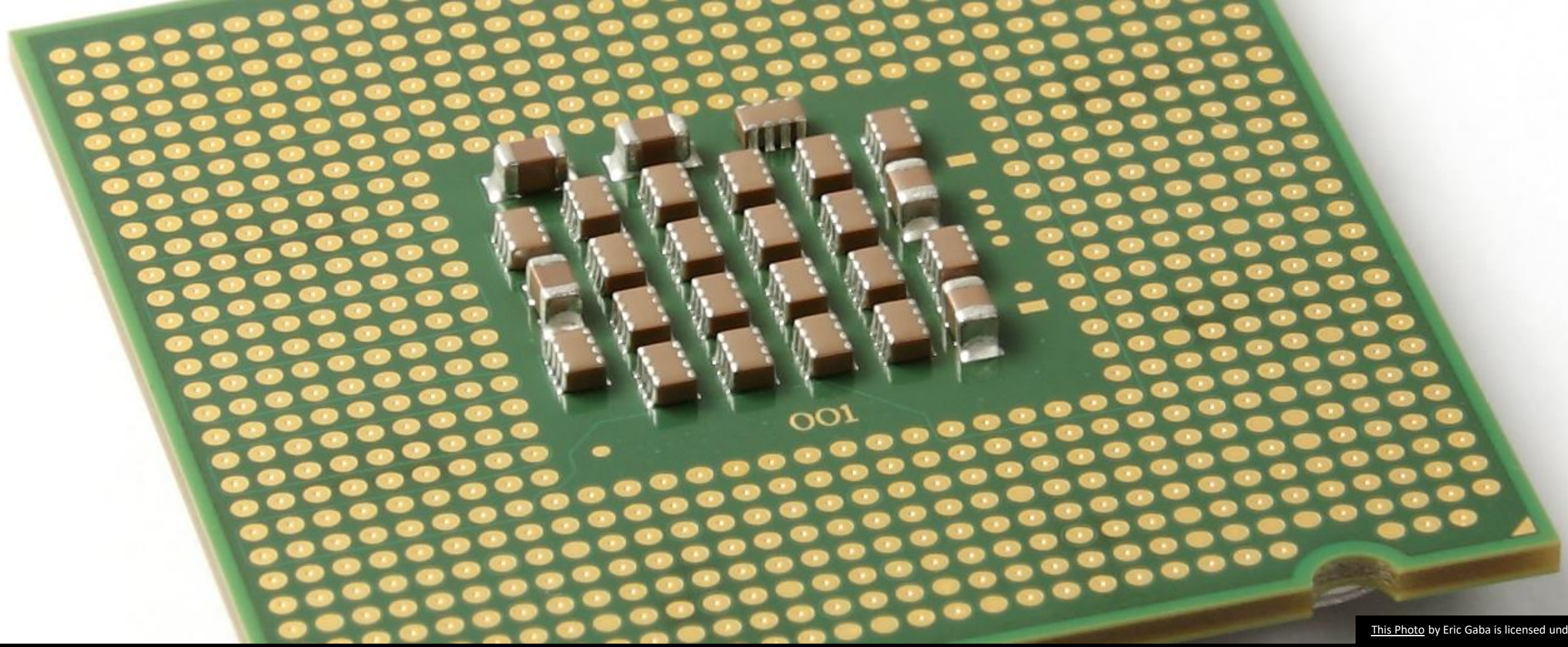


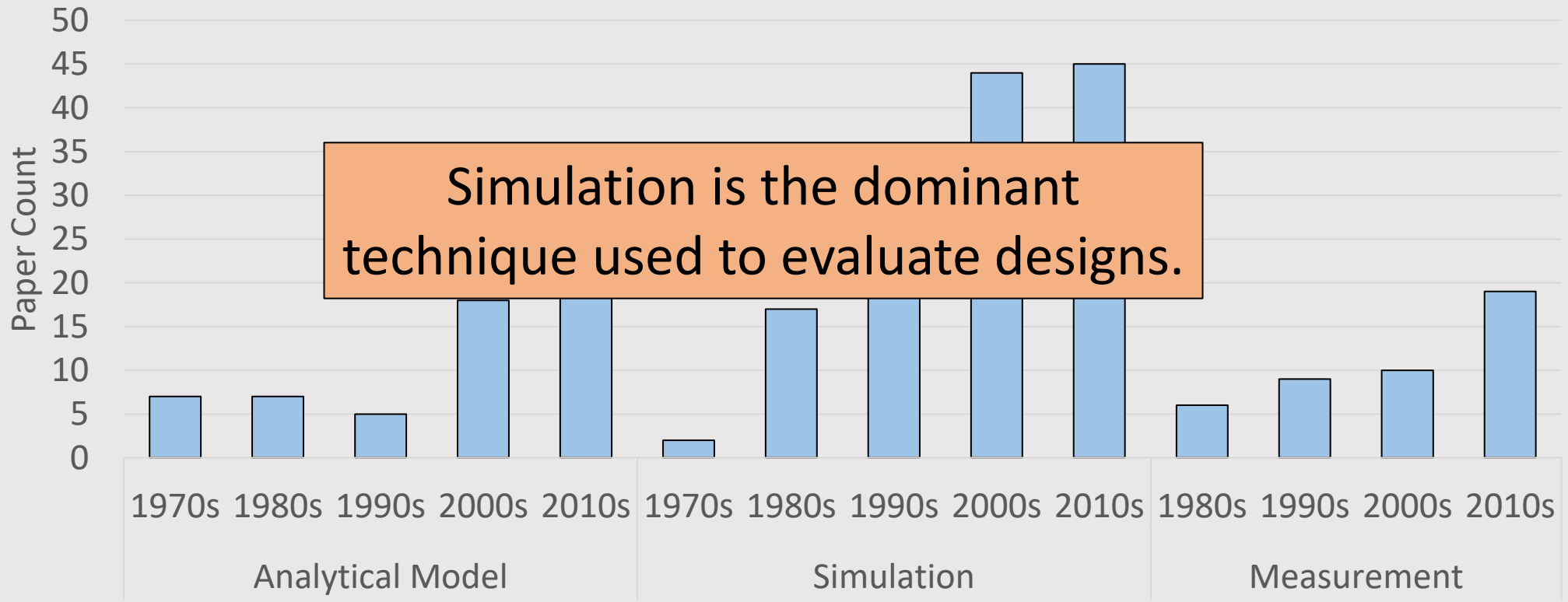
New Tools for Evaluating Parallel and Heterogeneous Architectures

Mario Badr

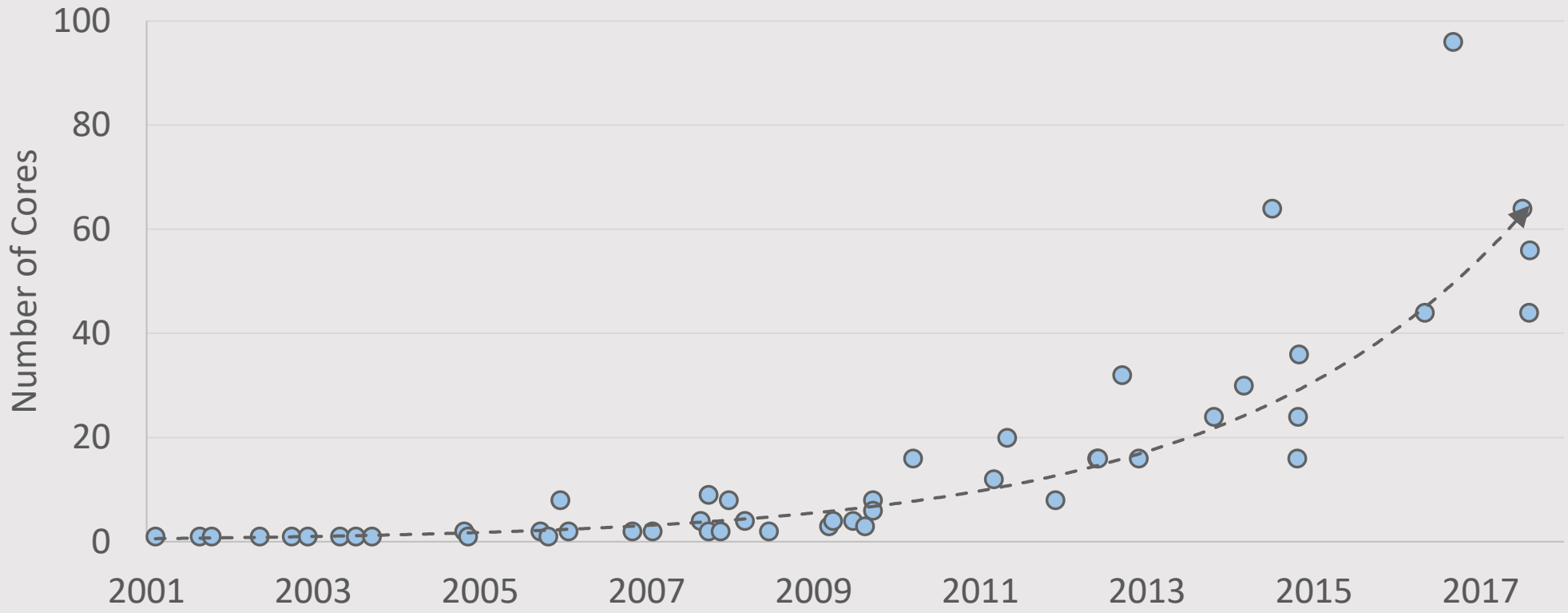
PhD Departmental Oral Exam

Supervisor: Natalie Enright Jerger

Evaluation Techniques Used (1973 – 2017)

