Flusso a costo minimo

Consideriamo un grafo $G=(N, A)$, con capacità $u_{ij} \geq 0$ sugli archi. Il problema:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = b(i) \qquad \forall i \in N$$

$$0 \leq x_{ij} \leq u_{ij} \qquad \forall (i,j) \in A$$

si dice problema di flusso a costo minimo.

Assumiamo $c_{ij} \geq 0$ e $b(N) = 0$.

Osservazioni

Osservazione 1

Supponiamo che le capacità sugli archi abbiano valore infinito e che esistano un nodo s e un nodo t tali che:

$b(s) = 1$

$b(t) = -1$

Se $b(i) = 0$ per ogni $i \in V$, $i \neq s$, t, il problema di flusso a costo minimo diventa il problema del cammino minimo da s a t

Osservazioni

Osservazione 2

Supponiamo che

1. $b(i) = 0$ per ogni $i \in N$;
2. $c_{ij} = 0$ per ogni arco $(i, j) \in V$.

Scegliamo 2 nodi di G, s e t e aggiungiamo l'arco (t, s) di capacità infinita e costo $c_{ts} = -1$ il problema di flusso a costo minimo diventa il problema di determinare il massimo flusso da s a t.

Condizioni di ottimalità

Associamo ad ogni vincolo di conservazione una variabile y_i e ad ogni vincolo di capacità una variabile z_{ij}

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

s.t.

$$y_i : \sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = b(i) \qquad \forall i \in N$$

$$z_{ij} : -x_{ij} \geq -u_{ij} \qquad \forall (i,j) \in A$$

$$x_{ij} \geq 0$$

Problema duale

$$\max \sum_{i \in N} b(i) y_i - \sum_{(i,j) \in A} u_{ij} z_{ij}$$

s.t.

$$y_i - y_j - z_{ij} \leq c_{ij} \qquad \forall (i, j) \in A \qquad (D)$$

$$z_{ij} \geq 0 \qquad \forall (i, j) \in A$$

Si definisce costo ridotto la quantità:

$$\bar{c}_{ij} = c_{ij} - y_i + y_j$$

Condizioni di ottimalità

Supponiamo che tutte le capacità siano infinite. Ciò significa che nel problema D non compaiono le variabili z. Pertanto, una soluzione ammissibile y per D ha la proprietà di avere

$$c_{ij} \geq y_i - y_j \qquad \text{che implica} \qquad \bar{c}_{ij} \geq 0$$

Se $u_{ij} \neq +\infty$, una soluzione ammissibile (y, z) di D ha la proprietà

$$z_{ij} \geq -\bar{c}_{ij}$$

ma si ha anche

$$z_{ij} \geq 0$$

In conclusione, $z_{ij} = \max\{0, -\bar{c}_{ij}\}$

Condizioni di complementarità

$$x_{ij} > 0 \Rightarrow y_i - y_j - z_{ij} = c_{ij} \Rightarrow$$

$$\Rightarrow \bar{c}_{ij} = -z_{ij} = \max\{0, -\bar{c}_{ij}\} \Rightarrow \bar{c}_{ij} \leq 0$$

$$z_{ij} > 0 \Rightarrow x_{ij} = u_{ij}$$

Pertanto, se $\bar{c}_{ij} < 0 \Rightarrow x_{ij} = u_{ij}$

Inoltre, $\bar{c}_{ij} > 0 \Rightarrow x_{ij} = 0$

Abbiamo quindi dimostrato il seguente teorema:

Condizioni di complementarità

Teorema (condizioni di ottimalità per il min cost flow)

Una soluzione x ammissibile per P è ottima se e solo se esiste un vettore $y \in R^{|N|}$ tale che, per ogni $(i, j) \in A$ soddisfa le seguenti condizioni:

Se $\bar{c}_{ij} = c_{ij} - y_i + y_j < 0$ allora $x_{ij} = u_{ij}$ (se $u_{ij} \neq +\infty$)

Se $0 < x_{ij} < u_{ij}$ allora $\bar{c}_{ij} = 0$

Se $\bar{c}_{ij} > 0$ allora $x_{ij} = 0$

Percorsi, cammini e cicli (richiami)

Sia $G = (N,A)$ un grafo orientato:

Un percorso in G è una sequenza $v_0, a_1, v_1, a_2, \ldots, a_k, v_k$ in cui ogni v_i è un nodo, ogni a_i è un arco e, per ogni $0 \leq i \leq k$, $a_i = (v_{i-1}, v_i)$ oppure $a_i = (v_i, v_{i-1})$

