Analisi dell'Hashing a catena aperta

Data una tabella di hashing T di grandezza N e con *n* elementi memorizzati, si definisce il fattore di carico λ per la tabella T come:

$$\lambda = \frac{n}{N}$$

Il fattore di carico rappresenta quindi il numero medio o atteso di elementi memorizzati in una catena, ammesso che la gestione delle collisioni sia gestita tramite strutture dati aggiuntive, cioè da sequenze o catene di elementi.

Il peggior comportamento che si può verificare in una struttura di questo tipo è pessimo: infatti tutte le chiavi potrebbero essere mappate nello stesso elemento della tabella T, creando così una catena aperta di lunghezza *n*. La complessità temporale degli algoritmi di gestione del dizionario diventa perciò $\Theta(n)$.
Il caso medio che si verifica in una tabella di hashing dipende da quanto uniformemente vengono distribuite le chiavi all'interno della tabella T. Si assumerà d'ora in avanti che la distribuzione degli elementi all'interno della tabella T avvenga nel modo più uniforme possibile. Il che significa che la funzione di hash mapperà una qualsiasi chiave dell'insieme di partenza in un valore intero compreso fra 0 e N-1 con eguale probabilità.

Teorema.
In una tabella di hash in cui le collisioni sono risolte con la tecnica della catena aperta, <u>una ricerca infruttuosa</u> richiede un tempo $\Theta(1+\lambda)$, nella media, assunto che la funzione di hashing utilizzata distribuisca le chiavi uniformemente.

Dimostrazione
Sotto l'assunzione che la funzione h() distribuisca le chiavi uniformemente sugli N elementi disponibili della tabella T, il calcolo del tempo medio impiegato per trovare un elemento di chiave k in T è equivalente al tempo di calcolo della funzione h(k) sommato al tempo necessario per raggiungere la fine della catena all'elemento selezionato. La lunghezza media di ogni catena collegata ad ogni elemento della tabella T è esattamente $\lambda = n/N$. Così, il tempo medio per effettuare una ricerca infruttuosa è $\Theta(1 + \lambda)$.

Teorema.
In una tabella di hash T in cui le collisioni sono risolte con la tecnica delle catene aperte, <u>una ricerca che termina in modo favorevole</u>, richiede un tempo dell'ordine di $\Theta(1 + \lambda)$, nella media e nell'ipotesi di distribuzione uniforme delle chiavi.

Dimostrazione
Il numero medio di elementi ricercati per recuperare un elemento in una tabella di questo tipo è esattamente 1 più il numero di elementi esaminati quando l'elemento cercato è stato introdotto, ammesso che ogni nuovo elemento vada a finire in fondo alla lista o catena aperta.
Quindi, per conoscere il numero medio di elementi visitati, bisogna calcolare la media, dati n elementi presenti nella tabella T, di 1 più la lunghezza attesa della catena aperta in cui

l'ultimo elemento i-esimo è stato introdotto. La lunghezza media di tale catena è: $(i-1)/N$. Così il calcolo della media diventa:

$$\frac{1}{n}\sum_{i=1}^{n}\left(1+\frac{i-1}{N}\right)=1+\frac{1}{nN}\sum_{i=1}^{n}(i-1)=1+\left(\frac{1}{nN}\right)\cdot\left(\frac{(n-1)n}{2}\right)=1+\frac{\lambda}{2}-\frac{1}{2N}$$

Quindi, il tempo totale richiesto per una ricerca effettuata con successo (incluso il tempo per il calcolo della funzione di hash) è:

$$\Theta\left(2+\frac{\lambda}{2}-\frac{1}{2N}\right)=\Theta(1+\lambda)$$

Che significato ha il teorema precedente? Se il numero degli elementi inseriti n nel dizionario realizzato tramite una tabella di hash a catena aperta di dimensione N è dell'ordine di $O(N)$, allora:

$\lambda = n/N = O(N)/N = O(1)$. Quindi una ricerca condotta in tale tabella richiederà un tempo medio pari a $O(1)$.

Funzioni di hashing universali

Se una persona volesse dimostrare maliziosamente che la funzione di hashing h() che avete scelto per la realizzazione di un dizionario con questa tecnica non è buona, gli sarebbe sufficiente scegliere n numeri n_0, n_1, n_2, …,n_{n-1} tali che $h(n_0) = h(n_1) = h(n_2) =\ldots= h(n_{n-1})$. Infatti ogni funzione h(), per il solo fatto che deve comprimere gli elementi di **N**, insieme dei numeri naturali, in un insieme ristretto [0, N-1], avrà il 'difetto' di presentare valori uguali per opportune chiavi.
L'unico metodo per avere una scappatoia a questa situazione è quella di avere, una volta scelto il campione di chiavi da testare, una funzione scelta a caso da un gruppo di 'buone' funzioni di hashing. Questo approccio è chiamato *hashing universale* (*universal hashing*).

