IIT KHARAGPUR

IIT GANDHINAGAR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

Scaling Data Science

Lecture 6: Introduction to Hashing

Anirban Dasgupta

Computer Science and Engineering

IIT GANDHINAGAR

IIT Gandhinagar
Indian Institute of Technology Gandhinagar

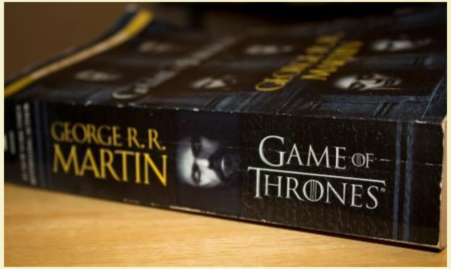
Outline

- Outline:
  - Hash tables and hash functions
  - Universal hashing
  - Chaining
  - Multiplicative hashing

Anirban Dasgupta
Computer Science and Engg.

Querying

Present?



Naïve algorithm: linear in dataset size

Hash Table

- Elements come from universe U, but we need to store only n items, $n < |U|$
- Hash table
  - array of size m
  - Hash function $h: U \to \{0,1, \ldots m-1\}$
- We typically use $m \ll |U|$ as well as $m < n$
  - Collisions happen when $x \neq y$, but $h(x) = h(y)$

Hash functions

- In theory, we design for worst-case behaviour of data
  - Need to choose hash function "randomly"
- Hash family $H = \{h_1, h_2, \ldots\}$
  - When creating hash table, a **single** function $h \in H$ is chosen randomly
  - We then analyse the expected query time
- However…

Hash functions

- In theory, we design for worst-case behaviour of data
  - Need to choose hash function "randomly"
- Hash family $H = \{h_1, h_2, \dots\}$
  - When creating hash table, a **single** function $h \in H$ is chosen randomly
  - We then analyse the expected query time
- Since the algo has to carry around the "description" of the hash function, it needs $\log(|H|)$ bits of storage
  - $|H|$ cannot too big, in particular, it cannot be the set $[m]^U$, all possible functions

Hash family

- We need to create small hash families H such that choosing from it gives a function with "good behaviour"

Hash family

- We need to create small hash families H such that choosing from it gives a function with "good behaviour"
- Uniform: $\Pr_{h \in H}[h(x) = i] = \frac{1}{m}$ for all x and i

Hash family

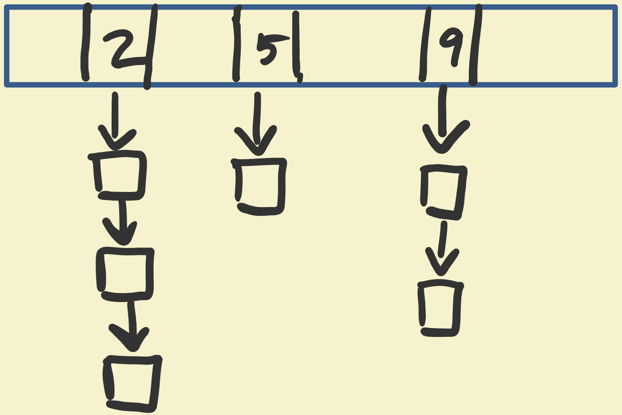
- We need to create small hash families H such that choosing from it gives a function with "good behaviour"
- Uniform: $\Pr_{h \in H}[h(x) = i] = \frac{1}{m}$ for all x and i
  - Not enough
- Universal: $\Pr_{h}[h(x) = h(y)] = \frac{1}{m}$ for all $x \neq y$

Hash family

- We need to create small hash families H such that choosing from it gives a function with "good behaviour"
- Uniform: $\Pr_{h \in H}[h(x) = i] = \frac{1}{m}$ for all x and i
  - Not enough
- Universal: $\Pr_{h}[h(x) = h(y)] = \frac{1}{m}$ for all $x \neq y$
- Near Universal: $\Pr_{h}[h(x) = h(y)] \leq \frac{2}{m}$ for all $x \neq y$

Chaining

- When collisions happen, we store elements using a linked list from that location



Chaining

- When collisions happen, we store elements using a linked list from that location
- $l(x) =$ length of chain at position $h(x)$
- Expected time to query $x = O(1 + E_h[l(x)])$
  - Same for insert and delete

