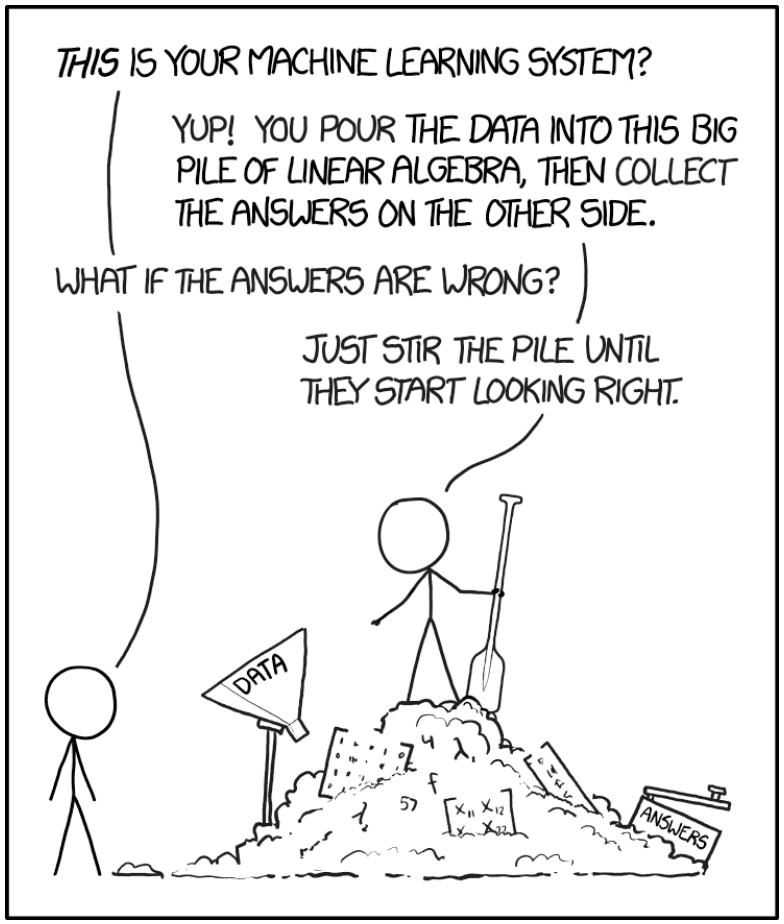
Machine Learning for Computer Systems

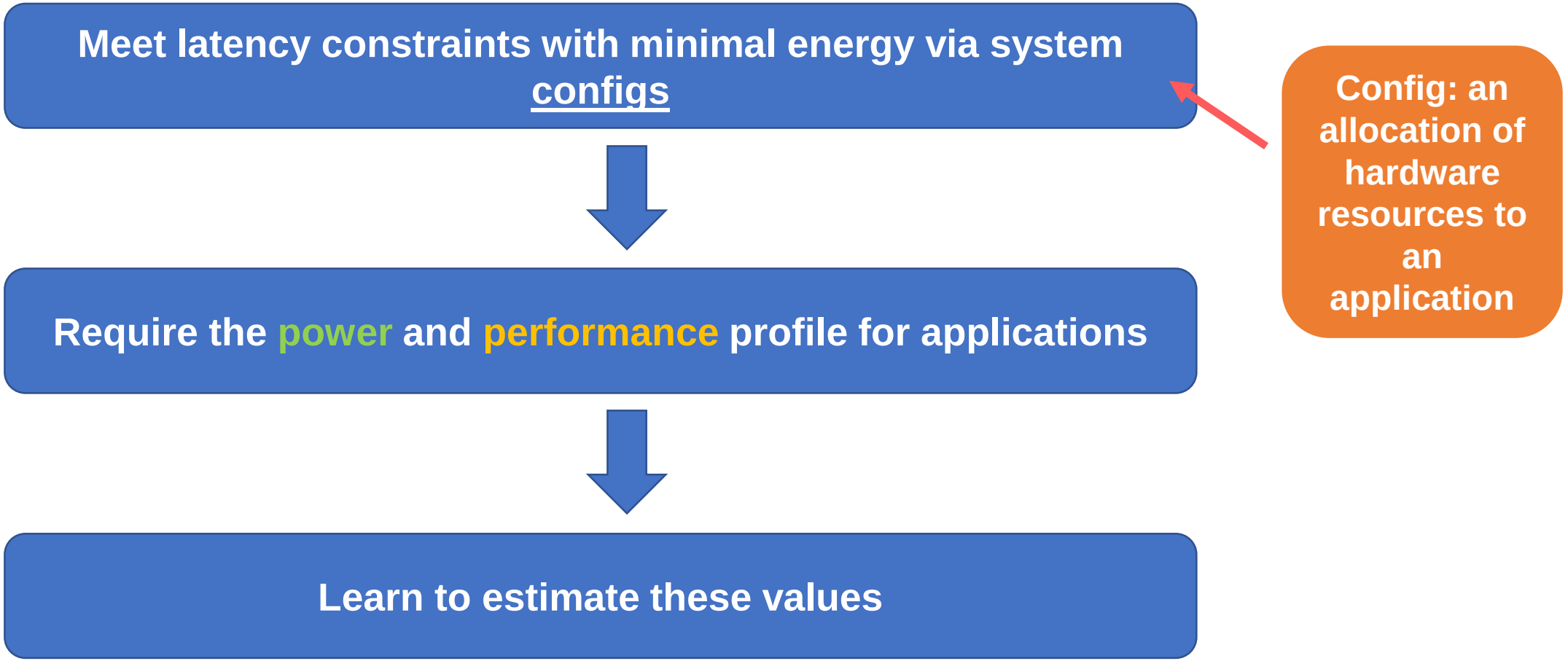
Henry Hoffmann

Machine Learning Overview



- What kind of answers do we want?
- What kind of data can we gather?
- What linear algebra do we use?
- How do you formulate the problems so you don’t have to “stir” too much?

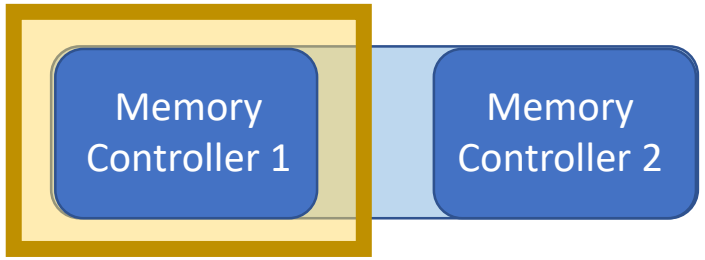
Example Problem Formulation



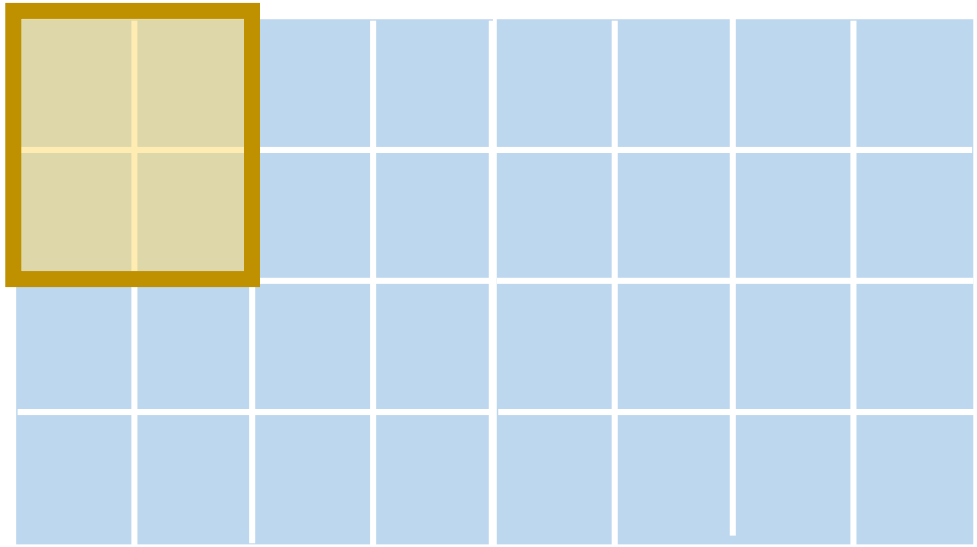
Example of a System Configuration Space $\mathcal{C}$

$$\mathcal{C} \leftarrow \{\text{Core assignment}\} \times \{\text{Clock speed assignment}\} \times \{\text{Memory controller}\}$$

Clock Speed

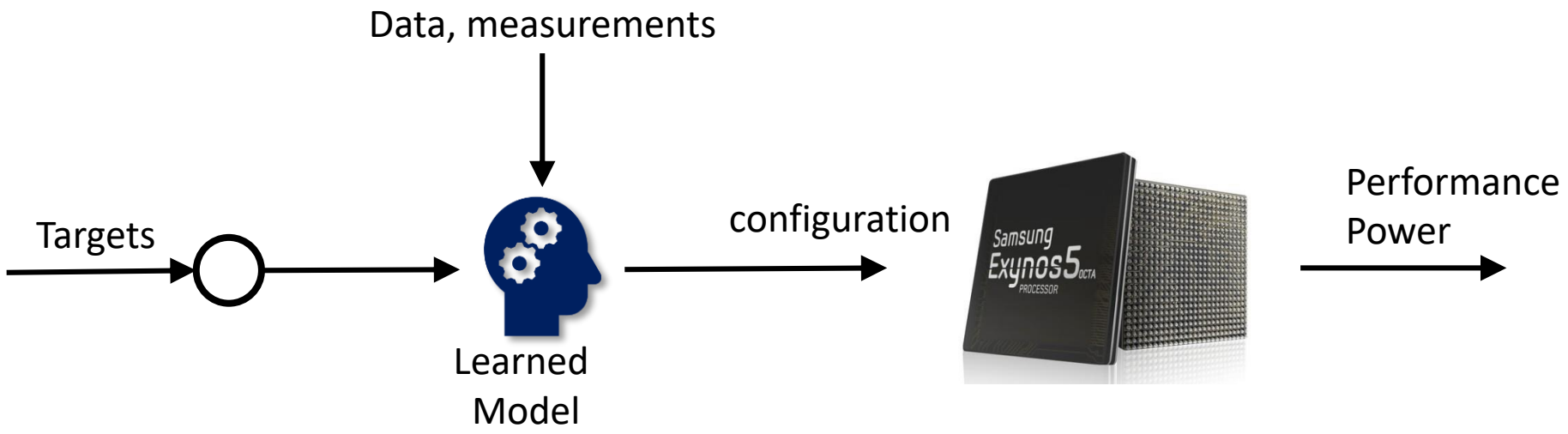


Memory controller



Cores

Machine Learning Overview II



- We want to drive the system to a certain power or performance
- We want to build a model that can map measured features to a configuration
- Configuration should achieve the target
- Note: ignoring reinforcement learning for now

What Data Can We Measure?

