Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

An In-Depth Analysis of Data Aggregation Cost Factors in a Columnar In-Memory Database

Stephan Müller, Hasso Plattner

Enterprise Platform and Integration Concepts

Hasso Plattner Institute, Potsdam (Germany)

DOLAP 2012 – November 2nd

Agenda

- Data aggregation
- Aggregations in Hyrise
- Cost factors and benchmarks
- Summary and future work

Motivation

- Aggregation
  - Abstractions for conceptualizing the real world
  - Resource-intensive operation in OLAP workloads
- Columnar in-memory databases
  - Optimized for set processing
  - Enables reunification of OLTP and OLAP [1]

[1] H. Plattner. A common database approach for OLTP and OLAP using an in-memory column database. In SIGMOD, 2009.

Data Aggregation

- SELECT product, SUM (amount) AS revenue
  FROM sales WHERE year=2011
  GROUP BY product

$$\gamma_{G_1,\ldots,G_n,F_1(A_1)\rightarrow N_1,\ldots,F_m(A_m)\rightarrow N_m}(E)$$

- Two phases
  - Grouping (hash-based, sorting, nested loops)

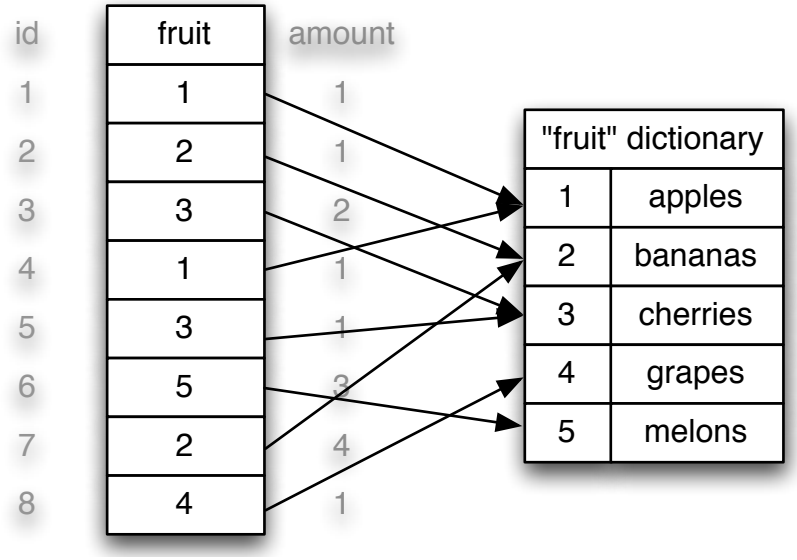
    $$B_{(G_1,\ldots,G_n)} = \{A_t | G_{t1} = G_1 \wedge \ldots \wedge G_{tn} = G_n\}$$
  - Calculation

