HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Workload-Driven Horizontal Partitioning and Pruning for Large HTAP Systems

Paris, 16 April 2018

Martin Boissier & Daniel Kurzynski

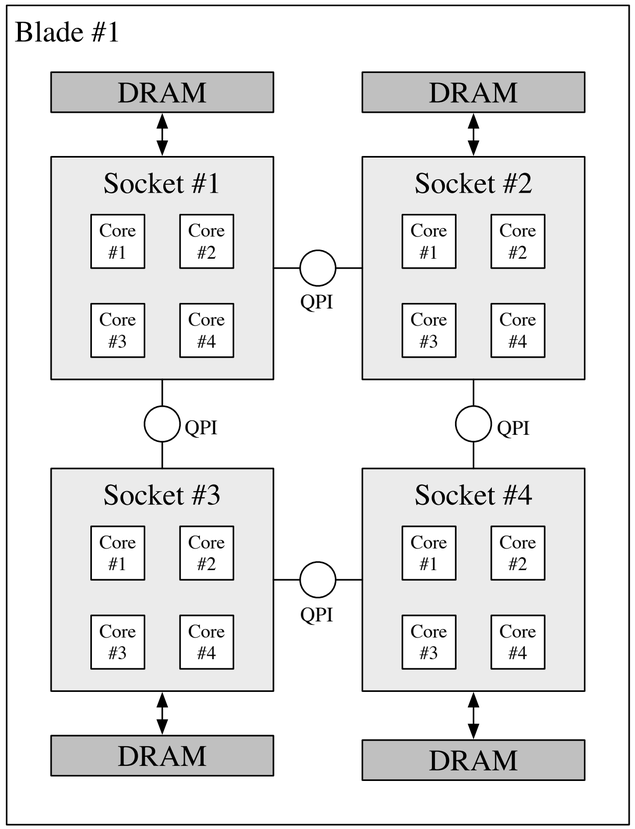
Project Goal

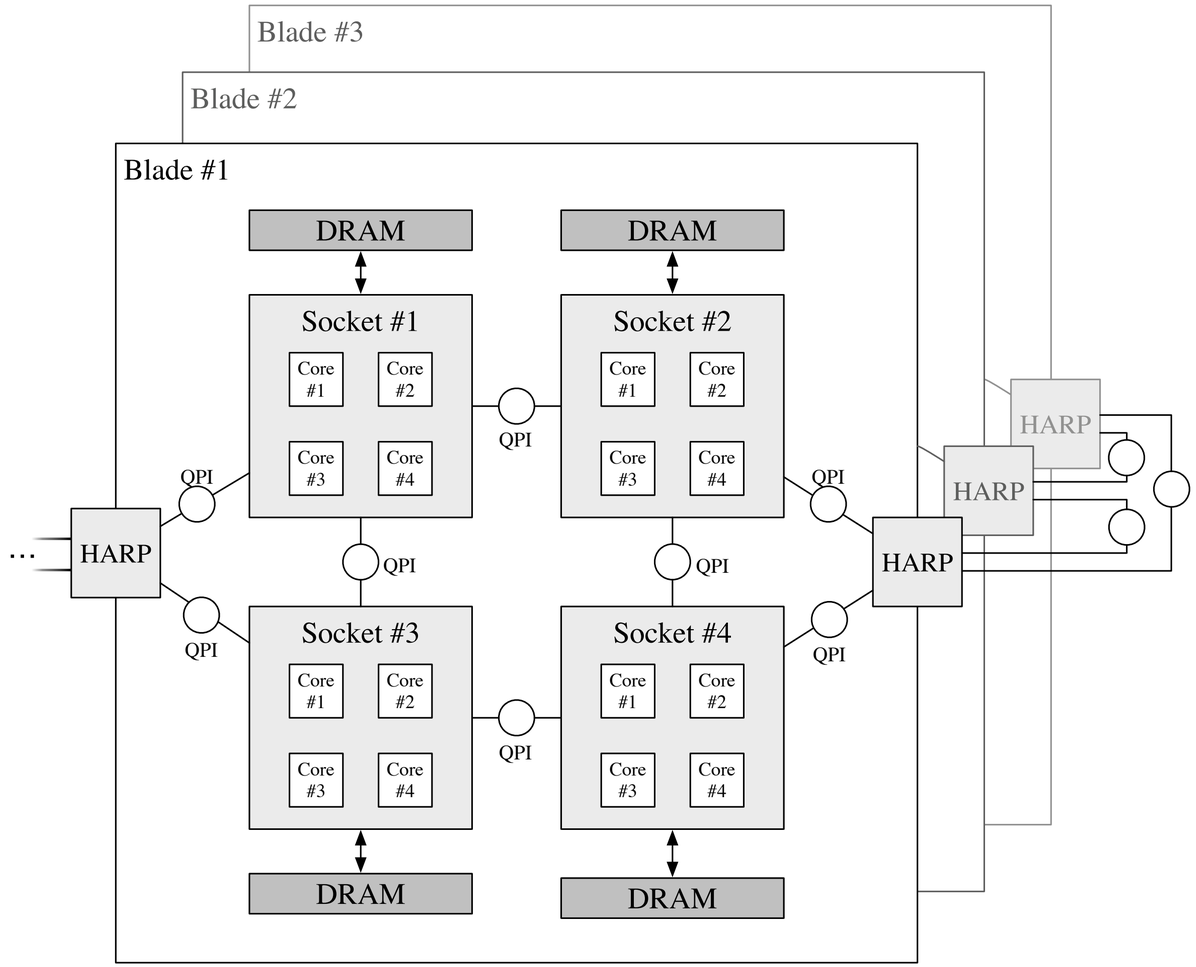
- **Background:**
  - In our research, we focus HTAP-optimized main memory-resident databases for modern enterprise systems
  - Own research database called Hyrise*, an open source columnar HTAP database
- **Current Situation:**
  - NUMA architectures pose new challenges for data distribution, scheduling, and query execution
  - Optimizing for NUMA has large impact (cf. [1, 2])
- **Our Goal:**
  - Evaluate various partitioning approaches to distribute a table's data to *n* NUMA nodes with the goal to *maximize tuples skipped* (i.e., data skipping)

[1] Leis et al., Morsel-Driven Parallelism: A NUMA-Aware Query Evaluation Framework for the Many-Core Age. SIGMOD '14
[2] Pandis et al., Data-Oriented Transaction Execution. VLDB '10

