For every key \( k \), probe sequence \( \langle h(k), h(k)_2, \ldots, h(k)_m \rangle \). 

\[
\begin{align*}
n : 0 \times 1, 0, 1, \ldots, m-1 \rightarrow 0, 1, 2, \ldots, m-1 \\
\end{align*}
\]

- Load factor \( \alpha \) can never exceed 1.
- Systematically examine table slots for searching an element.
- Open addressing
  - Overflow
  - Perfect hashing
  - Open Addressing

6.006 Recitation 8 27 February 2009
Return NIL

else if !+1

Return NIL

if f{x} == NIL

Return !+1

if f{x} == k

while i < m

x = \text{HASH-SEARCH}(T, k)

Return !+1

- prefer chaining when keys are to be deleted.
- no longer dependent on a
- mark the deleted set as deleted instead of NIL
m possible distinct probe sequences

- secondary clustering: h(K, 0) = h(K, 0) \Rightarrow h(K') = h(K')

- better than linear probing, but c/m is constrained

h(K') = \lfloor \frac{(c_1 + c_2 + c_3 + \cdots)}{m} \rfloor

- quadratic probing

- primary clustering problem: counter example

m possible values

determines the entire sequence

\[ \left\lfloor \frac{h(K) + c_1}{m} \right\rfloor \]

h(K') = \left\lfloor \frac{h(K) + c_1}{m} \right\rfloor

- linear probing
\( m \in \mathbb{Z} \backslash \{0\} \Rightarrow \exists \text{ distinct prime sequences } (h_1, h_2, (f)) \)

\( m \in \mathbb{Z}, \ h_1 \rightarrow m \)

\( m = 2, \ h_2 \rightarrow \text{odd number} \)

\( \text{relatively prime to } m \)

\( h'(f') = (h_1, C^2, h_2 \cdot (f)) \mod m \)

Double Hashing
Given an open address hash table with load factor \( \alpha = n/m < 1 \), expected number of probes in an unsuccessful search is \( 1/\alpha \) assuming uniform hashing.

A !: there is an occupied slot.

Random variable \( X \) : number of probes made in an unsuccessful search.

\[
Pr(A \cap \{X = 0\}) = \frac{m - 1}{m} \quad Pr(A \cap \{X = 1\}) = \frac{m - 2}{m} \quad Pr(A \cap \{X = 2\}) \quad \cdots \\
Pr(A \cap \{X = n - 1\}) = \frac{1}{m}
\]

\[
xm = \frac{m - 1}{m} \quad \text{for } m \geq n \geq 1 \quad \text{if } m > n \geq 1
\]

\[
Pr(A \cap \{X \geq 1\}) = \frac{1}{m} (\frac{m}{m - 1}) (\frac{m - 1}{m - 2}) \quad \cdots \\
Pr(A \cap \{X \geq n - 1\}) = \frac{1}{m} (\frac{m}{m - 1}) (\frac{m - 1}{m - 2}) \quad \cdots \\
Pr(A \cap \{X \geq 1\}) = \frac{1}{m} (\frac{m}{m - 1}) (\frac{m - 1}{m - 2}) \quad \cdots \\
Pr(A \cap \{X \geq n - 1\}) = \frac{1}{m} (\frac{m}{m - 1}) (\frac{m - 1}{m - 2}) \quad \cdots \\
Pr(A \cap \{X \geq n - 1\}) = \frac{1}{m} (\frac{m}{m - 1}) (\frac{m - 1}{m - 2}) \quad \cdots \\
\]
In a successful search: 

- \( T = (1 - \alpha)^{-1} \) in an open-addressed hash table with sharedeced a, expected number of people inserting an element into an open-addressed hash table requires at most \( 1 - \alpha \).

