Parallel Computing Systems

Universiteit van Amsterdam

Thermal management for S-NUCA Many-Cores via Synchronous Thread Rotations

Presenter: Yixian Shen

Cooperator: Sobhan Niknam

Co-supervisor : Anuj Pathania

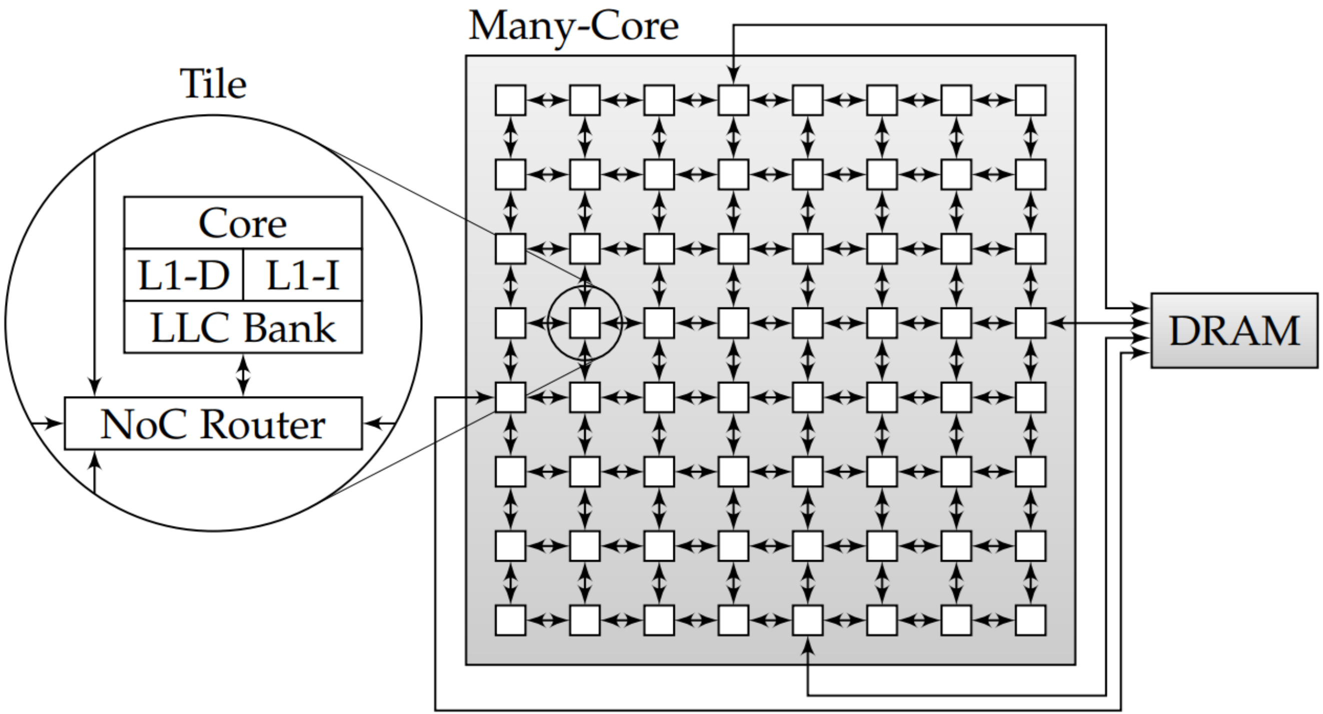
Supervisor: Andy D Pimentel

Outline

- ***Research background***
- A motivational example
- The key designs of our approach
- Experimental results
- Conclusion

Research background→S-NUCA architecture

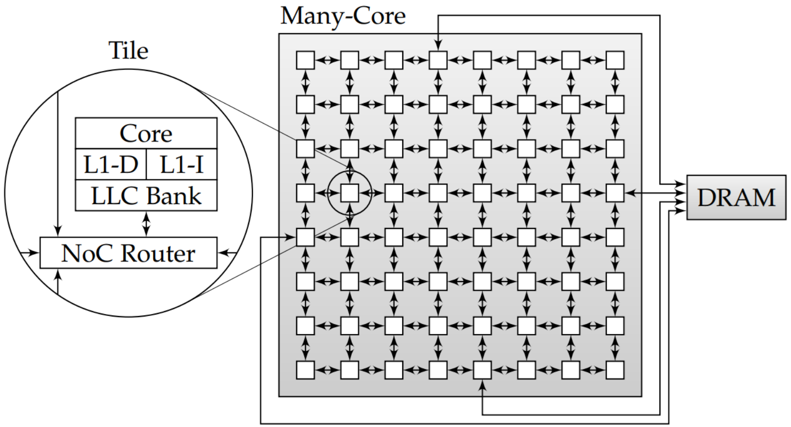
- Static Non-uniform Cache Architecture

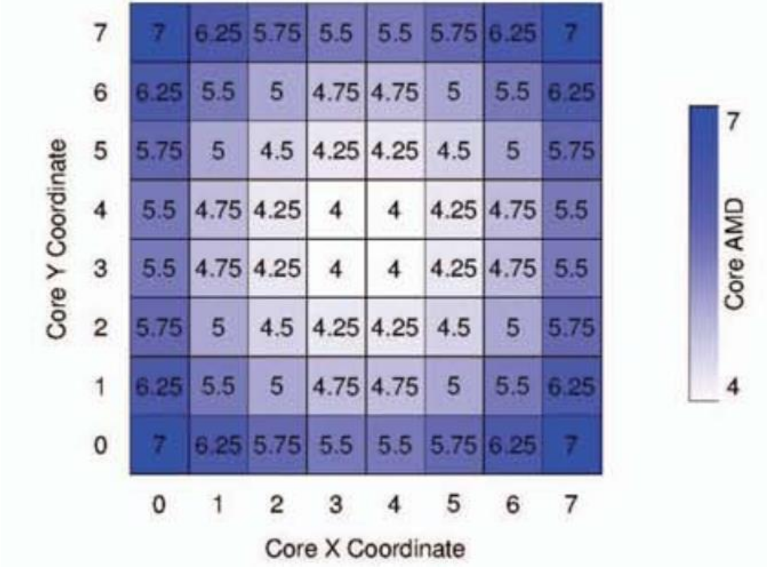


M. Rapp, A. Pathania, T. Mitra and J. Henkel, "**Neural Network-Based Performance Prediction for Task Migration on S-NUCA Many-Cores**," in *IEEE Transactions on Computers*,

Research background→S-NUCA architecture

- Characteristics of S-NUCA
  - Logically-shared yet physically-distributed.

