Ricerca Operativa

Ahmad Shatti

2020-2021

Indice

Chapter 1

Problemi e modelli

1.1 Problema di ottimizzazione

Un problema di ottimizzazione trova la migliore soluzione fra tutte le soluzioni ammissibili, cioè massimizza o minimizza una funzione sulla regione ammissibile, data dai vincoli.

$$\begin{cases} max \;\; 3x+7y & \text{Funzione obiettivo} \\ \quad x+y \leq 6 & \text{Vincolo} \\ \quad 2x-3y \leq 5 & \text{Vincolo} \end{cases}$$

1.2 Problema decisionale

Un problema decisionale riguarda un problema di scelta in cui si deve prendere una decisione tra le soluzioni ammissibili, sulla base di uno o più criteri, dove la risposta può essere solo *Sì* o *No.* Da un problema di ottimizzazione ci si può ricondurre a un problema decisionale.

$$\begin{cases} max \;\; 3x+7y & \text{Funzione obiettivo} \\ \quad x+y \leq 6 & \text{Vincolo} \\ \quad 2x-3y \leq 5 & \text{Vincolo} \end{cases}$$

Fissato una soglia v=10, esiste un valore di x e y che rispettano i vincoli e per cui la funzione obiettivo è $\geq$ v?
La risposta può essere Sì o No, quindi questo è un problema decisionale associato al problema di ottimizzazione.

1.3 Problema dello zaino

Dati n oggetti di valore V_i e peso P_i e un contenitore di capacità C, si deve scegliere quali oggetti inserire nel contenitore, rispettando la sua capacità e in modo di massimizzare il valore totale. Può essere formulato come:

$$\text{Variabili:} \qquad x_j = \begin{cases} 1 & \text{se oggetto } j \text{ viene inserito,} \\ 0 & \text{altrimenti.} \end{cases}$$

Modello:

$$\begin{cases} \max \sum_{j=1}^{n} v_j \, x_j \\ \sum_{j=1}^{n} p_j \, x_j \leq C \\ x_j \in \{0,1\}, \; j = 1, \dots, n \end{cases}$$

Il problema viene risolto con il metodo **Branch and Bound**.

1.4 Problema del Bin Packing

Dati n oggetti di peso P_i e M contenitori di capacità C, trovare il minimo numero di contenitori in cui inserire *tutti* gli oggetti. Il problema non è scegliere quali oggetti inserire, ma si devono inserire tutti con l'obiettivo di minimizzare il numero di contenitori utilizzati.

Variabili: $x_{ij} = \begin{cases} 1 & \text{se l'oggetto } j \text{ è inserito nel contenitore } i, \\ 0 & \text{altrimenti,} \end{cases}$ $\quad y_i = \begin{cases} 1 & \text{se } i \text{ è usato,} \\ 0 & \text{altrimenti.} \end{cases}$

Modello:

$$\min \sum_{i=1}^{m} y_i$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\, j = 1, \dots, n \tag{1}$$

$$\sum_{j=1}^{n} p_j\, x_{ij} \leq C\, y_i \quad \forall\, i = 1, \dots, m \tag{2}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, i, j$$

$$y_i \in \{0, 1\} \qquad \forall\, i$$

(1): ogni oggetto è inserito in un solo contenitore (semiassegnamento)
(2): capacità contenitori

x_{ij} indica in quale contenitore va inserito l'oggetto, ma non indica se il contenitore è stato utilizzato, quindi serve un'altra variabile y_i che indica se il contenitore è stato utilizzato. La funzione obiettivo è la somma dei contenitori utilizzati. Il primo vincolo indica che tutti gli oggetti devono essere inseriti, mentre nel secondo vincolo viene imposto che il peso totale degli oggetti che si inseriscono in un certo contenitore non superi la capacità.

1.5 Problema di selezione di sottoinsiemi

Dati un insieme di elementi I e una famiglia S di sottoinsiemi di I, e un un costo per ogni sottoinsieme S_i:

- **Problema di copertura** determina una sottofamiglia F di **costo minimo** tale che ogni elemento di I appartenga ad **almeno** un sottoinsieme di F. Scegliere il minor numero di sottoinsiemi in modo da coprire I;
- **Problema di partizione** determina una sottofamiglia F di **costo minimo** tale che ogni elemento di I appartenga **esattamente** un sottoinsieme di F. Scrivere I come l'unione di sottoinsiemi disgiunti;
- **Problema di riempimento** determina un sottofamiglia F di **valore massimo** tale che ogni elemento appartenga ad **al più** un sottoinsieme di F. Cercare di inserire più elementi possibili, con intersezione tra questi elementi pari a 0.

Esempio: I={1, 2, 3, 4, 5, 6}
S_1= {1, 3}, S_2= {2, 4}, S_3={2, 5, 6}, S_4={5, 6}, S_5={3, 4}

F_1= {S_1, S_2, S_3} è una copertura
F_2= {S_1, S_2, S_4} è una partizione
F_3= {S_3, S_5} è un riempimento

Definiamo la matrice: $a_{ij} = \begin{cases} 1 & \text{se } i \in S_j, \\ 0 & \text{altrimenti.} \end{cases}$

Ad ogni sottofamiglia $\mathcal{F}$ associamo una variabile x, dove $x_j = \begin{cases} 1 & \text{se } S_j \in \mathcal{F}, \\ 0 & \text{altrimenti.} \end{cases}$

Copertura

$$\begin{cases} \min \sum_{j=1}^{n} c_j\, x_j \\ \sum_{j=1}^{n} a_{ij}\, x_j \geq 1 \quad \forall\, i \\ x_j \in \{0, 1\} \quad \forall\, j \end{cases}$$

Partizione

$$\begin{cases} \min \sum_{j=1}^{n} c_j\, x_j \\ \sum_{j=1}^{n} a_{ij}\, x_j = 1 \quad \forall\, i \\ x_j \in \{0, 1\} \quad \forall\, j \end{cases}$$

Riempimento

$$\begin{cases} \max \sum_{j=1}^{n} c_j\, x_j \\ \sum_{j=1}^{n} a_{ij}\, x_j \leq 1 \quad \forall\, i \\ x_j \in \{0, 1\} \quad \forall\, j \end{cases}$$

Nelle righe della matrice ci sono gli elementi dell'insieme I, nelle colonne i sottoinsiemi.

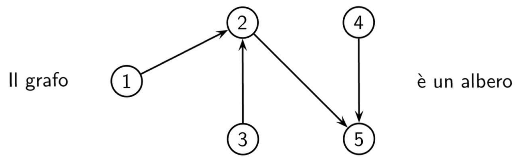
Chapter 2

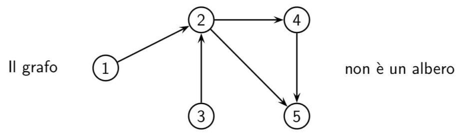
Albero di copertura di costo minimo

2.1 Albero

Un grafo orientato è un albero se è connesso e non contiene cicli.

- Se un albero ha N nodi, allora ha N-1 archi.
- L'albero contiene almeno due nodi che hanno un solo arco incidente (detti foglie)
- Esiste un unico cammino

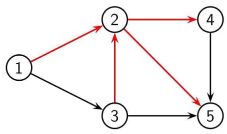




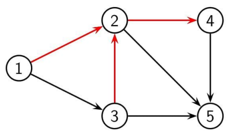
Il secondo grafo non è un albero perché contiene un ciclo cioè nodi = archi = 5.

2.1.1 Albero di copertura

Dato un grafo orientato, un albero di copertura è un insieme di archi tale che il sottografo sia un albero.



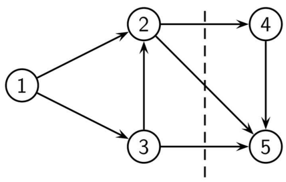
$T = \{(1,2),(2,4),(3,2),(2,5)\}$ è un albero di copertura



$T = \{(1,2),(2,4),(3,2)\}$ non è un albero di copertura

2.2 Taglio

Un taglio è una partizione dell'insieme dei nodi, cioè N',N"$\subseteq$ N, N' $\cap$ N" = 0, N' $\cup$ N"=N.
Gli archi del taglio sono gli archi aventi un estremo in N' e l'altro in N".



un taglio è (N', N''), dove $N' = \{1,2,3\}$ e $N'' = \{4,5\}$.
Gli archi del taglio sono (2,4), (2,5), (3,5).

2.3 Albero di copertura di costo minimo

Dato un grafo non orientato, in cui ogni arco è associato un costo c_{ij}, si deve trovare un albero di copertura T di costo minimo

Sia $N = \{1, \ldots, n\}$.

Per ogni $i, j \in N$ con $i < j$, definiamo le variabili decisionali:

$$x_{ij} = \begin{cases} 1 & \text{se l'albero contiene l'arco } (i,j), \\ 0 & \text{altrimenti.} \end{cases}$$

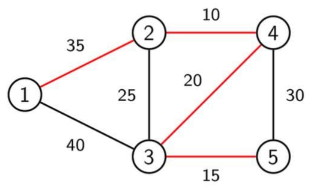
Modello di ottimizzazione:

$$\begin{cases} \min \displaystyle\sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij} x_{ij} & \\ \displaystyle\sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} x_{ij} = n - 1 & \text{(l'albero contiene } n-1 \text{ archi)} \\ \displaystyle\sum_{i \in S} \sum_{\substack{j \in S \\ j > i}} x_{ij} \leq |S| - 1 & \forall\, S \subset N \text{ tale che } |S| \geq 3 \\ & \text{(eliminazione dei cicli)} \\ x_{ij} \in \{0, 1\} & i, j \in N \text{ con } i < j \end{cases}$$

Il costo dell'albero è dato dalla somma dei cammini.

2.3.1 Condizioni di ottimalità basata sui cicli

T è un albero di copertura di costo minimo se e solo se per ogni arco (u,v) $\notin$ T si ha $c_{uv} \geq c_{ij}$ per ogni arco (i,j) appartenente al ciclo ottenuto aggiungendo all'arco (u,v).



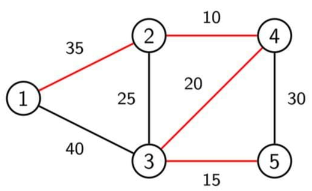
Applicando la condizione di ottimalità basata sui cicli, dire se l'albero di copertura $T_1 = \{(1,2), (2,4), (3,4), (3,5)\}$ è ottimo.

$(u,v) \notin T_1$	c_{uv}	ciclo C	altri archi di C	costi	cond. ott. vera
(1,3)	40	1-2-4-3-1	(1,2) (2,4) (3,4)	35, 10, 20	si
(2,3)	25	2-4-3-2	(2,4) (3,4)	10, 20	si
(4,5)	30	4-3-5-4	(3,4) (3,5)	20, 15	si

Quindi T_1 è un albero di copertura di costo minimo.

2.3.2 Condizioni di ottimalità basata sui tagli

T è un albero di copertura di costo minimo se e solo se per ogni arco (u,v) $\in$ T si ha $c_{uv} \leq c_{ij}$ per ogni arco (i,j) del taglio ottenuto eliminando dall'arco (u,v)



Applicando la condizione di ottimalità basata sui tagli, dire se l'albero di copertura $T_1 = \{(1,2), (2,4), (3,4), (3,5)\}$ è ottimo.

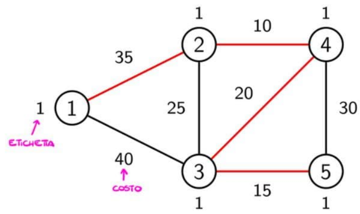
$(u,v) \in T_1$	c_{uv}	taglio (N', N'')	altri archi taglio	costi	cond. vera
(1,2)	35	({1}, {2,3,4,5})	(1,3)	40	si
(2,4)	10	({1,2}, {3,4,5})	(1,3) (2,3)	40, 25	si
(3,4)	20	({1,2,4}, {3,5})	(1,3) (2,3) (4,5)	40, 25, 30	si
(3,5)	15	({1,2,3,4}, {5})	(4,5)	30	si

Quindi T_1 è un albero di copertura di costo minimo.

2.4 Algoritmo di Kruskal

L'algoritmo di Kruskal trova un albero di copertura di costo minimo in tempo polinomiale.

1. Ordina gli archi in ordine crescente di costo e associa ad ogni nodo una etichetta d.
   T=0 e K=1 dove T è l'albero di copertura e K un indice
2. Se T = N - 1 allora stop (Se T è formato dal numero di Nodi - 1)
3. Sia l'arco $a_k = (p,q)$. Se $d_p \neq d_q$ (cioè non si forma un ciclo), allora si aggiunge l'arco a T e d=min$\{d_p, d_q\}$ si aggiornano le etichette.
4. K++ e torna al passo 2



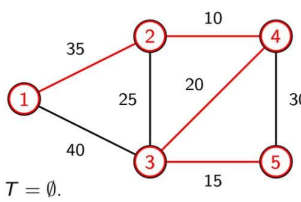
Archi in ordine crescente di costo: $\{(2,4),(3,5),(3,4),(2,3),(4,5),(1,2),(1,3)\}$.

$T = \emptyset$,	analizzo $(2,4)$: $d_2 = d_4$? NO
$T = \{(2,4)\}$,	analizzo $(3,5)$: $d_3 = d_5$? NO
$T = \{(2,4),(3,5)\}$,	analizzo $(3,4)$: $d_3 = d_4$? NO
$T = \{(2,4),(3,5),(3,4)\}$,	analizzo $(2,3)$: $d_2 = d_3$? SI
$T = \{(2,4),(3,5),(3,4)\}$,	analizzo $(4,5)$: $d_4 = d_5$? SI
$T = \{(2,4),(3,5),(3,4)\}$,	analizzo $(1,2)$: $d_1 = d_2$? NO
$T = \{(2,4),(3,5),(3,4),(1,2)\}$	STOP

2.5 Algoritmo di Prim

L'algoritmo di Prim trova un albero di copertura di costo minimo in tempo polinomiale.

1. Si sceglie un nodo i, poni S=i e T = 0, dove S indica quali sono i nodi connessi
2. Se T = N - 1 allora stop (Se T è formato dal numero di Nodi - 1)
3. Tra gli archi appartenenti al taglio (S, N \ S) aggiungi a T un arco (u,v) di costo minimo
4. Inserisci in S l'estremo di (u,v) che non appartiene ad S e torna al passo 2.



