Cost Model Improvements: An Architecture for Accurate and Consistent Cardinality Estimation

This document discusses state-of-the-art concerning consistent and accurate cardinality estimation and Presto's current limitations in this area. The discussion also presents detailed design considerations for removing these limitations.

In summary, this document proposes the following changes to the optimizer:

1. Eliminate "unknown" cardinality estimates in the optimizer. Replace "unknown" with heuristic values based on some specified rule.
2. Augment the filter statistics calculator to use the Adjustment Factor (AF) algorithm
3. Implement the Max Entropy (ME) algorithm for missing AFs
4. (Optionally) Use the History-Based Optimizer (HBO) data as a source for the ME implementation
5. (Optionally) Collect multivariate statistics through ANALYZE

The rest of document covers the details on why these changes are required, and the implementation considerations in Presto.

Background

The optimization process involves enumerating alternative execution plans from a search space of feasible alternatives. A cost model assigns a cost, an estimate of query execution plan efficiency, to each potential execution plan in the search space and the minimum cost execution plan executed. Various factors determine the cost of an execution plan; however, cardinality estimation, the process for determining the size of intermediate results after applying predicates or aggregation, plays an outsized role in cost estimation.

The goal of this document is to present the following:

1. Fundamental cardinality estimation techniques
2. Highlight the consistent cardinality estimation problem
3. Provide a high-level design to achieve consistent cardinality estimates
4. Design considerations for implementing the consistent estimations in Presto

Fundamental Cardinality Estimation Techniques

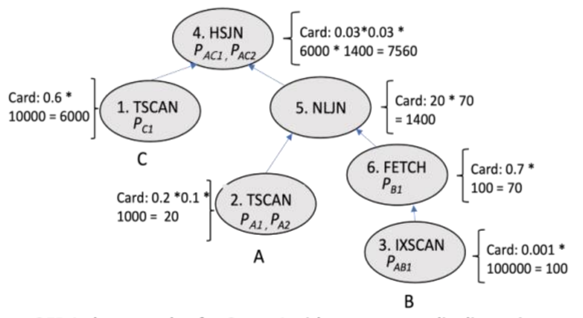
The cardinality estimation model forming the basis for the state-of-the-art originates from System R. That model performs cardinality estimation incrementally for each plan operator by multiplying the cardinalities of the operator’s inputs by estimates of the selectivity of the conjunctive predicates applied by the operator. System R assigned individual selectivity estimates to predicates based on statistical summaries maintained for both stored tables and the result tables produced by a plan operator. Its selectivity estimators are limited to using statistical summaries for individual attributes; hence, its cardinality estimation model assumed that the selectivity of each predicate was independent. The incremental cardinality estimation model had the desirable property of producing consistent cardinality estimates for equivalent subplans that may have applied individual predicates in different sequences, a critical aspect of any cardinality estimation model.

```
SELECT *
FROM A, B, C
```
WHERE P_{A1} AND P_{A2} AND P_{B1} AND P_{C1} AND P_{AB1} AND P_{AC1} AND P_{AC2}

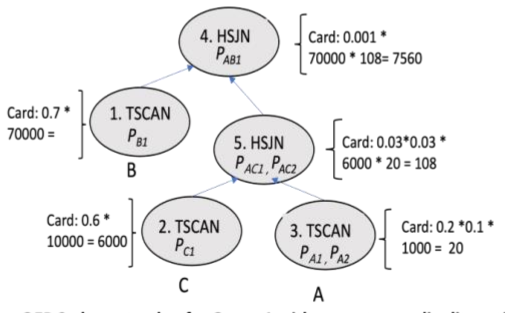
Table	Cardinality
A	1,000
B	100,000
C	10,000

Predicate	Selectivity	Predicate	Selectivity
P_{A1}	0.2	P_{AB1}	0.001
P_{A2}	0.1	P_{AC1}	0.03
P_{B1}	0.7	P_{AC2}	0.03
P_{C1}	0.6		