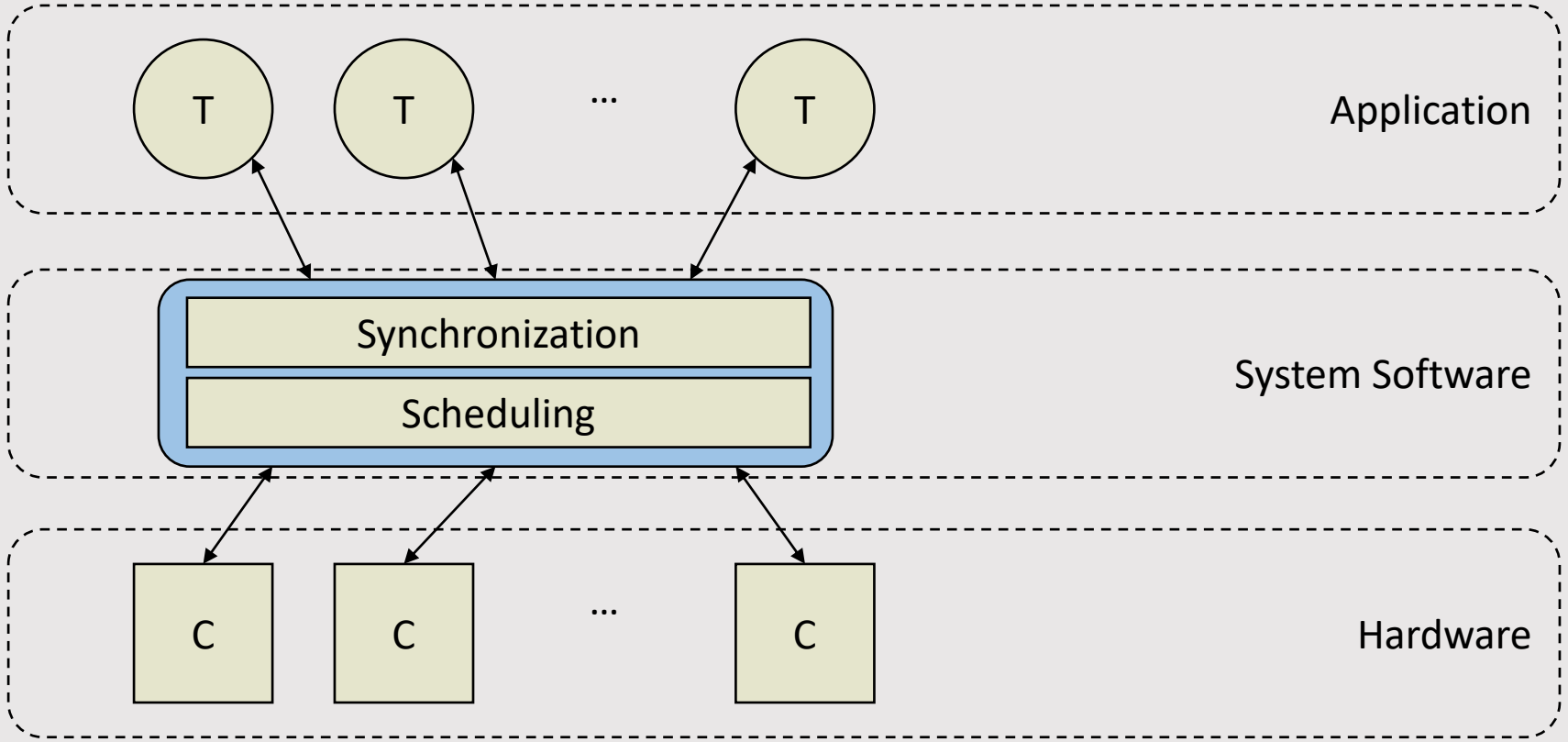


Core Counts are Increasing



https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

Parallelism Impacts All Layers



Key Contribution for Parallel Architectures

Trend

An increasing number of cores.

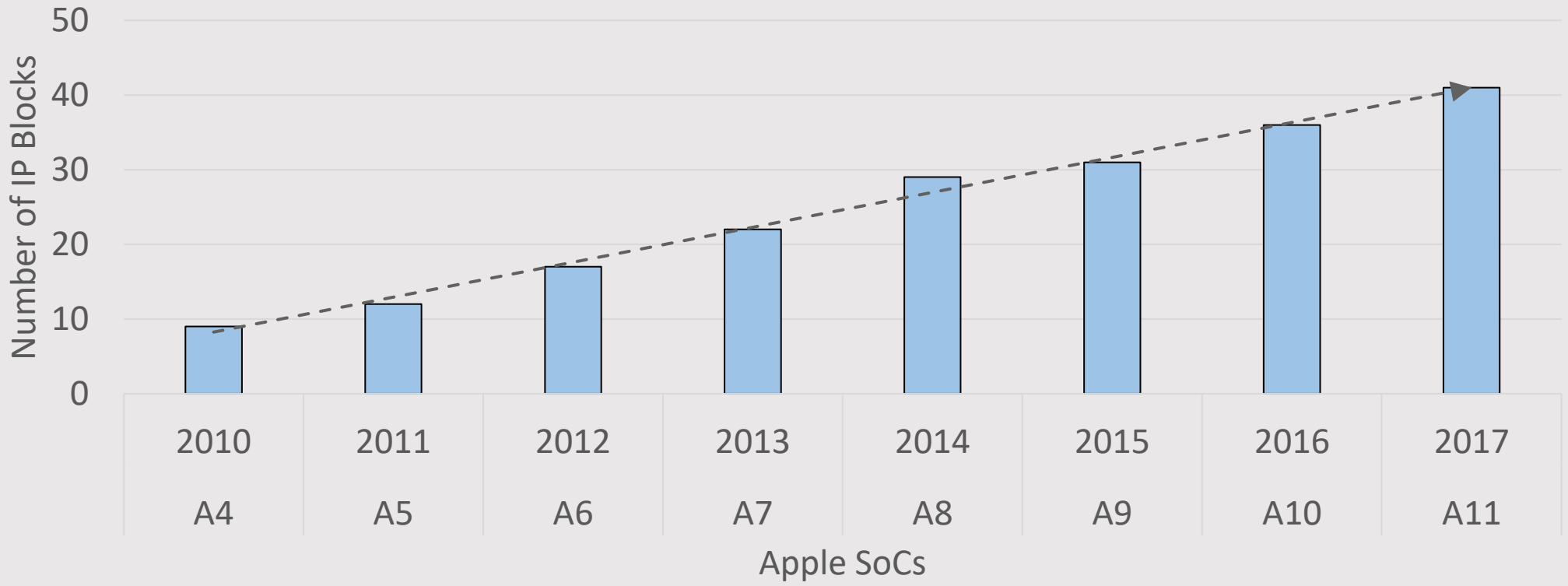
Challenge

Performance is difficult to predict.

Solution

Rhythm: Find the critical path of events in a workload.

Heterogeneity is Increasing



Y. S. Shao, B. Reagen, G. Wei, and D. Brooks, "The Aladdin approach to accelerator design and modeling," IEEE Micro, 2015.

Key Contribution for Heterogeneous Architectures

Trend

An increasing amount of heterogeneity.

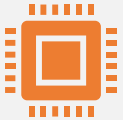
Challenge

The compute devices are proprietary.

Solution

Mocktails: Black-box modeling of compute devices.

