CS 315 Weeks 5 and 6 (Feb 26, Feb 28 and March 4) summary and review questions

Topics covered

- Queue as ADT, operations: insert, delete, isEmpty, isFull
- Queue implementation using a circular array
- Application of queue in recognizing connected sets of foreground pixels in an image. This problem is known as component labeling and is an important first step in optical character recognition.
- Dictionary ADT: Insert, Search and Delete.
- Goal of hashing is to perform search, insert and delete – each operation in constant time on average.
- Hash function
- Chaining
- Closed hashing

SUMMARY:

- Queue uses a FIFO policy.
- Because the growth is at one end and the deletion is at the other, a circular array is the correct way to implement a queue.
- Queue operations (insert, delete, isEmpty, isFull) take O(1) using an array implementation so there is no need to use a linked list for a queue.
- Component labeling is done by placing the first white pixel in a queue; from then on all the white pixels that are reachable from this are brought to the queue step by step. When the queue is empty, all pixels that are connected together with the first white pixels have entered and left the queue. While the queue grows and shrinks, we keep track the dimensions of the closest bounding box. When the queue becomes empty, we draw the bounding box for the current letter recognized. Then we continue until all the individual letters are identified.
- 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 ~ $1/ (1 – \lambda)$