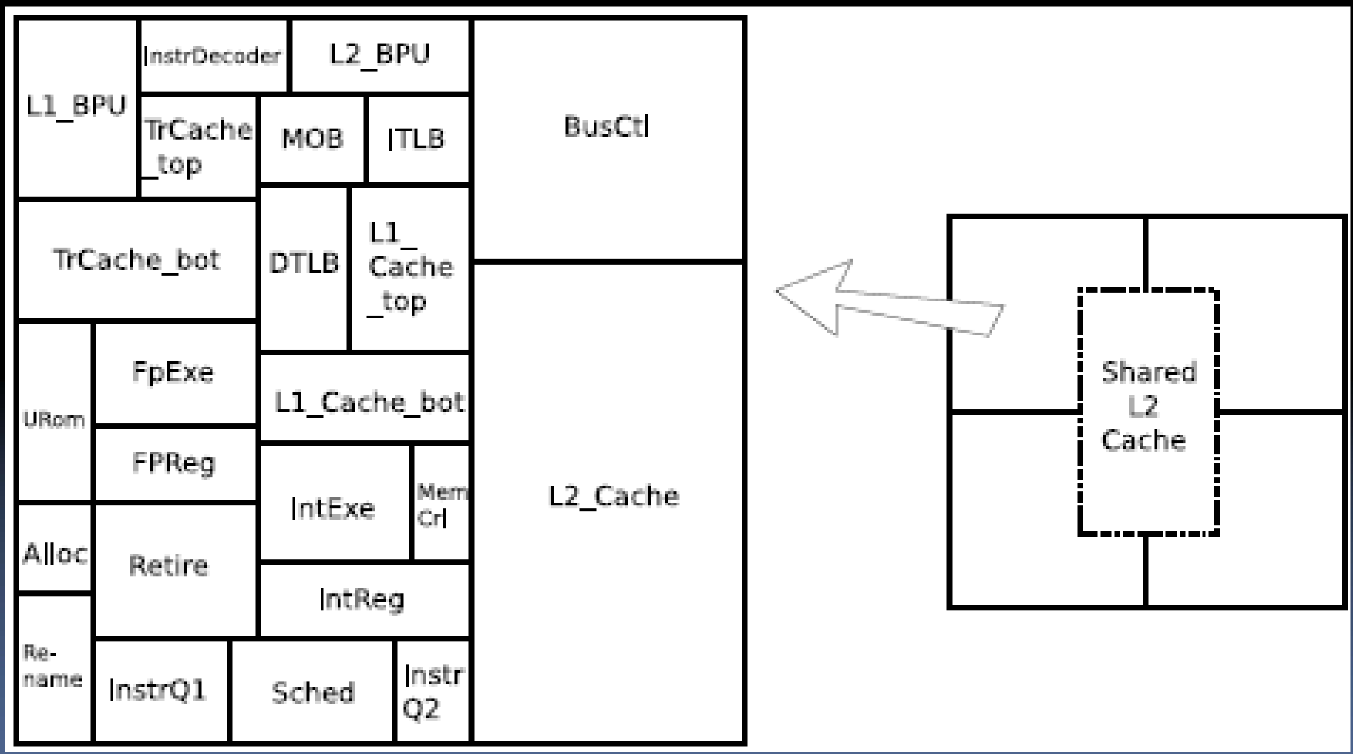
- Introduction
- ThresHot Algorithm
- Experiment and Results
- Conclusion

