THE OHIO STATE UNIVERSITY

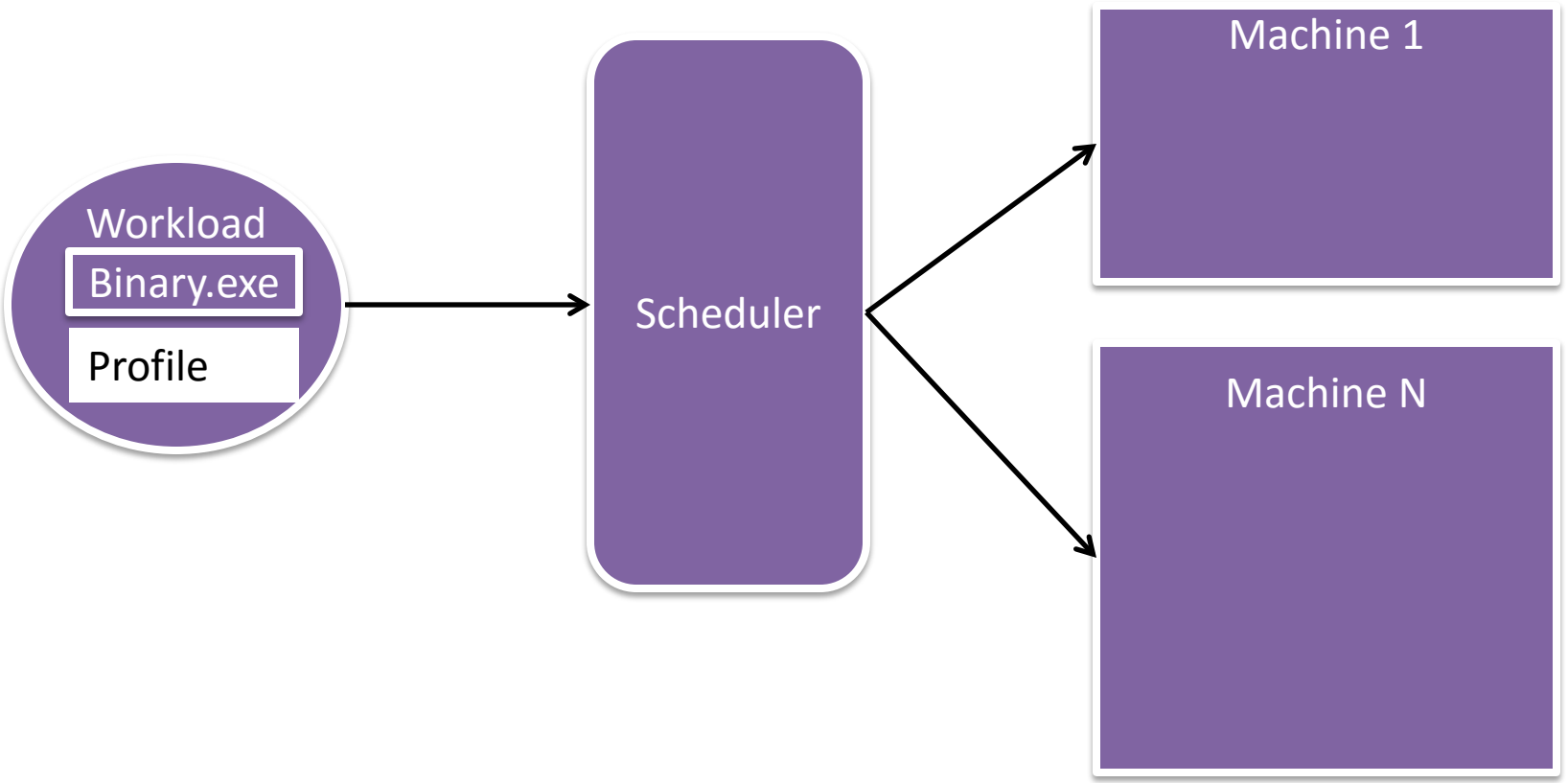
Adaptive Power Profiling for Many-Core HPC Architectures

Jaimie Kelley, Christopher Stewart
The Ohio State University
Devesh Tiwari, Saurabh Gupta
Oak Ridge National Laboratory

OAK RIDGE National Laboratory

State-of-the-Art

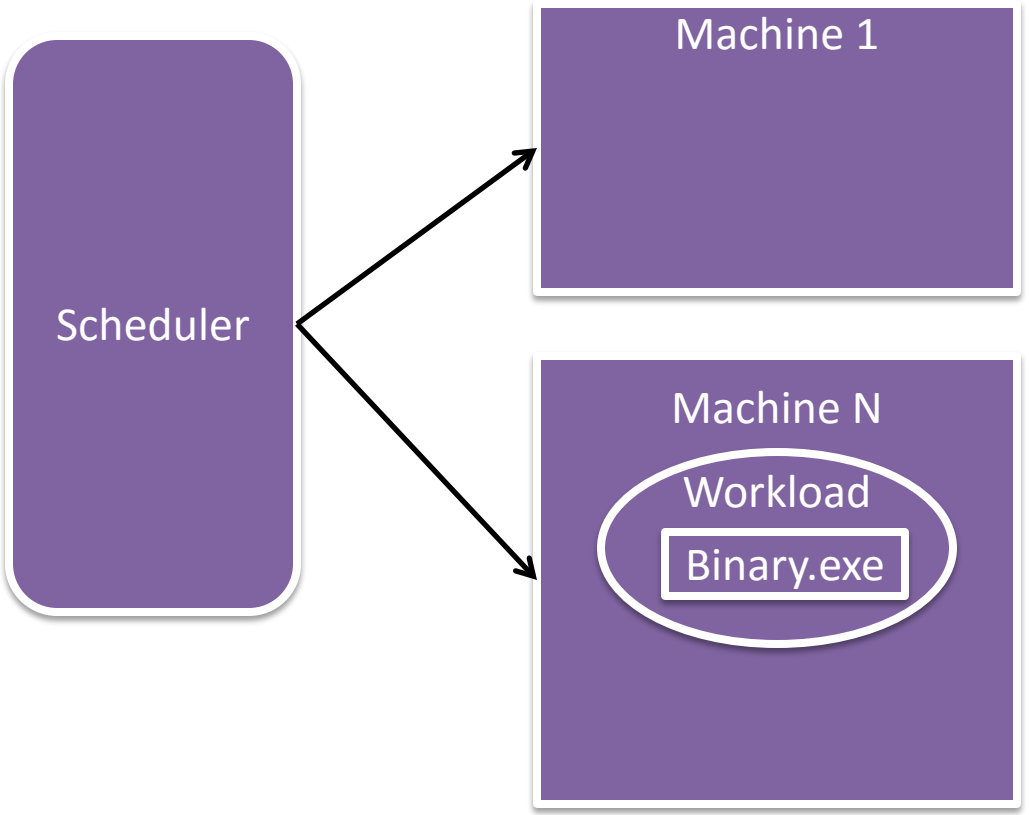
- Schedulers use profiles to assign resources.



State-of-the-Art

- Schedulers use profiles to assign resources.

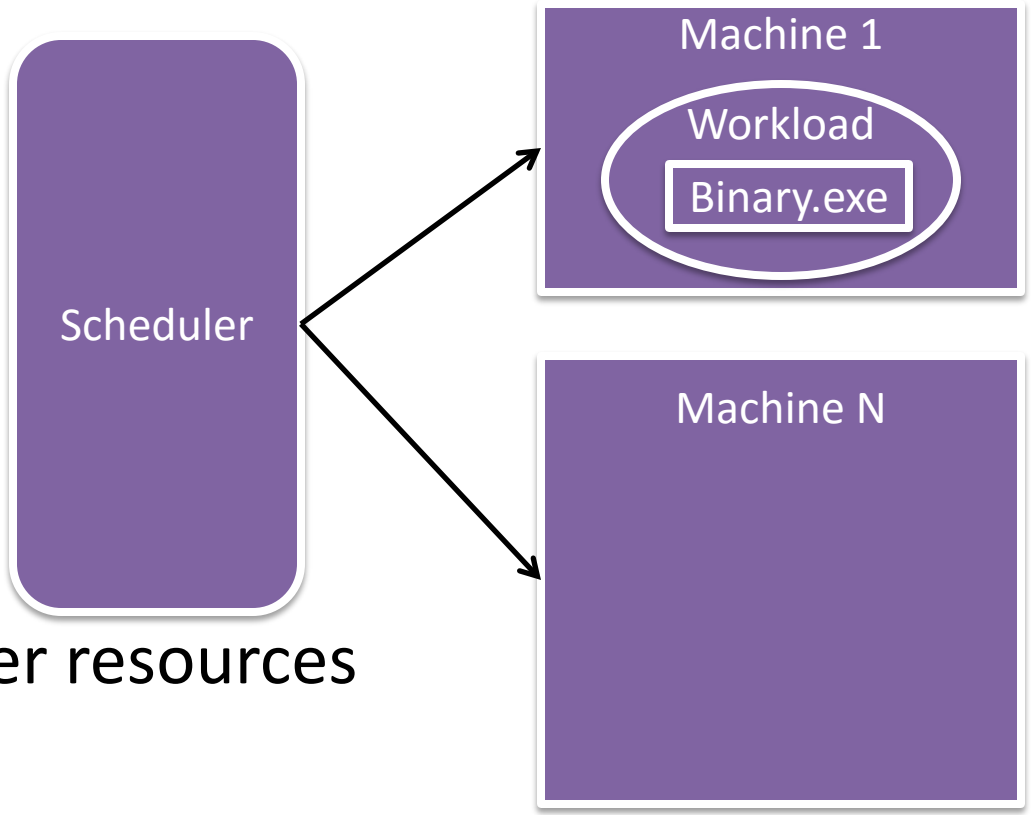
- Profiles contain:
  - CPU usage
  - Memory usage
  - Disk usage



