Karlsruhe Institute of Technology

Analyzing Plan Diagrams of Data Query Optimizers

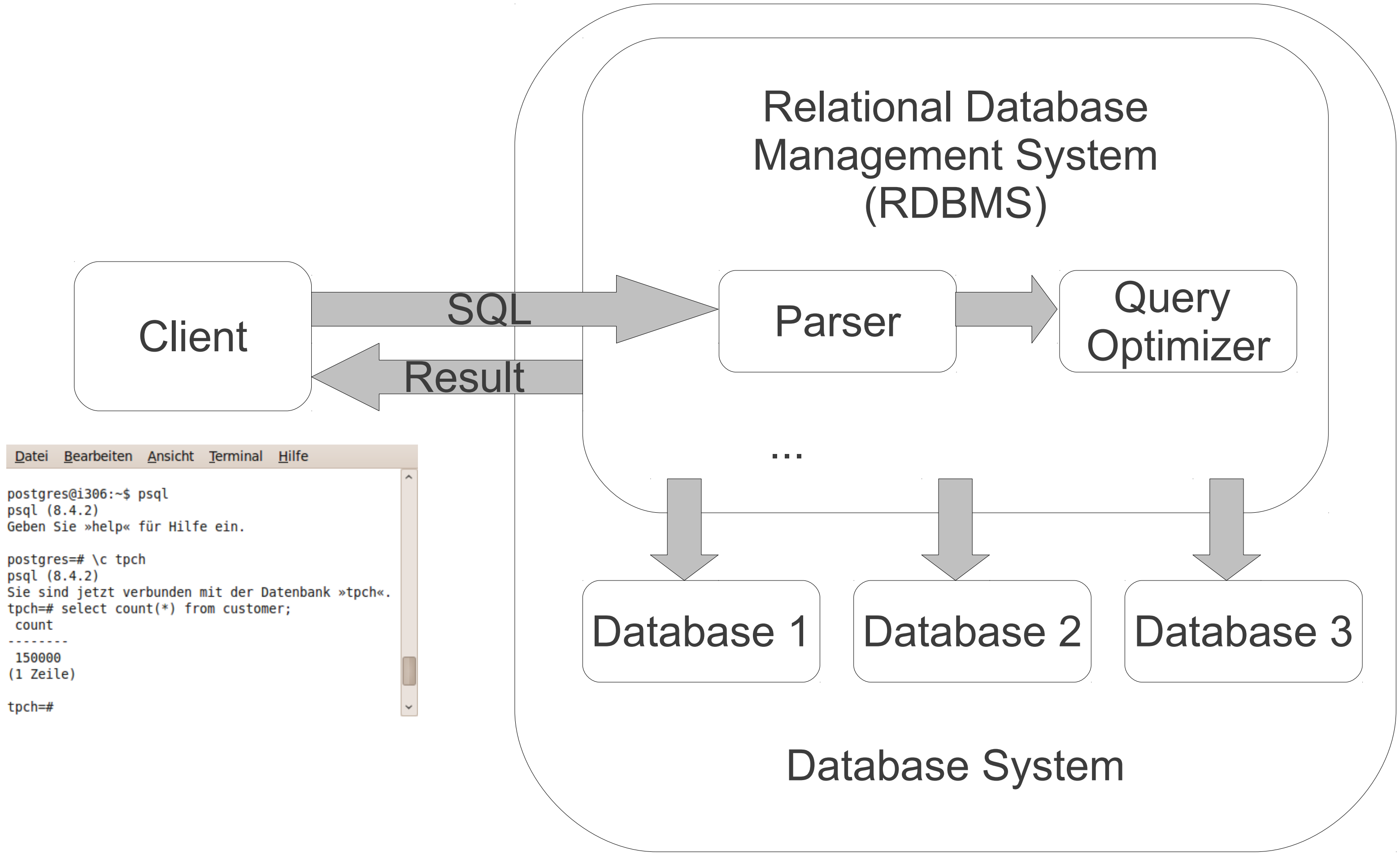
INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION (IPD), FAKULTY OF INFORMATICS

Paper

- Analyzing Plan Diagrams of Database Query Optimizers
- written in 2005
- Naveen Reddy, Jayant R. Haritsa
- Database Systems Lab, SERC/CSA
- Indian Institute of Science, Bangalore 560012, INDIA