Presentation Outline



Background

- Heterogeneity
- Memory hierarchy
- Statistical simulation

Mocktails

- Modeling Requests
- Uncovering patterns
- Synthesizing

Evaluation

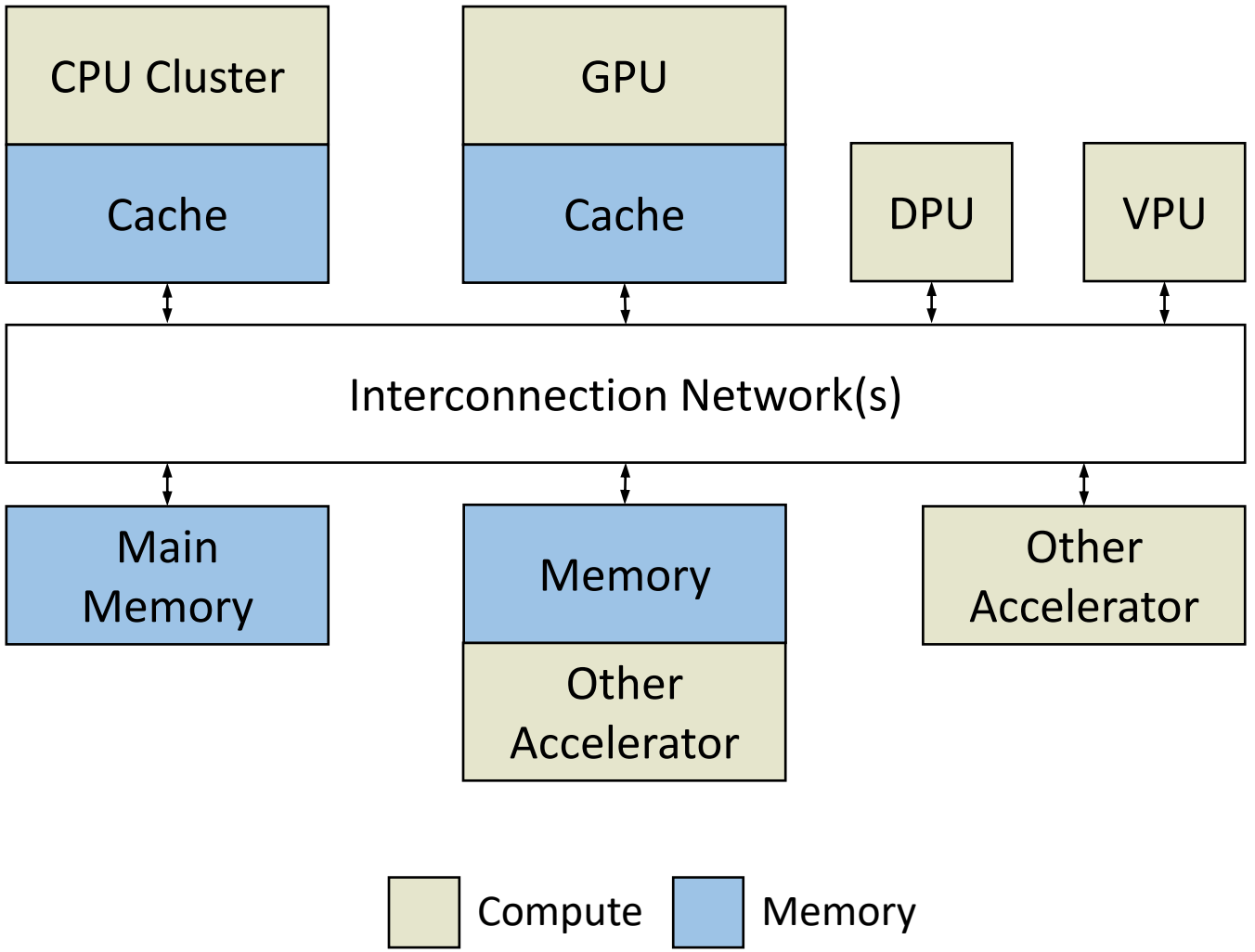
- Memory controller

Conclusion

- Summary of results
- Future directions

A System-on-Chip

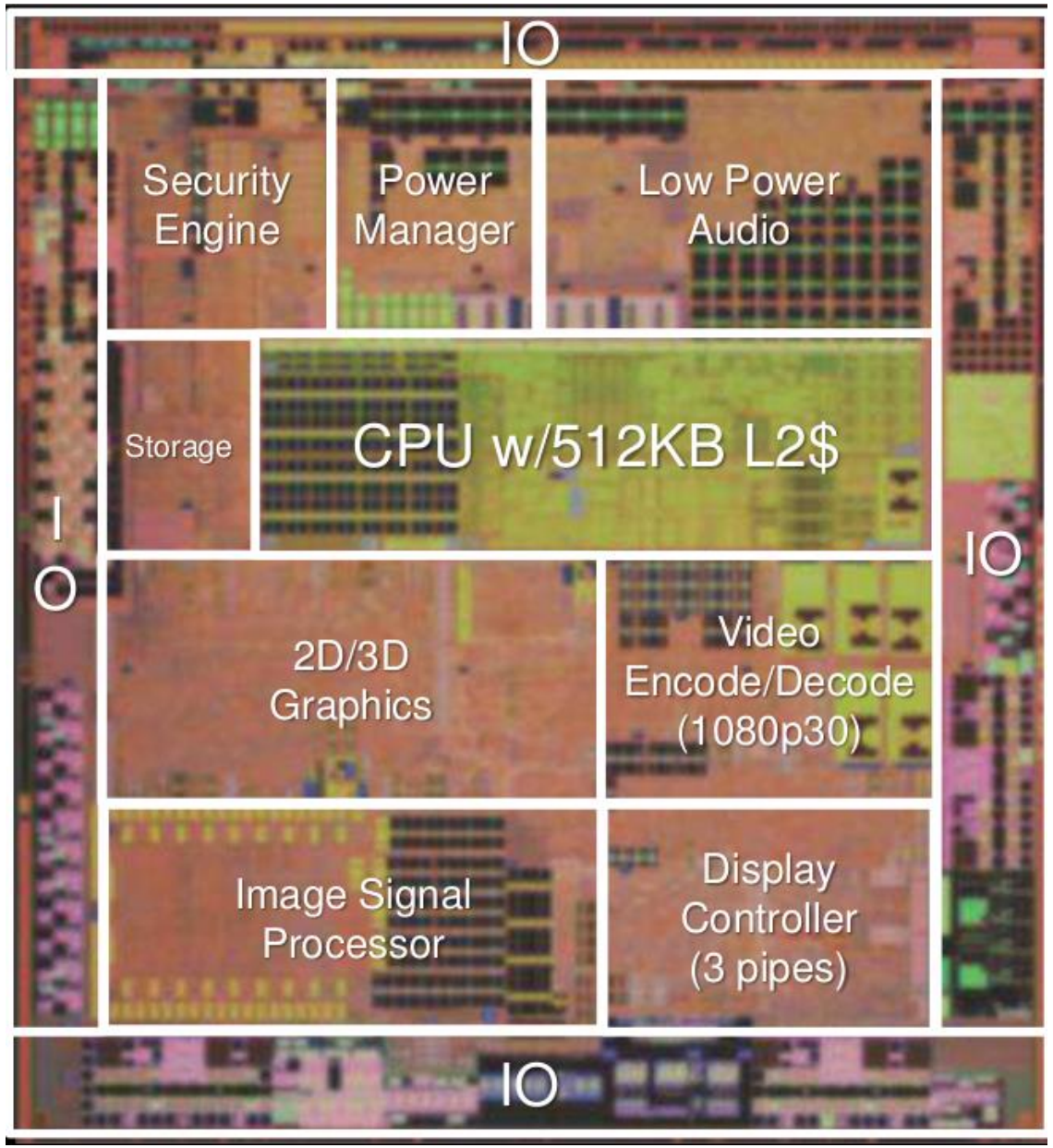
- General-purpose cores (CPU)
- Graphics-Processing Unit (GPU)
- Display-Processing Unit (DPU)
- Video Processing Unit (VPU)



The Intel Penwell SoC

- Mobile SoC from 2012
- 32 nm technology node
- 6 specialized architectures

Significant real estate allocated to accelerators.



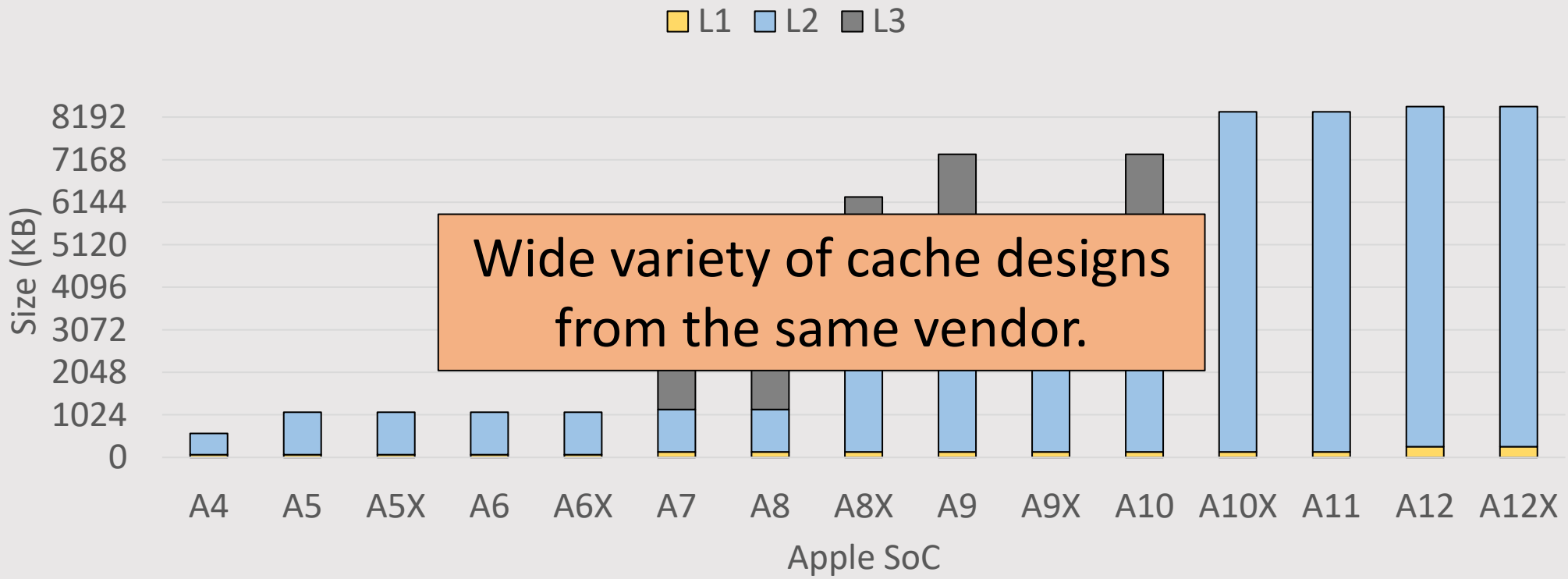
https://doi.org/10.1109/HOTCHIPS.2012.7476498

