ICOM 6005 – Database Management Systems Design

Dr. Manuel Rodríguez-Martínez

Electrical and Computer Engineering Department

Join Evaluation Techniques

- Read Chapter 14, sec. 14.4
- Paper about access paths
- The main issues with join processing
  - Join order
  - Join execution algorithm
- Recall that given two relation R and S, the following result holds:
  - $R \triangleright\triangleleft S = S \triangleright\triangleleft R$
- Thus, regardless of join order, the result is the same
- But, the **COST** of each join evaluation can be quite different!

Join Evaluation Algorithms

- Principal algorithms:
  - Nested Loops Join
  - Blocked Nested Loops Joins
  - Index Nested Loops Join
  - Sort-Merge Join
  - Hash Join
- All these are two-way join strategy
- In general, an n-way join is evaluated as a series of two join operations
  - R $\triangleright\triangleleft$ S $\triangleright\triangleleft$ T can be implemented as:
  1. M = R $\triangleright\triangleleft$ S
  2. N = M $\triangleright\triangleleft$ T

Notational conventions

- Let R be a relation, the following are some its properties:
  - |R| - the cardinality of relation R
  - ||R|| - the number of pages for R
  - B – number of buffer pages to help running a join
- Join cardinality is the number of tuples to satisfy the join condition
- Join condition also has a selectivity factor SF

Nested Loops Join

- Input: relations R and S
- Algorithm:

For each tuple r in R do

    For each tuple s in S do

        if r_i == s_j then add <r,s> to result

- This algorithm goes over all tuples of each relation
- R is called the outer relation – read once
- S is called the inner relation – read |R| times
- Number of repetitions: |R| * |S|
- Cost: ||R|| + (|R|*||S||)

Some issues …

- Nested loops is the simplex, most expensive algorithm
- Which relation should be the inner?
  - If the smallest relation fits into memory, this should be the inner relation.
    - We read each relation only once.
    - Cost = ||R||+ ||S|| I/Os
    - Loops are still repeated |R|*|S| times!
  - Otherwise, need to estimate which combination yields the fewer I/Os
- Example: |R| = 10000, ||R|| = 1000, |S| = 1000, ||S|| = 100
- R $\triangleright\triangleleft$ S : 1,000 + 100 * 10,000 = 1,001,000 I/Os
- S $\triangleright\triangleleft$ R: 100 + 1,000 * 1,000 = 1,000,100 I/Os

Page at a time Nested Loops

- Idea: Do the nested loops between pages!

For each page P in R

    For each page T in S

        For each tuple r in P

            For each tuple s in T

                if r_i == s_j then add <r,s> to result

- Cost : ||R|| + ||R||*||S||
- Example: |R| = 10000, ||R|| = 1000, |S| = 1000, ||S|| = 100
- R ▷◁ S : 1,000 + 100 * 1,000 = 101,000 I/Os
- S ▷◁ R: 100 + 100 * 1,000 = 100,100 I/Os

Blocked Nested Loops Join

- Idea:
  - use more buffers to keep page of outer relation,
  - then do page at a time nested loops with inner relation
- Need to have B buffers
  - B – 2 are used to read in pages from outer relation
  - 1 buffer is used to read pages from inner relation
  - 1 buffer is used to keep output tuples

Blocked Nested Loops

- Algorithm:

For each block of B – 2 pages
    For each page P in the block
        For each page of S do
            For each tuple r in P
                For each tuple s in T
                    if r_i == s_j then
                        add <r,s> to result