CARNEGIE MELLON UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
15-445/645 – DATABASE SYSTEMS (FALL 2023)
PROF. ANDY PAVLO AND JIGNESH PATEL

Homework #3 (by Anurag Choudhary)
Due: **Sunday, Oct 08, 2023 @ 11:59pm**

IMPORTANT:

- Enter all of your answers into **Gradescope by 11:59pm on Sunday, Oct 08, 2023**.
- **Plagiarism**: Homework may be discussed with other students, but all homework is to be completed **individually**.

For your information:

- Graded out of **100** points; **3** questions total
- Rough time estimate: $\approx$ 1 - 2 hours (0.5 - 1 hours for each question)

Revision : 2023/09/29 19:08

Question	Points	Score
Sorting Algorithms	36	
Join Algorithms	43	
Bloom Filters	21	
Total:	100	

Question 1: Sorting Algorithms . . . . [36 points]

We have a database file with nine million pages (N = 9,000,000 pages), and we want to sort it using external merge sort. Assume that the DBMS is not using double buffering or blocked I/O, and that it uses quicksort for in-memory sorting. Let B denote the number of buffers.

(a) **[6 points]** Assume that the DBMS has fifty buffers. How many sorted runs are generated? Note that the final sorted file does not count towards the sorted run count.
☐ 183,673 ☐ 183,674 ☐ 183,750 ☐ 183,751 ☐ 187,501 ☐ 187,502

(b) **[6 points]** Again, assuming that the DBMS has fifty buffers. How many pages (in general) does each sorted run have after the third pass (i.e. Note: this is Pass #2 if you start counting from Pass #0)?
☐ 49 ☐ 50 ☐ 2,450 ☐ 2,451 ☐ 2,452 ☐ 120,050 ☐ 120,051 ☐ 120,052

(c) **[6 points]** Again, assuming that the DBMS has fifty buffers. How many passes does the DBMS need to perform in order to sort the file?
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

(d) **[6 points]** Suppose the DBMS has 100 buffers. What is the total I/O cost to sort the file?
☐ 18,000,000 ☐ 36,000,000 ☐ 54,000,000 ☐ 72,000,000 ☐ 90,000,000

(e) **[6 points]** What is the smallest number of buffers B such that the DBMS can sort the target file using only three passes?
☐ 55 ☐ 56 ☐ 208 ☐ 209 ☐ 3,000 ☐ 3,001

(f) **[6 points]** Suppose the DBMS has twenty-three buffers. What is the largest database file (expressed in terms of the number of pages) that can be sorted with external merge sort using four passes?
☐ 10,648 ☐ 11,132 ☐ 12,167 ☐ 234,256 ☐ 244,904 ☐ 279,841 ☐ 5,153,632 ☐ 5,387,888 ☐ 6,436,343 ☐ 118,533,536

Question 2: Join Algorithms .... [43 points]

Consider relations `X(a, c)`, `Y(a, b, e)`, and `Z(a, d, f)` to be joined on the common attribute `a`. Assume that there are no indexes available on the tables to speed up the join algorithms.

- There are B = 750 pages in the buffer
- Table `X` spans M = 2,500 pages with 80 tuples per page
- Table `Y` spans N = 600 pages with 300 tuples per page
- Table `Z` spans O = 1,500 pages with 150 tuples per page
- The join result of `X` and `Y` spans P = 500 pages

For the following questions, assume a simple cost model where pages are read and written one at a time. Also assume that one buffer block is needed for the evolving output block and one input block is needed for the current input block of the inner relation. You may ignore the cost of the writing of the final results.

(a) **[3 points]** What is the I/O cost of a simple nested loop join with `X` as the outer relation and `Y` as the inner relation?
☐ 50,500 ☐ 200,600 ☐ 202,500 ☐ 1,502,500 ☐ 120,000,600
☐ 120,002,500 ☐ 300,002,500

(b) **[3 points]** What is the I/O cost of a block nested loop join with `Z` as the outer relation and `Y` as the inner relation?
☐ 1,500 ☐ 2,100 ☐ 2,700 ☐ 3,000 ☐ 3,300 ☐ 3,600 ☐ 3,900
☐ 5,100

(c) **[3 points]** What is the I/O cost of a block nested loop join with `Y` as the outer relation and `Z` as the inner relation?
☐ 1,500 ☐ 2,100 ☐ 2,700 ☐ 3,000 ☐ 3,300 ☐ 3,600 ☐ 3,900
☐ 5,100