Experiment Methodology

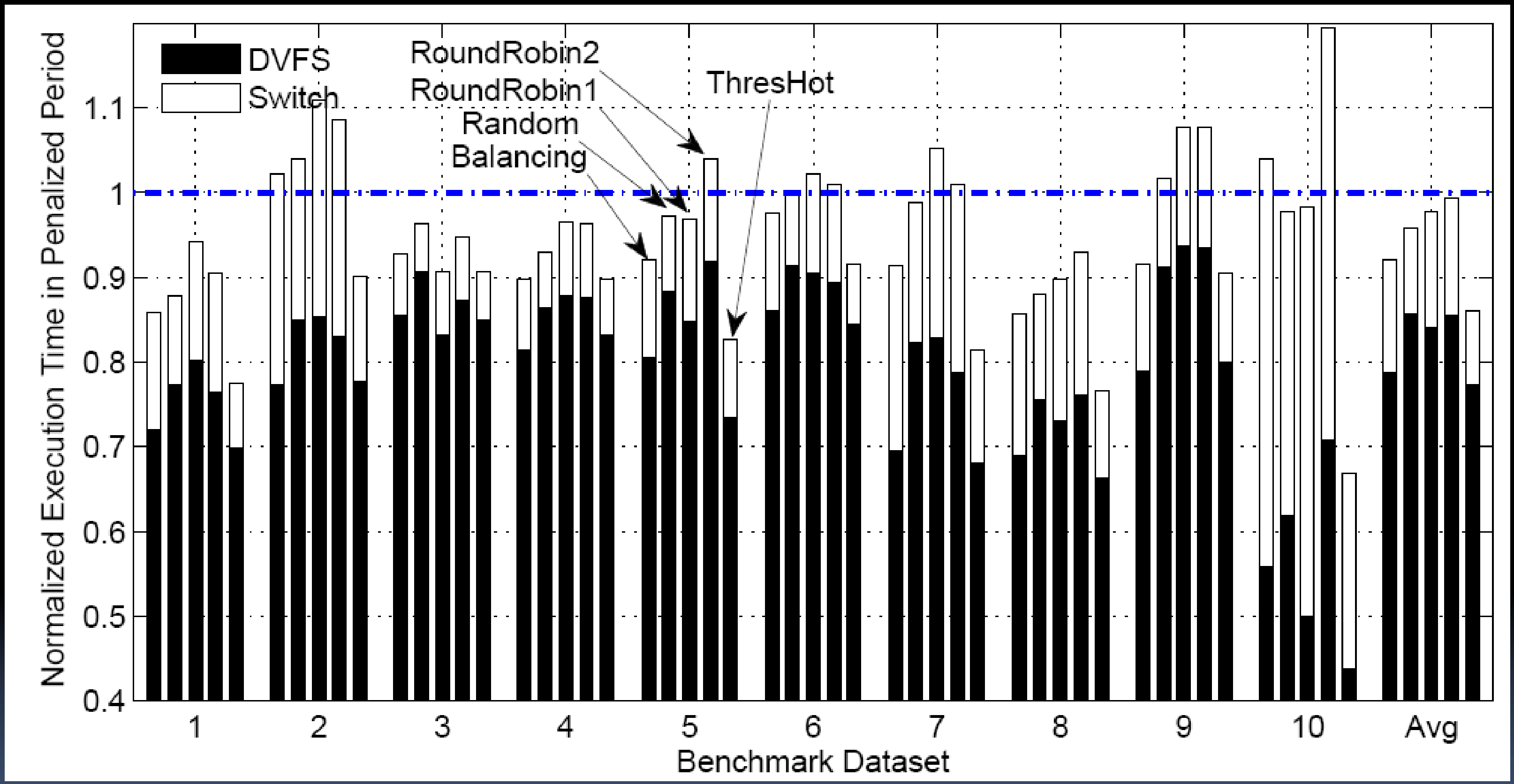
- Thermal model: HotSpot 4.0 + TILTS
- Power trace
  - Running real SPEC2K benchmarks
  - Extracted from performance counter
- Hardware DVFS:
  - Triggered on/off at 86.5/85.5
  - Frequency scaling: 0.7
  - Voltage scaling: 0.92
  - DTM triggering overhead: 30 us
  - Schedule interval: 8ms

Experiment Methodology

- Quad core floorplan based on P4 Northwood: 93 function units with shared L2 cache



Performance Comparison



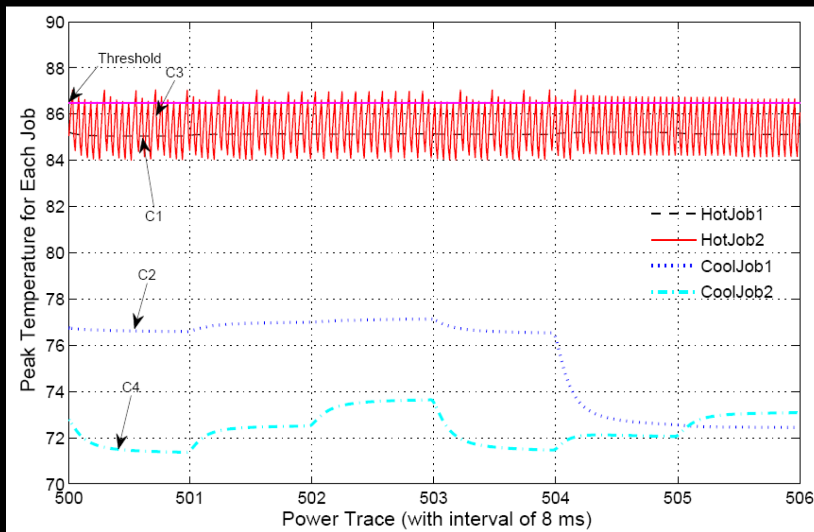
- ThresHot minimizes thermal emergencies to mitigate the performance loss from DVFS
- 13% and 6% reduction in performance penalty over “Base” and “Balancing”

