6.006 Recitation 8 — 27 February 2009

Note Title — 2/27/2009

* Open Addressing
* Perfect Hashing

Open Addressing

- all elements stored in the hash table itself.
- systematically examine table slots for searching an element
- load factor α can never exceed 1

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

For every key k, probe sequence $\langle h(k,0), h(k,1), \ldots, h(k,m-1) \rangle$.

should be a permutation of $\langle 0, 1, 2, \ldots m-1 \rangle$

```
HASH-INSERT(T,k)
  i = 0
  while i < m
      j = h(k,i)
      if T[j] == NIL
          T[j] = k
          return j
      else  i = i+1
```

```
HASH-SEARCH(T,k)
  i = 0
  while i < m
      j = h(k,i)
      if T[j] == k
          return j
      if T[j] == NIL
          return NIL
      i = i+1
  return NIL
```

DELETION

mark the deleted slot as DELETED instead of NIL

- no longer dependent on α
- prefer chaining when keys are to be deleted.

Linear Probing

$h' : U \to \{0, 1, \dots, m-1\}$

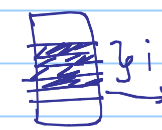
$h(k,i) = (h'(k) + i) \bmod m$

$\langle T[h'(k)], T[h'(k)+1], \dots \rangle$

determines the entire sequence

m possible values

→ Primary Clustering problem : clusters arise



fill probability $\left(\frac{i+1}{m}\right)$

Quadratic Probing

$h(k,i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$

→ better than linear probing, but c_1, c_2 & m constrained

→ secondary clustering : if $h(k_1,0) = h(k_2,0) \Rightarrow h(k_1,i) = h(k_2,i)$

m possible distinct probe sequences

Double Hashing

$h(k,i) = (h_1(k) + i\,h_2(k)) \bmod m$

→ $h_2(k)$ relatively prime to m

* $m = 2^p$, $h_2 \rightarrow$ odd number
* $m \rightarrow$ prime, $h_2 \rightarrow \, < m$

* $\Theta(m^2)$ probe sequences $(h_1(k), h_2(k)) \Rightarrow$ distinct probe sequence

Given an open-address Hash Table with load factor $\alpha = n/m < 1$, expected number of probes in an unsuccessful search is $\leq 1/(1-\alpha)$ assuming uniform hashing.

random variable X : number of probes made in an unsuccessful search

A_i : there is an i^{th} probe and it is an occupied slot.

$$\begin{aligned}
Pr(X \geq i) &= Pr(A_1 \cap A_2 \cap A_3 \cdots \cap A_{i-1}) \\
&= Pr(A_1)\, Pr(A_2 \mid A_1)\, Pr(A_3 \mid A_2 \cap A_1) \cdots Pr(A_{i-1} \mid A_1 \cap A_2 \cap \cdots A_{i-2}) \\
&= \frac{n}{m} \cdot \frac{n-1}{m-1} \cdot \frac{n-2}{m-2} \cdots \frac{n-i+2}{m-i+2} \\
&\leq \left(\frac{n}{m}\right)^{i-1} \\
&= \alpha^{i-1}
\end{aligned}$$

$Pr(A_1) = n/m$

$$Pr(A_j \mid A_{j-1} \cap A_{j-2} \cdots \cap A_1) = \frac{n-j+1}{m-j+1}$$

$$n < m \Rightarrow \frac{n-j}{m-j} \leq \frac{n}{m} \quad 0 \leq j < m$$

$$E[X] = \sum_{i=0}^{\infty} \Pr[X \geq i]$$

$$= \sum_{i=0}^{\infty} \alpha^{i-1}$$

$$= 1 + \alpha + \alpha^2 + \alpha^3 + \dots$$

$$= \frac{1}{1-\alpha}$$

$1+\alpha+\alpha^2+\alpha^3+\alpha^4+\dots$

1 probe is always made

with probability α, first probe finds occupied slot

Inserting an element into an open-address hash table requires at most $1/1-\alpha$ probes on average, assuming uniform hashing.

Given an open-address hash table with load factor α, expected number of probes in a successful search:

$$\frac{1}{\alpha} \ln \frac{1}{1-\alpha}$$

assuming uniform hashing and each key is equally likely to search for.