State-of-the-Art

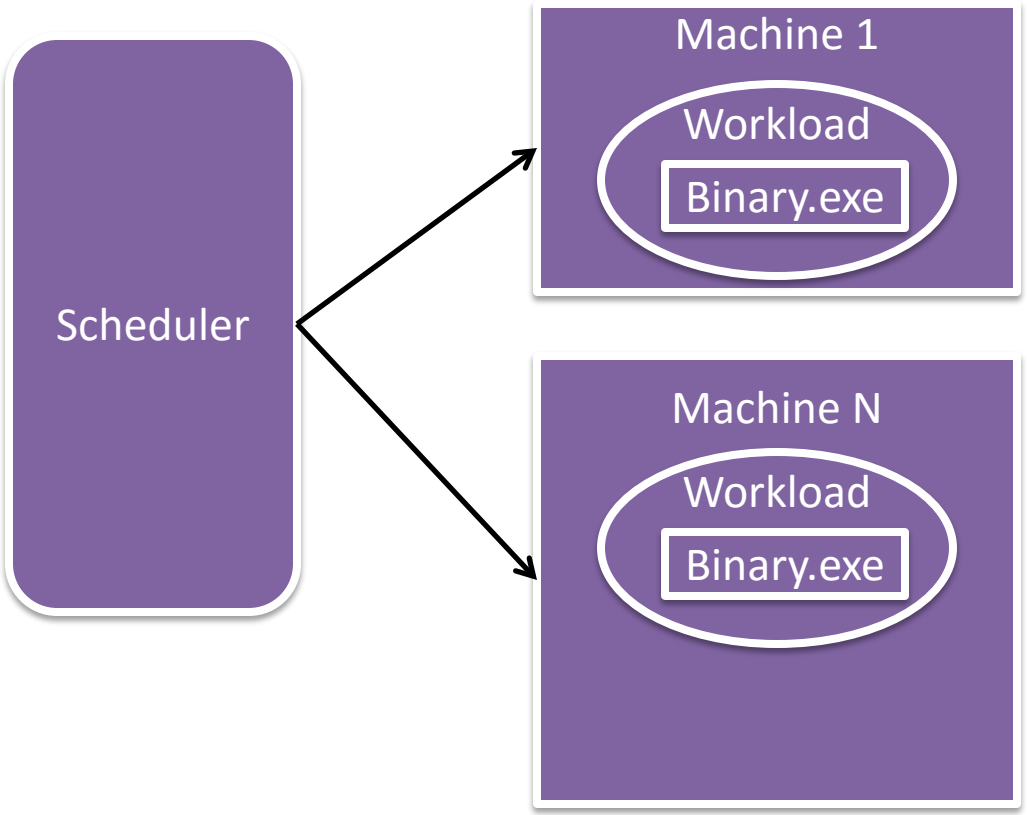
- Schedulers use profiles to assign resources.

- Profiles inform:
  - Assigning only what is needed
  - Which apps strain datacenter resources



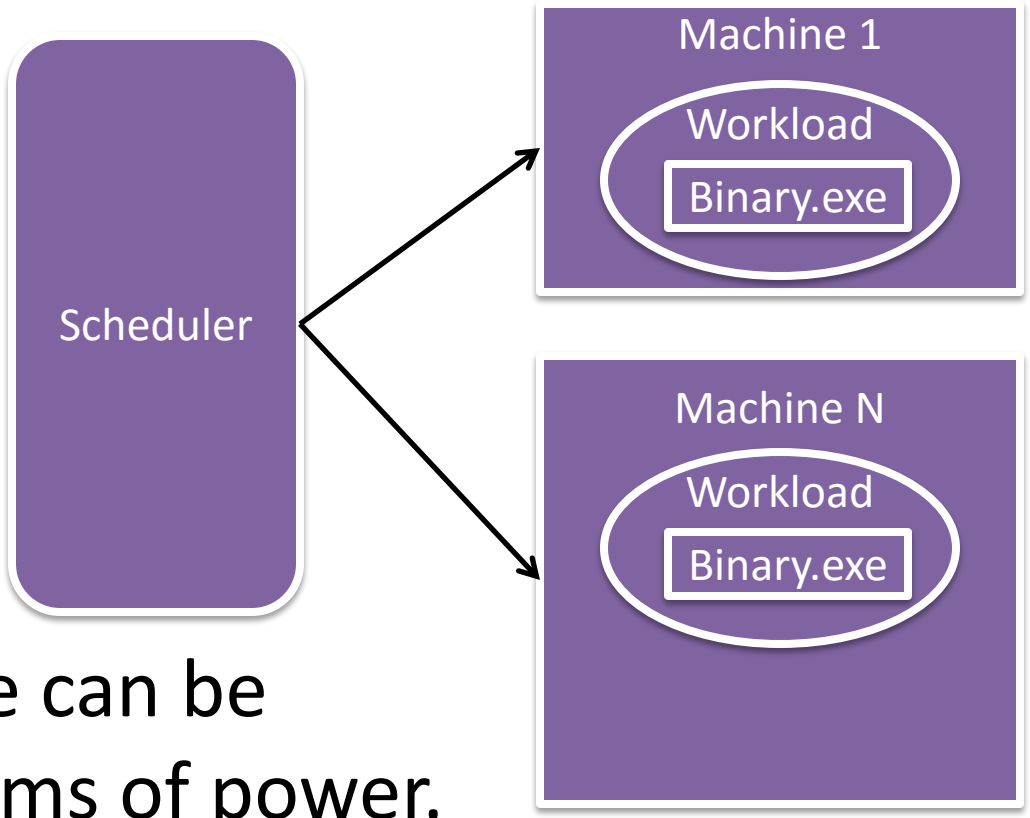
State-of-the-Art

- HPC workloads may use dozens of machines.
- Traditional HPC workloads use every available core.

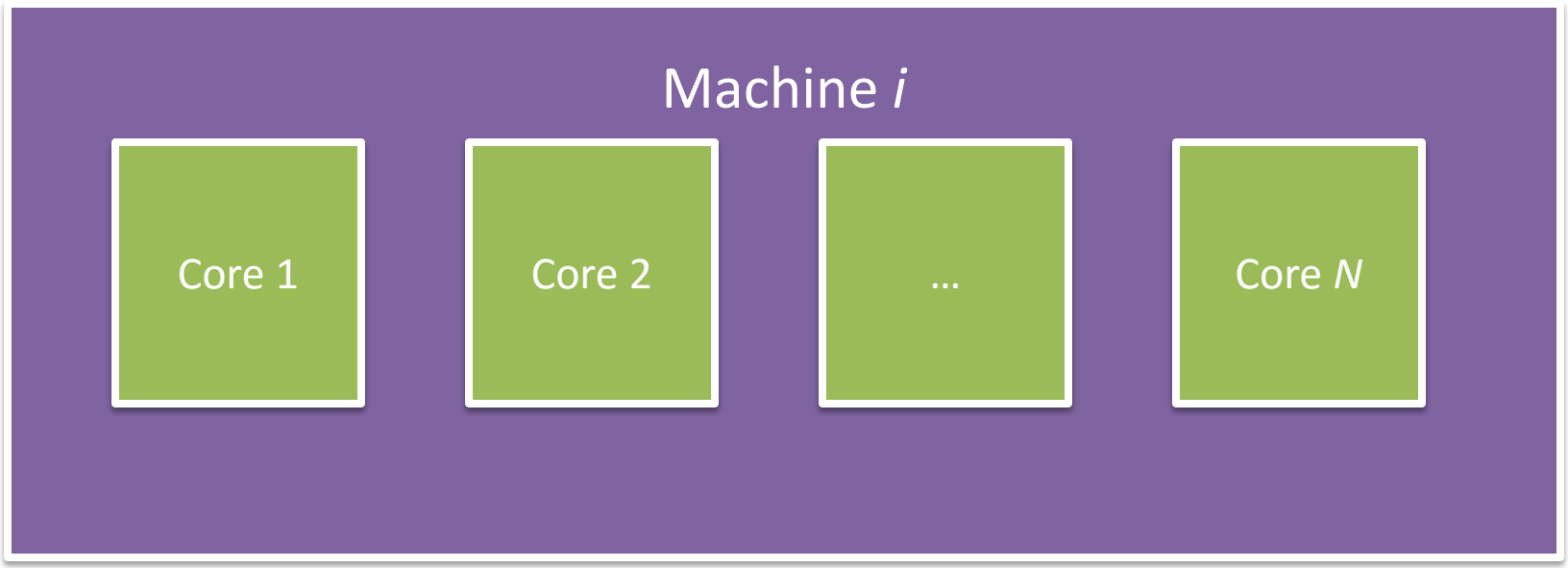


State-of-the-Art

- HPC workloads may use dozens of machines.
- Traditional HPC workloads use every available core.
- Using every core can be inefficient in terms of power.