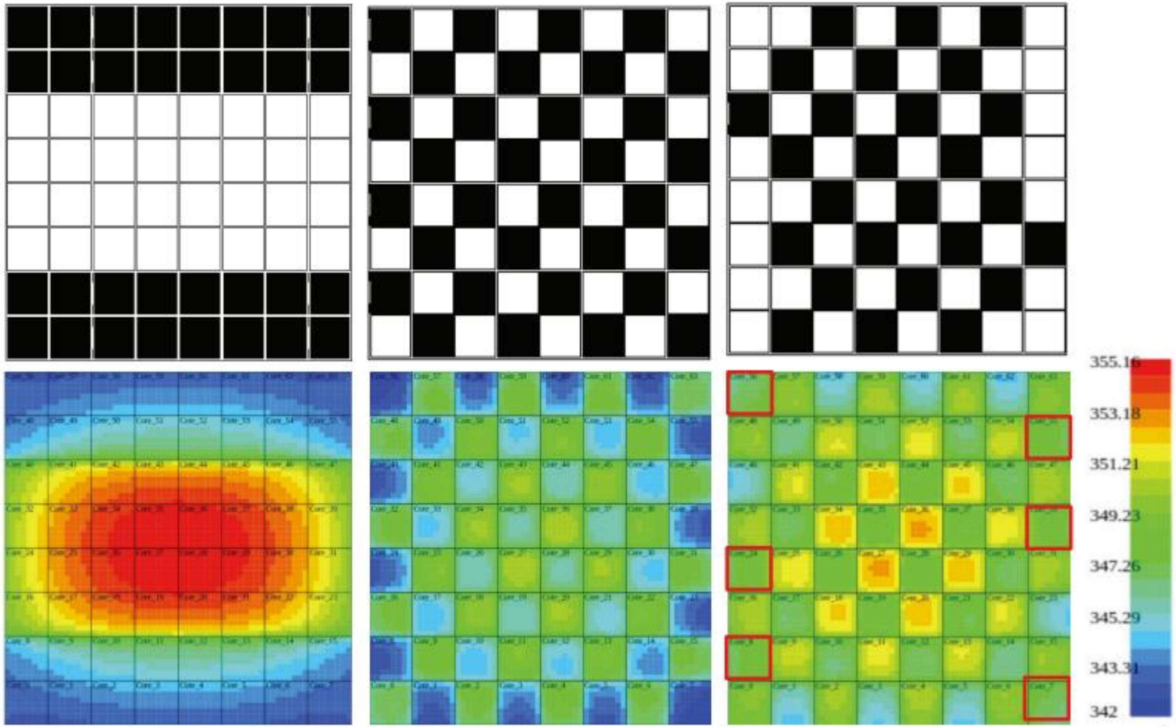




7	7	6.25	5.75	5.5	5.5	5.75	6.25	7
6	6.25	5.5	5	4.75	4.75	5	5.5	6.25
5	5.75	5	4.5	4.25	4.25	4.5	5	5.75
4	5.5	4.75	4.25	4	4	4.25	4.75	5.5
3	5.5	4.75	4.25	4	4	4.25	4.75	5.5
2	5.75	5	4.5	4.25	4.25	4.5	5	5.75
1	6.25	5.5	5	4.75	4.75	5	5.5	6.25
0	7	6.25	5.75	5.5	5.5	5.75	6.25	7
	0	1	2	3	4	5	6	7

A. Pathania and J. Henkel, "**Task scheduling for many-cores with S-NUCA caches**," *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018, pp. 557-562, doi: 10.23919/DATE.2018.8342069

Dark Silicon in S-NUCA many-cores



(a) TDP= 215 W Tsafe=80 ° C
Tpeak = 83 ° C Pused = 211W
#active cores = 32 @2.6Ghz

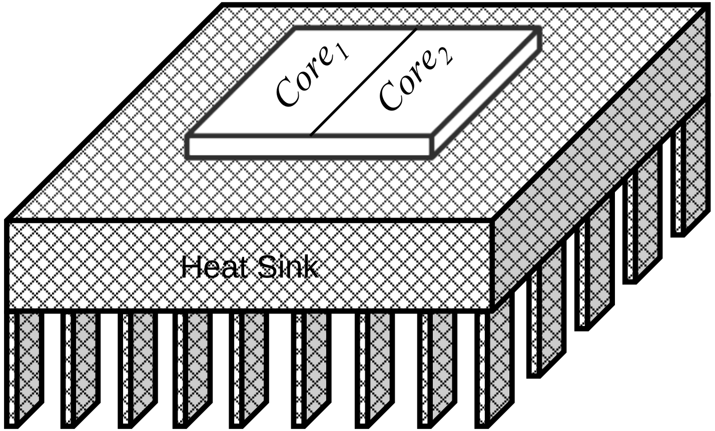
(b) TDP= 215 W Tsafe=80 ° C
Tpeak = 78 ° C Pused = 200W
#active cores = 32 @2.6Ghz

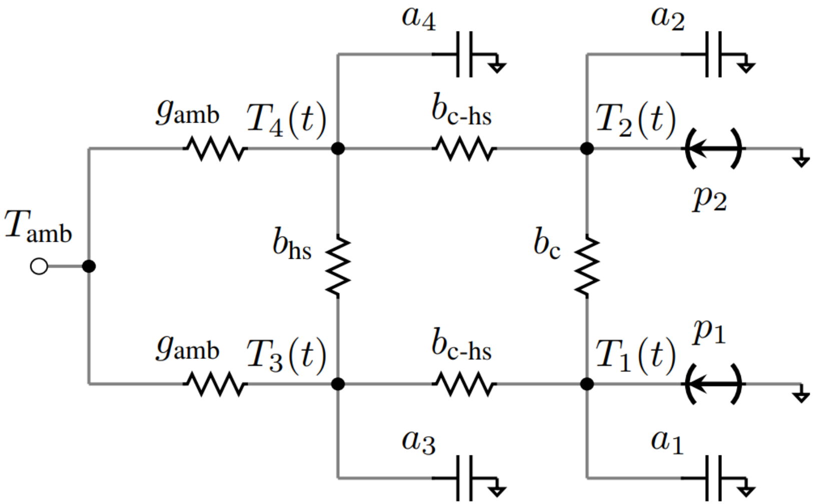
(c) TDP= 215 W Tsafe=80 ° C
Tpeak = 80 ° C Pused = 219W
#active cores = 32@2.6GHz + 6@ 1.8GHz

Shafique, Muhammad, et al. "**Dark silicon as a challenge for hardware/software co-design:** Invited special session paper." Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis. 2014..

Research background →R-C thermal model

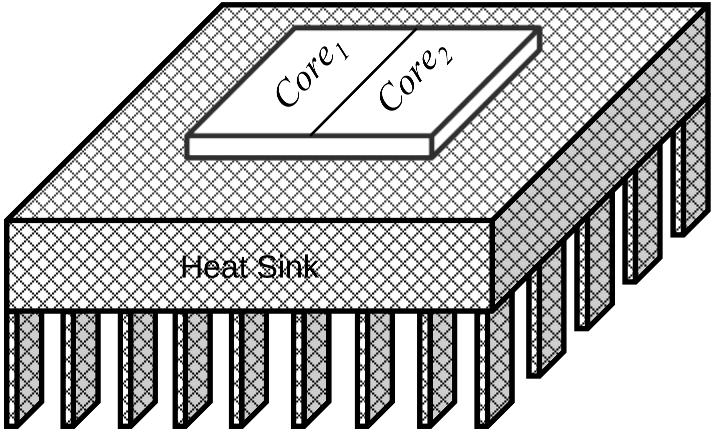
- Thermal behavior is dual to electronic circuit.

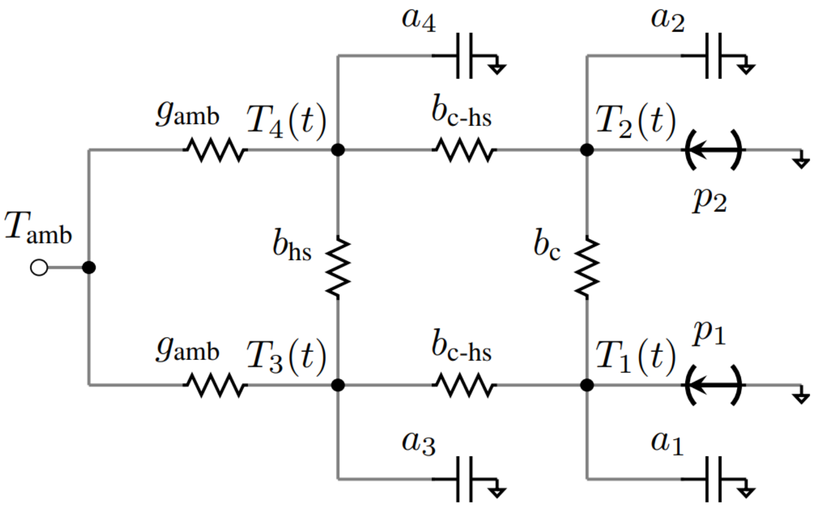




Research background →R-C thermal model

- Thermal behavior is dual to electronic circuit.





$$\mathbf{A}\mathbf{T}' + \mathbf{B}\mathbf{T} = \mathbf{P} + T_{\text{amb}}\mathbf{G}$$

Outline

- Research background
- ***A motivational example***
- The key designs of our approach
- Experimental results
- Conclusion.

