Diplomarbeitspräsentationen der Fakultät für Informatik

On Extending PostgreSQL with the Skyline Operator

Hannes Eder

Masterstudium
Computational Intelligence

Technische Universität Wien
Institut für Informationssysteme
Database and Artificial Intelligence Group
Betreuer: Univ.-Prof. Mag.rer.nat. Dr.techn. Reinhard Pichler
Univ.Ass. Dr.rer.nat. Fang Wei

Motivation and Goal

In the realm of multi-criteria optimization designing an appropriate objective function is a challenging task from a user's perspective. The *skyline operator* filters out the *interesting* tuples from a potentially large dataset. A tuple belongs to the *skyline* if it is *not dominated by* any other tuple, i.e. there is no tuple which is at least as good as in all and better in at least one criteria. No matter how we weight our preferences along the attributes, only those tuples which score best under a *monotone scoring function* are part of the skyline. In other words, the skyline does not contain tuples which are nobody's favorite. The notion of skyline is also called *Pareto optimal set* and its computation *maximum vector problem*.

In this thesis we aim at:

- extending PostgreSQL with the skyline operator,
- evaluating skyline algorithms in the RDBMS context, and
- building a skyline query optimizer to automatically generate a good query plan w.r.t. I/O, time, and memory consumption.

Formal Problem Definition

In a skyline query with the following skyline clause

$$\texttt{SKYLINE OF } a_1 \texttt{ MIN}, \ldots, a_k \texttt{ MIN}, a_{k+1} \texttt{ MAX}, \ldots, a_l \texttt{ MAX}, a_{l+1} \texttt{ DIFF}, \ldots, a_m \texttt{ DIFF}$$

a tuple r

$$r = (\underbrace{r_1, \ldots, r_k}_{\texttt{MIN}}, \underbrace{r_{k+1}, \ldots, r_l}_{\texttt{MAX}}, \underbrace{r_{l+1}, \ldots, r_m}_{\texttt{DIFF}}, \underbrace{r_{m+1}, \ldots, r_n}_{extra})$$

dominates a tuples s

$$s = (\underbrace{s_1, \ldots, s_k}_{\texttt{MIN}}, \underbrace{s_{k+1}, \ldots, s_l}_{\texttt{MAX}}, \underbrace{s_{l+1}, \ldots, s_m}_{\texttt{DIFF}}, \underbrace{s_{m+1}, \ldots, s_n}_{extra})$$

iff

$$C(r,s) = \overbrace{\left(\bigwedge_{1 \le i \le k} r_i \le s_i\right)}^{\text{“MIN at least as good”}} \wedge \overbrace{\left(\bigwedge_{k+1 \le i \le l} r_i \ge s_i\right)}^{\text{“MAX at least as good”}} \wedge \overbrace{\left(\bigwedge_{l+1 \le i \le m} r_i = s_i\right)}^{\text{“DIFF equal”}} \wedge \left(\left(\underbrace{\bigvee_{1 \le i \le k} r_i < s_i}_{\text{“MIN better than”}}\right) \vee \left(\underbrace{\bigvee_{k+1 \le i \le l} r_i > s_i}_{\text{“MAX better than”}}\right)\right).$$

We define the skyline of a relation **R** as

$$\text{skyline}_{\triangleright}(\mathbf{R}) := \{r \in \mathbf{R} \mid \nexists s \in \mathbf{R} \colon s \triangleright r\}$$

with

$$r \triangleright s \quad \text{iff} \quad C(r,s).$$

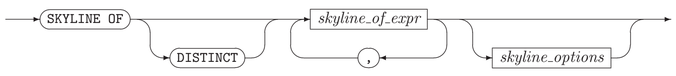
Query Syntax

We extend the syntax from Börzsönyi et. al. [2001]

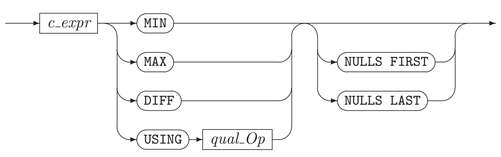
```
SELECT ... FROM ... WHERE ...
GROUP BY ... HAVING ...
SKYLINE OF [DISTINCT] a_1 [MIN|MAX|DIFF], ... ,
  a_m [MIN|MAX|DIFF]
ORDER BY ...
```

in the following way:

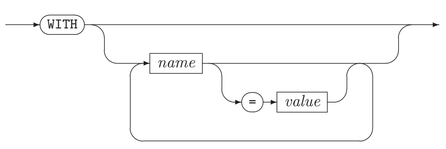
skyline_clause



skyline_of_expr

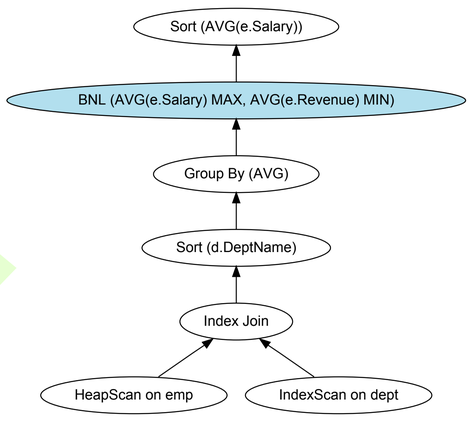


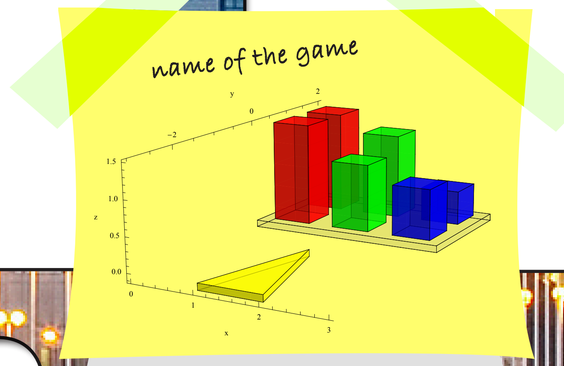
skyline_options



Typical Query-Plan

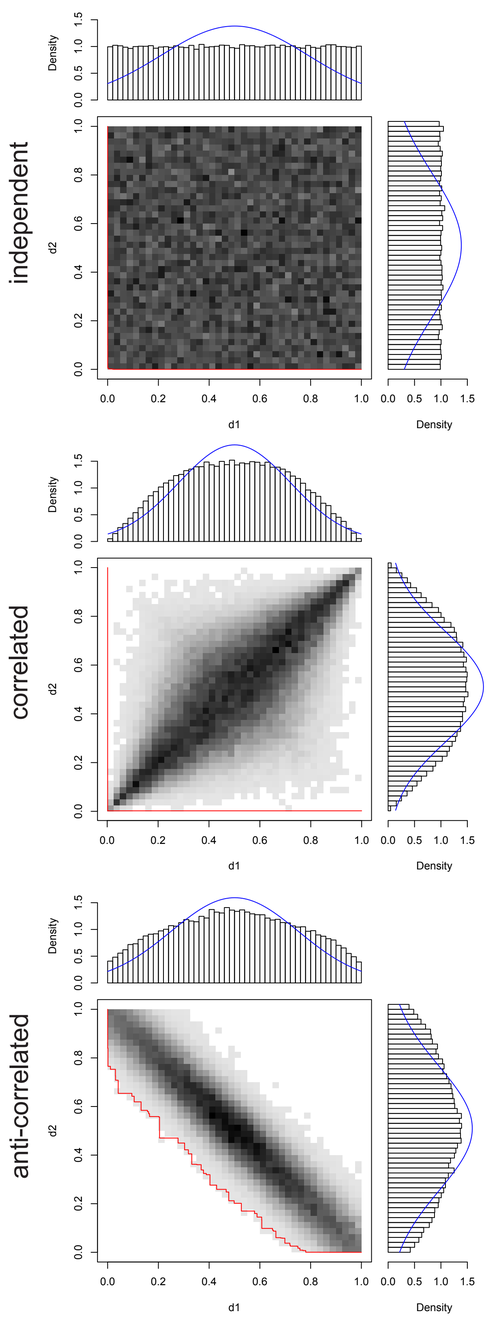
```
SELECT d.DeptName, AVG(e.Salary), AVG(e.Revenue)
FROM emp e JOIN dept d on (e.Dno = d.DeptId)
GROUP BY d.DeptName
SKYLINE OF AVG(e.Salary) MAX, AVG(e.Revenue) MIN
ORDER BY AVG(e.Salary) DESC
```