Scegliamo $i = 1$, $S = \{1\}$, $T = \emptyset$.
Taglio $(\{1\},\{2,3,4,5\})$, $(1,2) = \arg\min\{c_{12}, c_{13}\}$, $T = \{(1,2)\}$, $S = \{1,2\}$
Taglio $(\{1,2\},\{3,4,5\})$, $(2,4) = \arg\min\{c_{13}, c_{23}, c_{24}\}$, $T = \{(1,2),(2,4)\}$, $S = \{1,2,4\}$
Taglio $(\{1,2,4\},\{3,5\})$, $(3,4) = \arg\min\{c_{13}, c_{23}, c_{34}, c_{45}\}$, $T = \{(1,2),(2,4),(3,4)\}$,
$S = \{1,2,3,4\}$
Taglio $(\{1,2,3,4\},\{5\})$, $(3,5) = \arg\min\{c_{35}, c_{45}\}$, $T = \{(1,2),(2,4),(3,4),(3,5)\}$,
$S = \{1,2,3,4,5\}$ STOP

Chapter 3

Problema dei cammini minimi

3.1 Definizione

Dato un grafo orientato in cui è definito un costo c_{ij} per ogni arco e, dati un nodo origine s e un nodo destinazione t, trovare un cammino orientato da s a t di costo minimo. Dove il costo di un cammino è definito come la somma dei costi degli archi da cui è formato (somma potenziali).

Variabili: per ogni $(i, j) \in A$, definiamo x_{ij} = numero di volte che si attraversa l'arco (i, j) [flusso sull'arco (i, j)].

Funzione obiettivo: $\sum_{(i,j)\in A} c_{ij}x_{ij}$

Vincoli:

$$\sum_{(i,s)\in A} x_{is} - \sum_{(s,j)\in A} x_{sj} = -1 \qquad \text{(1 unità di flusso deve uscire da } s\text{)}$$

$$\sum_{(i,t)\in A} x_{it} - \sum_{(t,j)\in A} x_{tj} = 1 \qquad \text{(1 unità di flusso deve entrare in } t\text{)}$$

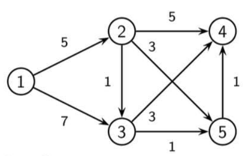
$$\sum_{(i,k)\in A} x_{ik} - \sum_{(k,j)\in A} x_{kj} = 0 \qquad \forall\, k \in N \setminus \{s, t\}$$

(i nodi diversi da s e t sono nodi di transito)

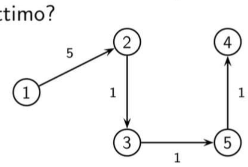
$\mathbf{x_{ij}}$ indica il numero di volte che si attraversa l'arco, cioè il **flusso**.
Il flusso si calcola facendo *flusso entrante - flusso uscente*. Il flusso per il nodo origine deve essere pari a -1 perchè si può solamente uscire. Il flusso per il nodo destinazione deve essere 1 perchè si può solo entrare. Tutti gli altri nodi devono avere flusso pari a 0. Il problema dei cammini minimi ammette almeno una soluzione ottima se e solo se non esistono cicli orientati di costo negativo.

3.1.1 Condizioni di Bellman

Un *potenziale* di un nodo i, è il costo che va dalla radice fino al nodo i. T è un albero dei cammini minimi se e solo se $\pi_j \leq \pi_i + c_{ij}$ per ogni arco (i,j) **non** appartenente a T.



Il seguente albero è ottimo?



I potenziali dei nodi sono $\pi = (0, 5, 6, 8, 7)$ Controlliamo le condizioni di Bellman:
arco $(1, 3)$: $6 = \pi_3 \leq \pi_1 + c_{13} = 0 + 7$? SI.
arco $(2, 4)$: $8 = \pi_4 \leq \pi_2 + c_{24} = 5 + 5$? SI.
arco $(2, 5)$: $7 = \pi_5 \leq \pi_2 + c_{25} = 5 + 3$? SI.
arco $(3, 4)$: $8 = \pi_4 \leq \pi_3 + c_{34} = 6 + 3$? SI. Pertanto l'albero è ottimo.

3.2 Algoritmo di Dijkstra

L'algoritmo trova un albero ottimo nel caso in cui il costo degli archi sia ≥ 0. Se nel grafo esistono archi di costo negativo, l'algoritmo non trova necessariamente un albero dei cammini minimi.

0. Poni

$$p_i = \begin{cases} 0 & \text{se } i = r \\ -1 & \text{se } i \neq r \end{cases} \qquad \pi_i = \begin{cases} 0 & \text{se } i = r \\ +\infty & \text{se } i \neq r \end{cases} \qquad U = N$$

(l'albero inizialmente è costituito solo da archi fittizi da r agli altri nodi)

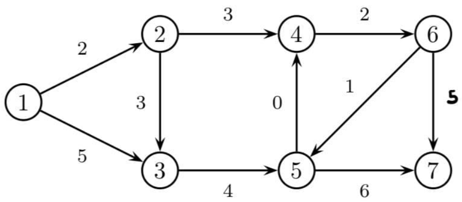
1. Se $U = \emptyset$ allora stop

2. Seleziona un nodo $u \in U$ con potenziale minimo: $u = \arg\min_{i \in U} \pi_i$

3. Per ogni arco $(u, v) \in A$ controlla la condizione di Bellman:
se $\boxed{\pi_v > \pi_u + c_{uv}}$ allora $p_v = u$, $\pi_v = \pi_u + c_{uv}$ HO TROVATO UN CAMMINO PER ARRIVARE AL NODO V CHE COSTA MENO RISPETTO AL CAMMINO CHE HO ATTUALMENTE ⇒ AGGIORNO L'ALBERO

4. $U = U \setminus \{u\}$ e torna al passo 1.

Si esaminano gli archi uscenti. Applicare l'algoritmo di Dijkstra per trovare l'albero dei cammini minimi di radice 1 sul grafo:



Per ciascuna iterazione la tabella riporta il nodo selezionato u, il vettore p dei predecessori, il vettore π delle etichette e l'insieme U dei nodi da esaminare:

Iter.	u	p	π	U
0	-	$(0,-1,-1,-1,-1,-1,-1)$	$(0,\infty,\infty,\infty,\infty,\infty,\infty)$	$\{1,2,3,4,5,6,7\}$
1	1	$(0,1,1,-1,-1,-1,-1)$	$(0,2,5,\infty,\infty,\infty,\infty)$	$\{2,3,4,5,6,7\}$
2	2	$(0,1,1,2,-1,-1,-1)$	$(0,2,5,5,\infty,\infty,\infty)$	$\{3,4,5,6,7\}$
3	3	$(0,1,1,2,3,-1,-1)$	$(0,2,5,5,9,\infty,\infty)$	$\{4,5,6,7\}$
4	4	$(0,1,1,2,3,4,-1)$	$(0,2,5,5,9,7,\infty)$	$\{5,6,7\}$
5	6	$(0,1,1,2,6,4,6)$	$(0,2,5,5,8,7,12)$	$\{5,7\}$
6	5	$(0,1,1,2,6,4,6)$	$(0,2,5,5,8,7,12)$	$\{7\}$
7	7	$(0,1,1,2,6,4,6)$	$(0,2,5,5,8,7,12)$	$\emptyset$

3.3 Algoritmo di Bellman-Ford

L'algoritmo di Bellman-Ford trova un albero ottimo anche nel caso in cui nel grafo ci siano archi di costo negativo. Ad ogni iterazione k l'algoritmo mantiene un albero di copertura radicato in r e orientato e un vettore π^k di etichette associate ai nodi.

0. Poni

$$p_i = \begin{cases} 0 & \text{se } i = r \\ -1 & \text{se } i \neq r \end{cases} \qquad \pi_i^0 = \begin{cases} 0 & \text{se } i = r \\ +\infty & \text{se } i \neq r \end{cases} \qquad k = 1$$

(l'albero inizialmente è costituito solo da archi fittizi da r agli altri nodi)

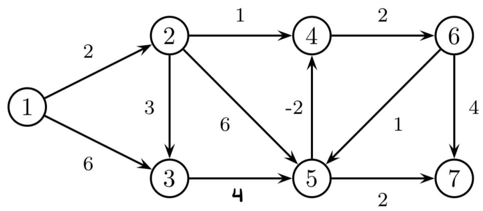
1. Per ogni nodo $j \in N$:
trova $u = \arg\min_{i \in BS(j)} \{\pi_i^{k-1} + c_{ij}\}$
$[BS(j) = \{i \in N : \text{ esiste un arco } (i, j) \in A\}$ è la stella entrante in $j]$
se $\pi_j^{k-1} > \pi_u^{k-1} + c_{uj}$ allora $p_j = u$, $\pi_j^k = \pi_u^{k-1} + c_{uj}$
altrimenti $\pi_j^k = \pi_j^{k-1}$

2. Se $\pi^k = \pi^{k-1}$ allora stop (p fornisce un albero ottimo)

3. Se $k = |N|$ allora stop (p fornisce un ciclo orientato di costo negativo)
altrimenti $k = k + 1$ e torna al passo 1.

Si esaminano a ogni iterazione tutti i nodi e si controllano gli archi entranti

Applicare l'algoritmo di Bellman-Ford per trovare l'albero dei cammini minimi di radice 1 sul grafo:



La tabella seguente riporta, per ciascuna iterazione dell'algoritmo, il vettore p dei predecessori ed il vettore d delle etichette dei nodi:

Iter	p_1	p_2	p_3	p_4	p_5	p_6	p_7	d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	0	-1	-1	-1	-1	-1	-1	0	∞	∞	∞	∞	∞	∞
1	0	1	1	-1	-1	-1	-1	0	2	6	∞	∞	∞	∞
2	0	1	2	2	2	-1	-1	0	2	5	3	8	∞	∞
3	0	1	2	2	2	4	5	0	2	5	3	8	5	10
4	0	1	2	2	6	4	6	0	2	5	3	6	5	9
5	0	1	2	2	6	4	5	0	2	5	3	6	5	8
6	0	1	2	2	6	4	5	0	2	5	3	6	5	8

3.4 Algoritmo di programmazione dinamica

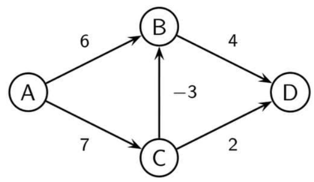
Se il grafo è aciclico, cioè non esistono cicli orientati, allora un albero dei cammini minimi si può trovare utilizzando l'algoritmo di programmazione dinamica.

Prima di tutto, è necessario fare un **ordinamento topologico** dei nodi:

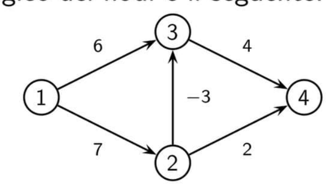
1. Si assegna 1 alla radice r, si elimina dal grafo r e tutti gli archi uscenti da r. Pongo k=2
2. Assegno k ad un nodo i che non ha archi entranti
3. Elimino il nodo i e tutti gli archi uscenti da i
4. Se il grafo è vuoto, allora stop, altrimenti k=k+1 e torna al passo 2

L'algoritmo di programmazione dinamica per trovare un albero di cammini minimi di radice 1 è il seguente:

1. Poni π_1=0
2. Per ogni nodo j=2, ..., n: trova u = argmin{ $\pi_i + c_{ij}$ }, poni p_j=u $\pi_j = \pi_u + c_{uj}$



L'ordinamento topologico dei nodi è il seguente:



Algoritmo di programmazione dinamica:
Nodo 1: $\pi_1 = 0$.
Nodo 2: $u = 1$, $p_2 = 1$, $\pi_2 = 7$
Nodo 3: $u = \arg\min\{\pi_1 + c_{13}, \pi_2 + c_{23}\} = 2$, $p_3 = 2$, $\pi_3 = 4$
Nodo 4: $u = \arg\min\{\pi_2 + c_{24}, \pi_3 + c_{34}\} = 3$, $p_4 = 3$, $\pi_4 = 8$

Chapter 4

Problema di flusso massimo

4.1 Definizione

Sia un grafo orientato in cui per ogni arco è definita una **capacità superiore $\mathbf{u_{ij}}$**. Dati un nodo origine s ed un nodo destinazione t, spediamo la massima quantità possibile di flusso da s a t in modo da rispettare le capacità superiori degli archi.

Variabili:
x_{ij} = flusso sull'arco (i, j), per ogni $(i, j) \in A$
v = flusso totale uscente dal nodo s (o flusso totale entrante nel nodo t)

Modello di PL:

$$\max \ v$$

$$\left.\begin{array}{ll} \sum_{(i,s)\in A} x_{is} - \sum_{(s,j)\in A} x_{sj} = -v & \\ \sum_{(i,t)\in A} x_{it} - \sum_{(t,j)\in A} x_{tj} = v & \\ \sum_{(i,k)\in A} x_{ik} - \sum_{(k,j)\in A} x_{kj} = 0 & \forall\ k \in N \setminus \{s, t\} \end{array}\right\} \text{vincoli di bilancio}$$

$$0 \le x_{ij} \le u_{ij} \qquad \forall\ (i, j) \in A \qquad \text{vincoli di capacità}$$

Nel problema di cammino minimo si conosce che vi è una unità di flusso che va da s a t, perché si vuole trovare un cammino. Nel problema di flusso massimo si vuole massimizzare il flusso, quindi non si conosce v.

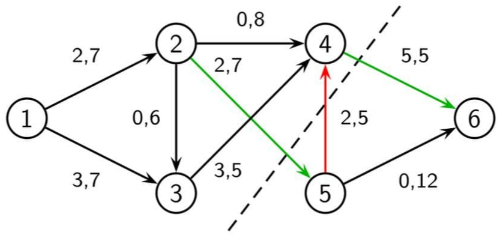
4.2 Flussi e tagli

- Un **taglio** è una partizione di N in due sottoinsiemi (N_s, N_t)
- Un **taglio ammissibile** è un taglio (N_s, N_t) tale che $s \in N_s$ e $t \in N_t$.
- Dato un taglio ammissibile (N_s, N_t) ed un flusso x, si definiscono:
  1. A^+ = arco(i,j) : $i \in N_s$, $j \in N_t$ insieme degli **archi diretti** del taglio
  2. A^- = arco(i,j) : $i \in N_t$, $j \in N_s$ insieme degli **archi inversi** del taglio
- La **capacità del taglio** è la sommatoria della **capacità superiore** degli **archi diretti**
- Il **flusso sul taglio** è la sommatoria del **flusso** sugli **archi diretti** - sommatoria del **flusso** sugli **archi inversi**

Se x è un flusso ammissibile e (N_s, N_t) è un taglio ammissibile, allora:

- $v = x(N_s, N_t)$ (valore del flusso = flusso sul taglio)
- $x(N_s, N_t) \le u(N_s, N_t)$ (flusso sul taglio $\le$ capacità sul taglio)

Sugli archi sono indicati in ordine **flusso** e **capacità superiore**

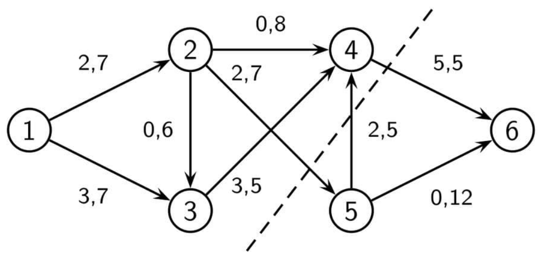


Il taglio (N_s, N_t), dove $N_s = \{1, 2, 3, 4\}$ e $N_t = \{5, 6\}$, è ammissibile.
archi diretti: $A^+ = \{(2, 5), (4, 6)\}$
archi inversi: $A^- = \{(5, 4)\}$
capacità del taglio: $u(N_s, N_t) = u_{25} + u_{46} = 7 + 5 = 12$
flusso sul taglio: $x(N_s, N_t) = x_{25} + x_{46} - x_{54} = 2 + 5 - 2 = 5$.