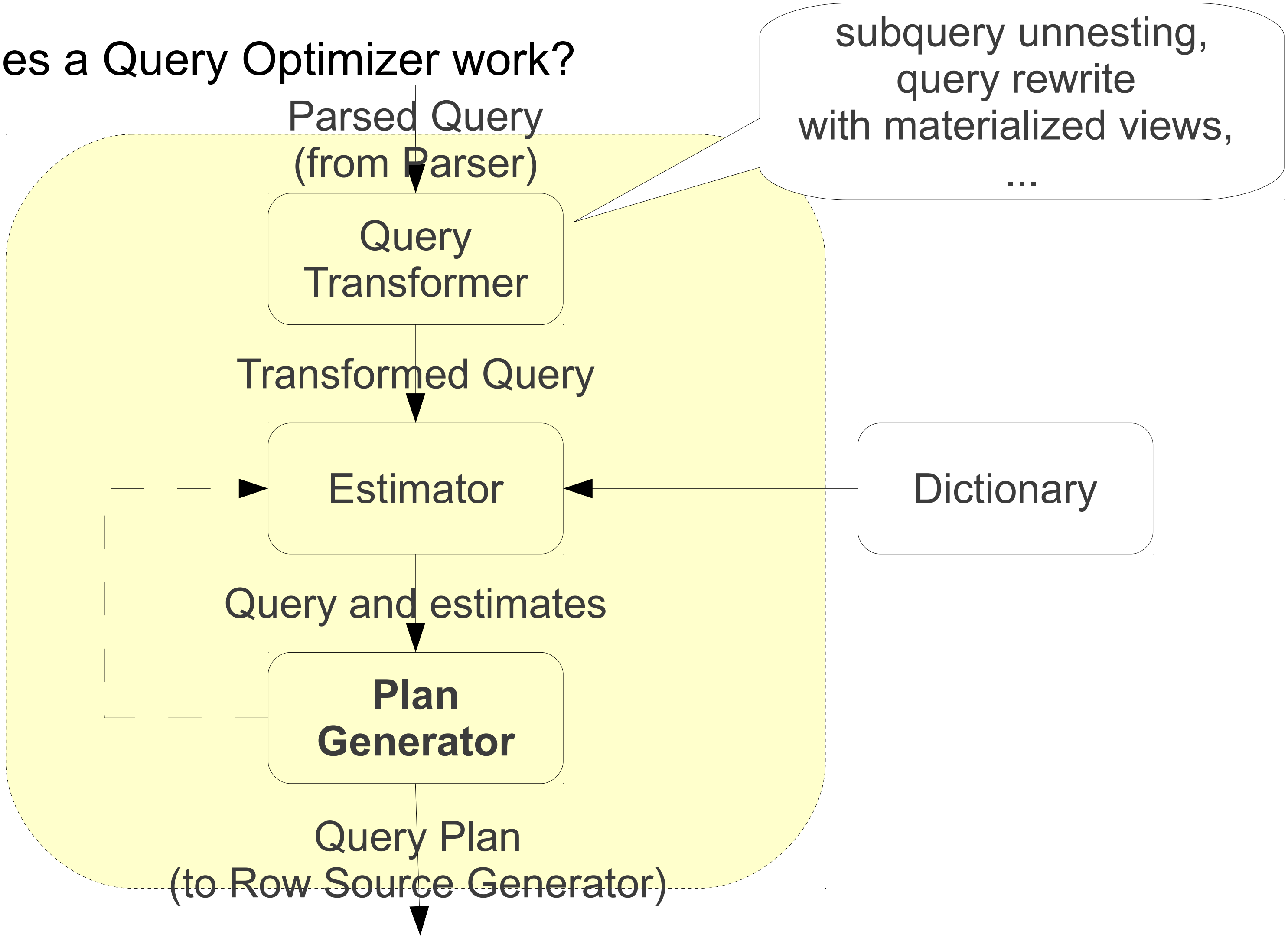
Relational Database Management System

- Where is a Query Optimizer located?



Query Optimizer

- How does a Query Optimizer work?



Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

Query Optimizer

- Identifies the most efficent strategy to execute SQL queries
- Returns query plan
- Part of the RDBMS
- Based on estimated number of rows of relevant relations
- Relevant join conditions are given in the query

Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

Query Plan

- A strategy to execute SQL queries
- Query Plan does not affect the result
- Example: (id is FK of Aid)
- select o,p from A inner join B on (A.id=B.Aid) where B.p=0;
- Plan P1: AB
  - A=3, B=6/3=2 => cost(AB)=6
- Plan P2: BA
  - B=3 (using where), A=1 (primary key) => cost(BA)=3

A

id	o
1	X
2	Y
3	Z

B

id	Aid	p
1	1	0
2	1	1
3	2	2
4	2	0
5	3	0
6	3	1

How to analyze a query optimizer?

1. Send SQL statement EXPLAIN SELECT … FROM … WHERE …
2. Optimizer returns query plan
3. This can be repeated by varying the where condition(s) systematically until the selectivity space is covered
4. Corresponding plan and cost for each set of conditions can be visualized

Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

Query plan costs

A

id	o
1	X
2	Y
3	Z

B

id	Aid	p
1	1	0
2	1	1
3	2	2
4	2	0
5	3	0
6	3	1

```
test=# explain select o,p
      from A inner join B on (A.id=B.Aid)
      where B.p=0;

                         QUERY PLAN
-----------------------------------------------------------
 Hash Join  (cost=1.07..2.18 rows=3 width=6)
   Hash Cond: (b.aid = a.id)
   ->  Seq Scan on b  (cost=0.00..1.07 rows=3 width=8)
         Filter: (p = 0)
   ->  Hash  (cost=1.03..1.03 rows=3 width=6)
         ->  Seq Scan on a  (cost=0.00..1.03 rows=3 width=6)
```


Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

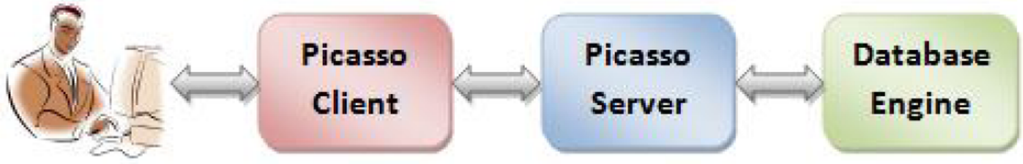
TPC-H

- Decision support benchmark
  - database schema, data
  - queries for testing (Q1 through Q22)
- We use same data set as the authors of the paper:

relation	#rows
part.tbl	200000
nation.tbl	25
customer.tbl	150000
region.tbl	5
partsupp.tbl	800000
supplier.tbl	10000
orders.tbl	1500000
lineitem.tbl	6001215

