

Sia  $\mathcal{U}$  un insieme con  $|\mathcal{U}| = m \gg 1$ . Consideriamo una tabella hash  $T$  di dimensione  $1 < n \ll m$  ed una funzione di hash  $h : \mathcal{U} \rightarrow \{0, \dots, n-1\}$ . Supponiamo di memorizzare un arbitrario sottoinsieme  $S \subseteq \mathcal{U}$  nella tabella hash usando la funzione  $h$ . Ovvero, memorizziamo ogni  $u \in S$  nella posizione  $h(u)$  di  $T$ . Dato che la tabella hash è molto più piccola della cardinalità di  $\mathcal{U}$ , potranno avvenire delle collisioni. Ovvero, potranno esserci elementi distinti  $u, v \in S$  tali che  $h(u) = h(v)$ . Assumendo quindi che le collisioni vengano gestite utilizzando, per esempio, delle liste associate ad ogni posizione di  $T$ , ogni ulteriore operazione (ricerca, inserimento o cancellazione) sulla tabella richiederà tempo proporzionale alla lunghezza della lista associata alla posizione della tabella dove viene fatta l'operazione.

In un'analisi di caso peggiore, il tempo per eseguire tale operazione è quindi proporzionale al massimo numero  $\ell_{\max}$  di elementi in  $S$  che sono stati mappati da  $h$  nella stessa posizione di  $T$ . Più precisamente,

$$\ell_{\max} = \max_{i=0, \dots, n-1} |\{u \in S : h(u) = i\}|.$$

Idealmente, vorremmo avere  $\ell_{\max} \simeq |S|/n$ , in modo da minimizzare il tempo di esecuzione di un'operazione nel caso peggiore.

Possiamo andare vicini a questo risultato usando la randomizzazione. Supponiamo di usare una funzione hash  $h$  che sia la realizzazione di una variabile casuale  $H$  corrispondente all'estrazione casuale (con probabilità uniforme) di un elemento da una famiglia  $\mathcal{H}$  di funzioni  $h : \mathcal{U} \rightarrow \{0, \dots, n-1\}$ . Supponiamo ora che  $\mathcal{H}$  soddisfa la condizione seguente

$$\mathbb{P}(H(u) = H(v)) \leq \frac{1}{n} \quad \text{per ogni } u, v \in \mathcal{U} \quad (1)$$

dove la probabilità è calcolata rispetto all'estrazione casuale della funzione hash  $H$  da  $\mathcal{H}$ .

La condizione (1) è sufficiente a garantire che

$$\max_{i=0, \dots, n-1} \mathbb{E} \left[ |\{u \in S : h(u) = i\}| \right] \leq \frac{|S|}{n}$$

dove il valore atteso è calcolato rispetto all'estrazione di  $H$  da  $\mathcal{H}$ . Infatti, supponiamo di aver estratto  $H$  a caso da  $\mathcal{H}$  e di averla usata per inserire  $S = \{u_1, \dots, u_s\}$  nella tabella. Sia  $u \in \mathcal{U}$  un elemento arbitrario che vogliamo aggiungere alla tabella. Definiamo le variabili casuali  $X_1, \dots, X_s$  dove  $X_i = \mathbb{I}\{H(u) = H(u_i)\}$  e  $\mathbb{I}\{\cdot\}$  è la funzione indicatrice di un evento, definita come

$$\mathbb{I}\{A\} = \begin{cases} 1 & \text{se } A \text{ è vero} \\ 0 & \text{altrimenti.} \end{cases}$$

Allora, il numero di elementi di  $S$  che sono mappati nella stessa posizione di  $u$  è dato da

$$\sum_{i=1}^s X_i.$$

Il valore atteso di questo numero è calcolato come

$$\begin{aligned}
\mathbb{E} \left[ \sum_{i=1}^s X_i \right] &= \sum_{i=1}^s \mathbb{E}[X_i] \quad \text{per linearità del valore atteso} \\
&= \sum_{i=1}^s \mathbb{P}(H(u) = H(u_i)) \quad \text{per definizione di } X_i \\
&\leq \sum_{i=1}^s \frac{1}{n} \quad \text{per ipotesi su } H \\
&= \frac{|S|}{n}.
\end{aligned}$$

Non è difficile trovare famiglie  $\mathcal{H}$  che soddisfano la condizione (1). Assumiamo per semplicità che  $\mathcal{U} \equiv \{0, \dots, m-1\}$  e consideriamo la classe  $\mathcal{H}$  di tutte le funzioni  $h : \{0, \dots, m-1\} \rightarrow \{0, \dots, n-1\}$ . Ognuna di queste funzioni hash corrisponde a un vettore  $\mathbf{h} = (h_0, \dots, h_{m-1}) \in \{0, \dots, n-1\}^m$  in modo che  $h(u) = h_u$ . Allora, il numero di vettori  $\mathbf{h} \in \{0, \dots, n-1\}^m$  tali che  $h_u = h_v$  è  $n^{m-1}$ . Quindi, se estraggo  $H$  a caso da  $\mathcal{H}$ ,

$$\mathbb{P}(H(u) = H(v)) = \frac{|\{\mathbf{h} \in \mathcal{H} : h_u = h_v\}|}{|\mathcal{H}|} = \frac{n^{m-1}}{n^m} = \frac{1}{n}$$

e la condizione (1) è soddisfatta. D'altra parte, la classe  $\mathcal{H}$  non è utilizzabile in pratica in quanto mi servono  $\lceil \log_2 |\mathcal{H}| \rceil = \Theta(m \log n)$  bit per memorizzare ciascuna  $h \in \mathcal{H}$  e stiamo assumendo che  $m \gg 1$ .

