NYU Tandon School of Engineering
Computer Science and Engineering
CS 6083-INET, Spring 2018

April 18, 2018

Homework # 4 (due April 29)

Problem 1 (15 Points)

You are given a sequence of 8 key values and their 8-bit hash values that need to be inserted into an extendible hash table where each hash bucket holds at most two entries. The sequence is presented in Table 1 below. (You do not need to know what function was used to compute the hashes, since the resulting hashes are already given.) In Figure 1 you can see the state of the hash table after inserting the first two keys, where we only use the first (leftmost) bit of each hash to organize the buckets. Now insert the remaining six keys (k_2 to k_7) in the order given. Sketch the bucket address table and the buckets after each insertion.

keys	Hash values
k_0	10101101
k_1	01001010
k_2	11000010
k_3	11101111
k_4	11010010
k_5	01101001
k_6	10101010
k_7	01101011

Table 1: Sequence of 8 keys and their corresponding 8-bit hashes.

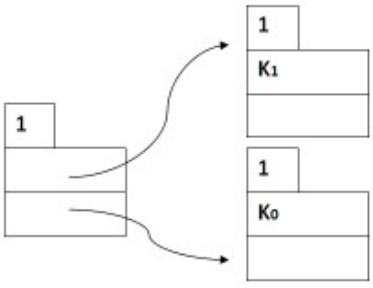

Figure 1: Hash Table state after the insertion of the first two keys

Problem 2 (20 Points):

(a) Suppose you have a hard disk being used to store the data that has 15000 RPM and 2 double sided platters. The divisions on hard disk are in forms of sectors and tracks. There are 250,000 tracks, 2000 sectors per track and 512 bytes per sector. Assume for simplicity that for inner and outer tracks number of sectors is the same. What is the capacity of the disk? What is the average rotational latency? Give the maximum rate for reading the data from the disk.

(b) Assume having the same disk and avg. seek time is 5 ms. How much time does it take to read files of size 100 KB, 1000 KB and 100 MB. Calculate using both block (4 KB) and latency/transfer-rate model. Also, give your comments comparing both models.

(c) You need to sort a file of size 36 GB, and you have only 1 GB of main memory to do the sort (plus unlimited disk space). Use the latency/transfer-rate model of disk performance and ignore CPU performance. Assume that in the merge phase, all sorted runs from the initial phase are merged together in a single merge pass. What would be the running time of I/O efficient merge sort algorithm?

(d) Suppose you use two (instead of one) merge phases in the scenario in (c). What is the running time now?

Problem 3 (15 Points)

In the following, assume the latency/transfer-rate model of disk performance, where we estimate disk access times by allowing blocks that are consecutive on disk to be fetched with a single seek time and rotational latency cost (as shown in class). Also, we use the term RID (Record ID) to refer to an 8-byte "logical pointer" that can be used to locate a record (tuple) in a table.

You are given the following very simple database schema that models a movie theatre chain theatres, screenings and customers buying movie tickets. We store the city where a customer lives, and we also store the information about theaters, movie screenings and tickets. The details of the schema are shown below:

Customer (cid, cname, ccity)
Theatre (tid, tname, tcity)
MovieScreening (mid, tid, mname, mduration, mtype)
MovieTicket (cid, mid, mtimestamp, price, qty)

Assume there are 100 million customers, 0.5 million movies and 5 billion movie tickets purchased over a period of 100 days. Each “movie ticket” tuple is of size 40 bytes, and all other tuples are 100 bytes.

Assume that 1% customers live in San Francisco and 0.1% of the movies are Horror. Otherwise, assume that data is evenly and independently distributed; e.g.: 50 million movie tickets purchased records were made on June 26, 2017 in whole USA.

Consider following queries:

1. select cid, cname
from customer
where ccity = ‘San Francisco’

2. select c.cid, c.cname
from customer c, movieticket m
where c.cid=o.cid and c.ccity=‘San Francisco’ and m. mtimestamp = ‘2017-11-10’

3. select m.mid, m.mname
from movie m, movieticket mt
where m.mid=mt.mid and m.type=‘Horror’ and mt. mtimestamp = ‘2017-10-11’

(a) For each query, describe in one sentence what it does. (That is, what task does it perform?) In the following questions, to describe how a query could be best executed, draw a query plan tree and state what algorithms should be used for the various selections and joins. Provide estimates of the running times, assuming these are dominated by disk accesses.

(b) Consider a sparse clustered B+-tree index on ccity in the Customer table, and a dense unclustered B+-tree index on cid in the MovieTicket table. For each index, what is the height and the size of the tree? How long does it take to fetch a single record with a particular key value value using these indexes? How long would it take to fetch all, say, 50 records matching a particular cid value in the case of the second index? Assume that ccity is of size 32 bytes, and cid of size 8 bytes. Given that 1.5GB of main memory is available for query processing, and assuming a hard disk with 10 ms for seek time plus rotational latency (i.e., a random access requires 6 ms to find the right position on disk) and a maximum transfer rate of 150 MB/s.

(c) Suppose that for each query, you could create up to two index structures to make the query faster. What index structures would you create, and how would this change the evaluation plans and running times?