* Hyrise on GitHub: https://github.com/hyrise/hyrise

Modern NUMA Systems





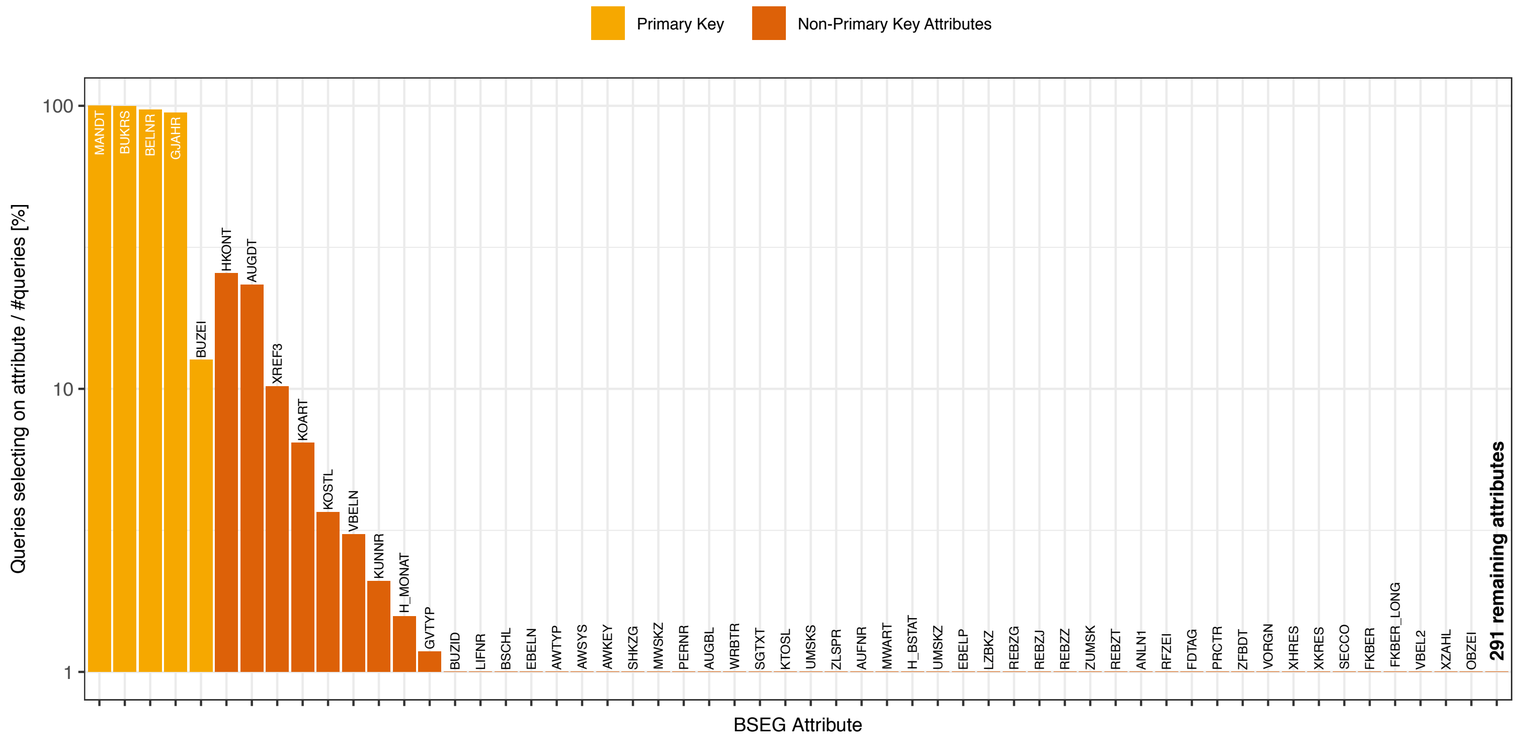
Modern NUMA Systems

- For optimal performance on NUMA systems:
  - data shall be equally distributed
  - processing shall be data-local *
- We see **two problems** with this statement:
  - Equal distribution works fine for both TPC-C and TPC-H
    it does not for real-world systems
  - Partitioning elimination/pruning is considered an orthogonal topic
    it should not be
- This projects evaluates means to “combine both worlds”

* Cf. “NUMA commandments”, Albutui et al., Massively Parallel Sort-Merge Joins in Main Memory Multi-Core Database Systems. PVLDB 2012

Workload-Driven Partitioning

- We evaluated several partitioning approaches
  - almost all of them to be *too simple*
  - *Aggressive Data Skipping* by Sun et al. is one exception [3]



[3] L. Sun et al., Fine-grained partitioning for aggressive data skipping. SIGMOD 2014