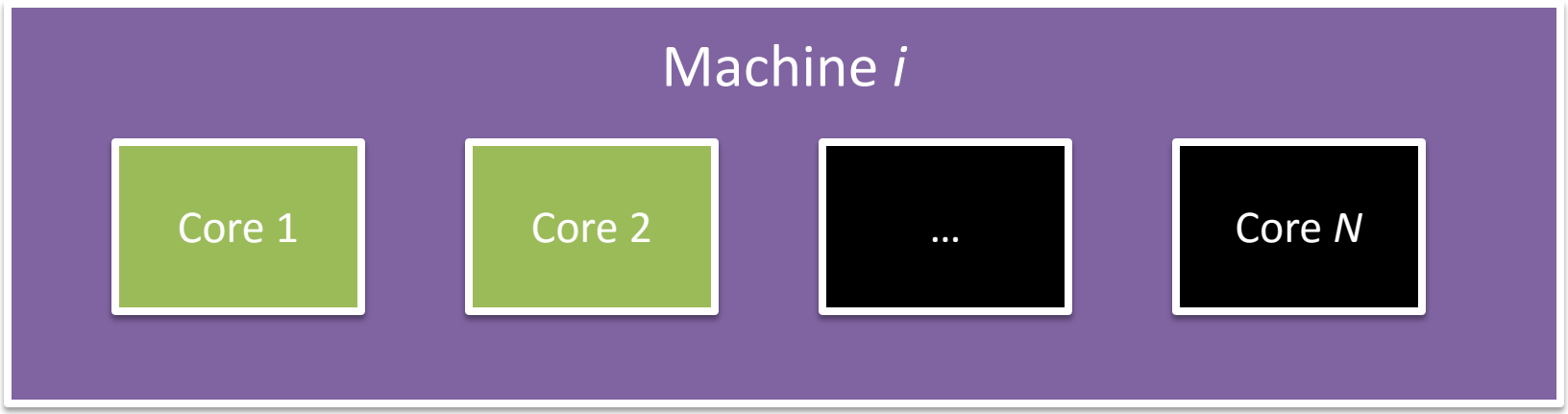


Core Scaling



- On how many cores should a workload execute?

Core Scaling



- Few workloads use every core effectively.
  - PARSEC achieves 90% of the speedup of 442 cores with only 35 cores.
  - [H. Esmaeilzadeh et al., ISCA, 2011]

Profiling Peak Power With Idle Cores

- Peak power is the largest aggregate power draw that lasts at least a tenth of a second.

- In an offline setting, workloads are run to completion
  - Power readings are taken
  - Multiple runs on each number of active cores
  - Very slow

Profiling Peak Power Online

- Common approaches to online profiling:
  - Use power readings from similar workloads
    - [C. Delimitrou et al., ASPLOS 2014]
  - K% or fixed time (10 minutes) sampling
    - [H. Nguyen et al., ICAC 2013]
- Accurate profiles are critical to schedulers and capacity planning.
  - [O. Sarood et al., Supercomputing 2014]

Our Contributions

- Empirical study of peak power on HPC
  - 5 observations about the interaction between architecture, workload, and power usage
- Novel approach to speedup peak power profiling with idle cores

Experimental Setup: Architectures

Values	Cores	L2 (KB)	L3 (MB)	Frequency (GHz)	TDP (W)	Size (nm)
Intel i7	4	256	8	3.4	95	32
Intel Sandy Bridge	8	256	20	2.6	115	32
Intel Xeon Phi	61	512	N/A	1.05	225	22

When cores idled, each used low-power modes:

- Intel i7: On-Demand CPU governor
- Intel Sandy Bridge: P-states
- Intel Xeon Phi: P-states and C3 package

Experimental Setup: Workloads

- 9 High Performance Computing kernels
- Each benchmark runs on 2^x cores.
- For NAS, size B is appropriate for 1 node.

Values	Features
CoMD	N-body molecular dynamics, Frequent inter-thread communication
Snap	PARTISN workload used at LANL, Iterative inter-thread communication
MiniFE	Finite-element workload composed from 4 parallelizable kernels
NAS benchmarks	Widely used benchmark suite (CG, EP, FT, IS, LU, MG)

Platform & Tools

Platform	Features
MPI	Message passing interface, limited support for shared memory
OpenMP (OMP)	Open Multi-Processing provides extensive shared memory support

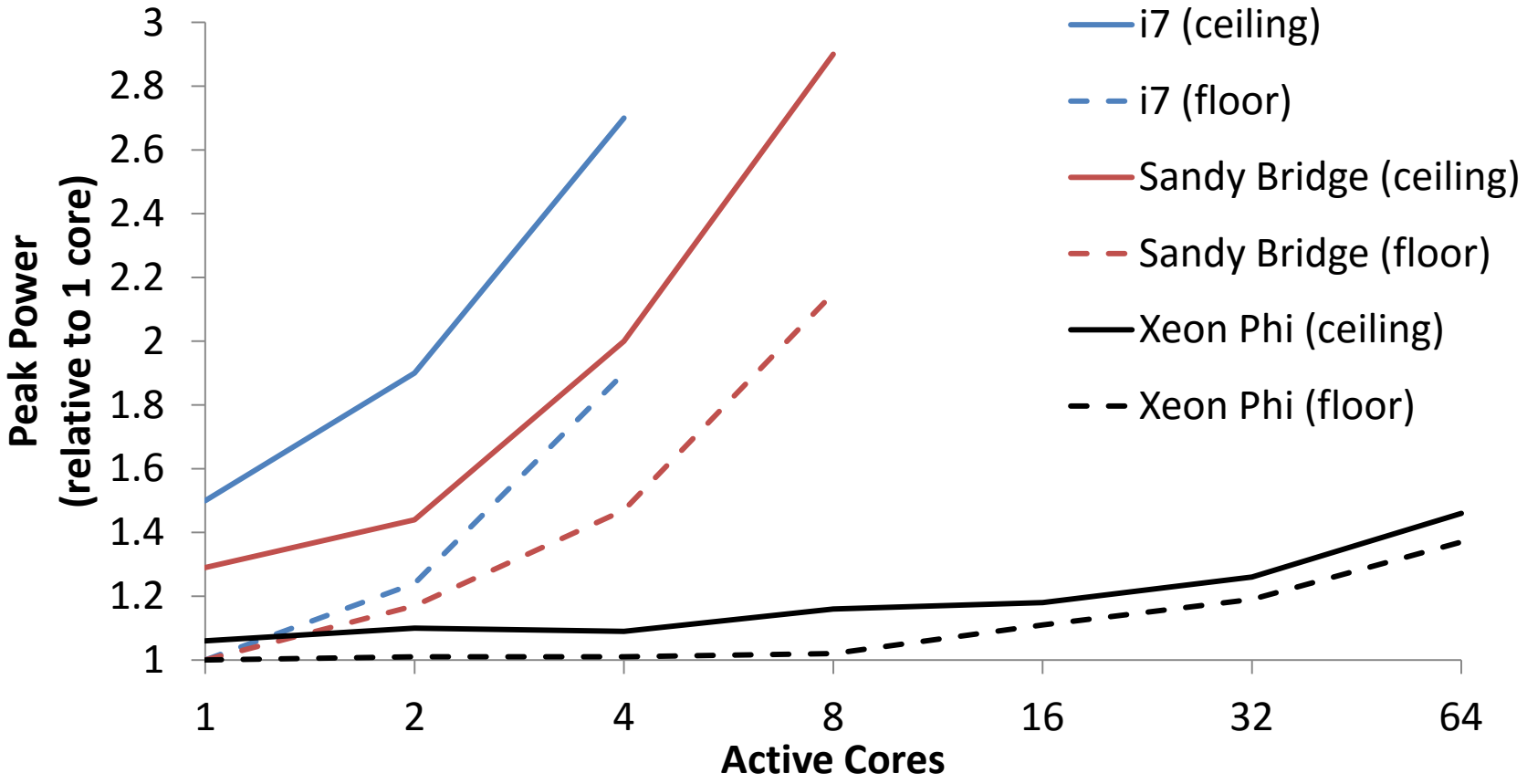
- We used thread pinning to ensure which cores were active.

- MPI & OMP exhibited similar peak power under micro-benchmarks (<2%)

Tools	Features
libRAPL	Reports on-core, uncore and total power readings per socket
likwid	Reports hardware counters related to data movement and total energy
Micsmc	Reports power readings on the Intel Xeon Phi Coprocessor