I **vincoli di bilancio** sono rispettati:

- Il flusso totale che esce dal nodo 1 è 2 + 3 = 5 che è lo stesso flusso che entra nel nodo 6 (5+0). In questo caso v=5
- I nodi 2, 3, 4, 5 sono tutti a bilancio nullo perchè per esempio dal nodo 2 entra un flusso pari a 2 e ne esce uno pari a 0 + 2 + 0 = 2.

Anche il **vincolo di capacità** è rispettato perchè flussi $\leq$ capacità superiori.

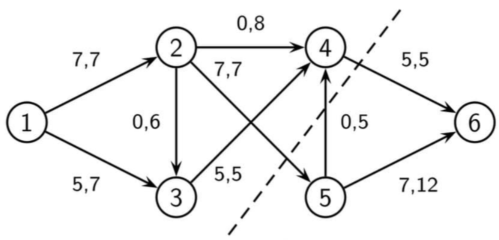


valore del flusso: $v = x_{12} + x_{13} = 2 + 3 = 5$
flusso sul taglio: $x(N_s, N_t) = x_{25} + x_{46} - x_{54} = 2 + 5 - 2 = 5$.
capacità del taglio: $u(N_s, N_t) = u_{25} + u_{46} = 7 + 5 = 12$

Su qualunque taglio ammissibile, il flusso sul taglio è uguale al valore del flusso.

4.3 Condizioni di ottimalità

Se esistono un flusso ammissibile x ed un taglio ammissibile (N_s, N_t) tali che $x(N_s, N_t) = u(N_s, N_t)$ allora x è un flusso di valore masssimo e (N_s, N_t) è un taglio di capacità minima. **(Teorema max flow - min cut)**



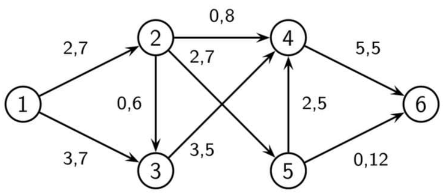
valore del flusso: $v = x_{12} + x_{13} = 7 + 5 = 12$
flusso sul taglio: $x(N_s, N_t) = x_{25} + x_{46} - x_{54} = 7 + 5 - 0 = 12$
capacità del taglio: $u(N_s, N_t) = u_{25} + u_{46} = 7 + 5 = 12$
Quindi il flusso è di valore massimo ed il taglio è di capacità minima.

Siccome il flusso su questo taglio è uguale alla capacità del taglio, significa che questo flusso di valore 12 è ottimo. Ce ne possono essere anche altri, ma non esiste un altro di valore > 12. Questo taglio che ha capacità 12 è anche di capacità minima, cioè non si può trovarne un altro taglio ammissibile che ha capacità < 12.

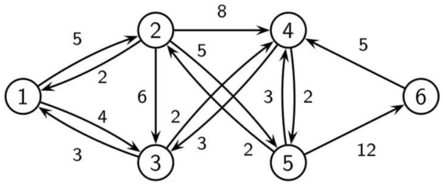
4.4 Grafo residuo

Dato un flusso ammissibile x, il grafo residuo ad x è il grafo avente gli stessi nodi del grafo originale, mentre gli archi e le loro **capacità residue $\mathbf{r_{ij}}$** sono definiti come:

- se $(i, j) \in A$ con $x_{ij} < u_{ij}$ (arco non saturo) allora $(i,j) \in A(x)$ con $r_{ij} = u_{ij} - x_{ij}$ (significa che lo possiamo usare ancora nel grafo residuo per spedire una quantità ulteriore di flusso)
- se $(i, j) \in A$ con $x_{ij} > 0$ (arco non vuoto) allora $(j,i) \in A(x)$ con $r_{ji} = x_{ij}$



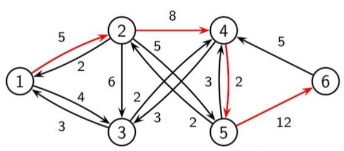
Il grafo residuo associato al flusso indicato sugli archi è il seguente (sugli archi sono indicate le capacità residue):



Sull'arco x_{24} il flusso non è > 0, quindi nel grafo residuo non c'è il cammino inverso. Sull'arco x_{46} l'arco è saturo, quindi nel grafo residuo non c'è, cioè non posso spedire flusso, posso solo spedirlo in verso opposto.

4.4.1 Cammino aumentante

Dato un flusso ammissibile x, un cammino aumentante rispetto ad x è un cammino orientato da s a t nel grafo residuo.



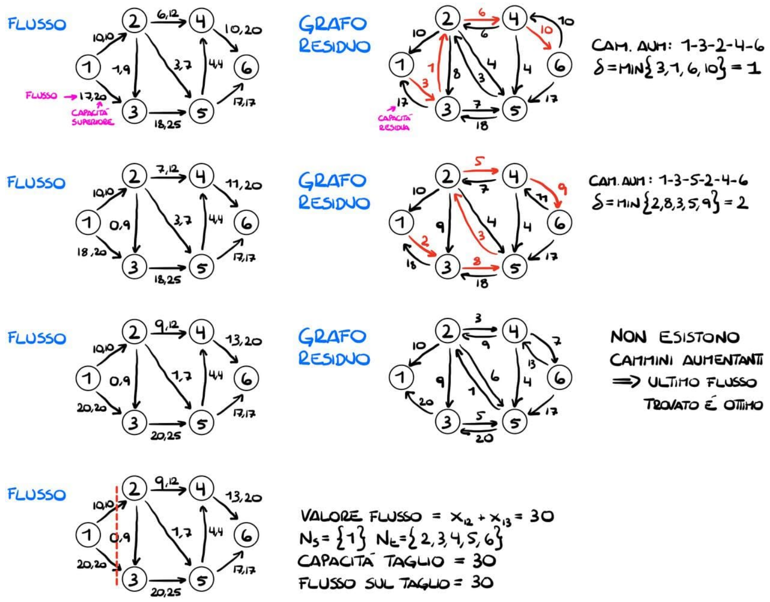
1-2-5-6 è un cammino aumentante (ed è un cammino orientato nel grafo iniziale)
1-2-4-5-6 è un cammino aumentante (ma non è un cammino orientato nel grafo iniziale)

Sia x un flusso ammissibile, allora x è un flusso di valore massimo se e solo se non esiste un cammino aumentante rispetto ad x.

4.5 Algoritmo di Ford-Fulkerson

1. Poni x=0 (flusso nullo su tutti gli archi) e quindi il grafo residuo coincide con il grafo originale.
2. Se esiste un cammino aumentante C rispetto ad x allora:
   (a) calcola $\delta = \min\{$ della capacità residua del cammino aumentante $\}$
   (b) se $(i,j) \in C$ allora $x_{ij} = x_{ij} + \delta$
   se $(j,i) \in C$ allora $x_{ij} = x_{ij} - \delta$
   se non esiste il cammino aumentante stop
3. aggiorna il grafo residuo e torna al passo 2

Si consideri il problema del flusso massimo dal nodo 1 al nodo 6:



Se non viene data una regola precisa per la ricerca del cammino aumentante, l'algoritmo può essere molto lento

4.5.1 Variante di Edmonds-Karp

Per rendere polinomiale, l'algoritmo di Ford-Fulkerson si specifica come scegliere il cammino aumentante ad ogni iterazione, cioè si sceglie il cammino aumentante con il minor numero di archi.

Chapter 5

Problema del flusso di costo minimo

5.1 Definizione

Dato un grafo orientato in cui sono definiti:

- un costo c_{ij} per spedire una unità di flusso per ogni arco
- una capacità superiore u_{ij} per ogni arco
- un bilancio b_i per ogni nodo, dove i è detto *sorgente* se $b_i < 0$, o *pozzo* se $b_i > 0$

trovare un flusso sugli archi che rispetti i bilanci dei nodi, le capacità degli archi e che sia di costo minimo.

Variabili decisionali: per ogni arco $(i, j) \in A$ definiamo x_{ij} = flusso sull'arco (i, j)

Modello di ottimizzazione:

$$\begin{cases} \min \sum_{(i,j)\in A} c_{ij}x_{ij} \\ \sum_{(i,k)\in A} x_{ik} - \sum_{(k,j)\in A} x_{kj} = b_k \quad \forall\, k \in N \quad \text{(vincoli di bilancio)} \\ 0 \leq x_{ij} \leq u_{ij} \quad \forall\, (i,j) \in A \quad \text{(vincoli di capacità)} \end{cases}$$

Il **problema dei cammini minimi** è un particolare problema di flusso minimo, dove le capacità superiori $u_{ij} = \infty$ per ogni arco. Il bilancio del nodo origine è -(n-1), invece tutti gli altri nodi sono 1.

Il **problema del flusso massimo** equivale ad un problema di flusso di costo minimo, dove per ogni nodo il bilancio è 0, per ogni arco il costo è 0, e aggiungere un arco che va dal nodo destinazione al nodo origine con costo = -1 e capacita superiore = ∞. Questo perchè ogni volta che si passa dall'arco destinazione all'arco origine abbassiamo il costo, quindi è conveniente passare per questo arco, ma per farlo dobbiamo prima fare arrivare il flusso fino al nodo destinazione.

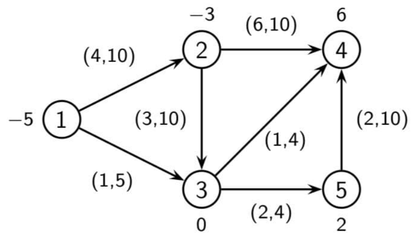
5.2 Condizioni di ottimalità

Ad ogni flusso x associamo il grafo residuo in cui l'insieme degli archi è definito come:

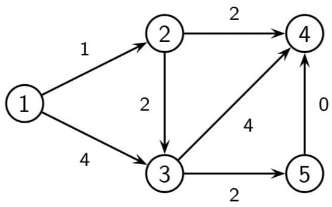
- se $x_{ij} < u_{ij}$ allora capacità residua $r_{ij} = u_{ij} - x_{ij}$ e costo $c'_{ij} = c_{ij}$
- se $x_{ij} > 0$ allora capacità residua $r_{ji} = x_{ij}$ e costo $c'_{ji} = -c_{ij}$

Sia x un flusso ammissibile, x è ottimo se e solo se non esistono cicli orientati di costo negativo

Sui primo grafo sono indicati i **costi** e **capacità residue**. Sul secondo grafo i **flussi**.



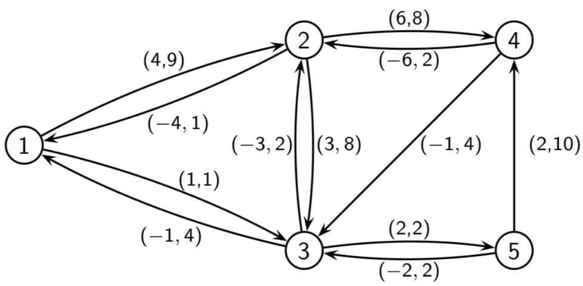
Dire se il seguente flusso ammissibile x è ottimo oppure no:



Il flusso è ammissibile perchè le capacità superiori sono rispettate e i bilanci sono rispettati (flusso entrante - flusso uscente = bilancio):

- dal nodo 1 escono 5 unità
- dal nodo 2 escono 3 unità (1-4)
- dal nodo 3 escono 0 unità (6-6)
- dal nodo 4 entrano 6 unità (4+2+0)
- dal nodo 5 entrano 2 unità

Grafo residuo:



Poiché il ciclo 1-3-2-1 ha costo −6, il flusso x non è ottimo.

5.3 Pseudoflussi

Un pseudoflusso x è un flusso che rispetta i vincoli di capacità degli archi ma non necessariamente i vincoli di bilancio dei nodi. Se x è uno pseudoflusso, allora **e_x(i)= differenza tra flusso entrante - flusso uscente - il bilancio** è chiamato lo sbilanciamento del nodo i rispetto ad x.

- E_x = insieme dei nodi con eccesso di flusso ($e_x(i) > 0$)
- D_x = insieme dei nodi con difetto di flusso ($e_x(i) < 0$)
- g(x) è la sommatoria dell'insieme dei nodi con eccesso di flusso e indica lo sbilanciamento complessivo

Un pseudoflusso x è ammissibile se e solo se $E_x = D_x = 0$ e $g(x) = 0$. Minimale significa che ha costo minimo tra tutti gli pseudoflussi aventi lo stesso vettore di sbilanciamenti. **Uno pseudoflusso x è minimale se e solo se non esistono cicli aumentanti rispetto a x di costo negativo**.

5.4 Algoritmi dei cammini minimi successivi

L'algoritmo dei cammini minimi parte con uno pseudoflusso minimale. Ad ogni iterazione l'algoritmo mantiene uno pseudoflusso x minimale e cerca nel grafo residuo un cammino di costo minimo da un nodo di E_x ad un nodo di D_x in modo da diminuire lo sbilanciamento complessivo al minor costo possibile.

0. Per ogni $(i, j) \in A$ poni $x_{ij} = \begin{cases} 0 & \text{se } c_{ij} \geq 0, \\ u_{ij} & \text{se } c_{ij} < 0. \end{cases}$

1. Se $E_x = \emptyset$ allora stop (x è ottimo)

2. Aggiungi a $G(x)$ un nodo r e gli archi (r, i) per ogni $i \in E_x$ con costo $c_{ri} = 0$ e capacità residua $r_{ri} = +\infty$.
Trova in $G(x)$ un albero dei cammini minimi T di radice r.

