Workload-Aware Column Imprints

Noah Slavitch

University of Waterloo, nmrslavi@uwaterloo.ca

1.Introduction

- In-memory columnar databases often use indexes to accelerate high-selectivity range queries
- Many indexes, such as the inverted index have a high storage cost, greatly increasing the Total Cost of Ownership (TCO)
- Column imprints [1] are a cache-sensitive, space-efficient alternative to the inverted index
- **This research seeks to improve column imprints using workload analysis, a skip bit, and vertical layout**

2. Prior Art: Column Imprints [1]

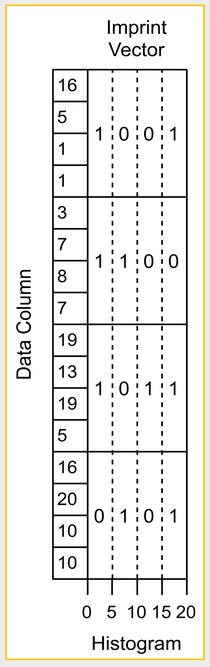

Figure 1: Column imprints example

- Indexes a data column by using a histogram to assign a 64-bit bit vector (named an imprint) to each cache line worth of data
- Column imprints are further compressed with an 'imprint dictionary' (seen in Fig. 2)
- Imprints accelerate high-selectivity queries, especially on data with local structure

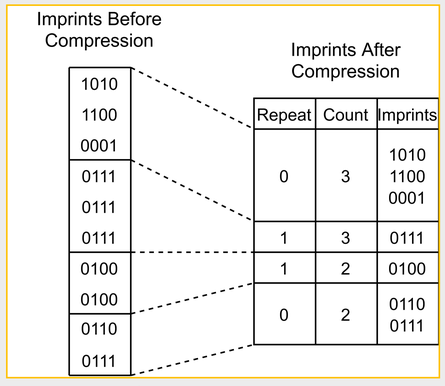

Figure 2: Column imprints compression

Contributions

3. Workload-Aware Histogram

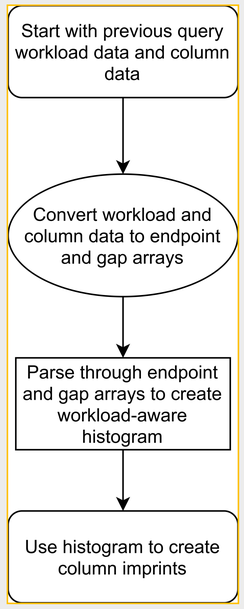

Figure 3: Histogram construction outline

- Column imprints use random selection from the column to construct a histogram
- The workload-aware histogram improves this by using the existing query workload to reduce false positive cache line scans
- Requires a dictionary-encoded column and a 'weight' for each unique value ID
- Step 1 takes recent query workload and value ID weights to create new data structures called gap and endpoint arrays
- Step 2 uses the gap and endpoint arrays to construct a workload-aware histogram in $O(nr)$ time, where n is the number of unique query endpoints, and r is the desired number of bins for the histogram

4. Histogram Experiment Results

- Setting: The PART table of TPCH-100, with out index on the RETAILPRICE column, synthetic workload (Zipfian distribution)
- The results show a 40% reduction in the average percentage of cache lines read using our histogram

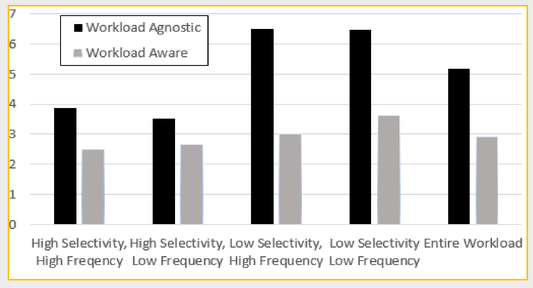

Figure 4: comparing average % of cache lines read

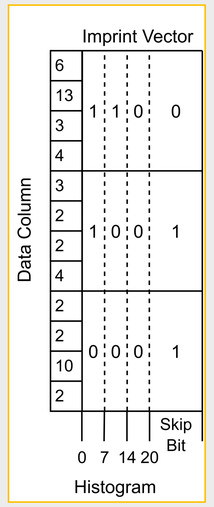

Figure 5: Skip Bit

5. Skip Bit

- High-frequency values can force column imprints to scan most of the data when querying, limiting imprint effectiveness
- The skip bit is a bit in each imprint vector reserved for such values, and essentially prunes them from rest of the histogram
- If a query seeks a high-frequency value, the column is simply scanned instead
- Figure 5 illustrates the skip bit, with 2 and 10 as the frequent values

6. Rotated Imprints

- For very high selectivity queries, reading every imprint bit is unnecessary as the query only targets 1 or 2 bits per imprint
- With a vertical layout, queries only check the relevant bits in each imprint vector
- Rotation also allows use of bitmap compression techniques, lowering TCO requirements

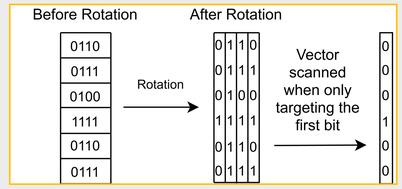

Figure 6: Rotated imprints example

7. References

[1] Lefteris Sidirourgos and Martin Kersten. 2013. Column imprints: a secondary index structure. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data.* ACM, 893–904.

ACM SIGMOD SRC

Tuesday, June 16th, 2020