CHAIR OF APPLIED COMPUTER SCIENCE III
UNIVERSITY OF MANNHEIM
Prof. Dr. Guido Moerkotte
Email: moerkotte@uni-mannheim.de

Query Optimization Exercise sheet 3

Exercise 1

Given the following schema:
$R_1(a, c, d, e)$
$R_2(a, b)$
$R_3(b, c)$
$R_4(d)$
$R_5(e)$

Determine and classify the query graph induced by each of the following queries. If we don't consider cross products, then how big is the search space for left-deep join trees and busy join trees? Are there deterministic algorithms that find the optimal join order in this query graph's search space in polynomial time?

Exercise 1 a)

```
SELECT DISTINCT *
FROM R1, R2, R3, R4
WHERE R1.c = R3.c
  AND R2.b = R3.b
  AND R1.d = R4.d;
```

Solution

Query Graph: chain
Number of join trees when cross products are not considered:
left-deep: 2^{n-1}
bushy: $2^{n-1}C(n-1)$
Complexity: We can solve this problem in a polynomial number of steps.

Exercise 1 b)

```
SELECT DISTINCT *
FROM R1, R2, R3, R5
WHERE R1.a = R2.a
  AND R1.c = R3.c
  AND R1.e = R5.e
  AND R2.b = R3.b;
```

Solution

Query Graph: cyclic
Number of join trees when cross products are not considered:
Im schlimmsten Fall eine clique, also:
left-deep: $n!$
bushy: $n! \cdot C(n-1)$
Complexity: NP-hard problem, we should not hope for an algorithm in P.

Exercise 1 c)

```
SELECT DISTINCT *
FROM R1, R2, R3, R5
WHERE R1.a = R2.a
  AND R1.c = R3.c
  AND R1.d = R4.d
  AND R1.e = R5.e;
```

Solution

Query Graph: star
Number of join trees when cross products are not considered:
left-deep: $2 \cdot (n-1)!$;
bushy: $2^{n-1} \cdot (n-1)!$
Complexity: Complexity depends on the cost function(s) used. For a cost function that fullfils the ASI property, we can find the optimal join order in a polynomial number of steps. If we consider two join methods, and therefore need two cost functions, the problem is NP-hard and we should not hope for an algorithm in P. In other cases, the complexity is not known.

Exercise 2

Exercise 2 a)

Implement `GOO`. You may use the `QueryGraph` and `Tree` classes provided in the solution code.

Exercise 2 b)

In *Introduction to the Design & Analysis of Algorithms* the author Anany Levitin desribes a greedy algorithm to compute Huffman encodings. Read chapter 9.4 Huffman Trees and Codes. Observe the similarity between the described algorithm and `GOO`. Adjust of your `GOO` implementation so that it can create Huffman trees for appropiate

inputs. Note that it is easy to deduce the corresponding Huffman encoding from a Huffman tree.
Try to read it up in the book.
If you have no access to the book, you'll find a description at
`https://en.wikipedia.org/wiki/Huffman_coding#Compression`

Exercise 3

For relation R_1 and R_2 you are given their cardinality $|R_i|$, the number of distinct values in an attribute A, denoted by $d(R_i.A)$, the mininimum and maximum value $min(R_i.A)$ and $max(R_i.A)$, and key/ foreign-key information.

Note: You will find the answers, as given in the solution, in the paper:
`https://www2.cs.duke.edu/courses/compsci516/cps216/spring03/papers/selinger-etal-197`
`pdf`
This paper is the first published article on (cost-based) query optimization!

Exercise 3 a)

How could you estimate the selectivity of the predicate $\sigma_{R_1.A=c}$, where c is a constant value? Under what assumptions is your estimate accurate?

Solution

$\hat{s}(R_1.A = c) = 1/d(R_1.A)$
Accurate if $R_1.A$ follows a uniform distribution. Note: We denote the selectivity of a predicate p by $s(p)$ and use $\hat(s)(p)$ for an estimate.

Exercise 3 b)

How could you estimate the selectivity of the predicate $\sigma_{R_1.A>c}$, where c is a constant value? Under what assumptions is your estimate accurate?

Solution

$\hat{s}(R_1.A > c) = \frac{max_{R_1.A} - c}{max_{R_1.A} - min_{R_1.A}}$
Accurate if $R_1.A$ follows a uniform distribution and the values are uniformly spread.

Exercise 3 c)

How could you estimate the selectivity of the join predicate $R_1.A = R_2.B$? Under what assumptions is your estimate accurate?

Solution

$\hat{s}(R_1.A = R_2.B) = 1/max(d(R_1.A), d(R_2.B))$
Assumes that the join is a key-/forign-key-join.

Note that for key-/foreign-key-joins we have that $A \bowtie B = B$ where B is the foreign-key table.

Exercise 3 d)

How could you estimate the selectivity of the conjunctive predicate $p_1 \wedge p_2$? Under what assumptions is your estimate accurate?

Solution

$\hat{s}(p_1 \wedge p_2) = \hat{s}(p_1) * \hat{s}(p_2)$
Assumes that p_1 and p_2 are independent.
In the lecture we apply the independence assumption to join predicates.