Query 1 with distribution statistics and selectivity estimates



QEP 1 alternate plan for Query 1 with operator cardinality estimates

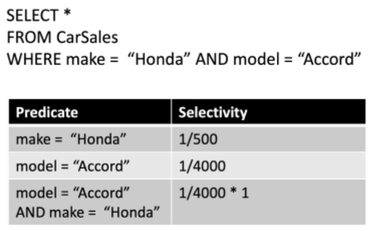


QEP 2 alternate plan for Query 1 with operator cardinality estimates

Figure 1: System R-style Incremental Cardinality Estimation

Figure 1 illustrates the cardinality estimation model of System R. It shows a query of tables A, B, and C having a WHERE clause with a search condition that includes both local and join predicates. Table 1 in the figure shows the cardinality of the input relations, and Table 2 selectivity estimates for the top-level conjuncts as derived from statistical summaries of column distributions not shown. The figure also exhibits two alternative execution plans for the query, with output cardinality estimates for each of the plan nodes. Note that even though the alternative execution plans apply predicates in different sequences, the final cardinality estimates of the plans are equivalent, which must always hold for plans that produce the same results. Put differently, cardinality estimates, and more generally, statistical summaries that characterize the result produced by a plan or subplan, are logical properties independent of any physical implementation.

Selectivity estimates in System R derive from simple statistical summaries that assume uniform column distributions such as the minimum, maximum, and number of distinct values of a column as well as independence between columns. These assumptions do not always hold which can lead to incorrect estimates. This document is not concerned with the predicate selectivity estimation aspect of cardinality estimation. **The focus is instead on presenting a state-of-the-art architecture for producing consistent and accurate cardinality estimates given a set of selectivity estimators that are presumed to be accurate**.



```
SELECT *
FROM CarSales
WHERE make =  “Honda” AND model = “Accord”
```

Predicate	Selectivity
make = “Honda”	1/500
model = “Accord”	1/4000
model = “Accord” AND make = “Honda”	1/4000 * 1

Figure 2: Statistically Correlated Predicates

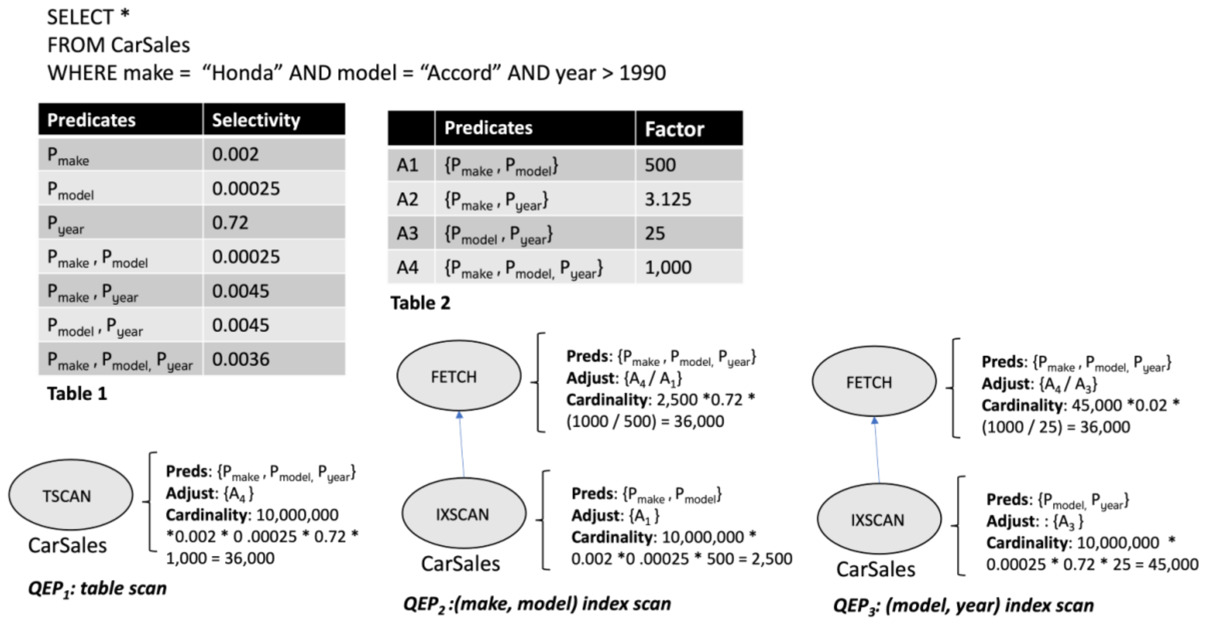
The Problem with Cardinality Estimation Under the Independence Assumption

