Hashing ~ 02

CSE 122 ~ Algorithms & Data Structures

Hashing: Review

- Two things you need for hashing
  - Define the hash function
  - Determine how to handle collisions

Collisions: Birthday Example

- Think of your birthday
  - Format (MM/DD/YYYY)
- What happens if we have a hash function that uses month and date to determine a storage index?
  - Collisions occur when people have the same month/day
- How many randomly chosen people in a room before it becomes likely two people will have the same birthday?
  - With just 40 people in a room, there is around a 54% chance that two people will have the same birth month and day.

Why Hashing?

- ➔ Hashing is restricted to Dictionary Operations
- ➔ Trees do insert, delete, and search(LookUp) in O(logn) time
- ➔ Hashing provides a way to have the best case and average case for insert, delete, and search(LookUp) be O(1).
  - Worst Case is O(n)

Hash Tables

- **Hash table:** a data structure in which keys are mapped to array positions by a hash function
  - Map keys to an array index
- Formal Definition
  - Let K be a key space, that is a large (possibly infinite) set from which indices can be drawn
  - Let $\{K_0, \ldots K_{n-1}\}$ be a particular set of keys on which dictionary operations insert, lookup, and possibly delete are to be performed
  - Store the members of this set in a hash table T[0, …, *m-1*] with the aid of a hash function *h*: k → {0, …, *m-1*}
  - For each *j* key K_j is to be stored in the table at position $h(K_j)$. If *h* can be computed quickly then to retrieve a key K one can compute h(K).
  - Cannot be a one-to-one mapping

Basic Principles of Hash Functions

- ➔ Uniform
  - A good hash function *h* tends to spread the keys out uniformly in the table
  - If a Key K is drawn at random from the key space K then the probability that h(K) = i should be 1/m that is independent of i
- ➔ Low Cost
  - The cost of computing a hash function can’t be expensive
  - Comparison search takes O(logn), so cost needs to take less time
- ➔ Deterministic
  - The same hash value must be generated for a given input value
  - No hashing based on memory address or time of day

Hash Functions

- ➔ Assume you have numeric keys
- ➔ If you have alphanumeric convert them to numeric by summing the ASCII value of the characters.
  - Example: Take “dog”
    - d = 100, o = 111, g = 103
    - $K = 100*128^0 + 111 * 128^1 + 103 * 128^2 = 1701860$
  - What about “bat”
    - b = 98, a = 97, t = 116
    - $K = 98*128^0 + 97*128^1 + 116*128^2 = 1913058$
- ➔ Now that we have a key, we need to hash it

Hash Functions: Division Methods

- A simple but good hashing function is
  - h(K) = K mod m
- Provided that *m* is chosen correctly
  - *m* is the size of the table
  - Best to choose *m* to be prime
  - Want to avoid systemic collisions