Workload-Driven Antijoin Cardinality Estimation
Florin Rusu¹, Zixuan Zhuang¹, Mingxi Wu², Christopher Jermaine³
frusu@ucmerced.edu, zzhuang@ucmerced.edu, mingxi.wu@turn.com, cmj4@cs.rice.edu
¹UC Merced, ²Turn Inc., ³Rice University

Introduction

Motivating example.
Consider a standard TPC-H query “What is the total supply cost for parts less expensive than $700 supplied in 2007 by suppliers who did not supply any part in 2006?” having the corresponding SQL statement:

```sql
SELECT SUM(ps_supplycost*1.1_quantity) FROM partsupp ps, lineitem l1
WHERE ps_suppkey=l1.l_suppkey AND
ps_partkey=l1.l_partkey AND
year(l1.l_shipdate)=2007 AND
ps_supplycost < 700 AND NOT EXISTS (SELECT * FROM lineitem l2
WHERE l2.l_suppkey=l1.l_suppkey AND
year(l2.l_shipdate)=2006)
```

Following figure sketches three alternative let-deep evaluation plans for this query.

Table: Alternative left-deep execution plans for query. (a) is the histograms, and (b) is the corresponding weights.

<table>
<thead>
<tr>
<th>Execution Plan</th>
<th>Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>(b)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Problem statement & contributions.
The problem we consider in this paper is sampling-based antijoin cardinality estimation. We explore how a set of prior (antijoin queries and their results can be used to devise better sampling-based antijoin cardinality estimators for query optimization. To this end, we introduce a novel sampling-based estimator for antijoin cardinality that – unlike existing estimators – provides the required efficiency to be implemented in a query optimizer. The solution we propose is the first to integrate query workload information in the estimator definition using a Bayesian statistics framework.

Work-load Driven Estimation

we introduce the four steps of the proposed Bayesian inference framework. The learning phase uses statistical methods to build a prior histogram model composed of a number of candidate match frequency histogram patterns.

![Image](1)

In the characterization phase, a likelihood distribution of the current sample histogram $h^r$ corresponding to each learned histogram pattern is generated.

The multinomial maximum likelihood estimator $h^r$ computed over a set $MC = \{h_1^r, h_2^r, \ldots, h_C^r\}$ of MCc sample histograms is given by:

$$h^r[j] = \frac{\sum_{i=1}^{MC} h_i^r[j]}{\sum_{i=1}^{MC} \sum_{j=0}^{m} h_i^r[j]} 0 \leq j \leq m$$

(2)

The inference phase uses the likelihood distribution computed in the characterization phase and the observed sample histogram $h^r$ to infer the posterior histogram probabilities. The prior weights of the histogram model are updated based upon the evidence seen in the sample histogram $h^r$.

$$w_j = \frac{w_j \cdot pdf_{mult}(h_{query}^r | h_i^r)}{\sum_{i=1}^{MC} w_i \cdot pdf_{mult}(h_{query}^r | h_i^r)} 0 \leq j \leq c$$

(3)

The optimization phase generates a large number MCo of histograms from the superpopulation computed in the inference phase. The optimal estimator is computed such that it minimizes the sum-squared error (SSE):

$$SSE = \sum_{i=1}^{MC2} \left( C_i[0] - \sum_{j=0}^{m} b_j \cdot h^r[j] \right)^2$$

(4)

Experimental Evaluation

Figure: Prior histogram model, 10 components, 10 buckets. (a) is the histograms, and (b) is the corresponding weights.

![Image](a)

Figure: Accuracy (a) and (b) and execution time (c) as a function of the number of bins in the sample match frequency histogram.

TPC-H Data

Figure: Accuracy as a function of the number of mixtures.

Conclusion

In this paper, we present a novel sampling-based estimator for antijoin cardinality that provides the required efficiency to be implemented in a query optimizer. We plan to extend the proposed estimator to parallel settings where samples are distributed across computing nodes in future work.