Un percorso $v_0, a_1, v_1, a_2, \ldots, a_k, v_k$, in G è detto orientato se, per ogni $0 \leq i \leq k$, $a_i = (v_{i-1}, v_i)$

Un percorso (orientato) senza ripetizione di nodi è detto cammino (orientato)

Il sottografo composto dal cammino $v_0, a_1, v_1, a_2, \ldots, a_k, v_k$ e dall'arco (v_k, v_0) oppure (v_0, v_k) è detto ciclo

Il sottografo composto dal cammino orientato $v_0, a_1, v_1, a_2, \ldots, a_k, v_k$ e dall'arco (v_k, v_0) è detto ciclo orientato

Problema di cammino minimo

Dato un cammino orientato P in un grafo G con pesi sugli archi l_{uv}, $(u, v) \in A$, definiamo lunghezza del cammino P la somma delle lunghezze dei suoi archi:

$$l(P) = \sum_{(u,v) \in P} l_{uv}$$

Sia $L = \max\,(l_{uv}: (u,v) \in A)$

Istanza: un grafo orientato $G = (N, A)$ con pesi (lunghezze) l_{uv}, $(u,v) \in A$, associati agli archi; un nodo $r \in V$.

Problema: trovare, per ogni $v \in N$, un cammino orientato da r a v di lunghezza minima.

Ipotesi 1

G contiene un cammino orientato da r a ciascuno degli altri suoi nodi

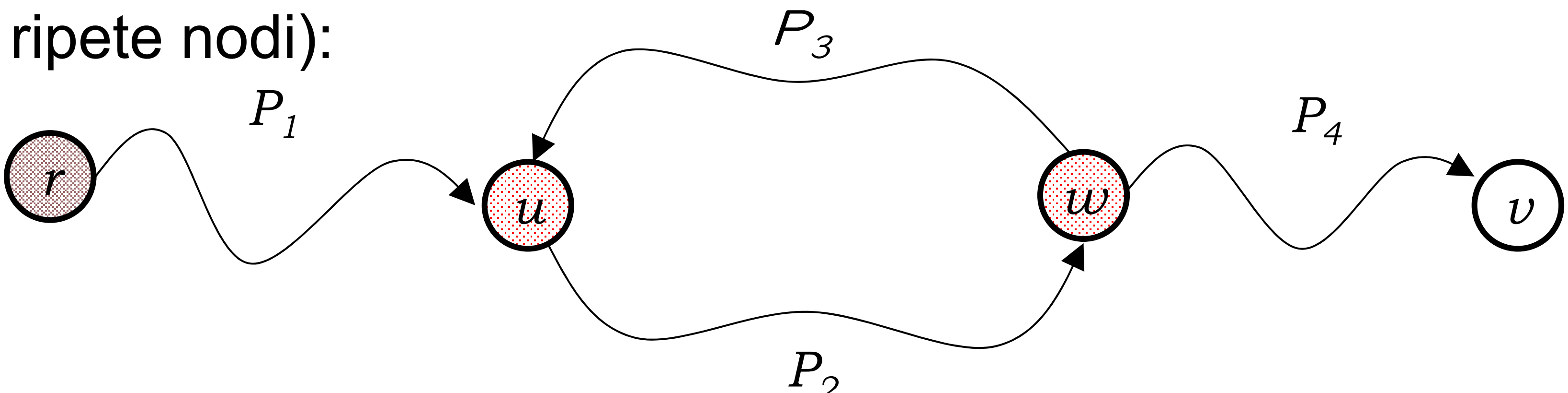
In caso contrario, si aggiungono al grafo archi del tipo (r,v) per ciascun nodo v per cui non esiste in G un cammino da r a v.

Associando a tali archi (r,v) un peso sufficientemente grande (quanto?), questi saranno contenuti in un cammino minimo se e solo se non esiste un cammino da r a v nel grafo originario.

Ipotesi 2

G non contiene cicli di lunghezza negativa raggiungibili da r

In questo caso, esiste almeno un percorso di lunghezza minima fra r e v, $v \in V$, che è un cammino (cioè non ripete nodi):



Il cammino P_1-P_2-P_4 ha lunghezza non maggiore del percorso (minimo) P_1-P_2-P_3-P_2-P_4

Ipotesi 2 (conseguenze)

Di fatto, gli algoritmi per il calcolo di cammini minimi determinano percorsi minimi da *r* agli altri nodi

Nel caso in cui *G* possa contenere cicli negativi raggiungibili da *r*, il percorso minimo può attraversare un ciclo infinite volte.