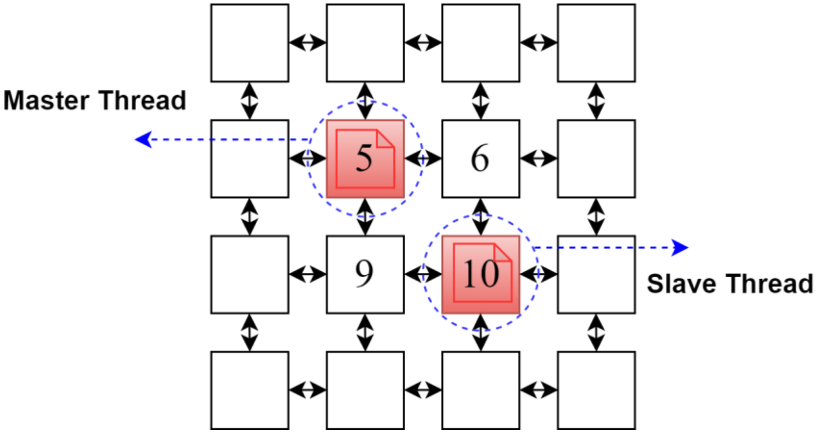
A motivational example

- Experimental platform: simulated 16-core S-NUCA processors
- Experimental configuration
- Thermal threshold 70(℃)

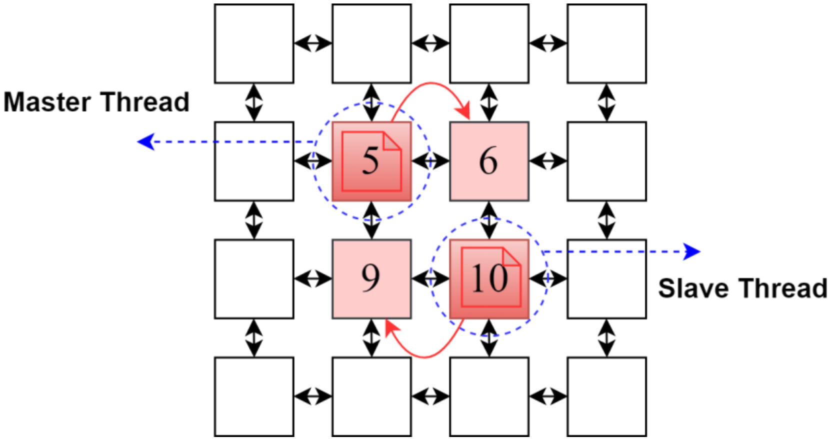
	Benchmark	Thread	DTM method	Execute at
Case 1	blackscholes	1 master, 1 slave	No	Core 5,10
Case 2	blackscholes	1 master, 1 slave	TSP	Core 5,10
Case 3	blackscholes	1 master, 1 slave	Task rotation	Core 5,6,9,10

A motivational example

	Benchmark	Thread	DTM method	Execute at
Case 1	blackscholes	1 master, 1 slave	No	Core 5,10
Case 2	blackscholes	1 master, 1 slave	TSP	Core 5,10
Case 3	blackscholes	1 master, 1 slave	Task rotation	Core 5,6,9,10



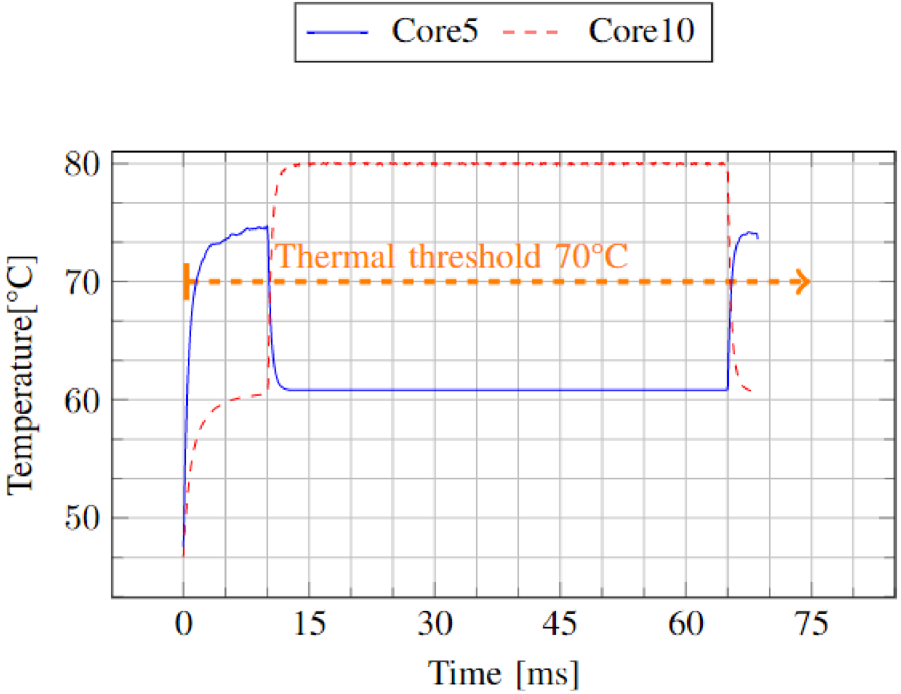
Case1 and Case 2



Case 3

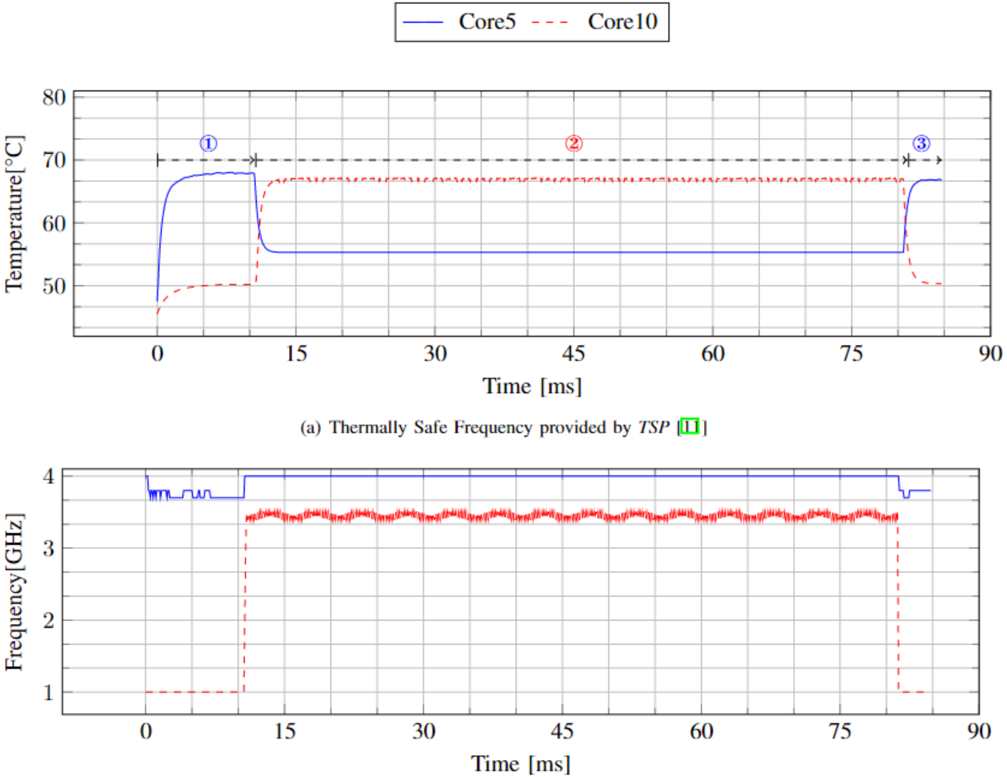
A motivational example---Case 1

- Peak temperature 80.03(℃)
- Execution time 67.97(ms)



A motivational example---Case 2

- ➢ Peak temperature 67.94(℃)
- ➢ Execution time 84.49(ms)