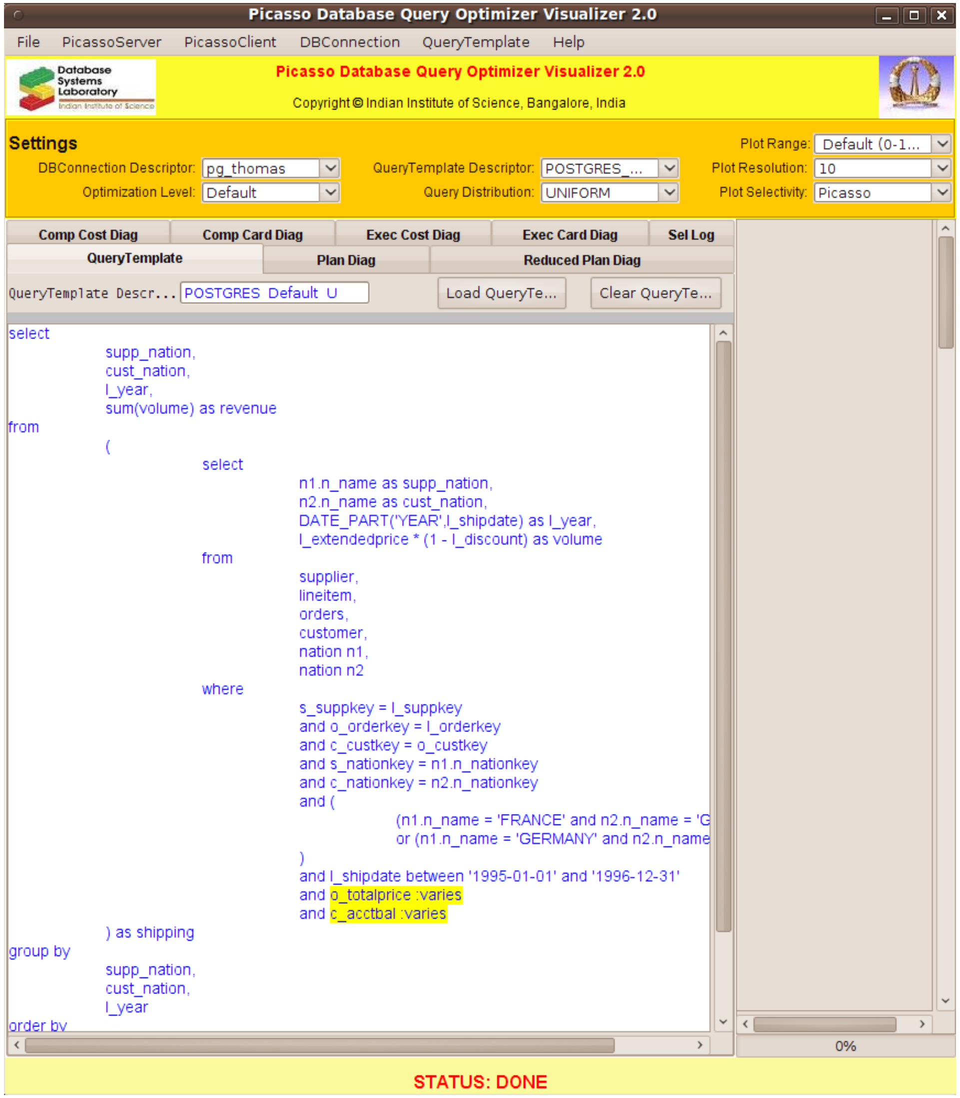
Picasso

- Picasso
  - Sends explain queries against the database
  - Stores and visualizes query plan and costs
- Picasso Server
  - Connects to Database (SQL-Server, Oracle, Sybase, postgresql, ...)
  - Multiuser
- Picasso Client
  - Connects to PICASSO Server
  - GUI for PICASSO Server



Run test with Picasso

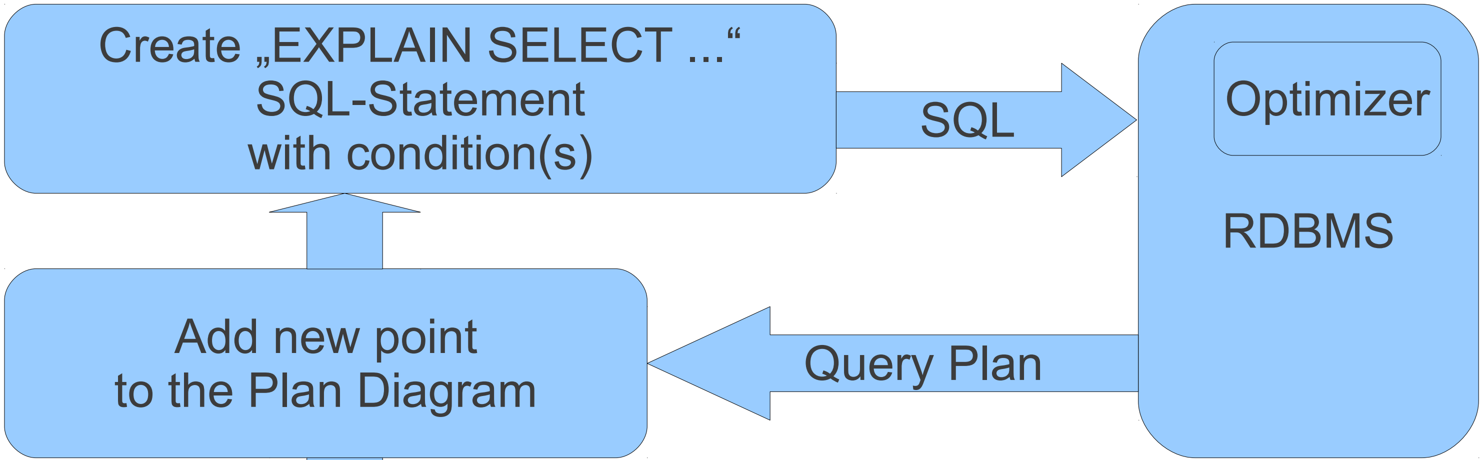
Karlsruhe Institute of Technology

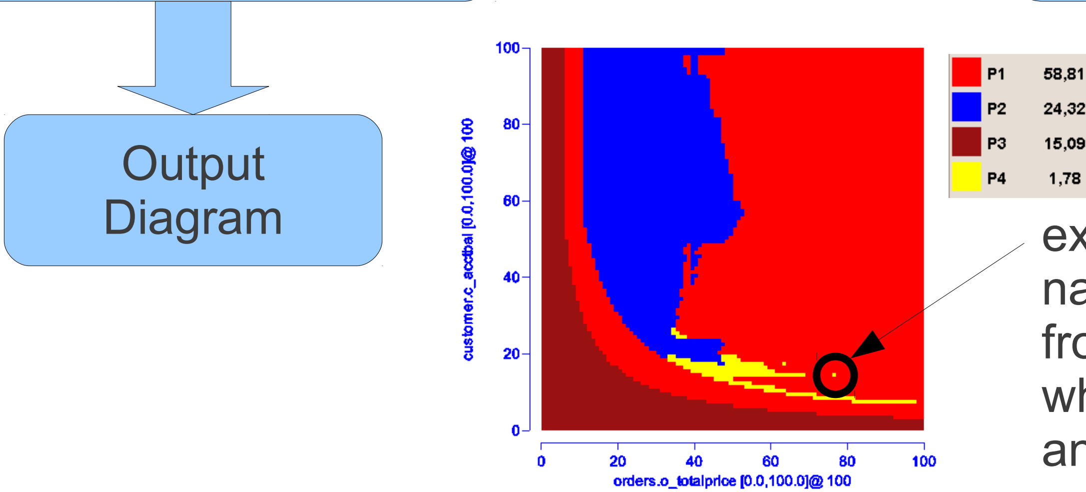


Run test with Picasso

- Picasso varies WHERE conditions according to plot resolution:
  - **and o_totalprice :varies**
  - **and c_acctbal :varies**
- 2 varying conditions ⇒ 2-dimensional selectivity space (2D)
- Picasso runs EXPLAIN-Statements to determine the query plan chosen by the optimizer.
- 2D and Resolution=100 ⇒ 100^2 = 10,000 EXPLAIN-Statements

