Massimo Benerecetti

Analisi dell'Hashing

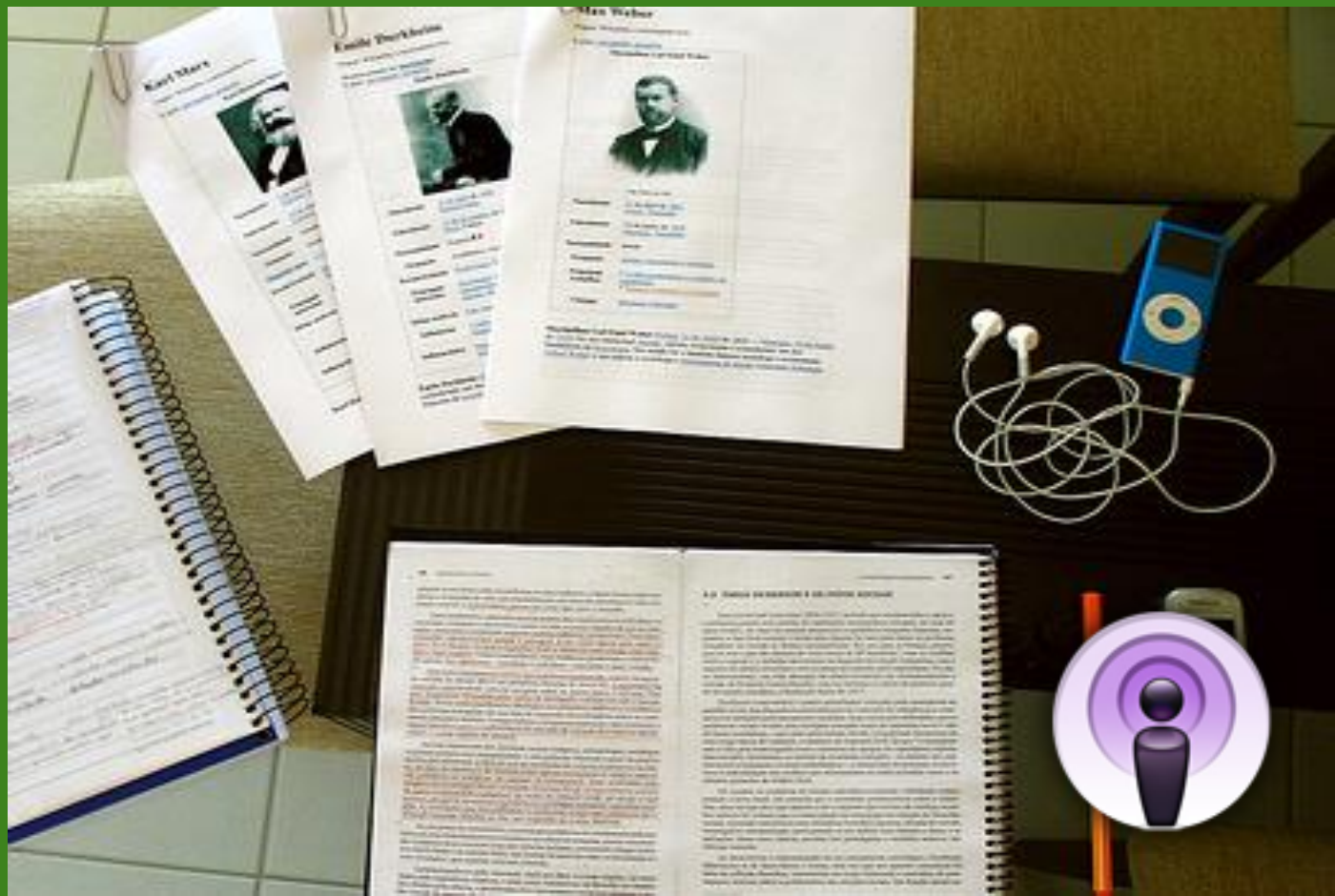
Lezione n.#
Parole chiave:
Inserire testo

