Risoluzione delle collisioni con indirizzamento aperto

Con la tecnica di *indirizzamento aperto* tutti gli elementi stanno nella tavola.
La funzione hash non individua una singola cella ma un ordine in cui ispezionare tutte le celle.

L'inserimento di un elemento avviene nella prima cella libera che si incontra nell'ordine di ispezione.

Nella ricerca di un elemento si visitano le celle sempre nello stesso ordine.

La funzione hash è una funzione $h(k,i)$ che al variare di i tra 0 ed $m-1$ fornisce, per ciascuna chiave k, una sequenza di indici

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

che rappresenta l'ordine di ispezione.

Siccome vogliamo poter ispezionare *tutte* le celle, la sequenza deve essere una permutazione dell'insieme degli indici $0,1,\ldots,m-1$ della tavola.

La realizzazione delle operazioni è:

```
Insert(T, k)
  i = 0
  repeat
     j = h(k, i)
    if T[ j ] == nil
        T[ j ] = k
        return  j
    i = i +1
  until i == m
  "Errore : tavola piena"
```

```
Search(T, k)
  i = 0
  repeat
     j = h(k, i)
     if T[ j ] == k
        return  j
     i = i +1
  until i == m or T[ j ] == nil
```

La realizzazione di ***Delete*** è più complicata
Non possiamo infatti limitarci a porre ***nil*** nella cella!!! Perché ???

La ***Delete*** si limita ad assegnare alla chiave dell'elemento da togliere un particolare valore diverso da ogni possibile chiave:

***Delete*(*T*, *i*)**

T*[*i*] = *deleted

La ***Search*** continua a funzionare invariata:

```
Search(T, k)
  i = 0
  repeat
     j = h(k, i)
     if T[ j ] == k
        return  j
     i = i +1
  until i == m or T[ j ] == nil
```

La ***Insert*** deve essere modificata:

```
Insert(T, k)
  i = 0
  repeat
    j = h(k, i)
    if T[ j ] == nil
        or T[ j ] == deleted
      T[ j ] = k
      return  j
    i = i+1
  until i == m
  "Errore: tavola piena"
```

Con l'indirizzamento aperto la funzione hash fornisce una sequenza di ispezione.

In questo caso l'ipotesi di hash uniforme diventa:

"Ogni chiave ha la stessa probabilità $\mathbf{1/\boldsymbol{m}!}$ di generare una qualsiasi delle $\boldsymbol{m}!$ possibili sequenze di ispezione"

Vi sono tre tecniche comunemente usate per determinare l'ordine di ispezione:

1. Ispezione lineare
2. Ispezione quadratica
3. Doppio hash

Nessuna delle tre genera tutte le $m!$ sequenze di ispezione.
Le prime due ne generano soltanto m e l'ultima ne genera m^2.

Ispezione lineare

La funzione hash ***h*(*k*,*i*)** si ottiene da una funzione hash ordinaria ***h*'(*k*)** ponendo

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

L'esplorazione inizia dalla cella ***h*(*k*,0) = *h*'(*k*)** e continua con le celle ***h*'(*k*)+1**, ***h*'(*k*)+2**, ecc. fino ad arrivare alla cella ***m*-1**, dopo di che si continua con le celle **0**,**1**,ecc. fino ad aver percorso circolarmente tutta la tavola.

L'ispezione lineare è facile da implementare ma soffre del problema dell'addensamento primario:

"i nuovi elementi inseriti nella tavola tendono ad addensarsi attorno agli elementi già presenti"

Una cella libera preceduta da t celle occupate ha probabilità $(t+1)/m$ di venir occupata dal prossimo elemento inserito.

Quindi sequenze consecutive di celle occupate tendono a diventare sempre più lunghe.

Ispezione quadratica

La funzione hash $\boldsymbol{h(k, i)}$ si ottiene da una funzione hash ordinaria $\boldsymbol{h'(k)}$ ponendo

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

dove $\boldsymbol{c_1}$ e $\boldsymbol{c_2}$ sono due costanti con $\boldsymbol{c_2 \neq 0}$.