Diagram creation process

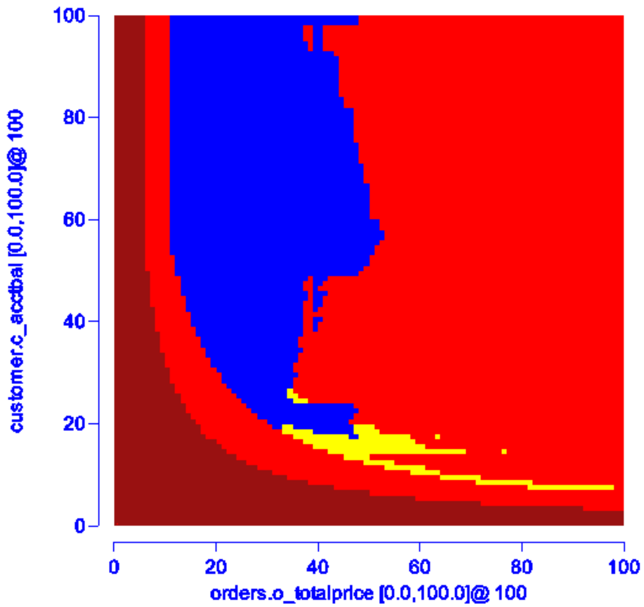


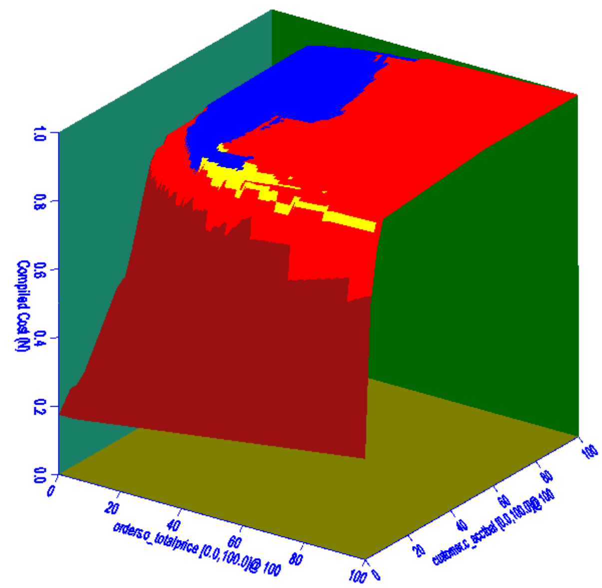


explain select
name, price
from customer, order
where […]
and o_totalprice <= 98679
and c_acctbal <= 5809;
result: P4

Institut für Programmstrukturen und
Datenorganisation (IPD), Fakultät für Informatik

Plan Diagram and Cost Diagram





	P1	58,81
	P2	24,32
	P3	15,09
	P4	1,78

Plan

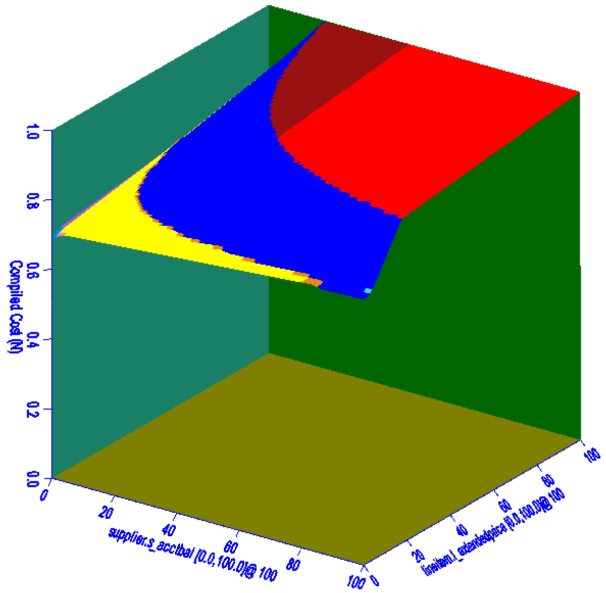
Coverage of selectivity space in %

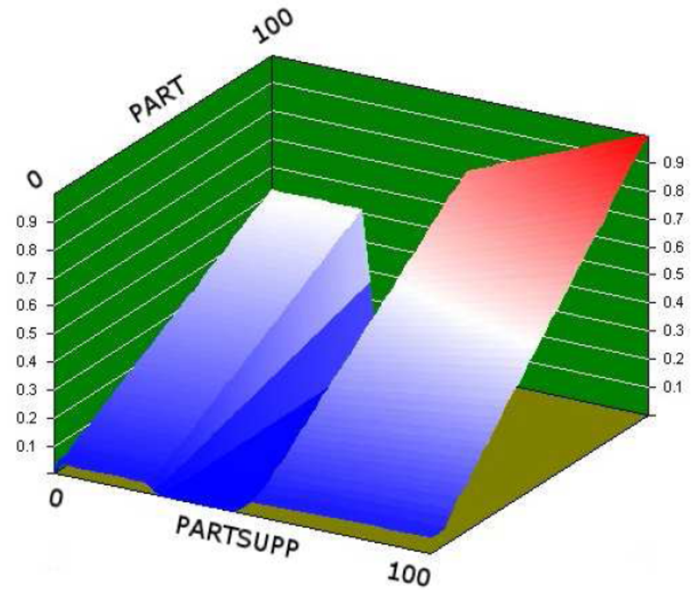
Plan Diagram:
2D-visualization of execution plans

Cost Diagram:
3D-visualization of **estimated** costs. Estimation is done by optimizer.

Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

Cost domination Principle





Monotonic cost behaviour:
Increasing selectivity space increases estimated costs

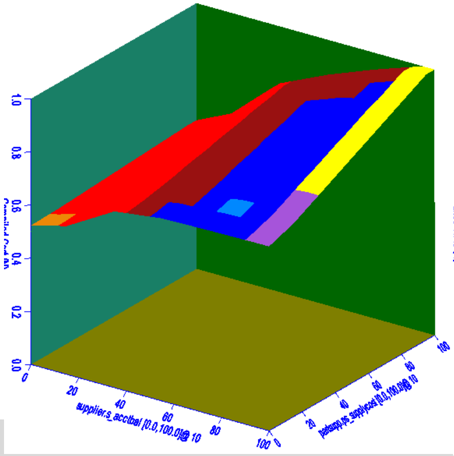
Non-monotonic cost behaviour:
Increasing selectivity space does not always increase estimated costs

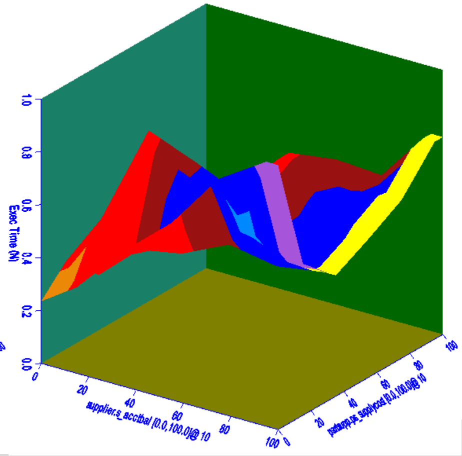
Cost domination principle:
When increasing selectivity space, estimated costs must increase monotonically.