Analyzing chaining

- Need to bound $E_h[l(x)]$
- For $x \neq y$, define $C_{xy} = \begin{cases} 1 \;\; if \; h(x) = h(y) \\ 0 \qquad\qquad\qquad\quad else \end{cases}$

Analyzing chaining

- Need to bound $E_h[l(x)]$

- For $x \neq y$, define $C_{xy} = \begin{cases} 1 \;\; if\; h(x) = h(y) \\ 0 \qquad\qquad\quad else \end{cases}$

- $E_h[l(x)] = E_h[\,\sum_y C_{xy}\,]$

Analyzing chaining: universal hashing

- Need to bound $E_h[l(x)]$
- For $x \neq y$, define $C_{xy} = \begin{cases} 1 & if\ h(x) = h(y) \\ 0 & else \end{cases}$
- $E_h[l(x)] = E_h[\sum_y C_{xy}] = \sum_y \Pr[h(x) = h(y)] = \frac{n}{m}$

universal

IIT Gandhinagar
Indian Institute of Technology Gandhinagar

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Multiplicative hashing

- How to design small + universal hash family?
- Prime multiplicative hashing:
  - Fix a prime number $p > |U|$
  - $H = \{\, h_a(x) = (ax \; mod \; p) \; mod \; m \,, a \in \{1, \ldots p-1\}\}$
  - Choosing a hash function is same as choosing $a \in \{1, \ldots p-1\}$

Multiplicative hashing

- $H = \{ h_a(x) = (ax \; mod \; p) \; mod \; m \,, a \in \{1, \ldots p-1\}\}$

- This family satisfies $\Pr_h[h(x) = h(y)] \leq \frac{1}{m}$

- Intuition: $h_a(x) \; - h_a(y) = (a(x \; - y) \; mod \; p) mod \; m$
- There are at most $\frac{p-1}{m}$ values in $\{1, .. p-1\}$ that are divisible by m

Multiplicative hashing

- $H = \{ h_a(x) = (ax \; mod \; p) \; mod \; m \, , a \in \{1, \dots p-1\}\}$

- This family satisfies $\Pr_h[h(x) = h(y)] \leq \frac{1}{m}$

- Intuition: $h_a(x) \; - h_a(y) = (a(x \; - y) \; mod \; p) mod \; m$
- There are at most $\frac{p-1}{m}$ values in $\{1, .. p-1\}$ that are divisible by m
- What is the probability of choosing a such that $(a(x \; - y) \; mod \; p)$ is one of these numbers?

A property of prime numbers

WLOG $x - y \in [1, p-1]$

<u>Property</u>: For every $\mathrm{t}, z \in [1, p-1]$ there exists unique $a \in [1, p-1]$ such that $az \bmod p = t$

This would imply that probability of choosing collision-causing a

$$\leq \frac{p-1}{m} \times \frac{1}{p-1} = \frac{1}{m}$$

A property of prime numbers

WLOG $x - y \in [1, p-1]$

Property: For every $t, z \in [1, p-1]$ there exists unique $a \in [1, p-1]$ such that $az \mod p = t$

By contradiction. If not, then $\exists a, b \in [1, p-1]$ such that $(a-b)z \mod p = 0$.

But this cannot be as p is prime.

k-wise universal

- For any distinct $(x_1, \ldots, x_k)$ and any (not necessarily distinct) $(y_1, \ldots y_k)$,

$$\Pr[h(x_1) = y_1 \wedge \cdots h(x_k) = y_k] = m^{-k}$$

- Needs only $O(k \log n)$ bit of storage

Summary

- Hashing
  - Simple and versatile
  - Main issue is design of good hash functions, much researched area
  - (near) universality guarantees small chain sizes
  - Other alternatives to chaining exist, e.g. open addressing, cuckoo hashing

References:

- Primary reference for this lecture
  - Algorithms and models of computation by Jeff Erickson: http://jeffe.cs.illinois.edu/teaching/algorithms/

- Others
  - Algorithms, by Cormen, Leiserson and Rivest
  - Randomized Algorithms by Mitzenmacher and Upfal.

Thank You!!