

Lezione 10

ELEMENTI DELLA STRATEGIA GREEDY

Un algoritmo greedy ottiene una soluzione ottima ad un problema di ottimizzazione attraverso una successione di scelte. Ad ogni "bito", l'algoritmo fa le scelte che sembra migliori al momento.

È una strategia evolutiva che non sempre produce una soluzione ottima.

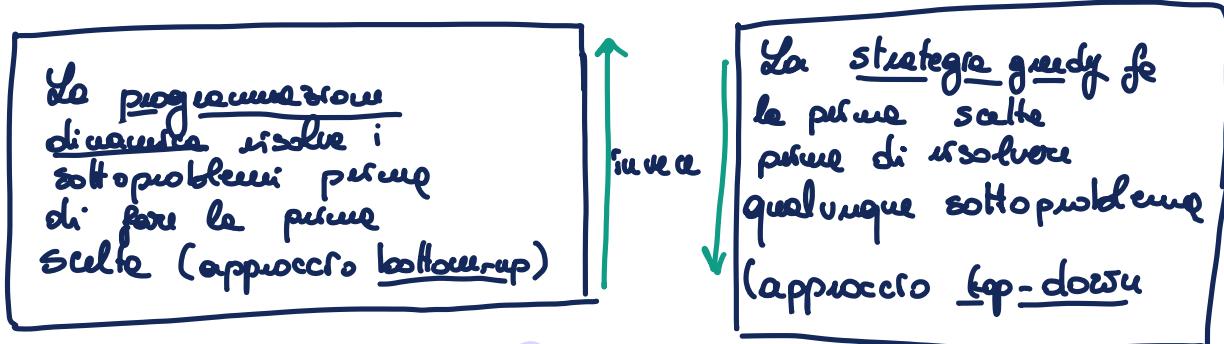
COME SVILUPPARE UN ALGORITMO GREEDY

1. Formulare il problema di ottimizzazione come uno nel quale si fa una scelta e si risolve con un solo problema.
2. Dimostrare che esiste sempre una soluzione ottima che contiene le scelte greedy, così che queste scelte siano "sicure".

3. Dimostrare le proprietà di sottostruttura ottima del problema.

PROPRIETÀ DI SCelta GREEDY

Un problema di ottimizzazione soddisfa la proprietà di scelta greedy se è possibile assemblare una soluzione ottima globale facendo scelte greedy che sono localmente ottime. Ovvero, se esiste una soluzione ottima che contiene la scelta greedy ad ogni iterazione.



COME SI DIMOSTA LA PROPRIETÀ DI SCELTA GREEDY?

Si esamina una soluzione ottima globale e poi si mostra come modificarla sostituendo le scelte greedy a sue altre scelte, ottenendo così un sotto problema simile a uno di dimensione più piccole.

COME SI DIMOSTA LA PROPRIETÀ DI SOTTOSTRUTTURA OTTIMA QUANDO SAPPIAMO GIÀ CHE VALE LA PROPRIETÀ DI SCELTA GREEDY?

Basta solo far vedere che combinando una soluzione ottima a un sotto problema \checkmark a sue scelte greedy si ottiene una soluzione globalmente ottima.

10.1 DEMOSTRAZIONE DELLA PROPRIETÀ DI SCELTA GREEDY

PER IL PROBLEMA ACTIVITY SELECTION

ACTIVITY SELECTION PROBLEM

- input:
- un insieme $S = \{a_1, \dots, a_n\}$ di attività;
 - per ogni attività a_i sono specificati un tempo di inizio (start time) s_i e un tempo di termine (finish time) f_i , $0 \leq s_i < f_i < \infty$ e $a_i = [s_i, f_i]$

goal: selezionare un sottoinsieme $A \subseteq S$ di attività tali che A abbia cardinalità massima e le attività in A siano compatibili, ovvero $\forall i, j \in A$ $[x_i, f_i] \cap [x_j, f_j] = \emptyset$.

Possiamo ordinare le attività su ordine crescente di tempo di termurazione: $f_1 \leq f_2 \leq \dots \leq f_m$

Euristiche: attività che termina prima.

L'algoritmo che ottengo ha complessità $O(n \log n)$ -Time.