Corso di Laurea:
Informatica

Insegnamento:
Algoritmi e Strutture Dati

Email Docente:
bene@na.infn.it

A.A. 2009-2010



- Data una funzione hash fissata è sempre possibile individuare un insieme di **n** chiavi che determinano il caso peggiore, per cui l'inserimento di una chiave impiegherebbe tempo **$\Theta(n)$**.
- Un modo per ovviare a questo problema è quello di ***scegliere casualmente***, ad ogni esecuzione dell'algoritmo, la funzione di hash da impiegare.
- È una forma di ***randomizzazione*** che rende indipendente la *funzione hash* dalle chiavi che si dovranno elaborare.
- In tal modo, ogni sequenza di chiavi in input ha la stessa probabilità di determinare il caso peggiore.
- Sia **U** l'universo delle chiavi. Una ***famiglia*** **H** di ***funzioni hash*** da **U** all'insieme **{0,1,…,n-1}** si dice ***universale***, se il numero di funzioni **$h \in H$** tali per cui **$h(k)=h(l)$** (con $k \neq l \in$ **U**) è *al massimo* **$|H|/n$**.

- Scegliamo un qualche numero primo **p** in modo che ogni possibile chiave sia minore di **p**.
- Siano **{0,1,…,n-1}** i possibili valori di hashing delle chiavi.
- Una ***famiglia*** di ***funzioni hash*** si dice ***universale*** se, per ogni coppia di chiavi $0 \le i,j < p$,

$$\Pr[h(i)=h(j)] < 1/n$$

- Data la dimensione **n** della *tabella hash*, selezioniamo ***in modo casuale*** due numeri interi, $1 \le a < p$ e $0 \le b < p$, e definiamo la seguente ***funzione hash***

$$h_{a,b}(k) = (ak+b \bmod p) \bmod n$$

Teorema: La famiglia di funzioni

$$H = \{h_{a,b}(k) \mid 1 \le a < p \text{ e } 0 \le b < p\}$$

è ***universale***.

Teorema: La famiglia di funzioni

$$H = \{(ak+b \bmod p) \bmod n \mid 1 \le a < p \text{ e } 0 \le b < p\}$$

è ***universale***.

Dim. Consideriamo due chiavi distinte $k \neq l$. Presa una qualsiasi funzione $h \in H$ siano

$$r = ak+b \bmod p$$

$$s = al+b \bmod p$$

È facile osservare che $r \neq s$. Infatti, avremo che

$$r - s = a(k-l) \bmod p$$

Poiché sia **a** che **(k-l)** sono non nulli, anche il loro prodotto lo sarà.

Essendo **p** un numero primo, MCD(**a**,**p**) = 1 e MCD(**k-l**,**p**) = 1. Quindi anche MCD(**a(k-l)** ,**p**) = 1. Il che garantisce che **a(k-l)** è non nullo modulo **p**. Quindi $r \neq s$.

Teorema: La famiglia di funzioni

$$H = \{(ak+b \bmod p) \bmod n \mid 1 \le a < p \text{ e } 0 \le b < p\}$$

è ***universale***.

Dim.(continua) Quindi $r \neq s$. Questo significa che non si verificano collisioni a livello **(mod p)**.

Poiché vi sono **p-1** valori possibili per **a** e **p** valori possibili per **b**, esistono **p(p-1)** differenti coppie **a**,**b** e quindi **p(p-1)** differenti funzioni in **H**.

Analogamente, vi sono **p(p-1)** differenti coppie **r**,**s**. Quindi c'è una ***corrispondenza biunivoca*** tra l'insieme delle possibili coppie **a**,**b** e l'insieme delle possibili le coppie **r**,**s**.

Se la coppia a,b è scelta *uniformemente a caso*, allora tutte le coppie r,s distinte hanno la stessa probabilità di essere le immagini di due date chiavi k,l.

Teorema: La famiglia di funzioni

$$H = \{(ak+b \bmod p) \bmod n \mid 1 \le a < p \text{ e } 0 \le b < p\}$$

è ***universale***.