\[
\begin{align*}
\frac{1}{T} &= \frac{1}{1 + \alpha + \alpha^2 + \alpha^3 + \cdots} \\
&= \frac{1}{1 - \alpha} \\
&= \frac{1}{1 - \alpha} = \frac{1}{N} \\
&= \frac{1}{2} \\
\mathbb{P}(X \geq 2) &= \frac{1}{N} = \frac{1}{2}.
\end{align*}
\]
\[ \frac{\alpha}{\frac{1}{T}} = \frac{(H_m - H_{m-n})}{\frac{1}{m}} \]

\[ \prod_{k=0}^{m-n-1} \left( 1 + \frac{1}{T} + \cdots + \frac{1}{T^{m-1}} \right) \frac{\sum_{\ell=0}^{m-1}}{m} \]

\[ \frac{\sum_{\ell=0}^{m-1}}{m} \]

After taking logs and using

expected runtime of probe made in searching for k, \( k \)

Key T, \((m-1)!\)st Key inserted into hash table,
\[ \frac{\nu + 1}{\nu - m} \frac{1}{\nu - m} \frac{\varphi}{\gamma} = \frac{\nu p(\nu/1)}{\nu} \sum_{\nu}^{\nu + 1} \varphi \frac{1}{\nu (\nu - m + n)} = 0 \]
Two-Level Hashing Scheme

First Level: n keys hashed into m slots

Secondary Hash Table: S_i with h_i - no collisions

Total space: O(nm), m = O(n^{1/2})

To perform search O(1).

- Set of reserved words in programming languages
- Set of files names on a cd-rom.

Perfect Hashing
If no more likely than not to have no collisions.

\[ \frac{2}{\binom{n}{2}} > \frac{1}{2^{n/2}} \]

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

\[ \frac{2}{\binom{n}{2} x} = \frac{n-1}{2} \]

\[ m \approx n^{1/2} \cdot x \cdot \text{number of collisions} \]

2 \text{ nc}_2 \text{ pairs of keys that can collide, each pair collided with probability } p \]

The probability any collisions is \( \ll \frac{1}{2} \).

Randomly chosen from a universal class of hash functions, probability of

If \( m \) we store \( n \) keys in a hash table of size \( m = n \), using a hash function
\[ \sum_{m=1}^{\infty} \frac{1}{m^2} \left( \sum_{n=1}^{m} \frac{1}{n^2} \right) = \frac{\pi^4}{90} = \frac{\pi^2}{6} + \frac{1}{2} \left( \frac{\pi^2}{6} \right) \]

A 2D, \quad a_z = a + 2c z = a + 2c (z - 1) = a + 2 \left( \frac{z}{2} - 1 \right)

If we store n keys in a hash table of size \( m = n \),

when in cache, \( m = n^2 \) to excess use.
If we store \( n \) keys in a hash table of size \( m \) = \( n \), using a hash function randomly chosen from a universal class of hash functions, and we test the size of each

secondary hash table to \( m_i = n \), then the probability that the total size of each

secondary hash table exceeds \( m = n \) is

\[
\frac{2^n}{\frac{m!}{2^n}} \leq \frac{2^n}{n!} \leq \frac{2^n}{\frac{n(n-1)}{2}} = \frac{2^n}{\frac{n(n-1)}{2}}
\]

and

\[
\sum_{m=1}^{n} \frac{2^n}{\frac{m!}{2^n}} \leq \sum_{m=1}^{n} \frac{2^n}{\frac{n(n-1)}{2}} = \sum_{m=1}^{n} \frac{2^n}{\frac{n(n-1)}{2}}
\]

\[
\frac{2^n}{m!} \leq \frac{2^n}{n!} \leq \frac{2^n}{\frac{n(n-1)}{2}}
\]

\[
\sum_{m=1}^{n} \frac{2^n}{\frac{m!}{2^n}} \leq \sum_{m=1}^{n} \frac{2^n}{\frac{n(n-1)}{2}} = \sum_{m=1}^{n} \frac{2^n}{\frac{n(n-1)}{2}}
\]