Per eliminare situazioni di questo genere, aggiungiamo alla condizione (1) una richiesta ulteriore, ovvero che ogni  $h \in \mathcal{H}$  possa essere rappresentata con al più  $\Theta(\log m)$  bit (che è anche lo spazio che mi occorre per rappresentare un elemento arbitrario di  $\mathcal{U}$ ) e calcolata in modo efficiente. Una famiglia  $\mathcal{H}$  di funzioni  $h : \mathcal{U} \rightarrow \{0, \dots, n-1\}$  che soddisfa entrambe queste condizioni è detta una *famiglia universale di funzioni hash*.

Dimostriamo ora l'esistenza di famiglie universali. Oltre a  $\mathcal{U} = \{0, \dots, m-1\}$  assumiamo anche  $n = p$  per un qualche  $p$  primo. Rappresentiamo ciascun elemento  $u \in \{0, \dots, m-1\}$  come un numero  $[u]_p$  in base  $p$ . Ovvero,  $[u]_p = x = (x_1, \dots, x_r)$  dove  $x_i \in [0, \dots, p-1]$  e  $r$  è il più piccolo intero tale che  $p^r \geq m$ . Ovvero,

$$r = \left\lceil \frac{\log_2 m}{\log_2 p} \right\rceil.$$

Sia  $\mathcal{A} = \{0, \dots, p-1\}^r$  l'insieme usato per rappresentare gli elementi di  $\mathcal{U}$ . Introduciamo ora la famiglia di funzioni hash  $\mathcal{H} = \{h_a : a \in \mathcal{A}\}$  di tipo  $h_a : \mathcal{A} \rightarrow \{0, \dots, p-1\}$  e definite come

$$h_a(u) = \left( \sum_{i=1}^r a_i x_i \right) \pmod{p} \quad \text{dove } x = [u]_p.$$

Si noti che posso rappresentare ogni  $h_a$  con  $\Theta(\log m)$  bit e posso calcolare  $h_a$  in modo efficiente. Per verificare la condizione (1) utilizziamo il lemma seguente.

**Lemma 1** Per ogni primo  $p$ , per ogni  $\alpha, \beta$  e  $z$  interi,

$$z \not\equiv 0 \pmod{p} \text{ e } \alpha z \equiv \beta z \pmod{p} \text{ implicano } \alpha \equiv \beta \pmod{p} .$$

DIMOSTRAZIONE. Chiaramente,  $\alpha z \equiv \beta z \pmod{p}$  se e solo se  $z(\alpha - \beta) \equiv 0 \pmod{p}$ . Inoltre,  $z \not\equiv 0 \pmod{p}$  e  $z(\alpha - \beta) \equiv 0 \pmod{p}$  implicano  $(\alpha - \beta) \equiv 0 \pmod{p}$ . Infatti, se  $p$  è primo e  $z$  non contiene  $p$  come fattore, allora  $\alpha - \beta$  lo deve contenere (se invece  $p$  non fosse primo, allora  $z$  e  $\alpha - \beta$  potrebbero spartirsi i fattori primi di  $p$ ).  $\square$

Siamo ora pronti a dimostrare che per  $\mathcal{H}$  vale la proprietà 1. Dato che, come abbiamo già osservato, per la stessa classe valeva anche la proprietà 2, concludiamo che  $\mathcal{H}$  è una famiglia universale di funzioni hash.

**Teorema 2**  $\mathcal{H}$  è tale che per ogni  $x, y \in \mathcal{A}$  distinti, la frazione di elementi  $a \in \mathcal{A}$  tali che  $h_a(x) = h_a(y)$  è al più  $\frac{1}{p}$ . Quindi, se  $H$  è estratta a caso da  $\mathcal{H}$ , allora

$$\mathbb{P}(H(x) = H(y)) \leq \frac{1}{p} .$$

DIMOSTRAZIONE. Se  $x \neq y$  allora esiste almeno una coordinata  $j \in \{0, \dots, r\}$  tale che  $x_j \neq y_j$ . Per estrarre  $H$  a caso in  $\mathcal{H}$  prendiamo  $a \in \mathcal{A}$  estraendo a caso ciascuna coordinata  $a_i \in \{0, \dots, p-1\}$ . Per qualunque estrazione dei valori  $a_i$  sulle coordinate  $i \neq j$ , abbiamo che

$$\begin{aligned} h_a(x) = h_a(y) &\iff \left( \sum_{i=1}^r a_i x_i \equiv \sum_{i=1}^r a_i y_i \right) \pmod{p} \\ &\iff a_j \underbrace{(x_j - y_j)}_z \equiv \underbrace{\sum_{i:i \neq j} a_i (x_i - y_i)}_k \pmod{p} \\ &\iff a_j z \equiv k \pmod{p} . \end{aligned}$$

Dimostriamo ora che esiste un unico  $a_j \in \{0, \dots, p-1\}$  tale che la congruenza  $a_j z \equiv k \pmod{p}$  sia soddisfatta. Per assurdo, supponiamo che esistano  $a_j, a'_j$  distinti che soddisfano entrambi la congruenza. Allora avremmo anche che  $a_j z \equiv a'_j z \pmod{p}$ . Dato che  $z \not\equiv 0 \pmod{p}$  per ipotesi, il lemma implica che  $a_j \equiv a'_j \pmod{p}$ . Siccome  $0 \leq a_j, a'_j < p$ , ciò implica che  $a_j = a'_j$  e si ha una contraddizione. Quindi c'è un solo valore della coordinata  $j$  che rende  $h_a(x) = h_a(y)$  e la probabilità di estrarre questo valore è al più  $\frac{1}{p}$ . Se  $x$  e  $y$  differiscono su più di una coordinata, allora la probabilità che  $h_a(x) = h_a(y)$  diventa ancora più piccola.  $\square$