Heterogeneous Systems-on-Chip

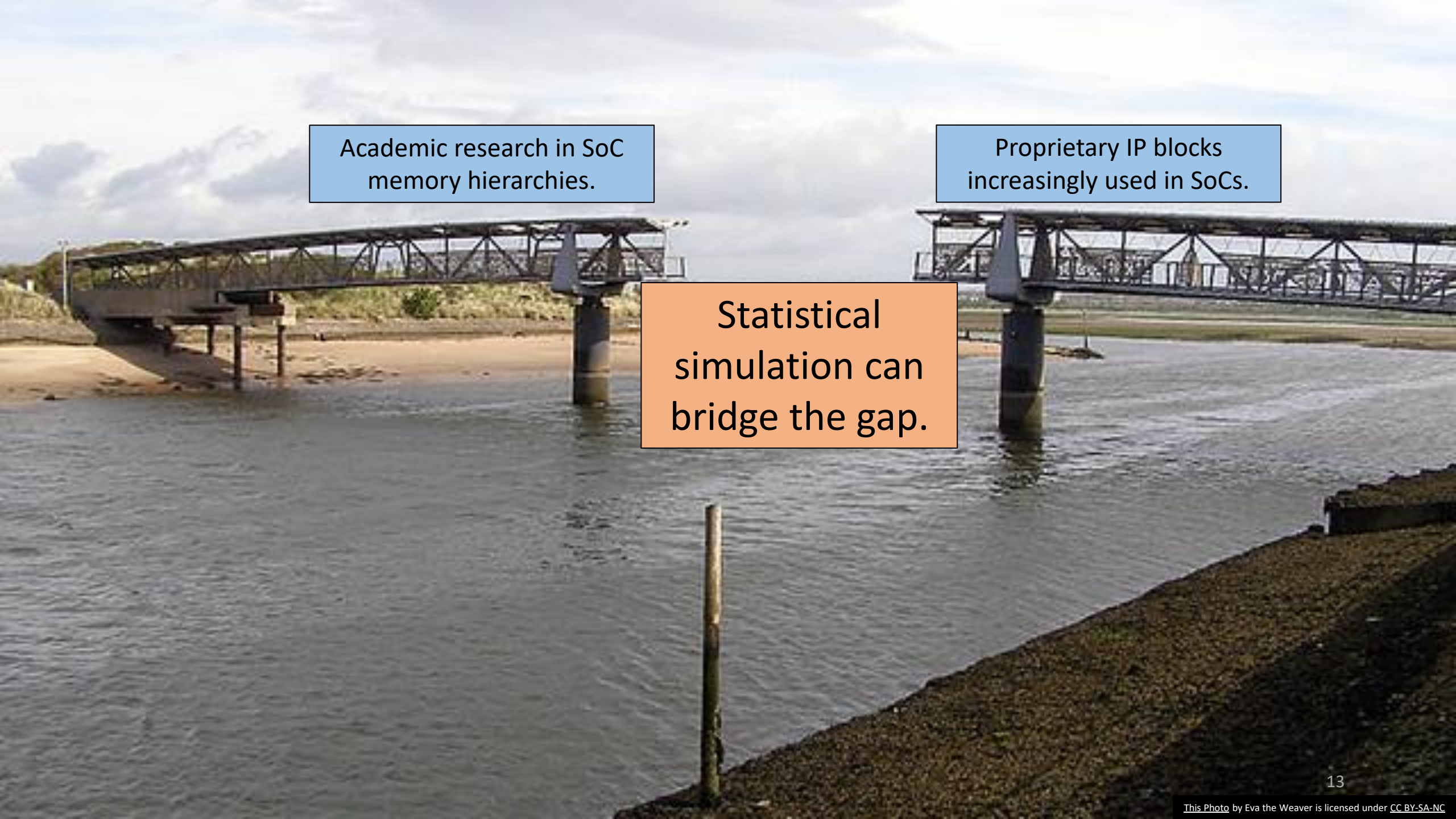
- Specialized hardware for commonly used workloads
- Spend area to b iciency
- More IP blocks = varying demands on memory

How do we evaluate the memory hierarchy?

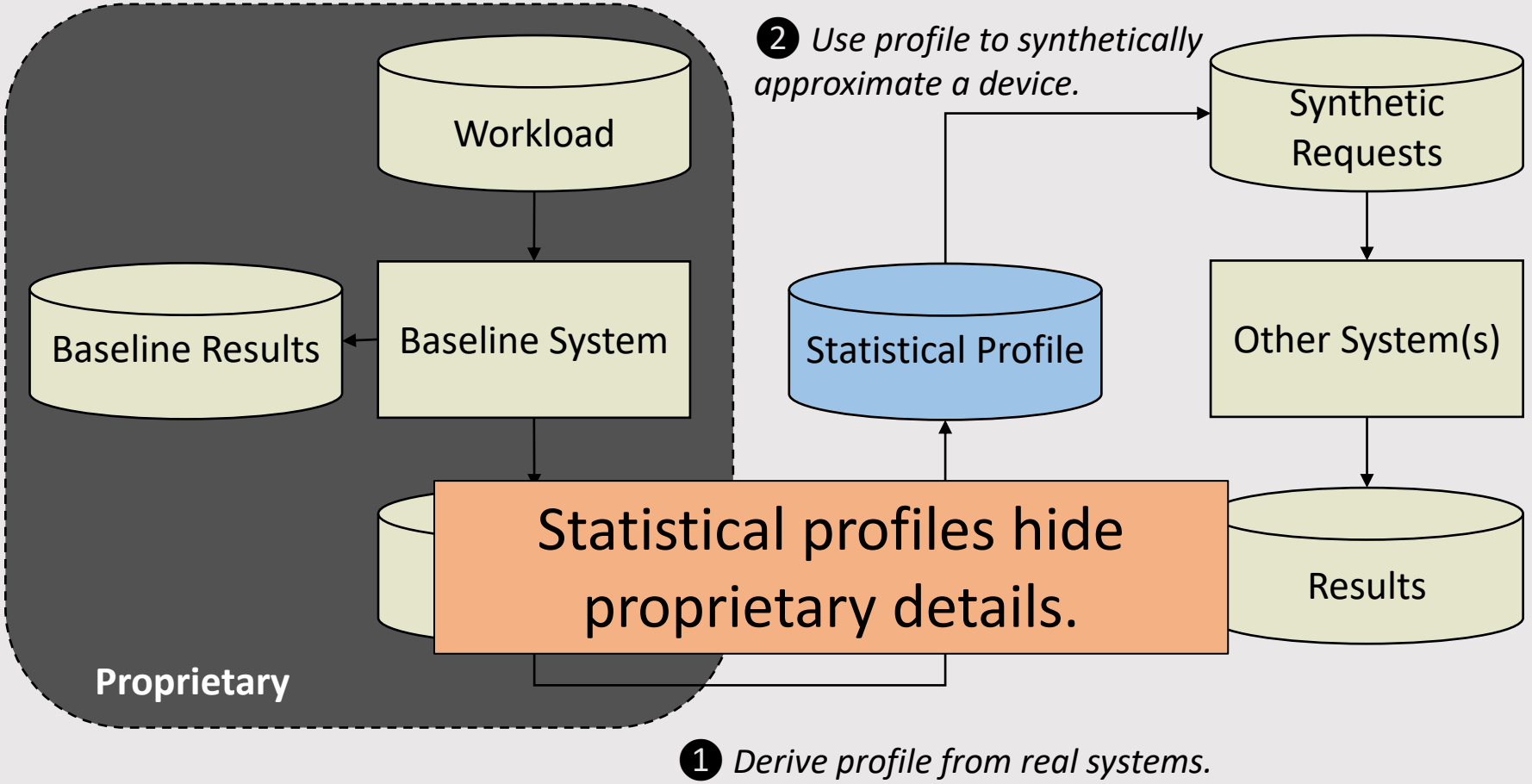
Apple SoCs – The Cache Hierarchy



https://en.wikipedia.org/wiki/Apple-designed_processors



Statistical Simulation

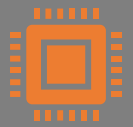


Prior Statistical Simulation Techniques

Technique	Timestamp	Address	Operation	Size
WEST	✕			✕
STM				
MeToo				
SLAB				
HRD	✕			
HALO				

Prior techniques are well-tuned for CPU architectures, but won't work for heterogeneous compute devices.