Sia H una collezione finita di funzioni di hashing che mappano l'insieme di partenza U (detto anche *Universo* delle chiavi) nell'insieme finito [0,1,..,N-1]. Una tale collezione è detta *universale* se, per ogni coppia di chiavi $x,y \in U$, il numero di funzioni di hash $h(\ ) \in H$ per i quali $h(x) = h(y)$ è precisamente $|H|/N$. In altre parole, con una qualsiasi funzione funzione h( ) di H scelta a caso, la possibilità che si verifichi una collisione fra le chiavi diverse x e y è esattamente $1/N$. Che è esattamente la possibilità di collisione se $h(x)$ e $h(y)$ fossero scelti uniformemente a caso nell'interavllo $[0, N-1]$.

Teorema.
Se h( ) viene scelta a caso da un insieme H di funzioni universali <u>ed è utilizzata per mappare n chiavi</u> in una tabella di dimensione N, dove $n \leq N$, il numero atteso di collisioni riguardanti un particolare chiave x è minore di 1.

Dimostrazione
Per ogni coppia distinta di chiavi y,z, sia C_{yz} una variabile casuale che vale 1 se $h(y) = h(z)$ e 0 altrimenti. Cioe: $C_{yz} = \begin{cases} 1 \ se \ h(y)=h(z) \\ 0 \quad altrimenti \end{cases}$

Dato che ogni coppia di chiavi collide, per definizione, con probabilità $1/N$, si ha:

$$E[C_{yz}] = 1/N$$

Sia ora C_x il numero totale di collisioni riguardante la chiave x in una tabella di hash T di dimensione N contenente n chiavi. Allora la media di tutte le collisioni sulla chiave x è la somma di tutte le medie delle collisioni della chiave x con tutte le altre chiavi diverse da x. Dato che si stanno mappando n chiavi, si ottiene:

$$E[C_x] = \sum_{\substack{y \in T \\ y \neq x}} E[C_{xy}] = \frac{n-1}{N}$$

Ma dato che $n \leq N$, allora $E[C_x] < 1$.

Ma quanto è difficile crearsi una propria collezione universale di funzioni di hashing? Per nulla!
Si immagini di scegliere la dimensione N della tabella di hashing in modo che N sia primo. Si scomponga ora ogni chiave in r+1 bytes (cioè caratteri o sottostringhe binarie di lunghezza fissa) in modo tale che $x = (x_0, x_1, \ldots, x_r)$. La sola richiesta aggiuntiva è che ogni valore corrispondente di ogni byte sia minore di N.
Sia $a = \langle a_0, a_1, a_2, \ldots, a_r \rangle$ una sequenza di $r + 1$ elementi scelti a caso nell'insieme {0, 1, .., N-1}(si tratta di r+1 cifre con N simboli). Si definisce la funzione hash corrispondente $h_a \in H$:

$$h_a(x) = \sum_{i=0}^{r} a_i x_i \bmod m$$

Con questa definizione si avrà: $H = \bigcup_a \{h_a\}$ insieme con N^{r+1} elementi.

Teorema.
La classe H sopra definita è una classe *universale* di funzioni.

Dimostrazione
Si consideri una coppia di chiavi distinte x e y, dove naturalmente $x = (x_0, x_1, \ldots, x_r)$ e $y = (y_0, y_1, \ldots, y_r)$. Si assuma che x_0 sia diverso da y_0, dato che lo stesso discorso può essere fatto per qualsiasi coppia di bytes. Per ogni valore fissato di $a_1, a_2, \ldots, a_r$, esiste esattamente un solo valore di a_0 tale da soddisfare l'equazione h(x) = h(y). Tale valore a_0 è soluzione di:

$$a_0(x_0 - y_0) = -\sum_{i=1}^{N} a_i(x_i - y_i) \pmod N$$

Dato che N è primo, la quantità positiva x_0-y_0 ha una inversa moltiplicativa modulo N, e quindi esiste una sola soluzione per a_0 modulo N. Perciò, ogni coppia di chiavi x e y collide per esattamente N^r valori di a, dato che essi collidono una sola volta per ogni possibile valore di $\langle a_1, a_2, a_3, \ldots, a_r \rangle$ (cioè per ogni unico valore di a_0). Dato che ci sono N^{r+1} possibili valori per la sequenza a, le chiavi x e y collidono con probabilità esattamente $N^r/N^{r+1} = 1/N$. Quindi H è *universale*.