Estimated costs vs. execution costs

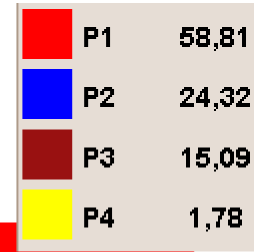
	Estimated costs	Execution costs
SQL Statement for each point in diagram	explain select ...	select ...
Measurement	#rows, ...	execution time
Creation time for a 10x10 diagram	10s	4h 10min
Monotonicity	Estimated cost diagram should always be monotonic	Execution cost diagram might not be monotonic, e.g. if estimations were not precise

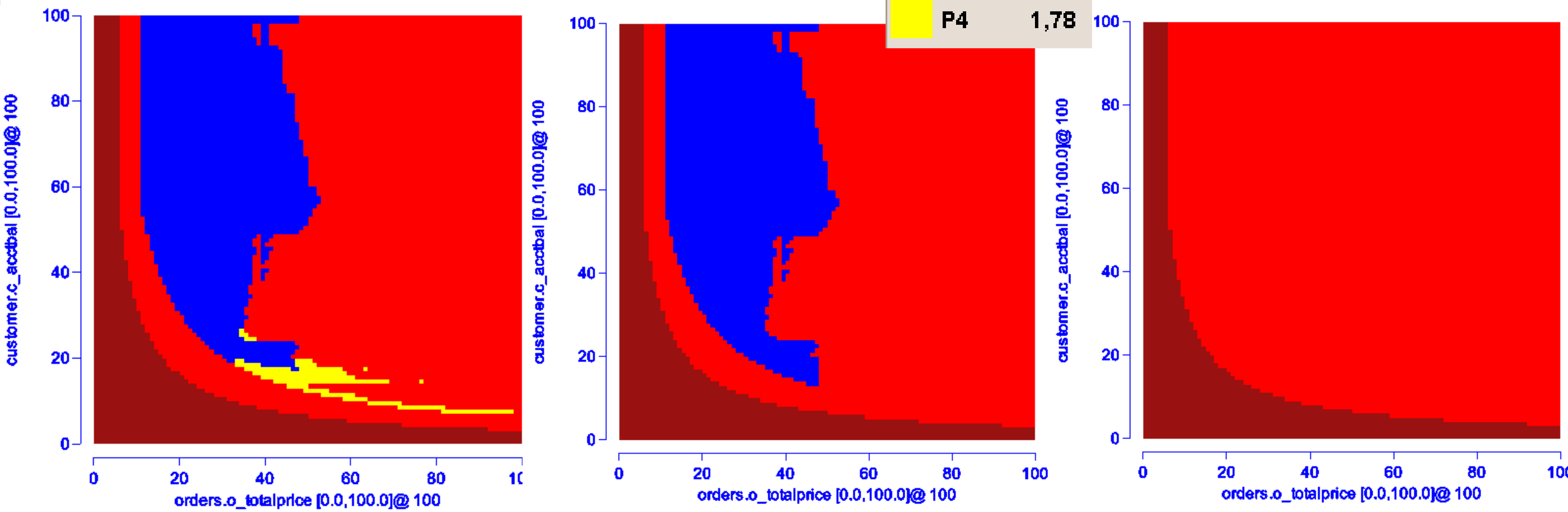
=> Even if we have Monotonic estimated cost behaviour, we want to reduce #Plans and #Segments.





Reduced Plan Diagram





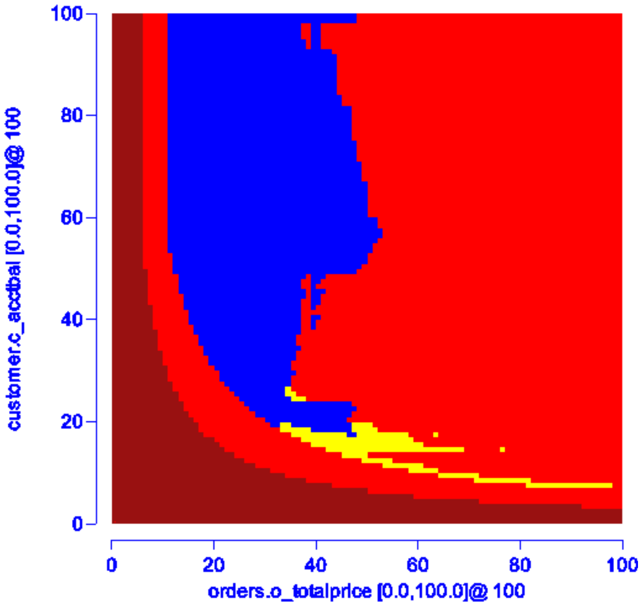
Plan Diagram

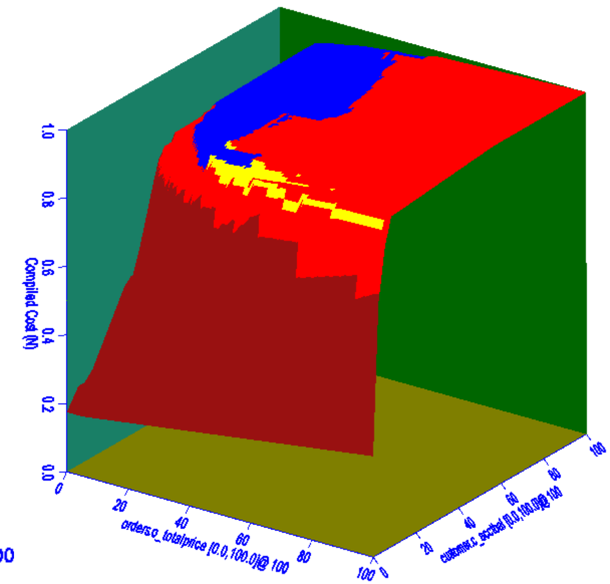
Reduced Plan Diagram
Maximum allowed Cost Increase:
5% 10%

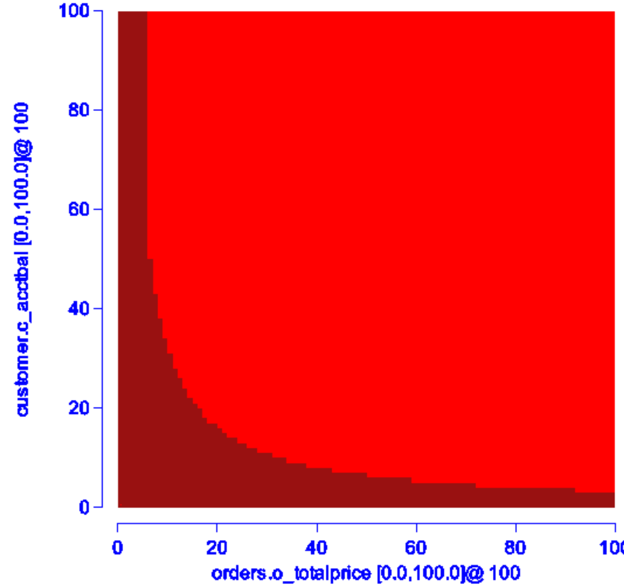
Reduced Plan Diagram:
Try to reduce #Plans without increasing **estimated** costs more than X% **in each** point
(X% is called Costgreedy Reduction or plan optimality tolerance threshold.)