Aggressive Data Skipping

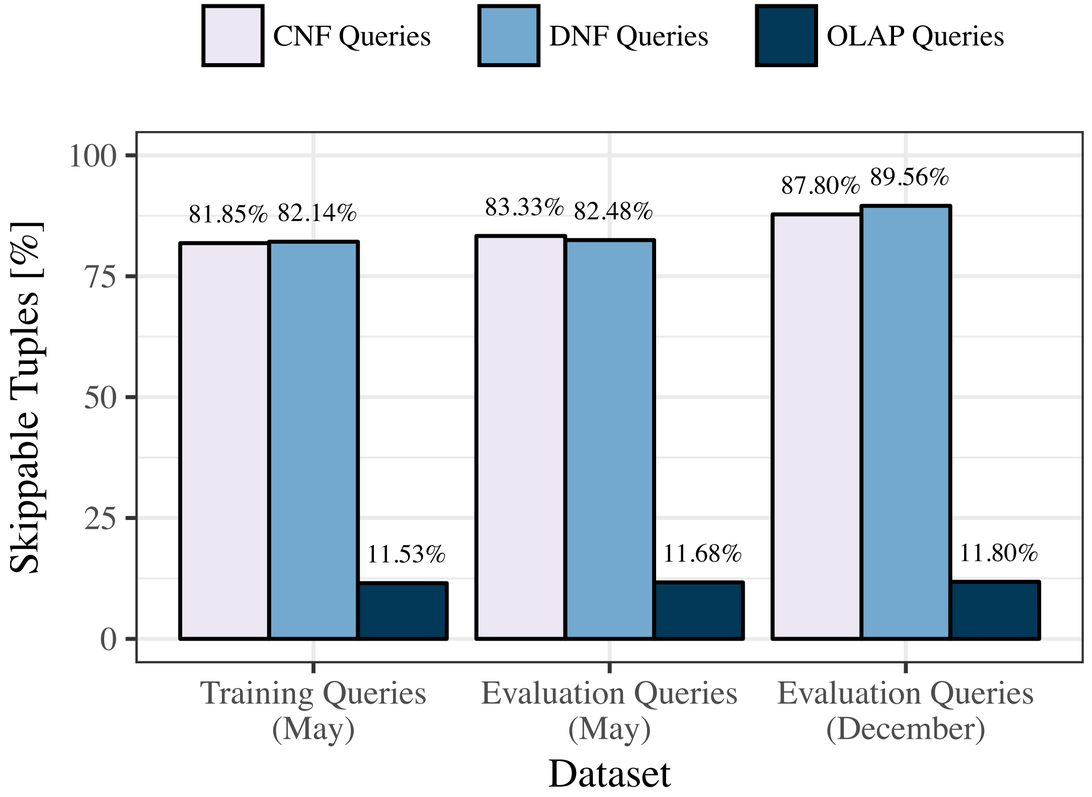
- Approach initially motivated for large-scale systems like Spark
- We misused the approach to create partitioning schemes
  - Configurable by the number of partitions to yield
  - We limit partition count to number of NUMA nodes
- The process
  - Parse workload and extract relevant selections + frequent item set mining
  - Scan data for distribution of features
  - Merge features to create partitions

Created Partitioning Scheme

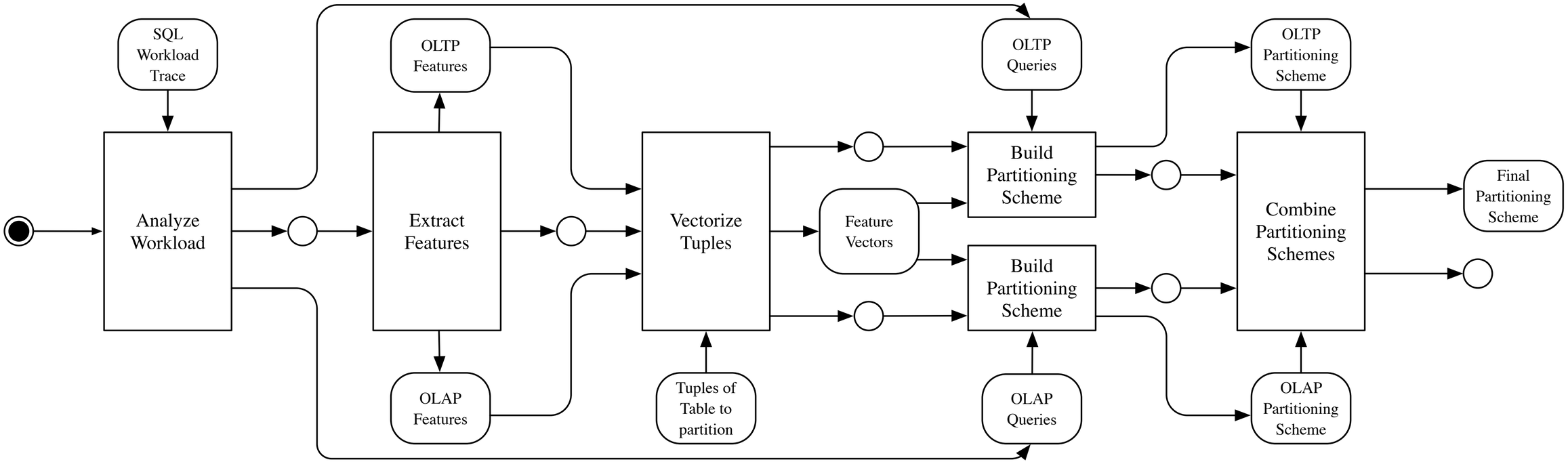
- clustered and non-trivial
  - Feature 1:
    ```
    mandt = 2 & koart <> ‘k’ & koart = ’d’
    ```
  - Feature 2:
    ```
    mandt = 2 & koart = ‘k’ & bukrs = ‘9999’
    ```
  - …
- Created partitions are freely defined by 15 features