- High-level outcomes:
  - Metrics of importance to architecture users: throughput, latency, energy, etc
- Low-level outcomes:
  - Metrics of importance to architects: IPC, MPKI, Branch mispredicts, etc
- Parameters:
  - The things in the architecture that we can change to affect outcomes

What Answers Can We Obtain?

- Prediction:
  - Given some subset of measurements, what values will the remaining measurements take?

- Structure:
  - How do the measured values interact and affect each other?

- A note on correlation vs causation:
  - Correlations may be helpful for prediction accuracy
  - Causal relationships should be more robust but require structural learning

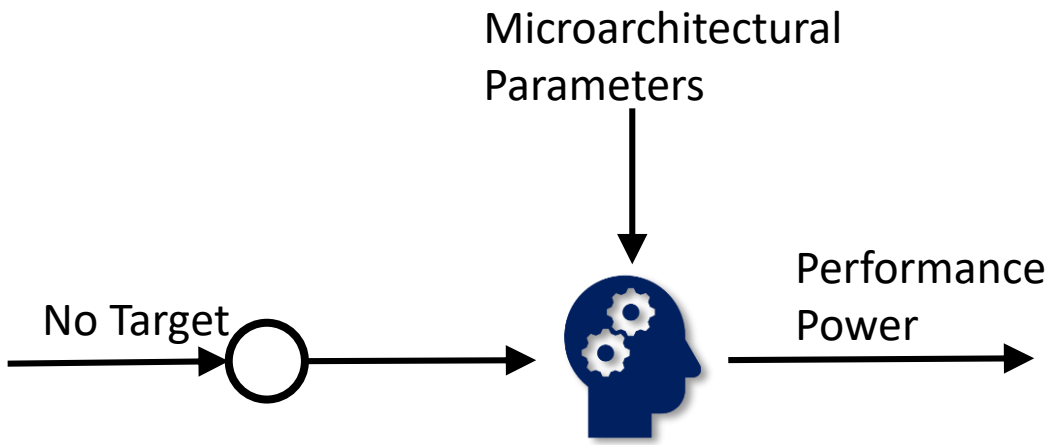
What Linear Algebra Can We Use?

- Too many options!

- Some examples to come:
  - Regression
  - Regularized Regression
  - Recommender Systems/Matrix Completion
  - Reinforcement Learners: Neural Networks and others
  - Evolutionary Learners

Regression Modeling
Lee and Brooks ASPLOS 2006

Problem: Reduce number of simulations required during microarchitectural design time by predicting performance



Performance:
Low-level outcome

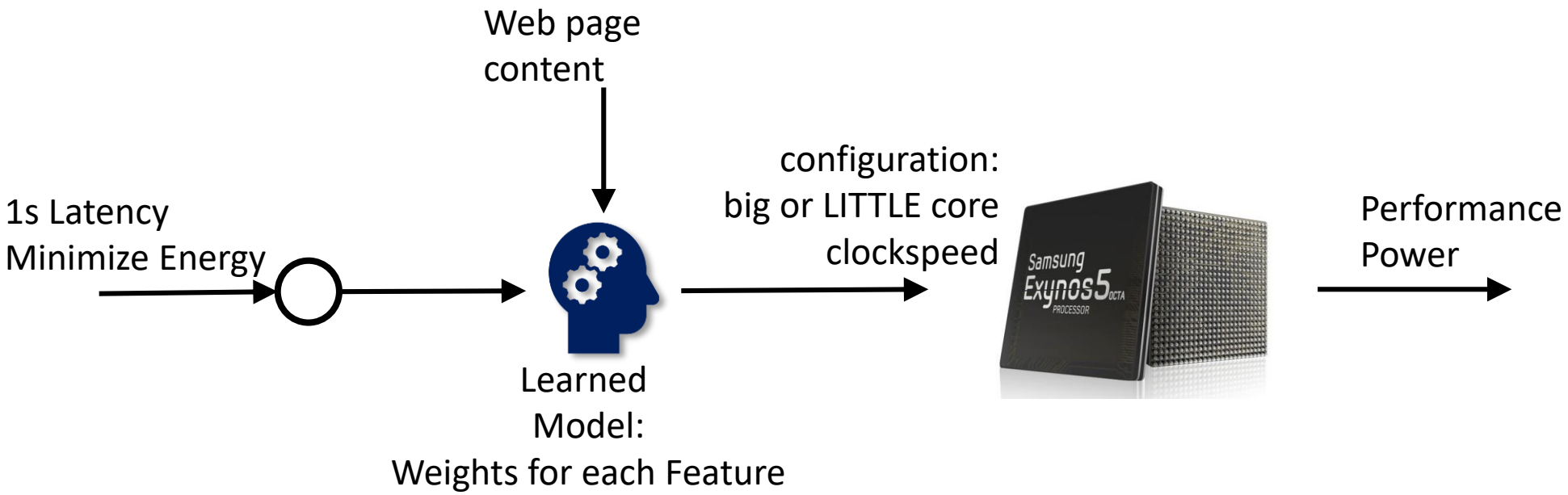
Features:
Architectural Parameters

Model:
Weights on the Parameters

$$\begin{bmatrix} \text{IPC} \end{bmatrix} = \begin{bmatrix} \text{L1 Size} & \text{LLC Size} & \text{Issue Width} & \text{Register File Size} \end{bmatrix} \times \begin{bmatrix} w0 \\ w1 \\ w2 \\ w3 \end{bmatrix}$$

Regression Modeling with too Many Features
Zhu and Reddi HPCA 2013

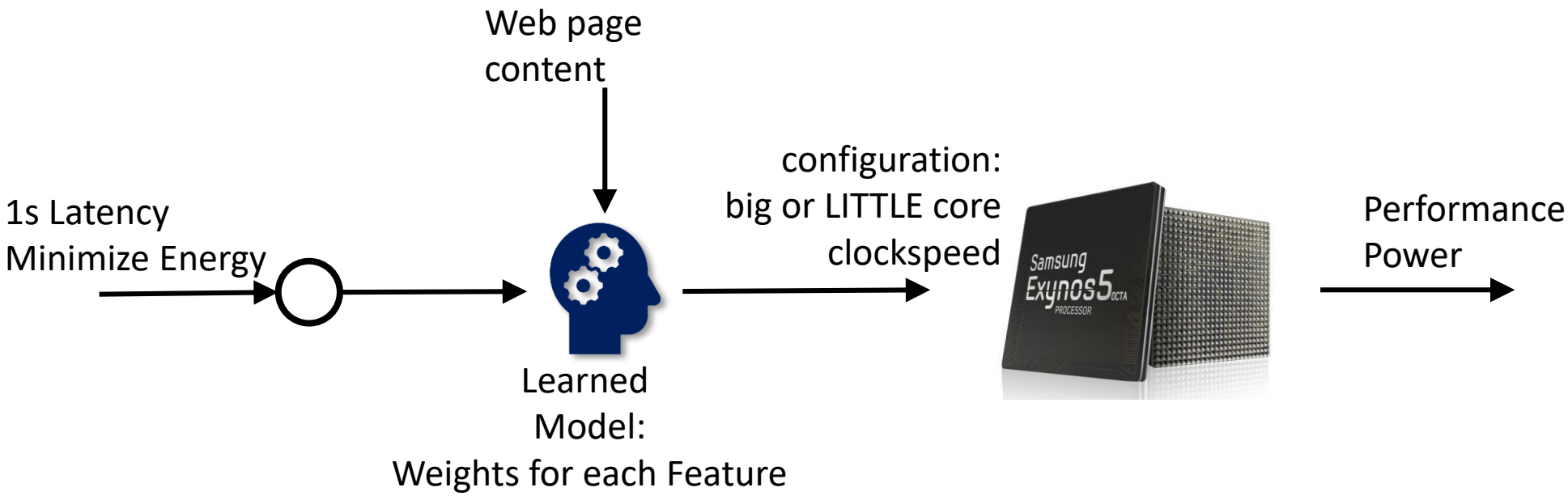
Problem: Energy efficient rendering of webpages on mobile devices



$$\begin{bmatrix}\text{Latency}\end{bmatrix} = \begin{bmatrix}\text{Tag} & \text{DOM nodes} & \text{Image Sizes} & \text{Image}\end{bmatrix} \times \begin{bmatrix} w0 \\ w1 \\ w2 \\ w3 \end{bmatrix}$$

Regression Modeling with too Many Features
Zhu and Reddi HPCA 2013

Problem: Energy efficient rendering of webpages on mobile devices



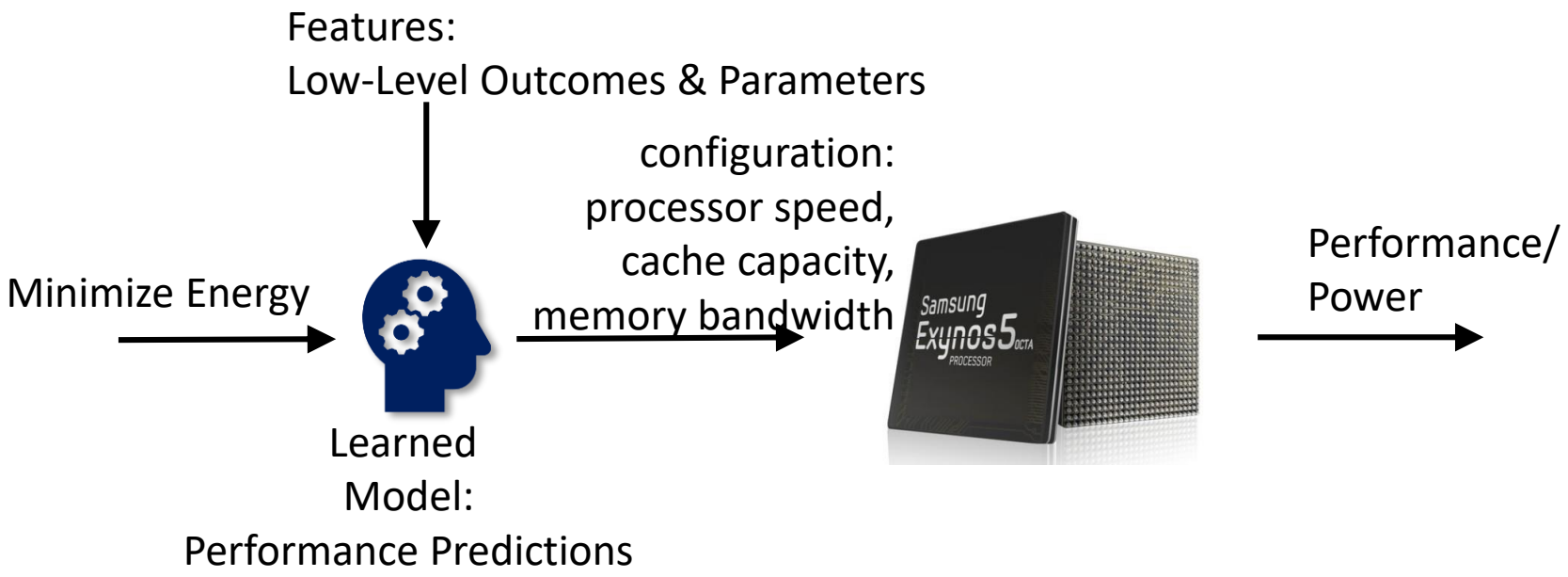
$$\sum_{k=0}^{n} w_i^2 \leq threshold$$

Regularized regression adds constraints and turns this into an optimization problem