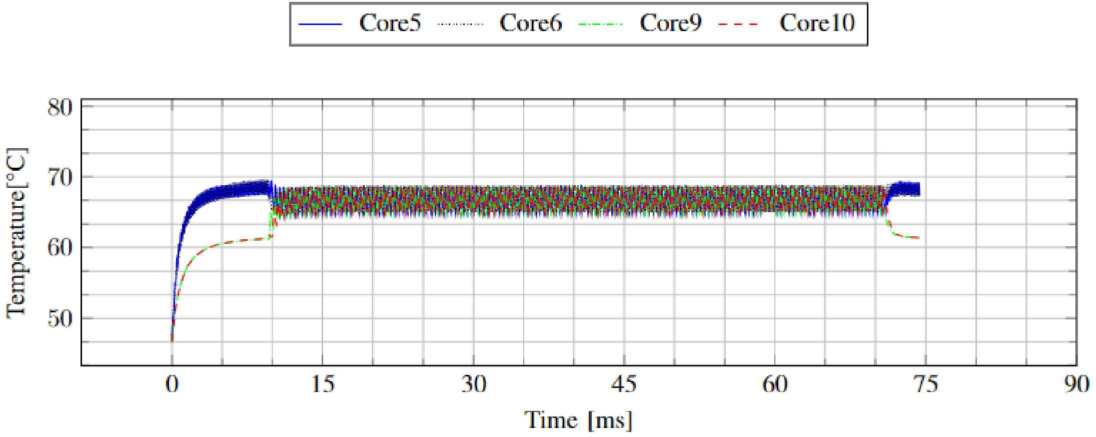


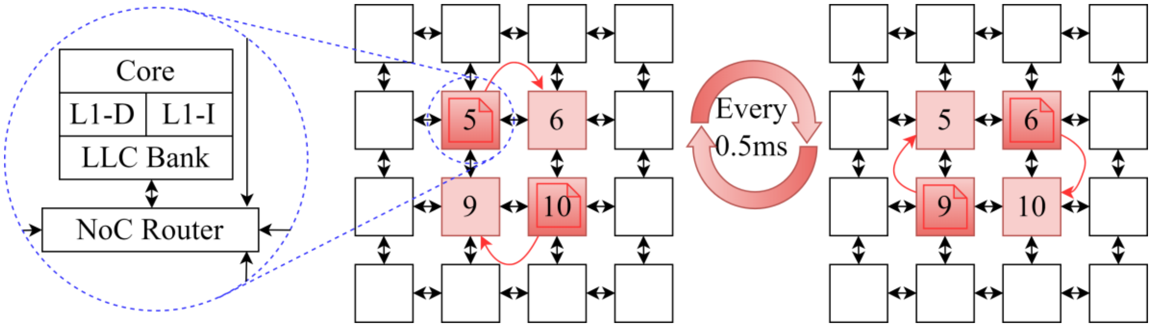
(a) Thermally Safe Frequency provided by *TSP* [11]

(b) Frequency profiling by TSP

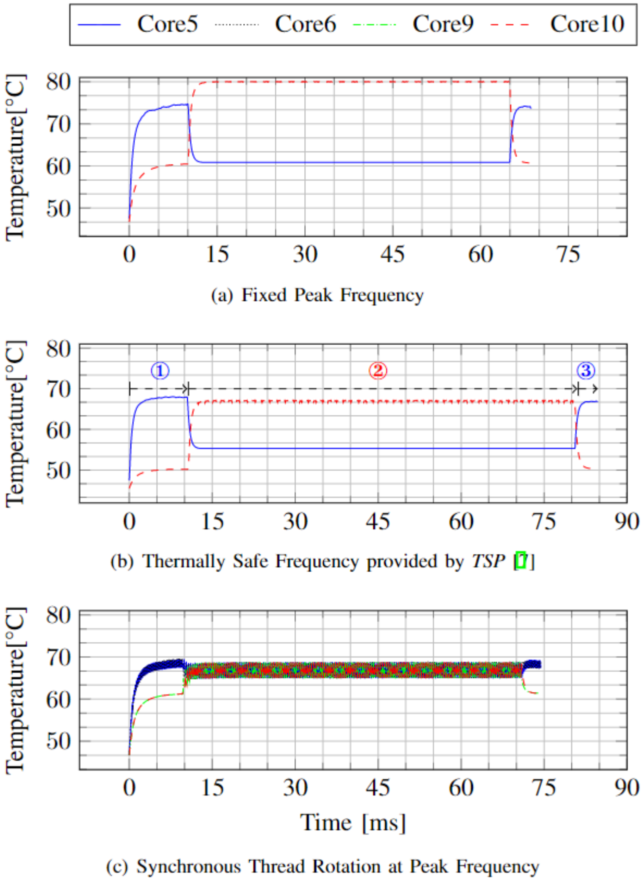
A motivational example---Case 3

- Peak temperature 69.32(℃)
- Execution time 74.47(ms)





A motivational example



(a) Fixed Peak Frequency

(b) Thermally Safe Frequency provided by *TSP* [7]

(c) Synchronous Thread Rotation at Peak Frequency

	Benchmark	Thread	DTM method	Execute at
Case 1	blackscholes	1 master, 1 slave	No	Core 5,10
Case 2	blackscholes	1 master, 1 slave	TSP	Core 5,10
Case 3	blackscholes	1 master, 1 slave	Task rotation	Core 5,6,9,10

	Case 1	Case 2	Case 3
Peak temp(℃)	80.03	67.94	69.32
Exec time(ms)	67.97	84.49	74.47
Penalty(％)		19.55	8.72

Case 3 is **10.83% faster** than Case 2

Rotation penalty **<** DVFS-based penalty

Outline

- Research background
- A motivational example
- ***The key designs of our approach***
- Experimental results
- Conclusion.

The key designs of our approach

- Thermal regulations
    - ❑ A theoretical one-shot peak temperature calculation

- Architecture-aware task scheduling
    - ❑ *Hotpotato* thermal-aware scheduler.

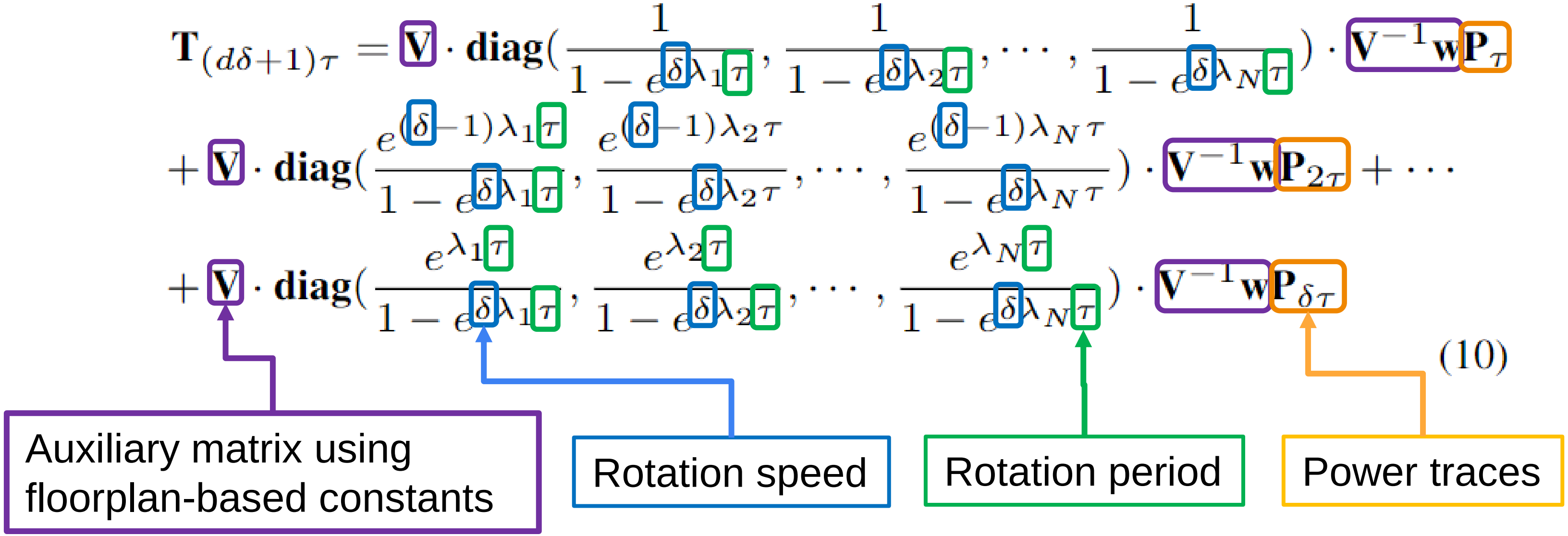
The key designs of our approach

- Thermal regulations
  - ❑ A theoretical one-shot peak temperature calculation

$$
\begin{aligned}
\mathbf{T}_{(d\delta+1)\tau} &= \mathbf{V}\cdot\mathbf{diag}(\frac{1}{1-e^{\delta\lambda_1\tau}},\frac{1}{1-e^{\delta\lambda_2\tau}},\cdots,\frac{1}{1-e^{\delta\lambda_N\tau}})\cdot\mathbf{V}^{-1}\mathbf{w}\mathbf{P}_{\tau} \\
&+\mathbf{V}\cdot\mathbf{diag}(\frac{e^{(\delta-1)\lambda_1\tau}}{1-e^{\delta\lambda_1\tau}},\frac{e^{(\delta-1)\lambda_2\tau}}{1-e^{\delta\lambda_2\tau}},\cdots,\frac{e^{(\delta-1)\lambda_N\tau}}{1-e^{\delta\lambda_N\tau}})\cdot\mathbf{V}^{-1}\mathbf{w}\mathbf{P}_{2\tau}+\cdots \\
&+\mathbf{V}\cdot\mathbf{diag}(\frac{e^{\lambda_1\tau}}{1-e^{\delta\lambda_1\tau}},\frac{e^{\lambda_2\tau}}{1-e^{\delta\lambda_2\tau}},\cdots,\frac{e^{\lambda_N\tau}}{1-e^{\delta\lambda_N\tau}})\cdot\mathbf{V}^{-1}\mathbf{w}\mathbf{P}_{\delta\tau}
\end{aligned}
\tag{10}
$$

