Accelerating Approximate Data Analysis with Parallel Processors

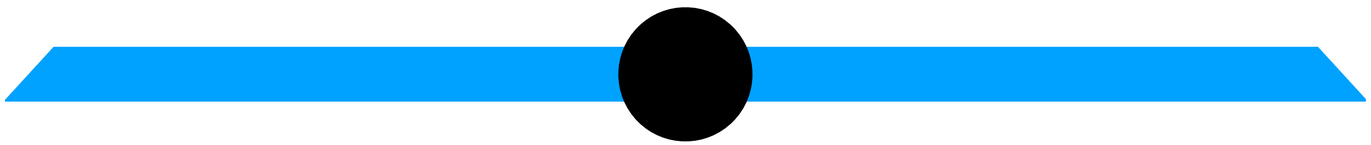
Martin Kiefer

PhD Defense

2.2.2023

Approximate Data Analysis

Trading accuracy for faster / cheaper results



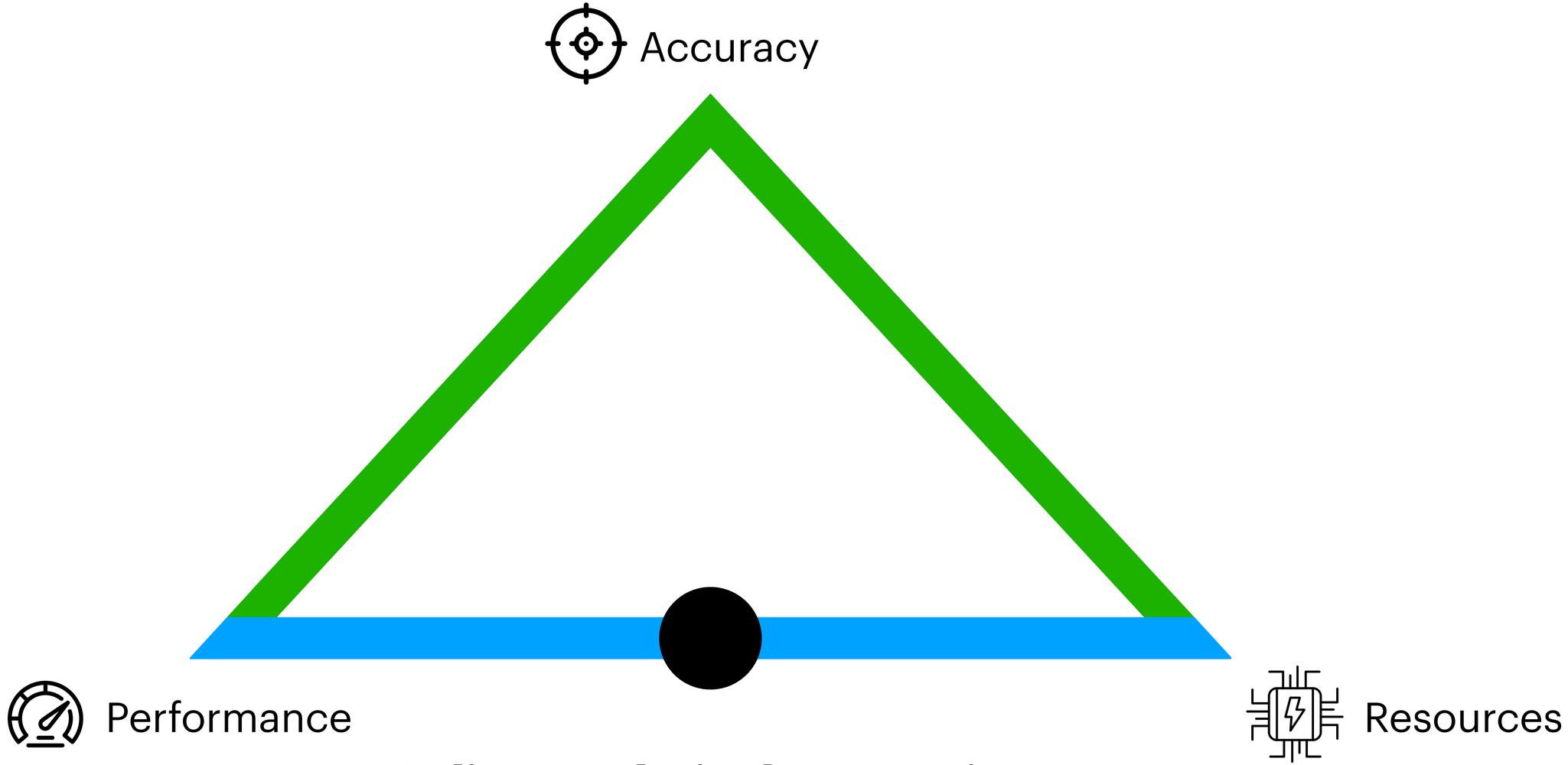
Performance

Resources

Online Analytical Processing

Approximate Data Analysis

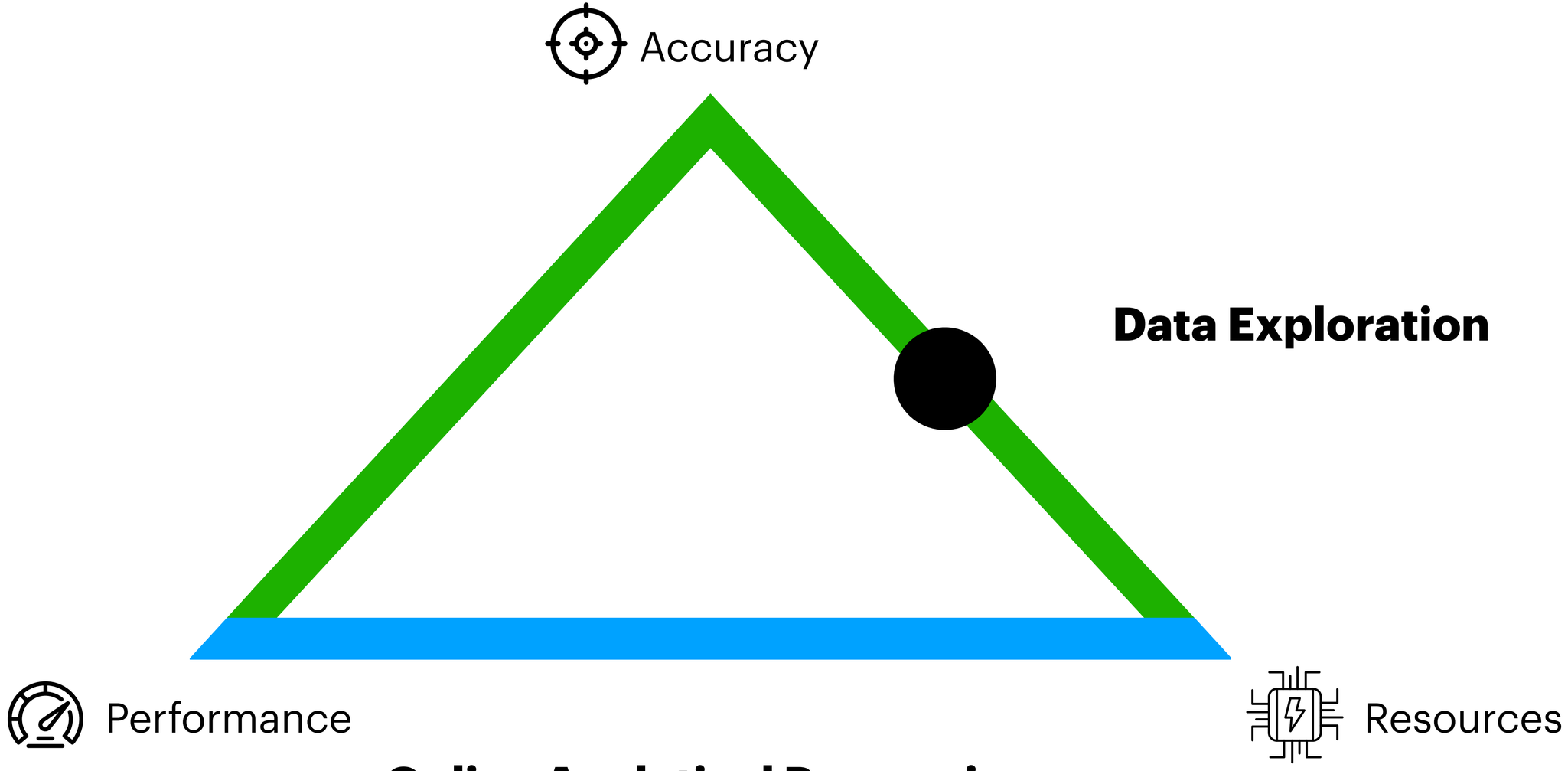
Trading accuracy for faster / cheaper results



Online Analytical Processing

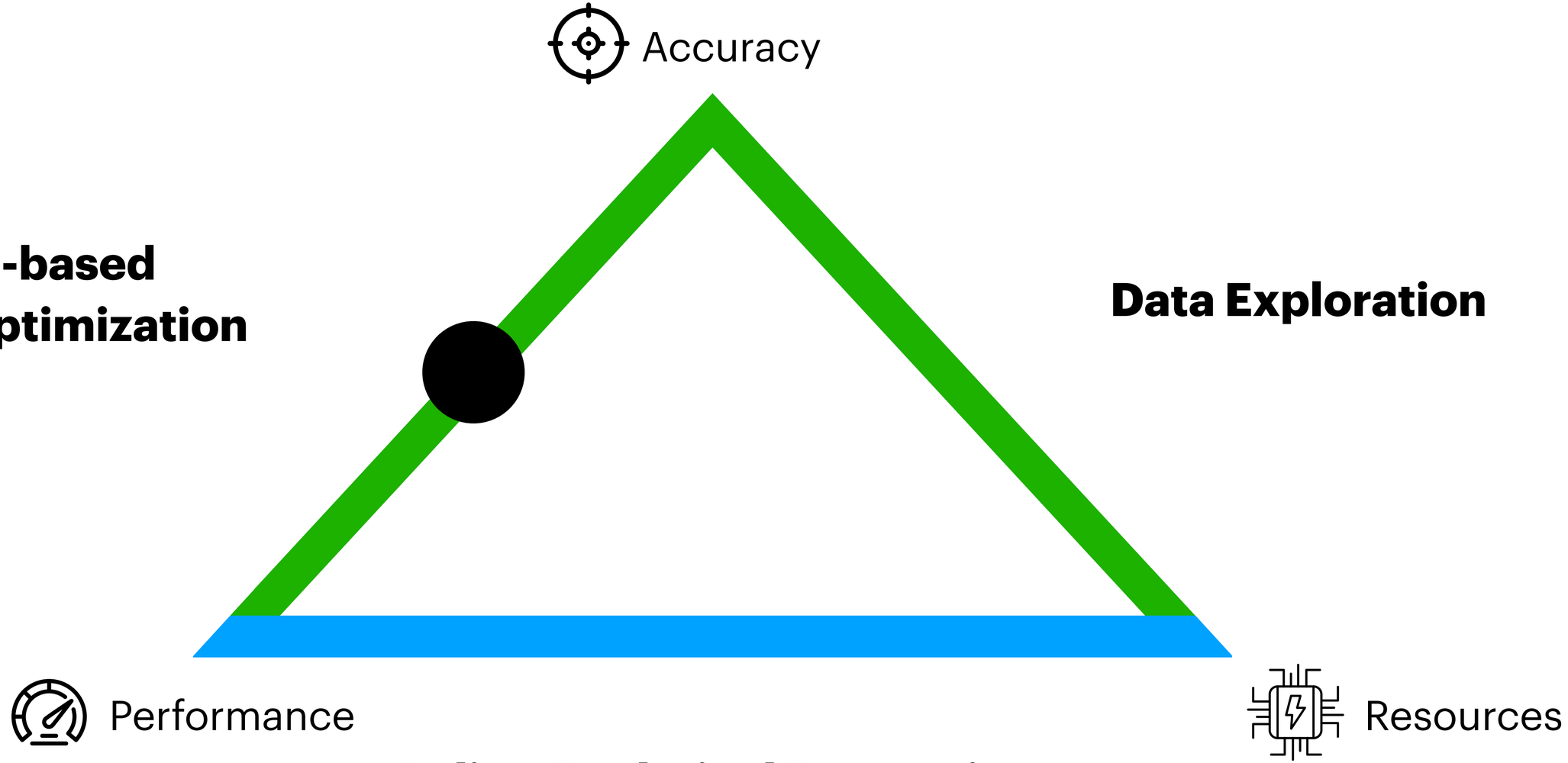
Approximate Data Analysis

Trading accuracy for faster / cheaper results



Approximate Data Analysis

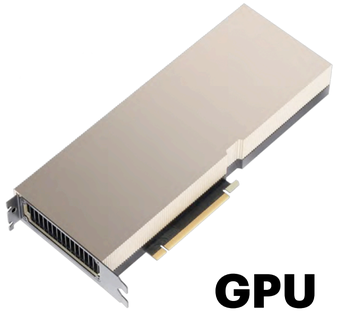
Why go for approximations?



