STAT 534
Lecture 7
Hashing and Hash Functions
April 25, 2019

mmp@stat.washington.edu
Scribes: Sizhe Chen, Yandi Jin, Jianshun Wu

1 Definitions

- **Dictionary**

  A data structure to store pairs (key, data) and support operations search, insert, and delete.

- **Universe**
  - U = {k keys}, the universe of all possible keys.
  - n: number of items in dictionary
  - typically $n << |U|$

- **Hash function and hash table**
  - Hash function: $h : U \to \{0, 1, ..., m - 1\}$
  - Hash table T
  - Load factor $\alpha = n/m < 1$, $m << U$

- Operations `Insert, Delete, Find (k [,data])`
  - `calculate j = h(k)`
  - `perform operation at T[j]`
  - `Insert operation runs in constant time`

- `Collision`
  - `h(k)=h(k') when k`$\neq$`k'` $\iff$ `collision`
  - `Deal with collision: insert items hashed to the same bucket in a list at that bucket`
  - $n_j = \#$ `of items at bucket j`
  - `Good h iff Pr[h(k)=j]`$\approx \frac{1}{m}$ `when k`$\sim$`Uniform(U)`
    $\to E[n_j] = \frac{n}{m} = \alpha$
  - `Delete and Find operations run in O(`n_j`) time`
    $\to E[\texttt{runtime}] = 1 + \alpha$

- Hash function examples
  - h=k mod m, where m is prime
    - e.g. if m not prime, suppose $m = 2^P$, $|U| = 2^{p'}$ ($p' > p$).
      $p' = 5$, $p = 3$, $k = (11010)_2 = (26)_{10}$: $h(k) = (2)_{10} = (010)_2$
    - e.g.$(25)_{10}$ mod 10 = 5
    - Thus, if m not prime, hashing only make use of the last few digits of k
  - $h(k) = \lfloor m \text{ frac}(kA) \rfloor$
    - m is the integer part, which can be any number
    - frac(kA) is the fractional part $\in [0, 1)$
      - e.g. $k = 359$, $A = 0.1 \to$ frac(kA)=0.9
    - $A < 1$

2 Universal Hashing

- **Why universal hashing?**
  - Any fixed hash function is vulnerable: e.g. choose n keys hash to same slot,yielding average time of $O(n)$.
  - Only effective way: Choosing the hash function **randomly** in a way that is **independent** of the keys that are actually going to be stored.
  - This approach can yield provably good performance on average.
- **What is Universal?**
  - **Universal**: for each pair of distinct keys $k, l \in U$, the number of hash functions $h \in H$ for which $h(k) = h(l)$ is at most $\frac{|H|}{m}$
  - Collision between keys k and l is no more than chance $\frac{1}{m}$
- **How good is Universal hashing of average time ?**
  - Given $h \in H$, use it to hash n keys into a table T of size m. if $k \notin T$, then the expected lengthh($n_{h(k)}$) of the list: $E(n_{h(k)}) = \alpha = \frac{n}{m}$.
  - If $k \in T$, $E(n_{h(k)}) = 1 + \alpha$.
  - It takes $O(n)$ expected time to handle any sequence of n INSERT, SEARCH, and DELETE operations containing $O(m) INSERT$ operations.
- **Design of Universal Hashing**
  - Family of all hash functions based on number theory: $H_{m,p} = \{h_{a,b}, a, b \in Z_p, a \neq 0\}$,$p$ is prime. And $p >> |U|$, $h_{a,b}(k) = [(ak + b) \mod p] \mod m$

- $a, b \in Z_p \Leftrightarrow a \sim U[1 : p-1], b \sim U[0 : p-1]$
- $h_{a,b} \sim U(H)$
- e.g. $p = 17$ and $m = 6$, then $h_{3,4}(8) = [(3 \times 8 + 4) \mod 17] \mod 5 = 11 \mod 6 = 5$
- We can prove that the hashing function above is Universal.

3 Local Sensitive hashing

- **Definition**
  - LSH is a set of techniques that dramatically speed up search-for-neighbors or near-duplication detection on data.
  - An LSH family F is defined for a metric space $\mathcal{M} = (M, d)$,a threshold $R > 0$ and an approximation factor $c > 1$, This family F is a family of functions $h : \mathcal{M} \rightarrow S$ which map elements from the metric space to a bucket $s \in S$. The LSH family satisfies the following conditions for any two points $p, q \in \mathcal{M}$, using a function $h \in \mathcal{F}$which is chosen uniformly at random: if $d(p, q) \leq R$, then $h(p) = h(q)$ (i.e.,p and q collide) with probability at least P_1;if $d(p, q) \geq cR$, then $h(p) = h(q)$ with probability at most P_2.
- **Example**
  - $h : [g_1(x), g_2(x), \ldots, g_k(x)] \rightarrow [0, 1], g_i(x) \in \{0, 1\}$
  - $x \in R^d, h = [\frac{a^T x + b}{w}] \in Z$, $a \in R^n \sim U(s^{d-1})$, $b \in R \sim U(0, w)$, $w = 1$ (constant)

4 Reference

1. Introduction to Algorithms,First Edition. by Thomas H. Cormen , Charles E. Leiserson , Ronald L. Rivest, Ronald Rivest. Chapter 12.2 Hash Tables 12.3 Hash Functions.
2. Wikipedia: Local Sensitive hashing.https://en.wikipedia.org/wiki/Locality-sensitive hashing