Reliability Comparison

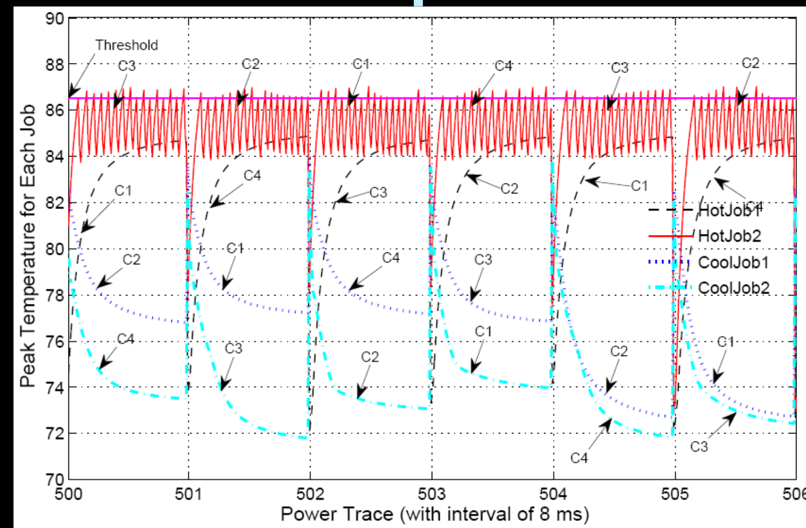
Algorithm	<10° C	[10° C~15	[15° C~20	>20° C
Baseline	99.91	0.07	0.02	0.01
Random	97.45	1.55	0.68	0.32
Balancing	95.50	2.67	1.23	0.60
RR-1	95.83	2.60	1.05	0.52
RR-2	96.91	1.93	0.78	0.38
ThresHot	98.22	1.21	0.43	0.14

- Thermal cyclings caused by the significant temperature variations are minimal in ThresHot

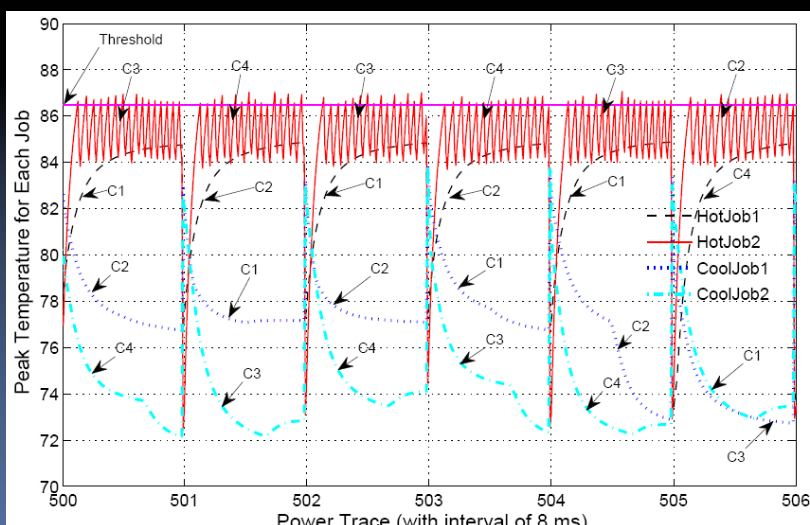
Thermal Behavior Comparison



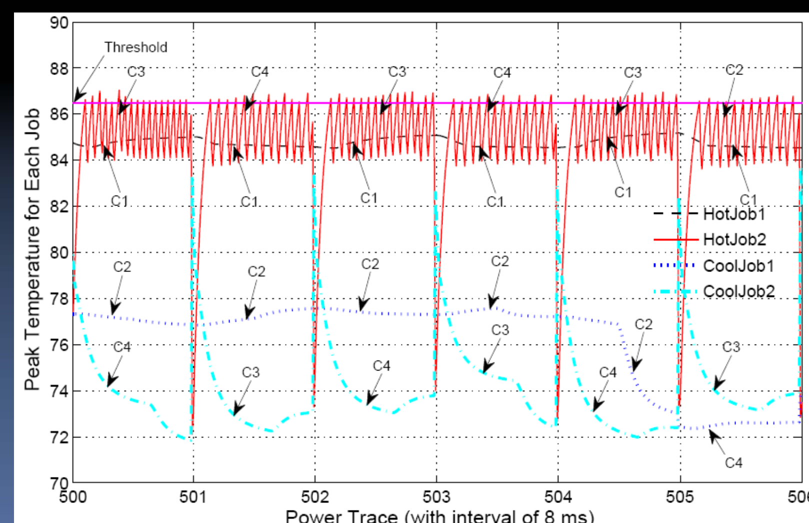
Baseline



RR



Balancing



ThresHot

Conclusion

- ThresHot algorithm does better than RR and Balancing in reducing thermal emergencies, and thermal cyclings
- ThresHot improves the performance in penalized time period by 13% and 6% compared to Baseline and Balancing
- Function unit level thermal control

Thank you

Questions?