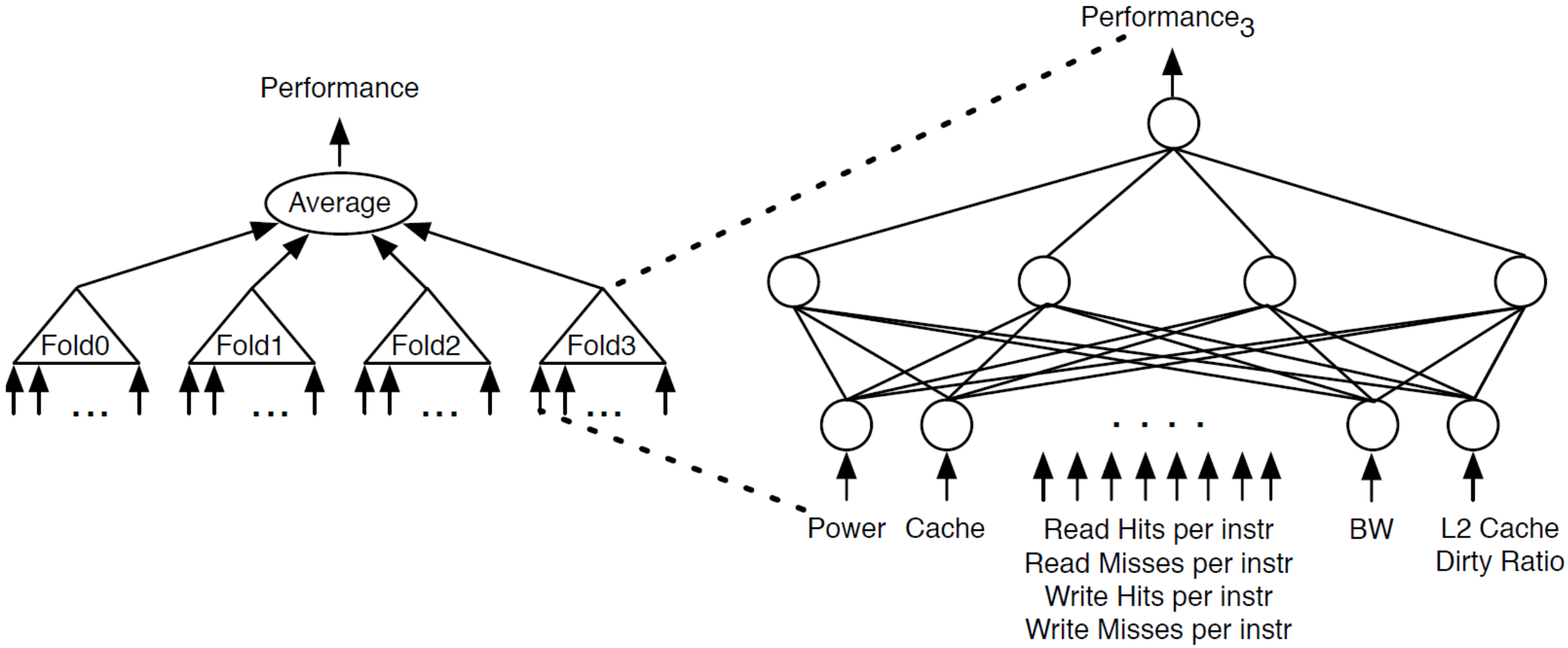
$$\begin{bmatrix} \text{Latency} \end{bmatrix} = \begin{bmatrix} \text{Tag} & \text{DOM nodes} & \text{Image Sizes} & \text{Image} \end{bmatrix} \times \begin{bmatrix} w0 \\ w1 \\ w2 \\ w3 \end{bmatrix}$$

Reinforcement Learning
Bitirgen et al. MICRO 2010

Problem: Find the most energy efficient configuration of a processor for multiapp workloads