Dim. Se la coppia **a**,**b** è scelta uniformemente a caso, allora tutte le coppie **r**,**s** distinte hanno la stessa probabilità di essere le immagini di due date chiavi **k**,**l**.

Quindi, la probabilità che due chiavi distinte **k**,**l** collidano, data una scelta casuale di **h** da **H**, dipende solo dalla probabilità che due valori distinti **r**,**s** siano mappati, modulo **n**, nello stesso valore, cioè che valga

$$r = s \bmod n.$$

Chiaramente, per ogni **r** fissato, il numero di valori di **s** diversi da **r** che collidono **mod n** con **r** è al più

$$\lceil p/n \rceil - 1$$

Esistono, infatti, al più $\lceil p/n \rceil$ intervalli di **n** interi (contigui) tra 0 e **p** (compreso quello che contiene **r**) e c'è un solo intero, per ciascuno di questi intervalli, che è uguale a **r mod n**.

Teorema: La famiglia di funzioni

$$H = \{(ak+b \bmod p) \bmod n \mid 1 \le a < p \text{ e } 0 \le b < p\}$$

è universale.

Dim. Chiaramente, per ogni **r** fissato, il numero di valori di **s** diversi da **r** che collidono **mod n** è al più

$$\lceil p/n \rceil - 1 \le ((p+(n-1))/n) - 1$$

$$= (p-1)/n$$

La probabilità di collisione **mod n** tra **r** e **s** è quindi il rapporto tra il numero di **s** che collidono con **r** e il numero totale degli **s**, cioè

$$\Pr[s = r \bmod n] \le ((p-1)/n)/(p-1) = 1/n$$

Da questo possiamo concludere che la probabilità di collisione tra chiavi $k \neq l$ rispetto alla funzione casualmente scelta $h_{a,b}$ è

$$\Pr[h_{a,b}(k) = h_{a,b}(l)] \le 1/n$$

È a questo punto facile dimostrare che data una collezione universale di funzioni di hashing, come quella appena definita, il tempo di esecuzione di un'operazione di ricerca (inserimento e cancellazione) è *mediamente buono*.

Teorema: Sia **h** una funzione di hash scelta casualmente da una collezione universale **H** e **n** la dimensione della tabella hash in cui sono state inserite **m** chiavi. Si supponga di usare il concatenamento per la gestione delle collisioni. Allora:

- se **k** non è nella tavola, la lunghezza media della sua lista di collisione è al massimo α = **m/n**;
- se **k** è presente nella tavola, la lunghezza media della sua lista di collisione è al massimo $1+\alpha$ = 1+ **m/n**.

Dim.

Introduciamo le variabili casuali indicatrici $X_{k,l} = I\{h(k)=h(l)\}$.

Dalla dimostrazione del teorema precedente, sappiamo che $Pr[h(k)=h(l)] \leq 1/n$.

Da cui possiamo dedurre che il valore atteso

$$E[X_{k,l}] = Pr[h(k)=h(l)] \leq 1/n$$

Sia ora Y_k la variabile casuale che corrisponde al *numero di chiavi* diverse da **k** presenti nella tabella **T** che sono *associate alla stessa cella di* **k**.

Quindi $Y_k = \sum_{l \in T, l \neq k} X_{k,l}$

La quantità desiderata è allora il valore atteso di Y_k.

$$E[Y_k] = E\left[\sum_{l \in T, l \neq k} X_{k,l}\right] = \sum_{l \in T, l \neq k} E[X_{k,l}] = \sum_{l \in T, l \neq k} 1/n$$

Dim. $\mathbf{E}[Y_k] = \mathbf{E}\left[\sum_{l \in T, l \neq k} X_{k,l}\right] = \sum_{l \in T, l \neq k} \mathbf{E}[X_{k,l}] \leq \sum_{l \in T, l \neq k} 1/n$

A questo punto consideriamo i due casi.