Analisi dell'hashing ad indirizzamento aperto

L'analisi della tecnica di indirizzamento aperto è espressa in termini del fattore di carico λ. Se n elementi sono memorizzati nella tabella T di hash di dimensione N, il numero medio di collisioni per ogni casella della tabella è n/N. Naturalmente, con l'indirizzamento aperto, esiste al più un elemento per ogni cella del vettore. Così è chiaro che deve essere $n \leq N$, cioè il numero degli

elementi memorizzati nella tabella deve essere minore od uguale alla dimensione della tabella stessa.
Si assume che le funzioni di hash utilizzate per effettuare le analisi siano tutte tali da distribuire uniformemente i valori sull'intervallo [0, N-1]. In uno schema ideale, la sequenza di tentativi <h(k,0), h(k,1), h(k,2), …, h(k,N-1)> per una determinata chiave k è ugualmente probabile ad ogni permutazione in <0, 1, 2, …, N-1>. Che vuol dire che ogni sequenza di tentativi sulla tabella T è ugualmente probabile. Naturalmente, per ogni chiave k fornita, esiste un'unica sequenza di tentativi ottenuta utilizzando la stessa funzione di hashing h().

Teorema.
Data una tabella di hash T organizzata ad indirizzamento aperto, con fattore di carico $\lambda = n/N < 1$, il numero atteso di tentativi in <u>una ricerca infruttuosa</u> di un elemento è al più $1/(1-\lambda)$, assumendo di avere una funzione h() di mappatura uniforme delle chiavi.

Dimostrazione
In un infruttuoso tentativo di ricerca di un elemento, ogni sondaggio tranne l'ultimo accedono a celle piene della tabella T che però non contengono la chiave cercata k. L'ultimo sondaggio accede ad una cella vuota.
Si definisce ora la seguente funzione di probabilità legata ad un certo evento:

p_i = Probabilità {esattamente i sondaggi accedono a celle occupate di T}

dove i vale per 1, 2, .., N tentativi. Per cui tale probabilità è nulla per i = N+1, …. Dato che si possono trovare al più N celle occupate. Così il numero atteso di sondaggi è uno (la cella vuota finale) più la media (aspettazione) del numero di celle piene incontrate:

$$1 + \sum_{i=0}^{\infty} i\, p_i$$

Per valutare l'espressione precedente bisogna definire il seguente evento:

q_i = Probabilità {al più i sondaggi accedono a celle occupate di T}

Per i = 0, 1, 2, …. Si può utilizzare la seguente identità:

$$\sum_{i=0}^{\infty} i\, p_i = \sum_{i=1}^{\infty} q_i$$

Qual è il valore di q_i per i ≥1? La probabilità che il primo sondaggio acceda ad una cella occupata è n/N, cioè:

$$q_1 = \frac{n}{N}$$

Con una funzione di hashing uniforme, un secondo tentativo, se necessario, è ad una delle N-1 celle non ancora visitate, n-1 delle quali sono occupate. Un secondo tentativo viene naturalmente fatto solamente nel caso che durante il primo tentativo si sia incontrata una cella occupata. Quindi:

$$q_2 = \left(\frac{n}{N}\right)\left(\frac{n-1}{N-1}\right)$$

In generale, l' *i*-esimo tentativo viene fatto solamente se gli *i*-1 tentativi precedenti hanno incontrato tutte celle occupate e la cella *i*-esima da sondare è fra *N*-*i*+1 celle rimaste, *n*-*i*+1 delle quali sono già occupate. Così:

$$q_i = \left(\frac{n}{N}\right)\left(\frac{n-1}{N-1}\right)\cdots\left(\frac{n-i+1}{N-i+1}\right)$$

Notando che $\frac{n-j}{N-j} \le \frac{n}{N}$ se $n \le N$ e $j \ge 0$, si può scrivere:

$$q_i = \left(\frac{n}{N}\right)\left(\frac{n-1}{N-1}\right)\cdots\left(\frac{n-i+1}{N-i+1}\right) \le \left(\frac{n}{N}\right)^i = \lambda^i$$

