Admin:

Today:
Hashing: Universal
   Perfect
"Dictionary": Abstract Data Type:

- supporting 1. create
- 2. insert(x)
- 3. search(key)
- 4. delete(x)

- Assume elements have distinct keys
- Note: balanced BST's solve in time $O(\log n)$ per op.
  (plus they do other ops, like "successor", though)
  **Can we do better? Yes!**

**Hashing:** $O(1)$ time per op; $O(n)$ space for n items

- $n =$ # keys in table
- $m =$ # slots in table

$h =$ hash fn mapping keys $\rightarrow 0, 1, \ldots, m - 1$  "pseudo-random"

```
\[ \begin{cases} 
    \text{items } x \\
    \text{s.t. } h(x, key) = j \\
    \text{in linked list from } j^{th} \text{ slot} \\
\end{cases} \]
```

"hashing with chaining"
Load factor $\alpha = n/m = E(n_j)$
where $n_j =$ length of $j$th list

Thm: Hashing with chaining achieves time (on average) \( \Theta(1+\alpha) \) per operation (insert, unsuccessful search, successful search, delete)

[Successful search is most interesting since longer lists more likely to be searched. See Thm 11.3 in book.]

Note problem: worst-case can be bad (all keys have same hash value) \( \Rightarrow \Theta(n) \) time (average & w.c) for that set of keys.
We'll fix this with universal hashing.
Distribution of list lengths $\mathbb{E}[n_j^2]$; average is $\alpha$.

Note: $e^\alpha = \sum_{k=0}^{\infty} \frac{\alpha^k}{k!}$ \hspace{1cm} (\forall k \geq 1 \text{ if } k=0)

Poisson dist with mean $\alpha$

$$\Pr \{X = k\} = \frac{e^{-\alpha} \alpha^k}{k!}$$

Good approximation to distribution of lengths (with random $h$)

E.g. Suppose $n = m$ so $\alpha = 1$

$\Pr(n_j = 0) = \frac{e^{-1} \alpha^0}{0!} = \frac{1}{e} = 0.3679 \quad 37\%$ empty

$\Pr(n_j = 1) = \frac{e^{-1} \alpha^1}{1!} = \frac{1}{e} = 0.3679 \quad 37\%$ size 1

$\Pr(n_j = 2) = \frac{e^{-1} \alpha^2}{2!} = \frac{1}{2e} = 0.1839 \quad 18\%$ size 2

$\Pr(n_j = 3) = \frac{e^{-1} \alpha^3}{3!} = \frac{1}{6e} = 0.0613 \quad 6\%$ size 3

$\Pr(n_j = 4) = \frac{e^{-1} \alpha^4}{4!} = \frac{1}{24e} = 0.0153 \quad 1.5\%$ size 4

\vdots

Thm: Let $n_* = \max_j n_j$ (for $\alpha = 1$)

Then $n_* \leq 3 \ln(n) / \ln \ln n$ with probability $\geq 1 - \frac{1}{n}$. 

[Ref: Probability & Computing by Mitzenmacher & Upfal]
Universal hashing
- gets good $O(1 + \alpha)$ performance for any fixed set of keys & searches (no bad inputs)
- Idea: choose hash function randomly when table is created, from a family of universal hash functions.

Let $\mathcal{H}$ be a family of hash functions, mapping into $\{0, 1, \ldots, m-1\}$.

**Def:** $\mathcal{H}$ is universal if for all keys $k_1, k_2$ ($k_1 \neq k_2$)

$$\Pr_{h \in \mathcal{H}} \{ h(k_1) = h(k_2) \} \leq \frac{1}{m}$$

Probability that $k_1, k_2$ collide (end up on same list) is $\leq 1/m$. 
Thm: Let $k_1, k_2, \ldots, k_n$ be distinct keys, $h \in$ universal $\mathcal{H}$

Let $k$ be any key (perhaps $= k_i$, perhaps not)

Then $E\left( | \{ k_i : h(k_i) = h(k) \} \right) \leq n/m$

Pf: Each other key $k_i$ is on $k$'s list with probability $1/m$.
Use linearity of expectation (see book). 

Thus Insert, Delete, Search cost $O(1 + \alpha)$ for
any set of keys (on average, over choice of $h \in \mathcal{H}$).

[Note: "1" is for key $k$ itself for successful search, and
for time to compute $h$.]

Do universal hash families exist? Yes!
Suppose \( m = p \) (a prime).

Consider \( h = h_{ab} \) where 
\[
h(x) = a \cdot x + b \pmod{p}
\]
\( a \neq 0, \ 0 \leq b < p. \)

\[
\mathcal{H} = \{ h_{ab} \} \quad |\mathcal{H}| = (p-1) \cdot p
\]

"lines" mod \( p \)

\[\begin{array}{c}
\text{Thm: } \mathcal{H} \text{ is universal (mapping } \mathbb{Z}_p \text{ into } \mathbb{Z}_p) \\
\text{Pf: } \text{Let } k, l \text{ be two keys, } k \neq l.
\end{array}\]

Given \( h_{ab} \), we can compute
\[
h_{ab}(k) = a \cdot k + b \pmod{p} = r
\]
\[
h_{ab}(l) = a \cdot l + b \pmod{p} = s
\]

\( r \neq s \) (\( h_{ab} \) is 1 to 1)

since \( r - s = a \cdot (k - l) \neq 0 \pmod{p} \)

since \( a \neq 0, l \neq k \)

So \( \text{prob}\{h_{ab}(k) = h_{ab}(l)\} = 0 \) !

