Algoritmo Greedy per matroidi

Idea Generalizzare il problema di trovare, dato un grafo G con pesi nei lati, un'albero generante (foresta generante) di peso minimale.

↳ Questo problema sarà un caso particolare del seguente problema di ottimizzazione combinatoria:

→ Sia $\mathcal{I}$ una collezione di sottoinsiemi di un insieme E e supponiamo che $\mathcal{I}$ soddisfa (I1) e (I2).
Sia w una funzione peso su E ($w(x) \in \mathbb{R}, \forall x \in E$) con $w(\emptyset) = 0$

e sia $w(X) = \sum_{x \in X} w(x)$.

Problema: Trovare un elemento massimale B di $\mathcal{I}$ con peso massimale.

OSS. Data una soluzione per questo problema per $(\mathcal{I}, w)$, se consideriamo lo stesso problema per $(\mathcal{I}, -w)$, otteniamo anche una soluzione per $(\mathcal{I}, w)$ con peso minimale.

Quindi, il problema di trovare un'albero generante di un grafo connesso di peso minimale è un caso particolare del problema matroidale associato $(\mathcal{I}(G), w)$.

Vedremo: un'algoritmo Greedy che dà una soluzione al problema di ottimizzazione per $(\mathcal{I}, w)$ e come potremo usare il fatto che esiste una tale soluzione per garantire che $(E, \mathcal{I})$ ha una struttura matroidale.

Siano $\mathcal{I}$ e w come prima.

L'algoritmo Greedy per trovare una soluzione B per $(\mathcal{I}, w)$ (ossia, un'elemento massimale B di $\mathcal{I}$ con peso massimo) è il seguente:

(i) Definire $X_0 = \emptyset$ e $j=0$;

(ii) • Se $E - X_j$ contiene un elemento e t.c. $X_j \cup e \in \mathcal{I}$, scegliere uno tra questi elementi, e_{j+1}, di peso massimale. Definire $X_{j+1} = X_j \cup e_{j+1}$ e proseguire per (iii)

• altrimenti

$X_j = B_G$ e proseguire per (iv)

(iii) $j \leftarrow j+1$ e proseguire per (ii)

(iv) Stop : $\boxed{\text{Output} : B_G.}$

<u>Lemma</u> Se $(E, \mathcal{I})$ e' una matroide, allora B_G e' una soluzione al problema di ottimizzazione per $(\mathcal{I}, w)$.

<u>dim</u> Se $r(M) = r$, allora $B_G = \{e_1, .., e_r\}$ e' una base di M.

Vediamo che B_G ha peso massimale.

Sia $B = \{f_1, .., f_r\}$, con $w(f_1) \geq .. \geq w(f_r)$ un'altra base di M. Il risultato e' una conseguenza della seguente

<u>Asserzione</u>: se $1 \leq j \leq r$, $w(e_j) \geq w(f_j)$

<u>dim</u> p.a., sia k l'intero più piccolo t.c. $w(e_k) < w(f_k)$.

Siano $I_1 = \{e_1, .., e_{k-1}\}$ e $I_2 = \{f_1, .., f_k\}$.

Siccome $|I_1| < |I_2|$, $(I3) \Rightarrow \exists f_t \in I_2 \setminus I_1 : I_1 \cup f_t \in \mathcal{I}$.

Ma $w(f_t) \geq w(f_k) > w(e_k)$, quindi nell'algoritmo avremo preso f_t anziché e_k. $\blacksquare$

<u>Teorema</u> Sia $\mathcal{I}$ una collezione di sottoinsiemi di un'insieme E. Allora $(E, \mathcal{I})$ è una matroide sse $\mathcal{I}$ soddisfa:

(I1) $\emptyset \in \mathcal{I}$;

(I2) se $I \in \mathcal{I}$ e $I' \subset I$, allora $I' \in \mathcal{I}$;

(G) Per tutte le funzioni peso $w : E \to \mathbb{R}$, l'algoritmo Greedy produce un'elemento massimale in $\mathcal{I}$ di peso massimale.

<u>dim</u> Se $(E, \mathcal{I})$ è una matroide, allora (I1), (I2) e (G) valgono.

Supp. che $(E, \mathcal{I})$ è tale che valgono (I1), (I2) e (G) e vediamo che vale anche (I3).