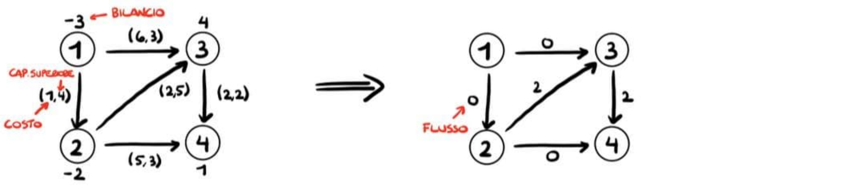
3. Trova $t = \arg\min_{i \in D_x} \pi_i$ (nodo di D_x con potenziale minimo).
Se $\pi_t = +\infty$ allora stop (non esistono flussi ammissibili)

4. Indica con P il cammino da r a t contenuto nell'albero T (P passa per un nodo $s \in E_x$).
Calcola $\delta = \min\{e_x(s), -e_x(t), \min\{r_{ij} : (i, j) \in P\}\}$ (max quantità da spedire lungo P).

5. Aggiorna x spedendo δ unità di flusso lungo il cammino P e torna al passo 1.

Alla prima iterazione tutti gli archi di costo positivo, si inseriscono con flusso nullo; tutti gli archi di costo negativo si inseriscono con flusso = capacità superiore.

Si risolva il problema utilizzando l'algoritmo dei cammini minimi successivi *a partire* dallo pseudoflusso minimale riportato sul grafo a destra:

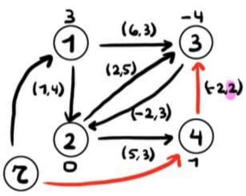


DIFFERENZA TRA FLUSSO ENTRANTE − FLUSSO USCENTE − IL BILANCIO

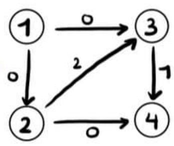
$x(1) = 0 - 0 + 3 = 3$ $x(3) = 2 - 2 - 4 = -4$

$x(2) = 0 - 2 + 2 = 0$ $x(4) = 2 - 0 - 1 = 1$

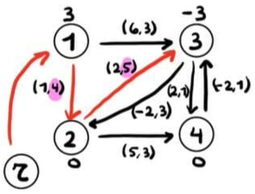
GRAFO RESIDUO
$s = 1 \quad t = 4$
$\text{MIN}\{1, 4, 2\} = 1$



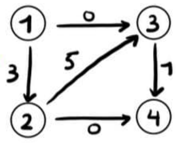
NUOVO PSEUDOFLUSSO



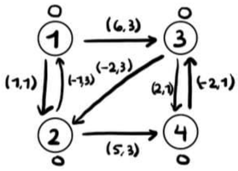
GRAFO RESIDUO
$s = 1 \quad t = 3$
$\text{MIN}\{3, 3, 4\} = 3$
$\text{MIN}\{4, 5\} = 4$



NUOVO PSEUDOFLUSSO



LO SBILANCIAMENTO COMPLESSIVO RISULTA NULLO, È UN FLUSSO DI COSTO MINIMO. COME ULTERIORE VERIFICA DELLA SUA OTTIMALITÀ, SI PUÒ OSSERVARE CHE NON ESISTONO CICLI DI COSTO NEGATIVO NEL GRAFO RESIDUO ASSOCIATO



Chapter 6

Programmazione lineare

6.1 Definizione

Un problema di PL consiste nel trovare il massimo o il minimo di una funzione lineare di n variabili reali soggette a vincoli lineari di uguaglianza o disuguaglianza, cioè

$$\begin{cases} \text{max (o min) } c^T x \\ A_1 x \leq b_1 \\ A_2 x \geq b_2 \\ A_3 x = b_3 \end{cases}$$

Ogni problema di PL può essere scritto in modo equivalente in **forma canonica**:

$$\begin{cases} \text{max } c^T x \\ Ax \leq b \end{cases}$$

$$\begin{cases} \text{MIN } 2x_1 + 5x_2 \\ 6x_1 + 9x_2 = 17 \\ x_1 \geq 0 \\ x_1 + 3x_2 \geq 1 \end{cases} \xrightarrow{\text{IN FORMA CANONICA}} \begin{cases} -\text{MAX } (-2x_1 - 5x_2) \\ 6x_1 + 9x_2 \leq 17 \\ -6x_1 - 9x_2 \geq -17 \\ -x_1 \leq 0 \\ -x_1 - 3x_2 \leq -1 \end{cases}$$

6.2 Geometria PL

La geometria della PL serve per definire i poliedri:

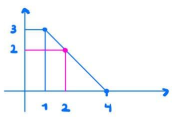
- **Combinazione convessa**: Un vettore è detto combinazione convessa se i coefficienti sono tutti numeri compresi tra 0 e 1, e la loro somma è uguale a 1. Dal punto di vista geometrico una combinazione convessa di 2 punti è un punto che sta sul segmento che congiunge i due punti. (Combinazione lineare)

  • (2,2) É COMBINAZIONE CONVESSA DI (4,0) E (1,3)

  $$\frac{1}{3}(4,0) + \frac{2}{3}(1,3) = (2,2)$$

  = 1

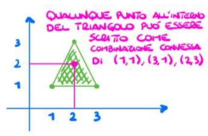
  ∈ [0,1]

- **Involucro convesso**: Un involucro convesso di un insieme K, denotato con conv(K), è l'insieme di tutte le combinazioni convesse di punti di K. (Span)

  • (2,2) É COMBINAZIONE CONVESSA DI (1,1), (3,1) E (2,3)

  $$\frac{1}{4}(1,1) + \frac{1}{4}(3,1) + \frac{1}{2}(2,3) = (2,2)$$

- **Insieme convesso**: Un insieme K è detto convesso se per ogni x, y ∈ K, il vettore αx + (1-α)y ∈ K per ogni $\alpha \in [0,1]$. (Sottospazio vettoriale)

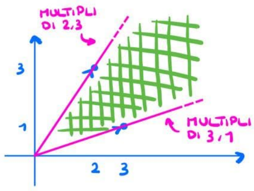
- **Combinazione conica**: Un vettore è detto combinazione conica se i coefficienti sono tutti numeri ≥ 0

(4,4) É COMBINAZIONE CONICA DI (2,3) e (3,1)

$$\frac{8}{7}(2,3) + \frac{4}{7}(3,1) = (4,4)$$

NON SONO < 0

- **Involucro conico**: L'involucro conico di un insieme K, denotato con cono(K), è l'insieme di tutte le combinazioni coniche di punti di K



TUTTI I PUNTI SONO COMBINAZIONI CONICHE DEI PUNTI 3,1 e 2,3 $\Rightarrow$ É ILLIMITATO

- **Cono**: Un insieme K è detto cono tale che se contiene un punto x, allora contiene anche tutta la semiretta uscente dall'origine e passante per x, ovvero contiene tutti i vettori del tipo αx dove $\alpha \geq 0$

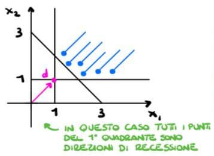
6.3 Poliedri

Un poliedro P è la regione ammissibile di ogni problema di PL. Dal punto di vista *geometrico* è l'intersezione di un numero finito di semispazi chiusi. Dal punto di vista *algebrico* è l'insieme delle soluzioni di un sistema di disequazioni lineari Ax $\leq$ b. Esistono poliedri limitati (involucro convesso) e illimitati

6.3.1 Direzioni di recessione

Le direzioni di recessione sono utili particolarmente per i poliedri illimitati, ed è un vettore d tale che i vettori del tipo x + αd $\in$ P per ogni x $\in$ P e $\alpha > 0$. L'insieme di tutte le direzioni di recessione di un poliedro P è un cono convesso ed è denominato rec(P).

$P = \; x_1 \geq 1, \quad x_2 \geq 1, \quad x_1 + x_2 \geq 3$

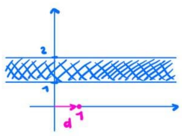


d = (1,1) É UNA DIREZIONE DI RECESSIONE DEL POLIEDRO PERCHÉ QUALUNQUE PUNTO PRENDIAMO NEL POLIEDRO LA SEMIRETTA APPARTIENE AL POLIEDRO L'IMPORTANTE É LA "DIREZIONE" DI d

6.3.2 Direzioni di linearità

Una direzione di linealità di un poliedro P è un vettore d tale che x + αd $\in$ P per ogni x $\in$ P e $\alpha \in \mathbb{R}$, quindi si tiene conto di α positivi e negativi, e la direzione opposta di d. L'insieme di tutte le direzioni di linealità di un poliedro P è un sottospazio vettoriale ed è denotato con lineal(P)

$P = \; 1 \leq x_2 \leq 2$

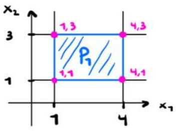


d = (1,0) É UNA DIREZIONE DI LINEARITÁ DEL POLIEDRO.

6.3.3 Vertici

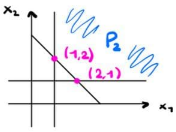
Sono gli elementi che **non** possono essere scritti come combinazione convessa degli altri elementi del poliedro. x è un vertice se non esistono 2 punti y, z diversi da x tali per cui x è una combinazione convessa di y e z. Un poliedro ha almeno un vertice se e solo se lineal(P) = 0.

$P_1 = \; 1 \le x_1 \le 4 \,, \; 1 \le x_2 \le 3$



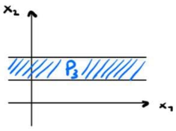
QUESTI 4 PUNTI NON POSSONO ESSERE COME SCRITTI COME COMB. CONVESSA DI ALTRI PUNTI DEL POLIEDRO P_1

$P_2 = \; x_1 \ge 1 \,, \; x_2 \ge 1 \,, \; x_1 + x_2 \ge 3$



P_2 NON É LIMITATO MA HA DEI VERTICI

$P_3 = \; 1 \le x_2 \le 2$



NON HA VERTICI

6.3.4 Teorema decomposizione dei poliedri

Se P è un poliedro, allora esistono:

- un insieme finito $\{v^1, ..., v^m\}$ di punti di P
- un insieme finito $\{d^1, ..., d^q\}$ di direzioni di recessione di P

tali che P=conv$\{v^1, ..., v^m\}$ + cono$\{d^1, ..., d^q\}$.
Se lineal(P)=0 allora $v^1, ..., v^m$ sono i vertici di P; se P è un poliedro limitato allora P=conv(V).
La decomposizione dei poliedri P precedenti:

NON HA CONO DI RECESSIONE PERCHÉ NON HA DIREZIONI DI RECESSIONE

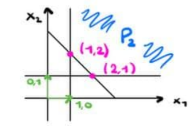
$P_1 = \text{CONV}\,\{(1,1),(1,3),(4,1),(4,3)\}$

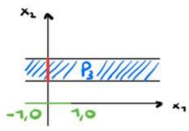
GENERA TUTTO IL 1° QUADRANTE

$P_2 = \text{CONV}\,\{\underline{(1,2),(2,1)}\} + \text{CONO}\,\{\underline{(1,0),(0,1)}\}$

$P_3 = \text{CONV}\,\{\underline{(0,1),(0,2)}\} + \text{CONO}\,\{(1,0),(-1,0)\}$

NON SONO VERTICI MA SONO IMPORTANTI PER COSTRUIRE QUESTO INSIEME





6.4 Teorema fondamentale della PL

Sia P=conv$\{v^1, ..., v^m\}$ + cono$\{d^1, ..., d^q\}$ allora

- Il valore ottimo di P è finito se $c^T d^j \le 0$ per ogni j=1,...,q, cioè nessuna direzione di recessione di P è una direzione di crescita
- Se il valore ottimo di P è finito, allora uno dei suoi vertice è una soluzione ottima

Esempio. Consideriamo il problema

$$\begin{cases} \max \; 2x_1 - 3x_2 \\ x_1 \ge 1 \\ x_2 \ge 1 \\ x_1 + x_2 \ge 3 \end{cases}$$

Sappiamo che $P = \text{conv}\{(1,2),(2,1)\} + \text{cono}\{(1,0),(0,1)\}$.
Il valore ottimo è $+\infty$ poiché la direzione di recessione $d = (1,0)$ è una direzione di crescita, cioè $(2,-3)^T(1,0) = 2 > 0$.

Esempio. Consideriamo il problema

$$\begin{cases} \max \; -2x_1 - 3x_2 \\ x_1 \ge 1 \\ x_2 \ge 1 \\ x_1 + x_2 \ge 3 \end{cases}$$

Il valore ottimo è finito perché nessuna delle due direzioni di recessione $(1,0)$ e $(0,1)$ è di crescita, cioè $(-2,-3)^T(1,0) = -2$ e $(-2,-3)^T(0,1) = -3$.
La soluzione ottima è il vertice $(2,1)$.

Un problema di PL può avere una, nessuna o infinite soluzioni:

- Se esiste una direzione di recessione di P che è anche una direzione di crescita, allora il valore ottimo è $+\infty$ e non esiste **nessuna** soluzione ottima;
- Se nessuna direzione di recessione di P è una direzione di crescita allora il valore ottimo è finito ed esiste almeno una soluzione ottima, e ci sono due casi:
  - la soluzione ottima è **unica**;
  - se esistono 2 soluzioni ottime x e x', allora tutti i punti che stanno sul segmento di estremi x e x' sono ottime e quindi vi sono **infinite** soluzione ottime.

6.5 Problema Duale

Il problema di PL definito come:

$$\begin{cases} \max \mathrm{c}^{\mathrm{T}}\mathrm{x} \\ Ax \leq b \end{cases}$$

d'ora in poi sarà chiamato **problema primale** P.