Aggressive Data Skipping

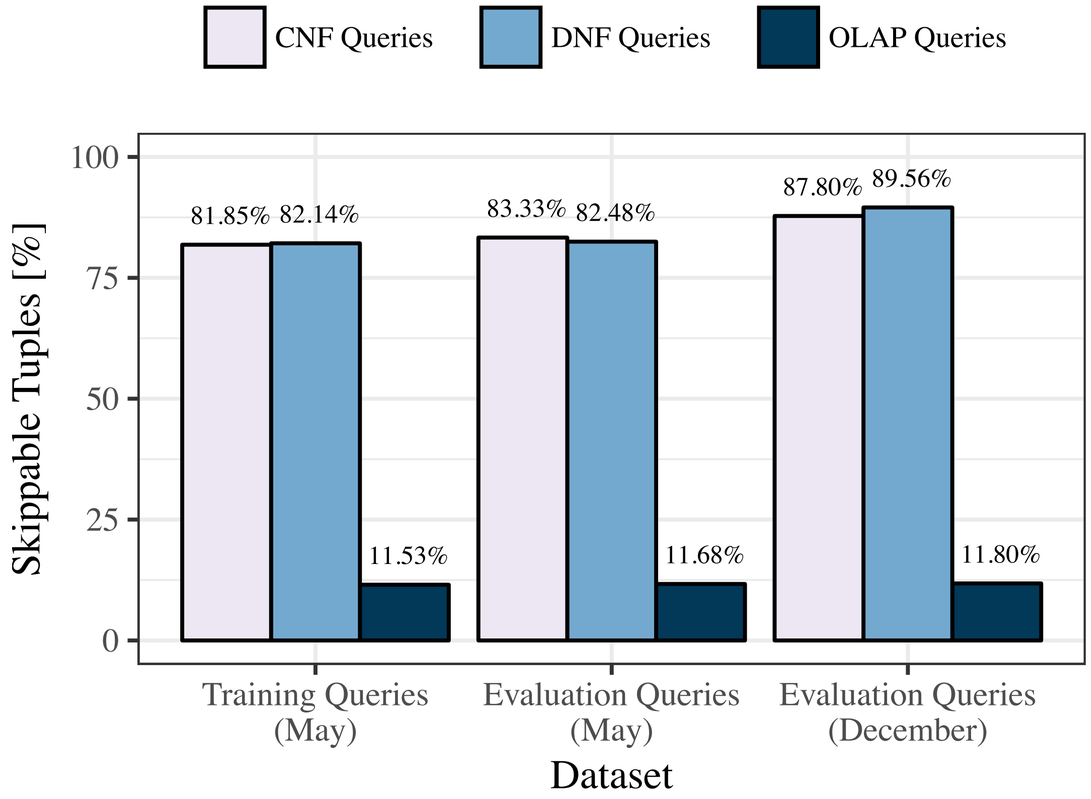
- **Problem:**
  - Objective is the number of pruned tuples
  - Heavily favors selective and frequent Queries
    - i.e., OLTP queries
- Many ways to adapt for multiple workload classes
  - Weighting by runtime, what-if based query costs, …
- **Idea:** execute partitioning twice and merge

