The Picasso Database Query Optimizer Visualizer

Jayant R. Haritsa
Database Systems Lab, SERC/CSA
Indian Institute of Science, Bangalore 560012, INDIA
haritsa@dsl.serc.iisc.ernet.in

1. INTRODUCTION

Modern database systems employ a *query optimizer* module to automatically identify the most efficient strategies for executing the declarative SQL queries submitted by users. The efficiency of these strategies, called "plans", is measured in terms of "costs" that are indicative of query response times. Optimization is a mandatory exercise since the difference between the costs of the best execution plan, and a random choice, could be in orders of magnitude. The role of query optimizers has become especially critical during this decade due to the high degree of processing complexity characterizing current data warehousing and mining applications, as exemplified by the TPC-H and TPC-DS decision support benchmarks [20, 21].

Over the course of the last five years, we have developed a visualization tool, called **Picasso** [22], for graphically profiling and analyzing the behavior of database query optimizers. The tool is operational on a rich set of industrial-strength optimizers, including IBM DB2 [15], Microsoft SQL Server [16], Oracle [17], Sybase ASE [18] and PostgreSQL [19]. Picasso, which is freely downloadable, is currently in use by leading industrial and academic institutions worldwide. It has been employed as

- a query optimizer analysis, debugging, and redesign aid by system developers;
- a query optimization test-bed by database researchers; and
- a query optimizer pedagogical support by database instructors and students.

The scientific underpinnings of the Picasso tool have previously appeared in a series of recent VLDB papers [12, 7, 8, 6, 1]. In this demo, we will first present a walk-through of the Picasso tool, and explain how it provides powerful visual metaphors to explore in detail the intriguing world of modern database query optimizers. We will then show how the tool can be used to efficiently determine *improvements* on the plan choices made by the optimizer – for example, to identify "robust plans" that are resistant to selectivity estimation errors on the query's base relations. Finally, we will indicate how these concepts have important implications for the design of next-generation query optimizers.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Articles from this volume were presented at The 36th International Conference on Very Large Data Bases, September 13-17, 2010, Singapore.
Proceedings of the VLDB Endowment, Vol. 3, No. 2
Copyright 2010 VLDB Endowment 2150-8097/10/09... $ 10.00.

2. PICASSO DIAGRAMS

Given a parametrized SQL query template that defines a relational selectivity space, and a choice of database engine, the Picasso tool automatically generates a variety of diagrams that characterize the behavior of the engine's optimizer over this space. For example, the so-called "Plan Diagram" [12], representing a color-coded pictorial enumeration of the plan choices made by the optimizer over the selectivity space. Specifically, plan diagrams visually capture the optimality regions of POSP [9], the parametric optimal set of plans.

To make these notions concrete, consider QT8, the parametrized 2D query template shown in Figure 2, based on Query 8 of the TPC-H benchmark. Here, selectivity variations on the SUPPLIER and LINEITEM relations are specified through the s_acctbal :varies and l_ extendedprice :varies predicates, respectively.

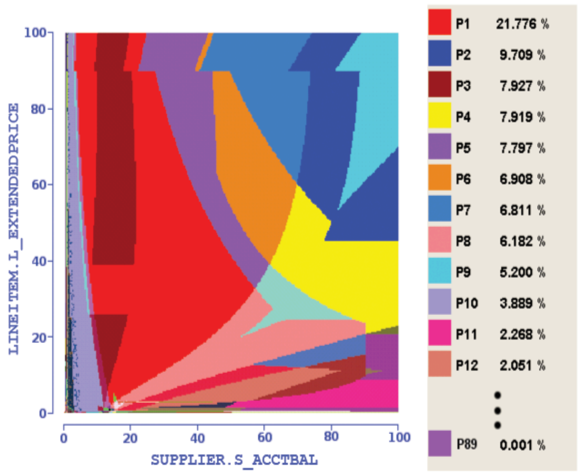
```
select o_year, sum(case when nation = 'BRAZIL' then volume else 0
    end) / sum(volume) as mkt_share
from (select YEAR(o_orderdate) as o_year, l_extendedprice * (1 -
    l_discount) as volume, n2.n_name as nation
      from part, supplier, lineitem, orders, customer,
          nation n1, nation n2, region
      where p_partkey = l_partkey and s_suppkey = l_suppkey
          and l_orderkey = o_orderkey and o_custkey = c_custkey
          and c_nationkey = n1.n_nationkey and n1.n_regionkey =
          r_regionkey and s_nationkey = n2.n_nationkey and r_name
          = 'AMERICA' and p_type = 'ECONOMY ANODIZED
          STEEL' and
          s_acctbal :varies and l_extendedprice :varies
      ) as all_nations
group by o_year
order by o_year
```