1. Se $k \notin T$: la *lista di collisione* non conterrà **k** e avrà proprio lunghezza pari Y_k. Poiché il numero di elementi diversi da **k** in tabella è pari al numero di elementi inseriti **m**, otteniamo

$$\mathbf{E}[Y_k] \leq \sum_{l \in T, l \neq k} 1/n = \sum_{1 \leq i \leq m} 1/n$$

$$\leq m/n = \alpha$$

2. Se $k \in T$: la chiave **k** compare nella cella $T[h(k)]$ e il valore di Y_k non include **k**. Quindi la *lista di collisione* di **k** avrà lunghezza Y_k+1. Poiché il numero di elementi diversi da **k** in tabella è pari al numero di elementi inseriti **m** meno 1 (per **k**), otteniamo

$$\mathbf{E}[Y_k+1] \leq 1+\sum_{l \in T, l \neq k} 1/n = 1+\sum_{1 \leq i \leq m-1} 1/n$$

$$\leq 1+(m-1)/n = 1+\alpha-1/n < 1+\alpha$$

La randomizzazione garantisce che, nel caso dell'indirizzamento chiuso, tutte le operazioni di inserimento, ricerca e cancellazione possano essere effettuate con un ***buon*** tempo medio di esecuzione.

Corollario. L'***hashing universale*** unito alla gestione delle collisioni per concatenazione in una tabella di dimensione **n** garantisce un tempo medio $\Theta(\mathbf{m})$ per eseguire **m** operazioni di inserimento, cancellazione e ricerca che contenga $\boldsymbol{O}(\mathbf{n})$ inserimenti.

Dim. Poiché gli inserimenti sono di ordine $\boldsymbol{O}(\mathbf{n})$, il numero di chiavi presenti in **T** è $\boldsymbol{O}(\mathbf{n})$. Da cui otteniamo che il tasso di carico α è

$$\alpha = \boldsymbol{O}(\mathbf{n})/\mathbf{n} = \boldsymbol{O}(\mathbf{1})$$

Per il teorema precedente, le ricerche, e quindi inserimenti e cancellazioni, richiedono tempo medio $\boldsymbol{O}(\alpha) = \boldsymbol{O}(\mathbf{1})$. Quindi le **m** operazioni richiedono complessivamente tempo medio $\boldsymbol{O}(\mathbf{m})$.

Poiché ogni operazione richiede tempo $\Omega(\mathbf{1})$, segue che il tempo medio complessivo necessario è $\Theta(\mathbf{m})$.

Analisi dell'indirizzamento aperto

Nell'indirizzamento aperto al massimo ogni cella potrà contenere una chiave. Quindi $\mathbf{m} \leq \mathbf{n}$. Il fattore di carico α varrà quindi $\alpha \leq 1$.

Sotto l'ipotesi di ***hashing uniforme***, ogni sequenza di sondaggi $\langle \mathbf{h}(\mathbf{k},0), h(\mathbf{k},1),\ldots, \mathbf{h}(\mathbf{k},\mathbf{m}-1)\rangle$ ha la stessa probabilità di essere una qualsiasi permutazione della sequenza $\langle 0, 1,\ldots, \mathbf{m}-1\rangle$.

Teorema: Nell'ipotesi di hashing uniforme e indirizzamento aperto con fattore di carico $\alpha = \mathbf{m}/\mathbf{n} < 1$, il ***numero medio di sondaggi*** per una ricerca senza successo è al più $1/(1-\alpha)$.

Dim. Indichiamo con la variabile casuale **X** il numero di sondaggi eseguiti nella ricerca.

Sia $\mathbf{A_i}$ (con i=1,2,…,) l'evento "*la i-esima ispezione trova la cella occupata*".

L'evento $\{\mathbf{X} \geq i\}$ è quindi l'intersezione degli eventi $\mathbf{A_j}$ con j=1,2,…,i-1. Cioè, $\{\mathbf{X} \geq i\} = \mathbf{A_1} \cap \mathbf{A_2} \cap \ldots \cap \mathbf{A_{i-1}}$.

Analisi dell'indirizzamento aperto

L'evento $\{\mathbf{X} \geq i\}$ è, quindi, l'intersezione degli eventi $\mathbf{A_j}$ con $j=1,2,\ldots,i-1$. Cioè, $\{\mathbf{X} \geq i\} = \mathbf{A_1} \cap \mathbf{A_2} \cap \ldots \cap \mathbf{A_{i-1}}$.

