**Plan Bouquet based Techniques for Variable Sized Databases**

Achint Chaudhary

February 2, 2020

MTech Project Report (CSA)

**0 Abstract**

OLAP applications often require a certain set of canned queries to be fired on database with the varying constants in queries. For optimal execution of these queries, query optimizer does select a strategy known as the query execution plan. These choices are based on cardinality estimates of various predicates that often hugely differ from actual cardinality values encountered during execution. Due to this reason, optimizer choice leads to high inflation in actual execution cost as compared to predicted cost during optimization.  
  
An altogether different approach for query processing was proposed in 2014, named Plan Bouquet. Basis of which is selectivity discovery at run-time by repeated cost bounded execution of carefully chosen to set of plans. This technique provides strong bounds independent of data distribution.

However, Plan Bouquet on cost sub-optimality is not designed to be robust against large updates in the database. This work focuses on observing limits up to which size of database can be increased without serious deterioration in performance guarantee. Also, we will provide incremental algorithms that can use information from plan bouquet compiled in past and extend it, to provide further robust execution without incurring overhead of re-compiling entire plan bouquet.

**Sec 1 Introduction**

Database query optimizer chooses a plan covering various structural choices of logical and physical operators for query execution. These choices are based on the cost of each operator which is calculated using number of tuples it will process known as cardinality. Cardinality normalized in range of [0, 1] is known as selectivity throughout our analysis.

These selectivity values are estimated before query execution based on some statistical models used in classical cost-based optimizers. An entirely different approach based on run-time selectivity discovery is proposed called Plan Bouquet, which provides for the first time strong theoretical bounds on worst-case performance as compared to optimal performance possible from all the available plan choices.

For each given query, predicates having the potential of selectivity error contribute as a dimension in Error-prone Selectivity Space (ESS). The set of optimal plans over the entire range of selectivity values in ESS is called Parametric Optimal Set of Plans (POSP). POSP is generated by asking optimizer's chose plans at various selectivity locations in ESS using Selectivity injection module. Cost surface generated over entire ESS is called Optimal Cost Surfaces (OCS). An Iso-cost surface is collection of all points from OCS which have same optimal plan cost at that location.

A subset of POSP is identified as Plan bouquet, which is obtained by the intersection of plans trajectory with OCS, creating multiple Iso-cost surfaces, each of which is placed at some ratio proportion (r\_pb) of cost from the previous surface.

Since each plan on an iso-cost surface has a bounded execution limit, and incurred cost by execution using bouquet will form geometric progress. The figure below shows the performance of bouquet w.r.t to optimal oracle performance.

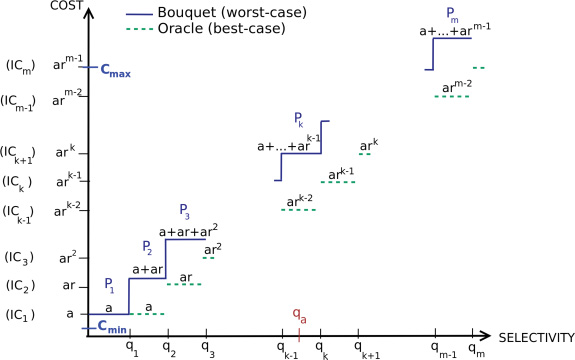


Fig 1. Cost incurred by Oracular vs Bouquet Execution

In above figure, various plans up to actual selectivity value are executed. Each plan has a limit provided by the next iso-cost surface. This yields total execution cost of

This value is minimized using r\_pb=2, which provides theoretical worst case bound of 4 times the optimal execution time.

Extending the same idea to multiple dimensional ESS, MSO guarantee will become 4\*p, where p is maximum cardinality of plans on any of iso-cost surface. Since computing value of p will need huge compile time effort, it is platform dependent and also low value of p is desired, which was obtained using Anorexic reduction at the time plan bouquet was developed.