In practice, the selectivity of predicates is not independent. In such cases, the basic incremental cardinality estimation approach described in the previous section leads to wildly underestimated cardinalities. The query in Figure 2, which seeks sales information for Honda Accords, illustrates the problem. The table in Figure 2 shows selectivity estimates for the predicates model = “Accord” and make = “Honda”, assuming 500 distinct car makers, an average of 8 models per manufacturer, and uniform distributions for both columns. If the CarSales table contains 10,000,000 rows, the optimizer would estimate the query cardinality as 10,000,000 * 1/500 * 1/4000 = 5 under the independence assumption. This represents a severe underestimation since rows satisfying the predicate model = “Accord” always satisfy the predicate make = “Honda” assuming only Honda manufactures Accords. In other words, the predicates are highly correlated and the joint selectivity estimates of the two are much lower than the product of the individual selectivity estimates.

Ideally, an optimizer would have multivariate statistics for every combination of columns referenced in the predicates of a workload; however, it is not feasible to collect and store all of this information. Thus, for a query search condition with conjunct predicates p_1, p_2, . . . , p_n , the optimizer typically has access to individual selectivity estimates s_1, s_2, . … s_n , as well as a limited collection of joint selectivity estimates, such as $s_{1,2}$, $s_{3,5}$, and $s_{2,3,4}$. The following sections describes an extension to the incremental cardinality estimation model that enables the derivation of consistent and accurate cardinality estimates using a, possibly incomplete, knowledge set consisting of individual and joint selectivity estimates.

Extending Incremental Cardinality Estimation with Adjustments

An adjustment is a pair of the form *(P, K)* where *P* is a set of predicates of size > 1 with a known joint selectivity estimate and where *K* is a numeric correlation factor used to adjust an incremental cardinality estimate when the predicates in *P* are applied. The value of *K* is the ratio of the joint selectivity estimate of *P* over the product of the individual selectivity estimates of the predicates in *P*. For example, an adjustment for predicates p_1, p_2 having selectivity estimates s_1, s_2 and a joint selectivity estimate $s_{1,2}$ is the pair *(P, K)* where $P = \{p_1, p_2\}$ and $K = s_{1,2} / s_1 * s_2$. The magnitude of *K* effectively represents the extent of statistical correlation between the predicates in *P*. Incremental cardinality estimates of a plan operator are corrected by the correlation factor of eligible adjustments to account for this correlation.



```
SELECT *
FROM CarSales
WHERE make =  "Honda" AND model = "Accord" AND year > 1990
```

Predicates	Selectivity
P_{make}	0.002
P_{model}	0.00025
P_{year}	0.72
P_{make} , P_{model}	0.00025
P_{make} , P_{year}	0.0045
P_{model} , P_{year}	0.0045
P_{make} , $P_{model,}$ P_{year}	0.0036

Table 1

	Predicates	Factor
A1	$\{P_{make}, P_{model}\}$	500
A2	$\{P_{make}, P_{year}\}$	3.125
A3	$\{P_{model}, P_{year}\}$	25
A4	$\{P_{make}, P_{model,} P_{year}\}$	1,000

Table 2

TSCAN (CarSales)
Preds: $\{P_{make}, P_{model,} P_{year}\}$
Adjust: $\{A_4\}$
Cardinality: 10,000,000 *0.002 * 0 .00025 * 0.72 * 1,000 = 36,000

QEP$_1$: table scan

FETCH
Preds: $\{P_{make}, P_{model,} P_{year}\}$
Adjust: $\{A_4 / A_1\}$
Cardinality: 2,500 *0.72 * (1000 / 500) = 36,000

IXSCAN (CarSales)
Preds: $\{P_{make}, P_{model}\}$
Adjust: $\{A_1\}$
Cardinality: 10,000,000 * 0.002 *0 .00025 * 500 = 2,500

QEP$_2$:(make, model) index scan

FETCH
Preds: $\{P_{make}, P_{model,} P_{year}\}$
Adjust: $\{A_4 / A_3\}$
Cardinality: 45,000 *0.02 * (1000 / 25) = 36,000