Lo scopo è quindi quello di calcolare $\Pr[\mathbf{X} \geq i]$, lo faremo calcolando $\Pr[\mathbf{A_1} \cap \mathbf{A_2} \cap \ldots \cap \mathbf{A_{i-1}}]$.

Si può facilmente dimostrare che

$$\Pr[\mathbf{A_1} \cap \mathbf{A_2} \cap \ldots \cap \mathbf{A_{i-1}}] = \Pr[\mathbf{A_1}] \cdot \prod_{2 \leq j \leq i-1} \Pr[\mathbf{A_j} \mid \mathbf{A_1} \cap \ldots \cap \mathbf{A_{j-1}}]$$

$\Pr[\mathbf{A_1}] = \mathbf{m/n}$, in quanto ci sono $\mathbf{m}$ elementi ed $\mathbf{n}$ celle.

Per $j \geq 2$, la probabilità di trovare la cella occupata al j-esimo sondaggio avendo trovato le celle occupate fino al $(j-1)$-esimo sondaggio è

$$\Pr[\mathbf{A_j} \mid \mathbf{A_1} \cap \ldots \cap \mathbf{A_{j-1}}] = (m-j+1)/(n-j+1)$$

Infatti, restano $(n-j+1)$ celle possibili da esaminare e $(m-j+1)$ elementi presenti ma non ancora incontrati durante i sondaggi.

La probabilità di una di queste celle sia occupata da una di quelle chiavi, sotto hashing uniforme, è proprio il rapporto tra le due quantità.

$$\Pr[\mathbf{A_1} \cap \mathbf{A_2} \cap \ldots \cap \mathbf{A_{i-1}}] = \Pr[\mathbf{A_1}] \cdot \prod_{2 \le j \le i-1} \Pr[\mathbf{A_j} \mid \mathbf{A_1} \cap \ldots \cap \mathbf{A_{j-1}}]$$

$$\Pr[\mathbf{A_1}] = \mathbf{m/n} \text{ e } \Pr[\mathbf{A_j} \mid \mathbf{A_1} \cap \ldots \cap \mathbf{A_{j-1}}] = (m-j+1)/(n-j+1)$$

Inoltre, poiché $\mathbf{m} < \mathbf{n}$, $\mathbf{(m-j)/(n-j)} \le \mathbf{m/n}$, per ogni $0 \le j < \mathbf{n}$.

Avremo, quindi, per ogni $0 \le i < \mathbf{n}$ la seguente:

$$\Pr[\mathbf{X} \ge i] = \prod_{1 \le j \le i-1} (m-j+1)/(n-j+1) \le \prod_{1 \le j \le i-1} \mathbf{m/n} = \mathbf{(m/n)^{i-1}} = \alpha^{i-1}$$

A questo punto, possiamo calcolare il valore medio di **X**.

$$E[\mathbf{X}] = \sum_{i \ge 0} i \cdot \Pr[\mathbf{X} = i]$$

$$= \sum_{i \ge 0} i \cdot (\Pr[\mathbf{X} \ge i] - \Pr[\mathbf{X} \ge i+1])$$

$$= \sum_{i \ge 1} \Pr[\mathbf{X} \ge i]$$

Dalla precedente otteniamo quindi

$$E[\mathbf{X}] = \sum_{i \ge 1} \Pr[\mathbf{X} \ge i] = \sum_{i \ge 1} \alpha^{i-1} = \sum_{i \ge 0} \alpha^{i} = \mathbf{1/(1-\alpha)}$$

Corollario: L'inserimento di una chiave in una tabella a hash indirizzamento aperto con fattore di carico α, richiede in media non più di $1/(1-\alpha)$ sondaggi, nell'ipotesi di hashing uniforme.

Dim.

Un inserimento può essere effettuato solo se la tabella non è piena, quindi se $\alpha < 1$.