Later a improved algorithm called SpillBound is invented, which is able to provide performance guarantee based only on query inspection and is quadratic function in number of error-prone predicates, which is same as dimensionality of ESS. MSO guarantee by Spillbound is

D\*\*2+3\*D

We

2 PROBLEM FORMULATION

2.a Notations

2. b Assumptions

Plan Cost Monotonicity (PCM)

Axis Parallel Concave(APC)

Axis Parallel Linear (APL)

Bounded Cost Growth (BCG)

EPP are the only predicates

Perfect Cost Model of Optimizer

2.c Performance Metrics

2.d Range of selectivity

Selectivity is the fraction of tuples out of maximum possible tuples that can come out of a query predicate. Notation of selectivity is devised to make study of ESS independent of cardinality values. This has motivated in past literature that selectivity value be always bounded in range [0,1].

Looking at type of changes possible in database, we first consider that Plan bouquet and techniques later developed on base of it are robust to data distribution, so only changes in distribution will not call for need of re-compilation since plan bouquet from past will provide same performance guarantee as it used to in some point of time. This is with the assumption that all tuple processing nodes are error prone

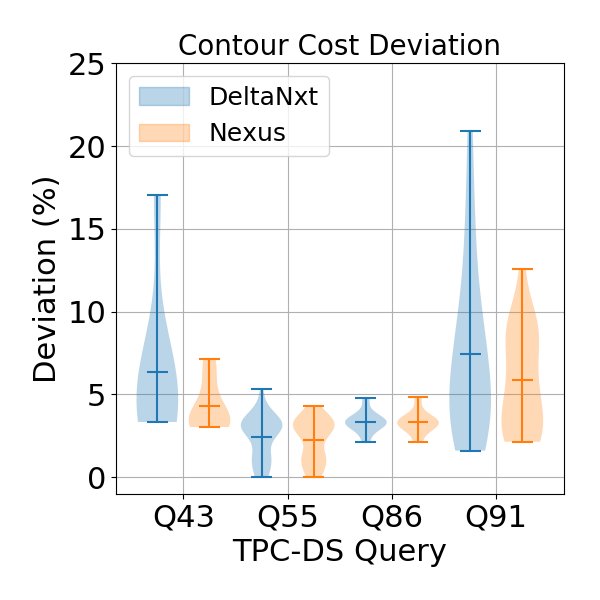
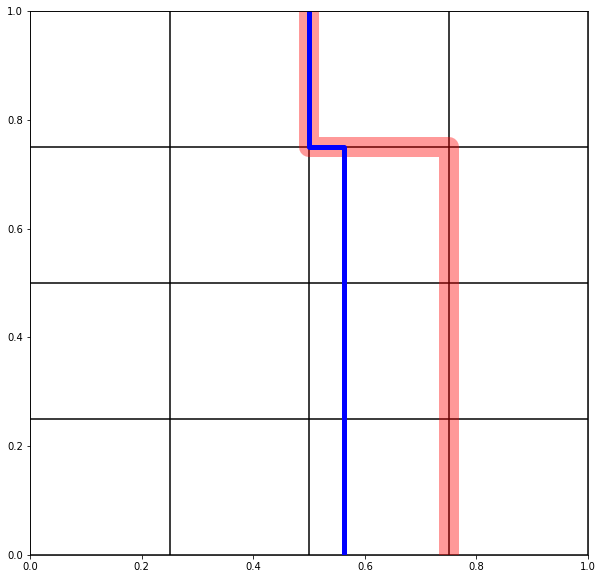
Considering increase in size of database, keeping same distribution as before. Number of tuples to process now has increased. So, if we go with [0,1] picture of selectivity again, same selectivity value will result in different cardinality. As query optimizers are cost based, cost model consider the cardinality and for sake of uniformity in picture of old and new database throughout our analysis should provide same cardinality provided same selectivity. In a loose sense, selectivity to cardinality mapping should not be changed.

Under this case, if a

3. CHALLENGES

NEXUS with linear step & Full space exploration

Cons of using low resolution (Both in case of NEXUS & Full space exploration)



Cons of using uniform distribution, using concavity of OCS