Figure 1: Example Query Template (QT8)

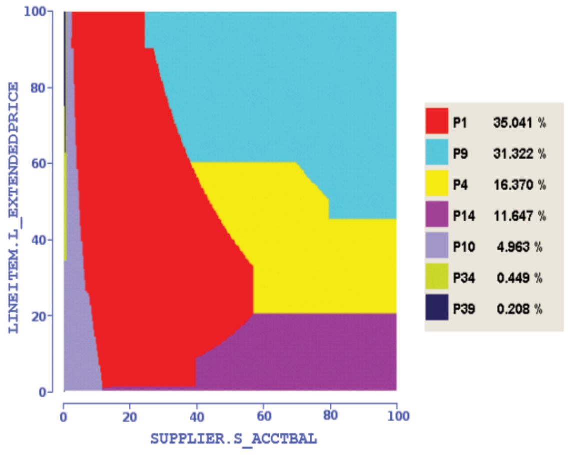
The associated plan diagram for QT8 is shown in Figure 2(a), produced by Picasso on a popular commercial database engine. In this picture[1], each colored region represents a specific plan, and a set of 89 different optimal plans, P1 through P89, cover the selectivity space. The value associated with each plan in the legend indicates the percentage area covered by that plan in the diagram – for example, the biggest, P1, covers about 22% of the space, whereas the smallest, P89, is chosen in only 0.001% of the space.

Compile-Time Diagrams. The complete suite of diagrams produced by the Picasso tool is enumerated in Table 1. It includes several compile-time diagrams that qualitatively and quantitatively describe the plan choices made by the optimizer. For instance,

[1]The figures in this paper should be viewed from a *color* copy, as the grayscale version may not clearly register the features.



(a) **Plan Diagram**



(b) **Reduced Diagram (Threshold λ = 10%)**

Figure 2: Sample Plan Diagram and Reduced Plan Diagram (QT8)

the Cost Diagram quantitatively depicts the estimated query processing costs of the plans shown in the associated plan diagram, while the Cardinality Diagram displays the estimated result cardinalities. These diagrams can be drilled-down at individual locations to determine the operator trees of the plans at those locations (Schematic Plan-tree diagram), with the tree nodes optionally annotated with cost and cardinality information (Compiled Plan-tree diagram). The structural differences between a given pair of plans can be identified through the Plan-Difference diagram, with the differences highlighted in a color-coded format.

Picasso also supports comparison of an optimizer's plan choices with those made by *other engines* at the same locations, or by the same engine operating at a different optimization level (Foreign Plan-tree diagram). So, for example, an IBM developer interested in a particular query instance, can visually ascertain and compare DB2's plan choice for this query with those made by other engines such as Oracle and SQL Server. Finally, recent versions of several optimizers have included a "foreign plan costing" (FPC) feature in their API, that is, of costing plans *outside* their native optimality regions (e.g. Optimization Profile in DB2, XML Plan in SQL Server, and Abstract Plan in Sybase ASE). This FPC feature is used by Picasso to visually characterize the cost behavior of a designated plan over the entire selectivity space (Abstract-Plan diagram).

Plan-replacement Diagrams. Perhaps the most appealing aspect of Picasso is that it also supports the construction of *plan-replacement diagrams*. Here, the query template's original plan/cost diagrams are taken as input, and new plan diagrams are constructed wherein a subset of the optimizer's original choices are replaced by alternative plans from the POSP set. The replacements are made on the expectation that they will perform better than the original choices (Reduced Plan and Robust Plan diagrams). The motivation for these substitute diagrams, and their construction techniques, are discussed in detail in Section 5.

Run-time Diagrams. Finally, apart from the above compile-time diagrams, Picasso also generates *run-time* diagrams that visually describe the *actual* query performance behavior, in terms of execution time and result cardinalities, on the current database platform (Execution Cost and Execution Cardinality diagrams). Comparing the predicted and actual diagrams helps in understanding and profiling the modeling quality of the optimizer.