Un **problema duale** D è un altro problema di PL definito come:

$$\begin{cases} \min \mathrm{y}^{\mathrm{T}}\mathrm{b} \\ y^{\mathrm{T}}A = c^{\mathrm{T}}, y \geq 0 \end{cases}$$

	Primale	Duale
Obiettivo	max	min
Variabili	n	m
Vincoli	m	n

PROBLEMA PRIMALE

$$\begin{cases} \text{MAX } 4x_1 + 5x_2 \\ x_1 \leq 1 \\ x_2 \leq 2 \\ x_1 + x_2 \leq 3 \end{cases} \quad A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \quad c = \begin{pmatrix} 4 \\ 5 \end{pmatrix}$$

IL PROBLEMA DUALE É:

MIN $y_1 + 2y_2 + 3y_3$

$$(y_1, y_2, y_3)\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} = (4, 5) \implies \begin{cases} \text{MIN } y_1 + 2y_2 + 3y_3 \\ y_1 + y_3 = 4 \\ y_2 + y_3 = 5 \\ y \geq 0 \end{cases}$$

CIOÉ $y_1 + y_3 = 4$
$y_2 + y_3 = 5$

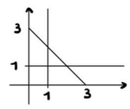
6.5.1 Lemma di Farkas

I sistemi:

$$\begin{cases} \mathrm{c}^{\mathrm{T}}\mathrm{x} > 0 \\ Ax \leq 0 \end{cases} \qquad \begin{cases} \mathrm{y}^{\mathrm{T}}\mathrm{A} = \mathrm{c}^{\mathrm{T}} \\ y \geq 0 \end{cases}$$

sono in alternativa, cioè uno ammette soluzioni se e solo se nell'altro non ha soluzioni. Nel problema primale esiste una direzione di recessione che è anche di crescita se e solo se la regione ammissibile del problema duale è vuota

$$\begin{cases} \text{MAX } 2x_1 - 3x_2 \\ -x_1 \leq -1 \\ -x_2 \leq -1 \\ -x_1 - x_2 \leq -3 \end{cases}$$



SAPPIAMO CHE ESISTE UNA DIREZIONE DI RECESSIONE CHE É DI CRESCITA

$d\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \quad 2 \cdot 1 + (-3) \cdot 0 = 2$

IL VALORE OTTIMO SARÁ $+\infty$, NON CI SARANNO SOLUZIONI OTTIME

PER IL LEMMA DI FARKAS IL POLIEDRO DUALE DEVE ESSERE VUOTO.
SCRIVIAMO IL DUALE

$$A = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ -1 & -1 \end{pmatrix} \quad b = \begin{pmatrix} -1 \\ -1 \\ -3 \end{pmatrix} \quad c = \begin{pmatrix} 2 \\ -3 \end{pmatrix}$$

$$\begin{cases} \text{MIN } -y_1 - y_2 - 3y_3 \\ -y_1 - y_3 = 2 \\ -y_2 - y_3 = -3 \\ y \geq 0 \end{cases}$$

IL PRIMO VINCOLO IMPLICA CHE $y_3 = -y_1 - 2$
CHE NON É COMPATIBILE CON LA CONDIZIONE $y \geq 0$,
QUINDI NON ESISTONO SOLUZIONI AMMISSIBILI
DEL PROBLEMA DUALE

6.5.2 Teorema di dualità debole e forte

Supponiamo che nel problema primale esistono delle soluzioni ammissibili allora

- **Dualità debole**: $\mathrm{v}(P) \leq \mathrm{v}(D)$
- **Dualità forte**:
  - se il poliedro duale D=0 allora il valore ottimo del primale $P = +\infty$
  - se il poliedro duale D $\neq 0$, allora $\mathrm{v}(P) = \mathrm{v}(D)$

$$\begin{cases} \text{MIN } y_1 + y_2 + y_3 + y_4 \\ y_1 + y_2 - y_3 - y_4 = 2 \\ y_1 - y_2 + y_3 - y_4 = 1 \\ y \geq 0 \end{cases}$$

QUAL É IL SUO VALORE OTTIMO?

POSSIAMO USARE IL TEOREMA DELLA DUALITÁ FORTE, CIOÉ USARE IL PROBLEMA DUALE

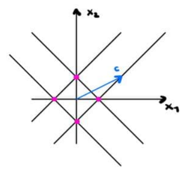
Y É UN VETTORE CON 4 COMPONENTI, QUINDI $b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$

HO 2 EQUAZIONI QUINDI $c = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$

E A DOVRÁ ESSERE UNA MATRICE CON 2 COLONNE E 4 RIGHE $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{pmatrix}$

QUINDI IL PROBLEMA DUALE É
$$\begin{cases} \text{MAX } 2x_1 + x_2 \\ x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \\ -x_1 + x_2 \leq 1 \\ -x_1 - x_2 \leq 1 \end{cases}$$

I 2 PROBLEMI HANNO LO STESSO VALORE OTTIMO MA IL VANTAGGIO É CHE ORA HO UN PROBLEMA IN 2 VARIABILI. QUINDI É FACILE DISEGNARLO GRAFICAMENTE
$\Longrightarrow$ LA SOLUZIONE OTTIMA É IL VERTICE (1,0)
E IL VALORE OTTIMO É 2



6.6 Teorema degli scarti complementari

Per riconoscere una soluzione ottima si usa il teorema degli scarti complementari. Supponiamo che $\overline{x}$ sia una soluzione ammissibile del primale, allora $\overline{x}$ è ottima se e solo se esiste una soluzione di $\overline{y}$ del sistema:

$$\begin{cases} \overline{y}^{\mathrm{T}}\mathrm{A} = \mathrm{c}^{\mathrm{T}} \\ \overline{y} \geq 0 \\ \overline{y}^{\mathrm{T}}(\mathrm{b} - \mathrm{A}\overline{x}) = 0 \end{cases}$$

Qualunque soluzione $\overline{y}$ di questo sistema è una soluzione ottima del duale

Esempio. Dire se $\bar{x} = (1, 1)$ è ottima per il problema

$$\begin{cases} \max\ 3x_1 + 4x_2 \\ 2x_1 + x_2 \leq 3 \\ x_1 + 2x_2 \leq 3 \\ -x_1 \leq 0 \\ -x_2 \leq 0 \end{cases}$$

$\bar{x}$ è ottima se e solo se esiste una soluzione del sistema

$$\begin{cases} 2y_1 + y_2 - y_3 = 3 \\ y_1 + 2y_2 - y_4 = 4 \\ y \geq 0 \\ y^T(0, 0, 1, 1) = 0 \end{cases}$$

che equivale a

$$\begin{cases} y_3 = y_4 = 0 \\ 2y_1 + y_2 = 3 \\ y_1 + 2y_2 = 4 \\ y_1, y_2 \geq 0 \end{cases}$$

Poiché $\bar{y} = (2/3, 5/3, 0, 0)$ è una soluzione del sistema, $\bar{x}$ è ottima.

Esempio. Dire se $\bar{x} = (0, 0)$ è ottima per il problema

$$\begin{cases} \max\ 3x_1 + 4x_2 \\ 2x_1 + x_2 \leq 3 \\ x_1 + 2x_2 \leq 3 \\ -x_1 \leq 0 \\ -x_2 \leq 0 \end{cases}$$

$\bar{x}$ è ottima se e solo se esiste una soluzione del sistema

$$\begin{cases} 2y_1 + y_2 - y_3 = 3 \\ y_1 + 2y_2 - y_4 = 4 \\ y \geq 0 \\ y^T(3, 3, 0, 0) = 0 \end{cases}$$

che equivale a

$$\begin{cases} y_1 = y_2 = 0 \\ y_3 = -3 \\ y_4 = -4 \\ y_3, y_4 \geq 0 \end{cases}$$

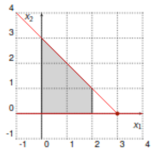
che è impossibile, quindi $\bar{x}$ non è ottima.

6.7 Basi e vertici

Una base è un insieme B di n indici di riga tali che la sottomatrice A_B sia invertibile, cioè $\det(A_B \neq 0)$. N è l'insieme degli indici non in base. Data una base B, il vettore $\overline{x} = A^{-1}{}_B b_B$ è chiamato **soluzione di base primale**.
$\overline{x}$ è **ammissibile** se $A_N \overline{x} \leq b_N$ (componente per componente). $\overline{x}$ è **degenere** se esiste $i \in N$ tale che $A_i \overline{x} = b_i$. Una soluzione degenere dal punto di vista geometrico significa che i vincoli non in base passano per il punto della soluzione ammissibile. **$\overline{x}$ è un vertice di P se e solo se $\overline{x}$ è una soluzione di base primale ammissibile.**

Esempio. Consideriamo

$$\begin{cases} \max\ 2x_1 + x_2 \\ x_1 \leq 2 \\ x_1 + x_2 \leq 3 \\ -x_1 \leq 0 \\ -x_2 \leq 0 \end{cases} \qquad A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \qquad b = \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \end{pmatrix}$$



$B = \{1,2\}$ è una base perché $A_B = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ è invertibile: $\det(A_B) = 1$.

La relativa soluzione di base primale è $\bar{x} = A_B^{-1} b_B = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$.

$\bar{x}$ è ammissibile perché $A_N \bar{x} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \end{pmatrix} = b_N$
e non degenere perché $A_i \bar{x} \neq b_i$ per ogni $i \in N$.

$B = \{1,3\}$ non è una base perché $A_B = \begin{pmatrix} 1 & 0 \\ -1 & 0 \end{pmatrix}$ non è invertibile: $\det(A_B) = 0$.

$B = \{2,4\}$ è una base e la relativa soluzione di base primale non è ammissibile:
$A_B = \begin{pmatrix} 1 & 1 \\ 0 & -1 \end{pmatrix}$, $\bar{x} = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$, $A_N \bar{x} = \begin{pmatrix} 3 \\ -3 \end{pmatrix} \not\leq \begin{pmatrix} 2 \\ 0 \end{pmatrix} = b_N$ e non degenere.

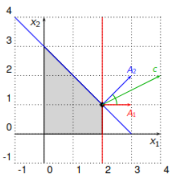
Data una base B, il vettore $\overline{y}^T{}_B = c^T A^{-1}{}_B$ e $\overline{y}_N = 0$ è chiamato **soluzione di base duale**.
$\overline{y}$ è **ammissibile** se $\overline{y}_B \geq 0$. $\overline{y}$ è **degenere** se esiste $i \in B$ tale che uno dei suoi componenti è 0 ($\overline{y}_i = 0$).

6.7.1 Condizioni di ottimalità

Sia $\overline{x}$ una soluzione di base primale ammissibile relativa alla base B, se la soluzione di base duale $\overline{y}$ relativa alla stessa base è **ammissibile**, allora $\overline{x}$ è ottima per il problema primale e $\overline{y}$ è ottima per il duale.

Esempio. Consideriamo il problema

$$\begin{cases} \max\ 2x_1 + x_2 \\ x_1 \leq 2 \\ x_1 + x_2 \leq 3 \\ -x_1 \leq 0 \\ -x_2 \leq 0 \end{cases} \qquad A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \qquad b = \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \end{pmatrix} \qquad c = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

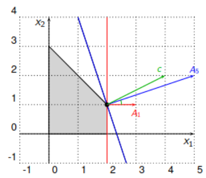


$\bar{x} = (2,1)$ è una soluzione di base primale relativa alla base $B = \{1,2\}$.
La soluzione di base duale relativa a B è

$$\bar{y} = \begin{pmatrix} \bar{y}_B \\ \bar{y}_N \end{pmatrix} \quad \text{dove} \quad \bar{y}_B^T = c^T A_B^{-1} = (2,1) \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} = (1,1), \quad \bar{y}_N = 0.$$

$\bar{y}$ è ammissibile perché $\bar{y}_B \geq 0$, ossia $c \in cono(A_1, A_2)$, quindi $\bar{x}$ è ottima.

$$\begin{cases} \max\ 2x_1 + x_2 \\ x_1 \leq 2 \\ x_1 + x_2 \leq 3 \\ -x_1 \leq 0 \\ -x_2 \leq 0 \\ 3x_1 + x_2 \leq 7 \end{cases} \qquad A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \\ 3 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \\ 7 \end{pmatrix} \qquad c = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$



$\bar{x} = (2,1)$ è ottima ed è una soluzione di base primale relativa alla base $B = \{1,5\}$.
La soluzione di base duale relativa a B è

$$\bar{y} = \begin{pmatrix} \bar{y}_B \\ \bar{y}_N \end{pmatrix} \quad \text{dove} \quad \bar{y}_B^T = c^T A_B^{-1} = (2,1) \begin{pmatrix} 1 & 0 \\ -3 & 1 \end{pmatrix} = (-1,1), \quad \bar{y}_N = 0.$$

$\bar{y}$ non è ammissibile perché $\bar{y}_1 < 0$, ossia $c \notin cono(A_1, A_5)$, ma $\bar{x}$ è ottima.

6.8 Algoritmo del simplesso primale

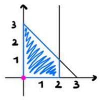
L'algoritmo del simplesso permette di risolvere qualsiasi problema di PL. L'algoritmo si muove attraverso i vertici del poliedro, e trova la soluzione ottima.

1. Trova una base B tale che la relativa soluzione di base primale $\bar{x} = A_B^{-1} b_B$ sia ammissibile.
2. Calcola la soluzione di base duale
$$\bar{y} = \begin{pmatrix} \bar{y}_B \\ \bar{y}_N \end{pmatrix}, \quad \text{dove} \quad \bar{y}_B^T = c^T A_B^{-1}, \quad \bar{y}_N = 0.$$
3. **Se** $\bar{y}_B \geq 0$ **allora** STOP, $\bar{x}$ è ottima.
altrimenti trova l'indice uscente
$$h = \min\{i \in B : \ \bar{y}_i < 0\} \qquad \text{(regola anticiclo di Bland),}$$
poni $W = -A_B^{-1}$ e denota W^h la h–esima colonna di W.
4. **Se** $A_i W^h \leq 0$ per ogni $i \in N$ **allora** STOP, valore ottimo del primale $= +\infty$.
altrimenti calcola $\vartheta := \min \left\{ \dfrac{b_i - A_i \bar{x}}{A_i W^h} : \quad i \in N, \ A_i W^h > 0 \right\}$,
trova l'indice entrante
$$k = \min \left\{ i \in N : \ A_i W^h > 0, \ \frac{b_i - A_i \bar{x}}{A_i W^h} = \vartheta \right\} \qquad \text{(regola anticiclo di Bland),}$$
aggiorna la base $B = B \setminus \{h\} \cup \{k\}$, calcola $\bar{x} = A_B^{-1} b_B$ e torna al passo 2.

$$\begin{cases} \max\ 2x_1 + x_2 \\ x_1 \le 2 \\ x_1 + x_2 \le 3 \\ -x_1 \le 0 \\ -x_2 \le 0 \end{cases} \qquad A = \begin{vmatrix} 1 & 0 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{vmatrix} \qquad b = \begin{vmatrix} 2 \\ 3 \\ 0 \\ 0 \end{vmatrix} \qquad c = \begin{vmatrix} 2 \\ 1 \end{vmatrix}$$

#1 $B = \{3,4\}$ $\quad A_B = \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$ $\quad \det A_B = 1 - 0 = 1 \Rightarrow$ É INVERTIBILE

SOLUZIONE DI BASE PRIMALE $\quad \bar{x} = \underbrace{\begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}}_{A_B^{-1}} \begin{vmatrix} 0 \\ 0 \end{vmatrix} = \begin{vmatrix} 0 \\ 0 \end{vmatrix}$



$\begin{vmatrix} 0 \\ 0 \end{vmatrix} \le \begin{vmatrix} 2 \\ 3 \end{vmatrix} \Rightarrow$ É AMMISSIBILE

SOLUZIONE DI BASE DUALE $\quad \bar{y} = \begin{vmatrix} 2 & 1 \end{vmatrix} \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix} = \begin{vmatrix} -2 \\ -1 \end{vmatrix} \not\ge 0$

TROVA L'INDICE USCENTE: $\quad h = \min\{3,4\} = 3$

$W = -A_B^{-1} = \begin{array}{c} \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \\ \text{INDICI: } 3 \quad 4 \end{array} \Rightarrow$ SICCOME $h = 3$ DOBBIAMO SCEGLIERE LA COLONNA $\binom{1}{0}$