I valori di $\boldsymbol{m}$, $\boldsymbol{c_1}$ e $\boldsymbol{c_2}$ non possono essere qualsiasi ma debbono essere scelti opportunamente in modo che la sequenza di ispezione percorra tutta la tavola.

Un modo per fare ciò è suggerito nel problema 11-3 del libro.

Osserviamo che se $h'(j) = h'(k)$ anche le due sequenze di ispezione coincidono.

Questo porta ad un fenomeno di addensamento secondario (meno grave dell'addensamento primario).

L'addensamento secondario è dovuto al fatto che il valore iniziale $h'(k)$ determina univocamente la sequenza di ispezione e pertanto abbiamo soltanto m sequenze di ispezione distinte.

Problema 11-3 del libro:
Consideriamo la seguente procedura:

$j = h'(k)$
$i = 0$
while $i < m$ **and** *"T[j] non è la cella cercata"*
  $i = i+1$
  $j = (j + i) \bmod m$

Dimostrare che la sequenza delle j che viene generata è una sequenza di ispezione quadratica.

Dobbiamo dimostrare che esistono due costanti c_1 e c_2 con $c_2 \neq 0$ tali che

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

sia una invariante del ciclo.

Calcoliamo i primi valori di h:

Per $i = 0$ $\quad j = h'(k)$

Per $i = 1$ $\quad j = [h'(k) + 1] \bmod m$

Per $i = 2$ $\quad j = [h'(k) + 1 + 2] \bmod m$

Per $i = 3$ $\quad j = [h'(k) + 1 + 2 + 3] \bmod m$

e in generale

$$j = [h'(k) + 1 + 2 + \ldots + i] \bmod m$$

Quindi

$$j = \left[h'(k) + \frac{i(i+1)}{2} \right] \bmod m$$

$$= \left[h'(k) + \frac{1}{2} i + \frac{1}{2} i^2 \right] \bmod m$$

Doppio hash

La funzione hash $h(k,i)$ si ottiene da due funzione hash ordinarie $h_1(k)$ ed $h_2(k)$ ponendo

$$h(k,i) = [h_1(k) + ih_2(k)] \bmod m$$

Perché la sequenza di ispezione percorra tutta la tavola il valore di $h_2(k)$ deve essere relativamente primo con m (esercizio 11.4-3 del libro).

Possiamo soddisfare questa condizione in diversi modi.

Possiamo scegliere $m = 2^p$ potenza di 2 ed

$$h_2(k) = 2\ h'(k) + 1$$

con $h'(k)$ funzione hash qualsiasi per una tavola di dimensione $m' = m/2 = 2^{p-1}$.

Un altro modo è scegliere m primo e scegliere $h_2(k)$ che ritorna sempre un valore minore di m.

Un esempio è:

$$h_1(k) = k \bmod m \quad \text{e} \quad h_2(k) = 1 + (k \bmod m')$$

dove m' è minore di m (di solito $m' = m-1$).

Con l'hash doppio abbiamo $\Theta(m^2)$ sequenze di ispezione distinte.

Questo riduce notevolmente i fenomeni di addensamento e rende il comportamento della funzione hash molto vicino a quello ideale dell'hash uniforme.

Analisi dell'indirizzamento aperto

Assumiamo l'ipotesi di hash uniforme, ossia che ogni permutazione di 0,1,..., m-1 sia ugualmente probabile come ordine di ispezione.

Valutiamo la complessità media di *Search* in funzione del *fattore di carico* $\alpha = n/m$.

Notiamo che con l'indirizzamento aperto $n \leq m$ e quindi $0 \leq \alpha \leq 1$.

Proprietà: Assumendo l'ipotesi di hash uniforme, il numero medio di celle ispezionate nella ricerca di una chiave *k* *non presente* in una tavola hash con indirizzamento aperto è *m* se $\alpha = 1$ e al più $1/(1-\alpha)$ se $\alpha < 1$.