But here key space has size \( p \), and memory has size \( p \).

What if we want \( p \) keys, but memory \( m < p \) ?
Thm: Suppose $p$ is prime, and $m < p$. Then

$$H = \left\{ h_{ab} : h_{ab}(x) = h_{ab}(x) \mod m^2 \right\}$$

is universal. [Note $a \in \mathbb{Z}_p^*$, $b \in \mathbb{Z}_p$ as before, so $|H| = p \cdot (p-1)$]

(Just reduce what we had, mod $m$.)

PF: Fix $k, l$ keys (mod $p$), $k \neq l$.

Then $r = h_{ab}(k)$, $s = h_{ab}(l)$ are r.v.'s & $r \neq s$.

All pairs $(r, s)$ s.t. $r \neq s$ are equally likely.

For any fixed $r$, # values of $s$ that are $\equiv r \mod m$ is $\left\lceil \frac{p}{m} \right\rceil - 1$ (the -1 since $r \neq s \mod p$).

But $\left\lceil \frac{p}{m} \right\rceil - 1 \leq \frac{(p-1)}{m}$

There are $p-1$ choices for $s$, but prob $= r \mod m$ is $\leq 1/m$. 

Perfect Hashing

- When set of keys is static, we can get time/operation down to $O(1)$ in worst-case.
- Uses universal hashing in two-level scheme
  
  Given keys $k_1, k_2, \ldots, k_n$

  ![Diagram]

  Store $k_i$ in position $h_{2,j}(k_i)$ of $j^{th}$ level, where $j = h_1(k_i)$.

  - First-level table has size $m = n$.
    First level hash fn drawn from universal family mapping to $0, 1, \ldots, m-1$. Each entry specifies: 2nd-level table of size $n_j^2$.
    Hash function $h_{2,j}$ for this table.

  - Second-level table has size $n_j^2$. 
Total space used = $\Theta(n + \sum_{j=0}^{m-1} \eta_j^2)$

Theorem 1: We can easily find $h_1$, s.t. space = $\Theta(n)$

Theorem 2: For each $j$, we can easily find $h_{2,j}$ s.t. $h_{2,j}$ is collision-free on keys with $h(k) = j$.

Thus: overall space = $\Theta(n)$
lookup time is $\Theta(1)$
**Proof of Theorem 2: (≈ Birthday Paradox)**

\[
\Pr \left\{ h_{x,j} (k_i) = h_{x,j} (k_{i'}) \text{ for some } i \neq i' \text{ in set } j \right\}
\]

\[
\leq \sum_{i \neq i'} \Pr \left\{ h_{x,j} (k_i) = h_{x,j} (k_{i'}) \right\} \quad \text{(Union bound)}
\]

\[
\leq \binom{n_j}{2} \cdot \frac{1}{n_j^2}
\]

\[
< \frac{1}{2}
\]

Therefore, each \( h_{x,j} \) we try has \( \frac{1}{2} \) chance of being collision-free. Only need to try 2 such \( h_{x,j} \)s, at most, on average.
Proof of Theorem 1:

\[ \Pr \left\{ \sum_{j=0}^{m-1} \eta_j^2 \geq c \cdot n \right\} \leq \frac{E \left[ \sum_{j=0}^{m-1} \eta_j^2 \right]}{c \cdot n} \]

Markov's Inequality

\[
E \left[ \sum_{j=0}^{m-1} \eta_j^2 \right] = E \left[ \sum_{i=1}^{n} \sum_{j=0}^{n} I_{i,j,i'} \right]
\]

where \( I_{i,j,i'} = \begin{cases} 
1 & \text{if } h_i(k_i) = h_{i'}(k_{i'}), \\
0 & \text{else}
\end{cases} \)

\[
= \sum_{i=1}^{n} \sum_{j=0}^{n} E \left[ I_{i,j,i'} \right] \quad \text{(linearity of } E) \]

\[
= \sum_{i=1}^{n} E \left[ I_{i,i,i'} \right] + 2 \sum_{i \neq i'} E \left[ I_{i,i,i'} \right] \quad \text{(universality)}
\]

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

\[
= O(n) \quad \text{(since } m=n) \]

\[ : E \left[ n + \sum_{j=0}^{m-1} \eta_j^2 \right] \leq 3n \]

\[ : \Pr \left\{ n + \sum_{j=0}^{m-1} \eta_j^2 \geq 6n^2 \right\} \leq \frac{1}{2} \]

Thus, each \( h_i \) we try has prob \( \geq \frac{1}{2} \) of having space \( \leq 6n \).

Only need to try 2 such fn's on average.