Specialized Parallel Processors

Highest efficiency for certain computations

- **Graphics Processing Unit**
  - Thousands of parallel threads
  - Throughput optimized



GPU

- **Field-Programmable Gate Array**
  - Parallelism in custom circuits
  - Reprogrammable ressources

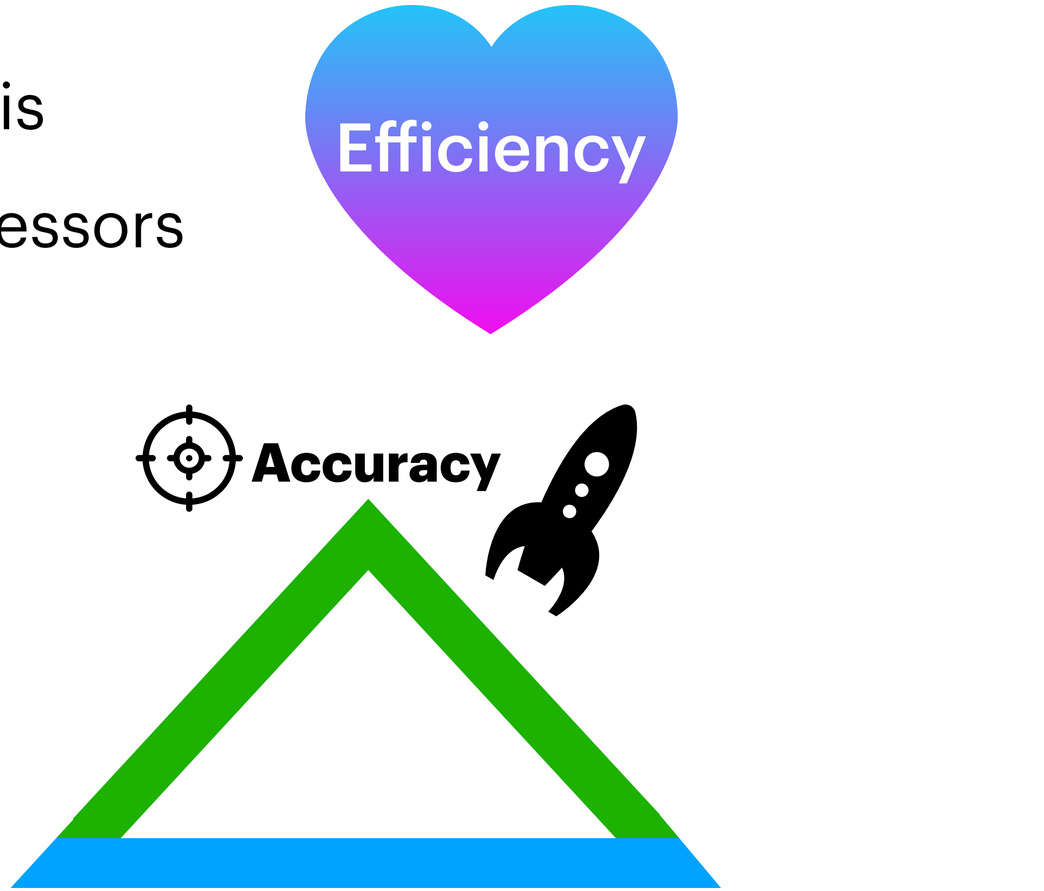
FPGA

- Higher performance, lower energy consumption
- Embrace parallelism, properties, programming models

Approximate Data Analysis + Parallel Processors

Efficiency beyond the approximation trade-off

- Approximate data analysis
- Specialized parallel processors



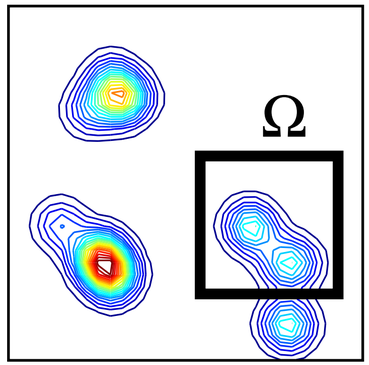
Approach
2 Use Cases, 2 Goals

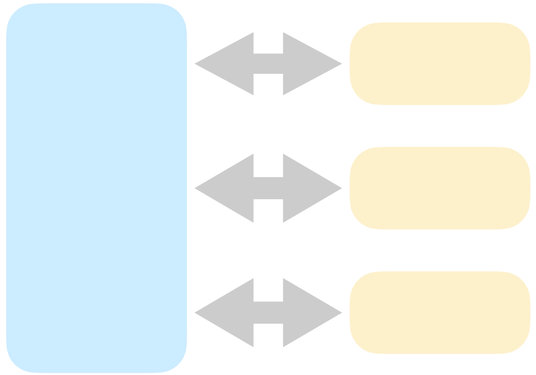
- For two relevant use cases, we
  - generalize existing approaches
  - hide the complexity of the processor

- **We solve problems limiting the applicability of the approaches**

Overview

Use Case #1: GPU-Accelerated Selectivity Estimation





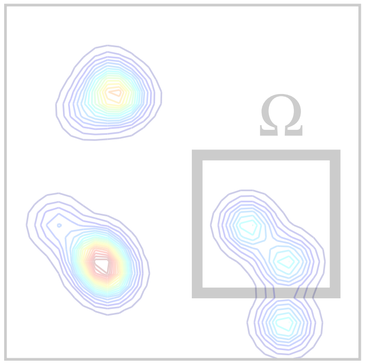
Estimating Join Selectivities using Bandwidth-Optimized Kernel Density Models

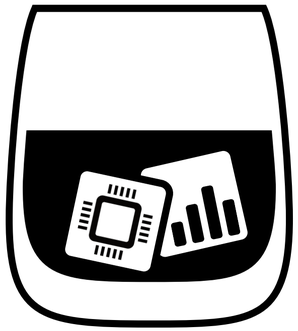
Martin Kiefer, Max Heimel, Sebastian Breß, Volker Markl

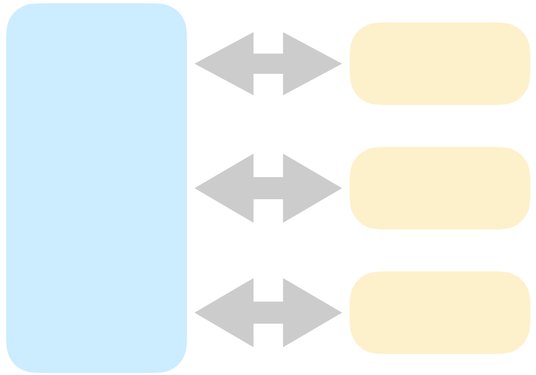
PVLDB, 10 (13), 2017

Overview

Use Case #2: FPGA-Accelerated Sketching







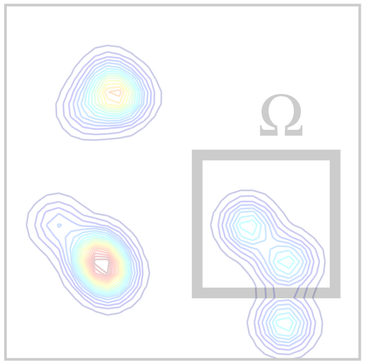
Scotch: Generating FPGA Accelerators for Sketching at Line Rate

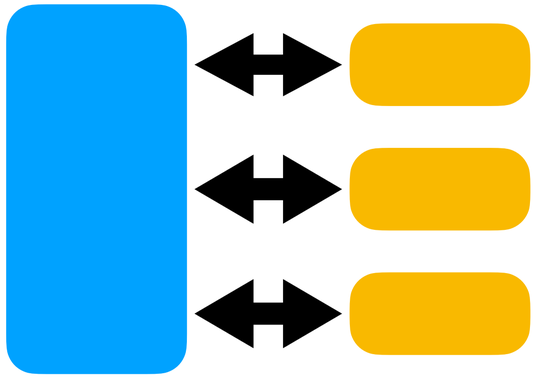
Martin Kiefer, Ilias Poulakis, Sebastian Breß, Volker Markl

PVLDB, 14 (3), 2021

Overview

Use Case #2: FPGA-Accelerated Sketching





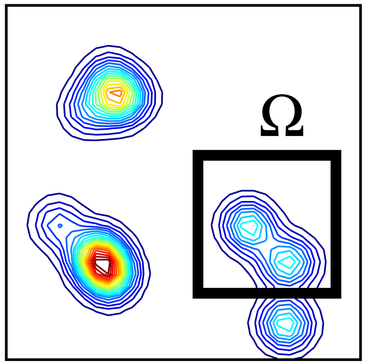
Optimistic Data Parallelism for FPGA-Accelerated Sketching

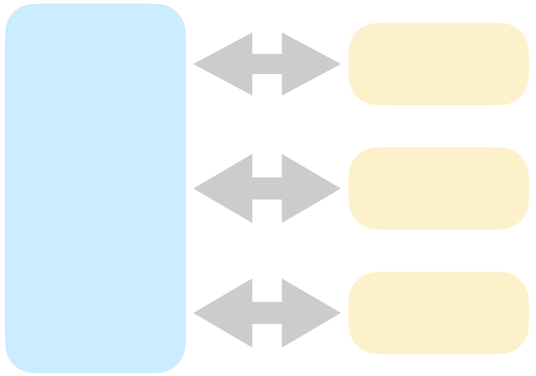
Martin Kiefer, Ilias Poulakis, Eleni Tzirita Zacharatou, Volker Markl

PVLDB, 16 (5), 2023

GPU-Accelerated Selectivity Estimation

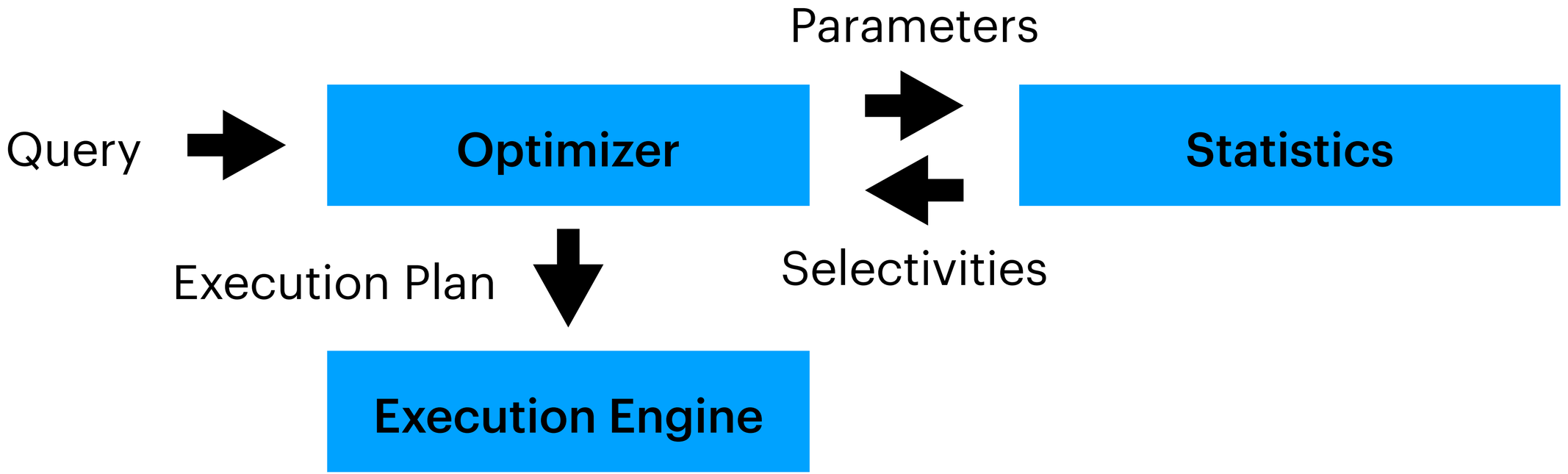
Use Case #1





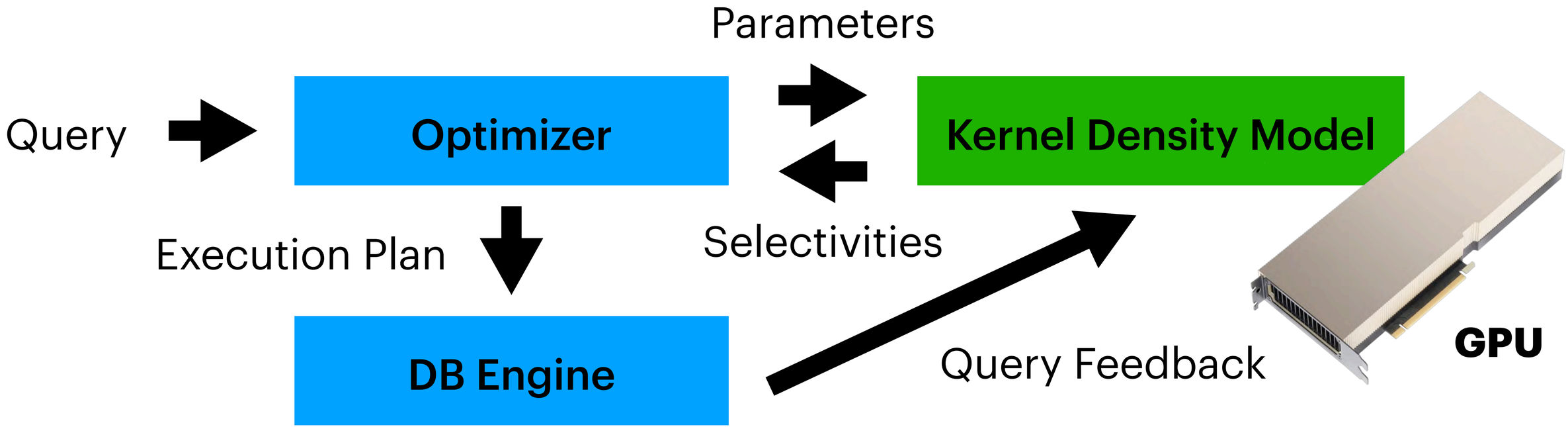
Use Case 1: GPU-Accelerated Selectivity Estimation

Statistical Coprocessing in Relational Databases



Use Case 1: GPU-Accelerated Selectivity Estimation

Statistical Coprocessing in Relational Databases



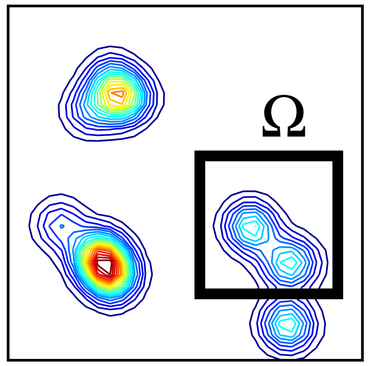
Problem: Limited operator support, no joins

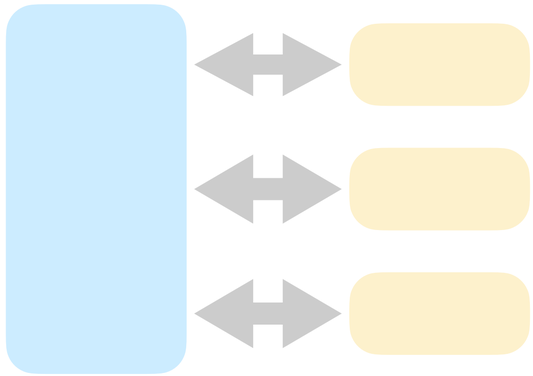
Self-Tuning, GPU-Accelerated Kernel Density Models for Multidimensional Selectivity Estimation. Max Heimel, Martin Kiefer, Volker Markl. SIGMOD'15.

