VITMAB04 – Databases – Tutorial 4

Teaching Assistant: Dr János Varga

27 October 2022 – Query Optimization

Notations

f_r: blocking factor in relation r

s_r: size of one record of r

b_r: number of blocks currently storing r

n_r: number of records in r

SC(A, r): selection cardinality of A in r.

$$SC(A,r) = \begin{cases} \mathbf{1}, \text{ if A is key,} \\ \dfrac{n_r}{V(A,r)}, \text{ if A is non-key*.} \end{cases}$$

V(A, r): number of distinct values of attribute A in r.

$$V(A,r) = \begin{cases} n_r, \text{ if A is key,} \\ |\Pi_A(r)|, \text{ if A is non-key.} \end{cases}$$

*: *assuming values in A follow uniform distribution.*

1. In a retail bank's database there is a relation Acct with schema Acct(City, Balance, …). We query the details of accounts that belong to citizens of Budapest. We know that:

   - $f_{Acct} = 40$
   - $n_{Acct} = 10\ 000$
   - V(Balance, Acct) = 500
   - V(City, Acct) = 50

   Assume that we store records with maximum block fill.

   a. How do you describe the query with relational algebra?

   b. What is the minimum, maximum and average cost if the engine uses linear search? What factor determines the cost?

   c. Assume records are ordered by branch. What is the expected cost of a binary search?

2. In the same database there're relations Deposit and Client. Join these on the common attribute Client_Name. It is key in Client and (by def.) foreign key in Deposit. System catalog holds the following data about the relations:

   - $n_{Client} = 10\ 000$
   - $f_{Client} = 25$
   - V(Client_Name, Deposit) = 2 500
   - $n_{Deposit} = 5\ 000$
   - $f_{Deposit} = 50$

   Calculate b_{Client}, $b_{Deposit}$, SC(Client_Name, Deposit).

   a. How many clients don't currently hold an account at the bank?

   b. What is the size of the natural join of Client and Deposit if the only attribute in common is Client_Name?

c. Generalize question b. to the following scenarios. What is the size of the natural join of relations R and S if

   i. $\boldsymbol{R} \cap \boldsymbol{S} = \emptyset$?

   ii. $\boldsymbol{R} \cap \boldsymbol{S}$ is a key in R?

   iii. $\boldsymbol{R} \cap \boldsymbol{S} \neq \emptyset$ is neither a key in R nor in S?

3. Find the cost of the hash join of R and S when bucket hash is used.

   Assume that: a) the hash function distributes values evenly; b) block size is 2 000 bytes (ignoring header); c) the hash table fits in RAM.

   What is the best method to execute the join?

   R: n_R = 120 000 records, s_R = 150 bytes, key 12 bytes, pointer 8 bytes,
   hash table size 10 000 bytes.

   S: n_S = 10 000 records, s_S = 250 bytes, key 15 bytes, pointer 8 bytes,
   hash table size 1 000 bytes.

4. Find the cost of the natural join (executed as a nested loop join) if a primary, B*-tree index is used to access records by the join attributes. Block size is 4 000 bytes.

   Which relation should be in the outer loop? What cost do we pay if the optimizer makes the wrong choice?

   R: n_R = 140 000 records, s_R = 140 bytes, key 10 bytes, pointer 4 bytes.

   S: n_S = 15 000 records, s_S = 300 bytes, key 6 bytes, pointer 4 bytes.

Theory – Brainteasers

5. When executing a natural join, the optimizer is allowed to execute other selections specified on participating relations before performing the join. Is the same allowed in case of outer joins?

6. Can you describe a scenario in which it is worth spending more time in the optimization phase than the cost of the slowest possible (most expensive) execution plan?

7. What kind of queries benefit from the use of the primary index during execution?

8. What kind of queries lose performance over the use of the primary index during execution?