Algorithm: Lecture 14 (2022-03-08)

**Today:** randomized edge

**Goal:** store seed vaccine ready

\[ \text{power, vaccine, etc.} \rightarrow \text{cleaner} \]

\\

1. A dictionary over \( U = \{0, \ldots, N-1\} \) is a data structure to store a set \( S \subseteq U \) of keys, along with associated values \( v \).

2. It supports:
   - \( \text{insert}(x, v) \): add key \( x \) to \( S \), with value \( v \).
   - \( \text{look-up}(x) \): decide if \( x \in S \), return value \( v \).

3. The complexity is measured in terms of \( n = |S| \). If \( c = \text{seed} \) then:

   - \( \text{insert}(x, v) \): \( \Omega(\log n) \)
   - \( \text{look-up}(x) \): \( \Omega(1) \)

4. If \( |S| < \log n \) then:

   - \( \text{insert}(x, v) \): \( \text{only need one random edge} \)
   - \( \text{look-up}(x) \): \( \text{only need one random edge} \)

5. If \( |S| = \log n \) then:

   - \( \text{insert}(x, v) \): \( \text{need } 2 \text{ edges, on average} \)
   - \( \text{look-up}(x) \): \( \text{need } 2 \text{ edge exchanges on average} \)
Q: Can we do this? 4 basis vectors in world 2

A: yes, via coordinatization.

A: yes, via coordinatization. A map, named the denominator.

Score re phone expressed as in many, 5.5, seen later something.

idea = simplify, I reduce using size via phone from 2

\[ n: U \rightarrow T \]

[T] = 5

so we connected an array data [T]

lookups (c)

Q: what is the problem?

A: cell tree

lookups (c)

\[ \text{lookups} (c) = \text{lookups} (c) + \text{lookups} (c) \]

lookup (c)

\[ \text{lookup} (c) = \text{lookup} (c) \]

Q: how are people stored?

A: hash table \[ h: U \rightarrow T \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

A: insert (key) takes all time plus 2 evaluations of \( h \) and check if \( h \) of element is in a list.

\[ \text{lookup} (c) = \text{lookup} (c) \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

Q: choose \( h \) so looks at small?

A: no single \( h \) can work to \[ S \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

\[ n: U \rightarrow T \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

\[ \text{lookup} (c) = \text{lookup} (c) \]

\[ \text{lookup} (c) = \text{lookup} (c) \]
idean. choose h randomly

\[ P \leq \frac{1}{d} \text{ if } \sup_{x \in S} \mathbb{E} \left[ \mathbb{1}_{\{ h(x) \neq h'(x) \}} \right] \leq \frac{1}{d} \]

where \( S \) is a set of size \( d \) and \( h \) and \( h' \) are two different hash functions.

Q: does this work?
A: no. spacious \( h: U \rightarrow T \) where \( U \) space is to big so as many collisions

idean: choose \( h \) pseudo-randomly

- "not too random" to avoid

def. A random hash family is a collection of hash functions

\[ H = \{ h : U \rightarrow T \} \]

such that \( \forall x, y \in U \), \( \Pr [ h(x) = h(y) ] = \frac{1}{T} \)

it (c) univ. Any \( x \in U \), \( \mathbb{E} \left[ \mathbb{1}_{h(x) \neq h'(x)} \right] \leq \frac{1}{T} \)

proof: p prime

\[ H : \mathbb{Z}_p^k \times \mathbb{Z}_p^k \rightarrow \mathbb{Z}_p \] given by \( h(x, y) = x \cdot y \text{ mod } p \)

\[ H = \{ h : \mathbb{Z}_p^k \rightarrow \mathbb{Z}_p \} \\ h(x) = h(x_1, x_2, \ldots, x_k) \] is a

univ. hash family

each \( h \in H \) can be stored in \( O(kc) \) space

proof: can be evaluated in \( O(kc) + k \)

f - space: \( n \) input \( y \) and \( k \) integers (unique per argument over \( \mathbb{Z}_p \))

\[ h(x) = \sum x \cdot h(x) \leq \sum x \cdot k \text{ op. } T \geq 2 \]

univ. - needs more room

define \( \mu_k : \mathbb{Z}_p \rightarrow \mathbb{Z}_p \) multiplication map

\[ y \mapsto xy \]

then \( \mu_k \) is invertible

\[ x = \mu_k^{-1}(y) \]

- \( x, y \in \mathbb{Z}_p \)

\[ \mu_k(y) = \mu_k(z) = xy = xz \quad (p) \]

\[ x(y - 1) = 0 \quad (p) \]

\[ x | (y - 1) \text{ or } x | y - 1 \]

\[ x \neq 0, p \Rightarrow y = x^2 \]