Estimating Join Selectivities using Bandwidth-Optimized Kernel Density Models

Martin Kiefer, Max Heimel, Sebastian Breß, Volker Markl

Proceedings of the VLDB Endowment, 10 (13), 2017

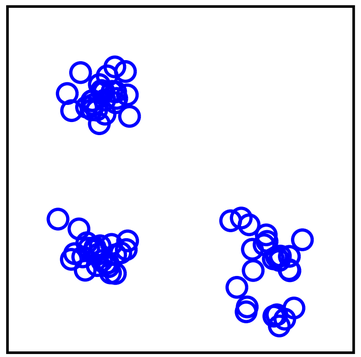




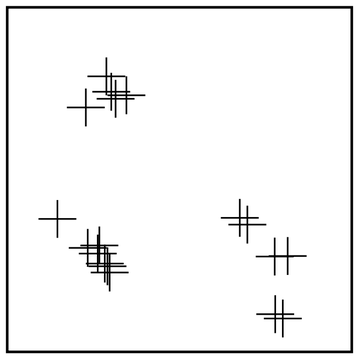
Kernel Density Estimation

Sampling meets histogram

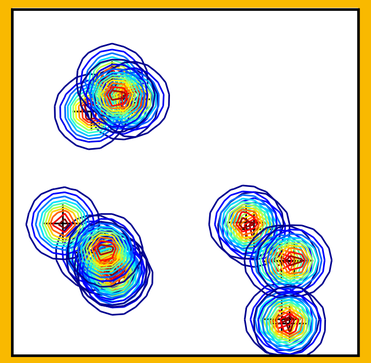
$$\hat{P}(\vec{x}) = \frac{1}{|S|} \sum_{i=1} K_H(\vec{s}_i - \vec{x})$$



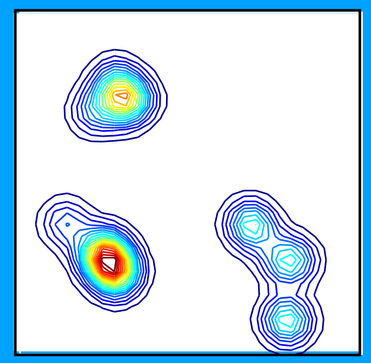
Dataset



Sample



Kernels K_H

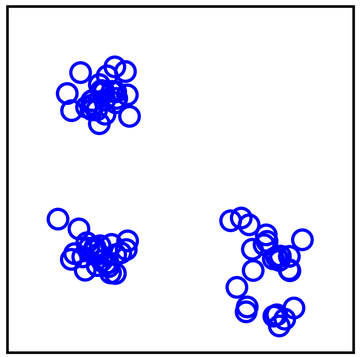


Estimate P_H

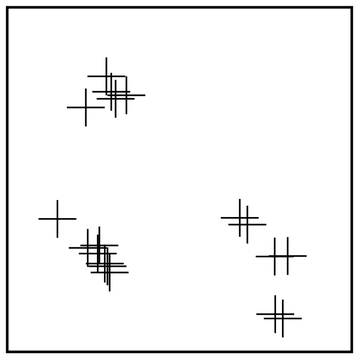
Kernel Density Estimation

Estimating the selectivity of a region Ω

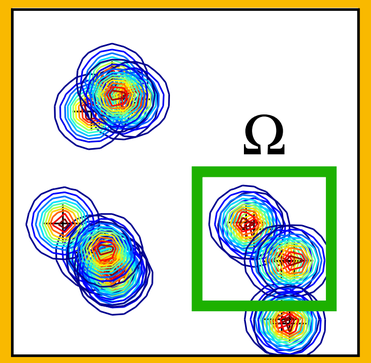
$$\hat{P}(\vec{x}) = \frac{1}{|S|} \sum_{i=1} \int_{\Omega} K_H(\vec{s}_i - \vec{x})$$



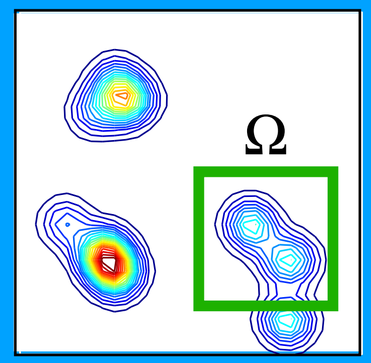
Dataset



Sample



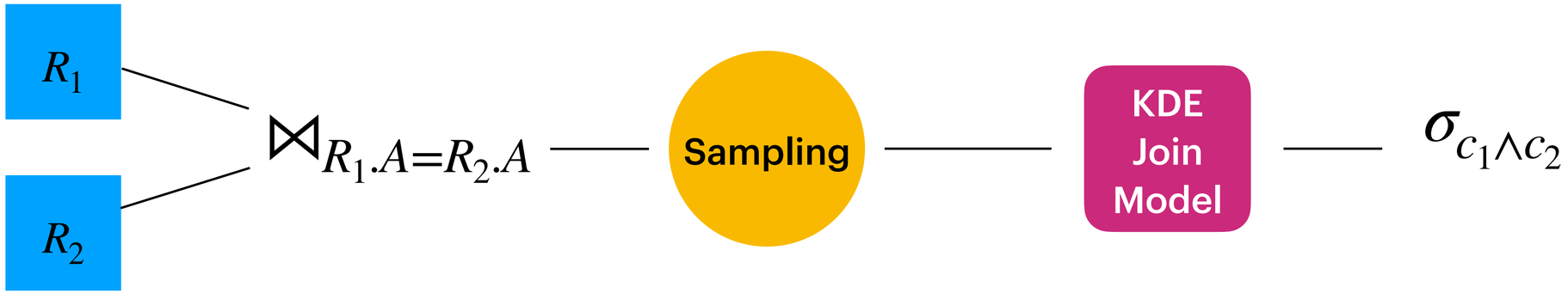
Kernels K_H



Estimate P_H

Sampling from the Join Result

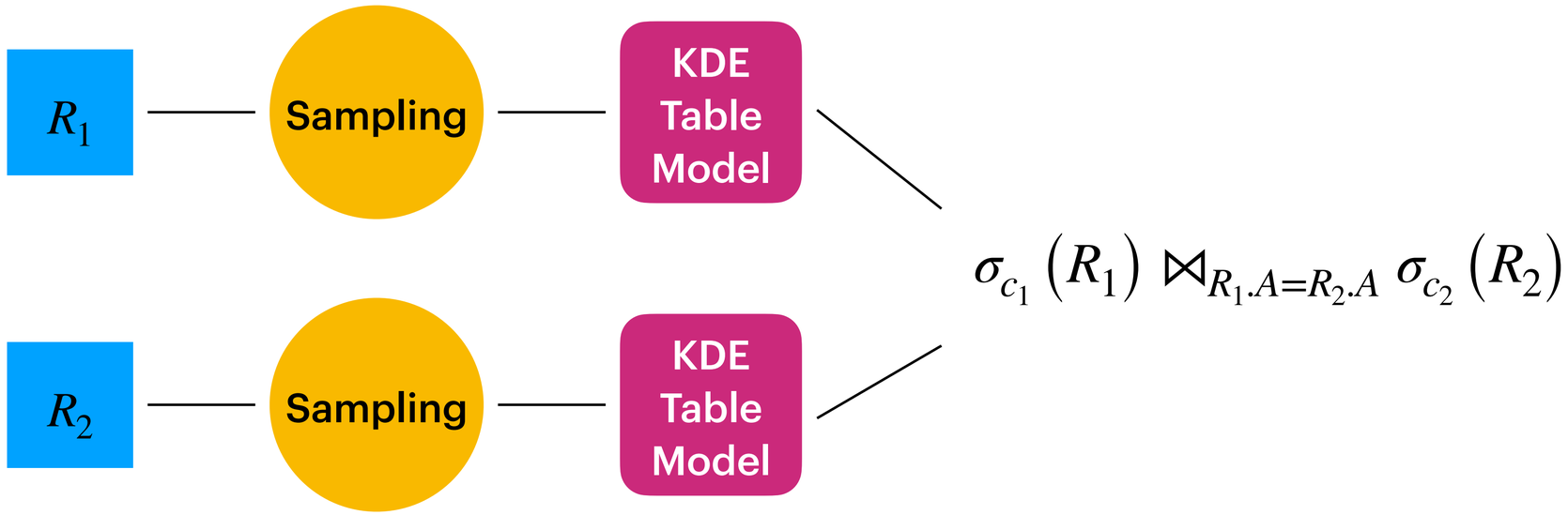
Just treat the join like a base table



- **Benefits**
  - Accurate model, no modifications required
- **Drawbacks**
  - KDE model tailored to one particular join

Combining Base Table Models

Modeling the join explicitly



- **Benefits**
  - Models can handle both base selections and joins
- **Drawbacks**
  - The join has to be applied explicitly on the estimated distribution

Computing joins over KDE models

Avoiding cross products

Sum over cross product

$$\sum_{i=1,j=1}^{s_1,s_2} \hat{p}_1{}^{(i)}(c_1) \cdot \hat{p}_2{}^{(j)}(c_2) \cdot \hat{J}_{i,j}$$

Computing joins over KDE models

Invariant Contributions

Sum over cross product

$$\sum_{i=1,j=1}^{s_1,s_2} \hat{p}_1^{(i)}(c_1) \cdot \hat{p}_2^{(j)}(c_2) \cdot \hat{J}_{i,j}$$

Invariant Contributions

(Effect of selections, depend on individual samples)

Filter samples by invariant contributions

(Predicate pushdown)

Computing joins over KDE models

Cross contributions

Sum over band join

$$\sum_{i=1,j=1}^{s_1,s_2} \hat{p}_1{}^{(i)}(c_1) \cdot \hat{p}_2{}^{(j)}(c_2) \cdot \underbrace{\hat{J}_{i,j}}$$

Cross Contributions

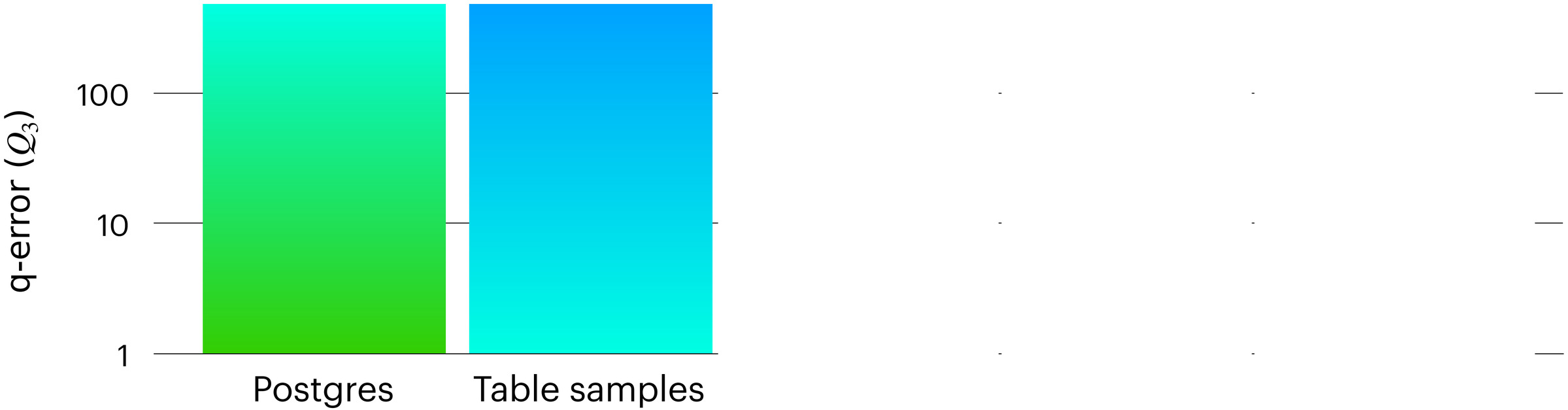
(Effect of the join, similarity function)

Join only sufficiently close tuples

(Binary Search Band Join)

Evaluation: Estimator Accuracy

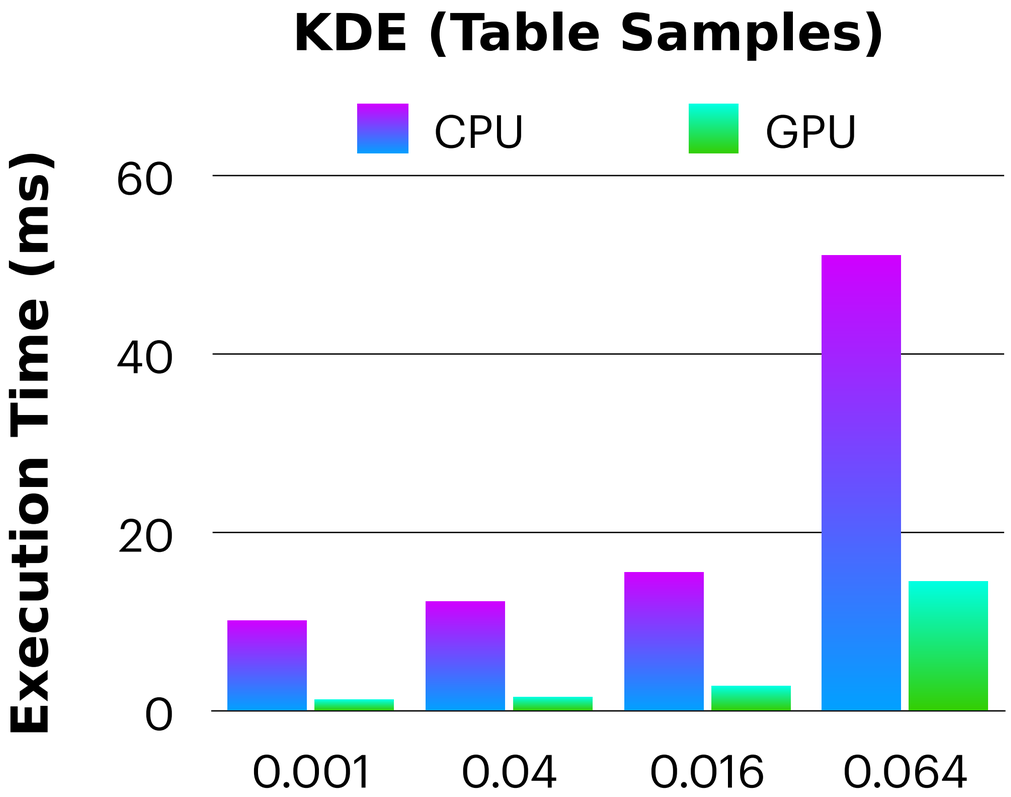
KDE prevents reduces severe misestimations