TPC-H Query 7

	P1	58,81
	P2	24,32
	P3	15,09
	P4	1,78





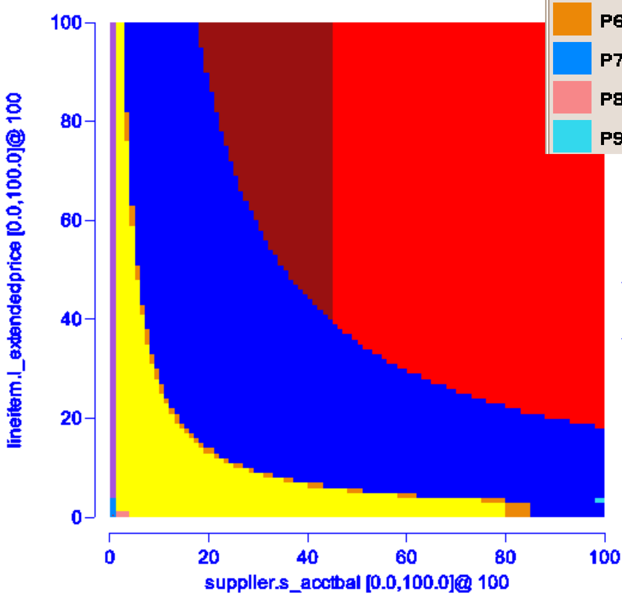


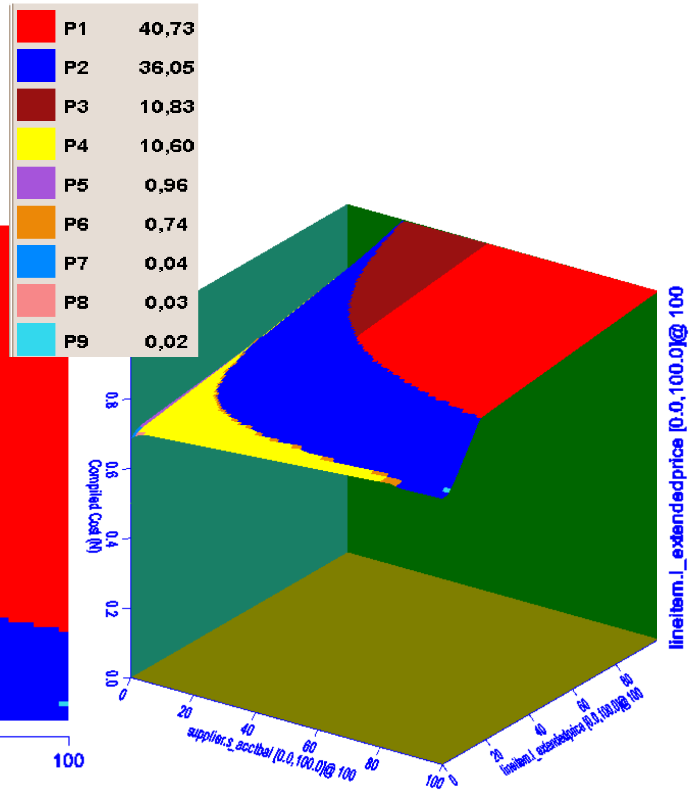
Plan Diagram
- 4 Plans
- P4 covers 1,78% only
- Many small segments
- Not smooth

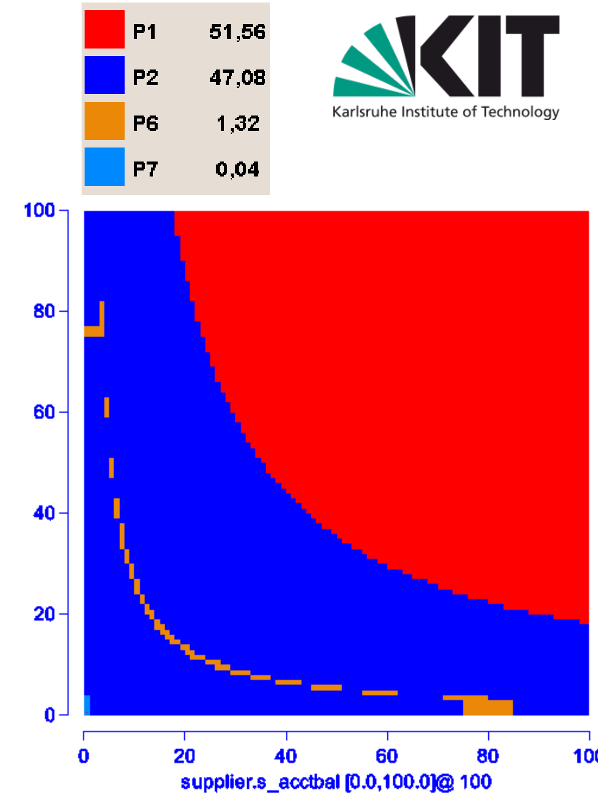
Cost Diagram
- monotonic

Reduced Plan Diagram
- 2 Plans only
- Max Increase ≤ 6.73%
- Avg Increase **≤ 0.72%**

TPC-H Query 8







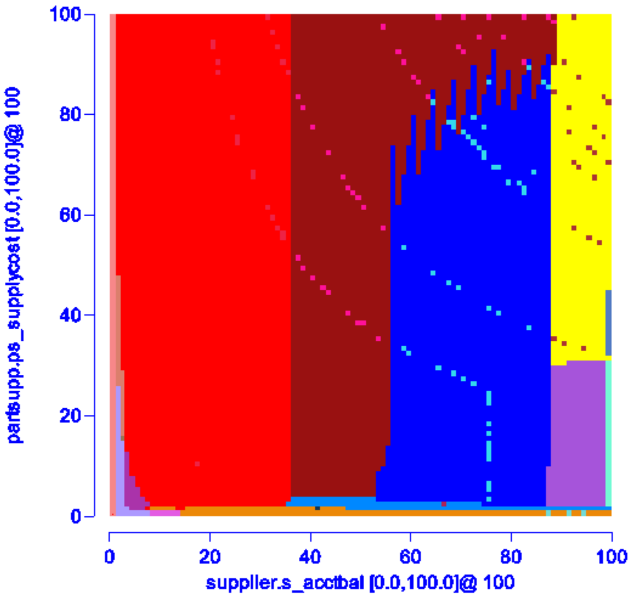
Karlsruhe Institute of Technology

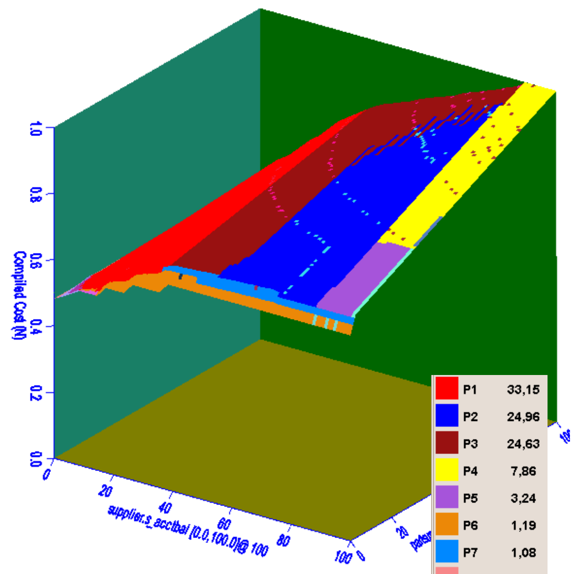
Plan Diagram
- 9 Plans
- 80% covered by 3 Plans
- Many small segments
- Not smooth