    $$R = \{(x, F_1(A_1), \ldots, F_n(A_n)) | (x, BL_x) \in H\}$$

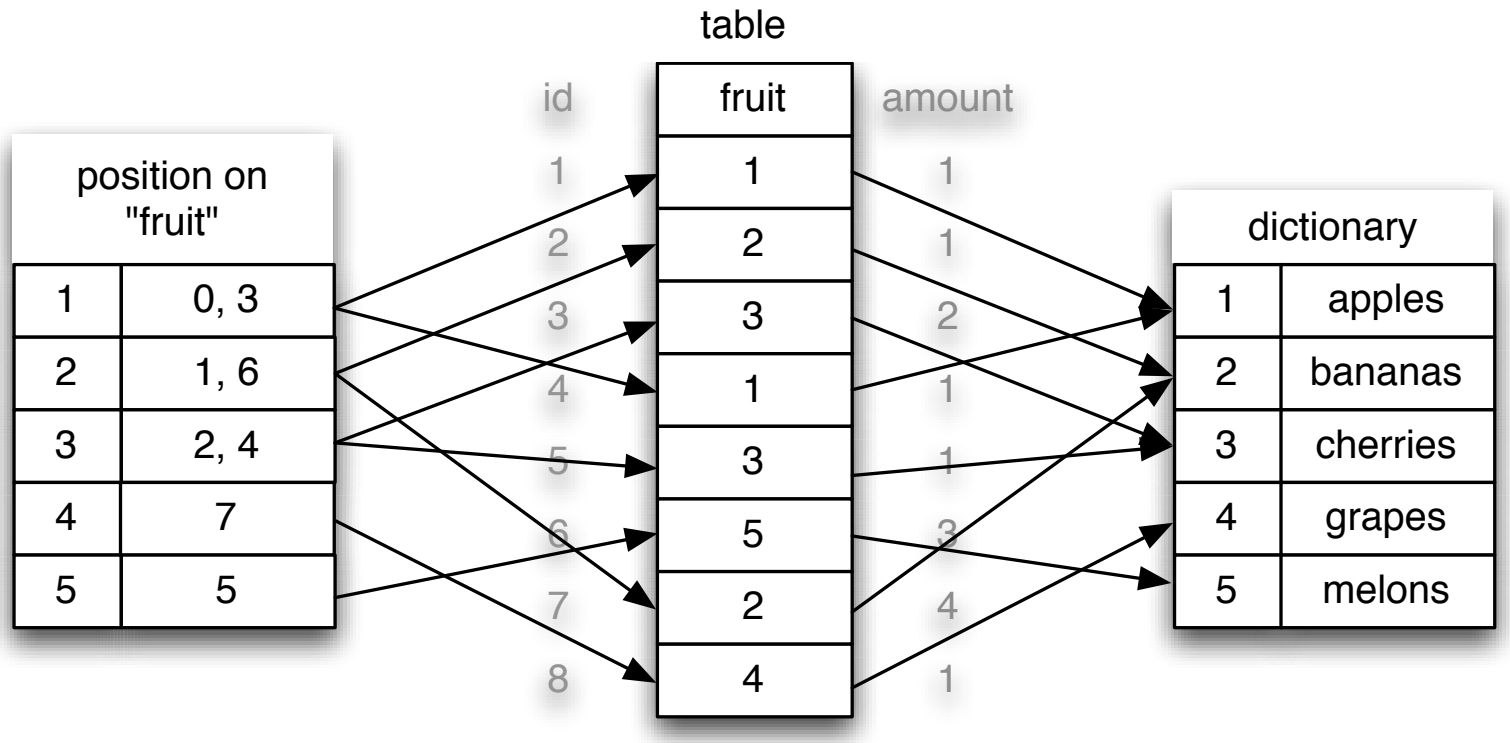
Data Aggregation in Hyrise

- Columnar Storage
- Dictionary compression

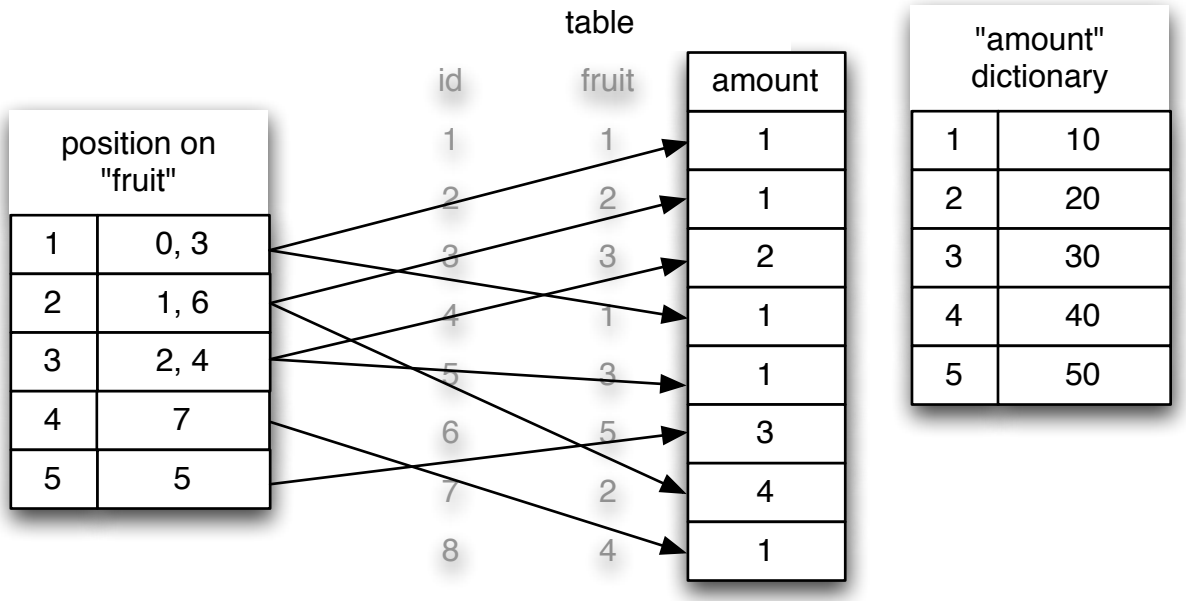
id	fruit	amount
1	apples	10
2	bananas	10
3	cherries	20
4	apples	10
5	cherries	30
6	melons	50
7	bananas	20
8	grapes	40