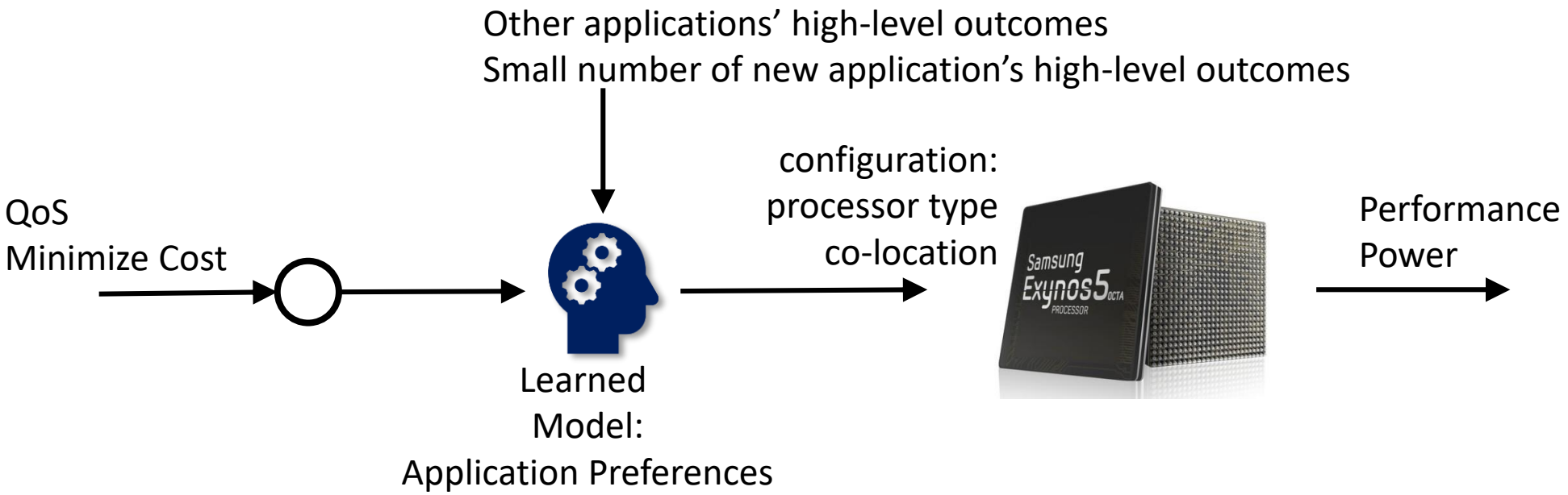


Reinforcement Learning
Bitirgen et al. MICRO 2010

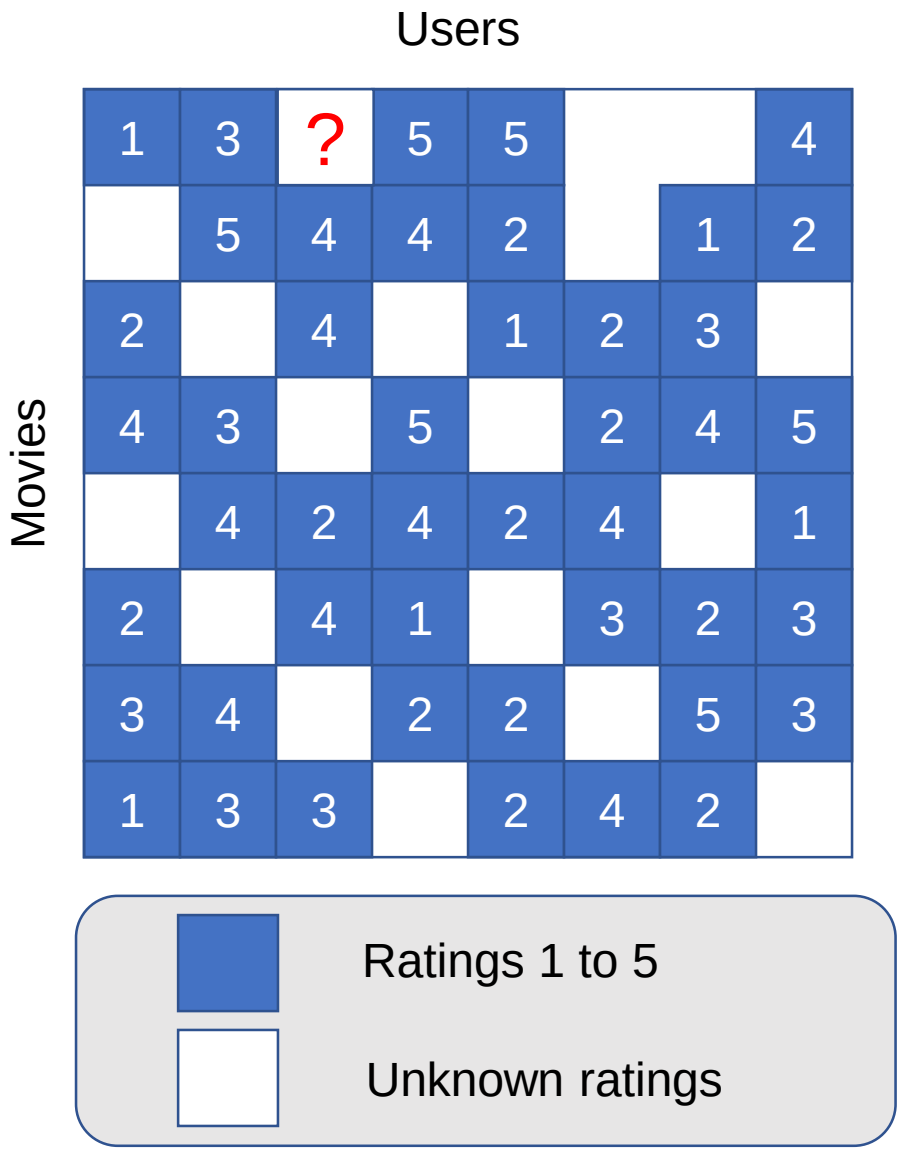


Recommender Systems
Delimitrou and Kozyrakis ASPLOS 2013,2014

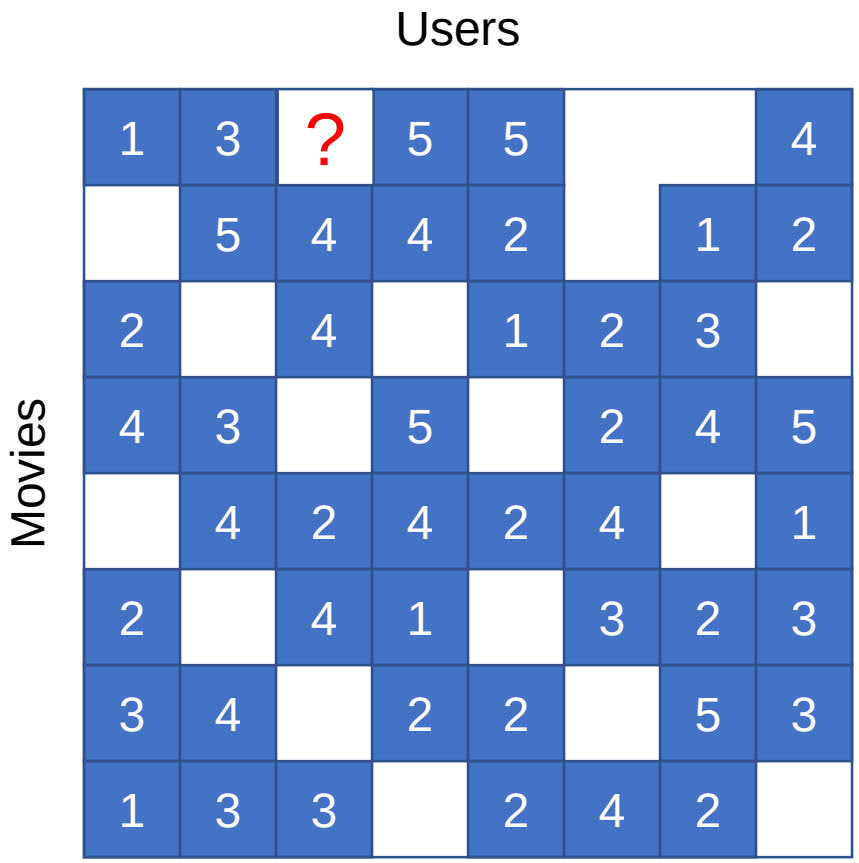
Problem: Assignment of jobs to machines in a heterogeneous datacenter

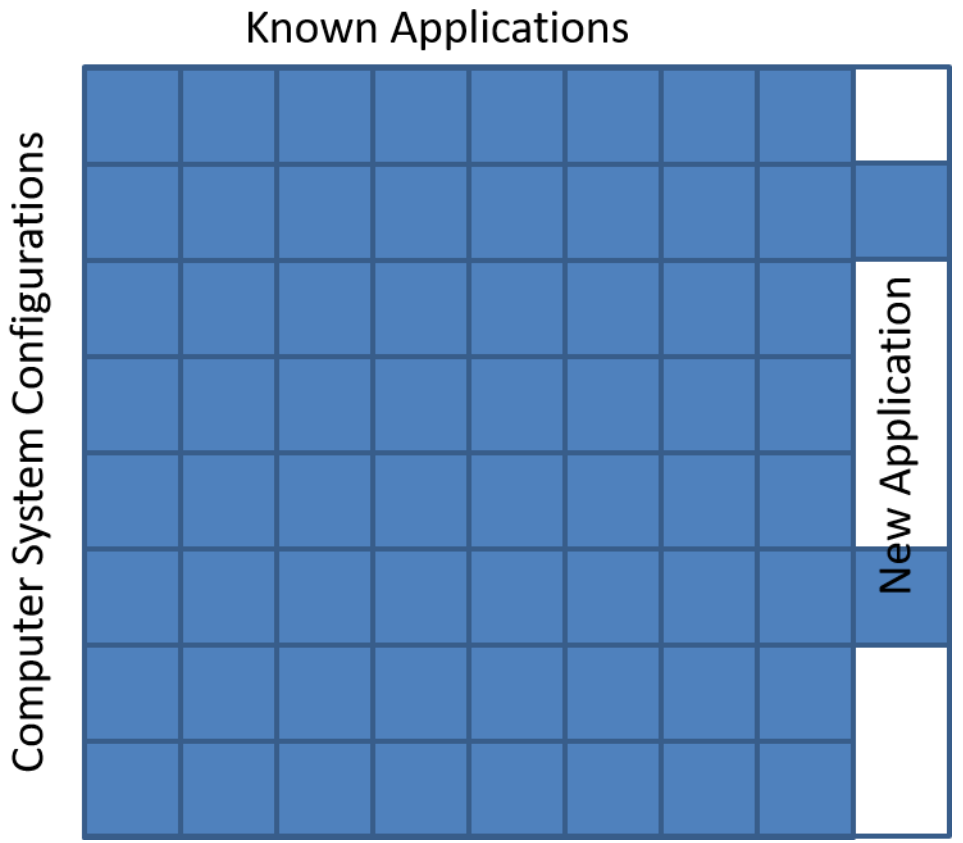


Recommender Systems



Recommender Systems





Ratings 1 to 5

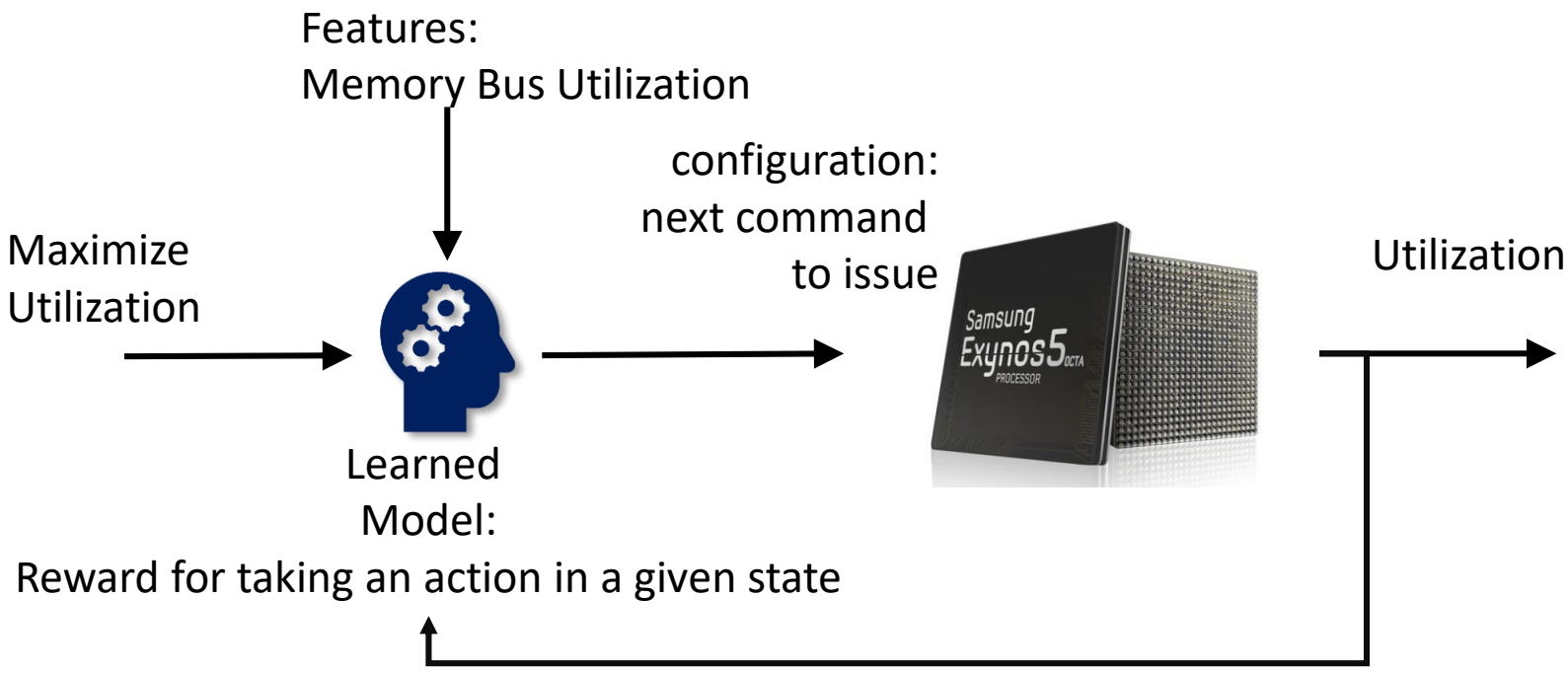
Unknown ratings

Performance/Power: high-level outcomes

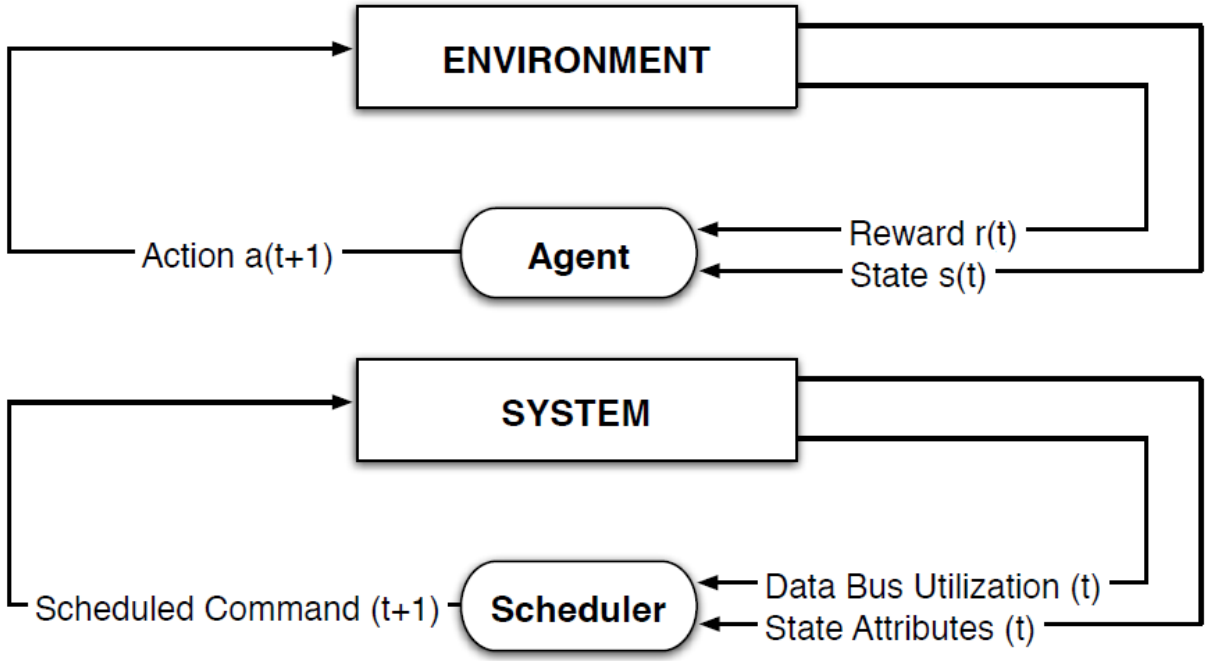
Unknown value

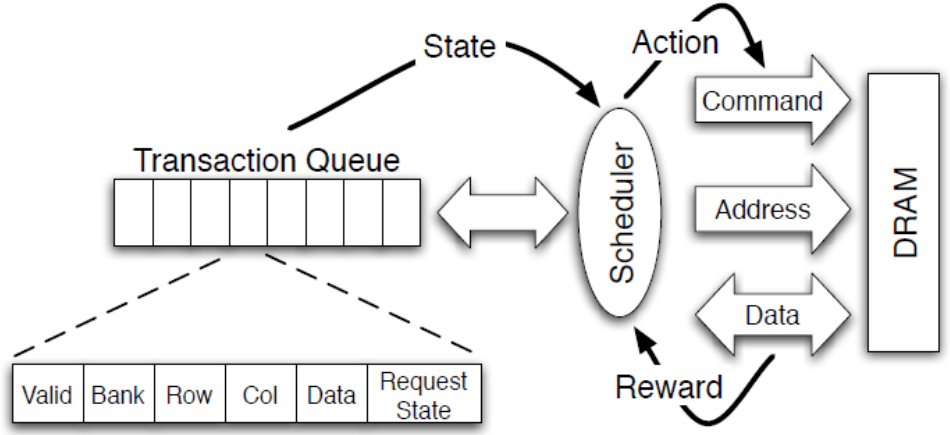
Reinforcement Learning
Ipek et al. ISCA 2008

