CS411 Database Systems

Fall 2009

HW#4

Due: 3:15pm CST, November 17, 2009

Note: Print your name and NetID in the upper right corner of every page of your submission. Hand in your stapled homework to Donna Coleman in 2106 SC. In case Donna is not in office, slide your homework under the door.

To grade homeworks faster, the homework is partitioned into two parts. **Please, submit each part separately.** For each part, make sure to write down your name and NetID. Handwritten submissions will be graded but they will take longer to grade. For clarity, machine formatted text is preferable: Expect to lose points if your handwritten answer is unclear or misread by the grader.

Discussions with other students are strongly encouraged. However, such collaboration should be limited to general strategies and ideas, but not write-ups of specific solutions. Your submitted work should represent your identifiable efforts and originality.

This homework is partitioned into two parts as follows:

- Part 1: Problem 1 - Problem 3
- Part 2: Problem 4 - Problem 7

Part 1

Problem 1 Nested-Loop Joins (16 points)

1. Suppose B(R) = B(S) = 10,000. For what value of M would we need to compute R $\bowtie$ S using the nested-loop join algorithm with no more than the following number of I/O's? (8 points, 4 points each)

(a) 100,000

(b) 25,000

2. If two relations R and S are both unclustered, it seems that the nested-loop join algorithm requires about $T(R)T(S)/M$ disk I/O's. How can you do significantly better than this cost? Describe your modified version of the nested-loop algorithm and give the number of disk I/O's required for your algorithm. We assume that M is large enough such that $M-1 \simeq M$, and that $B(R) \ll T(R)$ and $B(S) \ll T(S)$; that is, the number of tuples of a relation is much greater than that of blocks of the relation. (8 points)

<u>Problem 2</u> Two-pass Algorithms Based on Sorting (20 points)

1. Suppose we have a relation with 1,000,000 records and each records requires 10 bytes. Let the disk-block size be 4,096 bytes. (8 points, 4 points each)

(a) What is the minimum number of blocks in main memory required for using TPMMS (Two-Phase Multiway Merge-Sort) to sort these records?

(b) Following (a), how many disk I/Os are needed to sort all the records?

2. We have two relations R and S where $B(R) = B(S) = 10,000$. Give an approximate size of main memory M required and the number of disk I/O's in order to perform the two-pass algorithms for the following operations: (12 points, 4 points each)

(a) set union

(b) simple sort-join

(c) the more efficient sort-join described in Section 15.4.8 of the textbook

Problem 3 Hash-based and Index-based joins (14 points)

1. If $B(R) = 10,000$ and $B(S) = 30,000$ and $M = 101$, what is the number of disk I/O's required for the hash-join algorithm? (5 points)

2. Suppose $B(R) = 10,000$ and $T(R) = 500,000$. Let there be an index on R.a, and let $V(R, a) = k$ for some number k. Give the cost of $\sigma_{a=0}$(R), as a function of k, under the following circumstances. You may neglect disk I/O's needed to access the index itself. (9 points, 3 points each)

(a) the index is clustering.

(b) the index is not clustering.

(c) R is clustered, and the index is not used.

Part 2

Problem 4 Algebraic Laws (10 points, 5 points each)

1. We consider two relations $R(A, B, C)$ and $S(C, D, E)$. Convert the following expressions in relational algebra by applying algebraic laws so that we can perform selections and projections as early as possible.

(a) $\sigma_{B=3 \text{ AND } E=4}(R \bowtie \sigma_{C>10}(S))$

(b) $\pi_{A,D}(R \bowtie S)$

Problem 5 Dynamic Programming (15 points)

Compute the optimal plan for $R \bowtie S \bowtie T \bowtie U$ using the technique of dynamic programming. We make the following assumptions (as we did in the class):

- $B(R) = 400$, $B(S) = 800$, $B(T) = 600$, and $B(U) = 700$.
- The size of a join for two relations R_1 and R_2 is estimated as: $B(R_1 \bowtie R_2) = 0.01 \times B(R_1) \times B(R_2)$. If a subplan is a single relation and does not involve any join, the size of its intermediate result is zero.
- The cost of a join is estimated to be the cost of the subplans plus the size of the intermediate results.
- The cost of a scan is zero.

Draw the table for dynamic programming, to show how you compute the optimal plan for all possible join orders allowing all trees.

Problem 6 Cost Estimation (15 points, 3 points each)

Consider two relations $R(A, B, C, D)$ and $S(D, E)$ with the following statistics:

$T(R) = 100, V(R, A) = 100, V(R, B) = 10, V(R, C) = 1, V(R, D) = 50;$
$T(S) = 500, V(S, D) = 30, V(S, E) = 100.$

(a) Estimate the number of tuples in $\sigma_{B=25}(R)$

(b) Estimate the number of tuples in $\sigma_{(B=25) \text{ AND } (C=30)}(R)$

(c) Estimate the number of tuples in $\sigma_{B>25}(R)$

(d) Estimate the number of tuples in $\sigma_{(B>25) \text{ AND } (B=15)}(R)$

(e) Estimate the number of tuples in $R \bowtie S$

Problem 7 Pipelining Versus Materialization (10 points)

Consider physical query plans for the expression

$$(R(w, x) \bowtie S(x, y)) \bowtie U(y, z)$$

in Example 16.36 on page 831 of the textbook (We covered the same example in the class). If $B(R) = 2000$, how would you update the table in Figure 16.38 on page 834 of the textbook? Show the revised table. We use the same physical plans for the three cases in the table respectively.