Supp., p.a., che $I_1, I_2 \in \mathcal{I}$, con $|I_1| < |I_2|$ e che $I_1 \cup e \notin \mathcal{I}$, $\forall\ e \in I_2 - I_1$.

Ora, $|I_1 - I_2| < |I_2 - I_1|$ e $I_1 - I_2 \neq \emptyset$

$\Rightarrow \exists\ \varepsilon > 0 :\ 0 < (1+\varepsilon)|I_1 - I_2| < |I_2 - I_1|$

Definiamo $\omega : E \longrightarrow \mathbb{R}$

$$e \longmapsto \omega(e) := \begin{cases} 2 & \text{se } e \in I_1 \cap I_2 \\ \dfrac{1}{|I_1 - I_2|} & \text{se } e \in I_1 - I_2 \\ \dfrac{1+\varepsilon}{|I_2 - I_1|} & \text{se } e \in I_2 - I_1 \\ 0 & \text{altrimenti} \end{cases}$$

Allora l'algoritmo Greedy seleziona inizialmente tutti gli elementi di $I_1 \cap I_2$ e di seguito gli elementi di $I_1 - I_2$.

Inoltre, per ipotesi, non è possibile scegliere alcun elemento di $I_2 - I_1$, quindi gli elementi di B_G rimanenti sono elementi di $E - (I_1 \cup I_2)$.

Quindi

$$w(B_G) = 2|I_1 \cap I_2| + |I_1 - I_2| \cdot \frac{1}{|I_1 - I_2|}$$

$$= 2|I_1 \cap I_2| + 1 + 0$$

Ma, per (I_2), I_2 è contenuto in un elemento di $\mathcal{I}$ massimale. Sia I_2' questo elemento. Allora

$$w(I_2') \geq w(I_2) = 2|I_1 \cap I_2| + |I_2 - I_1| \frac{1+\varepsilon}{|I_2 - I_1|}$$

$$= 2|I_1 \cap I_2| + 1 + \varepsilon.$$

Quindi $w(I_2') > w(B_G)$, e l'algoritmo Greedy avrebbe dato un risultato sbagliato. $\not\!\!\delta$ $\boxtimes$

Applicazione: Assegnazione di lavori a lavoratori con diverse competenze. ⇝ matroidi trasversali

Supp. di avere una lista di lavori da eseguire (tasks) ognuna realizzabile da un solo lavoratore, con un'ordine di priorità.

Supp. ancora di avere a disposizione una lista di lavoratori, ciascuno con conoscenze adeguate a eseguire alcuni di quei lavori.

⇝ Pb. Cercare di attribuire a ciascun lavoratore un compito così da riuscire a realizzare il più possibile dei lavori da eseguire e rispettando le priorità.

Formulazione del problema in termine di matroidi trasversali.

S : insieme dei lavori da eseguire

Y : insieme dei lavoratori

$\forall y \in Y$, sia $A_y \subset S$ l'insieme dei lavori che y è capace di eseguire.

Sia $\mathcal{A} = (A_y, y \in Y)$

$\rightsquigarrow$ Il massimo numero di lavori che è possibile eseguire allo stesso tempo è la cardinalità del più grande trasversale parziale di $M[\mathcal{A}]$, ossia, è il rango di $M[\mathcal{A}]$.