Problem: Schedule DRAM commands to maximize utilization



Reinforcement Learning
Ipek et al. ISCA 2008





Summary of Examples

Approach	Inputs	Outputs	Key Technique	References
Regression	Microarchitectural Parameters	Performance Power	Cubic Regression with Splines	Lee and Brooks. ASPLOS 2006
Regularized Regression	Webpage Features	Performance Power	Lasso ElasticNet Cubic Regression	Zhao and Reddi. HPCA 2013
Neural Network Regression	Performance, Miss Rates, Resource Allocation	Aggregate Performance for Application Mix	Neural Network	Bitirgen, Ipek, and Martinez. MICRO 2010.
Recommender Systems	High-level Outcomes	Performance	Nuclear Norm Matrix Completion	Delimitrou and Kozyrakis. ASPLOS 2013 & 2014.
Reinforcement Learning	Memory-bus Utilization	Memory-bus Utilization	SARSA model	Ipek et al. ISCA 2008

Also: Penney and Chen.“A Survey of Machine Learning Applied to Computer Architecture Design.” arXiv 2019

A Note on Overhead

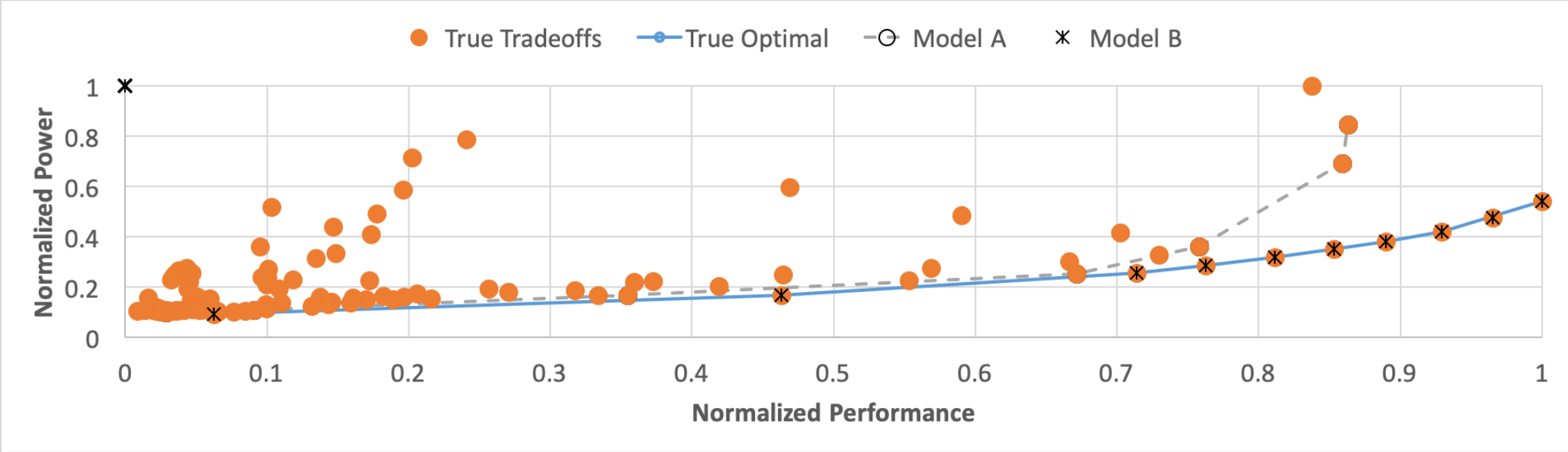
- Overhead is not monolithic
- Two components of overhead:
  - Number of samples required
  - Computation per sample
- These generally work against each other
  - Fewer samples → more work to extract meaning from samples
  - Less computation per sample → more samples needed to learn

Summary of Examples

- Diversity
  - Inputs could be any of High-, Low-level features, Parameters
  - Same for outputs
  - All use different linear algebra
  - For each example, we could find another paper that solves the same problem with a different underlying technique

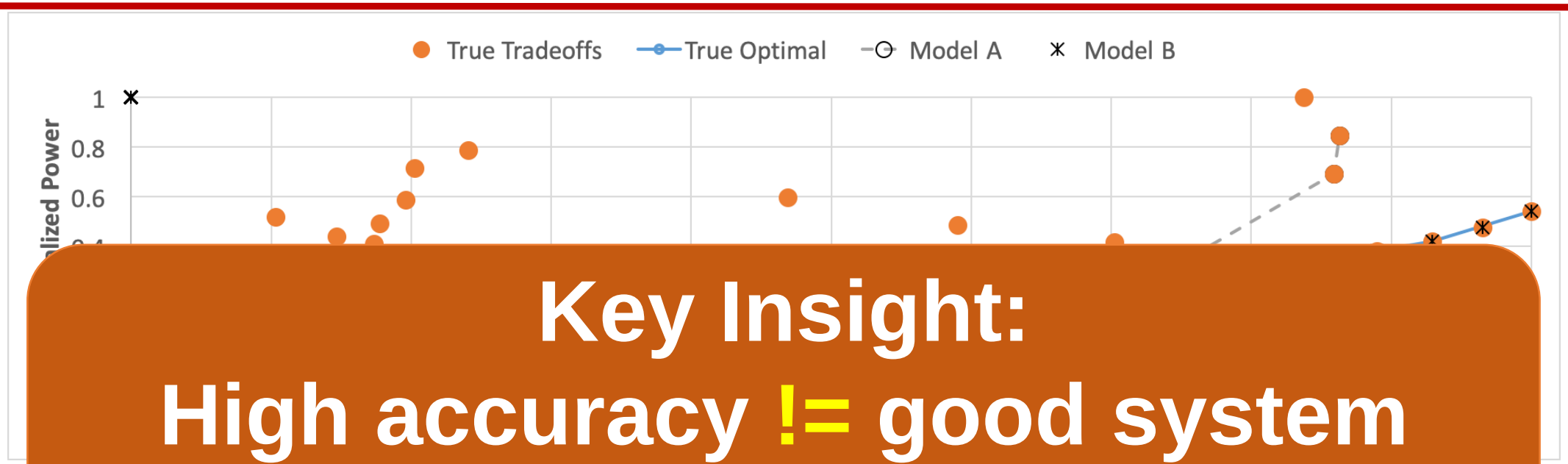
- Commonality
  - Predictions are only part of solving systems problems
  - Predictor is a module that is used by the rest of the system to make decisions
  - Predictor is not aware of the underlying problem structure

Predicting vs. Structural Learning
Ding et al. ISCA 2019



	Model A	Model B
Built for	Prediction	Structure
Optimal points	Just far enough	True data
Non-optimal points	True data	Very far
Goodness of fit	99%	0
Energy over optimal	22% ✗	0 ✓

Predicting vs. Structural Learning
Ding et al. ISCA 2019



Key Insight:
High accuracy != good system results

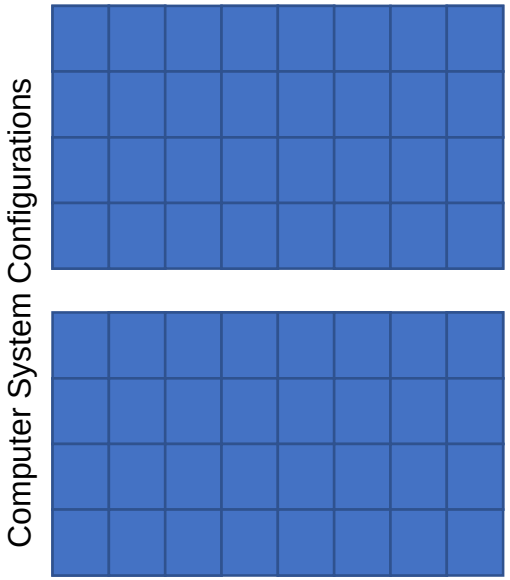
Built for	Prediction	Structure
Optimal points	Just far enough	True data
Non-optimal points	True data	Very far
Goodness of fit	99%	0
Energy over optimal	22% ✗	0 ✓

Case Study

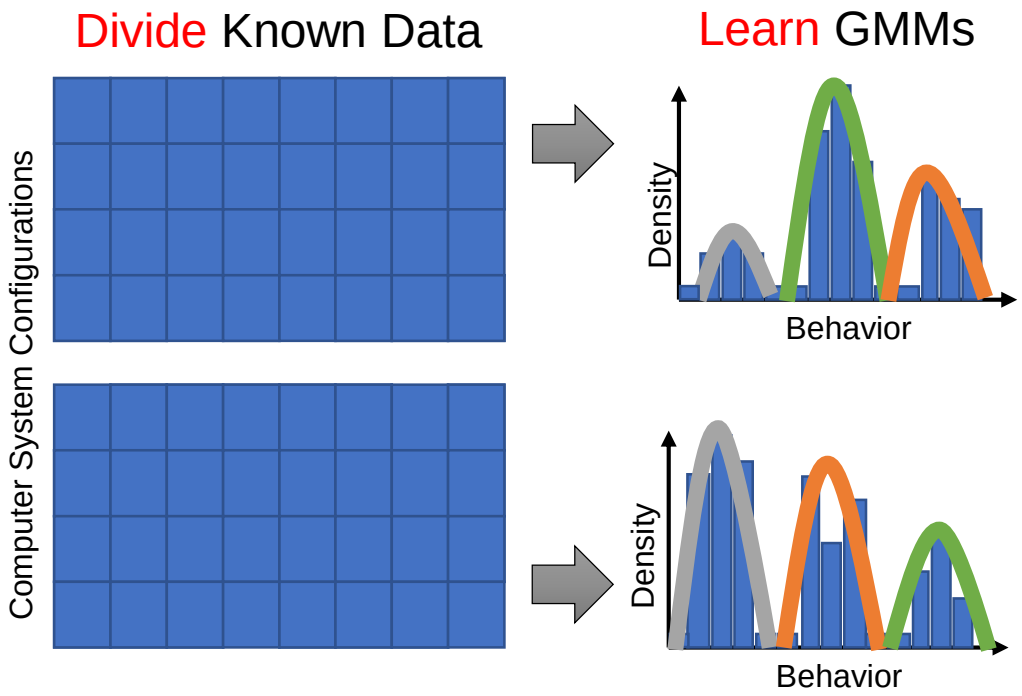
- Problem:
  - Meet latency with minimal energy through resource management
- Learning:
  - Use published methods to estimate latency and power
  - Mostly recommender based, but also Bayesian methods
- Structure:
  - Constrained optimization problem
- Known Issues
  - Improve accuracy with small data sets

