LECTURE 18

Last time
• Finding Hamiltonian cycles in random graphs

Today
• Hashing
Motivating example

Password checker to prevent people from using common passwords.

- $S$ is the set of common passwords

- **Universe:** set $U$
- $S \subseteq U$ and $m = |S|$  
- $m \ll |U|$

**Goal:** A data structure for storing $S$ that supports the search query “Does $w \in S$?” for all words $w \in U$. 

11/8/2022

Sofya Raskhodnikova; Randomness in Computing
Deterministic solutions

- Store $S$ as a sorted array (or as a binary search tree)

  **Search time:** $O(\log m)$,  **Space:** $O(m)$

- Store an array that for each $w \in U$ has 1 if $w \in S$ and 0 otherwise.

  **Search time:** $O(1)$,  **Space:** $O(|U|)$

A randomized solution

- Hashing
Chain Hashing

- **Hash table:** \( n \) bins, words that fall in the same bin are chained into a linked list.
- **Hash function:** \( h : U \rightarrow [n] \)

To construct the table

hash all elements of \( S \)

To search for word \( w \)

check if \( w \) is in bin \( h(w) \)

**Desiderata for \( h \):**
- \( O(1) \) evaluation time.
- \( O(1) \) space to store \( h \).
A random hash function

- **Simplifying assumption:** hash function $h$ is selected at random:
  \[ \Pr[h(w) = j] = \frac{1}{n} \text{ for all } w \in U, j \in [n] \]
- Once $h$ is chosen, every evaluation of $h$ yields the same answer.

**Search time:**
- If $w \notin S$, expected number of words in bin $h(w)$ is
- If $w \in S$, expected number of words in bin $h(w)$ is

If we set $n = m$, then
- the expected search time is $O(1)$
- max time to search is max load: w.p. close to 1, it is $\Theta\left(\frac{\ln m}{\ln \ln m}\right)$

Faster than a search tree, with space still $\Theta(m)$. 
Are we done?

- How many hash functions are there?
- How many bits do we need to store a description of a hash function?

This is prohibitively expensive!

**Idea:** Choose from a smaller family of hash functions.
Universal hash family

- A set $\mathcal{H}$ of hash functions is **universal** if for every pair $w_1, w_2 \in U$ and for $h$ chosen uniformly from $\mathcal{H}$

\[
\Pr[h(w_1) = h(w_2)] \leq \frac{1}{n}
\]

**Constructing a universal hash family**

- Fix a prime $p \geq |U|$ and think of the range as $\{0, 1, \ldots, n-1\}$.
- Define $h_{a,b}(x) = ((ax + b) \mod p) \mod n$
- $\mathcal{H} = \{h_{a,b} \mid a \in [p-1], 0 \leq b \leq p - 1\}$

**Theorem**

$\mathcal{H}$ is universal.

$U = \{0, 1, \ldots, u - 1\}$
Proof that $\mathcal{H}$ is universal

- Define $h_{a,b}(x) = ((ax + b) \mod p) \mod n$
  \[ \mathcal{H} = \{ h_{a,b} \mid a \in [p - 1], 0 \leq b \leq p - 1 \} \]

*Proof:* Fix $x_1 \neq x_2$ from $U$.

- **Idea:** count # of $h_{a,b}$ in $\mathcal{H}$ for which $x_1, x_2$ collide.
- **We will show that**
  - They can’t collide after performing $\mod p$.
  - So, they must map to different values $v_1, v_2$ at this point
  - Each $(v_1, v_2)$ corresponds to a unique pair $(a, b)$.
  - So, it suffices to count the number of pairs $(v_1, v_2)$ with $v_1 \neq v_2$, but $v_1 \equiv v_2 \ (\mod n)$
Proof that $\mathcal{H}$ is universal

- Define $h_{a,b}(x) = ((ax + b) \mod p) \mod n$

$$\mathcal{H} = \{ h_{a,b} \mid a \in [p - 1], 0 \leq b \leq p - 1 \}$$

**Claim 1.** If $x_1 \neq x_2$ then $ax_1 + b \neq ax_2 + b \pmod{p}$. 
Proof that $\mathcal{H}$ is universal

- Define $h_{a,b}(x) = ((ax + b) \mod p) \mod n$
  \[ \mathcal{H} = \{ h_{a,b} \mid a \in [p - 1], 0 \leq b \leq p - 1 \} \]

Claim 1. If $x_1 \neq x_2$ then $ax_1 + b \neq ax_2 + b \pmod p$.

Claim 2. For every pair $(v_1, v_2)$, where $v_1 \neq v_2$ and $0 \leq v_1, v_2 \leq p - 1$, \exists exactly one pair $(a, b)$:
  
  \[ ax_1 + b \equiv v_1 \pmod p; \]
  \[ ax_2 + b \equiv v_2 \pmod p. \]
Proof that $\mathcal{H}$ is universal

- Define $h_{a,b}(x) = ((ax + b) \mod p) \mod n$

  $\mathcal{H} = \{ h_{a,b} \mid a \in [p-1], 0 \leq b \leq p-1 \}$
Using a universal family

As before:
• If \( w \notin S \), expected number of words in bin \( h(w) \) is \( \leq \frac{m}{n} \).
• If \( w \in S \), expected number of words in bin \( h(w) \) is \( \leq 1 + \frac{m-1}{n} \).

The previous guarantee on max load no longer holds!

Goal for next time: Given \( S \), find a hash function with no collisions for words in \( S \).