Compile-time Diagrams

Plan Diagram	A pictorial enumeration of the optimizer's execution plan choices over the selectivity space.
Cost Diagram	A visualization of the associated estimated plan execution costs over the selectivity space.
Cardinality Diagram	A visualization of the associated estimated query result cardinalities over the selectivity space.
Schematic Plan-tree Diagram	A tree visualization of a selected plan in the plan diagram.
Plan-difference Diagram	Highlights the schematic differences between a selected pair of plans that appear in the plan diagram.
Compiled Plan-tree Diagram	A tree visualization of a selected plan at a specific location in the plan diagram, annotated with cost and cardinality information.
Foreign Plan-tree Diagram	At a given location in a plan diagram produced on a database engine, a tree visualization of the plan produced by another engine (or the same engine at another optimization level) at this location.
Abstract-Plan Diagram	A visualization of the estimated behavior of a selected plan in the plan diagram, when this specific plan is used throughout the selectivity space.

Plan-replacement Diagrams

Reduced Plan Diagram	Shows the extent to which the original plan diagram may be simplified (by replacing some of the plans with their siblings in the plan diagram) without increasing the cost of any individual query by more than a user-specified threshold value.
Robust Plan Diagram	Shows the extent to which the plans in the original plan diagram may be replaced by comparatively robust plans without increasing the cost of any individual query by more than a user-specified threshold value.

Run-time Diagrams

Execution Cost Diagram	A visualization of the runtime query response times over the selectivity space.
Execution Cardinality Diagram	A visualization of the runtime query result cardinalities over the selectivity space.

Table 1: Picasso Diagram Suite

3. DIAGRAM PRODUCTION

The schematic architecture of the Picasso system is shown in Figure 3. The plan diagram production strategy used is the following: Given a d-dimensional query template and a plot resolution of r, the Picasso tool generates r^d queries that are either uniformly or exponentially (user's choice) distributed over the selectivity space. Then, for each of these query locations, based on the associated selectivity values, a query with the appropriate constants is instantiated – the constants are determined from the statistical meta-data available from the optimizer, typically in the form of histograms. This query is then submitted to the query optimizer to be "explained", that is, to have its optimal plan computed and returned.

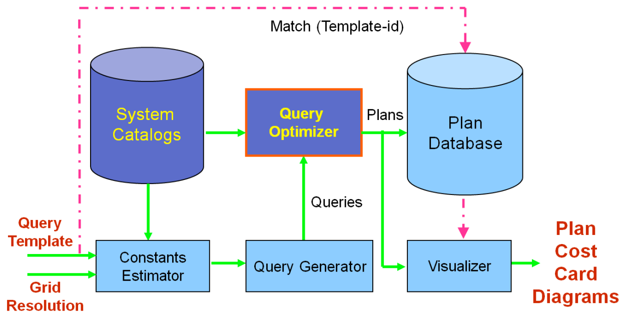


Figure 3: Picasso Architecture

After the plans corresponding to all the query points are obtained, a different color is associated with each unique plan, and all query points are colored with their associated plan colors. Then, the rest of the diagram is colored by painting the region around each point with the color corresponding to its plan. For example, in a 2D plan diagram with a uniform grid resolution of 10, there are 100 real query points, and around each such point a square of dimension 10x10 is painted with the point's associated plan color.

In parallel with the construction of the plan diagram, the (estimated) cost and cardinality diagrams are created using the quantitative information provided in the "explain plan" output. These diagrams, as well as the operator trees corresponding to the POSP plans, are persistently stored in the database to facilitate diagram reuse. Further, statistical estimators have been implemented to provide users with predictions of diagram production times.

The Picasso tool is completely written in Java and currently runs to about 50K lines of code with approximately 100 classes. Java3D, VisAd, Swing and JGraph libraries are used for visualization purposes, while database connections are established through JDBC drivers. Version 1 of Picasso was released in 2007, while Version 2 came out in 2009. With V2, users can restrict diagram production to desired *sub-regions* of the selectivity space – for example, near the origin, where high volatility in optimizer plan choices is typically encountered.

Another major feature of Version 2 is the introduction of *approximate* plan diagrams. The motivation here is that the exhaustive diagram production approach described above is practical only for diagrams on low-dimension query templates (1D and 2D) with coarse resolutions (up to 100 points per dimension). However, it becomes infeasibly expensive for higher dimensions and fine-grained resolutions, due to the exponential growth in overheads. For example, a 2D plan diagram with a resolution of 1000 on each dimension, or a 3D plan diagram with a resolution of 100 on each dimension, both require invoking the optimizer a *million* times. Even with a conservative estimate of 0.5 seconds per optimization, the total time required to produce the entire picture is close to a week!

This issue of computational overheads is now addressed in Picasso through the incorporation of powerful sampling and inference-based approximation techniques [6]. These techniques deliver diagrams with close to 90% accuracy while incurring only about 10% of the overheads of the brute-force approach.