Generating Data with a GMM

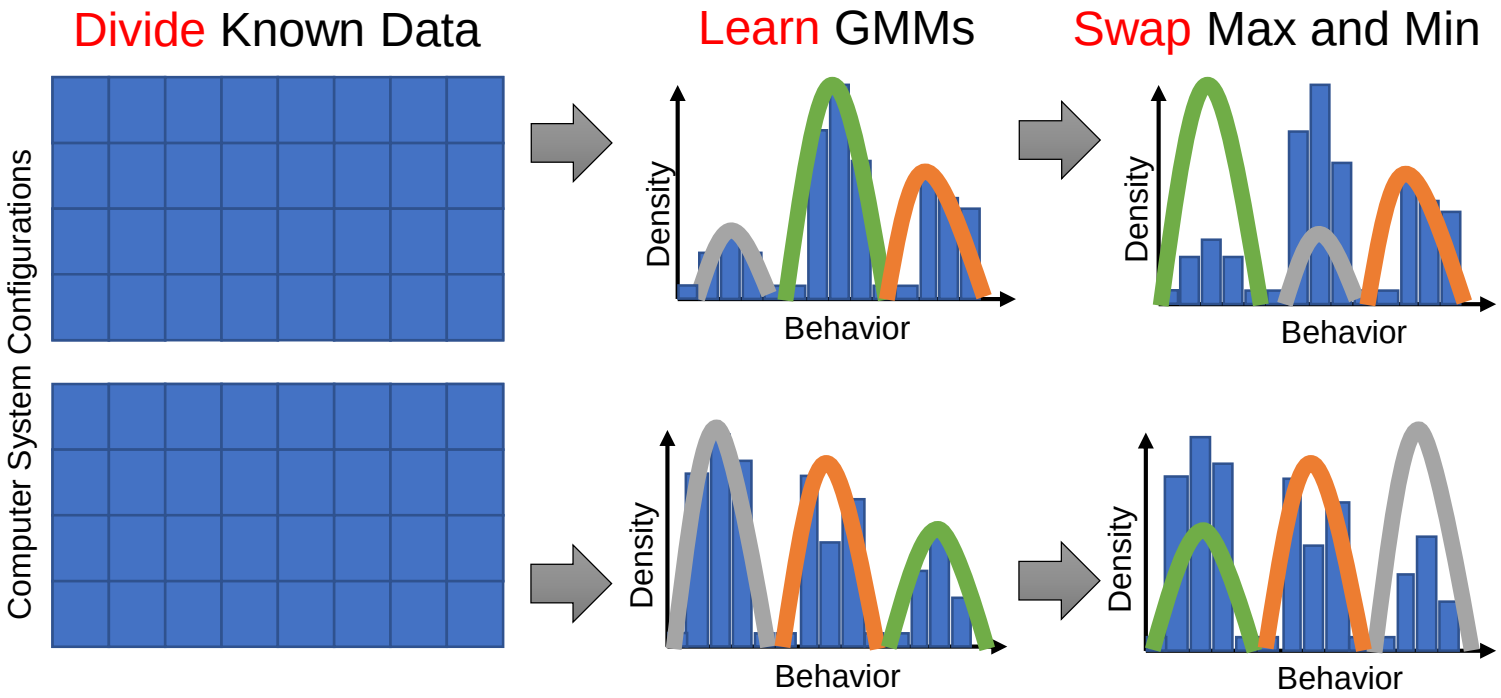
Divide Known Data



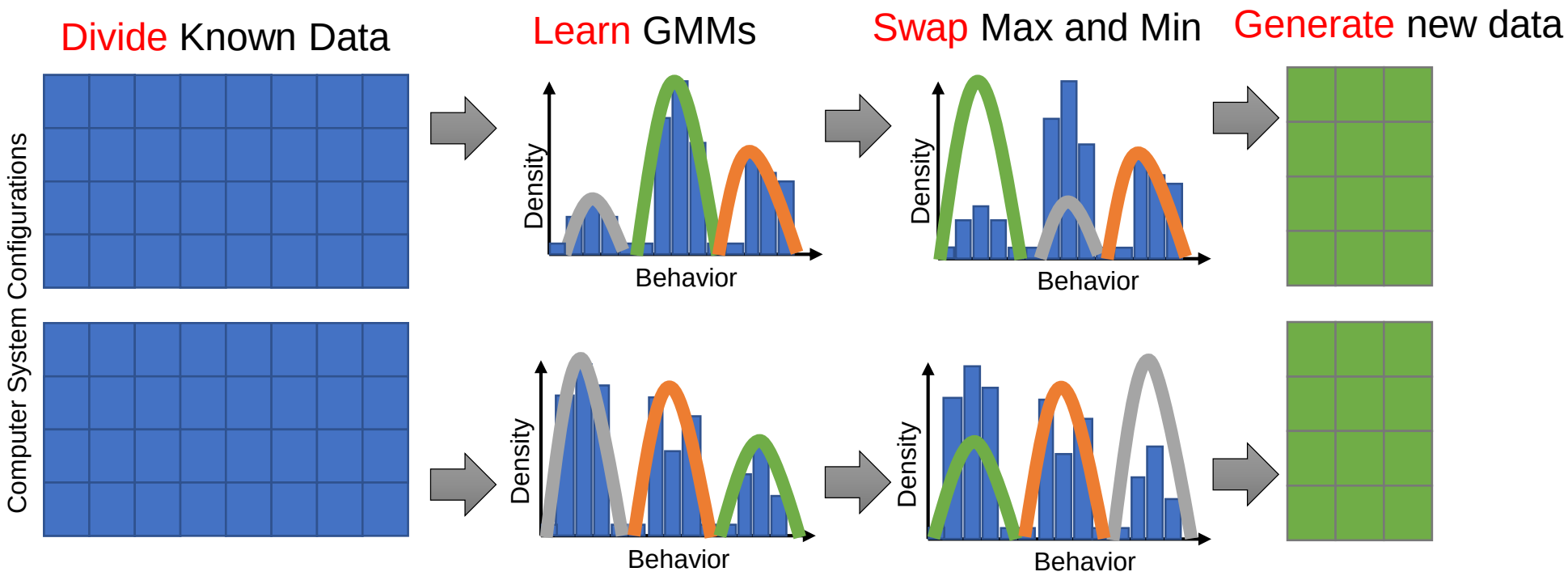
Generating Data with a GMM



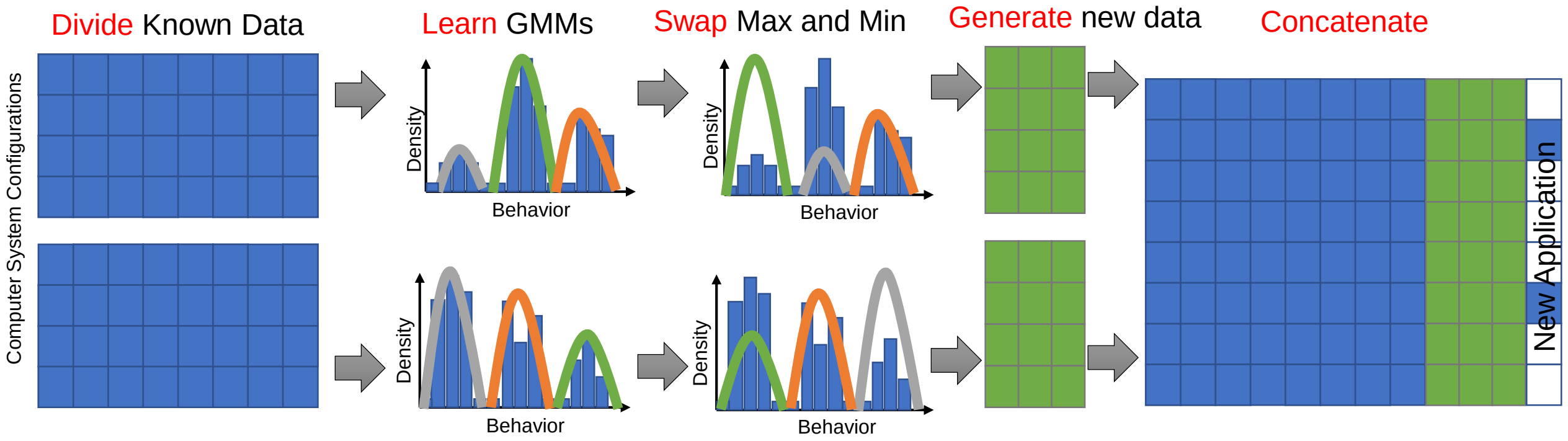
Generating Data with a GMM



Generating Data with a GMM



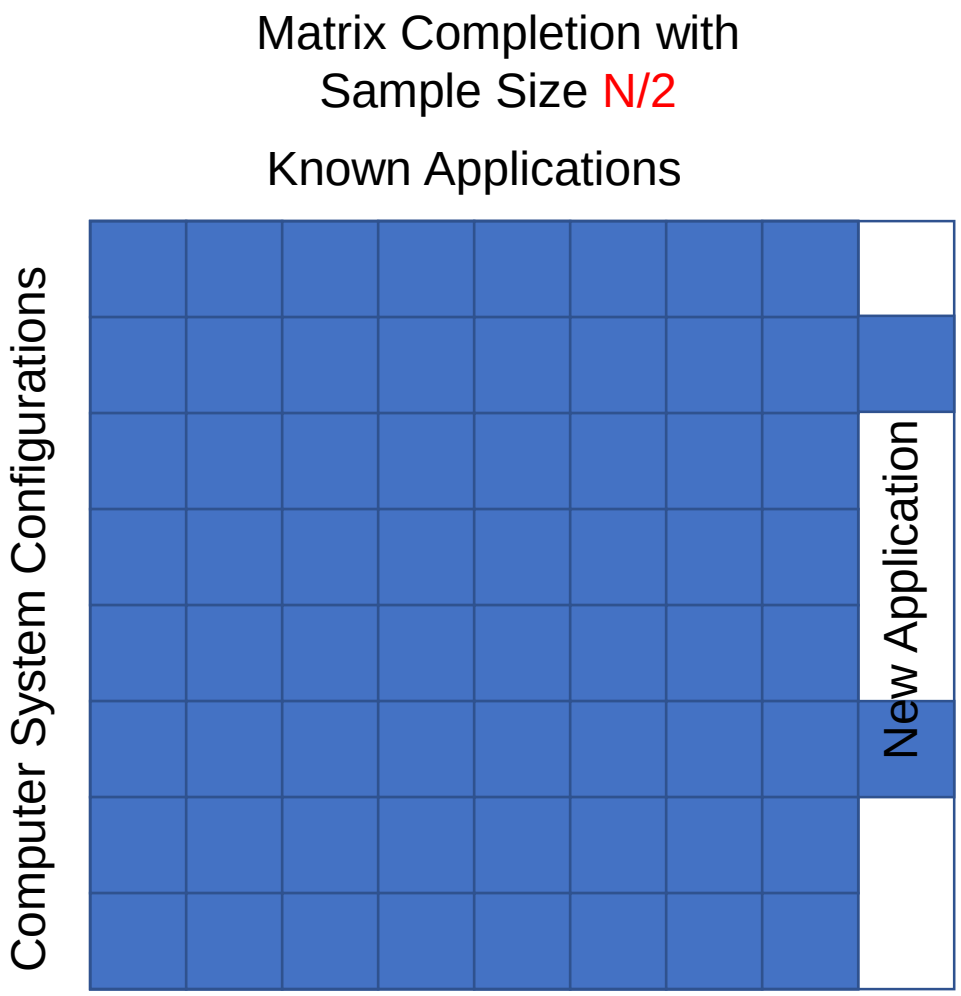
Generating Data with a GMM



Multi-phase Sampling

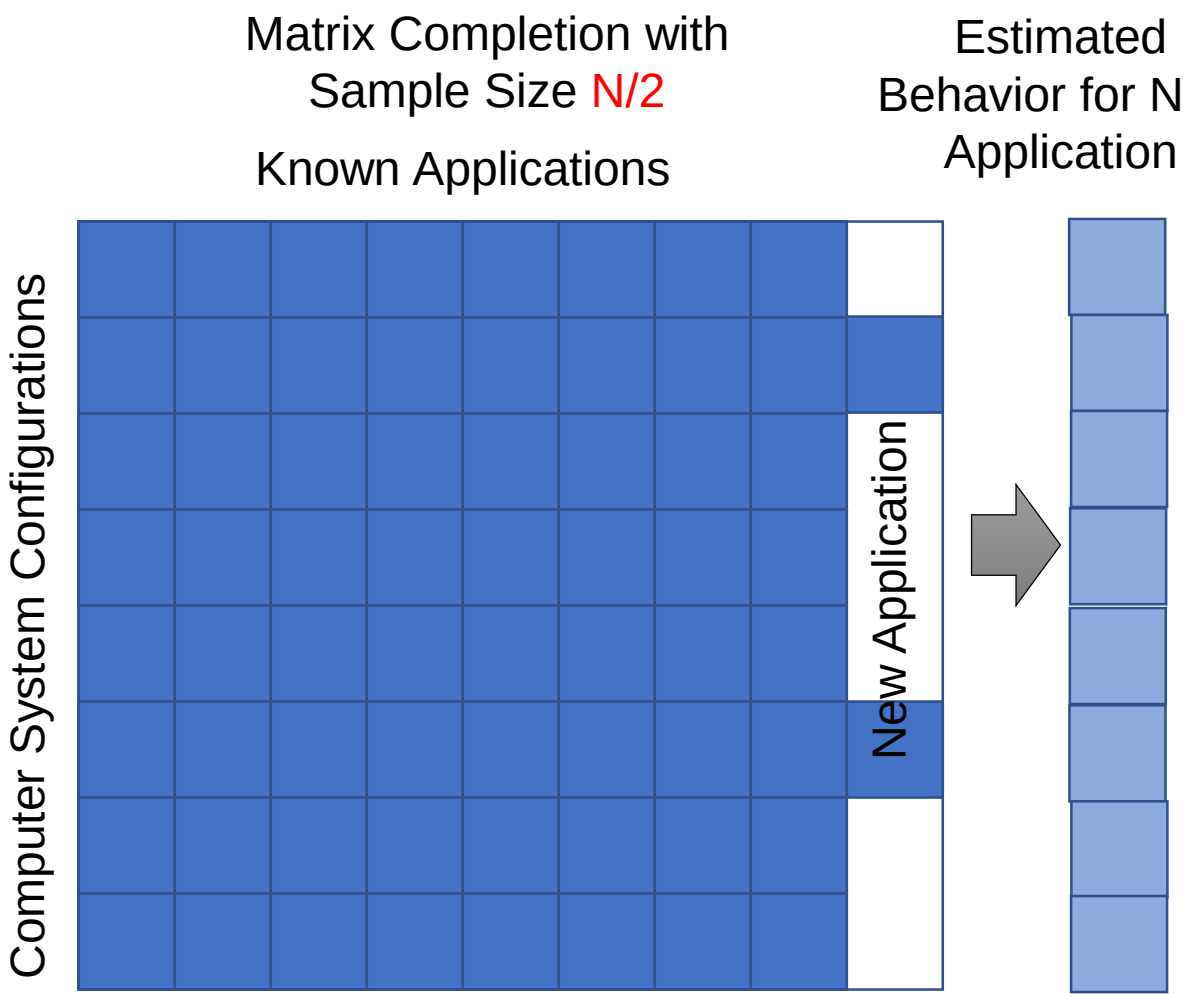
Input: Configuration-Application data matrix, Sampling budget N