Up Next: Mocktails



Background

- Heterogeneity
- Memory hierarchy
- Statistical simulation

Mocktails

- Modeling Requests
- Uncovering patterns
- Synthesizing

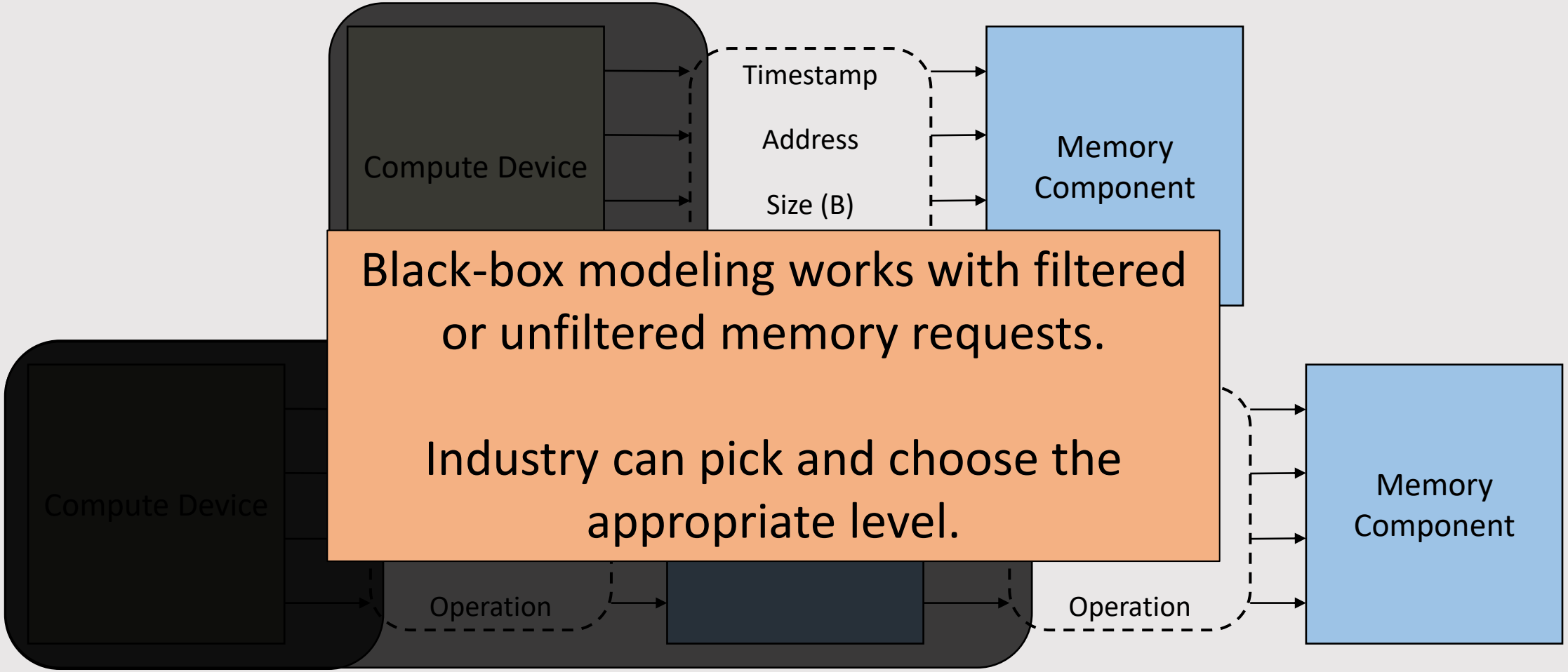
Evaluation

- Memory controller

Conclusion

- Summary of results
- Future directions

Modeling a Memory Request

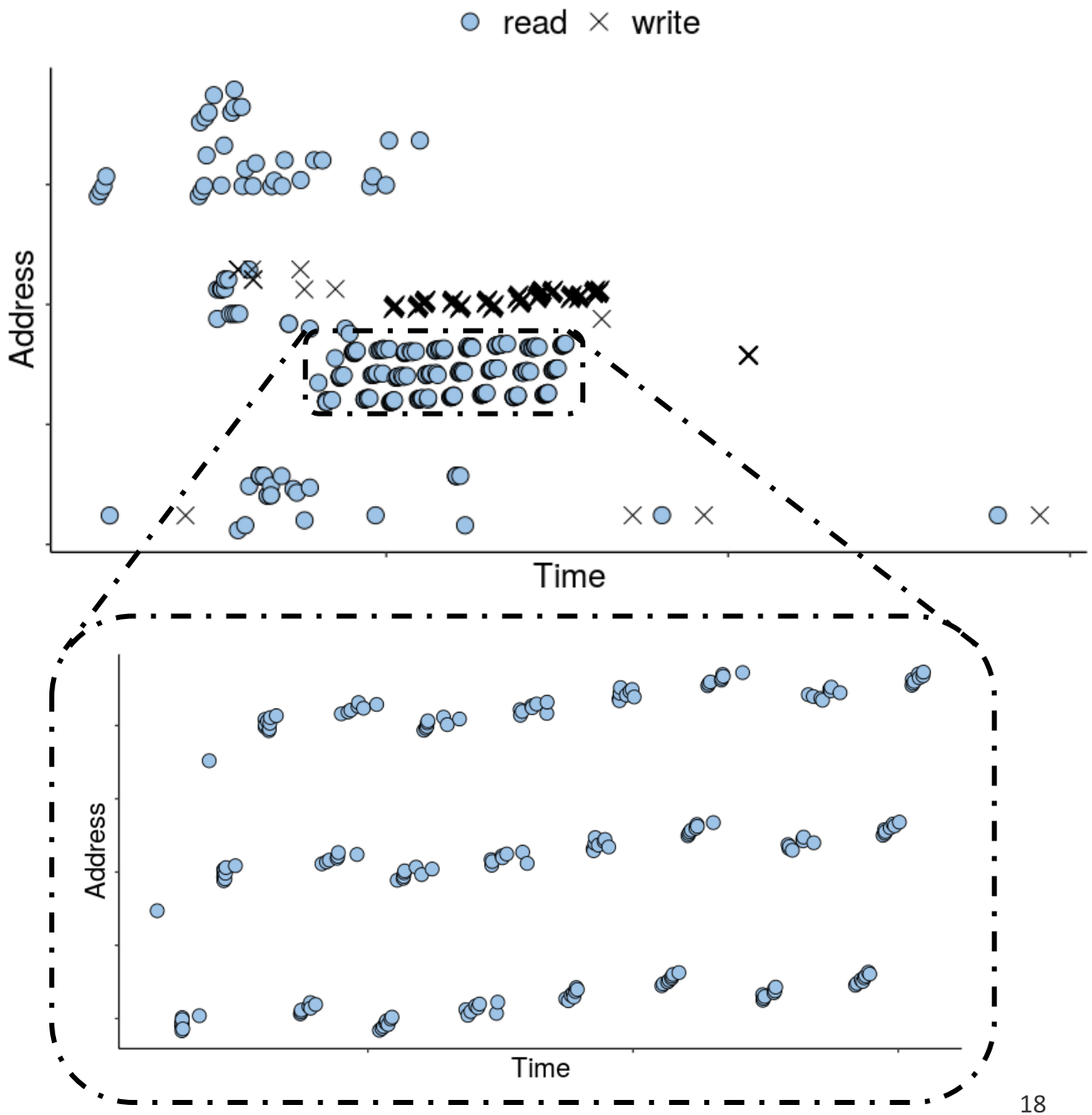


Black-box modeling works with filtered or unfiltered memory requests.

Industry can pick and choose the appropriate level.

An Example Workload

- Lots of variability
- Hard to find a pattern in the memory accesses
- We can zoom in



Modeling Addresses

- Given a starting address, model th strides
- Stride models used in prior art

Time	Address	Stride
1	D	
2	A	-12
3	C	8
4	B	-4
		4
		0
		16896
8	X	-8
9	Y	4
10	A	-16884
11	Y	16884
12	B	-16880

High variability in strides is difficult to model accurately.

Temporal Partitioning

- Divide the sequence of requests in two
  - Two starting add
  - Two stride mode
- Each partition has different behaviour
- Temporal partitioning used in prior art

	Time	Address	Stride
Time Interval 1	1	D	
	2	A	-12
	3	C	8
	4	B	-4
			4
			0
Time Interval 2	8	X	-8
	9	Y	4
	10	A	-16884
	11	Y	16884
	12	B	-16880

Time interval 2 has high variability in stride values.

Spatial Partitioning

- Divide requests into separate address ranges
- Each partition has different behaviour
- Spatial partitioning used in prior art
  - But tuned for CPUs

Spatial Partition 1

Time	Address	Stride
1	D	
2	A	-12
3	C	8
4	B	-4
5	C	4
		0
		-8
		4

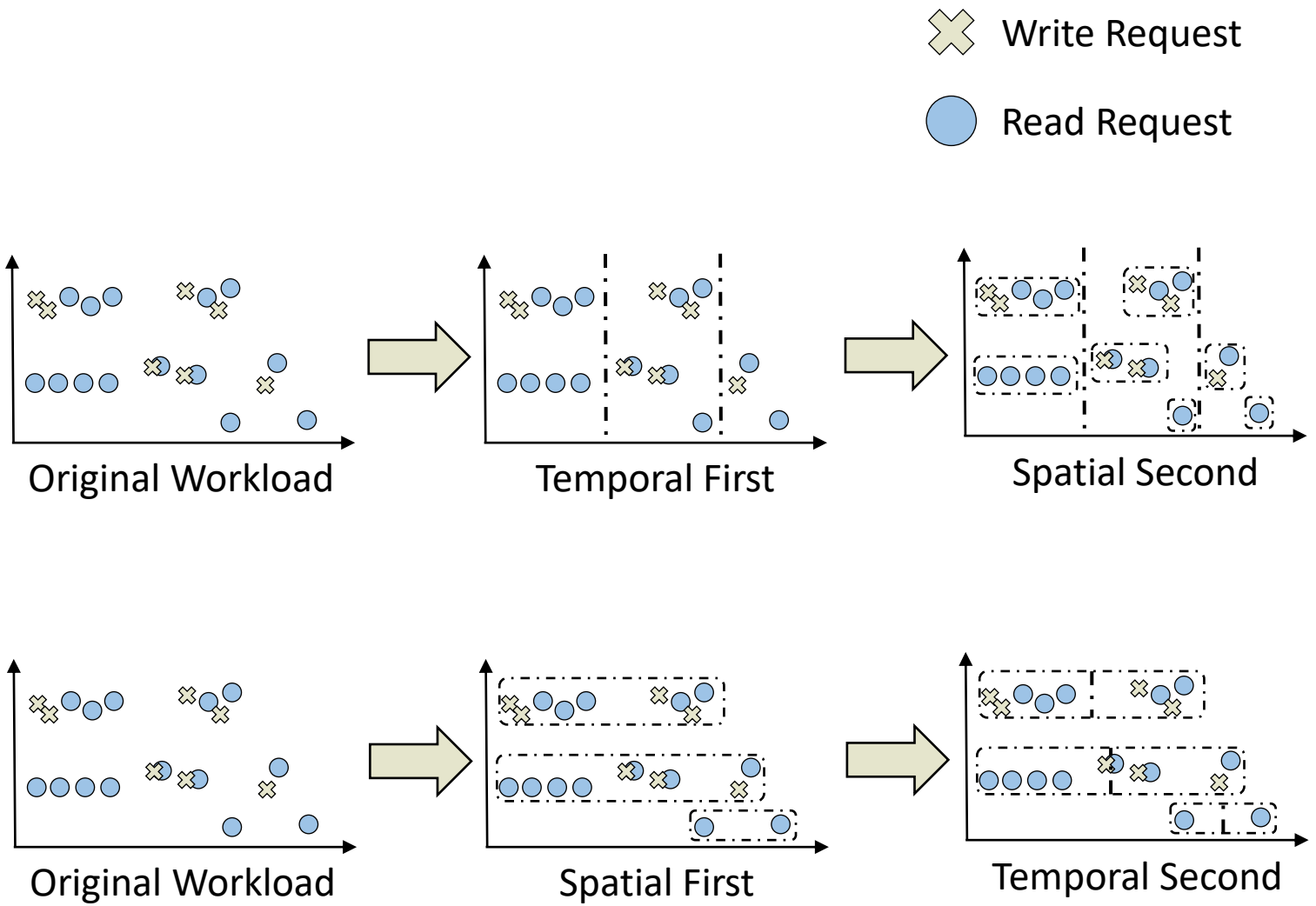
Spatial Partition 2

Time	Address	Stride
7	W	
8	X	-8
9	Y	4
11	Y	0

Spatial partitioning reduces the variance in the stride feature for both partitions.