The key designs of our approach

- Thermal regulations
  - ❑ A theoretical one-shot peak temperature calculation



$$
\begin{aligned}
\mathbf{T}_{(d\delta+1)\tau} = {} & \boxed{\mathbf{V}} \cdot \mathbf{diag}\Big(\frac{1}{1-e^{\boxed{\delta}\lambda_1\boxed{\tau}}}, \frac{1}{1-e^{\boxed{\delta}\lambda_2\boxed{\tau}}}, \cdots, \frac{1}{1-e^{\boxed{\delta}\lambda_N\boxed{\tau}}}\Big) \cdot \boxed{\mathbf{V}^{-1}\mathbf{w}}\boxed{\mathbf{P}_\tau} \\
& + \boxed{\mathbf{V}} \cdot \mathbf{diag}\Big(\frac{e^{(\boxed{\delta}-1)\lambda_1\boxed{\tau}}}{1-e^{\boxed{\delta}\lambda_1\boxed{\tau}}}, \frac{e^{(\boxed{\delta}-1)\lambda_2\tau}}{1-e^{\boxed{\delta}\lambda_2\tau}}, \cdots, \frac{e^{(\boxed{\delta}-1)\lambda_N\tau}}{1-e^{\boxed{\delta}\lambda_N\tau}}\Big) \cdot \boxed{\mathbf{V}^{-1}\mathbf{w}}\boxed{\mathbf{P}_{2\tau}} + \cdots \\
& + \boxed{\mathbf{V}} \cdot \mathbf{diag}\Big(\frac{e^{\lambda_1\boxed{\tau}}}{1-e^{\boxed{\delta}\lambda_1\boxed{\tau}}}, \frac{e^{\lambda_2\boxed{\tau}}}{1-e^{\boxed{\delta}\lambda_2\boxed{\tau}}}, \cdots, \frac{e^{\lambda_N\boxed{\tau}}}{1-e^{\boxed{\delta}\lambda_N\boxed{\tau}}}\Big) \cdot \boxed{\mathbf{V}^{-1}\mathbf{w}}\boxed{\mathbf{P}_{\delta\tau}}
\end{aligned}
\qquad (10)
$$

Auxiliary matrix using floorplan-based constants

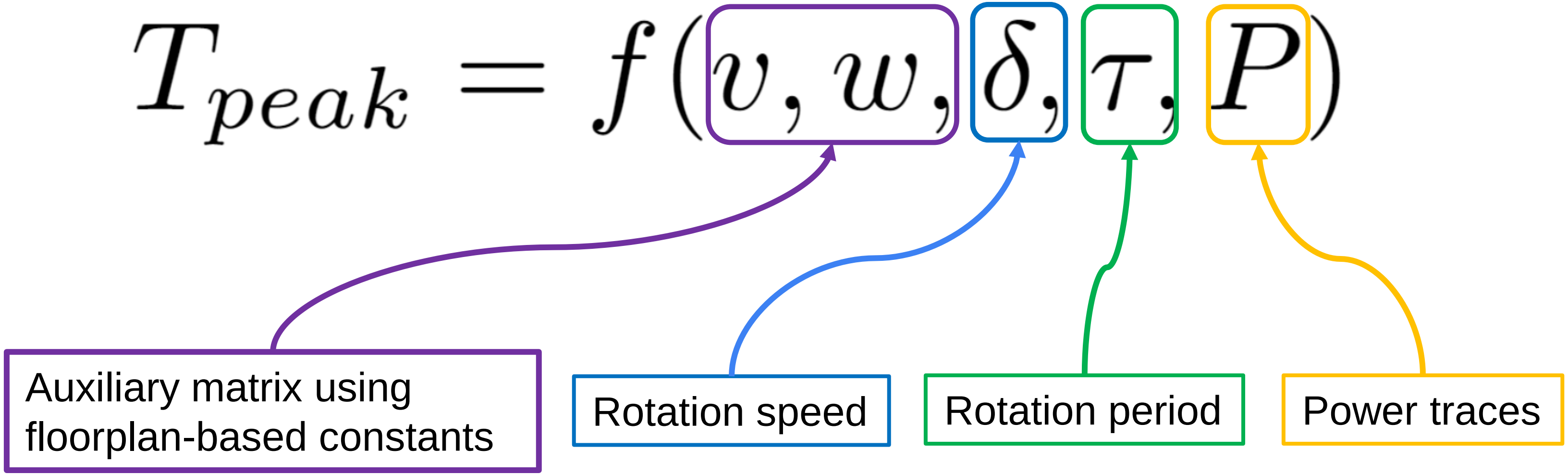
Rotation speed

Rotation period

Power traces

The key designs of our approach

- ➢ Thermal regulations
  - ❑ A theoretical one-shot peak temperature calculation



$$T_{peak} = f(\boxed{v, w,} \boxed{\delta,} \boxed{\tau,} \boxed{P})$$

Auxiliary matrix using floorplan-based constants

Rotation speed

Rotation period

Power traces

The hotpotato scheduling

- ➢ The inherent heterogeneity of S-NUCA

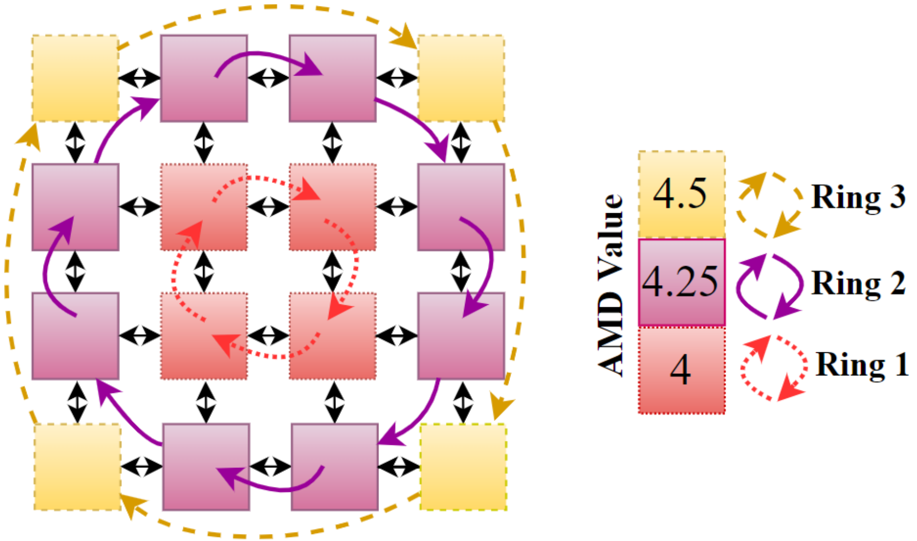


Fig. 3: Abstraction for concentric AMD-based rotation rings.

□ Access cache latency

Ring1 < Ring2 < Ring3

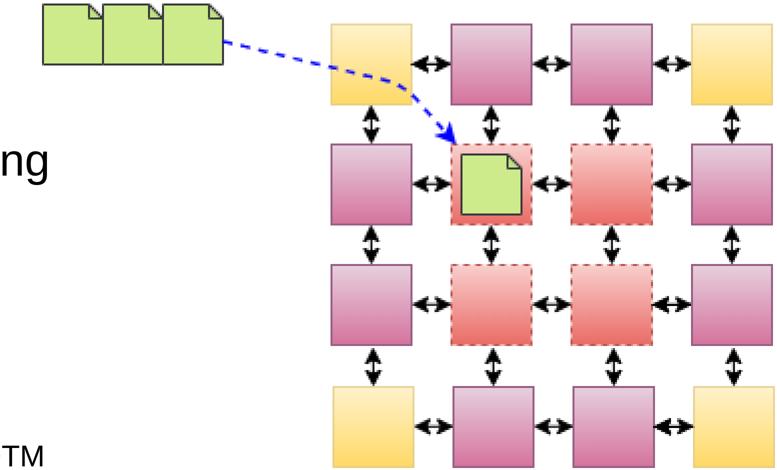
□ Thermal dissipation condition

Ring1 < Ring2 < Ring3

The hotpotato scheduling

- ***maximizing the performance*** under thermal threshold
  - Keep peak temperature T_{peak} close to thermal threshold T_{DTM}
    - Migrate to the thread to lower ring close to the center ring
  - Minimize the task rotation penalty
    - Slow down the rotation speed