Data Aggregation – Grouping



Data Aggregation - Calculation



fruit	sum(amount)
apples	20
bananas	50
cherries	30
grapes	10
melons	40

Cost Modeling in IMDBs

- Cost model based on cache misses [2]
  - Basic patterns for each cache level
  - Combine patterns to model complex operations

$$T_{Mem} = \sum_{i=1}^{N} (M_i^s * l_{i+1}^s + M_i^r * l_{i+1}^r)$$

[2] Manegold, S. et al. 2002. Generic database cost models for hierarchical memory systems. *VLDB*. (2002).

Cost Modeling of the Aggregation

$$GroupBy(V, W) := HashBuild(V, HT) \oplus Aggregate(V, HT, W)$$

$$HashBuild(V, HT) := s\_tra^s(V_G) \odot r\_tra(HT)$$

$$Aggregate(V, HT, W) := s\_tra^s(HT) \odot r\_acc(V.length, T_F) \oplus s\_tra^s(W)$$

Cost Factors

- Data-dependent
  - Dataset size
  - Distinct grouping values
  - Distinct grouping values distribution
  - Sorting of grouping values
- Data-independent
  - Aggregation function
  - Hash implementation

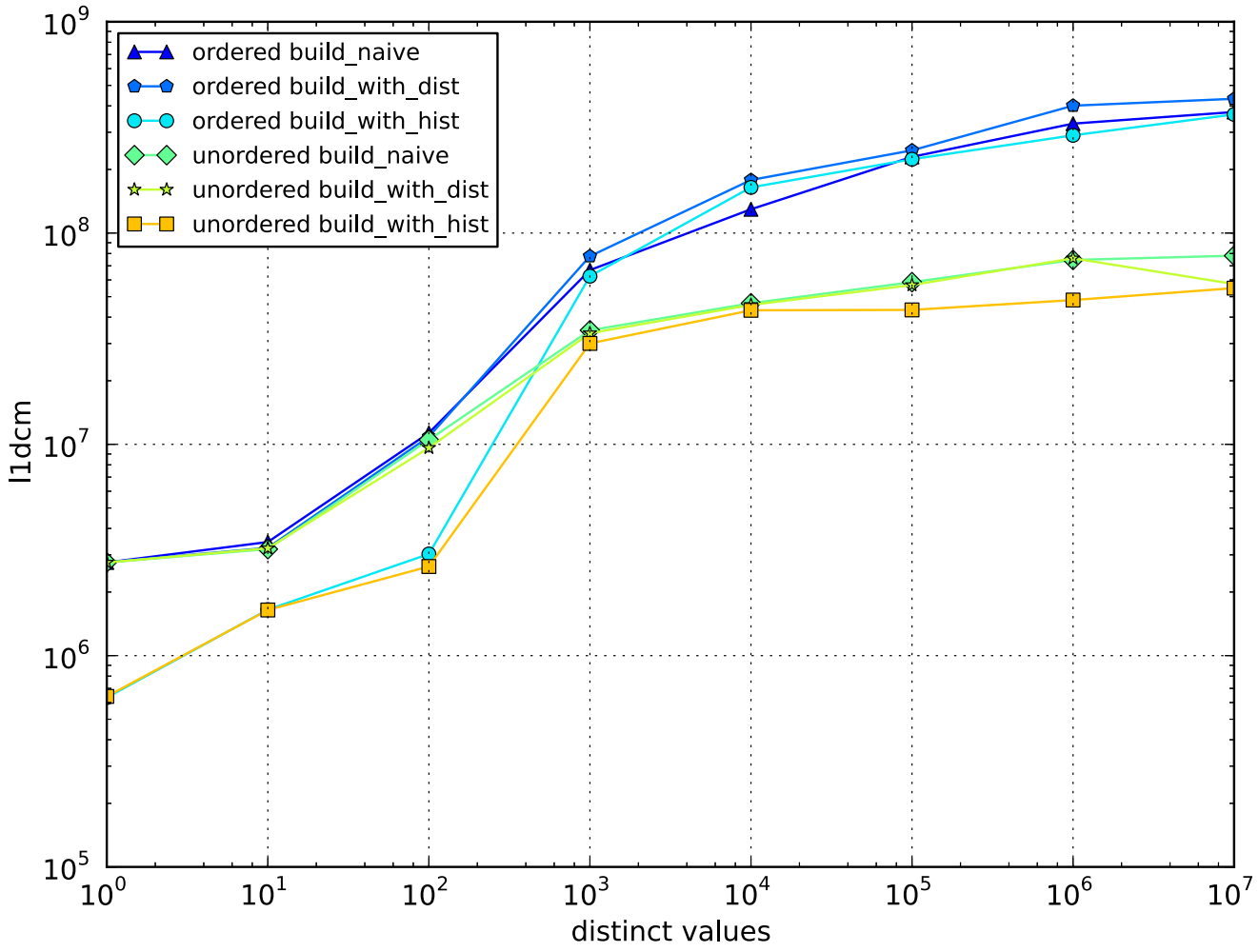
Hash Implementations

- Ordering
  - Sorted tree (std::map)
  - Unsorted hash table (std::unordered_map)
- Pre-allocation strategies
  - Naïve implementation (automatic growth)
  - Distinct value setup (scaffold of hash map)
  - Histogram-based setup (scaffold and position list)