IXSCAN (CarSales)
Preds: $\{P_{model,} P_{year}\}$
Adjust: : $\{A_3\}$
Cardinality: 10,000,000 * 0.00025 * 0.72 * 25 = 45,000

QEP$_3$: (model, year) index scan

Figure 3: Incremental Cardinality Estimation using Adjustment Factors

The adjust factor approach is further elaborated in Figure 3. It extends plan operators with two new properties: a predicate property, *Preds,* and an adjustment property, *Adjust,* which take into account, respectively, all predicates and adjustments applied thus far. An operator forms a cardinality estimate incrementally as before by computing the product of the input cardinality estimates of its source operators and the individual selectivity estimates of any predicates *OP* applied by the operator. The process then adjusts this estimate by the correlation factors of newly eligible adjustments. An adjustment *A= (P, K)* is *eligible* if *P* is a subset of the union of *Preds* and *OP,* and *A* is not a member of *Adjust*.

Incremental cardinality estimation with adjustments enables the optimizer to form consistent and accurate cardinality estimates using available joint selectivity information; however, consistency requires certain conditions. Consider again the example in Figure 3 and assume the joint selectivity estimate of the three predicates P_{make}, P_{model}, and P_{model} was unavailable. In this scenario, adjustment A4 is unavailable, and the overlapping adjustments A1, A2, and A3 all become eligible at the TSCAN operator of QEP1 since the subsuming adjustment A4 is not available to prune them. It isn't clear which of those three eligible overlapping adjustments to apply. The problem of eligible overlapping adjustments would also occur at the FETCH operators of QEP2 and QEP3 as none of the newly eligible adjustments would trigger the adjustment previously applied by the IXSCAN to get backed out; hence, inconsistent estimates would result since those IXSCAN operators applied different adjustments. The adjustment approach provides consistent and accurate cardinality estimates only if there is a subsuming adjustment for any overlapping adjustments. That is, if there are two adjustments (P1, K1) and (P2, K2) where P1 and P2 overlap, there is a third adjustment (P3, K3) where P1 + P2 is subsumed by P3. Hence, if given joint selectivity estimates for P1 and P2, we must also have a joint selectivity estimate for P3.

Filling in the Joint Selectivity Gaps with Max Entropy

The Max Entropy approach to cardinality estimation (ME) [8] formalizes the problem of selectivity estimation for conjunctive predicates given partial joint selectivity information. For example, from a known set of selectivity estimates such as {S_1, S_2, S_3, $S_{1,2}$, $S_{2,3}$} it can provide estimates for $S_{1,3}$ and $S_{1,2,3}$ that are consistent relative to the initial set of known estimates; hence, ME enables the adjustment approach to form consistent cardinality estimates by generating missing adjustments that subsume overlapping adjustments.

ME formulates an optimization problem where the objective is to find missing joint selectivity estimates subject to a set of constraints represented by an initial set of known individual and joint selectivity estimates. The solution determines a probability distribution that maximizes the entropy function [9] and is consistent concerning the known information. ME formulates the optimization problem for a given set of predicates $P = \{p_1, p_2, . ., p_n\}$ by encoding each predicate as a binary string of length n based on a disjunctive normal form (DNF) representation of those predicates. For example, when $n = 3$, the string $b = 100$ denotes the DNF representation $p_1 \wedge \neg p_2 \wedge \neg p_3$,

Predicates
$\{p_1, p_2, p_3\}$

Selectivity Constraints
(i) $s_1 = x_{100} + x_{110} + x_{101} + x_{111} = 0.1$
(ii) $s_2 = x_{010} + x_{011} + x_{110} + x_{111} = 0.2$
(iii) $s_3 = x_{001} + x_{011} + x_{101} + x_{111} = 0.25$
(iv) $s_{1,2} = x_{110} + x_{111} = 0.05$
(v) $s_{1,3} = x_{100} + x_{001} = 0.05$
(vi) $x_{000}\, x_{100} + x_{010} + x_{001} + x_{110} + x_{101} + x_{011} + x_{111} = 1$