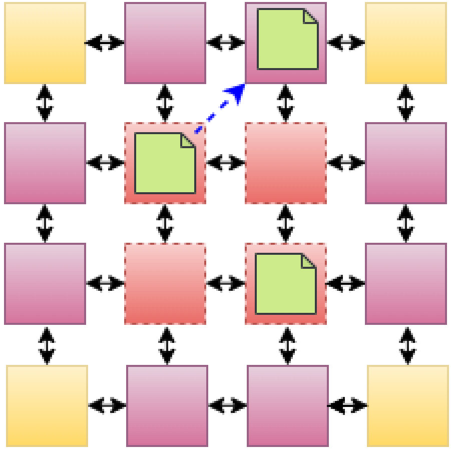
The hotpotato scheduling



- ➢ New threads enter S-NUCA
  - ❶ Attempt to assign thread to the lowest ring
  - ❷ Calculate the T_{peak}
  - ❸ If $T_{peak} < T_{DTM}$, rotate at that ring
  - ❹ Else, attempt to L+1 rings until $T_{peak} < T_{DTM}$
  - ❺ Reach R_{max} ,still not managed, speed up the rotation speed

The hotpotato scheduling

- Thermal violation during runtime
  - ❶ Rank the threads by CPI(cycle per instruction)
  - ❷ Select the thread with lowest CPI
  - ❸ Migrate thread to the L+1 rings
  - ❹ If $T_{peak} < T_{DTM}$ ， rotate at that ring
  - ❺ Else, attempt to L+1 rings until $T_{peak} < T_{DTM}$
  - ❻ Reach R_{max} ,still not managed, speed up the rotation speed



The hotpotato scheduling

- ➢ Headroom not exploited

❶ Rank the threads by CPI(cycle per instruction)

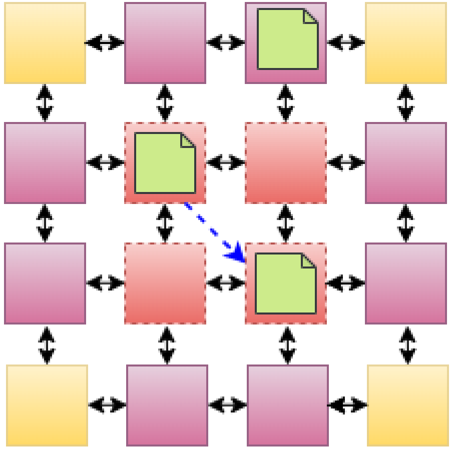
❷ Select the thread with highest CPI

❸ Migrate thread to the L-1 rings

❹ If T_{DTM} - T_{peak}< Δ, rotate at that ring

❺ Else, attempt to L-1 rings until T_{DTM} - T_{peak}< Δ

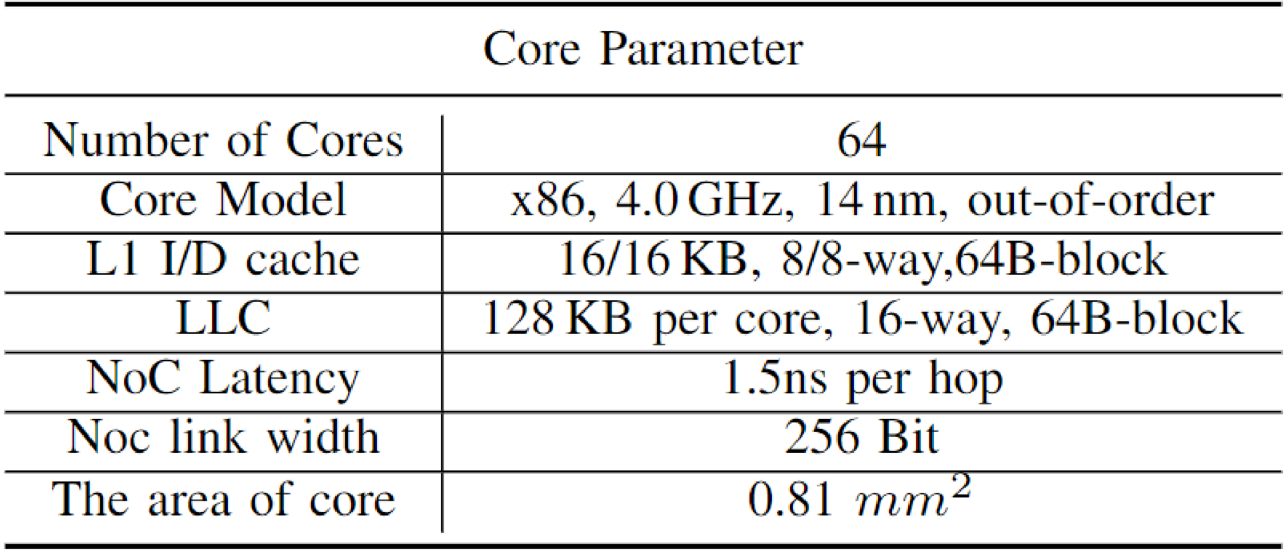
❻ Reach R_{min} ,still not managed, slow down the rotation speed



Outline

- Research background
- A motivational example
- The key designs of our approach
- ***Experimental results***
- Conclusion.

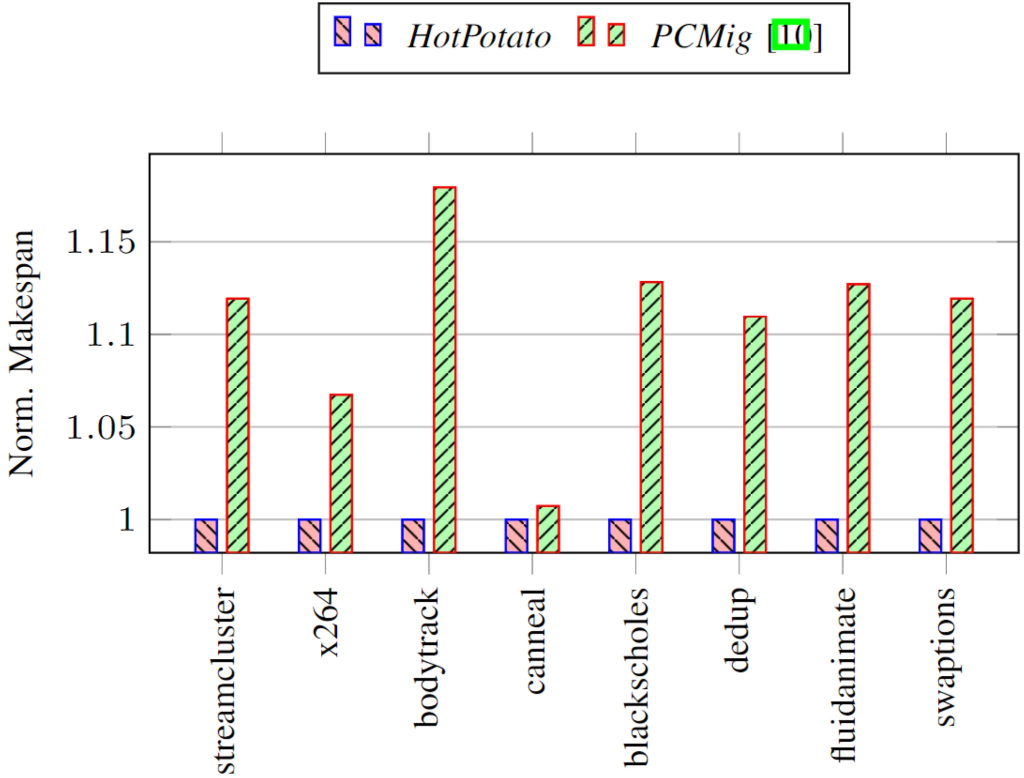
The experimental configuration



Core Parameter	
Number of Cores	64
Core Model	x86, 4.0 GHz, 14 nm, out-of-order
L1 I/D cache	16/16 KB, 8/8-way,64B-block
LLC	128 KB per core, 16-way, 64B-block
NoC Latency	1.5ns per hop
Noc link width	256 Bit
The area of core	0.81 mm^2