$\binom{1}{0}$ É LA DIREZIONE DI SPOSTAMENTO, ORA SI DEVE CALCOLARE DI QUANTO CI POSSIAMO MUOVERE IN QUESTA DIREZIONE PER NON USCIRE DAL POLIEDRO

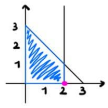
QUELLI NON IN BASE $\quad A_N W_3 = \begin{array}{c} \begin{vmatrix} 1 & 0 \\ 1 & 1 \end{vmatrix} \\ 1 \quad 2 \end{array} \begin{vmatrix} 1 \\ 0 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \end{vmatrix} \not\le 0$

$$\min = \begin{cases} \dfrac{b_1 - A_1\bar{x}}{A_1 W_3} = \dfrac{2-0}{1} \\ \dfrac{b_2 - A_2\bar{x}}{A_2 W_3} = \dfrac{3-0}{1} \end{cases} \Rightarrow \min = 2$$

LUNGO LA DIREZIONE $\binom{1}{0}$ CI POSSIAMO MUOVERE AL MAX DI UN PASSO = 2

$k = \min\{1,2\} = 1$

#2 $B = \{1,4\}$ $\quad A_B = \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix}$



$\bar{x} = \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} \begin{vmatrix} 2 \\ 0 \end{vmatrix} = \begin{vmatrix} 2 \\ 0 \end{vmatrix}$

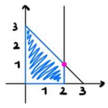
$\bar{y} = \begin{vmatrix} 2 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} = \begin{vmatrix} 2 \\ -1 \end{vmatrix} \begin{matrix} \ge 0 \\ \not\ge 0 \end{matrix} \Rightarrow h = 4$

$W = -A_B = \begin{vmatrix} -1 & 0 \\ 0 & 1 \end{vmatrix} \Rightarrow \begin{vmatrix} 0 \\ 1 \end{vmatrix}$

$A_N W_4 = \begin{vmatrix} 1 & 1 \\ -1 & 0 \end{vmatrix} \begin{vmatrix} 0 \\ 1 \end{vmatrix} = \begin{vmatrix} \underline{1} \\ 0 \end{vmatrix} \Rightarrow$ $A_N W_i > 0$ QUI NON C'É BISOGNO DI CALCOLARE LO SPOSTAMENTO, PERCHÉ É IL RAPPORTO DI UN SOLO NUMERO

$k = \min\{2,3\} = 2$

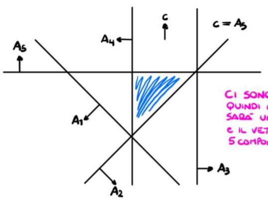
#3 $B = \{1,2\}$ $\quad A_B = \begin{vmatrix} 1 & 0 \\ 1 & 1 \end{vmatrix}$



$\bar{x} = \begin{vmatrix} 1 & 0 \\ -1 & 1 \end{vmatrix} \begin{vmatrix} 2 \\ 3 \end{vmatrix} = \begin{vmatrix} 2 \\ 1 \end{vmatrix}$

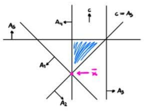
$\bar{y} = \begin{vmatrix} 2 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 \\ -1 & 1 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \end{vmatrix} \begin{matrix} \ge 0 \\ \ge 0 \end{matrix} \Rightarrow$ $\bar{x}$ É OTTIMO, $\bar{y}$ É OTTIMO

La regola di Bland serve per non trovare la stessa base già esplorata. Se si passa da una base all'altra ma la soluzione di base non cambia significa che la soluzione è *degenere.* L'algoritmo del simplesso termina dopo un numero finito di iterazioni: se il valore ottimo è $+\infty$, l'algoritmo trova una direzione di recessione che è anche di crescita; se il valore ottimo del problema è finito, l'algoritmo trova un vertice ottimo. L'algoritmo dal punto di vista geometrico lavora sui vertici del poliedro, ma dal punto di vista algebrico lavora sulle basi. Utilizzare l'algoritmo del simplesso primale dal punto di vista geometrico, cioè senza utilizzare dati ma direttamente dalla sua rappresentazione geometrica. Non ci interessa dove sia l'origine o degli assi x e y.



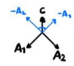
CI SONO 5 VINCOLI QUINDI LA MATRICE A SARÁ UNA 5x2 E IL VETTORE B AVRÁ 5 COMPONENTI

#1 $B = \{1,2\}$



$y_B^T = c^T A_B^{-1} \Rightarrow c^T = y_B^T A_B = y_1 A_1 + y_2 A_2$

NON INTERESSA IL VALORE DI y_1, y_2 MA SAPERE IL LORO SEGNO, PERCHÉ SE SONO ENTRAMBI ≥ 0 ALLORA $\bar{x}$ É OTTIMO



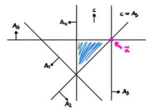
$c \notin \text{CONO}(A_1, A_2)$

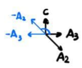
$c \in \text{CONO}(-A_1, -A_2)$ QUINDI $\bar{y}_1 < 0$ E $\bar{y}_2 < 0$

$h = \min\{1,2\} = 1 \Rightarrow$ SE ESCE L'INDICE 1 DALLA BASE, L'INDICE 2 NON ESCE QUINDI IL VINCOLO A_2 RIMANE ATTIVO E W SARÁ LUNGO LA DIREZIONE LA RETTA CHE NON ESCE

W MUOVENDOSI LUNGO QUELLA DIREZIONE, RIMANE ALL'INTERNO DEL POLIEDRO FINO AD ARRIVARE A UN VERTICE DEL POLIEDRO. I VINCOLI 3 E 5 CI IMPEDISCONO DI USCIRE DAL POLIEDRO QUINDI $k = \min\{3,5\} = 3$

#2 $B = \{2,3\}$

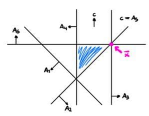


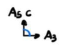


$c \in \text{CONO}(-A_2, A_3) \Rightarrow h = 2$

LA DIREZIONE DI SPOSTAMENTO SARÁ ZERO PERCHÉ DI QUALUNQUE PASSO CI MUOVIAMO, USCIAMO DAL POLIEDRO. L'ALTRO VINCOLO ATTIVO É 5 $\Rightarrow k = \min\{5\} = 5$

#3 $B = \{3,5\}$





$c \in \text{CONO}(A_3, A_5) \Rightarrow \bar{x}$ É OTTIMO

6.9 Algoritmo del simplesso duale

1. Trova una base B tale che la relativa soluzione di base duale
$$\bar{y} = \begin{pmatrix} \bar{y}_B \\ \bar{y}_N \end{pmatrix}, \quad \text{dove} \quad \bar{y}_B^T = c^T A_B^{-1}, \quad \bar{y}_N = 0,$$
sia ammissibile.
2. Calcola la soluzione di base primale $\bar{x} = A_B^{-1} b_B$.
3. **Se** $A_N \bar{x} \leq b_N$ **allora** STOP, $\bar{x}$ è ottima.
altrimenti trova l'indice entrante
$$k = \min\{i \in N : \ A_i \bar{x} > b_i\} \qquad \text{(regola anticiclo di Bland)},$$
poni $\eta_B := A_k A_B^{-1}$.
4. **Se** $\eta_B \leq 0$ **allora** STOP, la regione ammissibile del primale è vuota.
altrimenti calcola $\vartheta = \min \left\{ \frac{\bar{y}_i}{\eta_i} : \quad i \in B, \ \eta_i > 0 \right\}$,
trova l'indice uscente
$$h = \min \left\{ i \in B : \ \eta_i > 0, \ \frac{\bar{y}_i}{\eta_i} = \vartheta \right\} \qquad \text{(regola anticiclo di Bland)},$$
aggiorna la base $B = B \setminus \{h\} \cup \{k\}$, calcola $\bar{y}_B^T = c^T A_B^{-1}$ e torna al passo 2.

$$\begin{cases} \text{MAX } x_2 \\ x_2 \leq 4 \\ -x_1 + 2x_2 \leq 10 \\ -x_1 \leq 1 \\ -2x_1 + x_2 \leq 4 \\ -x_1 \leq 0 \end{cases} \quad A = \begin{vmatrix} 0 & 1 \\ -1 & 2 \\ -1 & 0 \\ -2 & 1 \\ -1 & 0 \end{vmatrix} \quad b = \begin{vmatrix} 4 \\ 10 \\ 1 \\ 4 \\ 0 \end{vmatrix} \quad c = \begin{vmatrix} 0 \\ 1 \end{vmatrix}$$

#1 $B = \{1,2\}$ $\quad A_B = \begin{vmatrix} 0 & 1 \\ -1 & 2 \end{vmatrix} \quad \det A_B = 1 \Rightarrow$ É INVERTIBILE

$$\bar{x} = \begin{vmatrix} 2 & -1 \\ 1 & 0 \end{vmatrix} \begin{vmatrix} 4 \\ 10 \end{vmatrix} = \begin{vmatrix} -2 \\ 4 \end{vmatrix}$$

$$\bar{y} = \begin{vmatrix} 0 & 1 \end{vmatrix} \begin{vmatrix} 2 & -1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} 1 & 0 \end{vmatrix} \Rightarrow \text{É AMMISSIBILE}$$

$$A_N \bar{x} = \begin{vmatrix} -1 & 0 \\ -2 & 1 \\ -1 & 0 \end{vmatrix} \begin{vmatrix} -2 \\ 4 \end{vmatrix} = \begin{vmatrix} 2 \\ 8 \\ 2 \end{vmatrix} \begin{matrix} \nleq \\ \nleq \\ \nleq \end{matrix} \begin{vmatrix} 1 \\ 4 \\ 0 \end{vmatrix}$$

TUTTI I VINCOLI SONO VIOLATI, QUINDI LA SOLUZIONE PRIMALE NON É AMMISSIBILE

INDICE ENTRANTE

$k = \text{MIN}\{\text{INDICI VIOLATI}\} = \text{MIN}\{3,4,5\} = 3$

$$\eta_B = A_3 A_B^{-1} = \begin{vmatrix} -1 & 0 \end{vmatrix} \begin{vmatrix} 2 & -1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} -2 & 1 \end{vmatrix}$$

DOBBIAMO CONTROLLARE SE SONO ≤ 0

1 NON É ≤ 0 E CORRISPONDE ALL'INDICE 2 $\Rightarrow h = 2$ ← INDICE USCENTE

#2 $B = \{1,3\}$ $\quad A_B = \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$

$$\bar{x} = \begin{vmatrix} 0 & -1 \\ 1 & 0 \end{vmatrix} \begin{vmatrix} 4 \\ 1 \end{vmatrix} = \begin{vmatrix} -1 \\ 4 \end{vmatrix} \qquad \bar{y} = \begin{vmatrix} 0 & 1 \end{vmatrix} \begin{vmatrix} 0 & -1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} 1 & 0 \end{vmatrix}$$

$$A_N \bar{x} = \begin{vmatrix} -1 & 2 \\ -2 & 1 \\ -1 & 0 \end{vmatrix} \begin{vmatrix} -1 \\ 4 \end{vmatrix} = \begin{vmatrix} 9 \\ 6 \\ 1 \end{vmatrix} \begin{matrix} \leq \\ \nleq \\ \nleq \end{matrix} \begin{vmatrix} 10 \\ 4 \\ 0 \end{vmatrix} \Rightarrow k = \text{MIN}\{4,5\} = 4$$

$$\eta_B = A_4 A_B^{-1} = \begin{vmatrix} -2 & 1 \end{vmatrix} \begin{vmatrix} 0 & -1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} 1 & 2 \end{vmatrix}$$

← ENTRAMBI SONO POSITIVI, QUINDI DEVO CALCOLARE I RAPPORTI

$$\vartheta = \text{MIN} \left\{ \frac{\bar{y}_i}{\eta_i} \quad \begin{matrix} \bar{y}_1 = 1 \\ \eta_1 = 1 \end{matrix} \quad \begin{matrix} \bar{y}_3 = 0 \\ \eta_3 = 2 \end{matrix} \right\} \Rightarrow \text{MIN} \left\{ \frac{1}{1}, \frac{0}{2} \right\} = 2 \Rightarrow h = 3$$

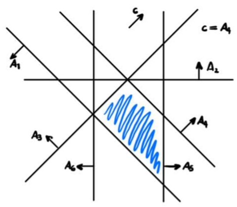
#3 $B = \{1,4\}$ $\quad A_B = \begin{vmatrix} 0 & 1 \\ -2 & 1 \end{vmatrix}$

$$\bar{x} = \begin{vmatrix} 1/2 & -1/2 \\ 1 & 0 \end{vmatrix} \begin{vmatrix} 4 \\ 4 \end{vmatrix} = \begin{vmatrix} 0 \\ 4 \end{vmatrix} \qquad \bar{y} = \begin{vmatrix} 0 & 1 \end{vmatrix} \begin{vmatrix} 1/2 & -1/2 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} 1 \\ 0 \end{vmatrix}$$

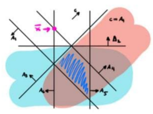
$$A_N \bar{x} = \begin{vmatrix} -1 & 2 \\ -1 & 0 \\ -1 & 0 \end{vmatrix} \begin{vmatrix} 0 \\ 4 \end{vmatrix} = \begin{vmatrix} 8 \\ 0 \\ 0 \end{vmatrix} \begin{matrix} \leq \\ \leq \\ \leq \end{matrix} \begin{vmatrix} 10 \\ 1 \\ 0 \end{vmatrix} \qquad \text{STOP} \qquad \begin{matrix} \bar{x} = (0,4) \\ \bar{y} = (1,0,0,0,0) \end{matrix}$$

SOLUZIONI OTTIME

Utilizzando il metodo geometrico:



#1 $B = \{4,6\}$



$\bar{x}$ NON É AMMISSIBILE, É FUORI DAL POLIEDRO

A_6 c A_4 $\Rightarrow \bar{y}$ É AMMISSIBILE $\quad \bar{y}_4 = 1 \quad \bar{y}_6 = 0$

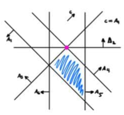
BISOGNA CONTROLLARE I VINCOLI VIOLATI $K = \text{MIN}\{2,3\} = 2$

A_2 A_6 A_4 $\Rightarrow A_2 \in \text{CONO}(A_4, A_6) \quad \eta_4 > 0 \quad \eta_6 > 0$

BISOGNA CALCOLARE I RAPPORTI

$$\vartheta = \text{MIN} \left\{ \frac{\bar{y}_4}{\eta_4} ; \frac{\bar{y}_6}{\eta_6} \right\} \Rightarrow h = 6$$

#2 $B = \{2,4\}$



$\bar{x}$ É AMMISSIBILE

A_2 c A_4 $\Rightarrow \bar{y}$ É AMMISSIBILE, $\bar{y}_2 = 0 \quad \bar{y}_4 = 1$

NESSUN VINCOLO É VIOLATO

$\bar{x}, \bar{y}$ SONO OTTIMI

Chapter 7

Programmazione lineare intera

7.1 Definizione

Un problema di PLI è nella forma

$$\begin{cases} \max \ \mathrm{c}^{\mathrm{T}}\mathrm{x} \\ Ax \leq b \end{cases}$$

dove i dati A, b e c sono tutti a componenti intere e la regione ammissibile Ω è limitata.