4. APPLICATIONS OF PLAN DIAGRAMS

As evident from Figure 2(a), plan diagrams can be surprisingly complex and dense, with a large number of plans covering the space – several such instances spanning a representative set of benchmark-based query templates on current optimizers are available at [22]. In fact, the very name of the Picasso tool stems from plan diagrams often appearing similar to "cubist paintings".[2]

Our interactions with industrial development teams indicate that Picasso plan diagrams have often proven to be contrary to the prevailing conventional wisdom. The reason is that while optimizer behavior on *individual queries* has certainly been analyzed extensively by developers, plan diagrams provide a completely different perspective of behavior *over an entire space*, vividly capturing plan transition boundaries and optimality geometries. So, in a literal sense, they deliver the "big picture".

Plan diagrams are currently in use at various industrial and academic sites for a diverse set of applications including analysis of existing optimizer designs; visually carrying out optimizer regression testing; debugging new query processing features; comparing the behavior between successive optimizer versions; investigating the structural differences between neighboring plans in the space; evaluating the variations in the plan choices made by competing optimizers; etc. As a case in point, visual examples of *non-monotonic* cost behavior in commercial optimizers, indicative of modeling errors, were highlighted in [12].

Apart from aiding optimizer design, plan diagrams can also be used in operational settings. Specifically, since they identify the optimal set of plans for the entire relational selectivity space at compile-time, they can be used at run-time to immediately identify the best plan for the current query without going through the time-consuming optimization exercise. Further, they can prove useful to adaptive plan selection techniques (see [5] for a recent survey) which, based on run-time observations, may dynamically choose to re-optimize the query and switch plans mid-way through the processing. In this context, plan diagrams can help to eliminate the re-optimization overheads incurred in determining the substitute plan choices.

5. PLAN-REPLACEMENT DIAGRAMS

As mentioned earlier, apart from visually profiling optimizer behavior, Picasso also incorporates mechanisms for *improving* on the optimizer's plan choices through the production of "reduced plan diagrams" and "robust plan diagrams", described below.

5.1 Reduced Plan Diagrams

Consider a dense original plan diagram and a cost-increase-threshold (λ) specified by the user. Our reduction algorithms recolor the dense diagram to a simpler picture, featuring only a subset of the original plans – that is, some of the original plans are "completely swallowed" by their siblings, leading to a reduced number of plans in the diagram. Most importantly, the recoloring process guarantees that the cost of *any* recolored query point does *not* increase by more than λ percent, relative to its original cost.

[2]Pablo Picasso is considered to be a founding-father of the cubist school of painting [14].

It has been empirically shown in [7] that if users are willing to tolerate minor cost increases of up to λ =20%, the absolute number of plans in the final reduced picture could usually be brought down to *within or around ten*. In short, that complex plan diagrams can be made "anorexic" while retaining acceptable query processing performance. For example, the QT8 plan diagram (Figure 2(a)) can be reduced with λ =10% to the diagram shown in Figure 2(b), where only 7 of the original 89 plans are retained.

Anorexic plan diagram reduction has significant practical benefits, as described in detail in [7], including quantifying the redundancy in the plan search space, enhancing the applicability of parametric query optimization (PQO) techniques [9, 10], identifying error-resistant and least-expected-cost plans [3, 4], and minimizing the overhead of multi-plan approaches [2, 11].

5.2 Robust Plan Diagrams

An implicit assumption in producing reduced plan diagrams with the λ-guarantee is that the optimizer's compile-time estimates of query locations in the selectivity space are accurate. However, in practice, these selectivity estimates are significantly in error with respect to the run-time values encountered during query execution. Such errors, which can even be in orders of magnitude in real database environments, arise due to a variety of reasons, including outdated statistics, attribute-value independence assumptions and coarse summaries [13].

Given the above, the replacements suggested by the reduced plan diagrams, while within the λ threshold at the estimated query locations, may turn out to be *arbitrarily* better or worse substitutes in the presence of selectivity estimation errors. Therefore, we would ideally like to only permit replacements that are guaranteed to either improve the query processing performance or not have any adverse effects, *no matter where the actual query location turns out to be at run-time*. Surprisingly enough, it is actually possible to efficiently identify such replacements for optimizers supporting the foreign-plan-costing (FPC) feature, as described in [8]. This approach is implemented in Picasso to ensure *globally safe* replacements which can only improve, but never harm query processing performance. Further, our empirical results suggest that significant improvement is often provided, effectively resulting in *robust plans*. Interestingly, robust plan diagrams typically *retain* the anorexic properties of reduced plan diagrams. Therefore, in a nutshell, our results indicate that it is indeed possible to *simultaneously achieve plan safety, robustness and anorexia*, in industrial-strength database settings.