TEOREMA (SCHEMA GREEDY DI ACTIVITY SELECTION): Sia S_k un sottoinsieme non vuoto, e sia $a_m \in S_k$ con il più piccolo tempo di termurazione. Allora, a_m è inclusa in qualche sottoinsieme di dimensione massima di attività compatibili di S_k , ovvero in una soluzione ottima ad S_k .

Proof: Supponiamo che A_k sia una soluzione ottima ad S_k .

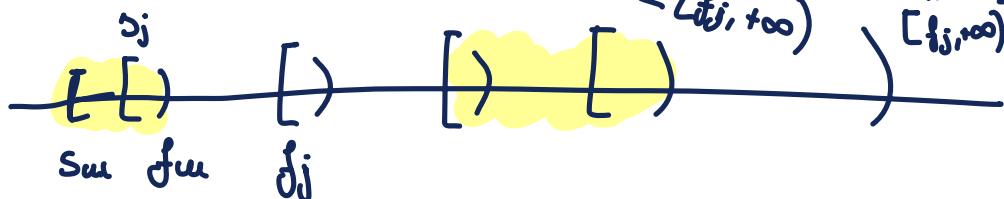
Sia a_j l'attività che finisce prima su A_k .

- se $a_m = a_j \Rightarrow a_m \in A_k \vee$ (abbiamo finito)
- se $a_m \neq a_j \Rightarrow a_m \notin A_k$

definiamo $A'_k := (A_k \setminus \{a_j\}) \cup \{a_m\}$

A_k' è soluzione: $(A_k \setminus \{a_j\}) \cap \{a_j\} = \emptyset$ e
oltre

che fruisce perche di a_j
 $\phi = (A_k \setminus \{a_j\}) \cap [s_j, f_j] \Rightarrow (A_k \setminus \{a_j\}) \cap [s_{k+1}, f_k]$
 $\leq [l_{kj}, +\infty) \quad \stackrel{\wedge}{=} \phi$
 $[f_{kj}, +\infty)$



A_k' è ottima: $|A_k'| = (|A_k|-1) + 1 = |A_k|.$
 $\Rightarrow A_k'$ ottima. \square

10.2 DIMOSTRAZIONE DELLA PROPRIETÀ DI SCelta GREEDY PER L'ALGORITMO DI HUFFMAN

L'algoritmo di Huffman costruisce un albero di codifica prefix-free, in cui le foglie corrispondono ai caratteri da codificare. Si rinterpreta la codifica di un carattere con il cammino (semplice) dalla radice al carattere:
"0": vai al nodo figlio sx,
"1": vai al nodo figlio dx.
L'albero ha $|C|$ foglie (C è l'alfabeto), una per ogni carattere e ha esattamente $|C|-1$ nodi interni.

NOTAZIONE:

Sia T un albero di codifica prefix-free.

$c.\text{freq}$ = frequenza del carattere c , $\forall c \in C$

$d_T(c)$: profondità di c nell'albero T = lunghezza delle codifiche di c (= # di bit necessari).

Il numero totale di bit per codificare un file è

$$B(T) = \sum_{c \in C} c.\text{freq} \cdot d_T(c) \quad (\text{costo dell'albero}).$$

EURISTICA PER LA SCELTA GREEDY : carattere più frequente.

Esiste sempre un albero ottimo (rispetto a $B(T)$) che contrarie le scelte greedy?

PROP (PROPRIETÀ DI SCELTA GREEDY PER I CODICI PREFIX-FREE):

Sia C un alfabeto su cui ogni carattere c ha frequenze $c.\text{freq}$. Siano $x, y \in C$ i caratteri con frequenze minime. Allora, esiste una codifica prefix-free ottima su cui le codifiche di x e y hanno le stesse lunghezze massime (-profondità nell'albero) e differiscono per l'ultimo bit.

PROOF:

Sia T un albero che rappresenta una codifica prefix-free ottima. Siano a e b i due

caratteri che corrispondono alle foglie di minore profondità di T .

