CS 315 Weeks 5 and 6 (March 11 and 13) summary and review questions

Topics covered

- Hashing (completed)
- Application of hashing : LZW compression algorithm

SUMMARY:

Some of the following topics on hashing were reviewed:

- Hashing: goal is to perform each of Search, Insert and Delete – each in O(1) time in average.
- Hash function: A map h: U → {0, 1, 2, … , m – 1} where m is the size of the hash-table. A well-known exampke is h(x) = c(x) % m where c(x) is a (unique) integer that represents the object x. For example, if x is a string over an alphabet of size k, one way to map x to an integer is to treat x as an integer in base 256 where each character of the string is assigned its ASCII value. The ASCII values of the alphabetical symbols range from 65 (for A) to 90 (for Z). Thus the word "DATA" can be viewed as the integer $68 * (256)^3 + 65 * (256)^2 + 84 * (256) + 65$.
- Some criteria for a good hash function:
    - It should map the universe all keys uniformly to the buckets.
    - It should use all the digits of the key.
    - It should be easy to compute.
- The hash table size is desirable to be a prime number.
- Load factor $\lambda = n / m$ where m = number of buckets in the hash table and n = the number of keys currently in the table.
- Chaining: keys that map the same location are chained to form a linked list.
    - Complexity of unsuccessful search ~ λ
    - Complexity of successful search ~ $\lambda / 2$
- Closed hashing: the keys are directly stored in the table. Need policy for rehashing in case of collisions.
    - Linear probing

    - Linear probing with increment d > 1
    - Quadratic probing
    - Double hashing
    - Random hashing
- Assuming random hashing, the complexity of unsuccessful search is ~ $1/(1-\lambda)$

LZW algorithm: The algorithm of Lempel-Ziv-Walsh is a **lossless** compression algorithm. This means decompressing the compressed text will give EXACTLY the original text back.

- Appropriate for text compression. (Image compression algorithms tend to be **lossy** since the human eye can compensate for significant loss of information in the original image.
- Compression ratio = size of the original text / size of the compressed text (so compression ratio is usually > 1.)
- Typical compression ratio achieved by LZW algorithm is ~ 3 (some lossy image compression algorithm can achieve compression ratios ~ 100.)
- Basic idea: create a table that provides an index in a table for some substrings in the text. When you scan the text, identify the longest prefix p of the remaining text that is stored in the table and replace it by the index I(p), then add to the table a new index for the word pa where a is the next letter following p in the text. Now delete p from the text and repeat the process.
- Decompression can be performed without the need for the code table since it can be reconstructed from the compressed text.
- Note that LZW algorithm ends up encoding the text as a sequence of integers and the number of integers used depends on the text being compressed. In practice, we can decide the size of the coding table and when the table reaches this size, we can continue the encoding process but will stop adding new entries to the table.
- The code presented in the lecture (from Sahni's book) uses a table size of 4096.

Review questions:

1) Exercises 5.1, 5.2

3) Consider the chaining based hash table. Suppose the hash table currently contains the keys 23, 8, 41, 33, 34, 19, 12. Assume also that U = {1, 2,…, 40} and that the hash function used is h(x) = x mod 11. (a) if one of the current keys in the table is being searched with uniform probability, what is the expected number of comparisons performed? (b) if one of the keys currently not in the table is being inserted into the table (again from uniform distribution), what is the expected number of comparisons performed? (Note that the insertion algorithm searches the list h(x) before inserting the key x.)

4) Answer (3) assuming that the closed hashing is used. Assume that the rehashing strategy used is linear probing with increment d = 1.

5) Consider a spelling checker application in which the size of the dictionary is ~ 10^5. Suppose the document on which spelling check is performed has about 10^4 distinct words. Answer the following questions for (1) chaining based hash table and (2) closed hashing with linear probing. (Assume that the table size chosen is 2 x 10^5.)

   (a) What is the expected number of operations performed to search the words?

   (b) What is the expected number of operations performed to insert words from the dictionary at the start and near the end of insertion process?

6) What is the successive sequence of indices probed while searching for the key x with h(x) = 10 on a hash table of size 17 (a) in the case of linear probing with d = 3 and (d) in the case of quadratic probing and (c) in the case of secondary hashing assuming g(x) = 7. (g(x) is the secondary hash function.)

7) Apply LZW algorithm to the following text: ababbababbabbababbabab