CS W186 - Spring 2024
Exam Prep Section 5
Sorting/Hashing

1 Sorting and Hashing

Suppose the size of a page is 4KB, and the size of the memory buffer is 1 MB (1024 KB).

1. We have a relation of size 800 KB. How many page IOs are required to sort this relation?

   Answer: 400
   200 to read in, 200 to write out. Since the relation is small enough to completely fit into the buffer we only need to read it in, sort it (no I/Os required for sorting), then write the sorted pages back to disk.

2. We have a relation of size 5000 KB. How many page IOs are required to sort this relation?

   Answer: 5000. 2 passes. 5000 KB with 4KB per page means 1250 pages are needed to store the relation. We have 1024 / 4 = 256 pages in our buffer. Number of Passes = $1 + \lceil \log_{255} \lceil 1250/256 \rceil \rceil = 2$

3. What is the size of the largest relation that would need two passes to sort?

   Answer: 261,120 KB. (255 * 256 pages).

4. What is the size of the largest relation we can possibly hash in two passes (i.e. with just one partitioning phase)?

   Answer: 261,120 KB.

6. Now suppose we were executing a GROUP BY on age instead. Would sorting or hashing be better here, and why?

   Answer: Sorting because hashing won't work; each partition is larger than memory, so no amount of hash partitioning will suffice.

2 External Sorting

Assume our buffer pool has 8 frames. In this question, we'll externally sort a 500 page file.

1. How many passes will it take to sort this file?

   Answer: 4 passes. Number of Passes = $1 + \lceil \log_7 \lceil 500/8 \rceil \rceil = 4$

2. Given the number of passes you calculated in 2.1, how many I/Os are necessary to externally sort the file?

   Answer: 4000 I/Os. 2 * Number of Pages * Passes = 2 * 500 * 4 = 4000 I/Os.

3. What is the minimum number of additional frames needed to reduce the number of passes found in 2.1 by 1?

   Answer: 1 additional frame. Given that we had 4 passes in 2.1, we need to calculate how many pages it will take to sort the relation in 3 passes.
   $B(B-1)^2 >= 500$
   B = 9 frames. 9 - 8 = 1 additional frame. I/Os.

4. What is the minimum number of additional frames needed to sort the file in one pass?

Answer: 492 pages. If we can fit the entire table into the buffer, our initial sorting pass will sort the table. Therefore 500 - 8 = 492 pages.

3 Hashing

1. Suppose the size of each page is 4KB, and the size of our memory buffer is 64KB. What would be the I/O cost of hashing a file of 128 pages, assuming that the first hash function creates 2 partitions of 32 pages, and all other partitions are uniformly partitioned?

   Answer: 721 pages.
   There are B = 64KB/4KB = 16 pages of RAM.
   We read the 128 pages of the file into B-1=15 partitions.

   - Partition 1: 32 pages
   - Partition 2: 32 pages
   - Partitions 3-15: $\lceil (128 - 64)/13 \rceil = 5$ pages
   - We write the 32 + 32 + 13*5 = 129 pages to memory.

   To recursively partition the partitions of 32 pages:

   - We read in 32 pages for each into B-1=15 partitions.
   - Partition size: $\lceil 32/15 \rceil = 3$ pages
   - We write out 15 partitions of 3 page each: 3*15 = 45.

   In the conquer phase:

   - From the first divide phase, we have 13 partitions that fit into memory, of 5 pages each. • From each of the recursive partitions, we have 15 partitions that fit into memory, of 3 pages each.
   - From the conquer phase, this is a total of (65+45+45) = 155 I/O's for reading, and (65+45+45) = 155 I/O's for writing.

   This gives us a total of (128 + 129) + 2*(32 + 45) + (155+155) = 721 pages.