Senza perdere generalità, possiamo assumere che

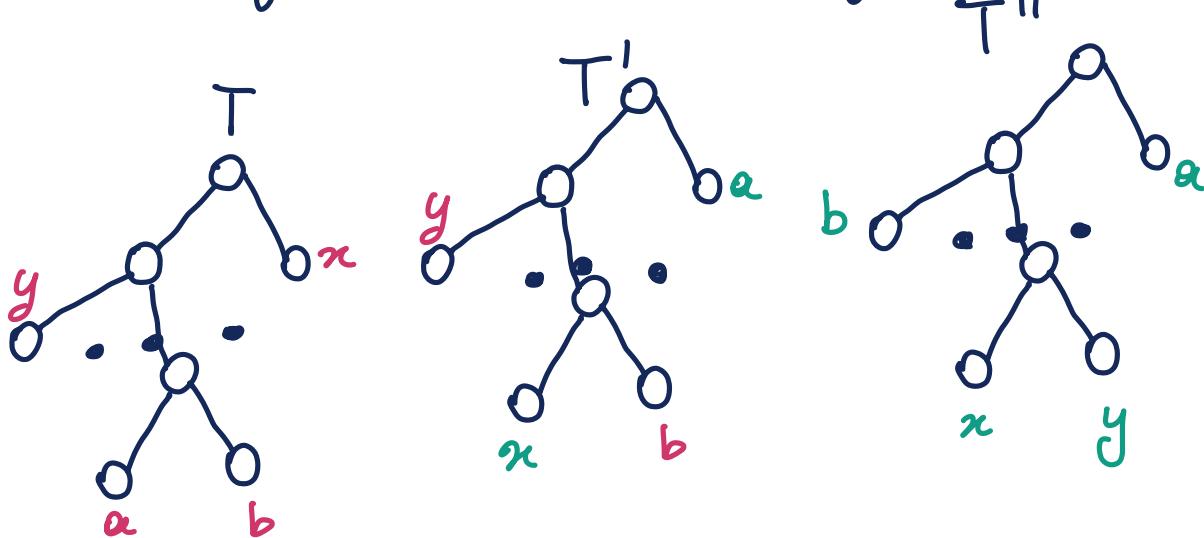
$$\begin{cases} a.freq \leq b.freq & e \\ x.freq \leq y.freq. \end{cases}$$

Averci che

$$\begin{cases} x.freq \leq a.freq \\ y.freq \leq b.freq \end{cases}.$$

Potrebbe accadere che $x.freq = a.freq$ oppure che $y.freq = b.freq$. Inoltre, se $x.freq = b.freq \Rightarrow$
 $x.freq = a.freq = b.freq = y.freq \Rightarrow$ la proposizione sarebbe immediatamente vera. Perciò possiamo assumere che
 $x.freq < b.freq$ ovvero che $x \neq b$.

Nell'albero T scegliiamo le posizioni di a e x
ottenendo T' e in T' scegliamo le posizioni
di b e y ottenendo T'' su cui x e y hanno profondità uguale.



Oss: Se $x=b$ ma $y \neq a$, T'' non ha x e y come foglie di profondità massima.

Facchiamo vedere che il costo di T' non è maggiore di quello di T , $B(T') \leq B(T) \Rightarrow \underline{B(T) - B(T') \geq 0}$

$$\begin{aligned}
 B(T) - B(T') &= \sum_{c \in C} c \cdot \text{freq} \cdot d_T(c) - \sum_{c \in C} c \cdot \text{freq} \cdot d_{T'}(c) = \\
 d_T(c) &= d_T(c) \quad \forall c \in C \setminus \{a, x\} \\
 &= x \cdot \text{freq} \cdot d_T(x) + a \cdot \text{freq} \cdot d_T(a) - x \cdot \text{freq} \cdot d_{T'}(x) - \\
 d_{T'}(x) &= d_T(a) \\
 d_{T'}(a) &= d_T(x) \\
 &= x \cdot \text{freq} \cdot d_T(x) + a \cdot \text{freq} \cdot d_T(a) - x \cdot \text{freq} \cdot d_T(a) - a \cdot \text{freq} \cdot d_T(x) = \\
 &= a \cdot \text{freq}(d_T(a) - d_T(x)) - x \cdot \text{freq}(d_T(a) - d_T(x)) = \\
 &= \underbrace{(a \cdot \text{freq} - x \cdot \text{freq})}_{\text{VII}} \underbrace{(d_T(a) - d_T(x))}_{\text{VII}} \geq 0
 \end{aligned}$$

$\Rightarrow B(T') \leq B(T)$. Analogamente, $B(T'') \leq B(T') \leq B(T)$.

□