Key k, $(i+1)^{st}$ key inserted into hash table,

expected number of probes made in searching for $k \le \frac{1}{1-i/m} = \frac{m}{m-i}$

Averaging over all n keys

$$\frac{1}{n}\sum_{i=0}^{n-1}\frac{m}{m-i} = \frac{m}{n}\sum_{i=0}^{n-1}\frac{1}{m-i} = \frac{m}{n}\left(\frac{1}{m}+\frac{1}{m-1}+\cdots+\frac{1}{m-n+1}\right)$$

$$= \frac{m}{n}\left(1+\frac{1}{2}+\cdots+\frac{1}{m-1}+\frac{1}{m} - \left(1+\frac{1}{2}+\cdots+\frac{1}{m-n}\right)\right)$$

$$= \frac{m}{n}\left(H_m - H_{m-n}\right) \qquad H_i = \sum_{k=0}^{i}\frac{1}{k}$$

$$= \frac{1}{\alpha}\left(H_m - H_{m-n}\right)$$

$$\frac{1}{\alpha}\left(H_m - H_{m-n}\right) = \frac{1}{\alpha}\sum_{k=m-n+1}^{m} 1/k$$

$$\leq \frac{1}{\alpha}\int_{m-n}^{m} (1/x)\,dx$$

$$= \frac{1}{\alpha}\Big[\ln x\Big]_{m-n}^{m}$$

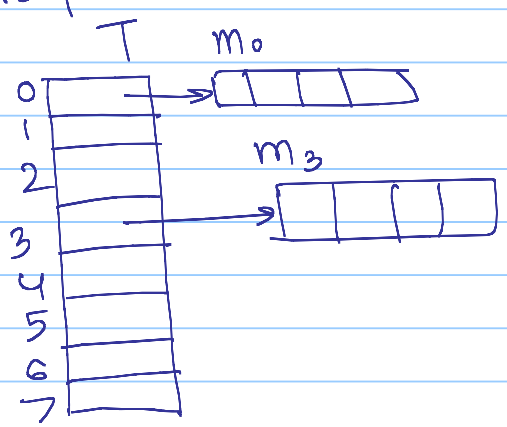
$$= \frac{1}{\alpha}\ln\frac{m}{m-n}$$

$$= \frac{1}{\alpha}\ln\frac{1}{1-\alpha}$$

Perfect Hashing

Excellent worst-case performance guarantee when keys are static
- set of reserved words in programming languages
- set of file names on a CD-ROM.

To perform search $O(1)$.



Two level hashing scheme

First level : n keys hashed into m slots

Secondary Hash table S_j with $h_j \to$ no collisions

$m_j = O(n_j^2)$

Total space $\to O(n)$

If we store n keys in a hash table of size $m = n^2$, using a hash function randomly chosen from a universal class of hash functions, probability of there being any collisions is $< 1/2$.

→ ${}^nC_2$ pairs of keys that can collide ; each pair collides with probability $1/m$

$m = n^2$, X: number of collisions

$$E[X] = {}^nC_2 \times \frac{1}{m}$$
$$= \frac{n(n-1)}{2} \times \frac{1}{n^2}$$
$$= \frac{1 - 1/n^2}{2}$$
$$< \frac{1}{2}$$

h is more likely than not to have NO collisions.

When n is large, $m = n^2$ is excessive.
If n_j keys hash to slot j, secondary hash table of size $m_j = n_j^2$ used.

If we store n keys in a hash table of size $m = n$,

$$E\left[\sum_{j=0}^{m-1} n_j^2\right] < 2n$$

$\forall\, a > 0$, $\quad a^2 = a + 2\,{}^{a}C_2 = a + 2\frac{a(a-1)}{2} = a + a^2 - a = a^2$

$$E\left[\sum_{j=0}^{m-1} n_j^2\right] = E\left[\sum_{j=0}^{m-1}\left(n_j + 2\,{}^{n_j}C_2\right)\right]$$

$$= E\left[\sum_{j=0}^{m-1} n_j\right] + 2E\left[\sum_{j=0}^{m-1} {}^{n_j}C_2\right]$$

$$= E[n] + 2E\left[\sum_{j=0}^{m-1} {}^{n_j}C_2\right]$$

$$E\left[\sum_{j=0}^{m-1} n_j^2\right] = n + 2E\left[\sum_{j=0}^{m-1} {}^{n_j}C_2\right]$$

total number of collisions

$${}^{n}C_2 \frac{1}{m} = \frac{n(n-1)}{2m} = \frac{n-1}{2}\Big|_{m=n}$$

$$E\left[\sum_{j=0}^{m-1} n_j^2\right] \leq n + 2\left(\frac{n-1}{2}\right)$$
$$\leq 2n-1$$
$$< 2n$$

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 set the size of each secondary hash table to $m_j = n_j^2$, then the probability that the total storage used for secondary hash tables equals or exceeds $4n$ is $< 1/2$.

$$\Pr\left\{\sum_{j=0}^{m-1} m_j \geq 4n\right\} \leq \frac{E\left[\sum_{j=0}^{m-1} m_j\right]}{4n} < \frac{2n}{4n} = 1/2$$