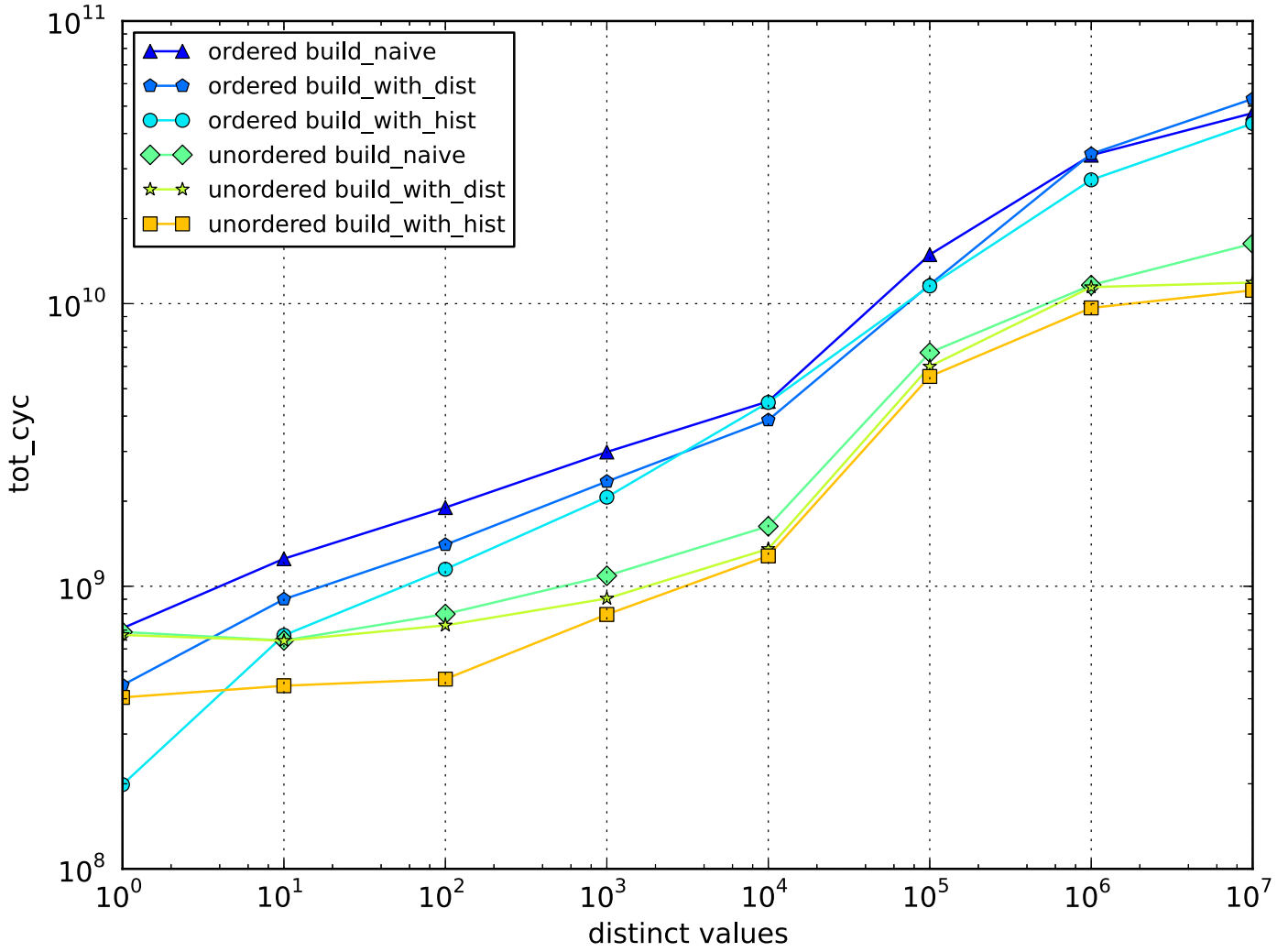
Data Set Size

- Number of bytes processed during query
  - Input table data
  - Intermediate result data
  - Final result data
- Influencing factors
  - Relation size
  - Relational expression
  - Grouping attributes
  - Aggregation attributes
  - Aggregation functions