Dopo n tentativi, tutte le n celle occupate sono state visitate e non lo saranno più per i tentativi successivi, così che si può stabilire che $q_i = 0$ per $i > n$.
Ora si è in grado di valutare la sommatoria:

$$1 + \sum_{i=0}^{\infty} i\, p_i = 1 + \sum_{i=1}^{\infty} q_i$$

$$1 + \sum_{i=0}^{\infty} i\, p_i = 1 + \sum_{i=1}^{\infty} q_i = 1 + \left\{1 \cdot \left(\frac{n}{N}\right) + 2 \cdot \left(\frac{n}{N}\right)\left(\frac{n-1}{N-1}\right) + \ldots + i \cdot \left(\frac{n}{N}\right)\left(\frac{n-1}{N-1}\right)\cdots\left(\frac{n-i+1}{N-i+1}\right) + \ldots\right\}$$

$$\le 1 + \left\{1 \cdot \left(\frac{n}{N}\right) + 2 \cdot \left(\frac{n}{N}\right)^2 + \ldots + i \cdot \left(\frac{n}{N}\right)^i + \ldots.\right\} = 1 + \lambda + \lambda^2 + \ldots + \lambda^i + \ldots. = \frac{1}{1-\lambda}$$

Sommatoria che ha una facile interpretazione: un sondaggio è sempre fatto, un secondo sondaggio invece è necessario con probabilità λ, un terzo sondaggio è fatto con probabilità λ^2, e così via.

Se λ è una costante, il teorema precedente afferma che una ricerca infruttuosa viene fatta in un tempo O(1). Per esempio, se una tabella di hash T è mezza piena, la media del numero di tentativi in una ricerca infruttuosa è 1/(1-0.5) = 2. Se la tabella è piena al 90% (λ = 0.9) allora il numero medio di tentativi salirà a 1/(1-0.9) = 10.

Il teorema precedente fornisce uno strumento per valutare le prestazioni della procedura di inserimento in una tabella di hash.

Corollario.
L'inserimento di un elemento in una tabella di hash T, organizzata ad indirizzamento aperto con un fattore di carico λ = n/N, richiede al più 1/(1-λ) tentativi nel caso medio. Assumendo una funzione di hash uniforme.

Dimostrazione
Un elemento è inserito nella tabella solamente nel caso ci sia ancora dello spazio libero, e quindi solo se $\lambda < 1$. L'inserimento di un elemento richiede una ricerca infruttuosa seguita dalla sistemazione del nuovo elemento nella prima cella libera della tabella T. Così il numero atteso di sondaggi è 1/(1-λ).

Calcolare il numero medio di sondaggi per una ricerca che termina favorevolmente è alquanto più difficile.

Teorema.
Data una tabella di hashing T organizzata ad indirizzamento aperto, con fattore di carico $\lambda < 1$, il numero atteso di sondaggi durante <u>una ricerca di un elemento che termini favorevolmente</u> è al più

$$\frac{1}{\lambda}\log\frac{1}{1+\lambda}+\frac{1}{\lambda}$$

Assumendo sempre che la funzione di hashing h() distribuisca uniformemente le chiavi nell'intervallo [0, N-1] e che la probabilità che venga cercata una qualsisi chiave k sia la stessa per tutte le chiavi.

Dimostrazione
Una ricerca per una certa chiave *k* segue la stessa sequenza di tentativi seguita durante le fasi di inserimento della chiave stessa. Per il corollario precedente, se *k* era la (*i*+1)-esima chiave inserita nella tabella T, allora il numero medio di sondaggi fatti per cercare la chiave *k* prima dell'inserimento è al più $\frac{1}{1-i/N}=\frac{N}{N-i}$. Facendo la media su tutte le *n* chiavi inserite nella tabella T si ottiene il numero medio dei sondaggi in una tabella T durante le fasi di ricerca di una chiave che termini in modo favorevole. Quindi:

$$\frac{1}{n}\sum_{i=0}^{n=1}\frac{N}{N-1}=\frac{N}{n}\sum_{i=0}^{n-1}\frac{1}{N-i}=\frac{1}{\lambda}\cdot\left(H_N-H_{N-n}\right)$$

Dove $H_i=1+\frac{1}{2}+\frac{1}{3}+...+\frac{1}{i}=\sum_{j=1}^{i}\frac{1}{j}$ è detta *funzione armonica*. Utilizzando la relazione $i\le H_i\le\log i+1$ si ottiene:

$\frac{1}{\lambda}\cdot\left(H_N-H_{N-n}\right)\le\frac{1}{\lambda}\cdot\left(\log N+1-\log(N-n)\right)=\frac{1}{\lambda}\log\frac{N}{N-n}+\frac{1}{\lambda}$ come limite superiore sulla media dei tentativi durante una ricerca terminata con esito favorevole.

Se la tabella T, per esempio è piena a metà (λ=1/2), allora il numero atteso di sondaggi è minore di 3.387. Se la tabella invece è piena al 90% (λ=0.9), allora il numero atteso di sondaggi diventa minore di 3.67.