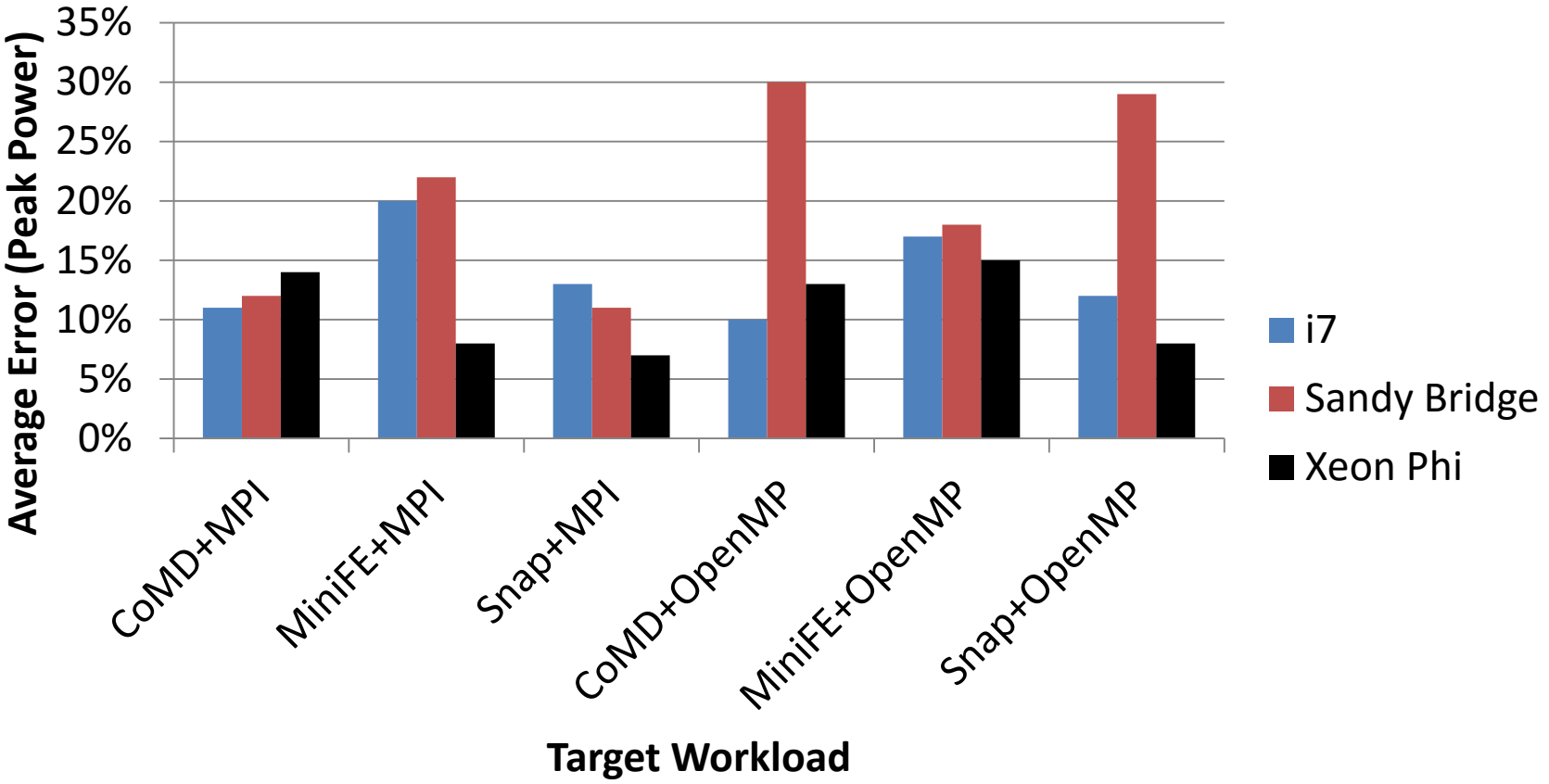
- We use data movement to explain the root causes for the effects of core scaling

Observation 1: Architectural Differences



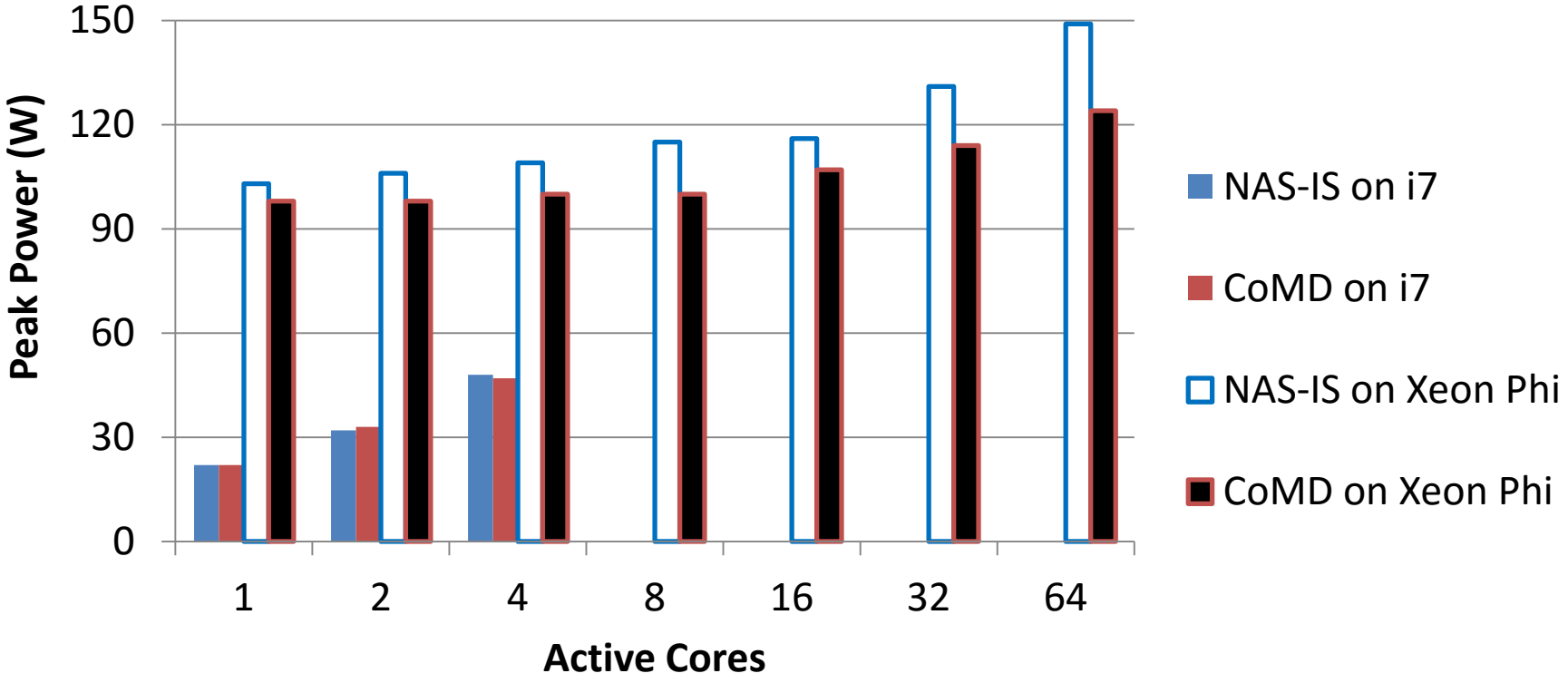
Different architectures observe significantly different relative increase in peak power as the number of active cores increases

Observation 2a: One Reference Workload



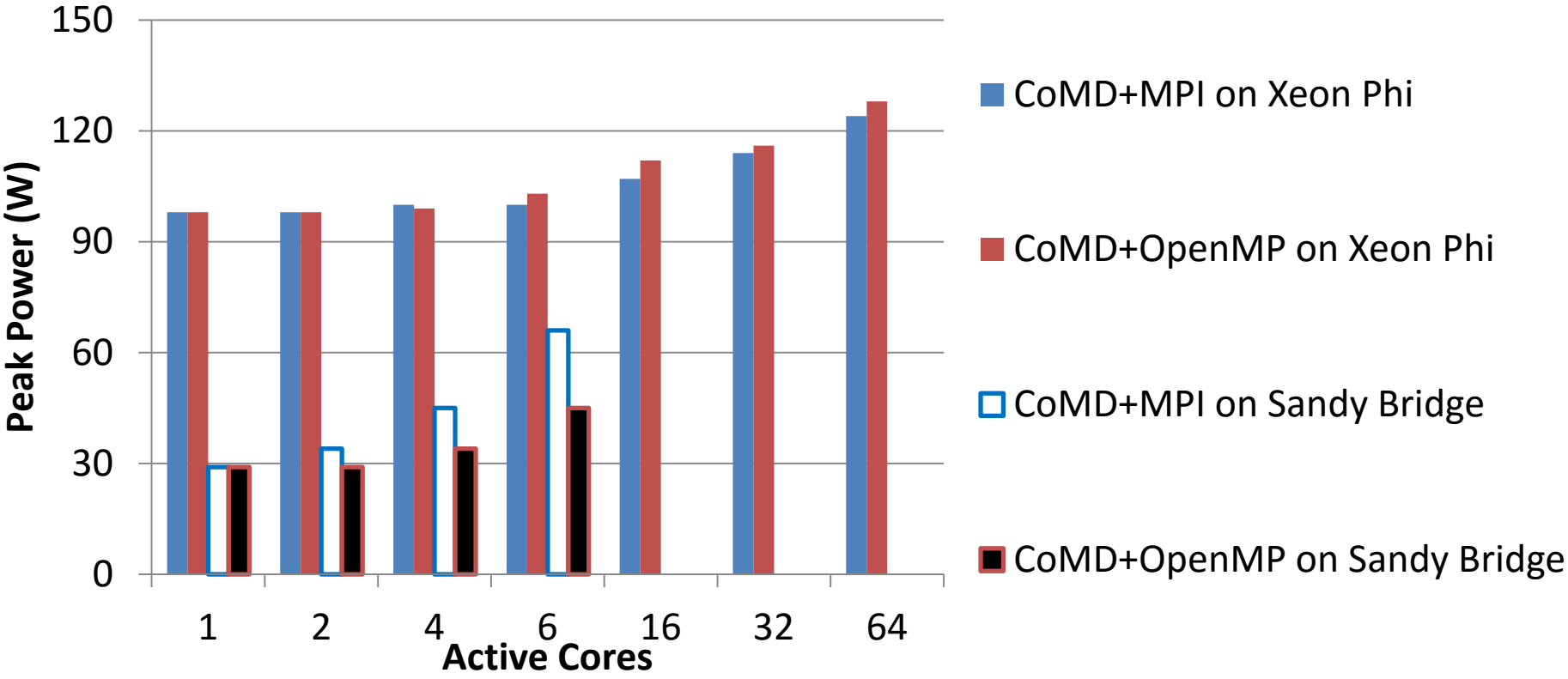
Using a randomly selected reference workload to predict peak power of other workloads can result in highly variable inaccuracy.