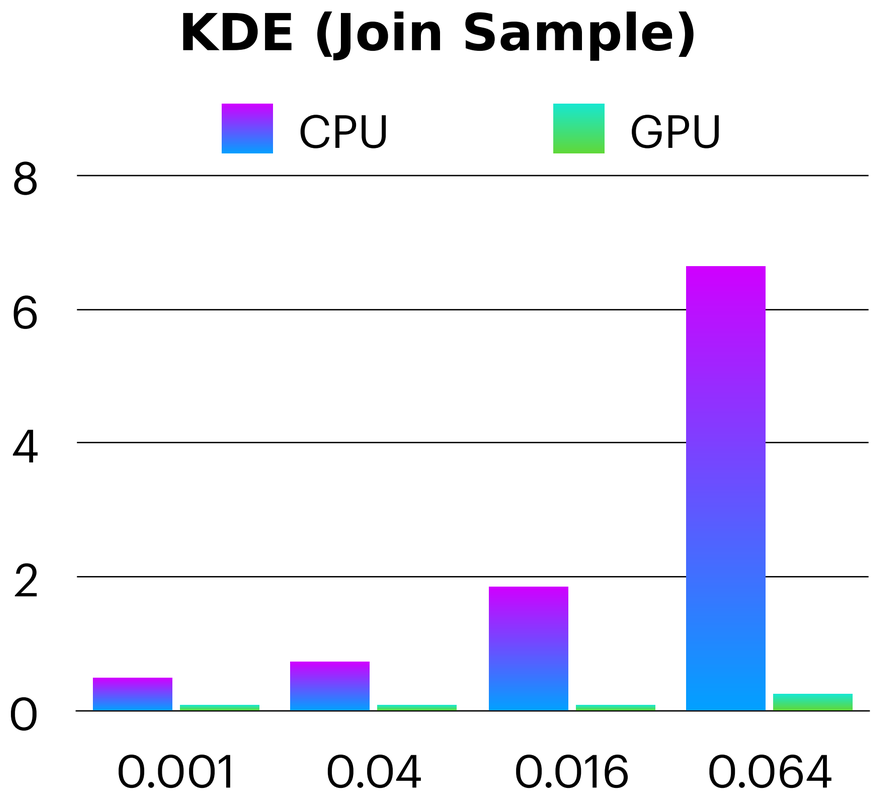
DMV, Distinct Workload, 3 joins, 8 selections, 300 Training Queries, 100 Test Queries, 1% sample size

Evaluation: Estimator Accuracy

KDE prevents reduces severe misestimations



GPU: 4x - 40x larger model sizes



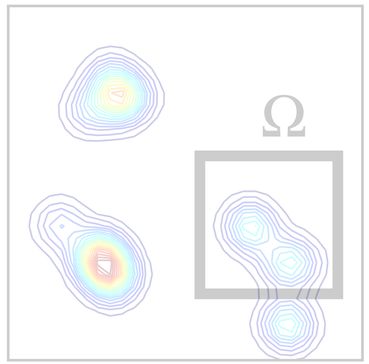
GPU: 30x - 140x larger model sizes

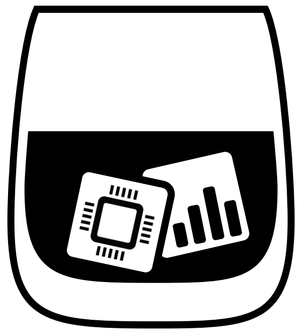
IMDb, 3 joins, 6 selections, 300 Training Queries, 50 KB - 6 MB

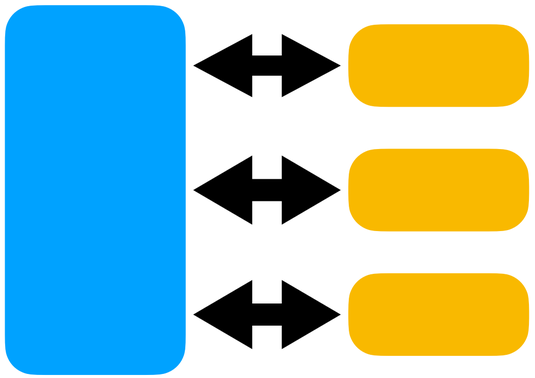
GPU: Nvidia V100, CPU: 2x Intel Xeon Gold 5115

FPGA-Accelerated Sketching

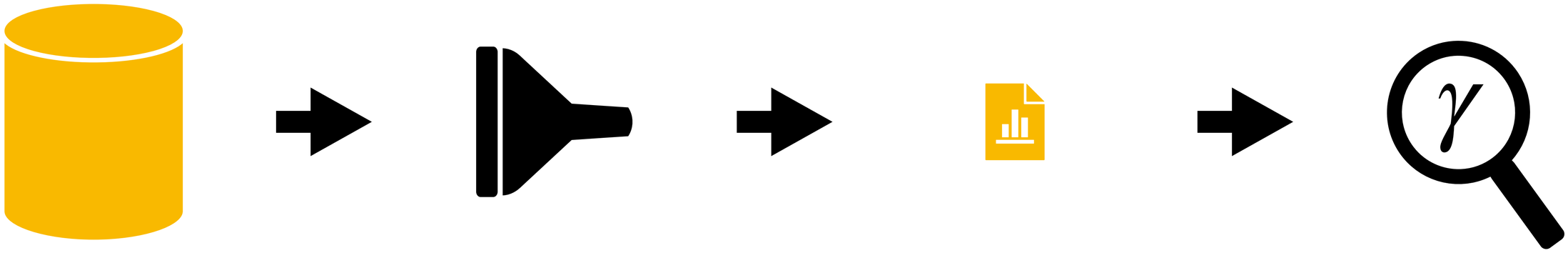
Use Case #2





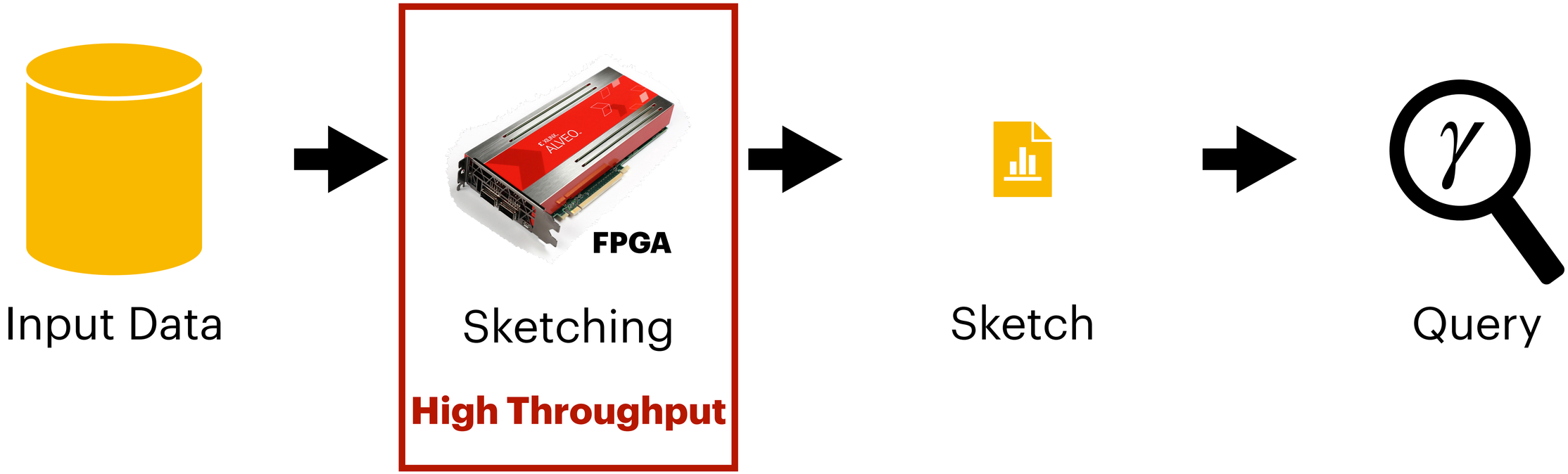


FPGA-Accelerated Sketching
for Approximate Query Processing



Use Case 2: FPGA-Accelerated Sketching

for Approximate Query Processing



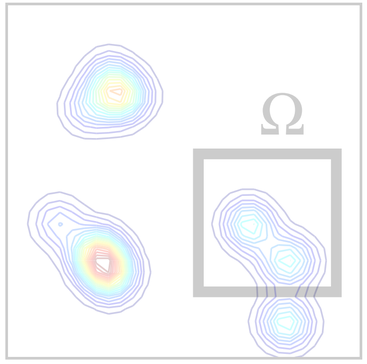
Problem 1: An FPGA expert is required

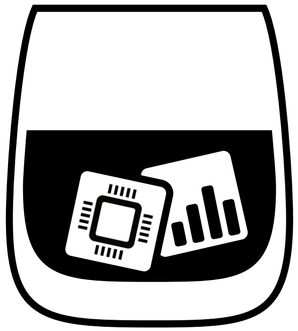
Problem 2: High resource consumption for replicated data parallelism

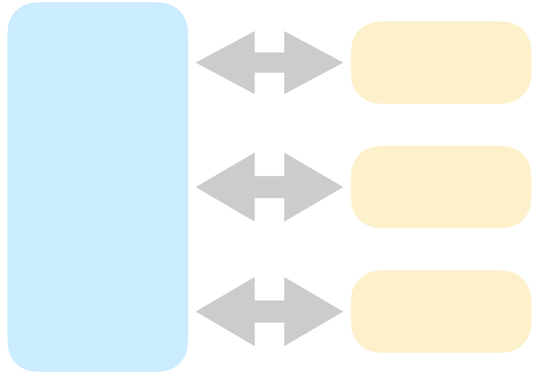
Scotch: Generating FPGA Accelerators for Sketching at Line Rate