Partitioning in Two Dimensions

- Temporal: reduces variability in delta times
- Spatial: reduces variability in strides



Carefully Dividing Requests

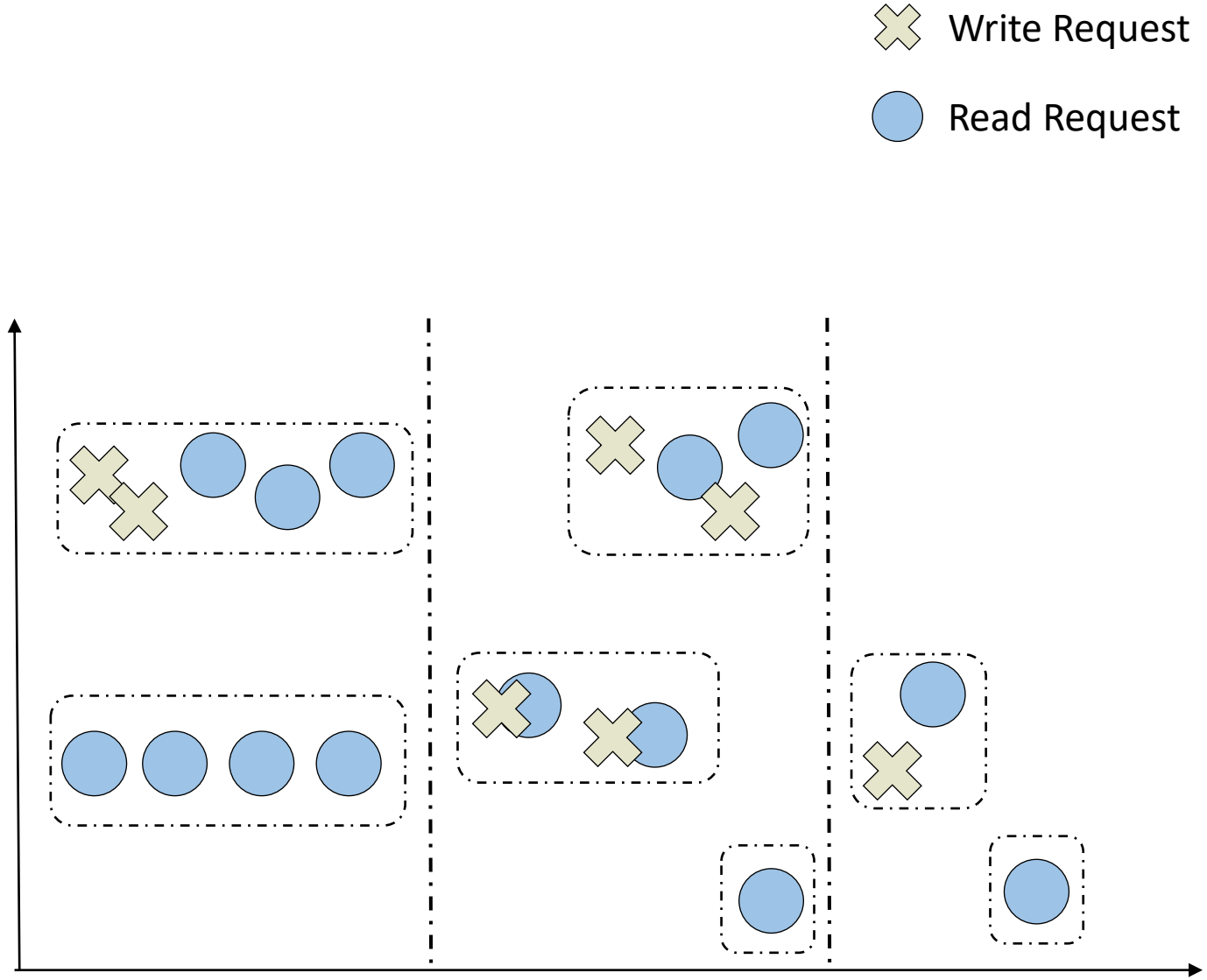
- Dynamic Spatial Partitioning
  - Find reque

Dynamic spatial partitioning adapts to the memory access behaviour of the workload and device.

- Spatial partitioning uncovers variable-sized time intervals
  - Phases with different start times and durations

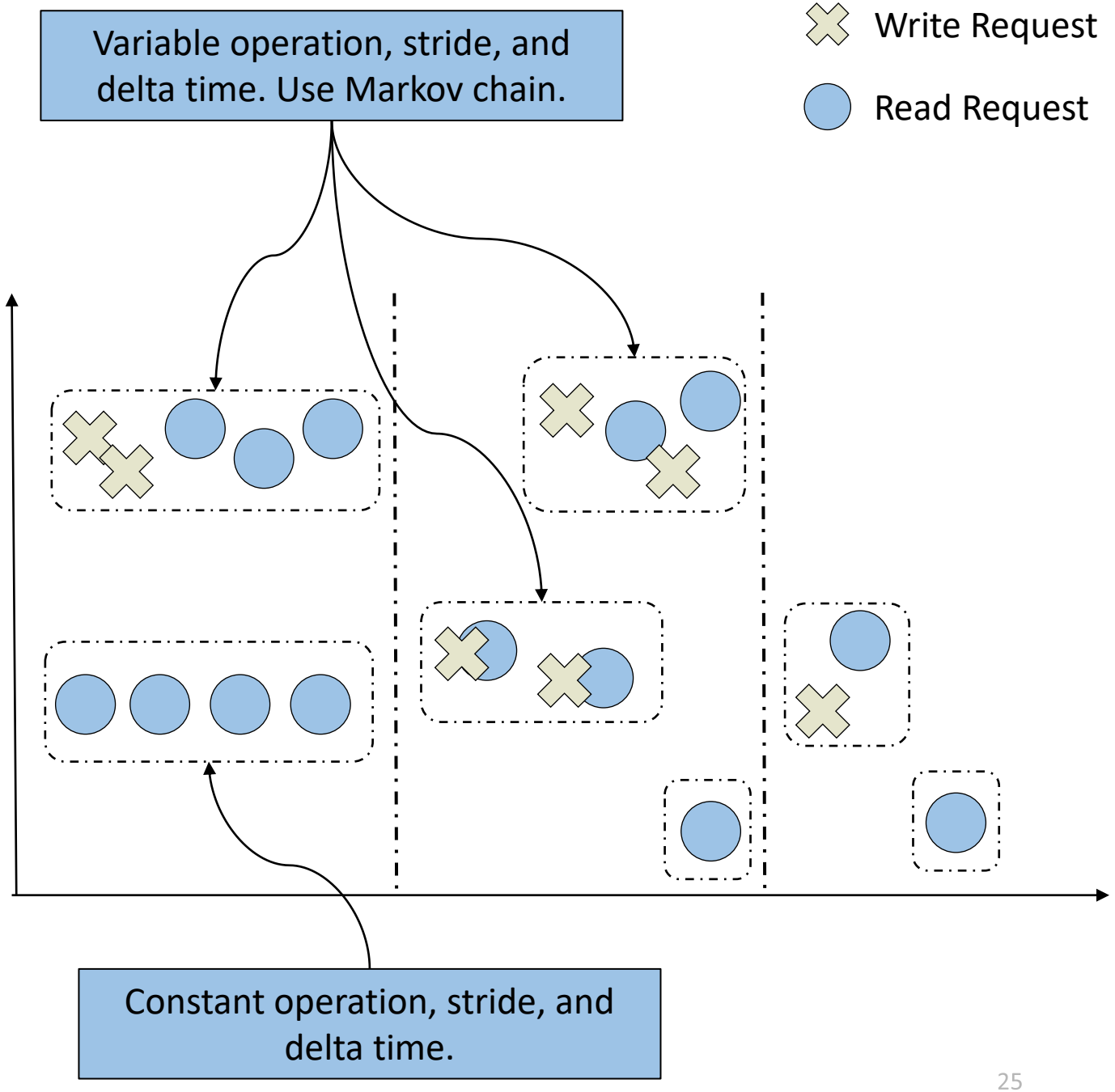
Modeling Each Partition

- Each partition consists of a sequence of memory requests
- Model each partition independently
- Save each partition's:
  - Start time
  - Start address



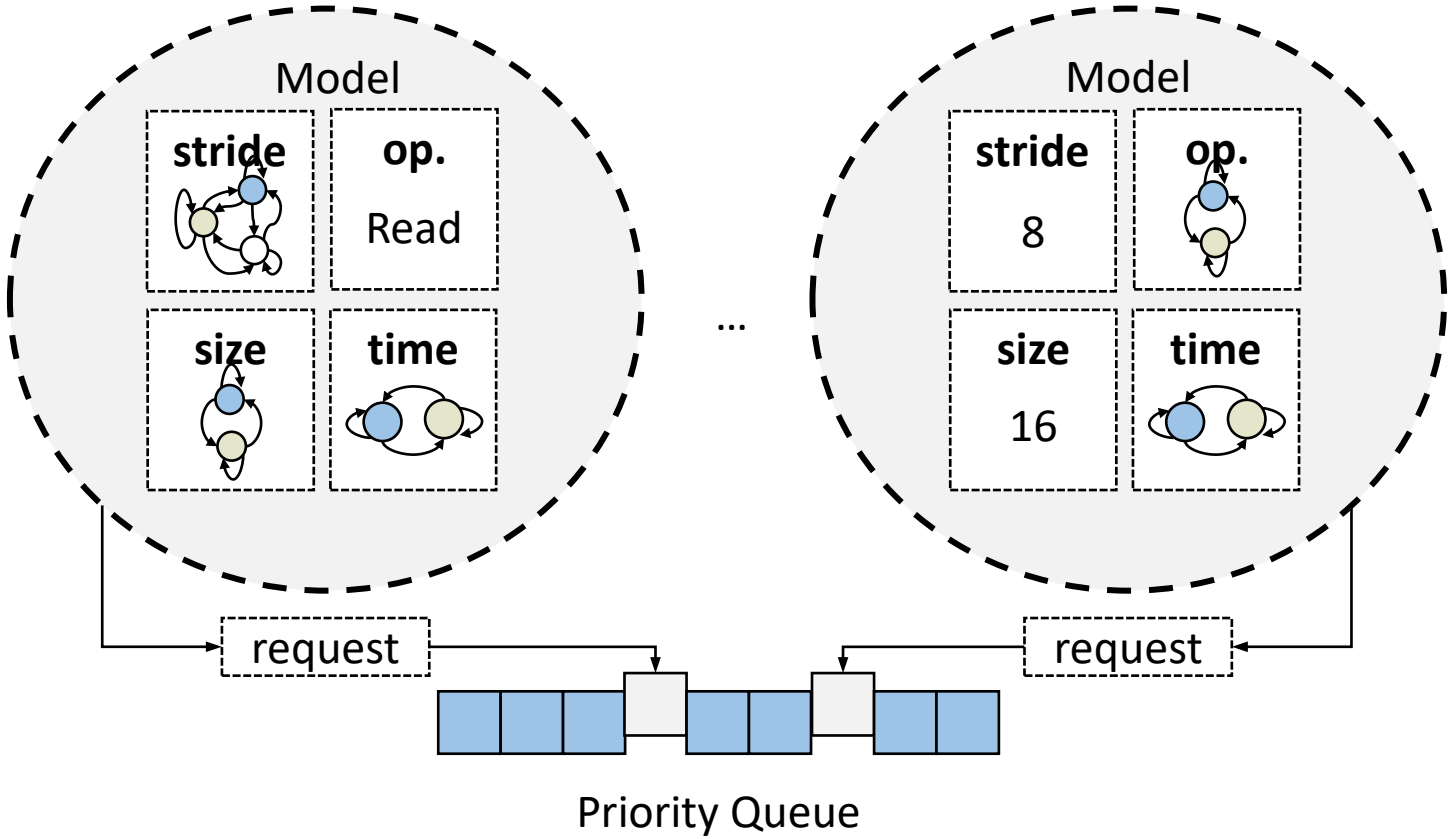
Modeling Each Feature

- Model each feature independently
- Features that do not change:
  - Constant value
- Features that do change:
  - Markov chain