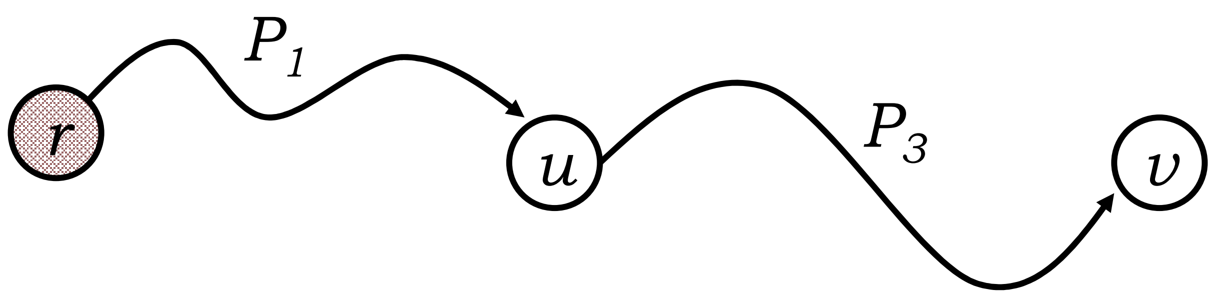
Per calcolare il cammino minimo è quindi necessario imporre un vincolo che eviti la ripetizione dei nodi

Tale vincolo rende il problema NP-hard.

Struttura della soluzione ottima

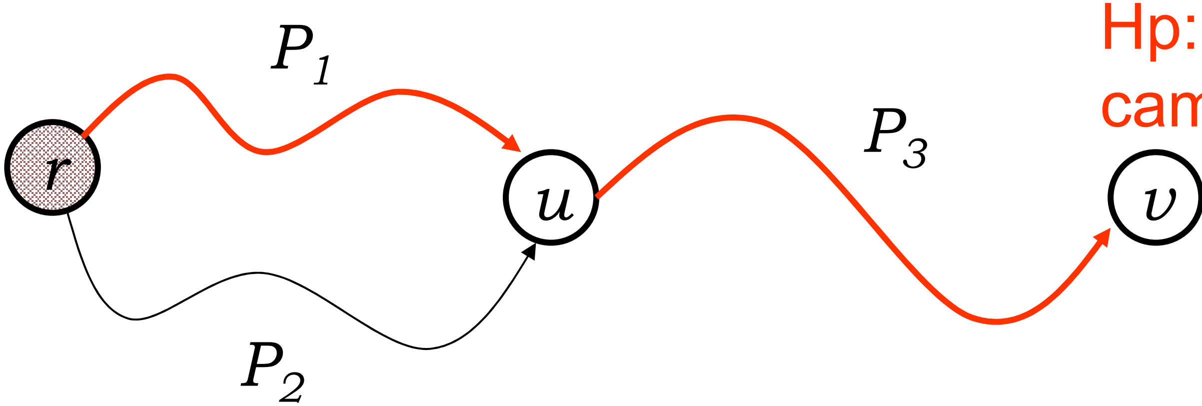
Proprietà 1. (Ereditarietà della struttura ottima)

Se un cammino $r, a_1, v_1, a_2, \ldots, a_k, v_k$ è un cammino minimo da r a v_k, allora per ogni q =1,2,…,k-1 il cammino $r, a_1, v_1, a_2, \ldots, a_q, v_q$ è un cammino minimo da r a v_q



P_1 - P_3 cammino minimo da r a v $\Rightarrow$ P_1 cammino minimo da r a u

Struttura della soluzione ottima



Hp: P_1 - P_3 cammino minimo

Supponiamo che P_1 non sia un cammino minimo da r a u e risulti $l(P_1) > l(P_2)$. Allora il percorso (notare che P_2 e P_3 possono essere non disgiunti) P_2-P_3 ha lunghezza inferiore a $l(P_1\text{-}P_3)$.

Dato che G non contiene cicli negativi, P_2-P_3 contiene almeno un cammino P da r a v con $l(P) < l(P_1\text{-}P_3)$

Albero dei cammini minimi

Definizione. Un albero $T = (N, A')$ di radice r è detto *albero dei cammini minimi* rispetto alla funzione lunghezza l: $A \to R$ se $A' \subseteq A$ è tale che, per ogni $v \in V$, il cammino da r a v in T è un cammino minimo da r a v in G.