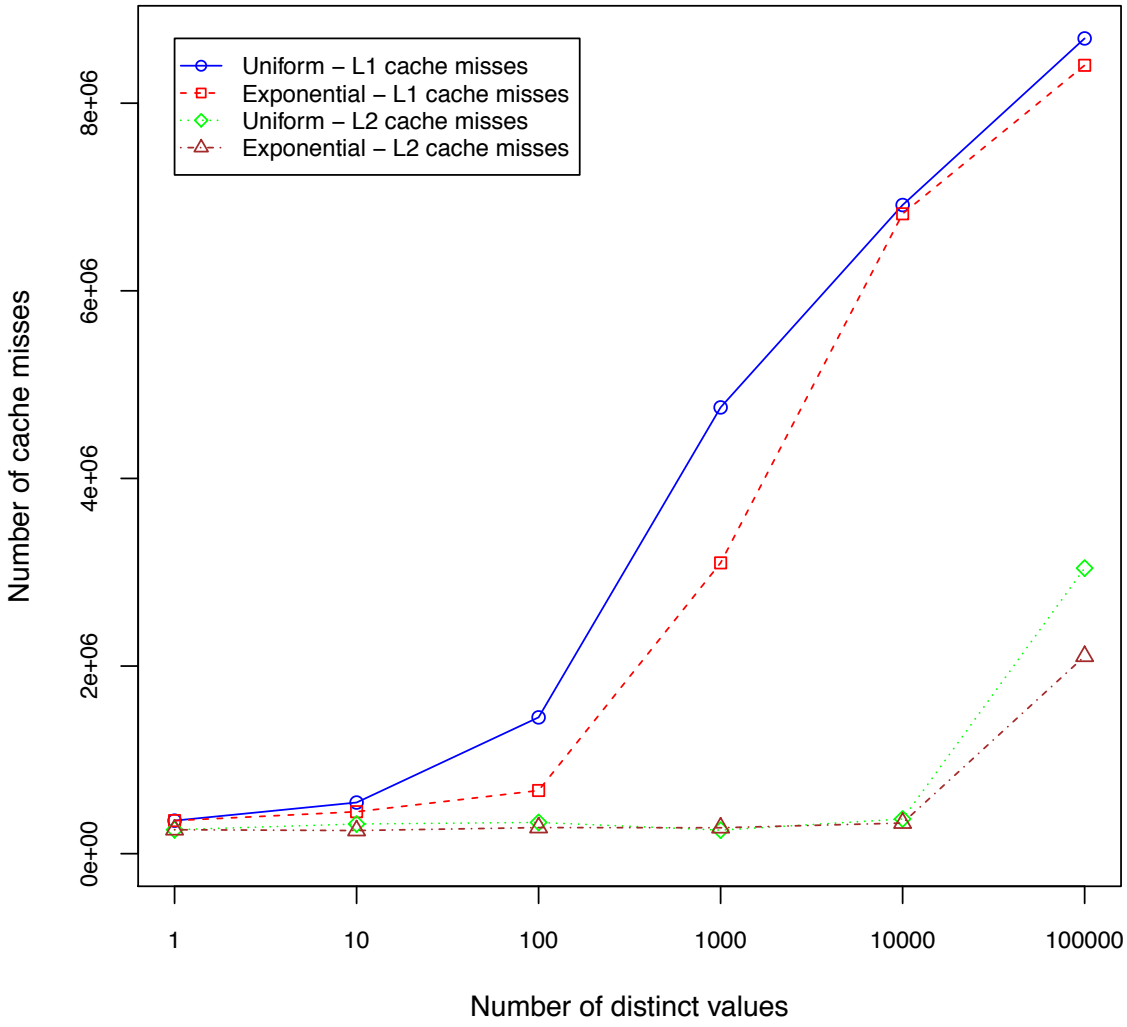
Distinct Values – L1 Cache Misses



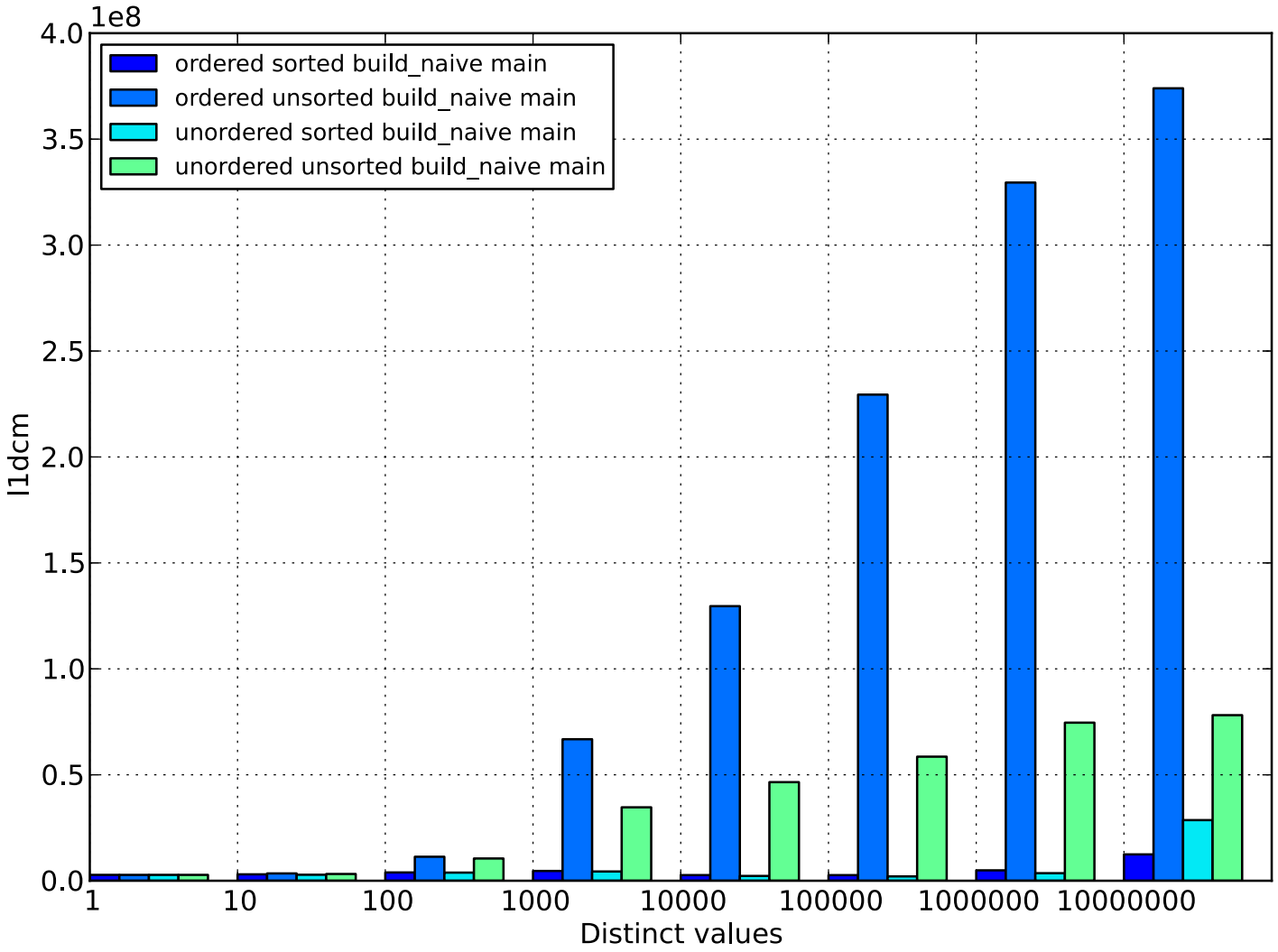
Distinct Values – CPU Cycles



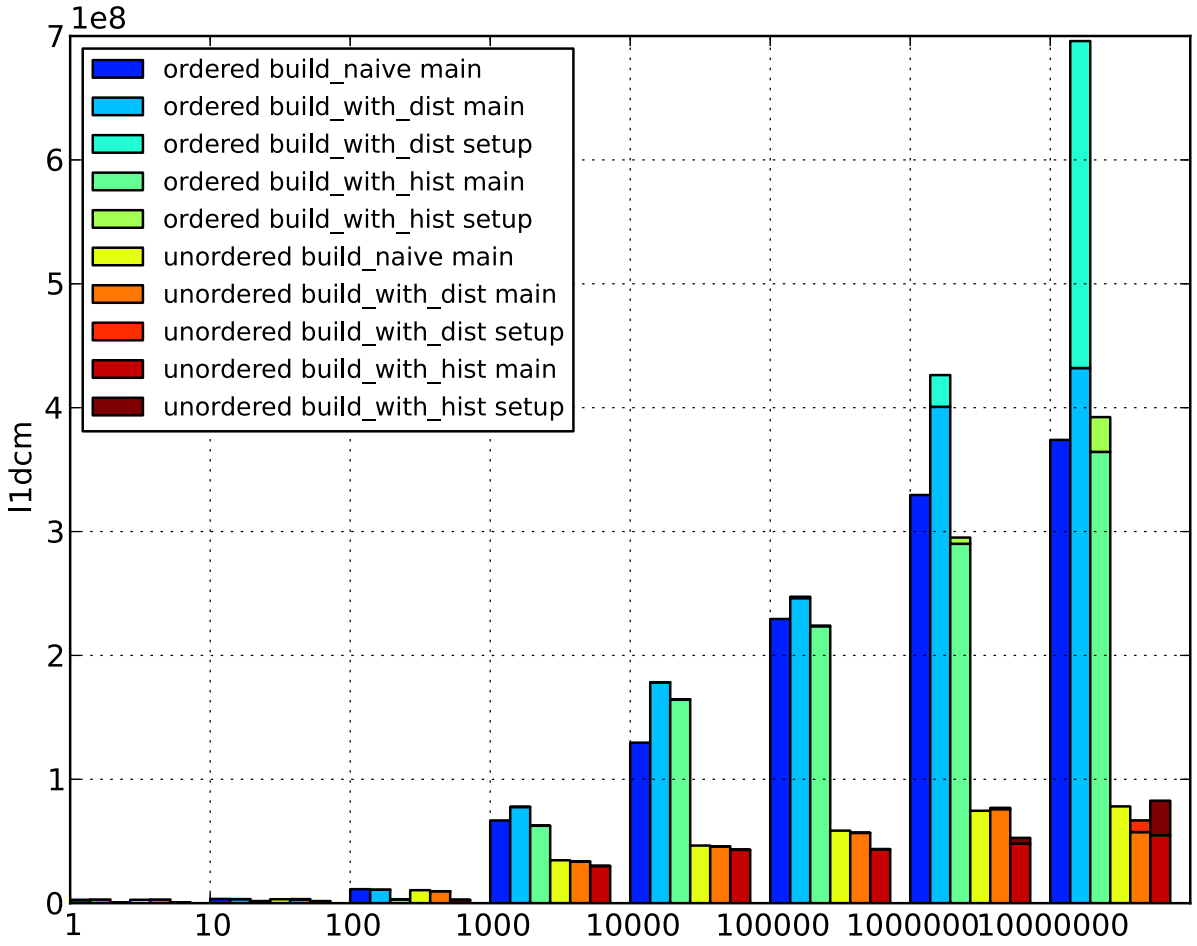