Martin Kiefer, Ilias Poulakis, Sebastian Breß, Volker Markl

Proceedings of the VLDB Endowment, 14 (3), 2021

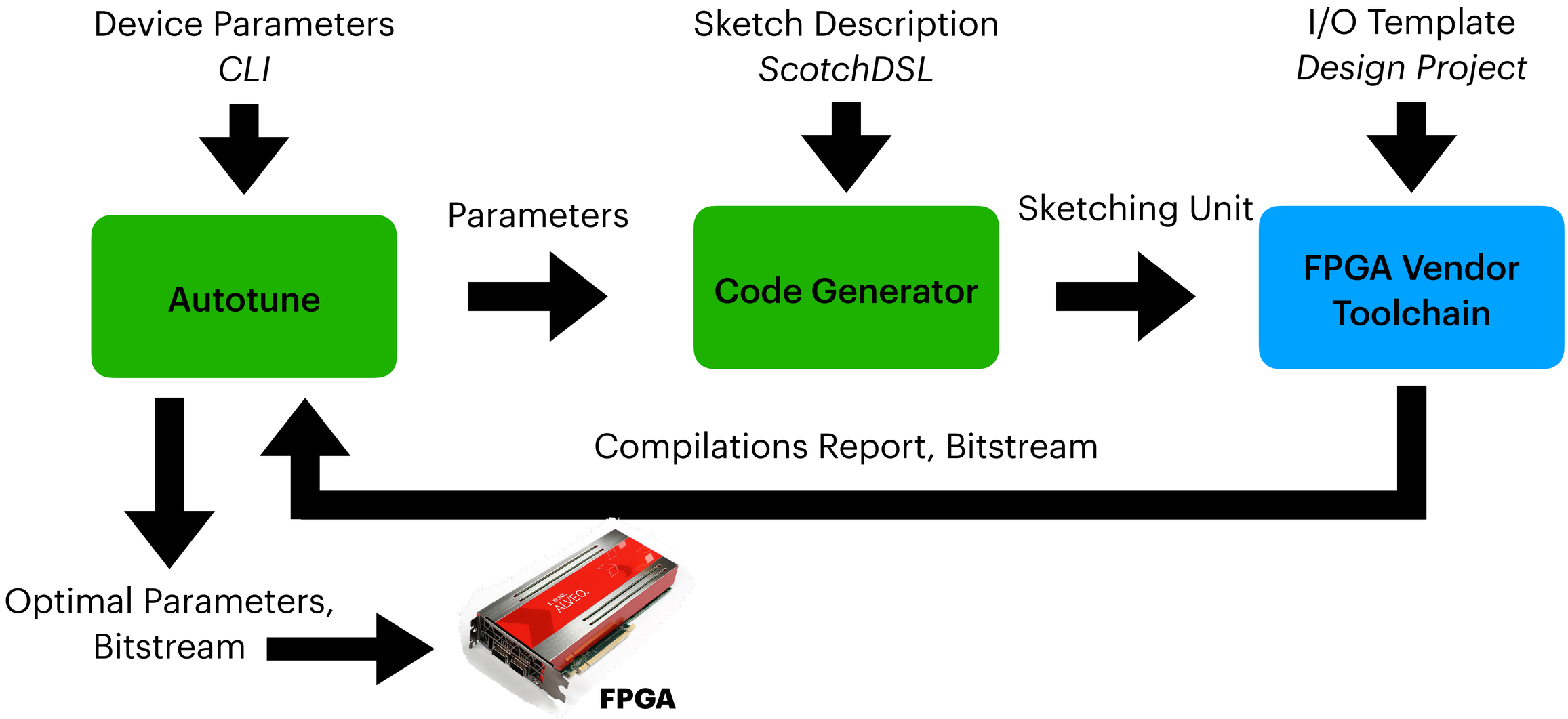






Scotch System Architecture

Generating and optimizing in a feedback-loop



Scotch System Architecture

Generating and optimizing in a feedback-loop

Device Parameters
CLI

Sketch Description
ScotchDSL

I/O Template
Design Project

Autotune

Parameters

Code Generator

Sketching Unit

FPGA Vendor Toolchain

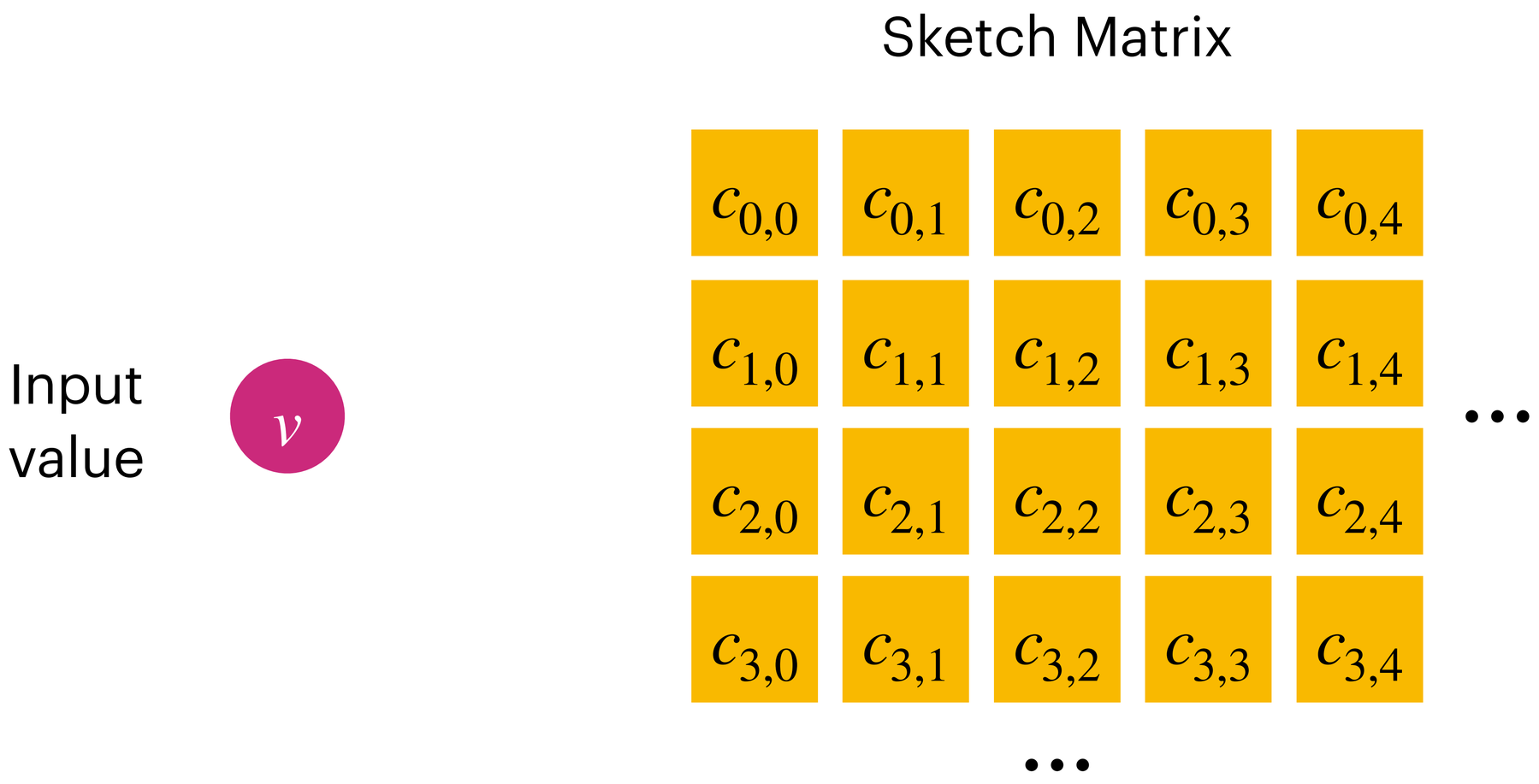
Compilations Report, Bitstream

Optimal Parameters,
Bitstream

FPGA

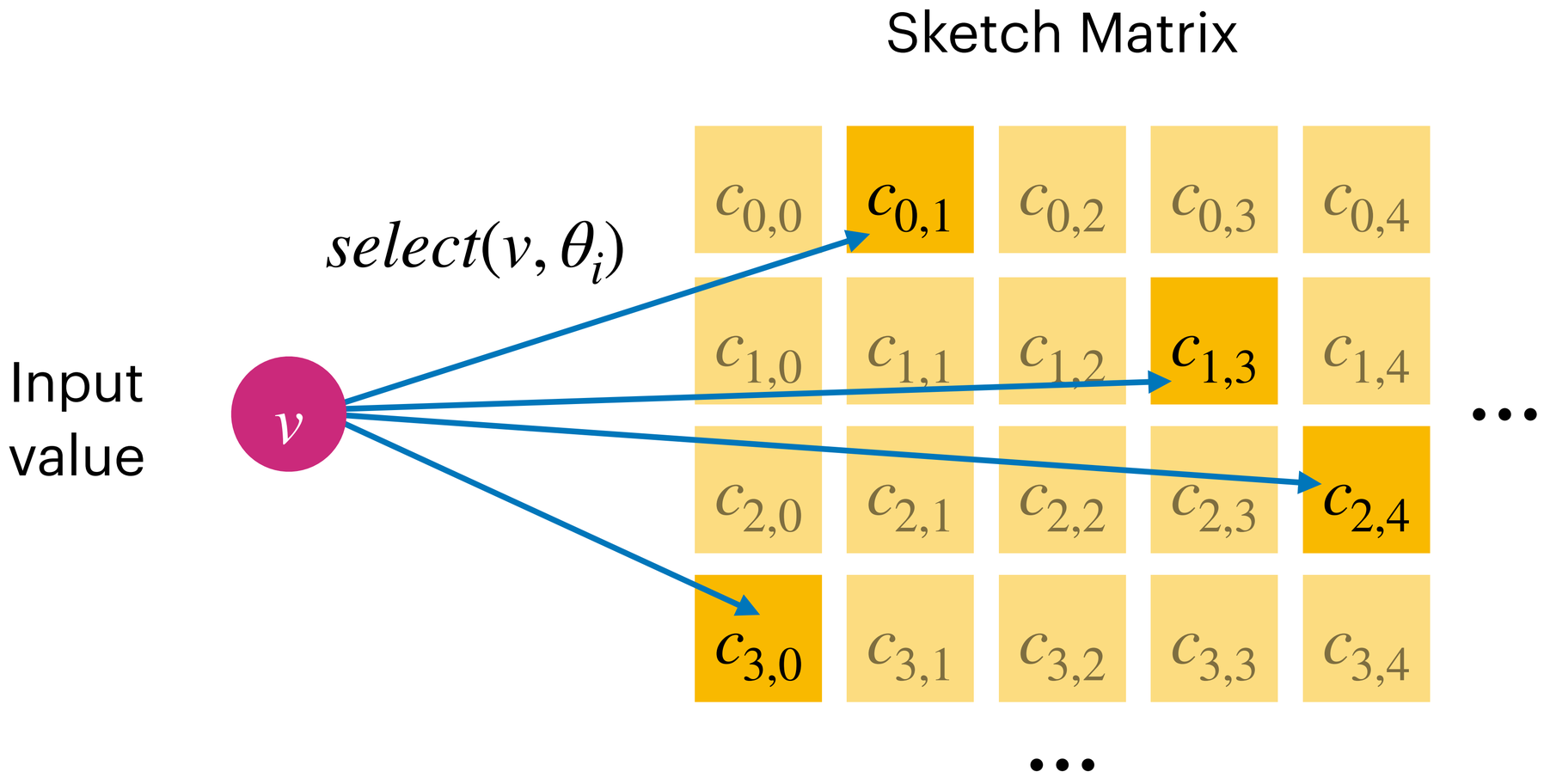
Select-Update Model

Lightweight sketch specification



Select-Update Model

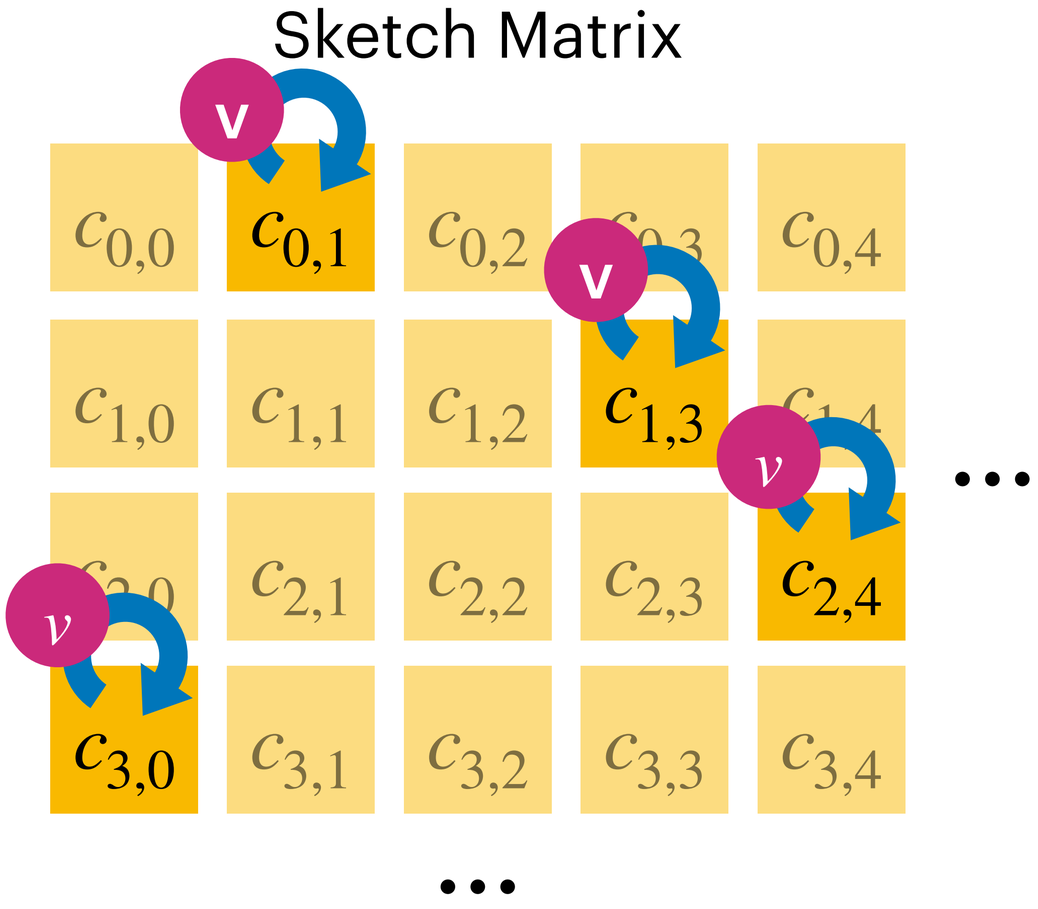
Select one entry in each row



Select-Update Model

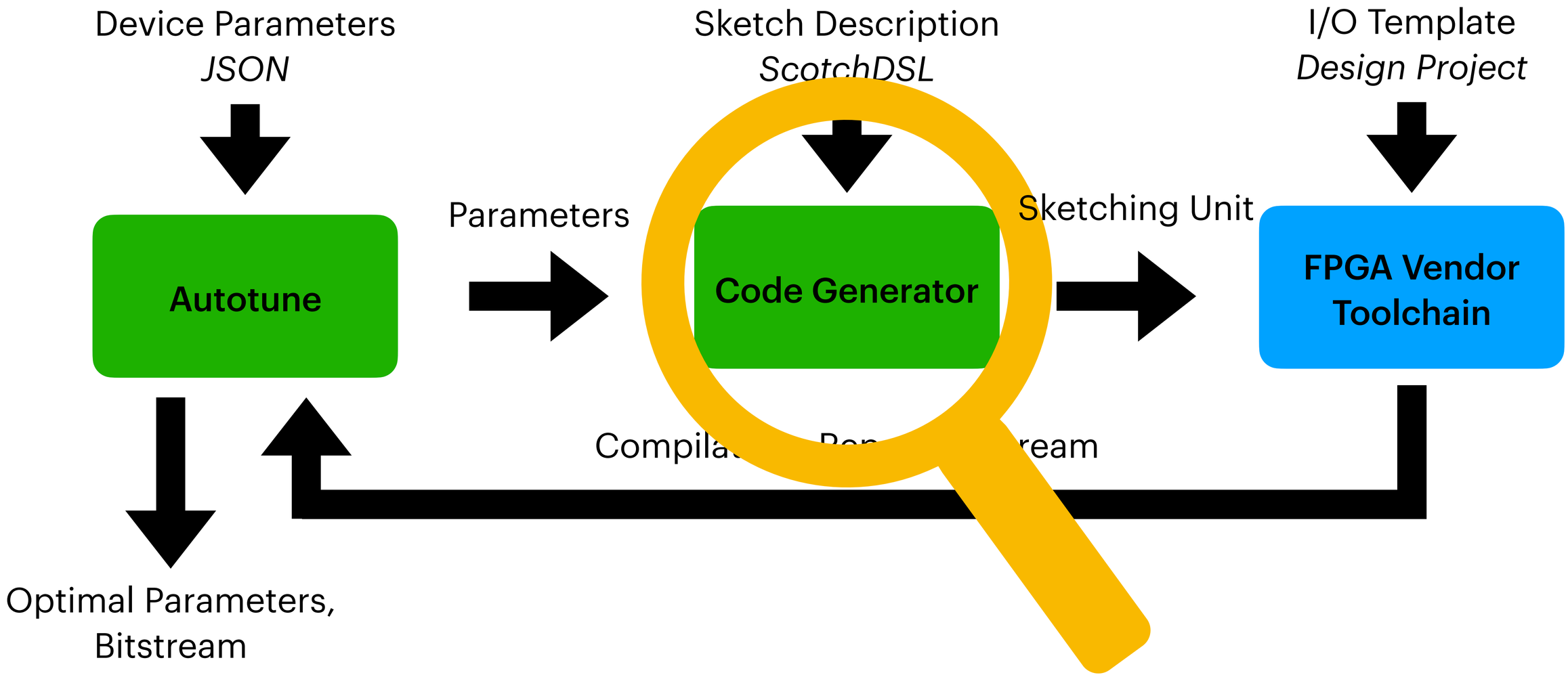
Update the selected entry

$update\ (v, \theta_i, c_{0,1})$



Scotch System Architecture

Generating and optimizing in a feedback-loop



High-Level RTL Architecture

Processing at line-rate

For each row:

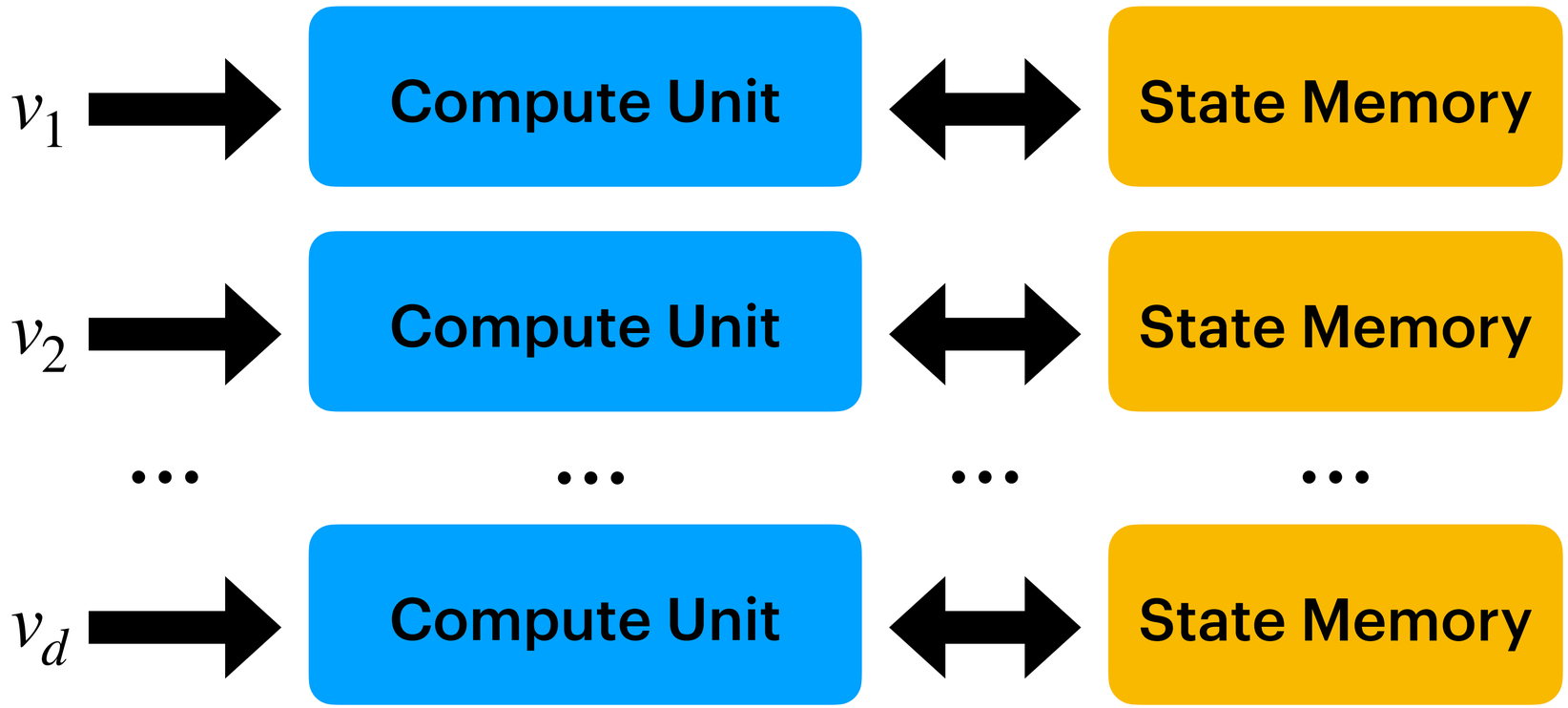
- Computes evaluates select/update
- One input value per cycle

- Pipelined random access memory
- One read & write operation per cycle

Data Parallelism

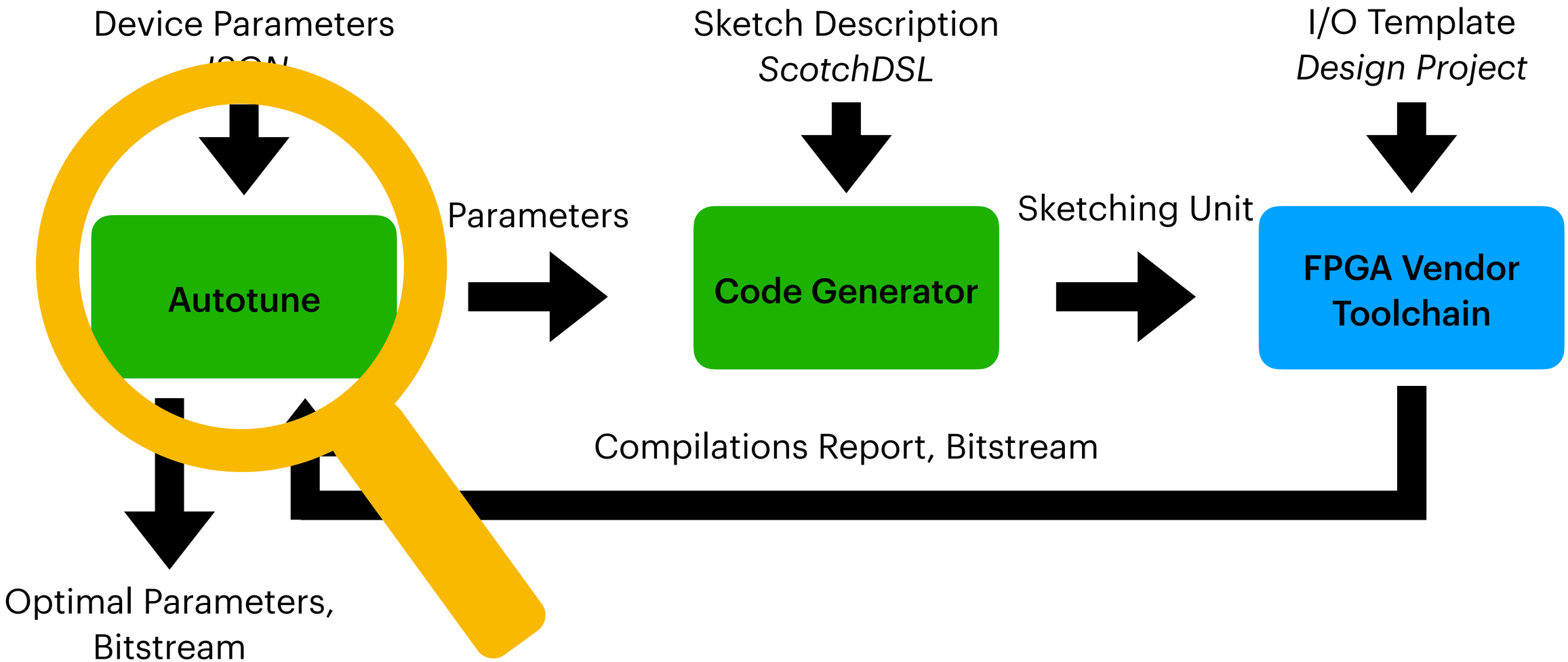
General Approach: Processing at line-rate via full replication

For each row:



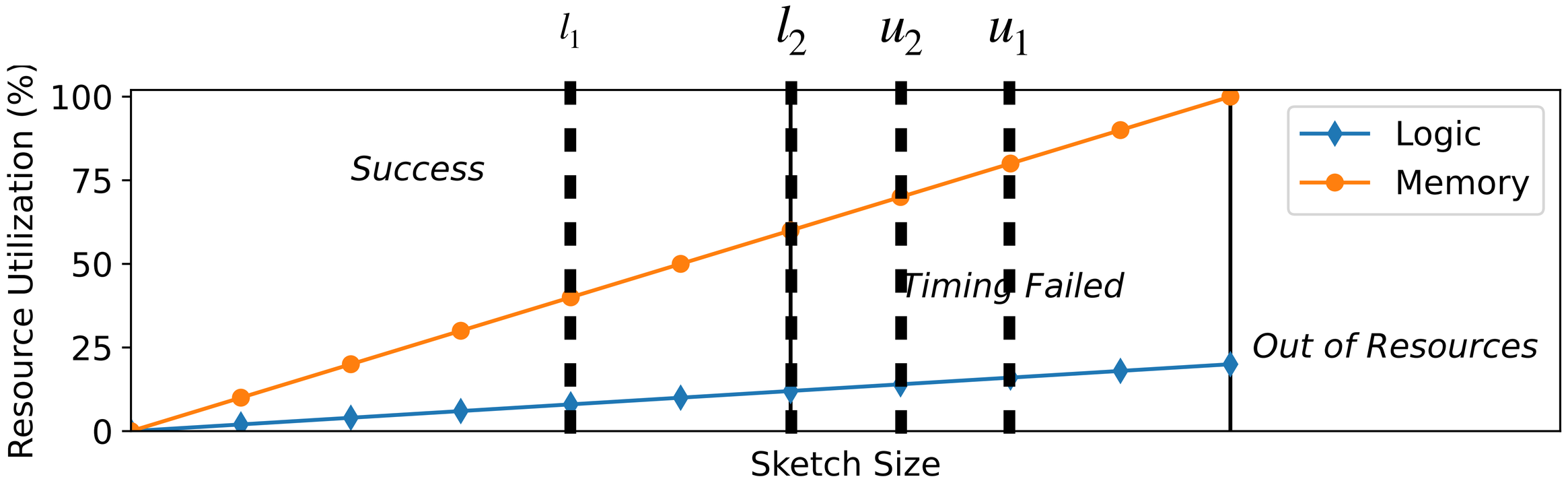
Scotch System Architecture

Generating and optimizing in a feedback-loop



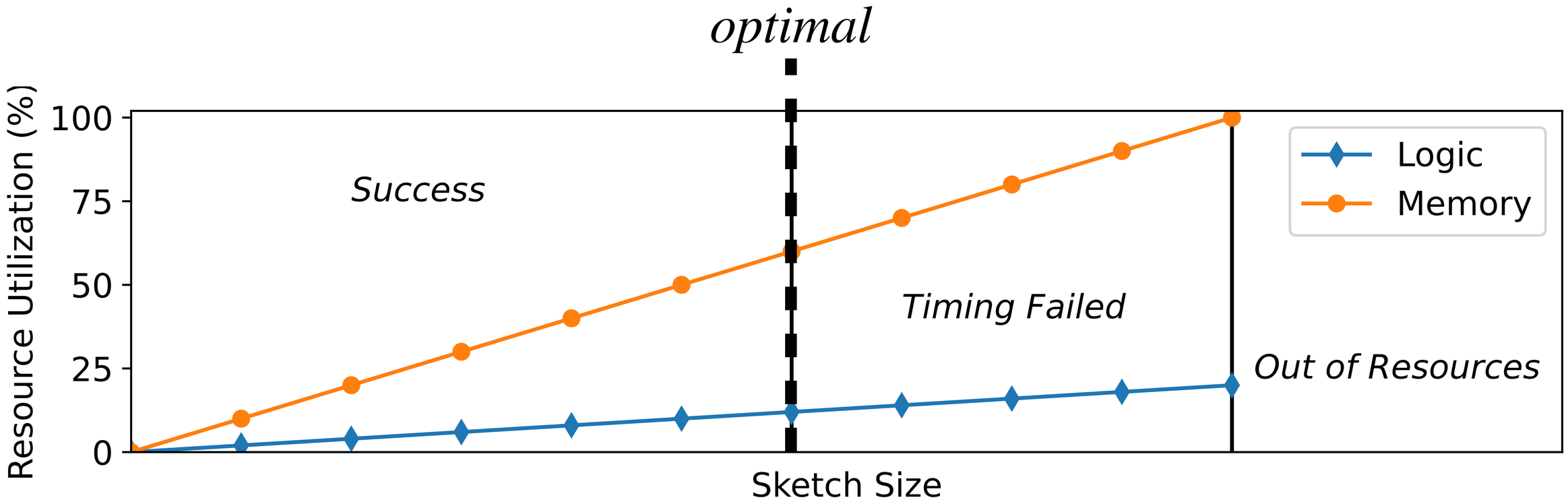
Automated Tuning

Finding the largest sketch size that works



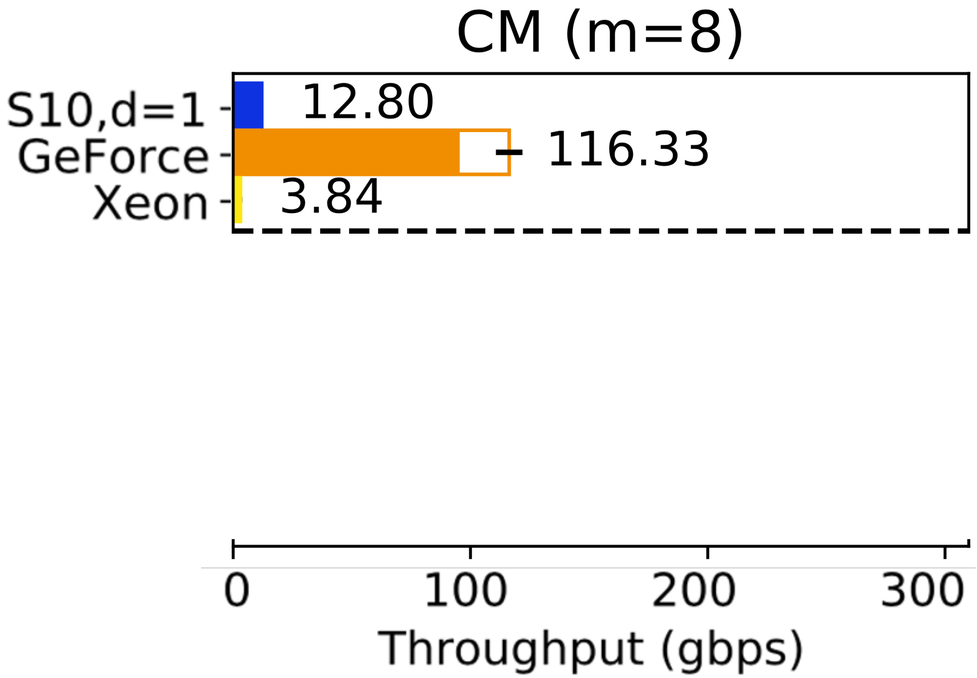
Automated Tuning

A binary search for optimal parameters



Evaluation: Throughput

Matrix sketch (m=8 rows)