Teorema. Sia $G = (N, A)$ un grafo orientato, $r \in V$, e l: $A \to R$. Allora esiste un albero dei cammini minimi (di radice r) rispetto ad l

Albero dei cammini minimi

Dimostrazione

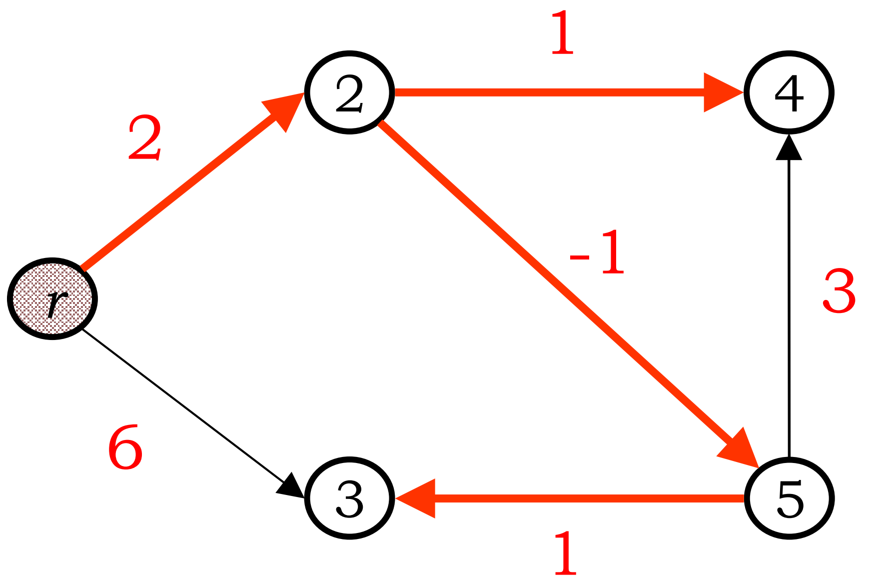
Sia $A' \subseteq A$ un insieme minimale contente per ogni $v \in N$, un cammino minimo da r a v in G,

La Proprietà 1 implica che, per ciascun nodo $v \in N$, A' contiene, un unico arco entrante.

Quindi, $|A'| = n - 1$ ed il corrispondente sottografo contiene esattamente un cammino da r a v.

$\square$

Esempio



Albero dei cammini minimi

Potenziali

Una funzione $y: V \to \mathcal{R}$ è detta potenziale se

$$y_v - y_u \leq l_{uv}, \text{ per ogni } (u,v) \in A \qquad (1)$$

Osservazione: y non dipende dal nodo r.

Teorema Sia $G = (N, A)$ un grafo orientato, $r \in V$, ed l: $A \to \mathcal{R}$. Allora esiste un potenziale se e solo se ogni ciclo orientato ha lunghezza non negativa. Inoltre, se l è intera, esiste un potenziale intero.

Dimostrazione

Necessità

Sia y un potenziale e $C = v_0, a_1, v_1, a_2, \ldots, a_k, v_k = v_0$ un ciclo orientato. Allora:

$$l(C) = \sum_{i=1}^{k} l(a_i) \geq \sum_{i=1}^{k} \left(y_{v_i} - y_{v_{i-1}} \right) = 0$$

Sufficienza

Definisco un algoritmo che costruisce un potenziale nell'ipotesi che non ci siano cicli di lunghezza negativa.

Costruzione di un potenziale

Per ogni nodo v, sia y_v la lunghezza di un percorso di lunghezza minima da un qualsiasi altro qualsiasi nodo $u \neq v$ a v.

La funzione y_v è un potenziale.

Infatti, se non lo fosse, esisterebbe un arco (u,v) per cui $y_v - y_u > l_{uv}$. Indicando con P_1 il percorso minimo entrante in u, si ha che il percorso P_1-$\{(u,v)\}$ sarebbe un percorso di lunghezza inferiore a y_v, ovvero una contraddizione.