Distinct Values Distribution



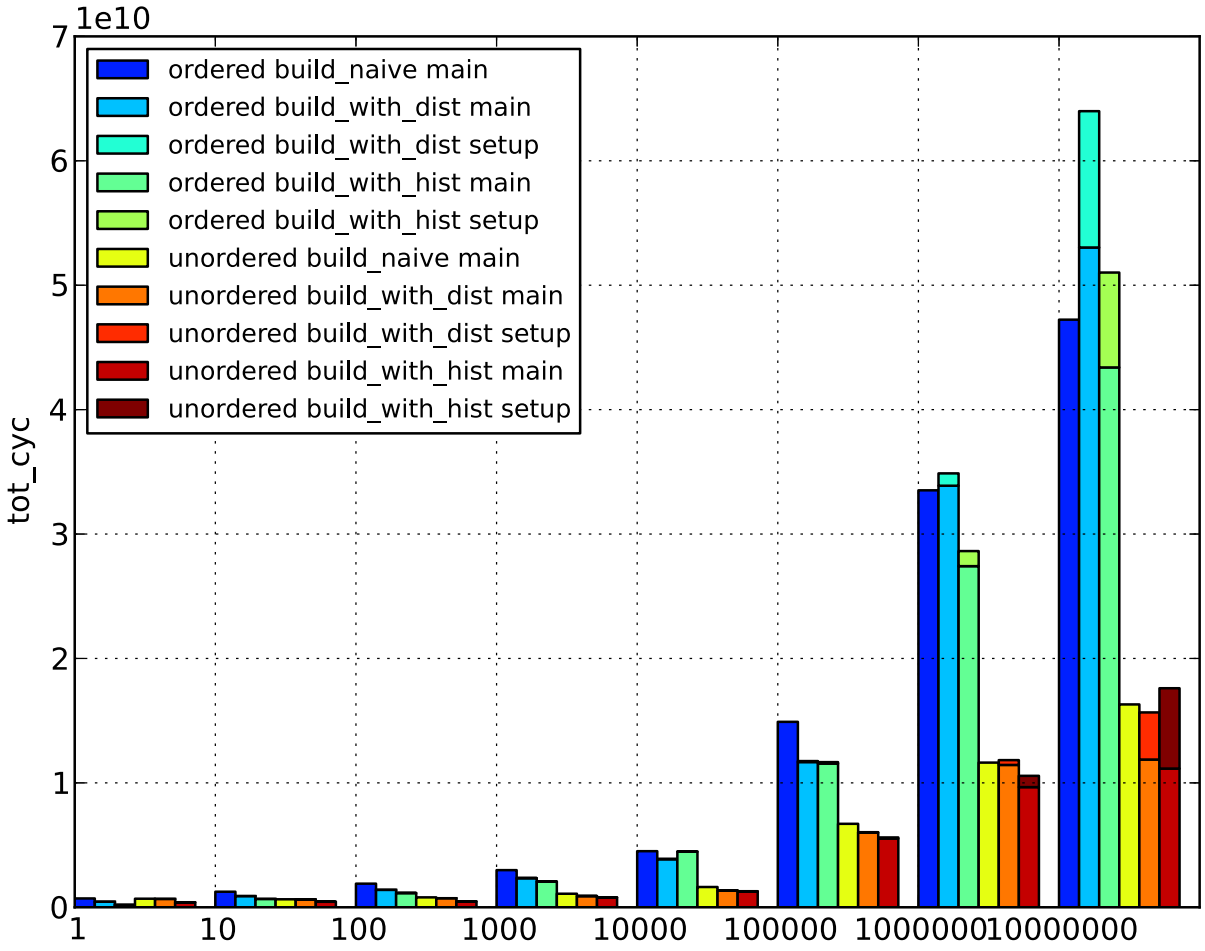
Impact of Sorted Attributes – L1 Misses



Hash Implementations – L1 Misses



Hash Implementations – CPU Cycles



Summary

- High impact of aggregation operation in a mixed OLTP and OLAP workload
- Data characteristics influence aggregation performance
- Hash implementation choice matters
- Pre-allocation can reduce cache misses
- Future work
  - Multithreaded aggregations
  - Extension of existing cost models

Thank you!

stephan.mueller@hpi.uni-potsdam.de

http://epic.hpi.uni-potsdam.de