Entropy function to maximize
$-\Sigma\, x_b \log x_b$

Obtained selectivity estimates
$s_{1,2,3} = x_{111} = 0.015$
$s_{2,3} = x_{111} + x_{011} = 0.05167.$

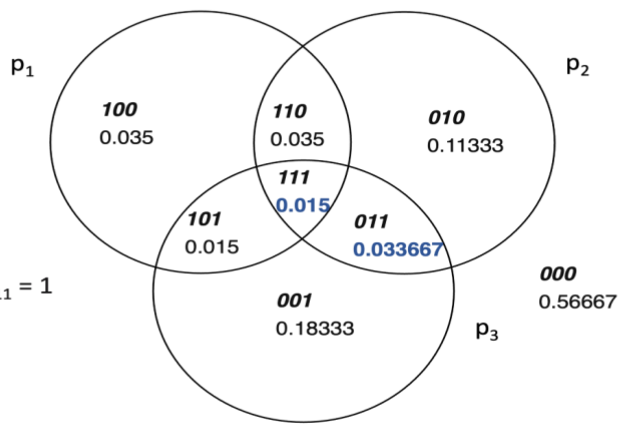


Figure 4: Max Entropy Cardinality Estimation Example

Figure 4 shows the probability space and optimization problem formulation for P = {p1, p2, p3} given known selectivity estimates $S_1 = 0.1$, $S_2 = 0.2$, $S_3 = 0.25$, $S_{1,2} = 0.05$, S1,2=0.03. Each x_{abc} in the probability space represents a DNF encoding of P. For example, if p_1 represents make = “Honda” and p_2 represents model = “Accord”, x_{11} represents the query predicate ‘make = “Honda” AND model = “Accord”’ whereas x_{01} represents ‘make != “Honda” AND model = “Accord”‘. Moreover, an algebraic constraint of x_{abc}terms is formed for each known selectivity estimate. For example, $S_{1,2}$ is represented by the constraint $x_{110} + x_{111} = 0.5$. Using the previous example of makes and models you had run a previous query where you know the selectivity of just ‘make = “Honda”’ = 0.4, you could model this as a constraint as $x_{10} + x_{11} = 0.4$. ME computes a solution for all x_{abc} in the probability space using Lagrange multipliers and an iterative scaling algorithm as described in [8]. The selectivity estimates computed for the example problem in Figure 4 are $s_{1,2,3} = x_{111} = 0.015$ and $s_{2,3} = x_{111} + x_{011} = 0.05167$.

ME also addresses practical details such as zero-term elimination, which occurs when $S_1 = S_{1,2}$, and the elimination of mutual inconsistencies between selectivity estimates in the initial knowledge set like S1 >= S1,2. The ME algorithm is parallelized by partitioning the problem space into non-empty disjoint subsets. For example, if the initial knowledge set is S1,2 S2,3, S3,4, S5,6, S6,7. ME can be solved separately for disjoint partitions P1 = S1,2 S2,3, S3,4 and P2 = S5,6, S6,7. The final solution ensures subsuming joint selectivity estimates for any two overlapping estimates in P1 and P2, as there are no overlapping adjustments that span disjoint sets P1 and P2.

Presto Design Considerations

Current State of Optimizer Cardinality Estimation

Join order enumeration is the primary context where the optimizer makes cost-based decisions. When computing the cost of a plan node, the optimizer determines global a cardinality estimate and distribution statistics for the output variables of each source plan node in a bottom-up recursive fashion starting from base table-level or partition-level statistics.

The optimizer derives selectivity estimates for filters, joins, and other operators that apply predicates using the propagated variable statistics assuming variable distributions are statistically independent. Local cardinality estimates are formed by uniformly apportioning the global estimate to local execution costs depending on the degree of parallelism for a stage. The basic model does not support state-of-the-art estimators such as single or joint column histograms, statistics on views, predicate sampling, or AI models; hence, it is analogous to the System R cardinality estimation model as previously described.

Cost and Cardinality Estimation Model Uncertainty

The Presto cost-based optimizer effectively gives up and resorts to heuristics when it loses confidence in its cardinality and cost estimates. This occurs in various situations, like when it confronts complex predicates or those involving variables where statistics cannot be derived. The optimizer does not continue on using default selectivity values, as would most state-of-the-art optimizers. The cases where the estimation model enters the uncertain state result in unpredictable optimization behavior; hence, eliminating this modus operandi is highly recommended. There are a variety of techniques for mitigating uncertainty, including exploiting optimization-time sampling [10, 11]. The history-based optimizer is also a mitigator of the uncertainty problem. Default selectivity estimates are the preferred fallback when all other options are exhausted.

