

Tabelle e indirizzamento diretto

Strutture usate principalmente per le ricerche

Rappresentiamo

UNIVERSO

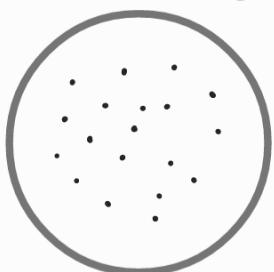


TAVOLA A
INDIRIZZAMENTO
APERTO

$\forall i \in U$

i contiene l'informazione sulle sue locazione.

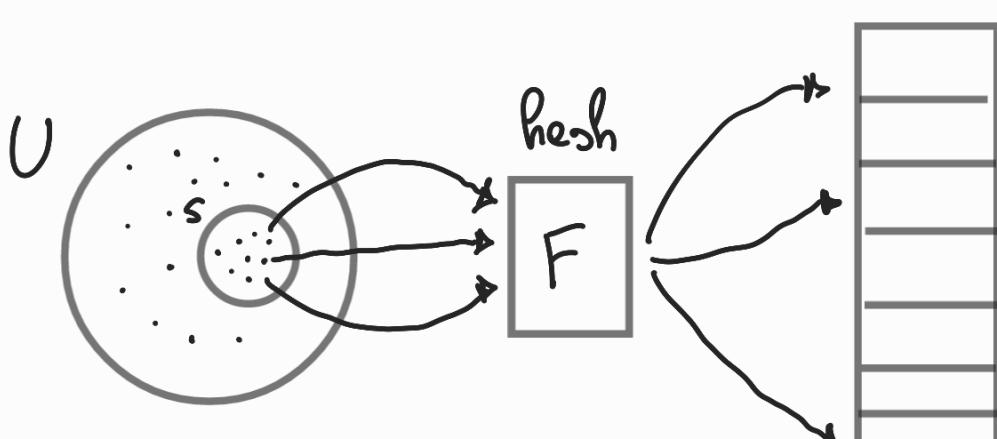
Queste soluzioni ha le stesse problematiche di
verificabilità come il counting-sort

TABELLE HASH

Cerca di comportarsi in maniera simile a quelle
a indirizzamento aperto

$$|T| = m \quad \text{prendete}$$

$$|U| > m$$



Ogni i -esimo elemento pone delle funzioni di hash; le funzioni hash ritornano le posizioni nel range e per un determinato valore K ritornano le stesse posizioni.

Questo causa il problema delle **collisioni**.

Il problema viene risolto in due modi:

- concatenazione
- indirizzamento aperto

CONCATENAZIONE

Il problema si risolve aggiungendo una lista a ogni posizione delle tabelle.

RISULTATI:

CASO PEGGIORI: avremo liste

$$O(\alpha)$$

CASO MEDIO: avremo $O(\alpha)$

$$\alpha = \frac{m}{m} \rightarrow \text{elementi inseriti}$$

α è il fattore di carico

Se $\alpha > 1$ sicuramente avremo delle collisioni.

OPERAZIONI HASH TABLE

INSERT (T, K)

$T[h(K)] \leftarrow K$

CANC(T, K)

T[h(K)]. delete(K)

SEARCH(T, K)

RETURN T[h(K)]. search

FUNZIONI HASH

Si basano sulle ipotesi di hashing uniforme semplice:

$$\forall K \in U \rightarrow \Pr\{h(K) = i\} = \frac{1}{m} \quad 0 \leq i \leq m$$

In breve indice che ogni locazione deve avere le stesse probabilità di essere scelta.

METODO DELLA DIVISIONE

$$h: U \rightarrow \{0, 1, 2, \dots, m-1\}$$

$$h(K) = \lfloor K \text{ mod } m \rfloor$$

In questo metodo è importante usare valori di m diversi di potenze del 2.

Questo dovuto al fatto che renderebbe le funzioni hash dipendente da 2^p bit meno significativi. Così da evitare noi vogliamo che sia dipendente da tutti i bit.

METODO DELLA MOLTIPLICAZIONE

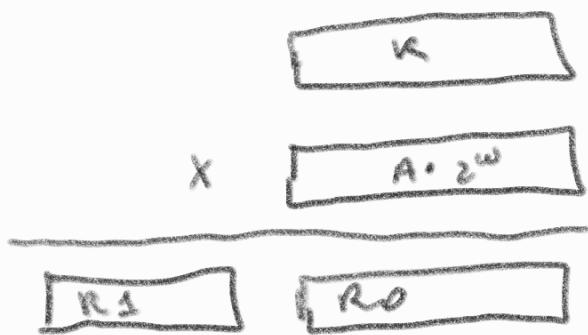
$$h : U \rightarrow \{0, 1, 2, \dots, m-1\}$$

prendiamo un valore costante A tale che
 $0 \leq A < 1$ valore compreso tra 0 ed 1

$$h(k) = \lfloor m(\underbrace{k \cdot A \bmod 1}) \rfloor$$

uso il floor perché ottengo
un numero decimale compreso tra 0 ed $m-1$

In questo caso conviene prendere m come potente del 2 perché



un p bit che servono per
l'hash

Con quest'approccio utile
l'utilizzo delle operazioni
bitwise

INDIRIZZAMENTO APERTO

È una soluzione usata in caso
di memoria M inserimenti

in breve

$$\alpha \leq 1$$

E' usata quindi per scegliere le operazioni di ricerca

Non avremo più una funzione hash che ci torna un valore univocale, ma una permutazione delle m posizioni

QUINDI AVREMO:

$h(k, i)$ numero tentativo di inserimento

$$h: T: \{0, 1, 2, \dots, m-1\} \rightarrow P\{0, 1, \dots, m-1\}$$

AVENDO $m!$ sequenze di scusizioni

USIAMO: IPOTESI DI HASHING UNIFORME

Ogni chiave può essere associata a ogni blocco di sequenze

È ogni sequenza di inserimento
con le stesse probabilità

ATTENZIONE!!

USA UNO UNA SEQUENZA DI SCANSIONE
DOBBIAMO ETICHETTARE LE CELLE
RIMOSSE CON LA CANCELLAZIONE.

OPERAZIONI

INSERT (T, K)

$i \leftarrow 0$

WHILE ($i < m$ AND $T[h(K, i)] \neq \text{vuota}$)

$i++$

IF ($i < m$) THEN

$T[h(K, i)] \leftarrow K$

SEARCH (T, K)

$i \leftarrow 0$

WHILE ($i < m$ AND $T[h(K, i)] \neq \text{vuoto}$)

IF ($T[h(k, i)] == k$)

RETURN TRUE

$i \leftarrow i + 1$

RETURN FALSE

Si dimostre che se $\alpha = \frac{m}{m}$

Tempo medio = $O\left(\frac{1}{1-\alpha}\right)$

CON SUCCESSO:

$O\left(\frac{1}{\alpha} \ln\left(\frac{1}{1-\alpha}\right)\right)$

LA DIFFICOLTA' STA NEL CREARE
UNA BUONA FUNZIONE HASH

ABBIAMO 3 METODI

- SCANSIONE LINEARE

$h': T \rightarrow \{0, 1, 2, \dots, m-1\}$

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

Es: se il primo risultato è l
il secondo sarà z e così
via

HA SOLO m PERMUTAZIONI E SOFFRE
DEL PROBLEMA DI AGGLOMERAZIONE
PRIMARIA

- SCANSIONE QUADRATICA

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

USA m permutazioni

- DOPPIO HASHING

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

USA m^2 permutazioni