Dimostrazione: Se $\alpha = 1$ non ci sono celle vuote e la ricerca termina dopo aver ispezionato tutte le *m* celle.

Se $\alpha < 1$ la ricerca termina con la prima cella vuota incontrata durante la sequenza di ispezione.

Per l'ipotesi di hash uniforme la prima cella ispezionata può essere con uguale probabilità una qualsiasi delle m celle.

Siccome ci sono n celle occupate la probabilità che la prima cella ispezionata risulti occupata e che quindi si debba ispezionare anche la successiva è $\alpha = n/m$.

La probabilità che si debba ispezionare una terza cella è la probabilità $\alpha = n/m$ che la prima cella risulti occupata moltiplicata per la probabilità $(n\text{-}1)/(m\text{-}1)$ che anche la seconda cella risulti occupata, ossia

$$\frac{n}{m} \cdot \frac{n-1}{m-1} < \alpha^2$$

In generale la probabilità che si debba ispezionare la i-esima cella della sequenza è

$$\frac{n}{m} \cdot \frac{n-1}{m-1} \cdot \ldots \cdot \frac{n-i+1}{m-i+1} < \alpha^{i-1}$$

Dunque noi ispezioniamo una prima cella con probabilità 1, una seconda cella con probabilità α, una terza cella con probabilità minore di α^2, una quarta con probabilità minore di α^3 e così via.

Il numero atteso di celle ispezionate è quindi minore di

$$1 + \alpha + \alpha^2 + \alpha^3 + \ldots + \alpha^{m-1} =$$

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

Conseguenza : Assumendo l'ipotesi di hash uniforme, il numero medio di celle ispezionate quando inseriamo una nuova chiave in una tavola hash con indirizzamento aperto è m se $\alpha = 1$ e al più $1/(1-\alpha)$ se $\alpha < 1$.

Proprietà: Assumendo l'ipotesi di hash uniforme, il numero medio di celle ispezionate nella ricerca di una chiave k *presente* in una tavola hash con indirizzamento aperto è $(m+1)/2$ se $\alpha = 1$ e al più $1/\alpha \ln [1/(1-\alpha)]$ se $\alpha < 1$.

Dimostrazione: Se $\alpha = 1$ la chiave cercata può trovarsi, con uguale probabilità, nella prima, seconda, ..., ultima cella e quindi il numero medio di celle ispezionate è

$$\frac{1}{m}\sum_{i=1}^{m} i = \frac{1}{m}\cdot\frac{m(m+1)}{2} = \frac{m+1}{2}$$

Se $\alpha < 1$ la ricerca ispeziona le stesse celle visitate quando la chiave cercata è stata inserita nella tavola.

Supponiamo che la chiave cercata sia stata inserita dopo altre i chiavi.

Il numero medio di celle ispezionate è al più $1/(1-\alpha) = 1/(1-i/m)$, ossia $m/(m-i)$.

Mediando su tutte le n chiavi presenti nella tavola otteniamo:

$$\frac{1}{n}\sum_{i=0}^{n-1}\frac{m}{m-i} = \frac{m}{n}\sum_{i=0}^{n-1}\frac{1}{m-i} = \frac{1}{\alpha}\sum_{k=m-n+1}^{m}\frac{1}{k}$$

Possiamo maggiorare la sommatoria con un integrale ottenendo

$$\frac{1}{\alpha}\sum_{k=m-n+1}^{m}\frac{1}{k} \leq \frac{1}{\alpha}\int_{m-n}^{m}\frac{1}{x}dx$$

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

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

Ecco una tavola dei valori di **$1/\alpha \ln [1/(1-\alpha)]$**

α	$1/\alpha \ln [1/(1-\alpha)]$
0.3	**1.19**
0.5	**1.39**
0.7	**1.72**
0.9	**2.56**
0.95	**3.15**
0.99	**4.65**