As a final step, we have shown very recently in [1], how the above post-processing steps on plan diagrams can be *internalized* into the *online* query optimization process, resulting in an intrinsically improved optimizer that delivers better plan choices. It is particularly noteworthy that this desirable outcome is achieved in spite of the online process lacking the global behavioral information available to the offline algorithms.

6. DEMO ORGANIZATION

In the demo of the Picasso tool, we will first present the full suite of optimizer diagrams listed in Table 1, highlighting the presence of complex plan diagrams. Then we will demonstrate how anorexic and robust plan replacement diagrams can be generated from these dense diagrams. Finally, we will demonstrate the mechanisms for internalizing these concepts in the core of the query optimizer. The entire demo will be conducted on popular industrial-strength optimizers, using a variety of query templates based on the TPC-H and TPC-DS benchmarks.

Acknowledgements

The Picasso tool has been built at the Indian Institute of Science, Bangalore, through the dedicated efforts of a large group of students, listed on the Picasso website [22]. This work was supported in part by the Dept. of Science and Technology, Govt. of India, and by research grants from Google, IBM, Microsoft and Sybase.

7. REFERENCES

[1] M. Abhirama, S. Bhaumik, A. Dey, H. Shrimal and J. Haritsa, "On the Stability of Plan Costs and the Costs of Plan Stability", *Proc. of 36th Intl. Conf. on Very Large Data Bases (VLDB)*, September 2010.

[2] G. Antonshenkov, "Dynamic Query Optimization in Rdb/VMS", *Proc. of 9th IEEE Intl. Conf. on Data Engineering (ICDE)*, April 1993.

[3] F. Chu, J. Halpern and P. Seshadri, "Least Expected Cost Query Optimization: An Exercise in Utility", *Proc. of ACM Symp. on Principles of Database Systms (PODS)*, May 1999.

[4] F. Chu, J. Halpern and J. Gehrke, "Least Expected Cost Query Optimization: What Can We Expect", *Proc. of ACM Symp. on Principles of Database Systems (PODS)*, May 2002.

[5] A. Deshpande, Z. Ives and V. Raman, "Adaptive Query Processing", *Foundations and Trends in Databases*, Now Publishers, 1(1), 2007.

[6] A. Dey, S. Bhaumik, Harish D. and J. Haritsa, "Efficiently Approximating Query Optimizer Plan Diagrams", *Proc. of 34th Intl. Conf. on Very Large Data Bases (VLDB)*, August 2008.

[7] Harish D., P. Darera and J. Haritsa, "On the Production of Anorexic Plan Diagrams", *Proc. of 33th Intl. Conf. on Very Large Data Bases (VLDB)*, September 2007.

[8] Harish D., P. Darera and J. Haritsa, "Identifying Robust Plans through Plan Diagram Reduction", *Proc. of 34th Intl. Conf. on Very Large Data Bases (VLDB)*, August 2008.

[9] A. Hulgeri and S. Sudarshan, "Parametric Query Optimization for Linear and Piecewise Linear Cost Functions", *Proc. of 28th Intl. Conf. on Very Large Data Bases (VLDB)*, August 2002.

[10] A. Hulgeri and S. Sudarshan, "AniPQO: Almost Non-intrusive Parametric Query Optimization for Nonlinear Cost Functions", *Proc. of 29th Intl. Conf. on Very Large Data Bases (VLDB)*, August 2003.

[11] N. Kabra and D. DeWitt, "Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans", *Proc. of ACM SIGMOD Intl. Conf. on Management of Data*, May 1998.

[12] N. Reddy and J. Haritsa, "Analyzing Plan Diagrams of Database Query Optimizers", *Proc. of 31st Intl. Conf. on Very Large Data Bases (VLDB)*, August 2005.

[13] M. Stillger, G. Lohman, V. Markl and M. Kandil, "LEO, DB2's LEarning Optimizer", *Proc. of 27th Intl. Conf. on Very Large Data Bases (VLDB)*, August 2001.

[14] www.artlex.com/ArtLex/c/cubism.html

[15] www.ibm.com/db2

[16] www.microsoft.com/sqlserver/2008/

[17] www.oracle.com/technology/products/database/oracle11g/

[18] www.sybase.com/linux/ase

[19] www.postgresql.org

[20] www.tpc.org/tpch

[21] www.tpc.org/tpcds

[22] dsl.serc.iisc.ernet.in/projects/PICASSO/