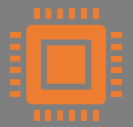


Synthesizing Requests

- Each model is used to generate requests
  - Need initial time and address
- Requests pushed into a priority queue.
  - Ordered by timestamp



Up Next: Evaluation



Background

- Heterogeneity
- Memory hierarchy
- Statistical simulation

Mocktails

- Modeling Requests
- Uncovering patterns
- Synthesizing

Evaluation

- Memory controller

Conclusion

- Summary of results
- Future directions

Methodology

- Proprietary memory access traces from Arm
  - CPU, DPU, GPU, VPU devices
- Trace-based simulation with gem5
  - Baseline: Arm traces
  - Requests sent to main memory over a crossbar

The Memory Controller

- Four channels
  - Each channel has a read and write queue
- Memory requests are dynamically scheduled
  - First-ready, first-come first-serve

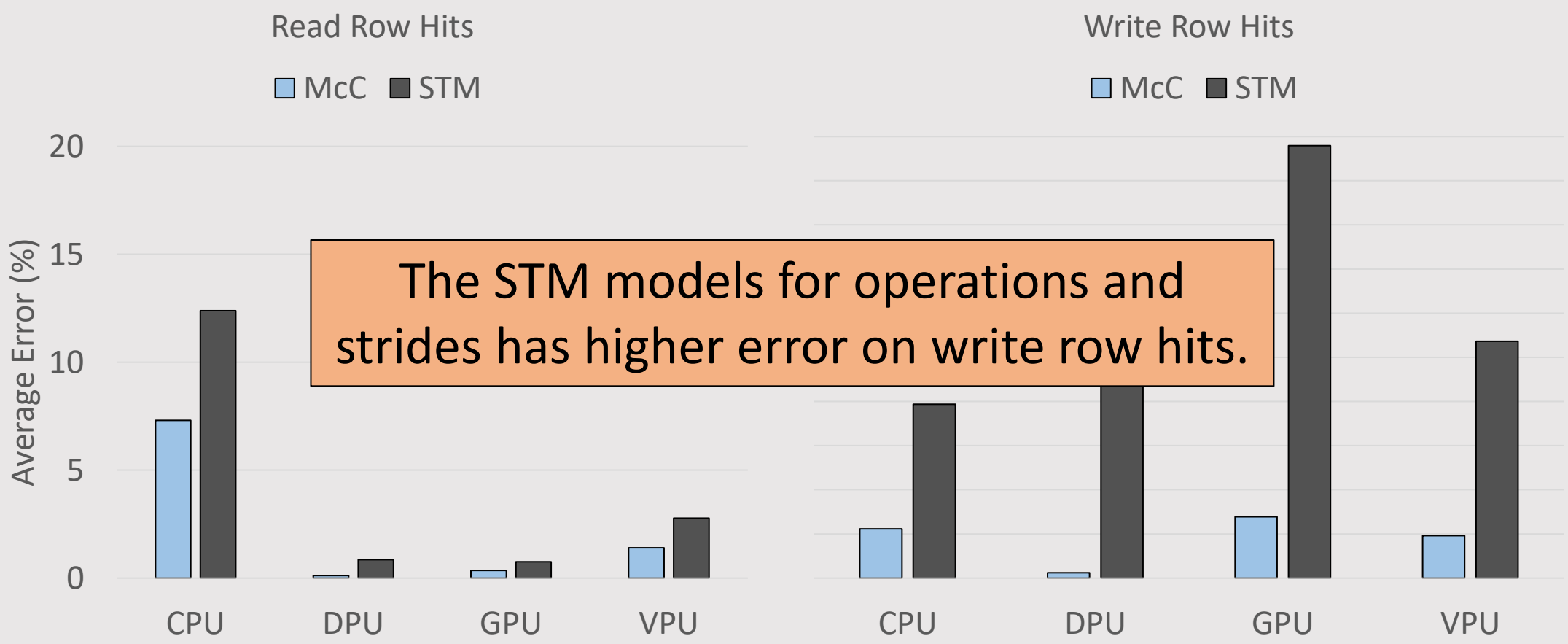
Model Comparison

- Perform hierarchical partitioning
  - Model each partition with two different approaches
  - Configuration: Temporal then Spatial (i.e., 2L-TS)
  - 500,000 cycle time intervals

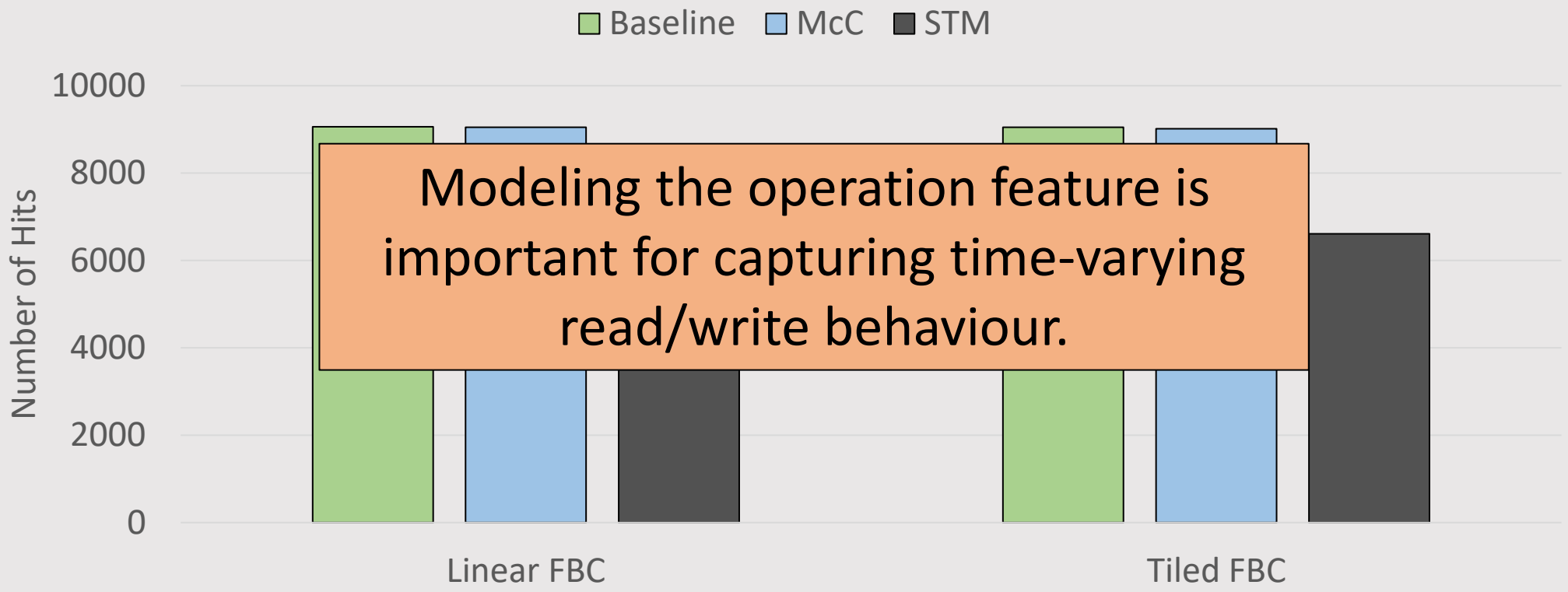
- Mocktails Approach
  - Markov chain or Constant value for each feature (i.e., the McC model)

- STM Approach
  - A statistical simulation technique for the CPU
  - Weighted coin flip for operation feature
  - Markov chain with history for stride feature
  - Other features use Mocktails approach