Observation 2b: Best Reference Workload



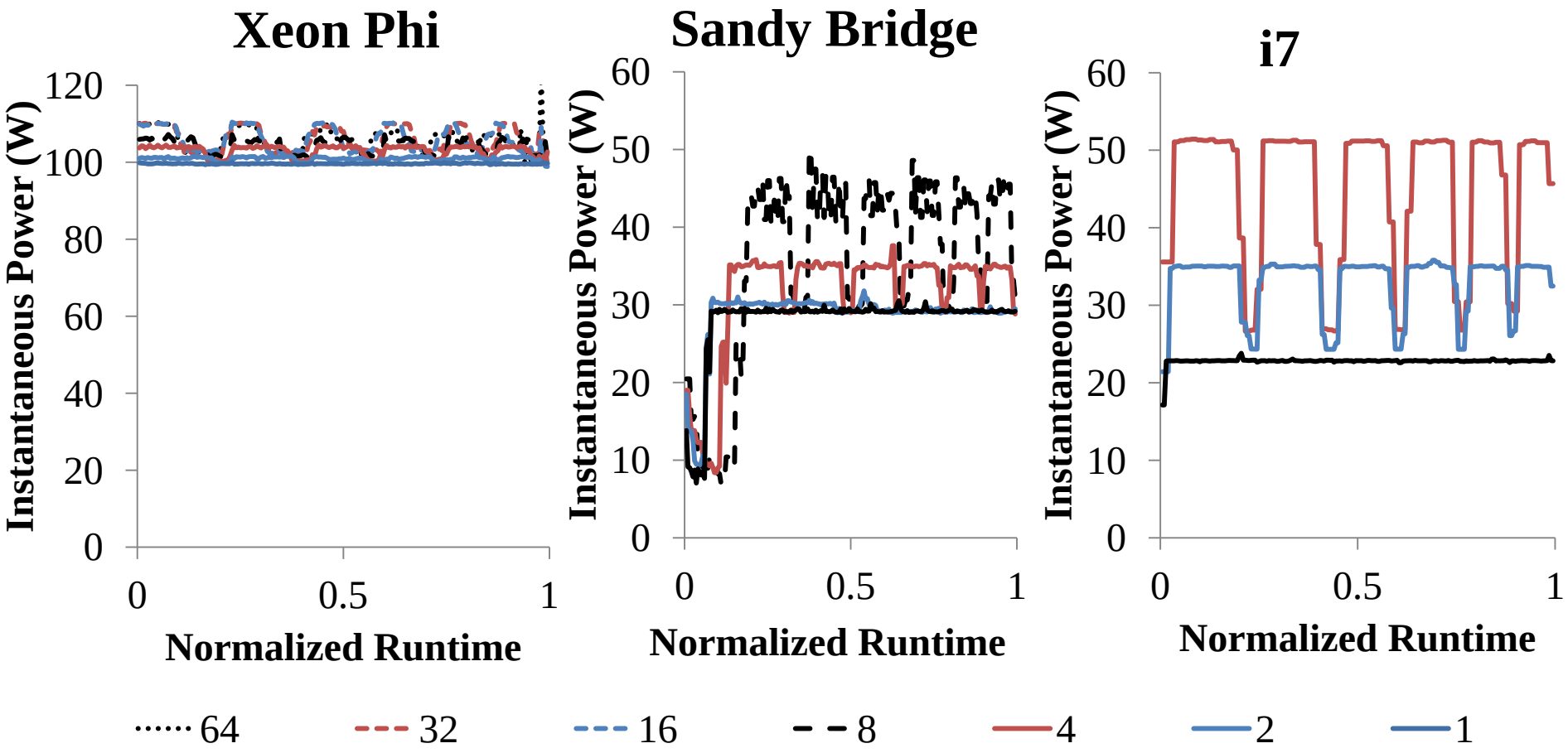
While NAS IS and CoMD have similar power on Intel i7, their peak power on Xeon Phi differ especially at high core counts.

Observation 2b: Best Reference Workload



CoMD MPI and OpenMP have similar peak power on Intel Xeon Phi, but their peak power on Intel Sandy Bridge differ especially at high numbers of active cores.

Observation 3: Power Phases (Snap)



Interestingly, for a given application the power profiles are similar across different numbers of active cores on a fixed architecture given normalized execution time.

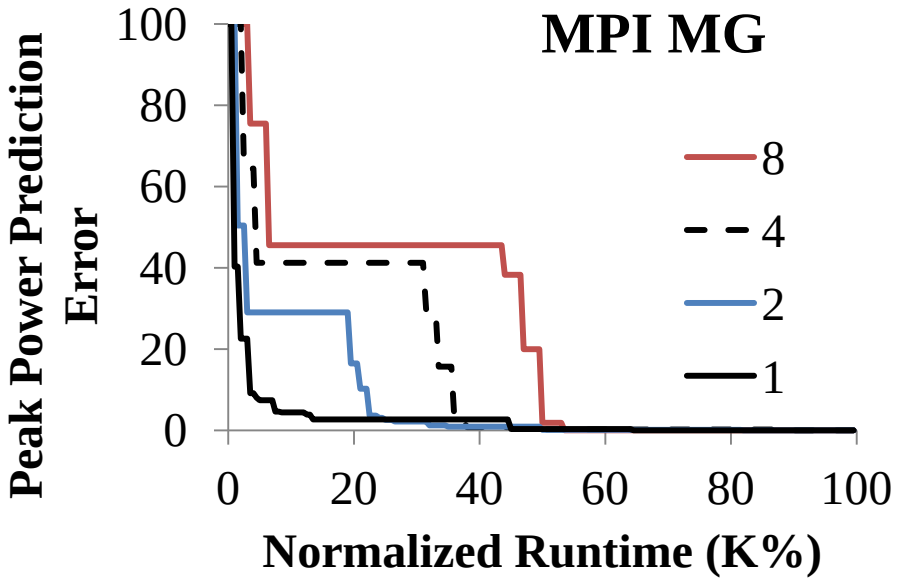
Our Contributions

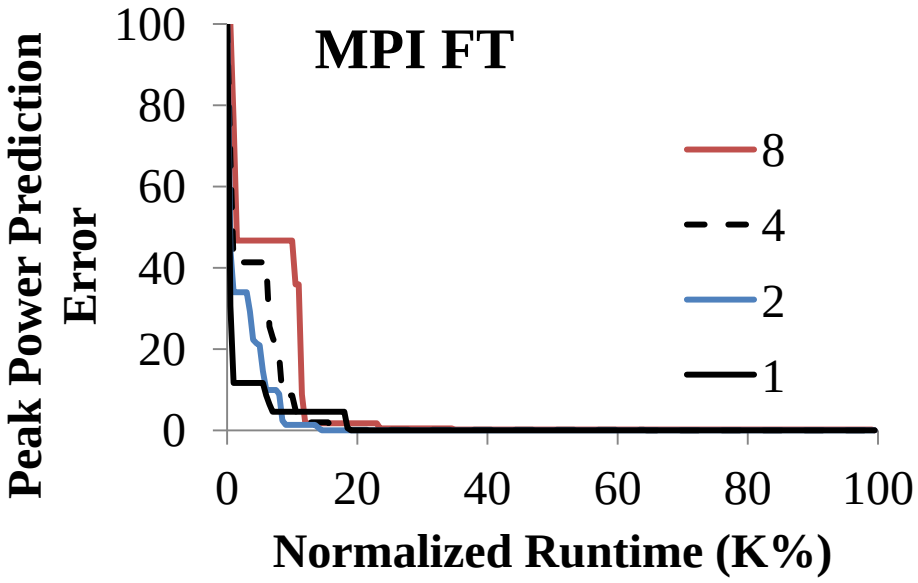
- Empirical study of peak power on HPC
  - 5 observations about the interaction between architecture, workload, and power usage

- Novel approach to speedup peak power profiling with idle cores

K% Sampling (Sandy Bridge Examples)

- Profiles for only K% of a workload





Workloads exhibit similar phases for different numbers of active cores.

Adaptive Power Profiling

- We can acquire power profiles faster than K% by taking advantage of these similar phases.

1. Profile the highest # of cores first.

2. Only profile at other cores until the most power hungry phase has occurred (or until K%, whichever is first).

Inputs

- Sampling Duration (k%)
- Inaccuracy (a%) i.e., expected error
- Active cores for initial profiling run
  - By default this is all available cores.

Output

- The duration of normalized time $\hat{k}\%$ to run the workload at other active core settings.