(d) For a sort-merge join with `X` as the outer relation and `Z` as the inner relation:

i. **[3 points]** What is the cost of sorting the tuples in `X` on attribute `a`?
☐ 2,000 ☐ 4,000 ☐ 6,000 ☐ 8,000 ☐ 10,000 ☐ 12,000

ii. **[3 points]** What is the cost of sorting the tuples in `Z` on attribute `a`?
☐ 2,000 ☐ 4,000 ☐ 6,000 ☐ 8,000 ☐ 10,000 ☐ 12,000

iii. **[3 points]** What is the cost of the merge phase in the worst-case scenario?
☐ 1,500 ☐ 2,000 ☐ 2,500 ☐ 4,000 ☐ 900,000 ☐ 2,000,000
☐ 3,250,000 ☐ 3,750,000 ☐ 4,500,000 ☐ 5,000,000

Question 2 continues...

iv. **[3 points]** What is the cost of the merge phase assuming there are no duplicates in the join attribute?
☐ 1,500 ☐ 2,000 ☐ 2,500 ☐ 4,000 ☐ 900,000 ☐ 2,000,000
☐ 3,250,000 ☐ 3,750,000 ☐ 4,500,000 ☐ 5,000,000

v. **[3 points]** Now consider joining X, Y and then joining the result with Z. What is the cost of the final merge phase assuming there are no duplicates in the join attribute?
☐ 1,000 ☐ 2,000 ☐ 3,000 ☐ 5,000 ☐ 1,000,000

(e) Consider a hash join with `Y` as the outer relation and `Z` as the inner relation. You may ignore recursive partitioning and partially filled blocks.

i. **[3 points]** What is the cost of the probe phase?
☐ 1,000 ☐ 2,000 ☐ 2,100 ☐ 4,200 ☐ 6,400 ☐ 7,200 ☐ 10,000

ii. **[3 points]** What is the cost of the partition phase?
☐ 1,000 ☐ 2,000 ☐ 2,100 ☐ 4,200 ☐ 6,400 ☐ 7,200 ☐ 10,000

(f) **[3 points]** Assume that the tables do not fit in main memory and that a large number of distinct values hash to the same bucket using hash function h_1. Which of the following approaches works the best?
☐ Create hashtables for the inner and outer relation using h_1 and rehash into an embedded hash table using h_1 for large buckets.
☐ Use linear probing for collisions and page in and out parts of the hashtable needed at a given time.
☐ Create hashtables for the inner and outer relation using h_1 and rehash into an embedded hash table using h_2 != h_1 for large buckets.
☐ Create two hashtables half the size of the original one, run the same hash join algorithm on the tables, and then merge the hashtables together.

(g) For each of the following statements about nested loop joins, pick True or False.

i. **[2 points]** In a simple nested loop join where both tables fit entirely in memory, the choice of which table to use as the inner/outer table significantly affects I/O costs.
☐ True ☐ False

ii. **[2 points]** In a simple nested loop join where one of the tables fits entirely in memory, it is beneficial to use that table as the inner table.
☐ True ☐ False

iii. **[2 points]** If neither table fits entirely in memory, I/O costs would be lower if we process both tables on a per-block basis rather than per-tuple basis.
☐ True ☐ False

iv. **[2 points]** For a block nested loop join, in the worst case, each block in the inner table has to be read once for each tuple in the outer table.
☐ True ☐ False

v. **[2 points]** In an index nested loop join where only one of the tables has an index on the join attribute, the choice of inner/outer tables should be made based on the table sizes.
☐ True ☐ False

Question 3: Bloom Filters .... [21 points]

Assume that we have a bloom filter that is used to register names. The filter uses two hash functions h_1 and h_2 which hash the following strings to the following values:

input	h_1	h_2
"Nas"	1973	1994
"Cole"	1985	2014
"Kendrick"	1987	2015
"Rocky"	1988	2013
"JID"	1990	2022
"Denzel"	1995	2018

(a) **[4 points]** Suppose the filter has 6 bits initially set to 0:

bit 0	bit 1	bit 2	bit 3	bit 4	bit 5
0	0	0	0	0	0

Which bits will be set to 1 after "Cole" and "JID" have been inserted?
☐ 0 ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

(b) **[3 points]** If "Denzel" is inserted next, which bits will now be set to 1 (including those already set in part (a))?
☐ 0 ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

(c) **[3 points]** What will the filter return if we now lookup "Nas"?
☐ True ☐ False

(d) **[3 points]** Suppose the filter has 6 bits set to the following values:

bit 0	bit 1	bit 2	bit 3	bit 4	bit 5
0	1	0	1	0	1

What will the filter return if we lookup "Rocky"?
☐ True ☐ False

(e) **[3 points]** What will the filter from part (d) return if we lookup "Kendrick"?
☐ True ☐ False

Question 3 continues…

(f) **[5 points]** Suppose the filter has 6 bits set to the following values:

bit 0	bit 1	bit 2	bit 3	bit 4	bit 5
0	0	1	1	0	1

Which names are guaranteed to have NOT been inserted?
☐ "Nas" ☐ "Cole" ☐ "Kendrick" ☐ "Rocky" ☐ "JID" ☐ "Denzel"