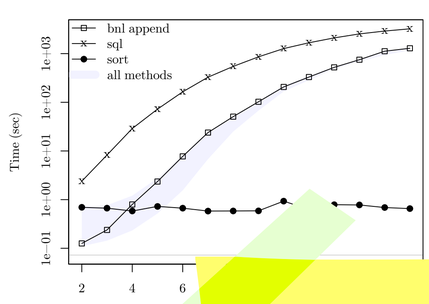


Experimental Results

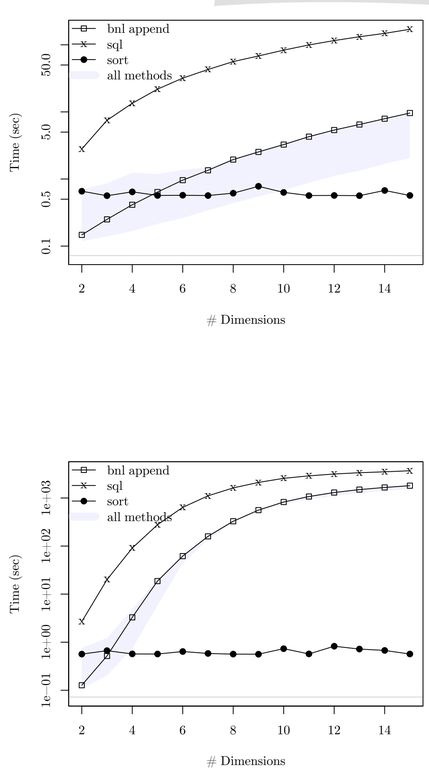
Data Distribution



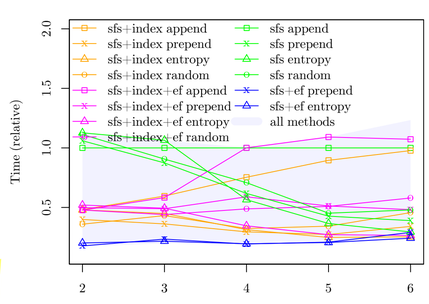
Absolute Timing (100k Tuples)



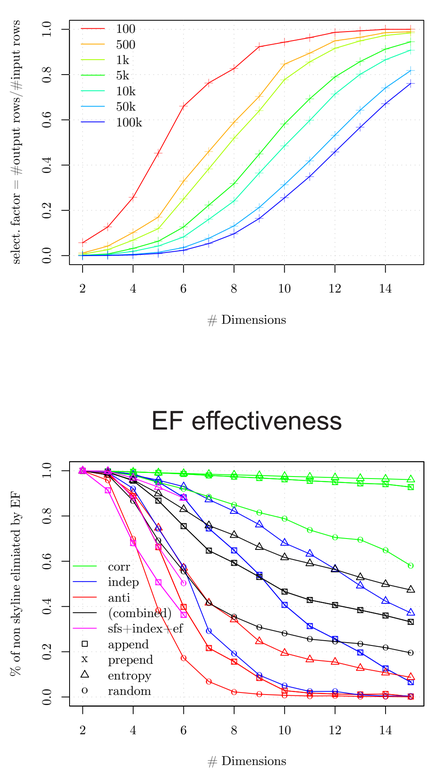
point out speedup!



Relative Timing w/ index (100k tuples)



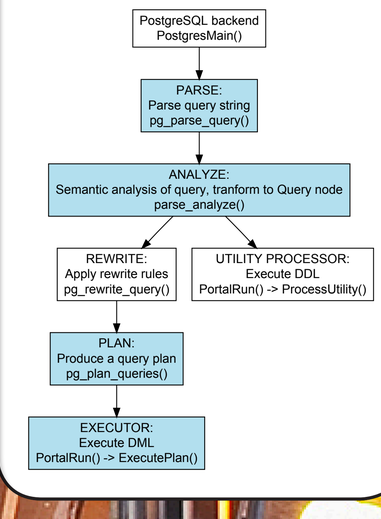
Selectivity Factor (independent dataset)



EF effectiveness

Implementation

In this architecture diagram of PostgreSQL the light blue background indicates the components we have modified to support skyline queries.



Take Home Message

We provide:

- an open source implementation of BNL and SFS in PostgreSQL
- a proof for a flaw in BNL and a fixed version
- a noteworthy experimental environment
- extensive experimental results which are beneficial for developing heuristics for skyline query optimization
- a website with all results, source code, and a web-interface to test drive our implementation:

website: http://skyline.dbai.tuwien.ac.at/
contact: hannes@hanneseder.net