Homework 2

Thinh Nguyen

Problem 1: External Sorting

Assume a file consisting of 200 blocks, which we need to sort.

(a) Assume we only have 5 memory buffers available

1. What is the total number of I/Os to sort the entire file?

Pass 0: `ceil(200 / 5) = 40` sorted runs of 5 blocks each.

Pass 1: `ceil(40 / 4) = 10` sorted runs of 20 blocks each.

Pass 2: `ceil(10 / 4) = 3` sorted runs of 80 blocks each (last run is only 40 blocks).

Pass 3: `ceil(3 / 4) = 1` sorted run for sorted file of 200 blocks.

`Total I/O cost = 2N * (# of passes) = 2 * 200 * 4 = 1600`

2. How many sorted runs (lists) are there after the 1st merge step?

After the 1st merge step, 10 sorted runs.

3. How many sorted runs (lists) are there after the 2nd merge step?

After the 2nd merge step, 3 sorted runs.

(b) Assume we only have 8 memory buffers available

1. What is the total number of I/Os to sort the entire file?

Pass 0: `ceil(200 / 8) = 25` sorted runs of 8 blocks each.

Pass 1: `ceil(25 / 7) = 4` sorted runs of 56 blocks each (last run is only 32 pages).

Pass 2: `ceil(4 / 7) = 1` sorted run for sorted file of 200 blocks.

`Total I/O cost = 2N * (# of passes) = 2 * 200 * 3 = 1200`

2. How many sorted runs (lists) are there after the 1st merge step?

After the 1st merge step, 4 sorted runs.

3. How many sorted runs (lists) are there after the 2nd merge step?

After the 2nd merge step, 1 sorted run.

Problem 2: Indexing

Consider a data file R consisting of 1,000,000 blocks that are contiguous on disk. Each block contains 20 records, i.e., the total number of records in R is (20 * 10^6). Each record consists of attributes (K1, K2, ...). Let K1 be the primary key of the relation, and hence R's records are physically sorted by K1. Also, let K2 be another attribute of R (unsorted attribute). Assume the size of K1 and K2 attributes is 20 bytes each, a record pointer is 8 bytes long, and a block is 8KB (8 * 1024 bytes). Assume no single record (data record or index record) can be divided across two blocks.

1. Is it possible to construct a dense sequential index (1-level) on K1 over R? Describe the layout, and how large (how many blocks) will the index be?

A dense sequential index on K1 over R is possible. Each index entry will consist of a key (K1) and a pointer to the record in R. The size of each index entry is `20 bytes (K1) + 8 bytes (pointer) = 28 bytes`. The number of index entries that can fit in a block is `floor(8 * 1024 bytes / 28 bytes) = 292`. To index all records in R, we need `ceil(20 * 10^6 / 292) = 68,494` blocks for the index.

2. Is it possible to construct a sparse sequential index (1-level) on K1 over R? Describe the layout, and how large (how many blocks) will the index be?

A sparse sequential index on K1 over R is also possible. In a sparse index, we only store the first index entry of each block in R. Since R spans 1,000,000 blocks, we will have corresponding 1,000,000 index entries. Each index entry will consist of a key (K1) and a pointer to the first record in the block. The size of each index entry is `20 bytes (K1) + 8 bytes (pointer) = 28 bytes`. To index all blocks in R, we need `ceil(1,000,000 / 292) = 3,425` blocks for the index.

3. Is it possible to construct a dense sequential index (1-level) on K2 over R? Describe the layout, and how large (how many blocks) will the index be?

A dense sequential index on K2 over R is possible. Each index entry will consist of a key (K2) and a pointer to the record in R. The size of each index entry is `20 bytes (K2) + 8 bytes (pointer) = 28 bytes`. The number of index entries that can fit in a block is `292`. To index all records in R, we need `68,494` blocks for the index.

4. Is it possible to construct a sparse sequential index (1-level) on K2 over R? Describe the layout, and how large (how many blocks) will the index be?

It is not possible to construct a 1-level sparse sequential index on K2 over R since K2 is unsorted. A sparse index requires the indexed attribute to be physically sorted.

5. Is it possible to build a second-level index on the one built in 1? If yes, what will be the size of the index (how many blocks)? Report the size of the second-level alone and the total index size (both levels).

Yes, it will be a sparse sequential index. Each 2nd-level index entry will hold the first index entry of each block in the 1st-level index. Since the 1st-level index spans `3,425` blocks, we will have `3,425` corresponding 2nd-level index entries. Each 2nd-level index entry will consist of a key (K1) and a pointer to the first index entry in a block. The size of each 2nd-level index entry is `20 bytes (K1) + 8 bytes (pointer) = 28 bytes`. The number of 2nd-level index entries that can fit in a block is `292`. To index all blocks in the 1st-level index, we need `ceil(3,425 / 292) = 12` blocks for the 2nd-level index. The total index size (both levels) will be `3,425 + 12 = 3,437` blocks.

6. Is it possible to build a second-level index on the one built in 2? If yes, what will be the size of the index (how many blocks)? Report the size of the second-level alone and the total index size (both levels).

It is possible to build a 2nd-level sparse sequential index. Each 2nd-level entry will hold the first index entry of each block in the 1st-level index. Since the 1st-level index spans `3,425` blocks, we will have `3,425` corresponding 2nd-level index entries. Each 2nd-level index entry will consist of a key (K1) and a pointer to the first index entry in a block. The size of each 2nd-level index entry is `20 bytes (K1) + 8 bytes (pointer) = 28 bytes`. The number of 2nd-level index entries that can fit in a block is `292`. To index all blocks in the 1st-level index, we need `ceil(3,425 / 292) = 12` blocks for the 2nd-level index. The total index size (both levels) will be `3,425 + 12 = 3,437` blocks.