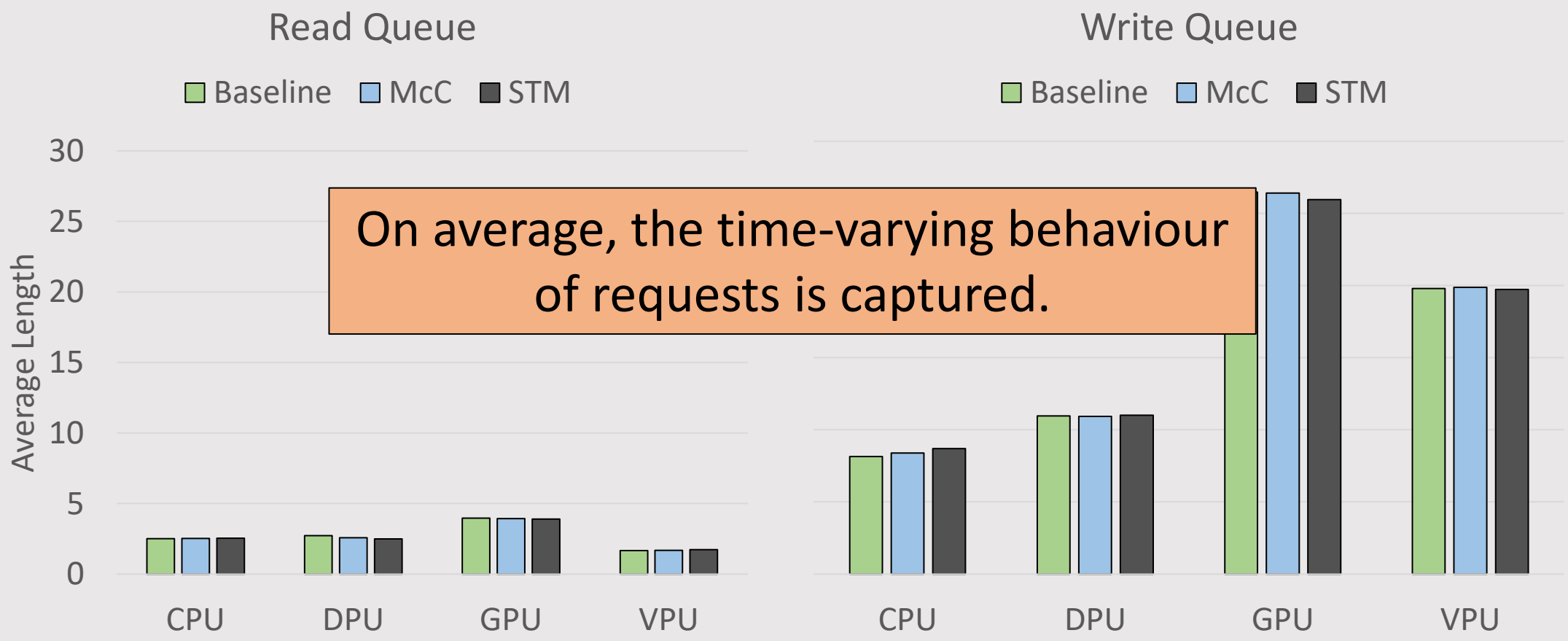
Absolute Accuracy of Row Hits



Write Row Hits (DPU)

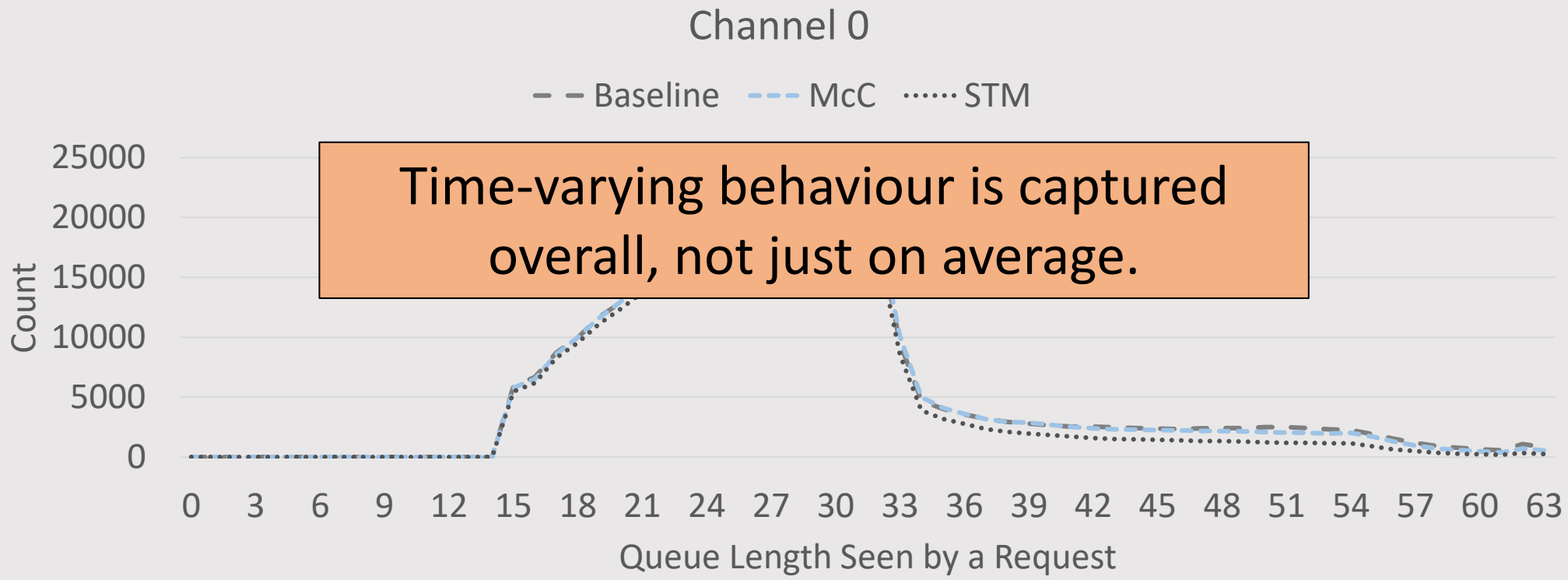


Average Queue Length

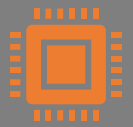


On average, the time-varying behaviour of requests is captured.

Write Queue Length (Manhattan GPU)



Up Next: Conclusion



Background

- Heterogeneity
- Memory hierarchy
- Statistical simulation

Mocktails

- Modeling Requests
- Uncovering patterns
- Synthesizing

Evaluation

- Memory controller

Conclusion

- Summary of results
- Future directions

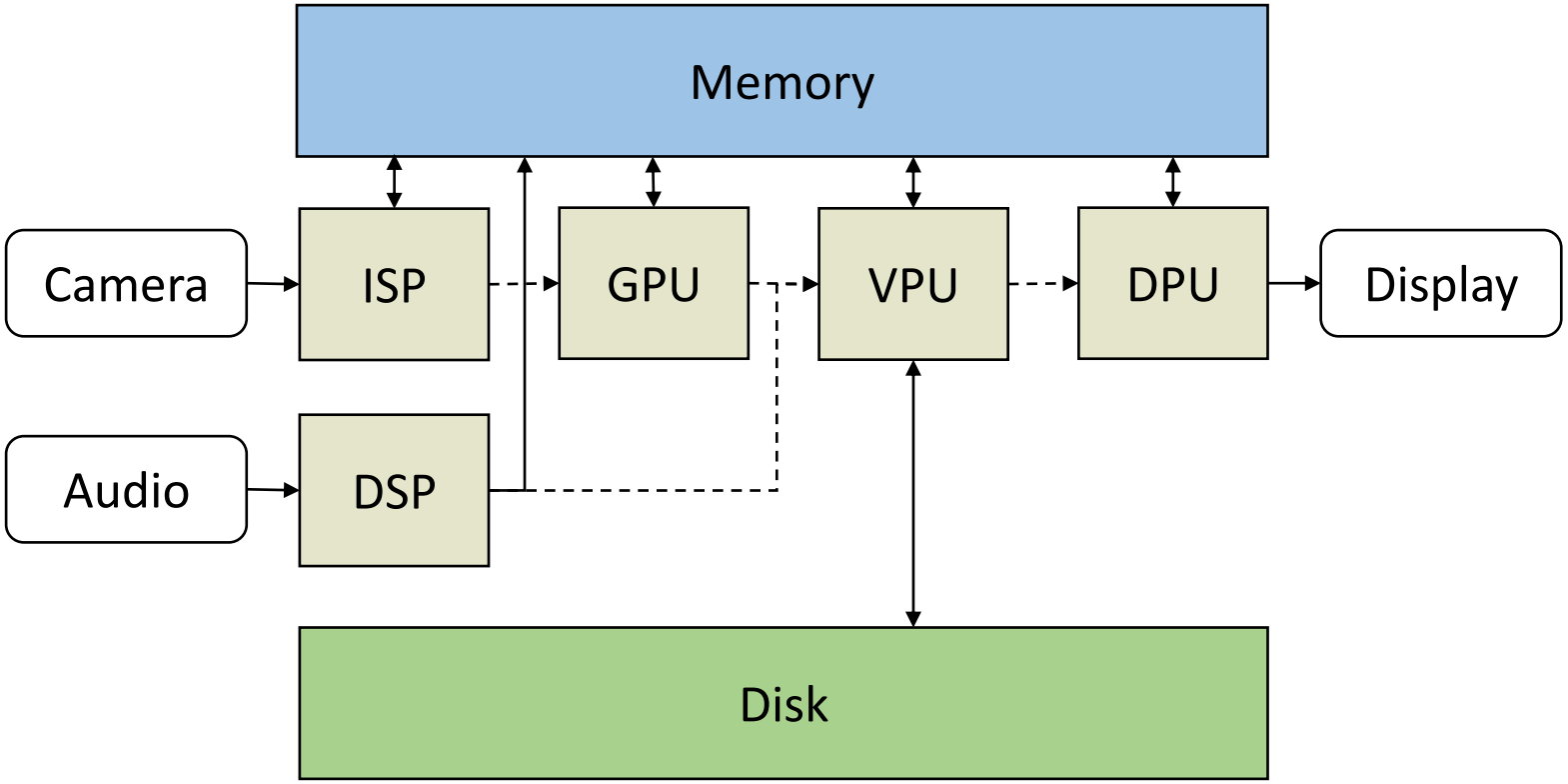
Summary of Results

- Mocktails is accurate for CPU, DPU, GPU, and VPU devices
  - Black-box models
  - Proprietary workloads: 1% error on page hit rates
  - SPEC CPU2006: 5.6% error on L1 cache miss rates

- Mocktails profiles are distributable
  - 84% smaller than trace files
  - Do not include proprietary details

Future Directions

- IP blocks are used concurrently.
- Combine Rhythm with Mocktails to simulate concurrent, heterogeneous workloads.



Questions & Answers

Thank you for your time