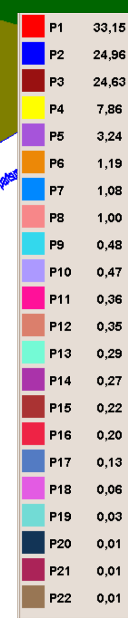
Cost Diagram
- monotonic

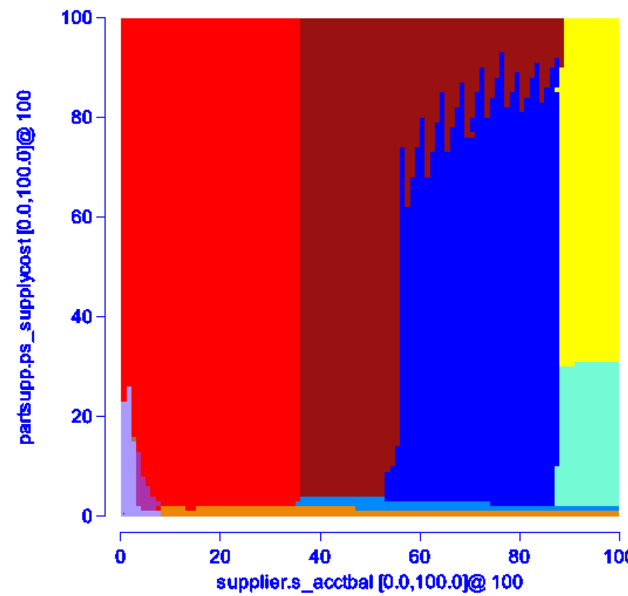
Reduced Plan Diagram
- 4 Plans only
- Max Increase ≤ 9.8%
- Avg Increase **≤ 0.41%**

TPC-H Query 9









Plan Diagram

- 22 Plans
- 80% covered by 3 Plans
- Many small segments
- Not smooth
- Highly instable

Cost Diagram

- monotonic

Reduced Plan Diagram

- 10 Plans
- Max Increase ≤ 9.15%
- Avg Increase **≤ 0.15%**

Clarifications

- #Plans does not necessarily correspond with execution time of one query. We do **not** rate performance!
- But: We consider the **stability** of an optimizer.

Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

Hypothesis

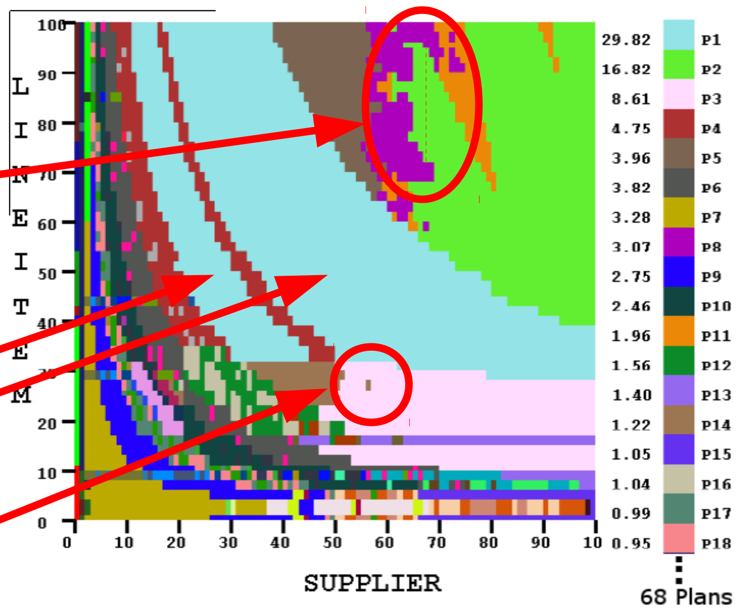
- Modern query optimizers make extremely fine-grained plan choices
- smaller plans occupy less than 1% of selectivity space
- Optimizers are over-sophisticated
- Processing overheads with query optimization could be lowered

Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik

Parametric query optimization (PQO)

- PQO
  - apriori identify the optimal set of plans at compile time
  - at run time, use plan corresponding to selectivity parameter(s).
- PQO Assumptions:
- Plan Convexity
  - If Plan P is optimal at point A and at point B, it is also optimal at all points joining the two points
- Plan Uniqueness
  - An optimal plan P appears at only one contiguous region in the entire space
- Plan Homogenity
  - An optimal plan P is optimal within the entire region enclosed by its plan boundaries.

But: today's optimizers can look like that:

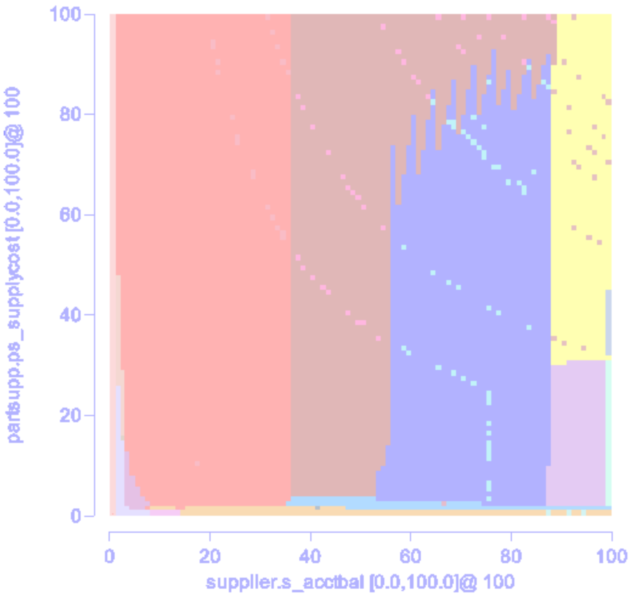


Better: reduce #plans for PQO

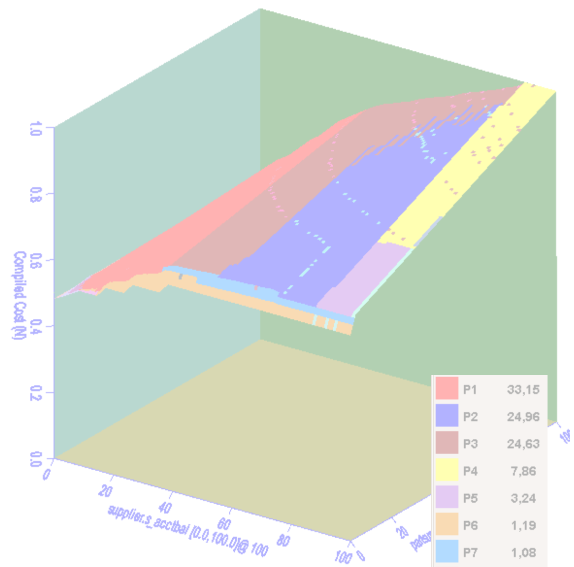
Conclusion

- Often highly intricate diagrams
- Typically 80% of space covered by 20% of plans (postgresql: 37%)
- Cardinality of plan diagram can be reduced, without materially affecting the query cost
- Assumptions of parametric query optimization literature do not hold in practice. (Reduction needed)
- Needed:
  - Mechanism for pruning the plan search space
  - Directly reduce #plans by optimizer