Il problema di PL

$$\begin{cases} \max \ \mathrm{c}^{\mathrm{T}}\mathrm{x} \\ Ax \leq b \end{cases}$$

è detto **rilassamento continuo**.

7.1.1 Relazioni tra PL e PLI

Relazione tra programmazione lineare intera (P) e rilassamento continuo (RC):

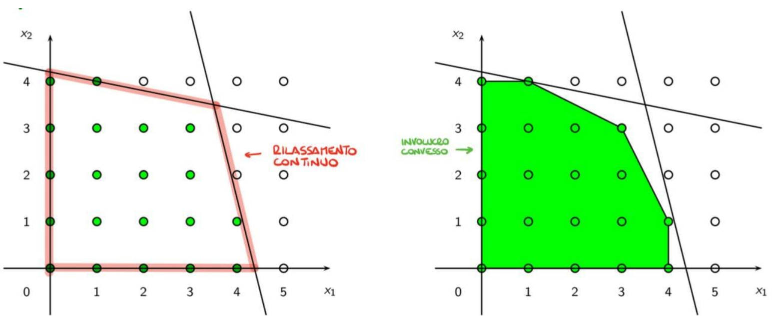
- Il valore ottimo di RC è $\geq$ del valore ottimo di P
- Se la soluzione ottima di RC è ammissibile per P, allora è ottima anche per P

Spesso la soluzione ottima di RC non è ammissibile per P.

Consideriamo i problemi

$$\begin{cases} \max \ \mathrm{c}^{\mathrm{T}}\mathrm{x} \\ x \in \Omega \end{cases} \qquad \begin{cases} \max \ \mathrm{c}^{\mathrm{T}}\mathrm{x} \\ x \in conv(\Omega) \end{cases}$$

dove conv(Ω) è l'involucro convesso delle soluzioni ammissibili, cioè il più piccolo insieme convesso che contiene Ω. Nel RC il poliedro non coincide con conv(Ω)



- conv(Ω) è un poliedro
- i vertici di conv(Ω) sono a componenti intere

- i problemi

$$\begin{cases} \max \mathrm{c^T x} \\ x \in \Omega \end{cases} \qquad \begin{cases} \max \mathrm{c^T x} \\ x \in conv(\Omega) \end{cases}$$

hanno lo stesso valore ottimo e almeno una soluzione ottima comune.

In generale è difficile trovare i vincoli che definiscono conv(Ω) (problema NB-hard). Per alcuni problemi si riesce a caratterizzare conv(Ω): se A è una matrice totalmente unimodulare, cioè il determinante di ogni sottomatrice quadrata è 0, 1 o -1 allora conv(Ω) coincide con la regione ammissibile del rilassamento continuo.

7.2 Piani di taglio di Gomory

In generale è difficile caratterizzare conv(Ω), quindi si aggiungono dei vincoli più stringenti possibili.
Una **disuguaglianza valida** è una disuguaglianza $\mathrm{p^T x} \leq \mathrm{p_0}$ per ogni $\mathrm{x} \in \Omega$. Un **piano di taglio** è una disuguaglianza valida $\mathrm{p^T}\overline{x} \leq \mathrm{p_0}$ per Ω tale che $\mathrm{p^T}\overline{x} > \mathrm{p_0}$, dove $\overline{x}$ è l'ottimo rilassamento continuo, cioè si costruisce il piano di taglio $\mathrm{p^T}\overline{x} \leq \mathrm{p_0}$ in modo da tagliare fuori $\overline{x}$ e poi si risolve nuovamente il problema di PL per ottenere una soluzione migliore.
Un tecnica per trovare un piano di taglio è con i **tagli di Gomory**. Supponiamo che il problema di PLI sia nella forma:

$$\begin{cases} \max \mathrm{c^T x} \\ Ax = b \\ x \geq 0 \end{cases}$$

e che $\overline{x}$ sia una soluzione di base ottima del rilassamento continuo P.
Poniamo:

$$A = (A_B \ A_N) \qquad x = \begin{pmatrix} x_B \\ x_N \end{pmatrix} \qquad \widetilde{A} = A_B^{-1} A_N \qquad \widetilde{b} = \bar{x}_B.$$

La parte frazionaria di un numero reale z è $\{z\} := z - \lfloor z \rfloor$, dove $\lfloor z \rfloor$ è la parte intera inferiore di z (o arrotondamento per difetto all'intero più vicino).
Esempio: $\{2.3\} = 2.3 - 2 = 0.3$, $\quad \{-1.4\} = -1.4 - (-2) = 0.6$.

Teorema
Se esiste un indice $r \in B$ tale che $\widetilde{b}_r \notin \mathbb{Z}$, allora

$$\sum_{j \in N} \{\widetilde{a}_{rj}\} x_j \geq \{\widetilde{b}_r\}$$

è un piano di taglio (detto di Gomory) per il problema (P).

dove $\tilde{\mathrm{a}}_{\mathrm{rj}}$ sono le righe della matrice $\tilde{\mathrm{A}}$

$$\begin{cases} \text{MAX } x_1 + 3x_2 \\ x_1 + 5x_2 \leq 21 \\ 8x_1 + 2x_2 \leq 35 \\ x \geq 0 \end{cases} \Longrightarrow$$

TRASFORMO I VINCOLI DI ≤ IN VINCOLI DI = E AGGIUNGO LE VARIABILI DI SCARTO x_3, x_4

$$\Longrightarrow \begin{cases} \text{MAX } x_1 + 3x_2 \\ x_1 + 5x_2 + x_3 = 21 \\ 8x_1 + 2x_2 + x_4 = 35 \\ x \geq 0 \end{cases}$$

ORA È NELLA FORMA GIUSTA

$$A = \begin{vmatrix} 1 & 5 & 1 & 0 \\ 8 & 2 & 0 & 1 \end{vmatrix} \qquad b = \begin{vmatrix} 21 \\ 35 \end{vmatrix} \qquad c^T = \begin{vmatrix} 1, 3, 0, 0 \end{vmatrix}$$

LA SOLUZIONE OTTIMA DEL RILASSAMENTO CONTINUO È $\left(\frac{7}{2}, \frac{7}{2}, 0, 0\right)$
QUINDI LA BASE OTTIMA È $B = \{1, 2\}$ COLONNE

$$A = \begin{vmatrix} 1 & 5 \\ 8 & 2 \end{vmatrix} \qquad A_B^{-1} = \begin{vmatrix} -1/19 & 5/38 \\ 4/19 & -1/38 \end{vmatrix} \qquad \widetilde{A} = A_B^{-1} A_N = \begin{vmatrix} -1/19 & 5/38 \\ 4/19 & -1/38 \end{vmatrix}$$

POICHÉ $\widetilde{b} = \left(\frac{7}{2}, \frac{7}{2}\right)$ HA ENTRAMBE LE COMPONENTI NON INTERE, ESISTONO 2 TAGLI DI GOMORRY

$r = 1 \quad N = \{3, 4\}$ LE VARIABILI CHE NON SONO IN BASE

IL TAGLIO DI GOMORY È $\left\{-\frac{1}{19}\right\} x_3 + \left\{\frac{5}{38}\right\} x_4 \geq \frac{7}{2}$

PARTE INTERA — È TRA 0 E 1

$$= \frac{18}{19} x_3 + \frac{5}{38} x_4 \geq \frac{1}{2}$$

$$= 36 x_3 + 5 x_4 \geq 19$$

CHE NELLE VARIABILI (x_1, x_2) EQUIVALE A $36(21 - x_1 - 5x_2) + 5(35 - 8x_1 - 2x_2) \geq 19$
CIOÈ $2x_1 + 5x_2 \leq 24$

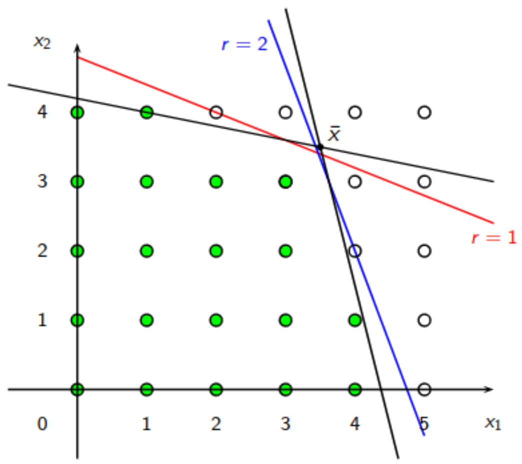
$r = 2 \quad N = \{3, 4\}$

IL TAGLIO DI GOMORY È $\left\{\frac{4}{19}\right\} x_3 + \left\{-\frac{1}{38}\right\} x_4 \geq \frac{7}{2}$

$$= \frac{4}{19} x_3 + \frac{37}{38} x_4 \geq \frac{1}{2}$$

$$= 8 x_3 + 37 x_4 \geq 19$$

CHE NELLE VARIABILI (x_1, x_2) EQUIVALE A $8(21 - x_1 - 5x_2) + 37(35 - 8x_1 - 2x_2) \geq 19$
CIOÈ $8x_1 + 3x_2 \leq 38$

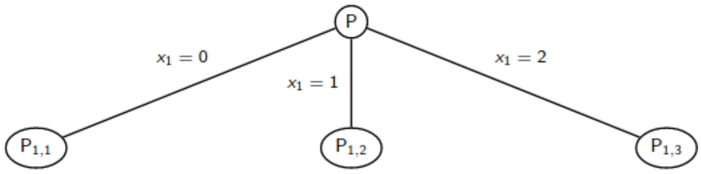


7.3 Enumerazione esplicita

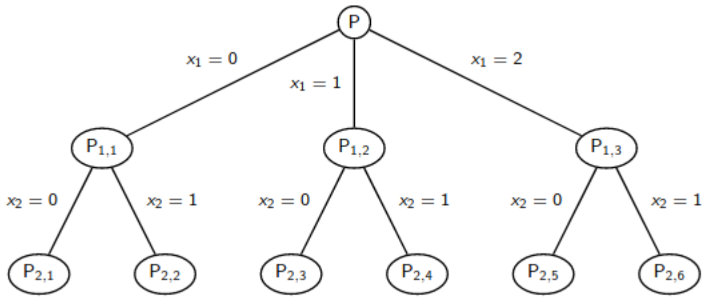
Consideriamo il problema di PLI:

$$\begin{cases} \max\ 5\mathrm{x}_1\ +\ 6\mathrm{x}_2 \\ \ 3x_1 + 4x_2 \leq 7 \\ \ x_1 \leq 0, x_2 \leq 0 \end{cases}$$

I vincoli implicano che $\mathrm{x}_1 = 0$, $\mathrm{x}_1 = 1$ o $\mathrm{x}_1 = 2$ e quindi possiamo fare una partizione della regione ammissibile in 3 sottoinsiemi:



che corrispondono al primo livello dell'albero decisionale. Analogamente $\mathrm{x}_2 = 0$ e $\mathrm{x}_2 = 1$, pertanto l'albero decisionale completo è:



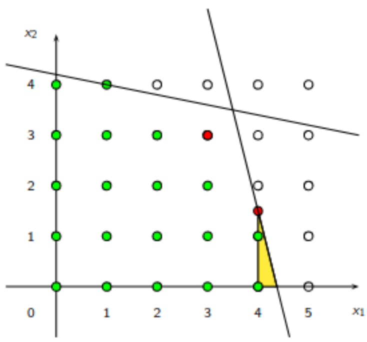
I nodi $\mathrm{P}_{2,1}$, ..., $\mathrm{P}_{2,5}$ corrispondono a soluzioni ammissibili, mentre il nodo $\mathrm{P}_{2,6}$ corrisponde alla soluzione x=(2,1) che non è ammissibile. I valori della funzione obiettivo in corrispondenza dei nodi $\mathrm{P}_{2,1}$, ..., $\mathrm{P}_{2,5}$ sono 0, 6, 5, 11, 10 pertanto la soluzione ottima è ottenuta nel nodo $\mathrm{P}_{2,4}$ con x=(1,1).
L'enumerazione esplicita è molto costosa, possiamo esaminare e scartare le soluzioni a gruppi, anziché singolarmente e utilizzare l'*enumerazione implicita*.

7.4 Enumerazione implicita

Consideriamo di nuovo il problema di PLI:

$$\begin{cases} \max\ \mathrm{x}_1\ +\ 3\mathrm{x}_2 \\ x_1 + 5x_2 \leq 21 \\ 8x_1 + 2_2 \leq 35 \end{cases}$$

Sappiamo che il punto (3,3) è ammissibile con valore 12. Se ora consideriamo il sottoproblema in cui aggiungiamo il vincolo $\mathrm{x}_1 \leq 4$: la soluzione ottima del rilassamento continuo è $(4, \frac{3}{2})$ con valore 8.5 (la regione gialla), che è la migliore soluzione del rilassamento continuo in quella regione, allora tutte le soluzioni ammissibili del sottoproblema sono peggiori di (3,3). Non serve controllare tutto l'albero singolarmente.



7.5 Metodo Branch and Bound

Componenti principali e algoritmo:

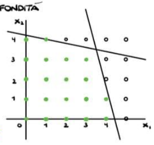
- **Branch**: partizionare la regione ammissibile in sotto-regioni aggiungendo vincoli. Il risultato deve essere una partizione della regione ammissibile per non perdere nessuna soluzione ammissibile; Il branch può essere binario o non binario;
- **Bound**: stimare il valore ottimo del sottoproblema. La **valutazione inferiore v_I** del valore ottimo è data da qualunque soluzione ammissibile, invece la **valutazione superiore v_S** è data dal valore ottimo del rilassamento;
- **Potatura**: scartare le sottoregioni che non contengono soluzioni migliori di quella corrente. Un nodo P_i può essere chiuso se vale una qualunque delle seguenti condizioni:
  - il sottoproblema P_i non ha soluzioni ammissibili
  - $v_S(P_i) \leq v_I(P)$, cioè le soluzioni ammissibili di P_i non sono migliori della soluzione ammissibile corrente
  - $v_S(P_i) > v_I(P)$ e la soluzione ottima di P_i è ammissibile per P, in tal caso si aggiorna $v_I(P) = v_S(P_i)$
- **Visita**: in quale ordine visitare i nodi nell'albero decisionale:
  - *Profondità*: il prossimo nodo da visitare è uno dei figli attualmente visitato (se rimasto aperto). Questa strategia trova rapidamente una soluzione ammissibile, occupa poco memoria, ma non tiene conto della qualità della soluzione trovata.
  - *Best first*: il prossimo nodo da visitare è il più promettente, cioè quello con il massimo valore di v_S. Trova rapidamente una soluzione ammissibile, occupa molto memoria, ma tiene conto della qualità della soluzione trovata
  - *Ampiezza*: si esplorano tutti i nodi dello stesso livello, in generale non fornisce buone prestazioni dal punto di vista computazionale.