Inoltre, un inserimento richiede che prima sia stata effettuata una ricerca *senza successo* seguita dal vero e proprio inserimento.

Il teorema precedente garantisce, quindi, che il numero medio di sondaggi necessari per l'inserimento sia non superiore a $1/(1-\alpha)$.

Teorema: Nell'ipotesi di hashing uniforme e indirizzamento aperto con fattore di carico $\alpha = \frac{m}{n} < 1$, il ***numero medio di sondaggi*** per una ricerca con successo è al più $\frac{1}{\alpha} \cdot \ln\left(\frac{1}{1-\alpha}\right)$.

Dim.

La ricerca di una chiave **k** presente in tabella segue gli stessi sondaggi fatti per il suo inserimento.

Per il corollario precedente, se **k** è stata la (i+1)-esima chiave inserita (con i≥0), il numero medio di sondaggi è stato 1/(1-(i/n)).

Infatti, al momento dell' inserimento della (i+1)-esima chiave il fattore di carico era i/n.

Poiché 1/(1-(i/n)) = n/(n-i)

dalle considerazioni precedenti, possiamo concludere che una ricerca con successo di quella chiave **k** richieda, in media, n/(n-i) sondaggi.

Teorema: Nell'ipotesi di hashing uniforme e indirizzamento aperto con fattore di carico $\alpha = \frac{m}{n} < 1$, il ***numero medio di sondaggi*** per una ricerca con successo è al più $\frac{1}{\alpha} \cdot \ln\left(\frac{1}{1-\alpha}\right)$.

Dim.

Possiamo, quindi, concludere che una ricerca con successo di quella chiave **k** richieda, in media, **n/(n-i)** sondaggi.

Per ottenere la quantità desiderata, è ora sufficiente calcolare la media su tutte le **m** chiavi inserite in tabella.

Avremo quindi che il numero medio di sondaggi richiesti è pari a

$$\frac{1}{m} \cdot \sum_{0 \le i \le m-1} \frac{n}{n-i} = \frac{n}{m} \cdot \sum_{0 \le i \le m-1} \frac{1}{n-i}$$

$$= \frac{1}{\alpha} \cdot (H_n - H_{n-m})$$

dove $H_i = \sum_{1 \le j \le i-1} \frac{1}{j}$.

Teorema: Nell'ipotesi di hashing uniforme e indirizzamento aperto con fattore di carico $\alpha = \frac{m}{n} < 1$, il ***numero medio di sondaggi*** per una ricerca con successo è al più più $\frac{1}{\alpha} \cdot \ln\left(\frac{1}{1-\alpha}\right)$.

Dim. Avremo quindi che il numero medio di sondaggi richiesti è:

$$(1/m)\cdot\Sigma_{0\le i\le m-1}\ n/(n-i) = (1/\alpha)\cdot(H_n - H_{n-m})$$

dove $H_i = \Sigma_{1\le j\le i-1}\ 1/j$.

$$\begin{aligned}(1/\alpha)\cdot(H_n - H_{n-m}) &= (1/\alpha)\cdot\left(\Sigma_{1\le i\le n}\ 1/i - \Sigma_{1\le i\le n-m}\ 1/i\right)\\ &= (1/\alpha)\cdot(\ln n - \ln(n-m))\\ &= (1/\alpha)\cdot \ln n/(n-m)\\ &= (1/\alpha)\cdot \ln 1/((n-m)/n)\\ &= \frac{1}{\alpha}\cdot\ln\left(\frac{1}{1-\alpha}\right)\end{aligned}$$

Hashing Perfetto

- Si supponga che l'insieme delle chiavi da memorizzare sia fissato a priori (cioè non cambia nel tempo).
- Esempi: le parole chiave di un linguaggio di programmazione, i nomi dei file in un CDROM, etc.
- Chiameremo ***Hashing Perfetto*** un meccanismo di hashing che garantisca tempo di accesso alle chiavi ***O(1)*** nel ***caso peggiore***.
- L'idea è quella di utilizzare uno ***schema di hashing a due livelli***. Ognuno dei quali utilizza la tecnica dell'***Hashing Universale***.
- Il ***primo livello*** impiega ***hashing universale con concatenazione***
- Il ***secondo livello*** impiega una ***tabella hash secondaria*** (una per ogni cella della tabella di primo livello).
- La dimensione della tabella di secondo livello e la relativa funzione hash possono essere scelte in modo da garantire assenza di collisioni.