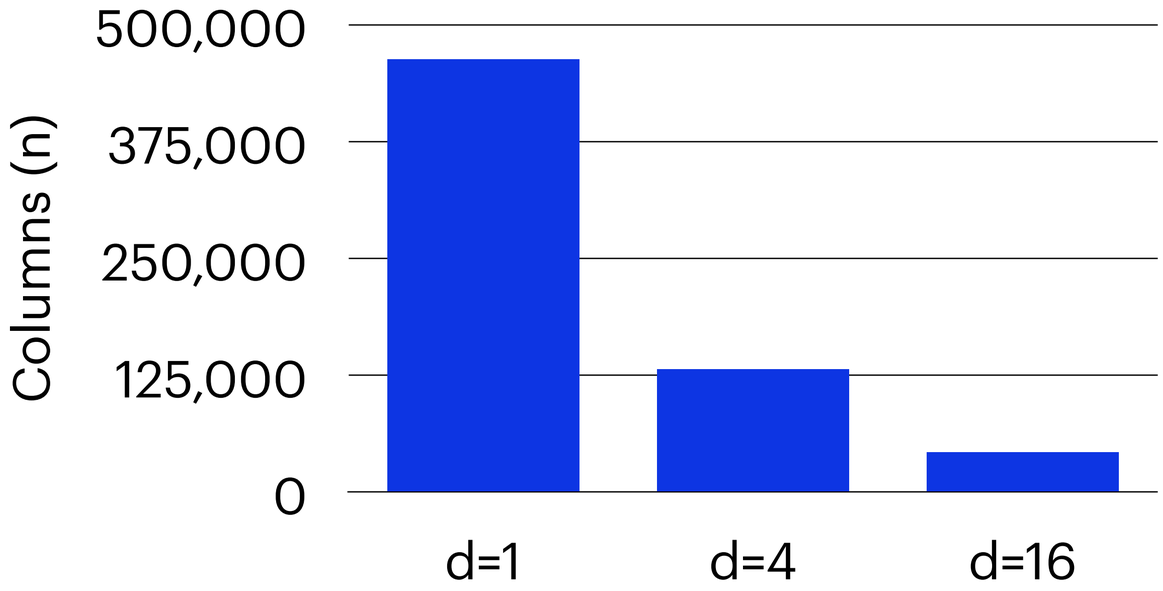


- Data parallelism is crucial for high throughput

Stratix 10 GX 2800, 400MHz | Nvidia GeForce RTX 2080 | 2x Intel Xeon Silver 4214 | 32 Bit State & Input | Uniform Data

Evaluation: Sketch Size

Matrix sketch (m=8 rows)



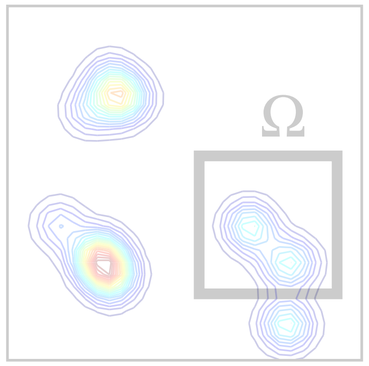
- Summary size reduces proportionate to number of inputs d

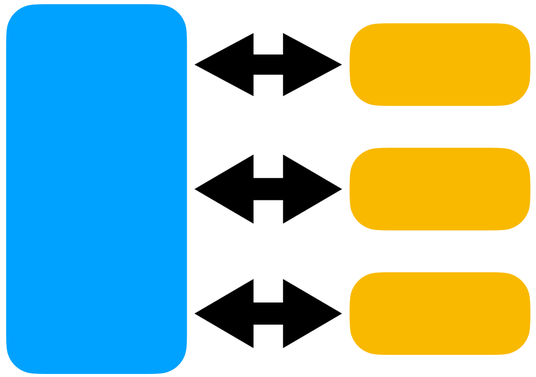
Stratix 10 GX 2800, 400MHz | Nvidia GeForce RTX 2080 | 2x Intel Xeon Silver 4214 | 32 Bit State & Input | Uniform Data

Optimistic Data Parallelism for FPGA-Accelerated Sketching

Martin Kiefer, Ilias Poulakis, Eleni Tzirita Zacharatou, Volker Markl

Proceedings of the VLDB Endowment, 16 (5), 2023

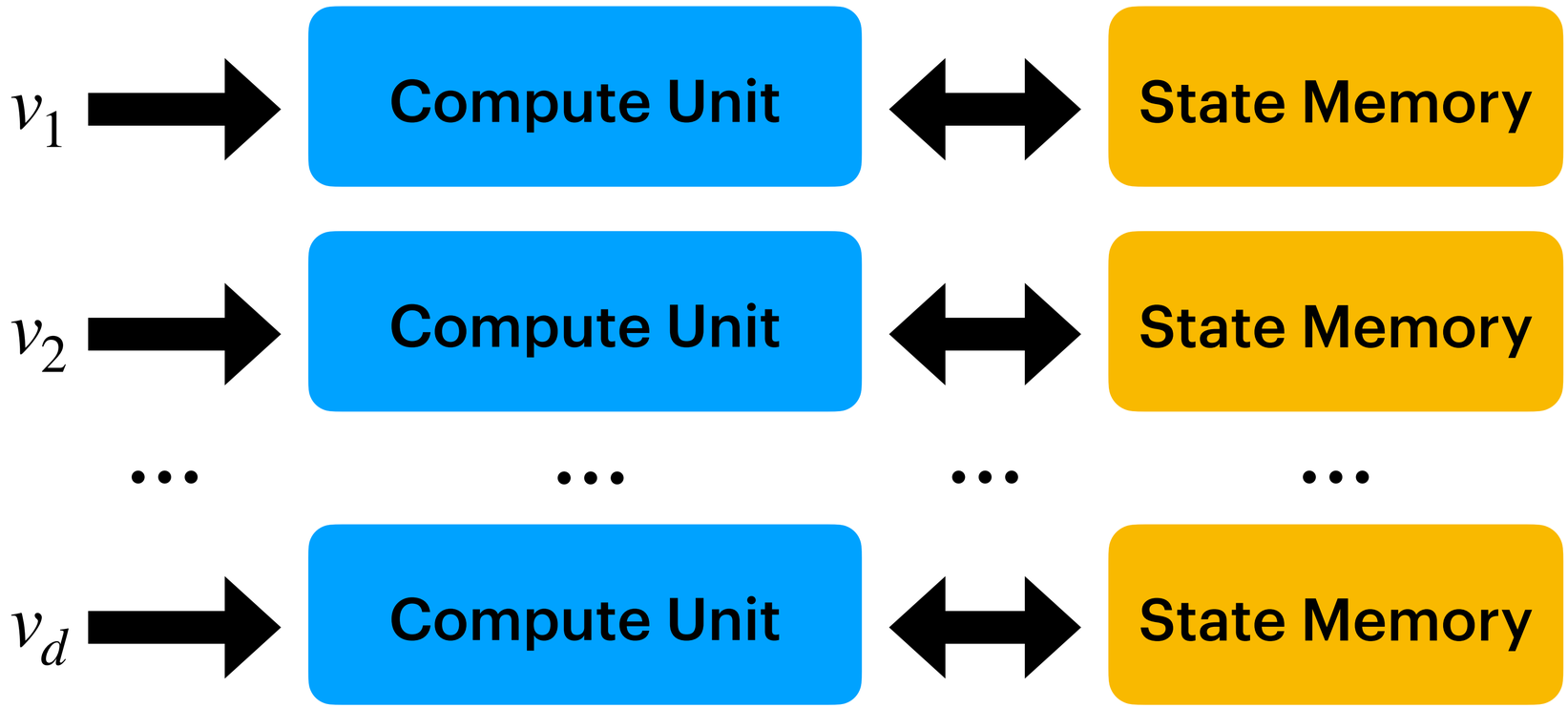




Pessimistic Data Parallelism

Guaranteed processing rates at the

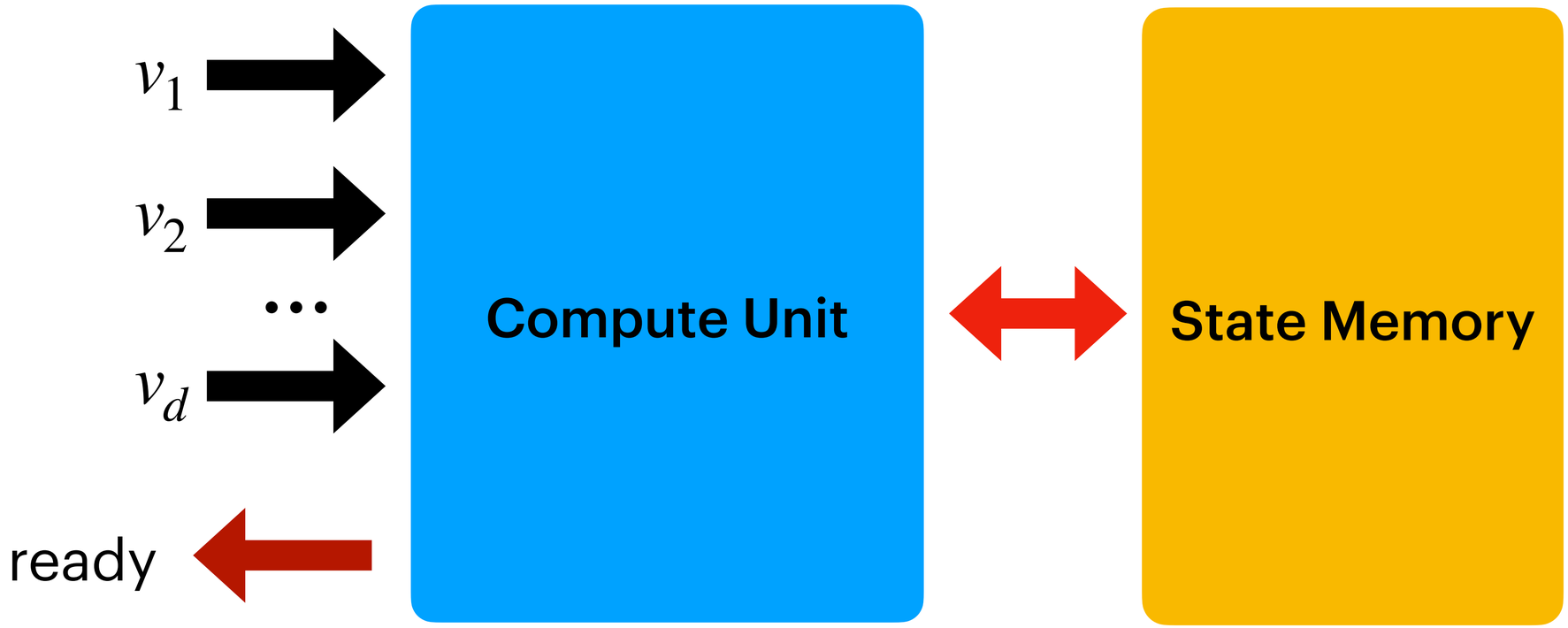
For each row:



Sharing the State Memory

Sharing the state memory across input values

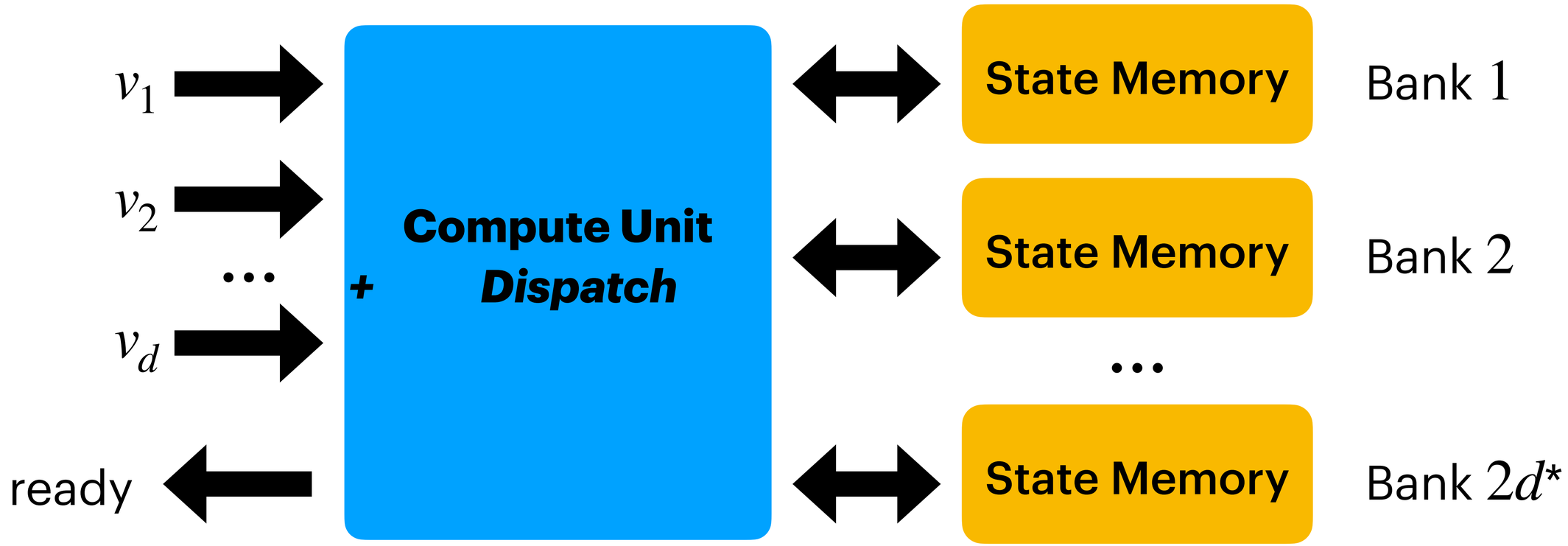
For each row:



Optimistic Architecture

Split the state memory into multiple banks

For each row:

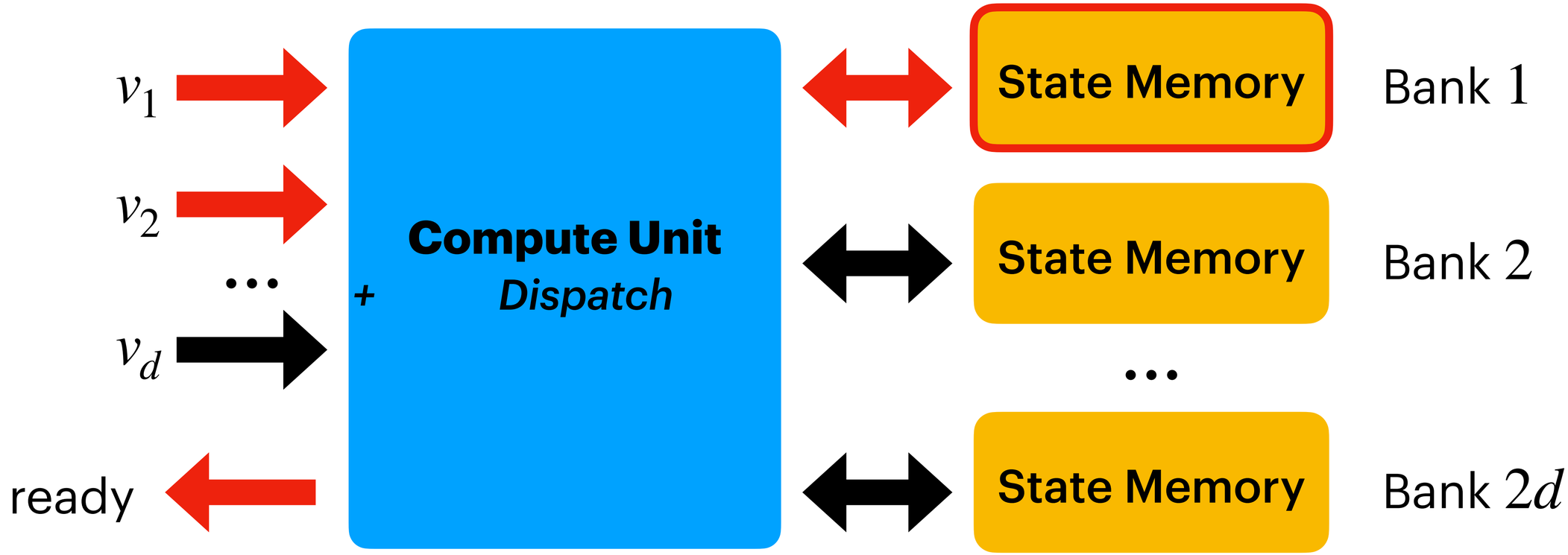


*** Oversubscribed Configuration**

Optimistic Architecture

Problem: Resource congestion + uniform data

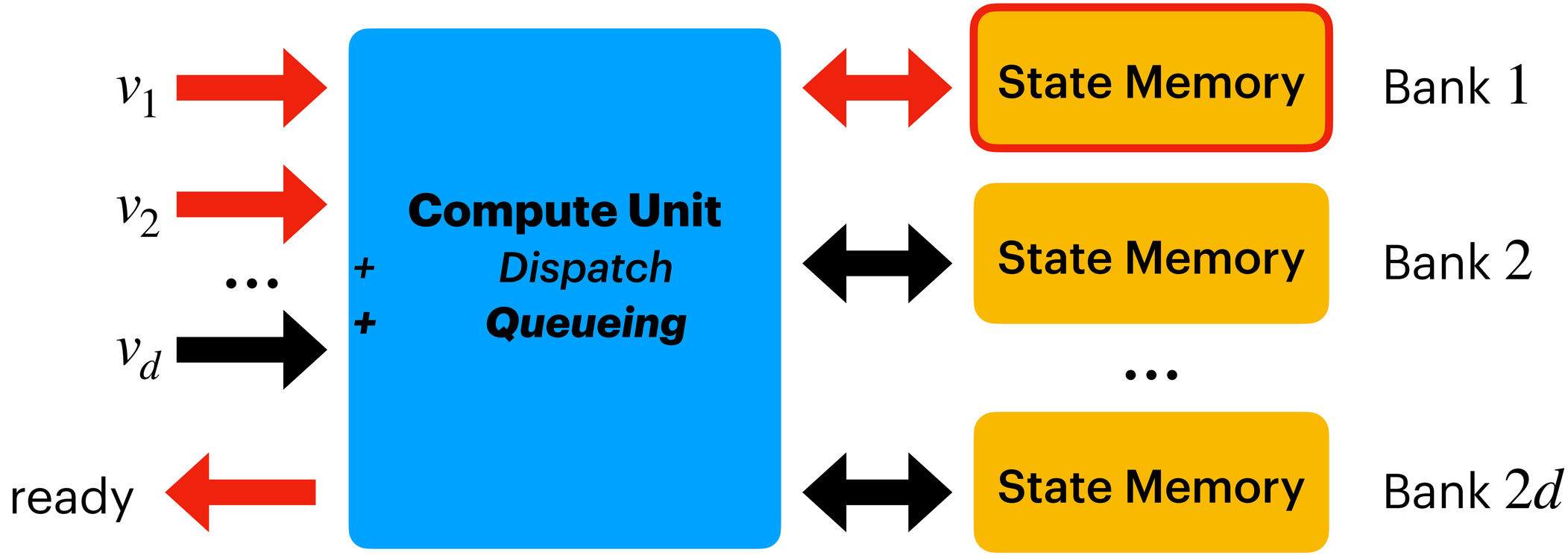
For each row:



Optimistic Architecture

Problem: Resource congestion + uniform data

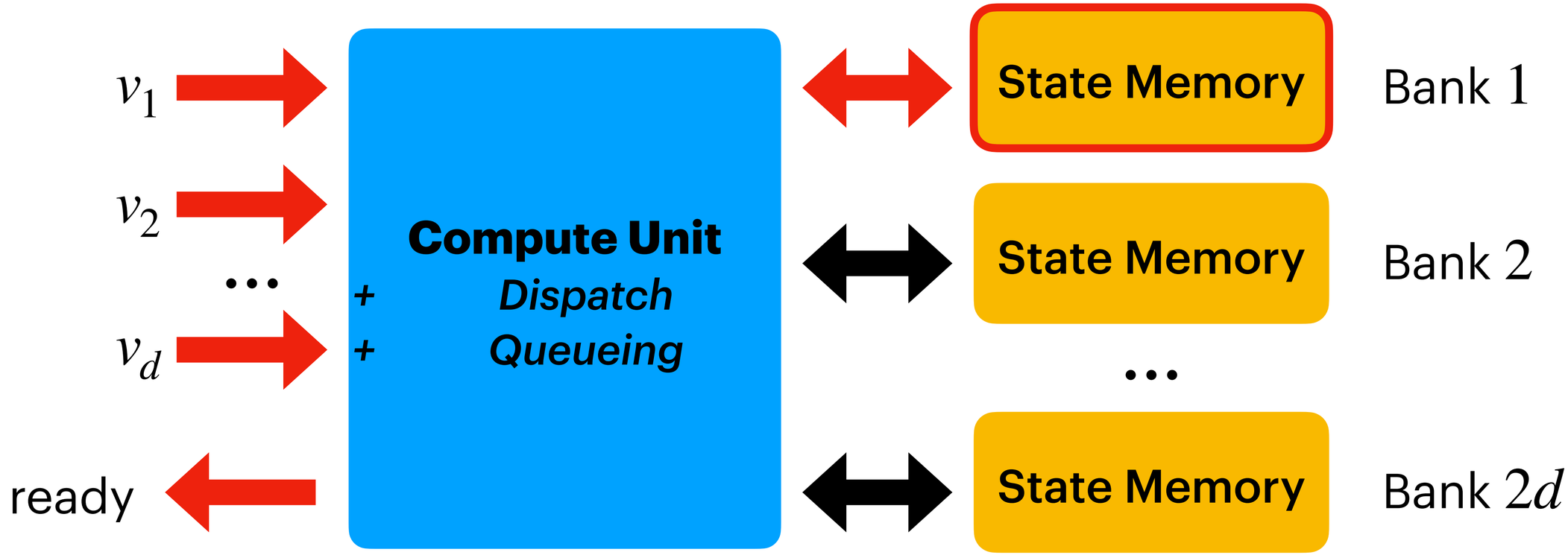
For each row:



Optimistic Architecture

Problem: Resource congestion + skewed data

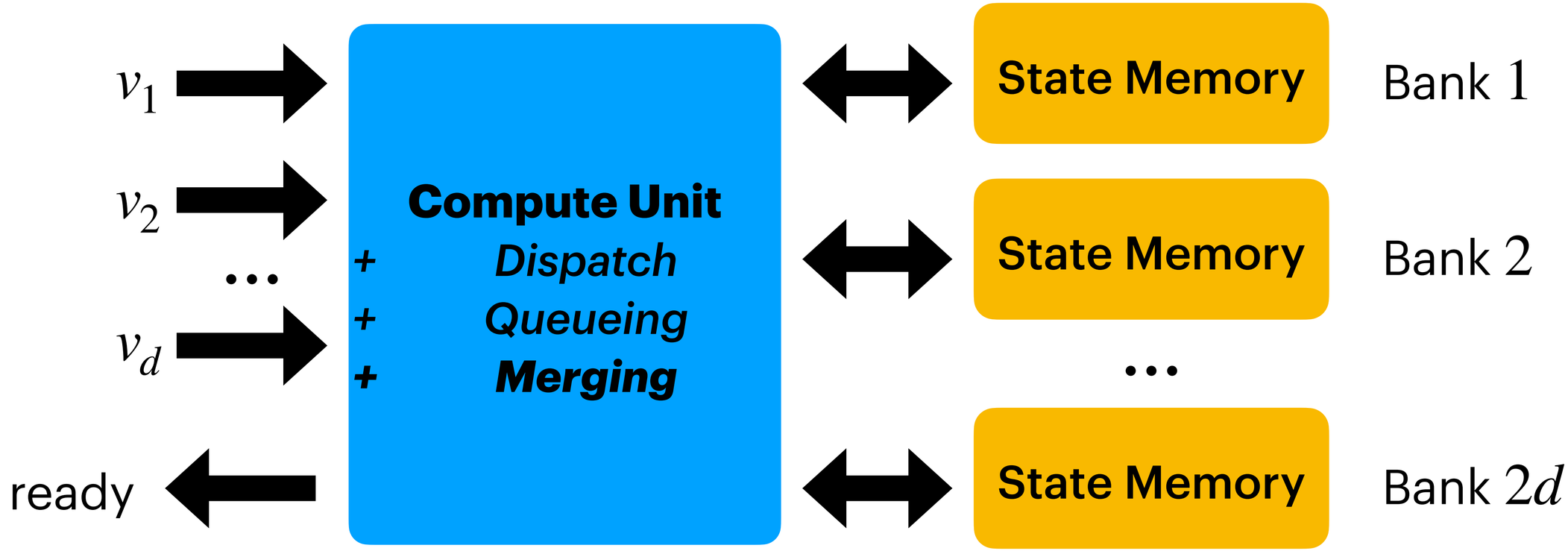
For each row:



Optimistic Architecture

Banking + Queues + Mergers

For each row:

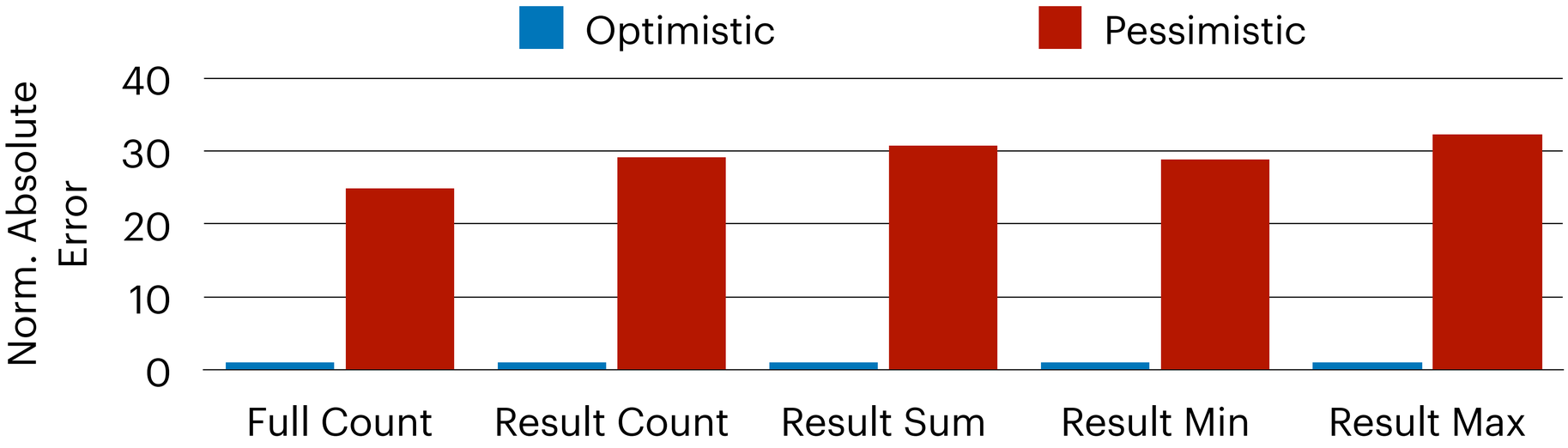


Evaluation: Sketch Size & Accuracy

Approx. Query Processing: Optimistic vs. Pessimistic

- Maximum sketch size
  - Pessimistic: $n = 2,\ m = 4096 \cdot \mathbf{5}$
  - Optimistic: $n = 2,\ m = 4096 \cdot \mathbf{32}$

$6.4x$
larger sketches



$25 - 32x$
better estimates

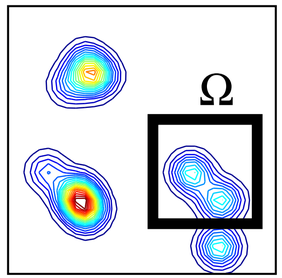
Queue Size = 64, NYT Dataset, Vertical Merging + Horizontal Merging h=2

Conclusion

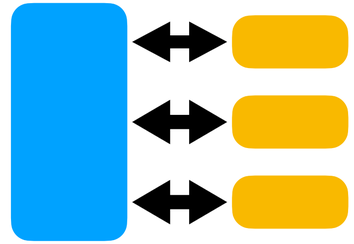
Summary

Approximate data analysis with parallel processors

- **Highly efficient approximate analyses in two use cases**
  - Selectivity estimation & sketching



- **Generalizations and abstractions**
  - Join support for KDE models
  - *M*odels & generators for sketching on FPGAs

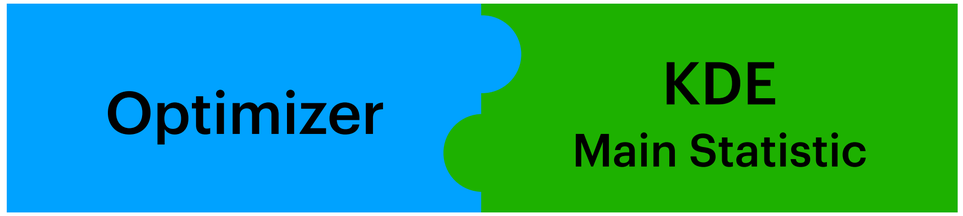


- **Hiding the complexity of parallel processors**
  - GPU hidden behind the query optimizer
  - Autotune & code generator perform expert tasks

Conclusion

The bigger picture: Applicability

- **KDE for selectivity estimation**
  - Towards a general framework
  - Sampling + histograms + ML
  - Statistical coprocessing



- **FPGA-accelerated sketching**
  - Accessible to systems engineers
  - Statistics collection in the data path