The experimental results

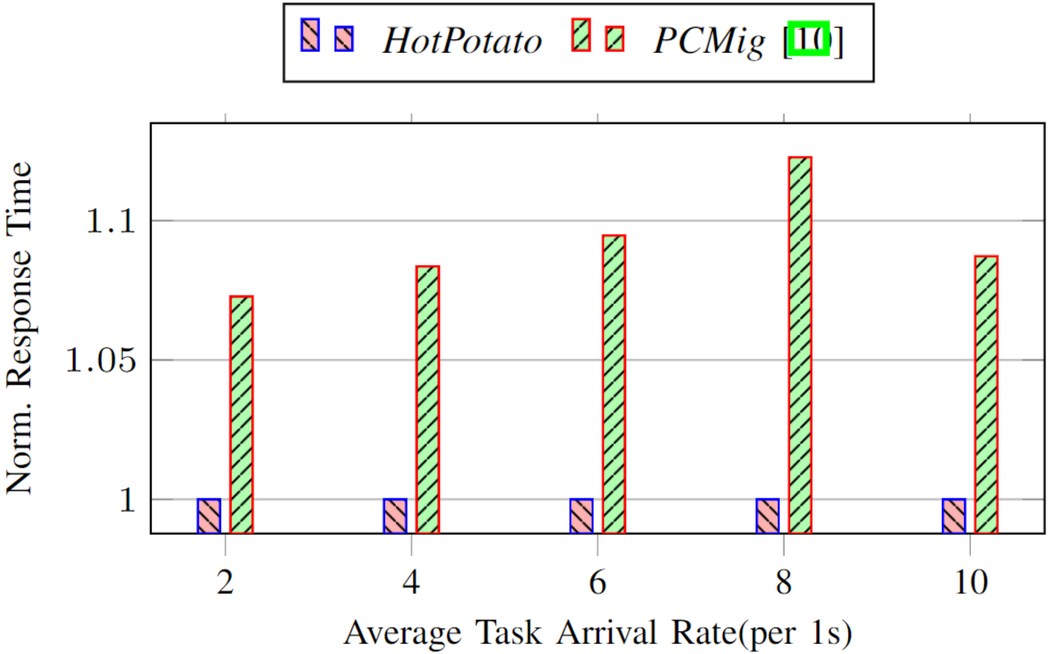
- Full loaded system



(a) Homogeneous Workload

The experimental results

- Open system
- The arrival time follow Poisson distribution.



(b) Heterogeneous Workload

Outline

- Research background
- A motivational example
- The key designs of our approach
- Experimental results
- ***Conclusion.***

The conclusion.

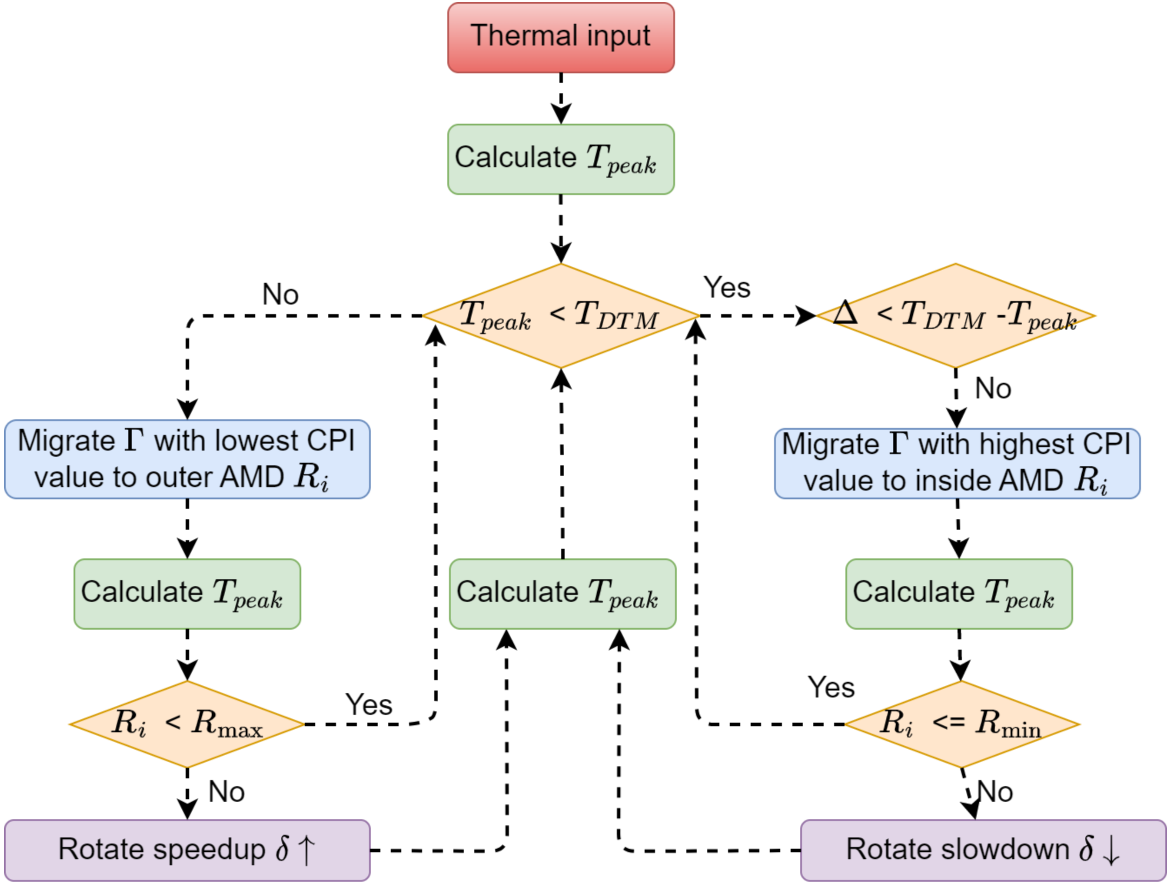
- We are the first to use synchronous thread rotations for thermal management.
- We propose a one-shot peak temperature calculation approach based on the RC thermal model.
- We present a *hotpotato* scheduling for task scheduling to speed up the performance.
- We implement our hotpotato scheduler in the state-of-the-art thermal simulator hotsniper