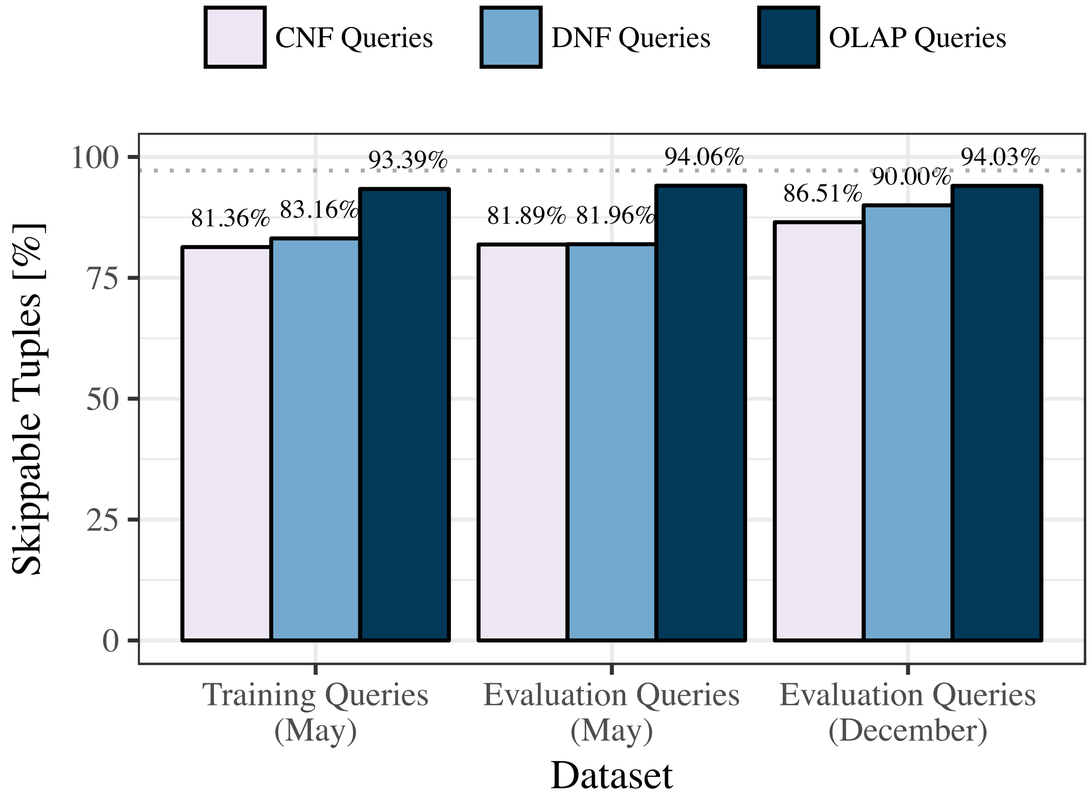


Merging Partition Schemes

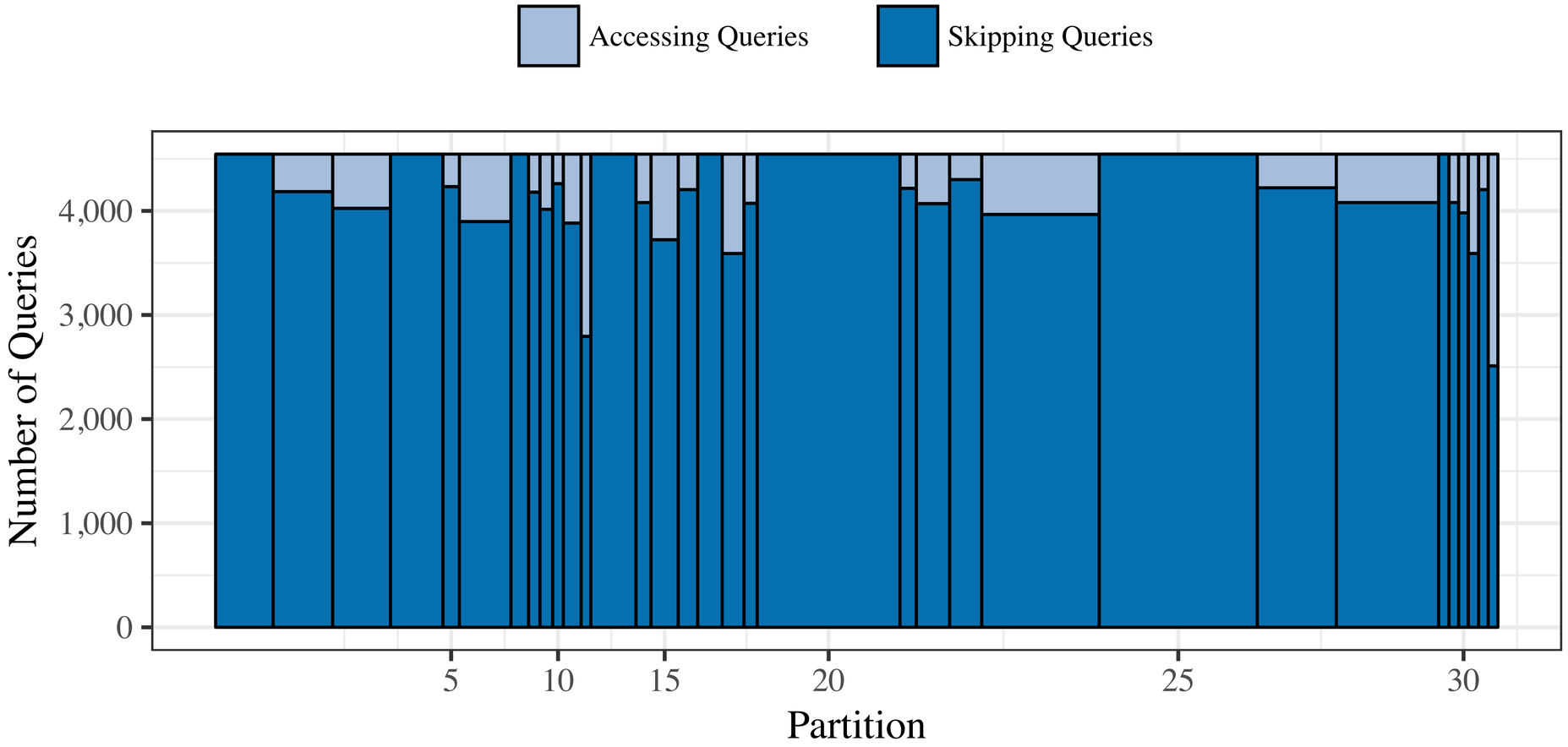


Results (i)


CNF Queries
DNF Queries
OLAP Queries
Skippable Tuples [%]
100
75
50
25
0
81.85%
82.14%
11.53%
83.33%
82.48%
11.68%
87.80%
89.56%
11.80%
Training Queries (May)
Evaluation Queries (May)
Evaluation Queries (December)
Dataset


CNF Queries
DNF Queries
OLAP Queries
Skippable Tuples [%]
100
75
50
25
0
81.36%
83.16%
93.39%
81.89%
81.96%
94.06%
86.51%
90.00%
94.03%
Training Queries (May)
Evaluation Queries (May)
Evaluation Queries (December)
Dataset

Results (ii)


Accessing Queries
Skipping Queries
Number of Queries
4,000
3,000
2,000
1,000
0
5
10
15
20
25
30
Partition

My Personal Outlook

- **Self-driving trend & distributed systems** force us to put more emphasis on (re-)partitioning

- **What is missing?**
  - More work on skew-aware and pruning-optimized partitioning
  - **Proper cost models** for adept partitioning schemes
  - Personally, **I doubt AI & DL will solve these problems** for us
  - More emphasis on repartitioning (not only for NUMA systems! [4])

[4] Binnig et al., The End of Slow Networks: It's Time for a Redesign. PVLDB 2016