First Trace

1. Collect a power trace PP(i) for the given number of active cores, for k% of the workload.
2. For each point in PP(i), calculate PP_{max}(i):

$$PP_{\max}(i) = \max(PP(1),\ldots,PP(i))$$

3. We are guaranteed that the most accurate peak power over k% will be found at PP_{max}(k).
4. Calculate the expected error in estimating peak power PPEC(i):

$$PPEC(i) = \frac{PP_{\max}(k) - PP_{\max}(i)}{PP_{\max}(k)}$$

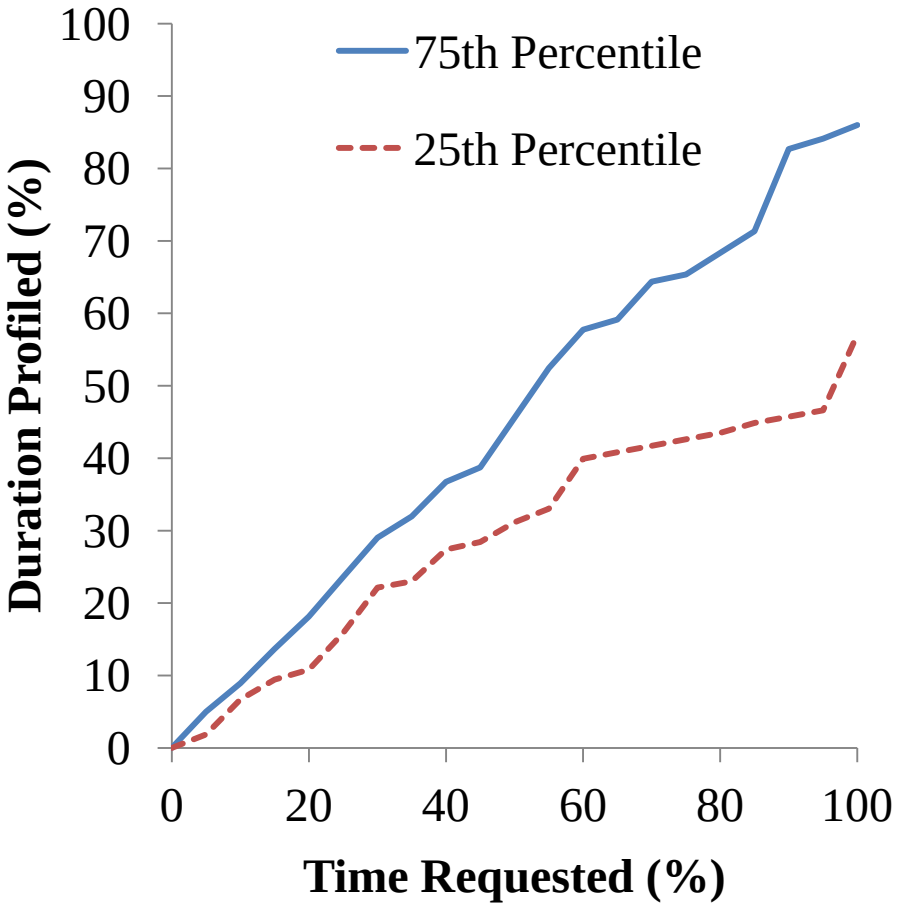
Algorithm

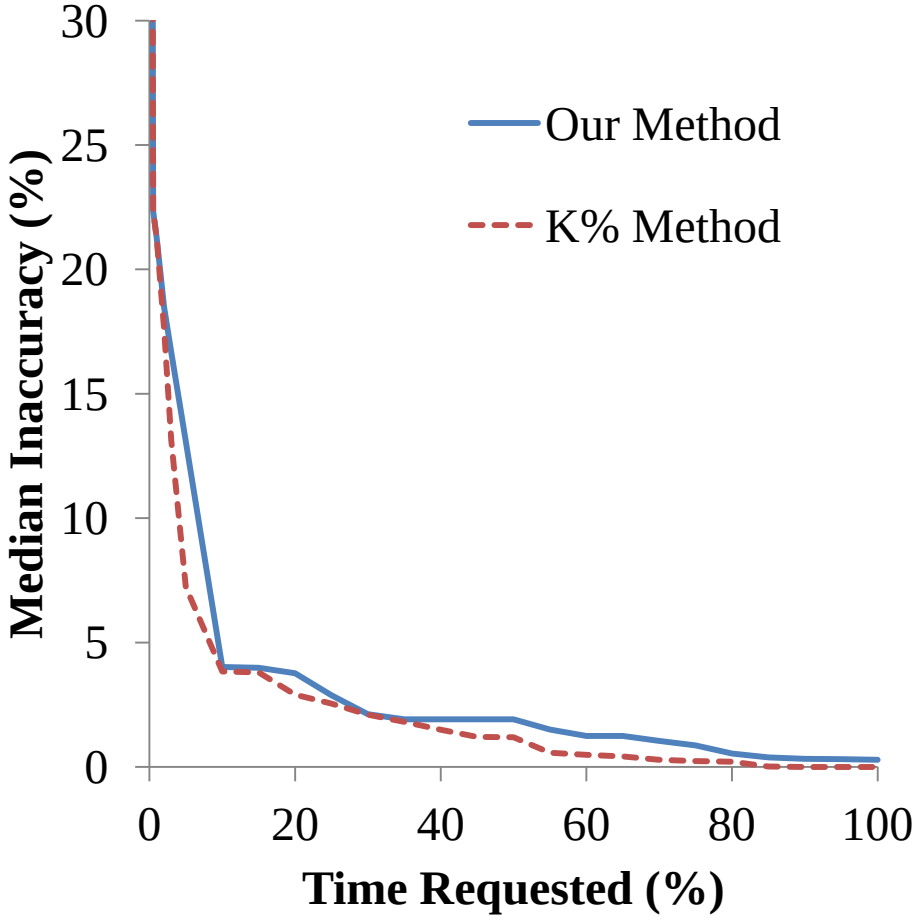
Data: PPEC[], k, a (expected inaccuracy)

Result: find sampling duration $\hat{k}\%$ to run at other active core settings

1. For i ← 0 to k do
2. if PPEC[i] < a then
3. return i;
4. end
5. end

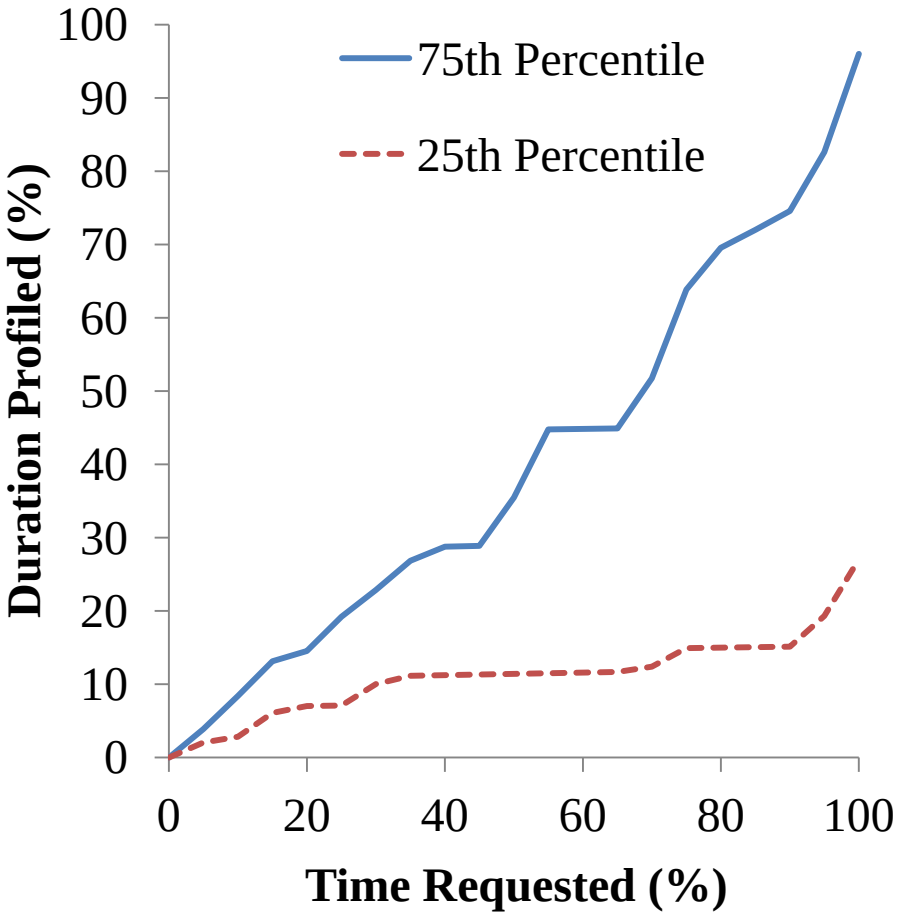
Varying Time Requested (i7)

