OH TODAY - 1:15 pm - 2:15 pm.

Hashing.

- data structure used to support dynamic set operation: Insert, Delete, Search.

$U$: universe of keys.

$U = \{0, 1, 2, \ldots, m-1\}$

Direct address (array)

```
T: [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]
0 1 2 3 4 5 6 7 8 9
```

$T[i]$ holds object with key $i$.

Insert ($T$, $x$)

\[
T[x, \text{key}] < x
\]

Search ($T$, $k$)

\[
O(1) \text{ in the worst case.}
\]
return \( T[k] \)

Delete \((T, x)\)

\( T[x \cdot \text{key}] \leftarrow \text{NIL} \).

What if \(|U|\) is very large?

or what if \(|U| \gg \# \text{keys actually stored in } T\)?

**Hash Table:**

\[ U \]

\[ \text{h} \]

\[ h(k) \]

\[ T \]

\( h(k) \)

\[ k \]

Since \(|U| > |T|\), by PHP, there will be
Collisions

multiple keys map to the same loc in T.

How do we resolve collisions?

**Using chaining.**

![Diagram of chaining]

Insert \((T, x)\)

- insert \(x\) at the head of
  - the linked list \(T[h(k)]\)

Search \((T, k)\)

- search for the object in the list \(T[h(k)]\).
Delete \((T,x)\)

- delete \(x\) from the list \(T[h(x\cdot\text{key})]\).

Analysis

\[ m : \lvert T \rvert \]

\[ n : \text{# keys stored in } T. \]

\[ \alpha : \text{load factor } = \frac{n}{m}. \]

**Simple Uniform Hashing Assumption:** "\(h\) is a good hash fn". In other words, any key is equally likely to be hashed to any of the \(m\) slots in \(T\).
Unsuccessful Search

Lemma: In hashing w/ chaining, assume the simple uniform hashing assumption, unsuccessful search takes $O(1 + \alpha)$ time in expectation.

Proof: Suppose we are searching for key $k$.

$$E[\text{time}] = \sum_{i=0}^{m-1} \Pr[k \text{ is mapped to } \text{ slot } i]$$

$$E[\text{time} | k \text{ is mapped to } i]$$

$$= \sum_{i=0}^{m-1} \frac{1}{m} \ E[|T(i)|]$$

$$= \sum_{i=0}^{m-1} \frac{1}{m} \cdot \frac{n}{m} \ (\text{using } 10/n)$$

$$= m \cdot \frac{n}{m^2} = \frac{n}{m}.$$
Assume \( h(k) \) takes \( \Theta(1) \) time, we have \( E[\text{search time}] = \Theta(1 + \frac{m}{w_m}) = \Theta(1+\alpha) \).

**Successful Search**

\[
\begin{align*}
\text{Knuth: } &\quad \Theta(1+\alpha) \\
T: &\quad \text{Diagram}
\end{align*}
\]

Support the elements are inserted \( n \times T \) in the order: \( x_1, x_2, \ldots, x_n \). We will assume that each of the \( n \) elements is equally likely to be searched.

\[E[\text{time}] = \sum_{i=1}^{n} Pr[x_i \text{ is searched}] \cdot E[\text{time} | x_i \text{ is searched}]\]
\[
\left( \frac{2n}{n(n+1)} \right) \left( \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n+1} \right) \cdot \frac{n}{\frac{n}{2} + 1} =
\]

\[
\frac{m}{1} \cdot \frac{1}{2} + \cdots + \frac{1}{n} =
\]

\[
\left( \frac{m}{1} \cdot \frac{1}{2} + \cdots + \frac{1}{n} \right) =
\]

\[
\bigg( \Phi \left( \frac{1}{x} \right) = 1 \bigg) + 1 \cdot \frac{m}{1} \cdot \frac{1}{2} =
\]

\[
\cdots 0.0.0.
\]

Based on the saw locus,

when \( X_{ab} = 1 \), if else and a 18g can...
\[= 1 + \frac{n}{2m} - \frac{n}{2m} - \frac{1}{2m}\]

\[= 1 + \frac{n}{2m} - \frac{n}{2m} \cdot \frac{1}{n}\]

\[= 1 + \frac{a^2}{2} - \frac{a^2}{2n}\]

\[\leq O\left(1 + x\right)\]

Suppose \(n = O(m)\) then the expected search time is \(O(1)\).

Hash funs

Suppose keys are uniformly distributed

\(b/w 0 \& 1\) then

\[0 \leq k < l\]
\[ h(k) = \lfloor k \mod m \rfloor \]

Division method:

\[ h(k) = k \mod m \]

(It is generally assumed that keys are positive integers).

In this case, it is not a good idea to choose \( m = 2^p \).

---

Generally, \( m \) is chosen to be a prime that is not too close to a power of 2.
**Multiplication method.**

\[ h(k) = \left\lfloor \frac{m(kA \mod 1)}{kA - \lfloor kA \rfloor} \right\rfloor \]

\[ A = \frac{\sqrt{5} - 1}{2} \]

**Open addressing.**

- All objects are stored in the table itself.
- What happens in case of a collision?

![Diagram of hash function and open addressing](image)
Find the "first" empty slot & insert $k$ there. We will modify our hash fn. as follows.

Until now:

$$h : U \rightarrow \{0, 1, \ldots, m-1\}$$

In open addressing

$$h : \{0, 1, \ldots, m-1\}^k \rightarrow \{0, 1, \ldots, m-1\}$$

We would like

$$\langle h(k,0), h(k,1), h(k,2), \ldots, h(k,m-1) \rangle$$

to be a permutation of the slots
\{0, 1, \ldots, m-1\}

Insert (T, k)

\[ i = 0 \]
\[ \text{repeat} \]
\[ \quad j = h(k, i) \]
\[ \quad \text{if } T[j] = \text{NIL} \text{ then} \]
\[ \quad \quad T[j] \leftarrow k \]
\[ \quad \text{return } j \]
\[ \quad i \leftarrow i + 1 \]
\[ \text{until} \quad i = m \]
\[ \text{print } "\text{Table is full}". \]

In open addressing, \( \leq m \)

Search (T, k)

\[ i = 0 \]
repeat
  \[ i = h(k, i) \]
  \[ j \]
  if \( T[j] = k \) then
    return \( j \)

\[ i \leftarrow i + 1 \]
until \( i = m \) or \( T[j] = \text{NIL} \)
\underline{Is no Delete.}

\underline{Analysis.}

Uniform Hashing assumption: each of the \( m! \) probe sequences is equally likely.

Linear probing
\[ h(k,i) = (h'(k) + i) \mod m. \]

# probe seq correspondly

Our hash fun: \( m. \)

\(< 18, 21, 1, 5, 6, 7, 8, 9 >. \)

\[ \frac{i}{m} \]

\(\text{NIL}\) element \( P_0 \) of an elem goes here

Start: \( h'(k) \)  
Offset: 1.

**Quadratic Probing**
\[ h(k, i) = \left( h(k) + c_1 + c_2 \right) \mod m. \]

**Double Hashing:**

\[ h(k, i) = \left( h_1(k) + i \cdot h_2(k) \right) \mod m. \]

Start: \( h_1(k) \), Offset: \( h_2(k) \).

\# probes avg: \( m^2 \).