History-Based Optimizer

The history-based optimizer (HBO) improves cardinality estimation by introducing a feedback loop from query execution to query optimization, similar in spirit to [6]. The current focus is improved cardinality estimates for joins and partial aggregation; however, it is applicable to any subplan types. The runtime component of HBO collects actual cardinality and output variable statistics for targeted subplans and associates them with a Merkle hash of a canonicalized logical representation of the subplan. The HBO maintains the combination of Merkle subplan hash and actual statistics in a history database. The query optimization component of HBO uses the actual cardinality and variable statistics stored in the history database rather than the usual estimates for subplans whose Merkle hash matches the history database hash. This aspect of HBO is similar to [4, 5], which improves cardinality estimates by exploiting statistics on views. Notably, HBO is a source for obtaining joint selectivity estimates and multivariate statistics that, if used consistently as per the prior art, would allow the presto optimizer to obtain more accurate cost estimates.

Summary

In order to achieve consistent cardinality estimations, Presto needs to develop a new framework within the optimizer for performing its cardinality calculations.

The first hurdle to tackle is removing “unknown” estimates, even when no statistics are available to the optimizer. A new heuristic needs to be agreed upon and used in the cases of unknown values. This requires an update to the `Estimate` class for Presto’s cost calculations and locating all the call-sites for unknowns in order to replace them with some type of heuristic. By removing unknown heuristics, the optimizer will be able to use the AF calculations

The AF algorithm for cardinality estimations introduces the ability for the optimizer to account for column correlations within the cardinality estimation framework. Introducing AF into the Presto optimizer’s current cardinality calculations enables correct calculations in the face of correlated columns.

Finally, for the search space to be explored properly plans need to come up with consistent estimates, regardless of whether the AF factors are known or not. This is where the ME algorithm helps generate consistent estimates even when AF factors are unknown. When using ME without any known AF factors, the algorithm converges on the independence assumption. However, with some AF factors known, you can achieve better *and* consistent estimates.

Lastly, the data sources used for AF factors to the ME estimation can be pulled from a variety of sources. This could come from the HBO, table or partition stats, or even stored samples. We should provide a way in the cardinality estimation framework to derive known AF factors from a variety of sources during the optimization phase.

References

[1] Selinger, P. G., Astrahan, M-M., Chamberlin. D. D., Lorie. R. A., Price T. G. Access Path Selection in a Relational Database System. In Readings in Database Systems Morgan Kaufman.

[2] Poosala.V.. Ioannidis. Y.. Harts, P., Shekita. E. Improved Histograms for Selectivity Estimation. In Proc. of ACM SIGMOD, Montreal, Canada 1996.

[3] Poosala, V., Ioannidis, Y.E. Selectivity Estimation Without the Attribute Value Independence Assumption. In Proc. of VLDB, Athens,1997.

[4] Query optimization technique for obtaining improved cardinality estimates using statistics on pre-defined queries **Patent number:** 8386450. David Simmen

[5] Exploiting statistics on query expressions for optimization. Bruno, N. Chaudhuri, S. Proceedings of the 2002 ACM SIGMOD international conference on Management of data. June 2002

[6] Stillger, M., Lohman, G., Markl, V., Kandil, M.: LEO – DB2’s learning optimizer. VLDB 19–28 (2001)

[7] NeuroCard: One Cardinality Estimator for All Tables
Zongheng Yang, Amog Kamsetty, Sifei Luan, Eric Liang, Yan Duan, Xi Chen, Ion Stoica. VLDB 2021.

[8] Consistent selectivity estimation via maximum entropy V Markl, PJ Haas, M Kutsch, N Megiddo, U Srivastava… - The VLDB journal, 2007

[9] Entropy (Information Theory)
https://en.wikipedia.org/wiki/Entropy_(information_theory)

[10] Reference to Zack's stored sample work.

[11] Reference to Anant's quick-stats work.