Matrix Completion with Sample Size N/2



Multi-phase Sampling

Input: Configuration-Application data matrix, Sampling budget N



Multi-phase Sampling

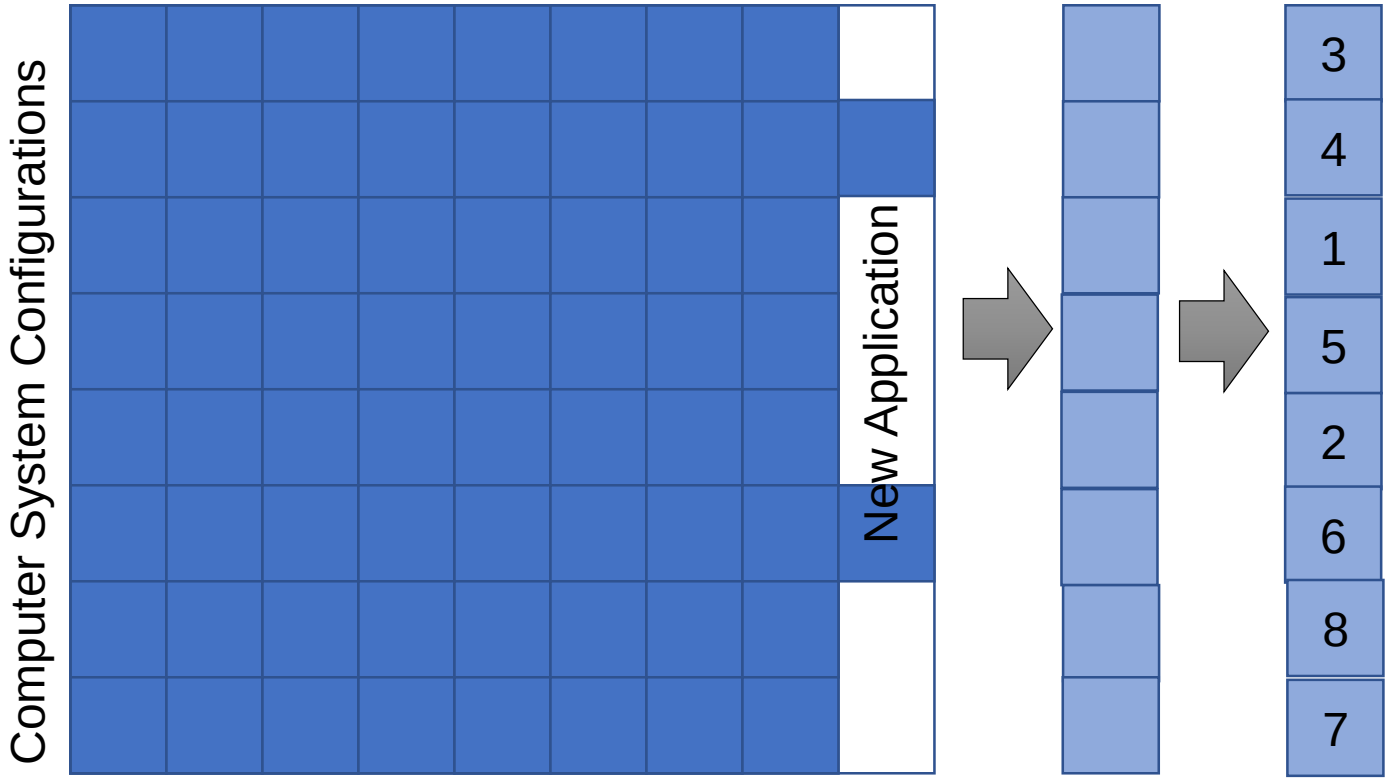
Input: Configuration-Application data matrix, Sampling budget N

Matrix Completion with Sample Size N/2

Known Applications

Estimated Behavior for New Application

Select N/2 Best Configs



$$\text{efficiency} = \frac{\text{estimated performance}}{\text{estimated power}}$$