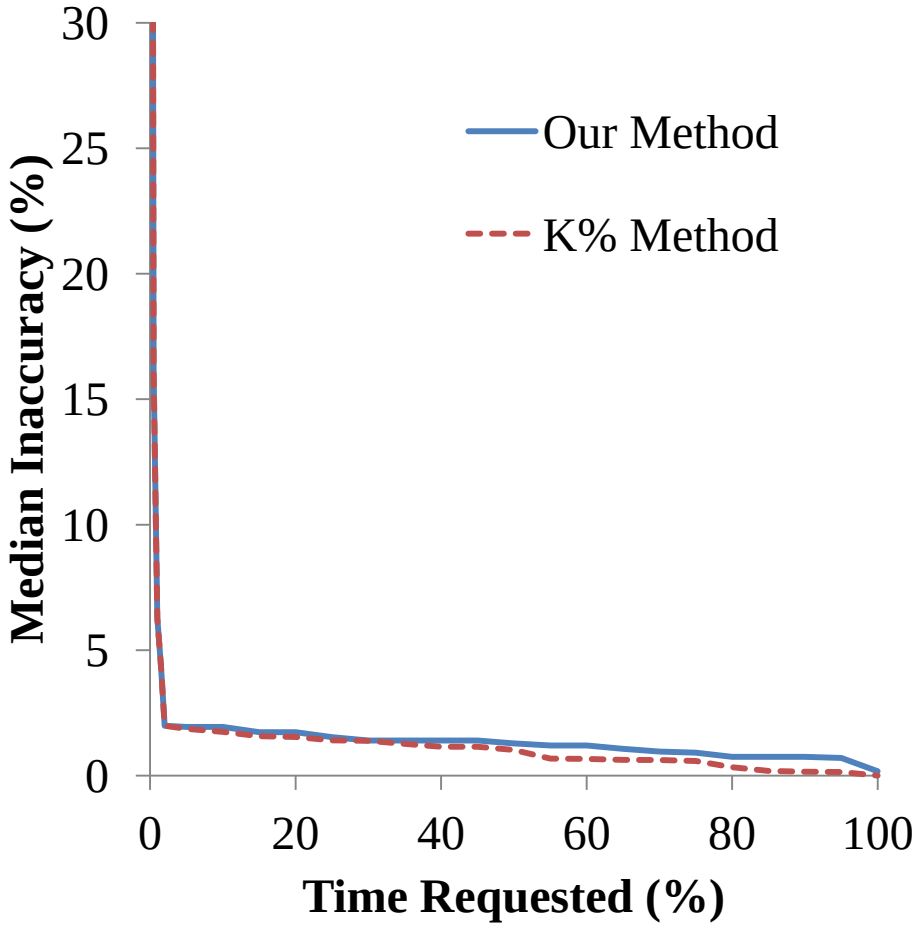




On the Intel i7, we saved up to 73% of total profiling duration, 11% on average.
On average, inaccuracy stays within 1.7% of *k*% sampling.

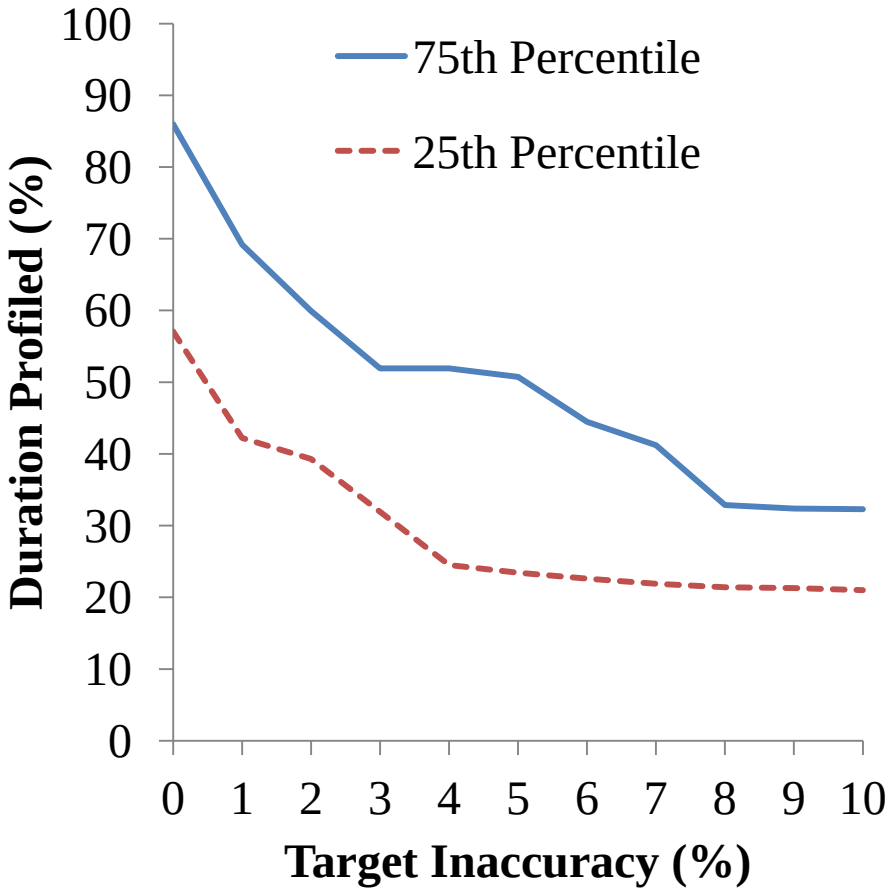
Varying Time Requested (Xeon Phi)

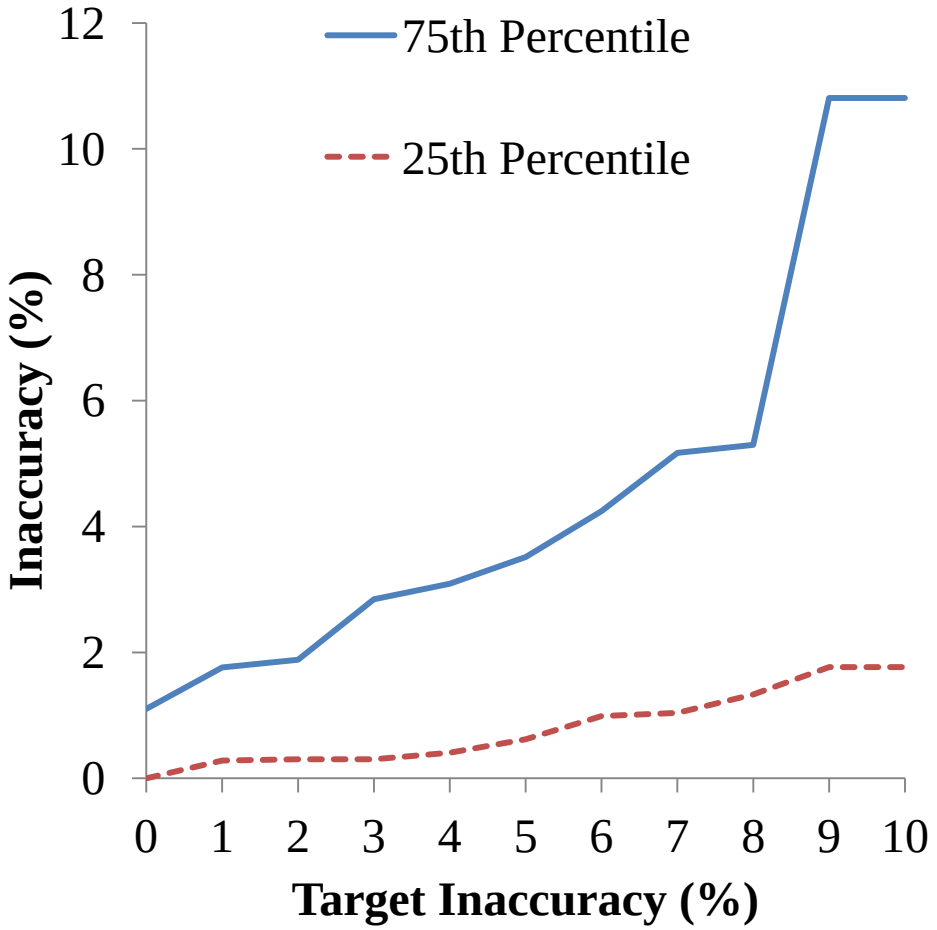




On the Intel Xeon Phi, we saved up to 93% of total profiling duration, 25% on average. On average, inaccuracy stays within 0.3% of *k*% sampling.

Requesting Inaccuracy (i7)





On the Intel i7, we saved more than 40% of total profiling duration in the median case. For target inaccuracy above 7%, the 75th percentile of workloads was above 5%.