- Per l'hashing di ***primo livello***, definiamo una famiglia universale $H_{p,n} = \{(ak+b \bmod p) \bmod n \mid 0<a<p \text{ e } 0<b<p\}$ di funzioni hash, dove **p** è un numero primo maggiore di qualsiasi chiave e **n** è la dimensione della tabella.
- La funzione **h** hash di primo livello viene scelta tra quelle contenute in $H_{p,n}$.
- Per ogni cella j della tabella primaria, chiamo m_j il numero di chiavi che la funzione **h** associa alla cella j.
- Ad ogni cella j è associata una tabella S_j di dimensione n_j.
- Ad ogni cella j è associata una funzione hash selezionata dalla famiglia $H_{p,n_j} = \{(a_j k+b_j \bmod p) \bmod n_j \mid 0<a_j<p \text{ e } 0<b_j<p\}$.

Teorema. Sotto l'ipotesi di hashing universale, se memorizziamo **m** chiavi in una tabella di dimensione $n=m^2$, la probabilità che si verifichi una collisione è inferiore ad ½.

Teorema. Sotto l'ipotesi di hashing universale, se memorizziamo **m** chiavi in una tabella di dimensione $n=m^2$, la probabilità che si verifichi una collisione è inferiore ad ½.

Dim. Il numero di coppie di chiavi che possono collidere è pari al numero di coppie **(k,l)** con $k<l$, che è pari a $m(m-1)/2$.

Sotto ipotesi che la funzione di hashing **h** provenga da una famiglia universale, sappiamo che la probabilità di collisione tra due chiavi in una tabella di **n** celle è $1/n$.

Essendo $n=m^2$, avremo quindi che $\Pr[h(k)=h(l)] = 1/m^2$.

Detta **X** la variabile casuale che conta il numero di collisioni, avremo quindi

$$E[X] = 1/m^2 \cdot m(m-1)/2 = 1/m^2 \cdot (m^2-m)/2 < ½$$

La probabilità che si abbiano almeno t collisioni è $\Pr[X \geq t] \leq E[X]/t$ (per la disuguaglianza di Markov). Nel nostro caso t=1. Da cui possiamo concludere $\Pr[X \geq 1] < ½$.

Lemma. (Disuguaglianza di Markov). Sia **X** una variabile casuale, allora per ogni t>0 vale:

$$\mathbf{Pr}[\mathbf{X} \geq t] \leq \mathbf{E}[\mathbf{X}]/t$$

Dim. Introduciamo la variabile casuale indicatrice **I** definita come segue:

I = **1** se **X** ≥ t

I = **0** se **X** < t

0 ≤ **I** ≤ **X/t**. Infatti se **X** < t, **0** = **I** < **X**/t; se **X** ≥ t, **X**/t ≥ **1** = **I**.

Per la linearità di **E**[] avremo che

$$\mathbf{E}[\mathbf{I}] \leq \mathbf{E}[\mathbf{X}/t] = \mathbf{E}[\mathbf{X}]/t$$

Inoltre, per definizione di **I**, **Pr**[**X**≥t] = **E**[**I**].

Segue, quindi, **Pr**[**X**≥t] ≤ **E**[**X**]**/t**.

- Quindi, scegliendo n_j uguale a m_j^2 siamo sicuri che la probabilità di collisione sia sempre inferiore ad **½**. Cioè, presa una funzione di hashing da una famiglia universale, è assai probabile che non si verifichino collisioni.
- È possibile dimostrare che lo ***spazio medio necessario*** per memorizzare la *tabella principale* più *tutte le tabelle secondarie* non è superiore a **2·n**, dove **n** è il numero di chiavi da memorizzare.
- La figura illustra un esempio dello schema di ***hashing perfetto***.