Thanks for your attention

y.shen@uva.nl

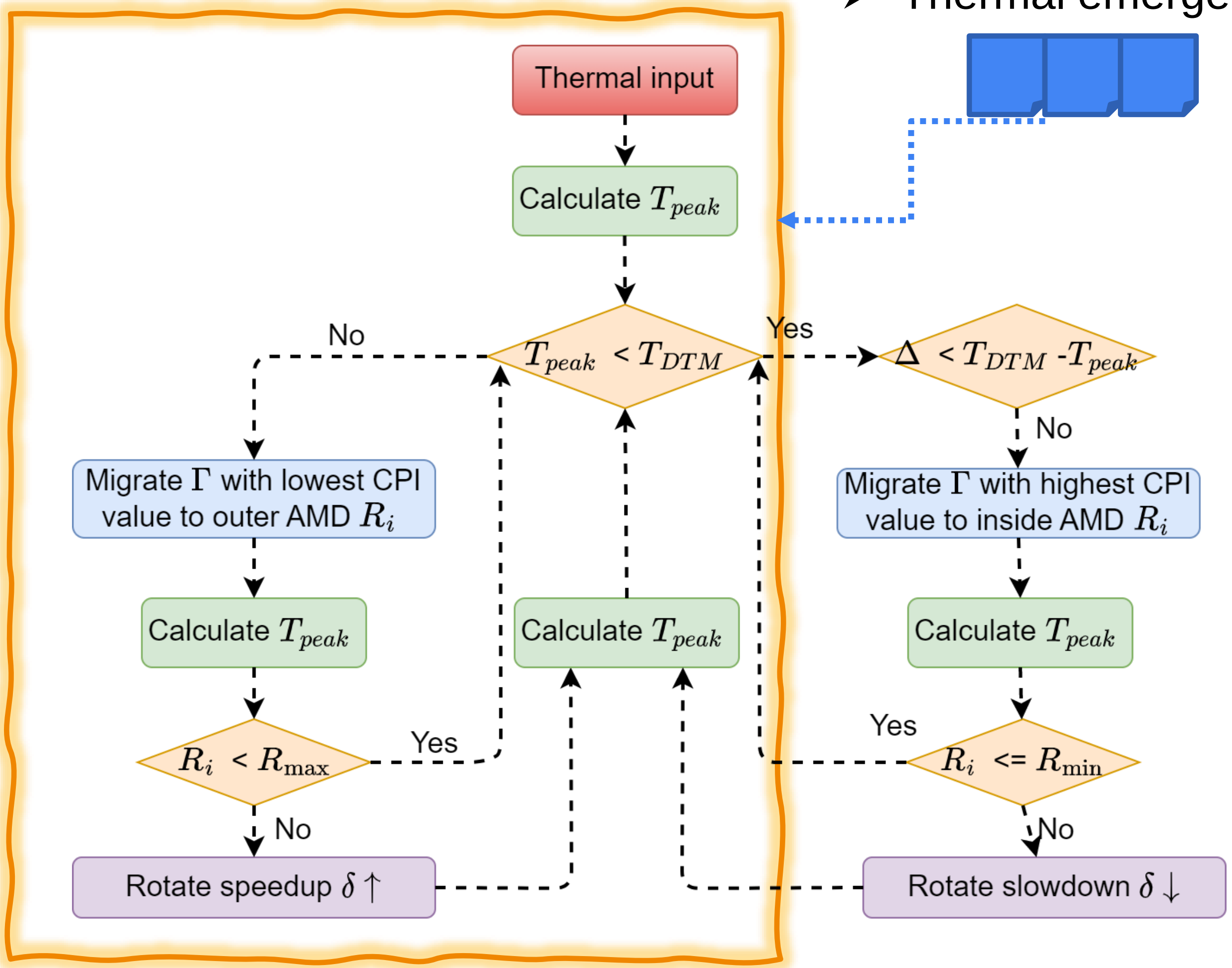
Questions

The hotpotato scheduling


Thermal input
Calculate T_{peak}
T_{peak} < T_{DTM}
No
Yes
Δ < T_{DTM} -T_{peak}
No
Migrate Γ with lowest CPI value to outer AMD R_i
Migrate Γ with highest CPI value to inside AMD R_i
Calculate T_{peak}
Calculate T_{peak}
Calculate T_{peak}
Yes
R_i < $R_{\max}$
Yes
R_i <= $R_{\min}$
No
No
Rotate speedup $\delta\uparrow$
Rotate slowdown $\delta\downarrow$

The hotpotato scheduling

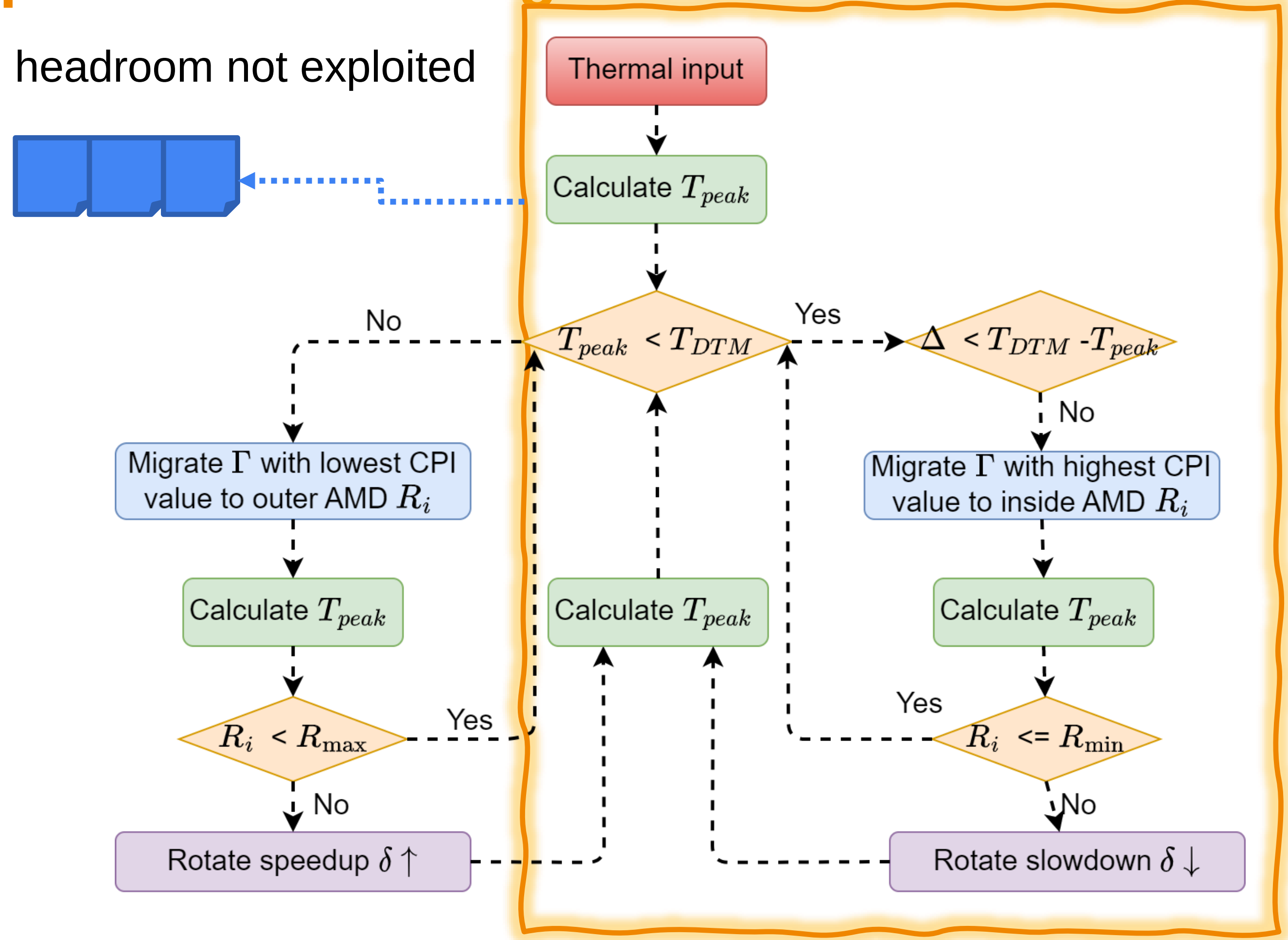
- Thermal emergency may happen



Thermal input
Calculate T_{peak}
T_{peak} < T_{DTM}
No
Yes
Δ < T_{DTM} -T_{peak}
No
Migrate Γ with lowest CPI value to outer AMD R_i
Migrate Γ with highest CPI value to inside AMD R_i
Calculate T_{peak}
Calculate T_{peak}
Calculate T_{peak}
R_i < $R_{\max}$
Yes
No
Yes
R_i <= $R_{\min}$
No
Rotate speedup $\delta\uparrow$
Rotate slowdown $\delta\downarrow$

The hotpotato scheduling

- Thermal headroom not exploited



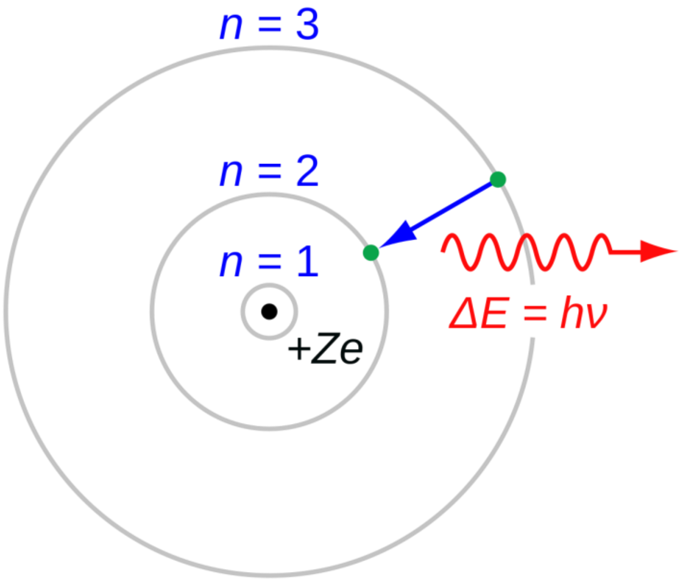
Thermal input
Calculate T_{peak}
T_{peak} < T_{DTM}
Yes
No
Δ < T_{DTM} -T_{peak}
No
Migrate Γ with lowest CPI value to outer AMD R_i
Migrate Γ with highest CPI value to inside AMD R_i
Calculate T_{peak}
Calculate T_{peak}
Calculate T_{peak}
R_i < $R_{\max}$
Yes
No
Yes
R_i <= $R_{\min}$
No
Rotate speedup $\delta\uparrow$
Rotate slowdown $\delta\downarrow$

The limitations of this work.

- Only suitable for the system with LLC.
- The heuristic of hotpotato is a little bit straightforward.
- Not sure if the data congestion crashes the system when executing on more core systems.

The hotpotato scheduling

- Bohr’s model



- ❑ Every orbit with different energy level
- ❑ The atom gains energy

  Jumping from low orbit to high orbit
- ❑ The atom lose energy

  falling from high orbit to low orbit