1. Genera il nodo radice P (da visitare), trova una soluzione ammissibile per P e calcola una $v_I(P)$.
2. Se tutti i nodi sono stati visitati, allora STOP (la soluzione ammissibile corrente è ottima).
3. Seleziona un nodo P_i da visitare.
4. Se P_i non contiene soluzioni ammissibili, allora chiudi il nodo P_i e torna al passo 2.
5. (Bound) risolvi un rilassamento di P_i e calcola $v_S(P_i)$.
   - se $v_S(P_i) \leq v_I(P)$, allora chiudi il nodo P_i e torna al passo 2.
   - se $v_S(P_i) > v_I(P)$ e la soluzione ottima del rilassamento di P_i è ammissibile per P, allora chiudi il nodo P_i, aggiorna la soluzione ammissibile, poni $v_I(P) = v_S(P_i)$ e torna al passo 2.
6. (Branch) Partiziona la regione ammissibile di P_i in sotto-regioni e genera nuovi nodi da visitare. Torna al passo 2.

APPLICHIAMO IL METODO BRANCH AND BOUND PER RISOLVERE IL PROBLEMA DI PLI

$$\begin{cases} \max\ x_1 + 3x_2 \\ x_1 + 5x_2 \leq 21 \\ 8x_1 + 2x_2 \leq 35 \\ x \geq 0,\ x \in \mathbb{Z}^2 \end{cases} \Longrightarrow$$

- RISOLVIAMO IL RILASSAMENTO CONTINUO,
- USIAMO UN BRANCH BINARIO
- VISITIAMO L'ALBERO IN PROFONDITÀ



$$A = \begin{pmatrix} 1 & 5 \\ 8 & 2 \end{pmatrix} \quad b = \begin{pmatrix} 21 \\ 35 \end{pmatrix} \quad c = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

$B = \{1, 2\}$

$$\bar{x} = \begin{vmatrix} -1/19 & 5/38 \\ 4/19 & -1/38 \end{vmatrix} \begin{vmatrix} 21 \\ 35 \end{vmatrix} = \begin{vmatrix} 7/2 \\ 7/2 \end{vmatrix}$$ OTTIMO RILASSAMENTO CONTINUO

STIMA PER ECCESSO

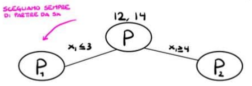
$(7/2, 7/2) \rightarrow 7/2 + 3 \cdot 7/2 \rightarrow 14 \rightarrow V_S(P) = 14$

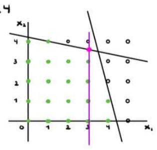
STIMA PER DIFETTO

ARROTONDO $\rightarrow (3,3) \rightarrow 3 + 3 \cdot 3 = 12 \rightarrow V_I(P) = 12$

LA PRIMA COMPONENTE NON INTERA VALE 3.5 $\rightarrow x_1 \leq 3$, $x_1 \geq 4$

SCEGLIAMO SEMPRE DI PARTIRE DA SX





VINCOLO ATTIVO

$x_1 + 5x_2 \leq 21 \rightarrow 3 + 5x_2 \leq 21 \rightarrow x_2 = 18/5$

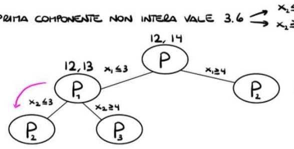
LA SOLUZIONE DEL RILASSAMENTO CONTINUO $P_1 = (3, 18/5)$

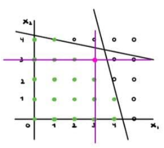
$\rightarrow 3 + 3 \frac{18}{5} = 13.8$

LA SOLUZIONE NON È AMMISSIBILE C'È UN COMPONENTE CHE NON È INTERA

$V_S(P) = 13 > 12 = V_I(P) \Rightarrow$ NON POSSO CHIUDERE IL NODO

LA PRIMA COMPONENTE NON INTERA VALE 3.6 $\rightarrow x_2 \leq 3$, $x_2 \geq 4$



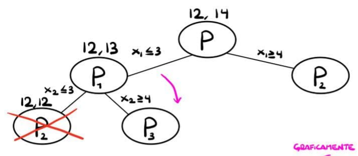


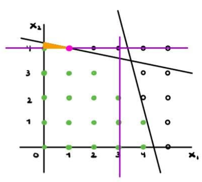
LA SOLUZIONE DEL RILASSAMENTO CONTINUO $P_2 = (3, 3)$

$\rightarrow 3 + 3 \cdot 3 = 12$

TRA TUTTE LE SOLUZIONI AMMISSIBILI CHE STANNO IN P_2 LA MIGLIORE VALE 12

$V_S(P) = 12 = V_I(P)$ CHIUDO IL NODO



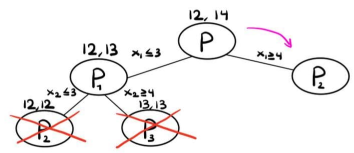


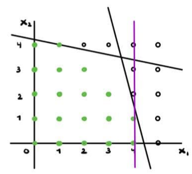
GRAFICAMENTE

LA SOLUZIONE DEL RILASSAMENTO CONTINUO $P_2 = (1, 4)$

$\rightarrow 1 + 4 \cdot 3 = 13$

$V_S(P_3) = 13 > 12 = V_I(P)$. POICHÉ $(1,4)$ È AMMISSIBILE, AGGIORNIAMO $V_I(P) = 13$ E CHIUDIAMO P_3





VINCOLO ATTIVO

$8x_1 + 2x_2 \leq 35 \rightarrow 8 \cdot 4 + 2x_2 \leq 35 \rightarrow x_2 = 3/2$

LA SOLUZIONE DEL RILASSAMENTO CONTINUO $P_1 = (4, 3/2)$

$\rightarrow 4 + 3 \cdot 3/2 = 8.5$

$V_S(P_4) = 8 < 13 = V_I(P)$ PERTANTO CHIUDO IL NODO

POICHÉ TUTTI I NODI SONO VISITATI, LA SOLUZIONE È $(1,4)$ ED IL VALORE OTTIMO È 13

7.6 Problema dello zaino

Si può risolvere con l'algoritmo del simplesso, ma essendo un problema particolare si può risolvere in un modo più rapido.

La **soluzione ammissibile** può essere ottenuta attraverso il seguente algoritmo

Ordina gli oggetti in ordine decrescente di rendimento $\frac{v_i}{p_i}$

Poni $\bar{C} = C$ ($\bar{C}$ rappresenta la capacità residua del contenitore)

for $i = 1, \dots, n$ **do**

  if $p_i \le \bar{C}$

    then $x_i = 1$, $\bar{C} = \bar{C} - p_i$

  else $x_i = 0$

end

cioè ordino gli oggetti in base al rendimento, per ogni oggetto in questo ordine vado a vedere se il peso è $\le$ alla capacità residua allora la variabile la pongo $= 1$ e aggiorno la capacità residua, altrimenti $= 0$

La **soluzione ottima** $\overline{x}$ del rilassamento continuo si può calcolare nel modo seguente:

1. Ordina gli oggetti in ordine decrescente di valore per unità di peso (rendimento) $\frac{v_i}{p_i}$
2. Trova l'indice h tale che $\sum_{i=1}^{h} p_i \le C$ e $\sum_{i=1}^{h+1} p_i > C$,
3. Poni $\bar{x}_1 = 1, \dots, \bar{x}_h = 1$, $\bar{x}_{h+1} = \frac{C - \sum_{i=1}^{h} p_i}{p_{h+1}}$, $\bar{x}_{h+2} = 0, \dots, \bar{x}_n = 0$.

cioè ordino gli oggetti in base al rendimento, parto dall'oggetto di rendimento massimo e la variabile la pongo $= 1$ e aggiorno la capacità; passo al successivo, la variabile lo pongo $= 1$, aggiorno la capacità e così via fino a che la capacità me lo permette. A un certo punto arriverò a un certo indice h dove il peso è $>$ della capacità allora inserisco solo una parte $\frac{\text{capacità residua rimasta}}{\text{peso dell'h elemento}}$ e il resto delle variabili li pongo $= 0$.

$$\begin{cases} \max\ 11x_1 + 23x_2 + 18x_3 + 6x_4 \\ 7x_1 + 6x_2 + 3x_3 + 2x_4 \le 8 \\ x_i \in \{0,1\} \quad i = 1, \dots, 4 \end{cases} \Rightarrow \frac{11}{7} = 1.6 \quad \frac{18}{3} = 6 \quad \frac{23}{6} = 3.8 \quad \frac{6}{2} = 3 \Rightarrow$$

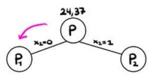
3	2	4	1
6	3.8	3	1.6

– VISITA IN AMPIEZZA –

SOLUZIONE AMMISSIBILE: $3 \le 8$ $x_3 = 1$ $\bar{C} = 5$; $6 \not\le 5$ $x_2 = 0$; $2 \le 5$ $x_4 = 1$ $\bar{C} = 3$; $3 \not\le 7$ $x_1 = 0$ $\Rightarrow$ $x = (0,0,1,1)$ $V_I(P) = 24$

SOLUZIONE OTTIMA: $3 \le 8$ $x_3 = 1$ $\bar{C} = 5$; $6 \le 5$ $x_2 = \frac{5}{6}$ $\Rightarrow$ $x = (0, 5/6, 1, 0)$ $V_S(P) = 37$

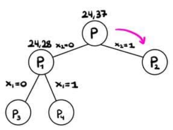
LA PRIMA COMPONENTE NON INTERA È $5/6$ → $x_2 = 0$, $x_2 = 1$



$V_S =$ $3 \le 8$ $x_3 = 1$ $\bar{C} = 5$; $2 \le 5$ $x_4 = 1$ $\bar{C} = 3$; $7 \not\le 3$ $x_1 = 7/3$

$x = (7/3, 0, 1, 1)$ $V_S(P_1) = 28$

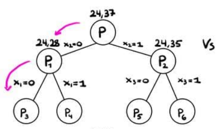
$\frac{3}{7}$ → $x_1 = 0$, $x_1 = 1$



$V_S =$ $3 \not\le 2$ $x_3 = \frac{2}{3}$

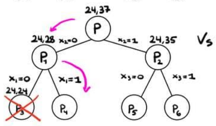
$x = (0, 1, \frac{2}{3}, 0)$ $V_S(P_2) = 35$

$\frac{2}{3}$ → $x_3 = 0$, $x_3 = 1$



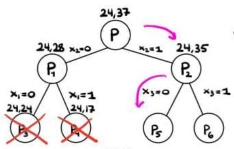
$V_S =$ $3 \le 8$ $x_3 = 1$ $\bar{C} = 5$; $2 \le 5$ $x_4 = 1$ $\bar{C} = 3$

$x = (0,0,1,1)$ $V_S(P_3) = 24 = V_I(P)$



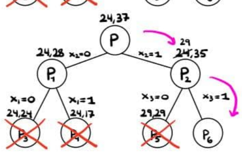
$V_S =$ $3 \le 1$ $x_3 = 1/3$

$x = (1, 0, \frac{1}{3}, 0)$ $V_S(P_4) = 17$ $V_I(P) > V_S(P_4)$



$V_S =$ $2 \le 2$ $x_4 = 1$

$x = (0,1,0,1)$ $V_S(P_5) = 29$ $V_S(P_5) > V_I(P)$ SI AGGIORNA



$\emptyset \Rightarrow 6 + 7 = 13 > $ CAPACITÀ

NON HA SOLUZIONI AMMISSIBILI

$\Rightarrow$ SOLUZIONE OTTIMA $x = (0,1,0,1)$ DI VALORE 29

7.7 Problema del commesso viaggiatore

Dato un grafo completo il cui c_{ij} è il costo dell'arco (i,j), trovare un ciclo che passi su tutti i nodi una ed una sola volta (**ciclo hamiltoniamo**) di costo totale minimo.

Variabili: $x_{ij} = \begin{cases} 1 & \text{se } \{i,j\} \in \text{ciclo hamiltoniano,} \\ 0 & \text{altrimenti,} \end{cases}$ con $i < j$.

$$\min \sum_{i \in N} \sum_{\substack{j \in N \\ j > i}} c_{ij} x_{ij}$$

$$\sum_{\substack{j \in N \\ j > i}} x_{ij} + \sum_{\substack{j \in N \\ j < i}} x_{ji} = 2 \qquad \forall\, i \in N \qquad (6)$$

$$\sum_{i \in S} \sum_{\substack{j \in S \\ j > i}} x_{ij} \le |S| - 1 \quad \forall\, S \subset N, \quad |S| \ge 3 \qquad (7)$$

$$x_{ij} \in \{0,1\} \quad \forall\, i,j \in N, \quad i < j$$

(6): ogni nodo ha grado 2.
(7): eliminazione di sottocicli.

Ogni nodo ha un arco entrante e uno uscente, per essere hamiltoniamo e quindi ha grado 2.
Un **r-albero** è un insieme di n archi di cui:

- 2 sono incidenti sul nodo r
- n-2 formano un albero di copertura sui nodi diversi da r

Ogni ciclo hamiltoniamo è un r-albero. Il problema dell'r-albero è un **rilassamento v_I**, ed è facile da risolvere perchè basta trovare:

- 2 archi di costo minimo incidenti su r
- un albero di copertura di costo minimo sui nodi diversi da r (algoritmo di Kruskal/Prim)

L'algoritmo del nodo più vicino fornisce $\mathbf{v_S}$:

0. Scegli un nodo qualunque i, poni $k = i$ (nodo corrente), $U = N \setminus \{i\}$ (nodi non visitati), $C = i$ (ciclo - sequenza di nodi).
1. Se $U = \emptyset$, allora aggiungi i in fondo a C e STOP.
2. Scegli il nodo di U più vicino a k, cioè $j = \arg\min\{i \in U : \ c_{ki}\}$.
3. Aggiungi j in fondo a C, poni $U = U \setminus \{j\}$ e $k = j$. Torna al passo 1.

L'algoritmo dipende dal nodo di partenza, se cambiamo nodo di partenza, possiamo trovare un ciclo diverso.