Multi-phase Sampling

Input: Configuration-Application data matrix, Sampling budget N

Matrix Completion with Sample Size N/2

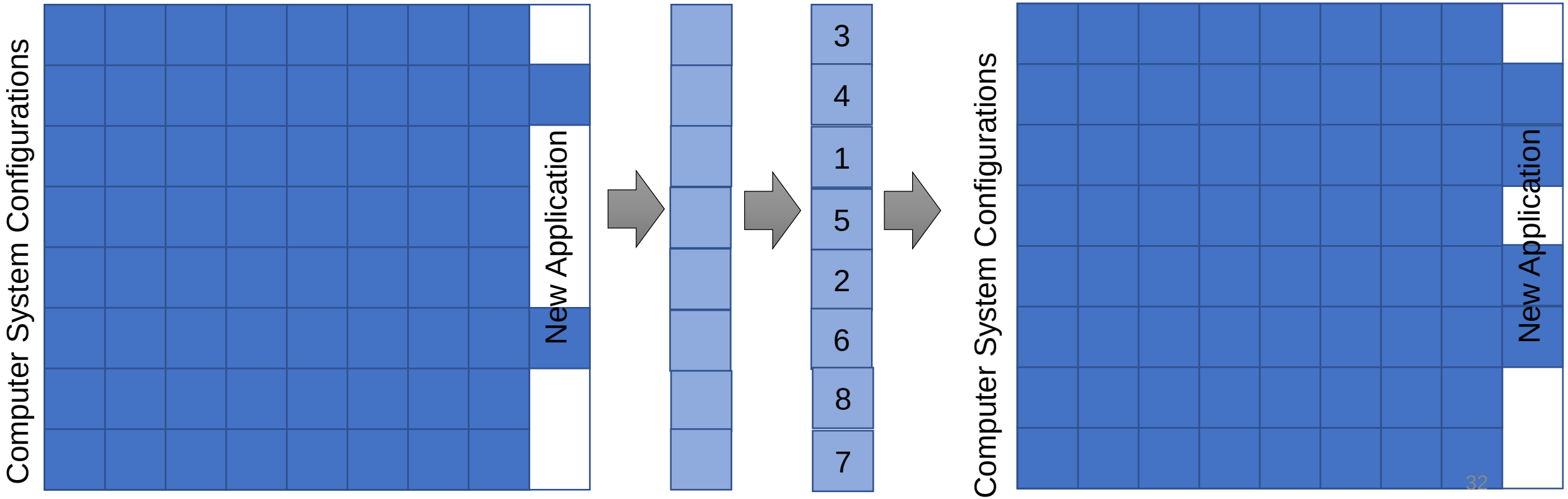
Known Applications

Estimated Behavior for New Application

Select N/2 Best Configs

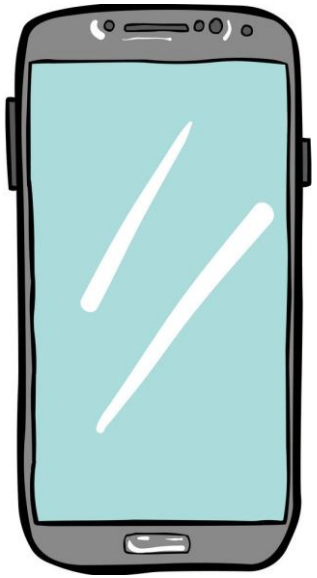
Matrix Completion with N/2 original samples and N/2 estimated best configs

Known Applications



Experimental Setup

	Mobile	Server
System	Ubuntu 14.04	Linux 3.2.0 system
Architecture	ARM big.LITTLE	Intel Xeon E5-2690
# Applications	21	22
# Configurations	128	1024



Learning Models and Frameworks

Learning Models	Category
MCGD	MC
MCMF	MC
Nuclear	MC
WNNM	MC
HBM	Bayesian

First comprehensive study of matrix completion (MC) algorithms for systems optimization task

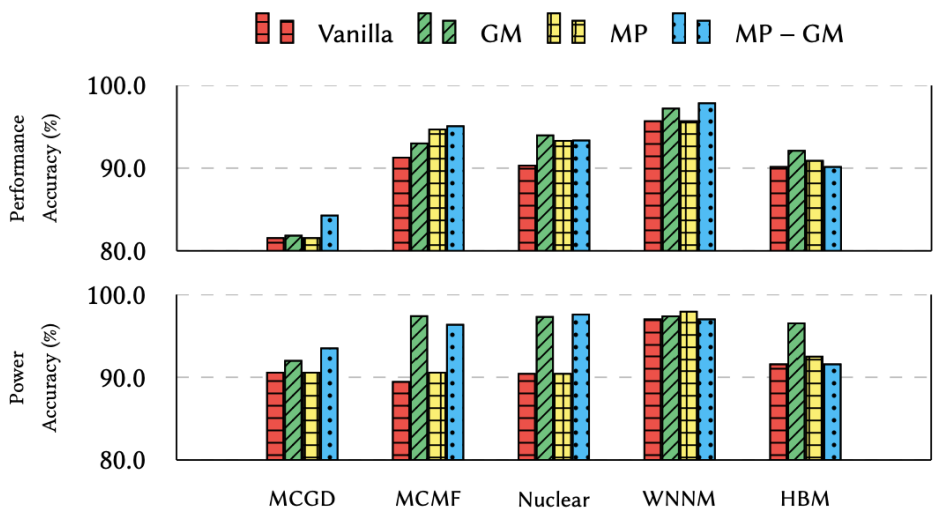
Learning Models and Frameworks

Learning Models	Category
MCGD	MC
MCMF	MC
Nuclear	MC
WNNM	MC
HBM	Bayesian

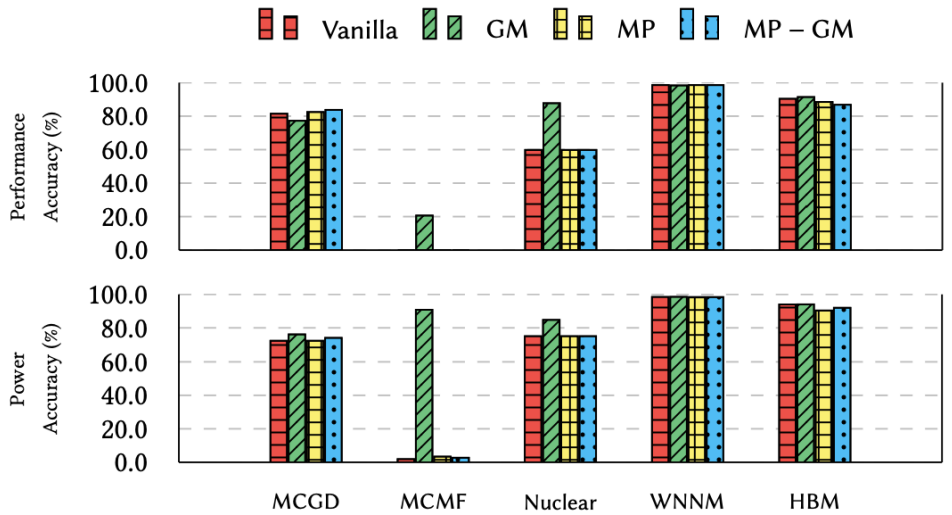
First comprehensive study of matrix completion (MC) algorithms for systems optimization task

Frameworks	Definitions
Vanilla	Basic learners
GM	Generative model
MP	Multi-phase sampling
MP-GM	Combine GM and MP

Improve Prediction Accuracy w/ GM



Mobile



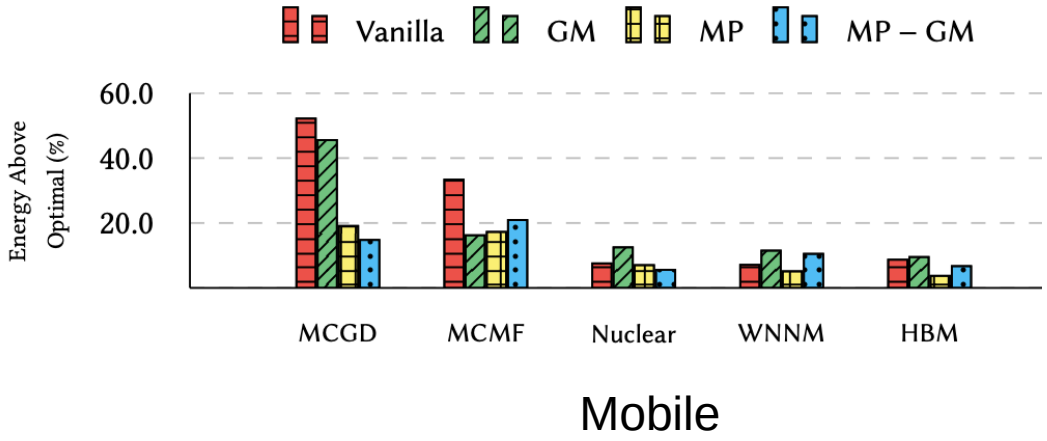
Server

High is Better

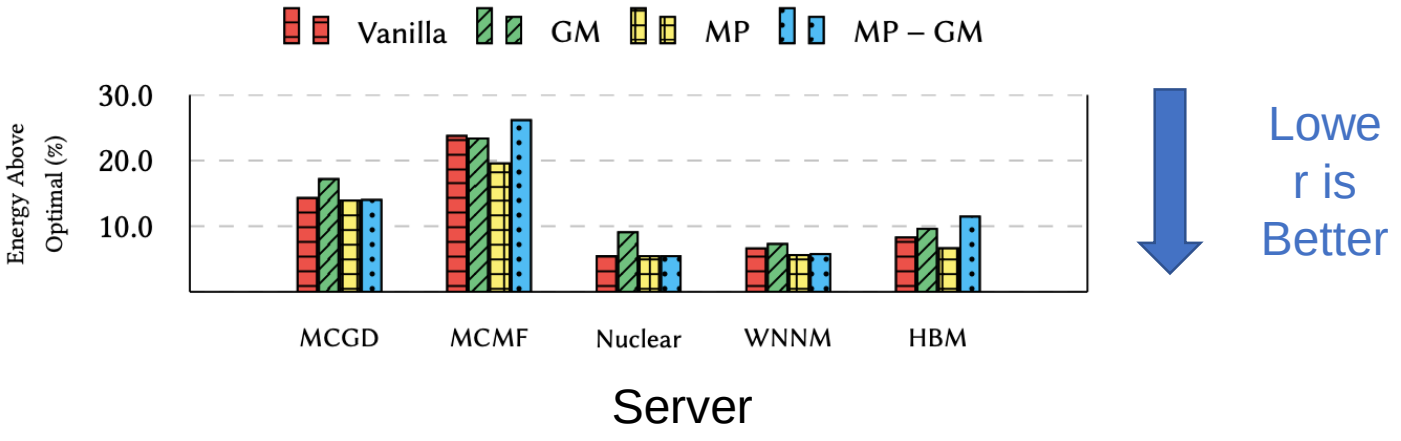
Average percentage points of accuracy improvement

		GM	MP	MP – GM
Mobile	Performance	1.8	1.4	2.3
	Power	4.3	0.6	3.4
Server	Performance	9.0	−0.2	−0.3
	Power	20.5	−0.4	0.1
Average		8.9	0.4	1.4

Improve Energy Savings w/ MP



Mobile



Server

Average energy improvement

	GM	MP	MP – GM
Mobile	−14%	41%	22%
Server	−22%	11%	−6.5%

Summary

- Applying ML to systems requires:
  - Data
  - Answers
  - Linear Algebra
- Many Examples of Learning for Systems:
  - Huge diversity of techniques
  - Common thread is that predictions by themselves are not enough
- Structure:
  - Often constrained optimization problem
- Structure of many systems problems means that:
  - Accuracy does not provide better systems results
  - Understanding the structure will lead to better outcomes