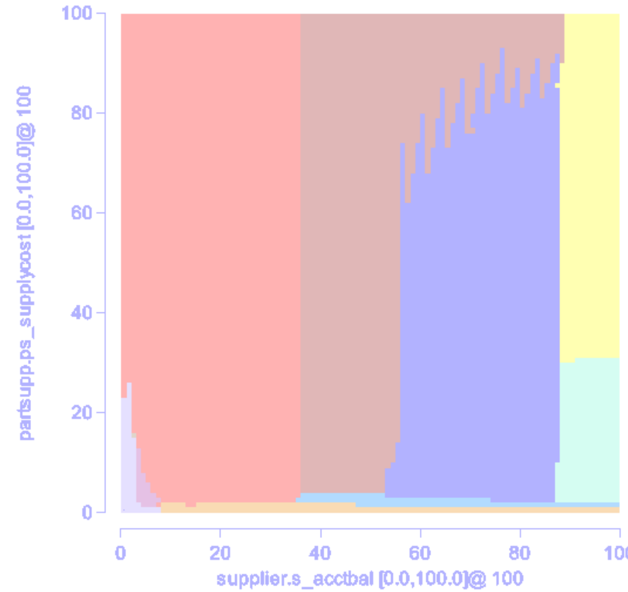
Discussion



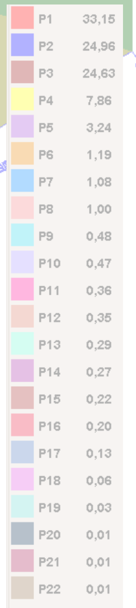
Plan Diagram



Cost Diagram



Reduced Plan Diagram



Do you have any questions?

?

Comparison of Database Analyzers

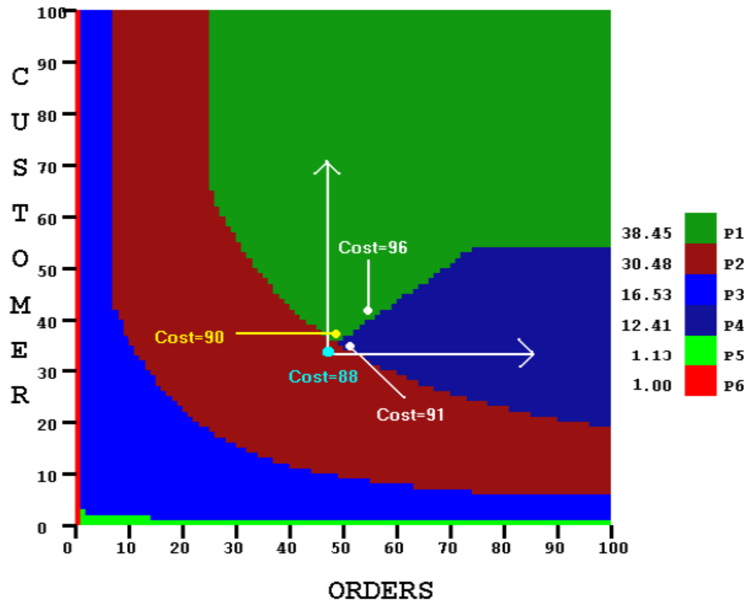
TPC-H Query	#Plans	%plans for 80%	Gini Index
2	2	100,00%	0,47
5	8	37,50%	0,65
7	4	50,00%	0,57
8	9	33,33%	0,68
9	22	13,64%	0,76
10	9	22,22%	0,48
18	7	14,29%	0,24
21	4	25,00%	0,28
postgresql	**8,13**	**37,00%**	**0,52**

OptA	28,7	17,00%	0,79
OptB	24,5	23,00%	0,72
OptC	28,8	16,00%	0,8

Measurement of statistical dispersion

- OptX: Commercial DBs tested by IIS
  - postgresql: Same dataset, similar reference system
- postgresql uses smaller #Plans to cover space.
- postgresql needs higher percentage of plans to cover 80% of selectivity space.
- postgresql's Gini Index is close to 0.5 which is better than 0.7 or 0.8.

Generation of Reduced Plan Diagram

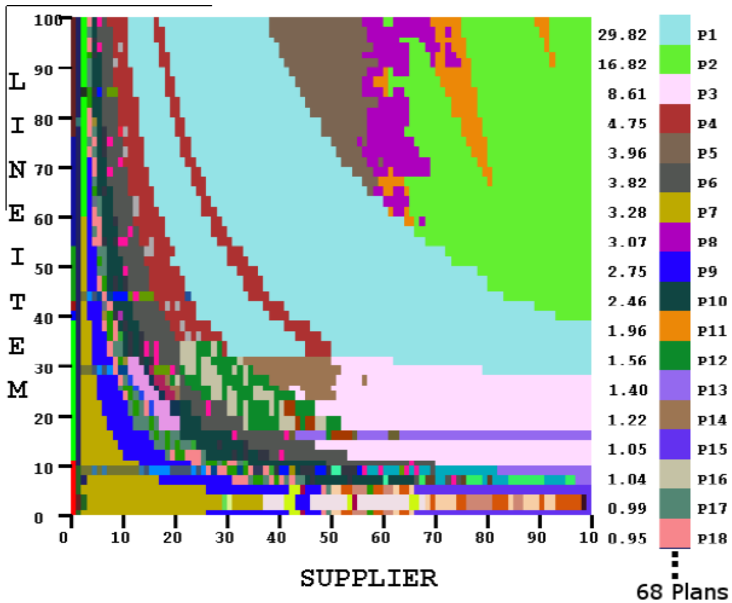


- P1, P4 are upper bounds of P2
- We move point to P1 which has lowest cost (90)
- If all Points of a plan are swit

Query plan costs

- The order of the operations in a query plan is important:
- cost(AB) != cost(BA)
- cost(ABC) != cost(A) * cost(BC)
- cost(ABC) != cost(AB) * cost(C)

Complex patterns



- Rapidly alternating choices
- not created on postgresql

Institut für Programmstrukturen und Datenorganisation (IPD), Fakultät für Informatik