Consideriamo adesso una funzione

$p: S \longrightarrow \mathbb{R}$

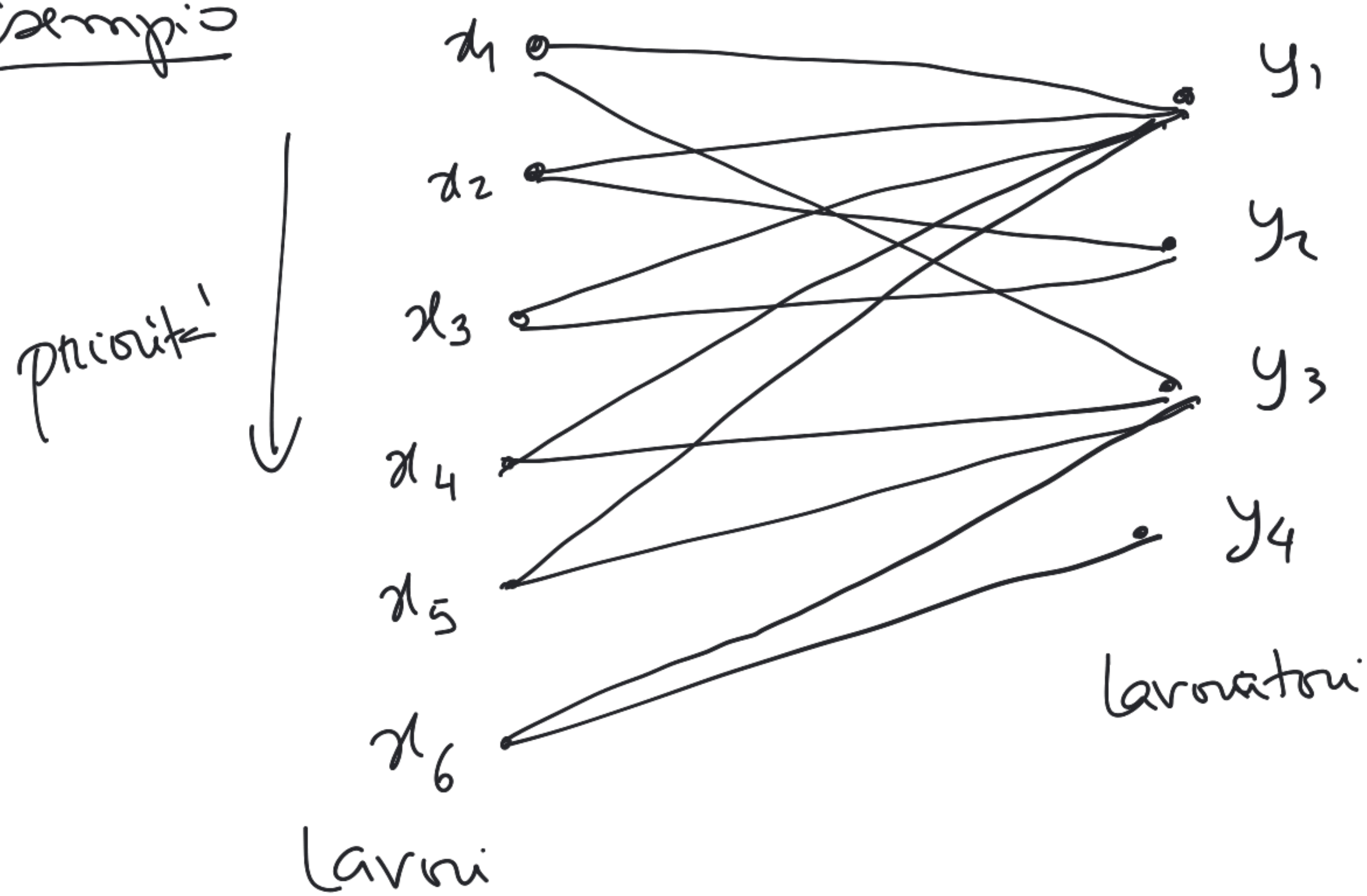
che corrisponde alla priorità nella esecuzione dei lavori. Un'assegnazione massimale di lavori è una base di $M[A]$. $\rightsquigarrow$ come trovare una base che soddisfa anche i criteri di priorità?

Supp. che un valore di p minore significa una più alta priorità nell'esecuzione del lavoro. Consideriamo una lista di lavori $B = \{x_1, \dots, x_n\}$, con $p(x_1) \leq \dots \leq p(x_n)$ e diciamo che B è ottimale se data un'altra lista $\{z_1, \dots, z_n\}$, con $p(z_1) \leq \dots \leq p(z_n)$, abbiamo che

$$p(z_i) \leq p(x_i), \quad \forall i \in \{1, \dots, n\}.$$

Se $\mathcal{I}$ è la collezione degli indipendenti di $M[A]$ e se $w = -p$, allora l'algoritmo Greedy che abbiamo visto produce una lista di lavori eseguibile ottimale!

Esempio



Lavori eseguibili ottimali:

$\{x_1, x_2, x_4, x_6\}$

(output dell'algoritmo Greedy)