Hand-out

Advanced database technology

March 6, 2006

A mystery. Suppose you are the database administrator of a large company. One day your boss comes to you complaining that the following query takes several hours to run, even though the end result is quite small.

```
SELECT Sales.amount, Events.type
FROM Sales, Events, Goods, Suppliers
WHERE Sales.date=Events.date
       AND Sales.partno=Goods.partno AND Suppliers.sid=Goods.sid
       AND Goods.category='engine' AND Suppliers.country='DK'
```

The query plan looks reasonable:

$$(\sigma_{\text{category='engine'}}(\text{Goods}) \bowtie \sigma_{\text{country='DK'}}(\text{Suppliers}) \bowtie (\text{Sales} \bowtie \text{Events})$$

What do you do? Propose queries on the relations that could help shed light on what the problem is? (Feedback on proposals, tests, etc. from teacher in class.) Propose possible cures.

Query optimization. Suppose we have relations R_1, R_2 and R_3 with common attributes A (appearing in R_1 and R_2) and B (appearing in R_2 and R_3). Tuples in all relations have fixed length, 5 tuples per disk block, and attributes A and B occupy 10% of the total tuple length. The relations occupy $B(R_1) = 8.000$, $B(R_2) = 10.000$ and $B(R_3) = 90.000$ blocks on disk, respectively. Consider the relational algebra expression:

$$\delta(\pi_A((\sigma_{A>200}(R_1)) \bowtie (\sigma_{B=4}(R_3)) \bowtie (\sigma_{A\leq 450}(R_2)))).$$

Using advanced statistics, our query optimizer comes up with the following estimates for the number of tuples in each subexpression:

- $|\sigma_{A>200}(R_1)| \approx 10.000$
- $|\sigma_{B=4}(R_3)| \approx 2.000$
- $|\sigma_{A\leq 450}(R_2)| \approx 5.000$
- $|(\sigma_{A>200}(R_1)) \bowtie (\sigma_{B=4}(R_3))| \approx 20.000.000$
- $|(\sigma_{A>200}(R_1)) \bowtie (\sigma_{A\leq 450}(R_2))| \approx 100.000$
- $|(\sigma_{B=4}(R_3)) \bowtie (\sigma_{A\leq 450}(R_2))| \approx 80.000$
- $|(\sigma_{A>200}(R_1)) \bowtie (\sigma_{B=4}(R_3)) \bowtie (\sigma_{A\leq 450}(R_2))| \approx 50.000$

Using these estimates, apply dynamic programming (Selinger-Style Optimization, see GUW page 845) to find the best physical query plan *not* using any indexes:

- Determine the order of joins.

- Determine the algorithms used for all operations. Assume that there is memory for either a two-pass sorting based join using $5(B(R_i) + B(R_j))$ I/Os to join R_i and R_j, or a two-pass hash join using $3(B(R_i) + B(R_j))$ I/Os to join R_i and R_j.
- Determine where to use pipelining. Assume that there are 10 extra memory buffers available for pipelining purposes.

1 Hand-in for March 20

To be handed in no later than March 20 (in connection with the lecture on that day):

- Exam June 2005, Problem 3.
- Exam June 2004, question 4.c (you may assume that answers to 4.a and 4.b are known).