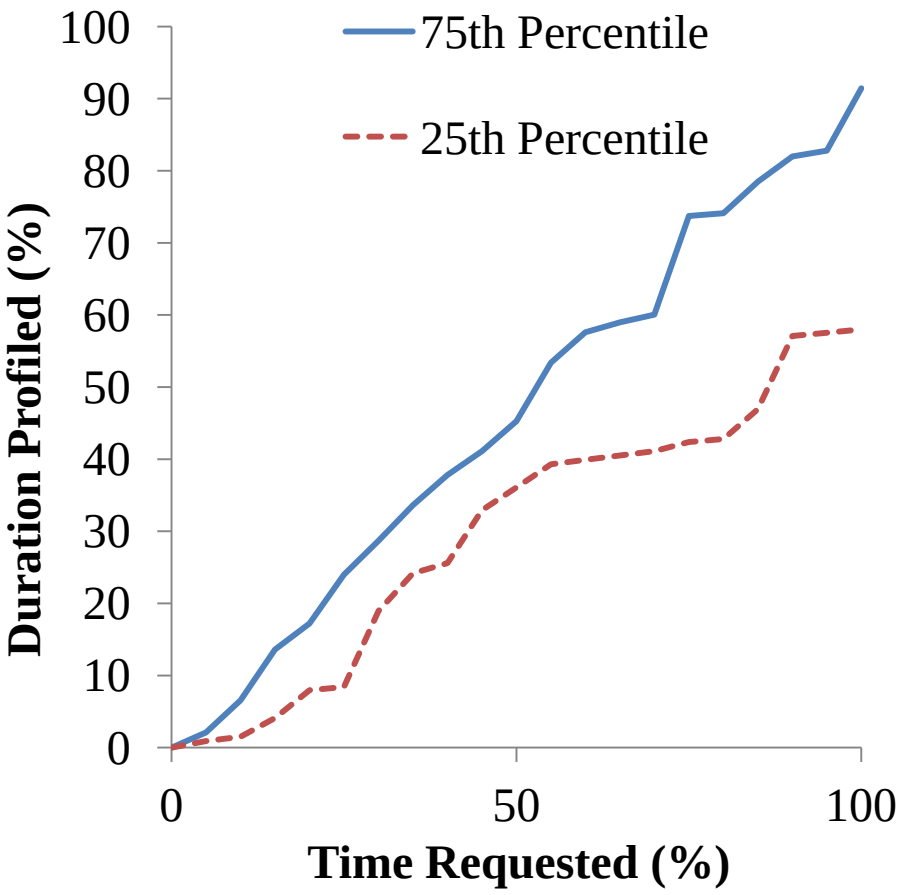
Conclusion

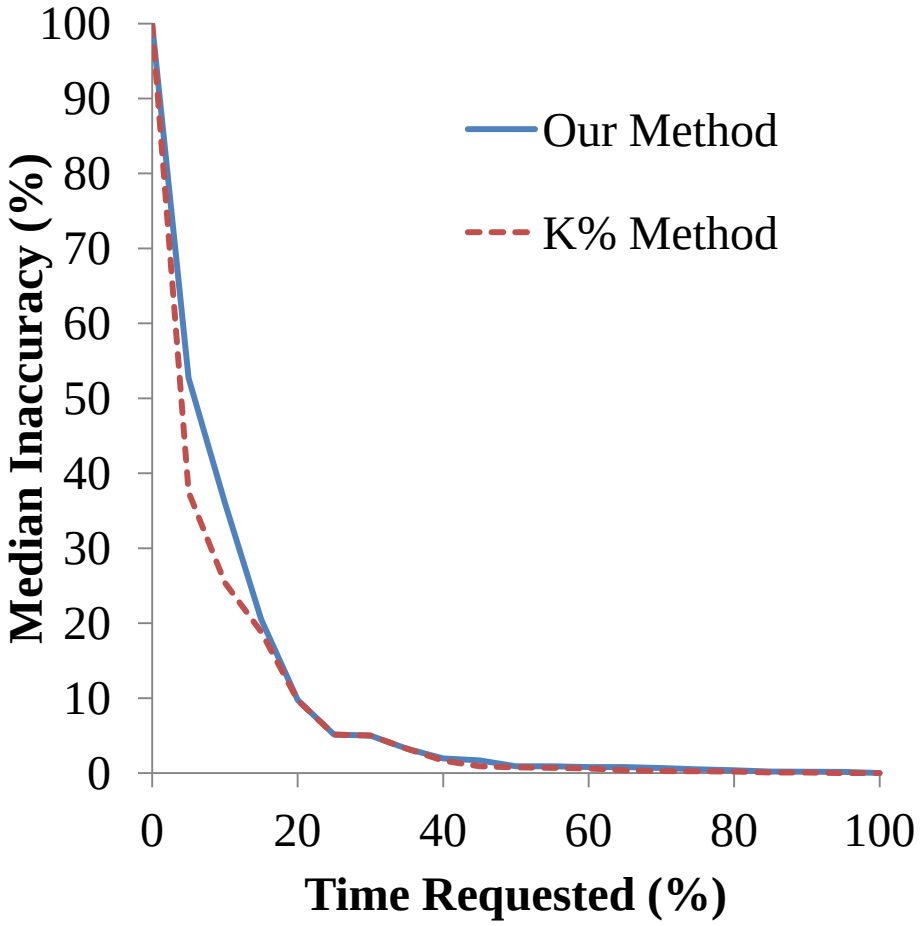
- Workloads exhibit similar stages under normalized runtime with different active cores.

- Our algorithm uses this observation to save time profiling for peak power online.
  - We can save up to 93% of profiling time while keeping inaccuracy within 3% on average.

Questions?

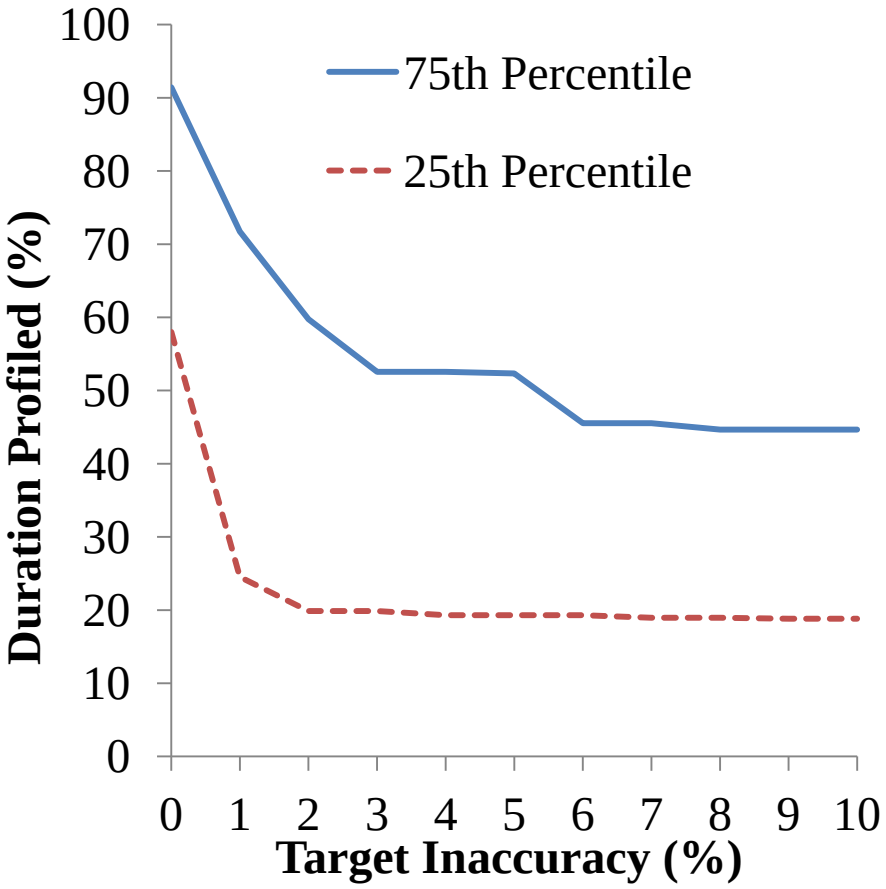
Varying Time Requested (Sandy Bridge)

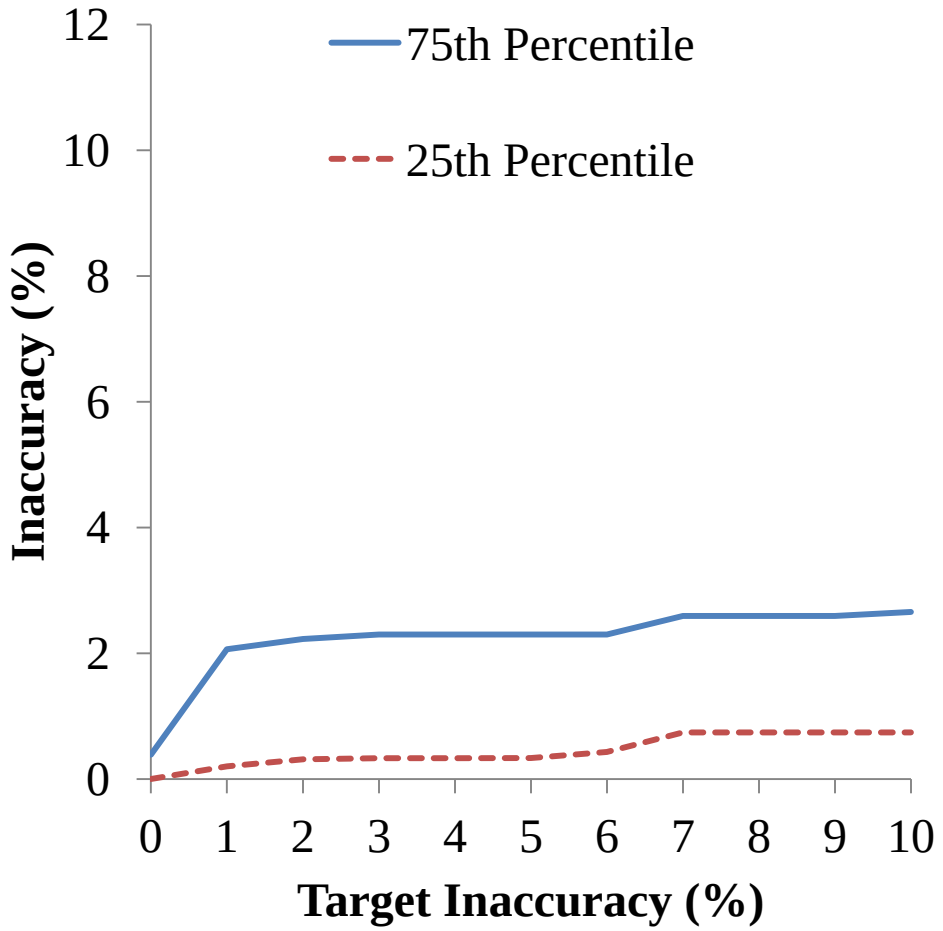




On the Intel Sandy Bridge, we saved up to 60% of total profiling duration, 12% on average. On average, inaccuracy stays within 3% of *k*% sampling.

Requesting Inaccuracy (Sandy Bridge)





With k=100% and inaccuracy > 2% our method used a profiling duration less than 60%.