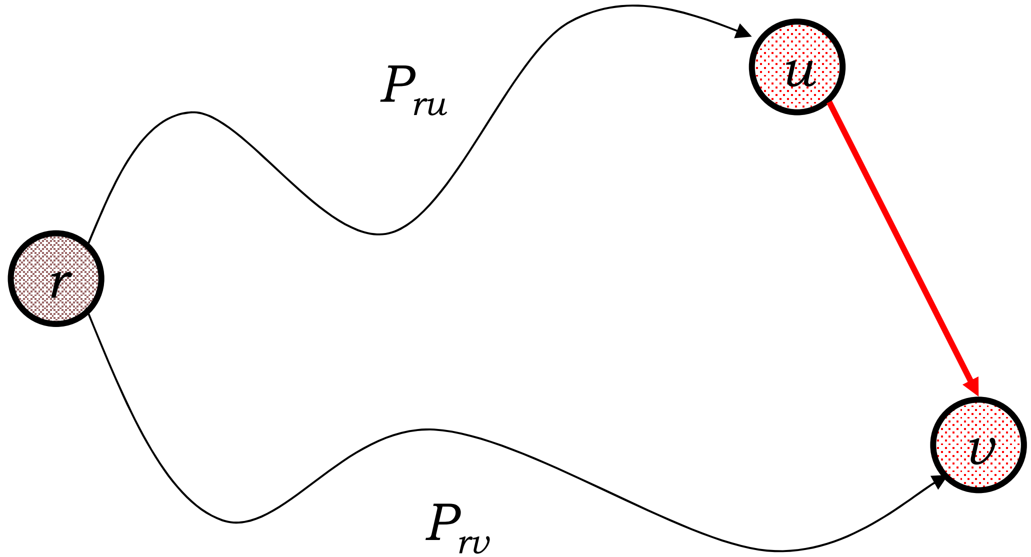
Osservazione

Questo algoritmo è valido anche scegliendo, per ogni nodo v, lo stesso nodo r da cui calcolare il percorso minimo. Ovvero:

Costruzione di un potenziale

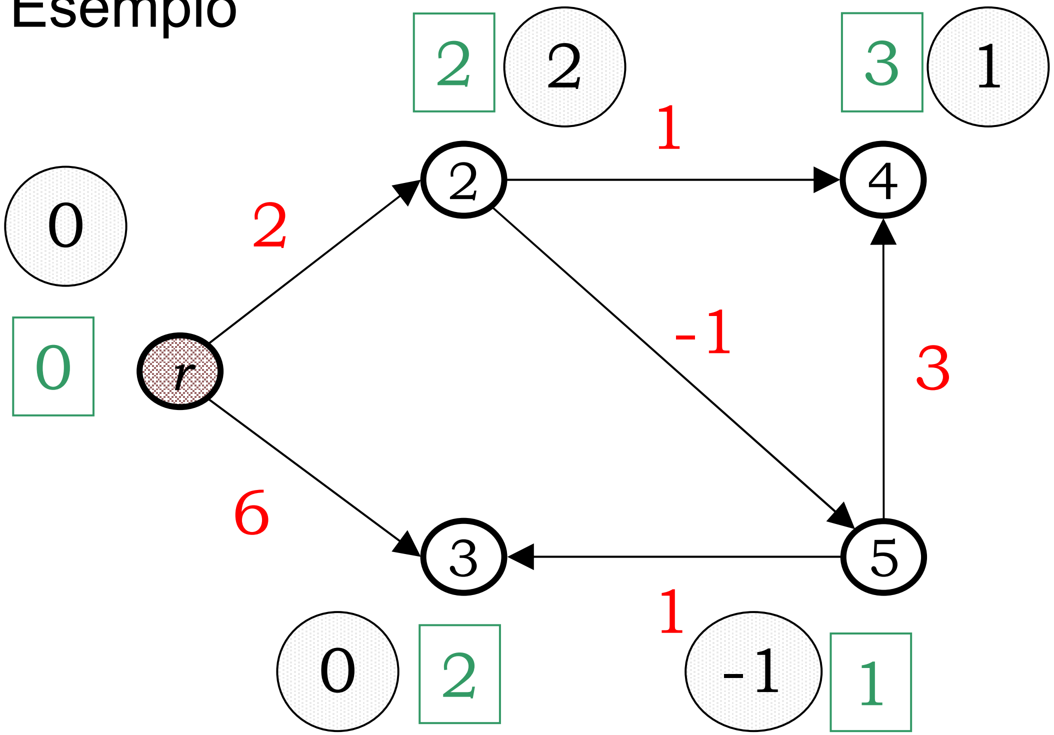
sia r un nodo di G. Per ogni $v \in N$, sia y_v la lunghezza minima di un percorso da r a v. La funzione y è un potenziale.

Se non lo fosse, esisterebbe un arco (u,v) per cui $y_v > y_u + l_{uv}$. Allora, concatenando l'arco (u,v) con il cammino P_{ru} si ottiene un nuovo cammino da r a v di lunghezza inferiore a y_v



Costruzione di un potenziale (II)

Esempio



	y'	y
r=1	0	0
2	2	2
3	2	0
4	3	1
5	1	-1

y ed y' sono due potenziali ammissibili, ma $y'_v \geq y_v$, per ogni $v \in V$

Domanda: è possibile “aumentare” y'_v per qualche v?

Costruzione di un potenziale (II)

Istanza: un grafo orientato $G=(V,A)$ con pesi (lunghezze) l_{uv}, $(u,v)\in A$, associati agli archi; un nodo $r \in V$.

Problema: determinare un potenziale y tale che $y_v - y_r$ sia massimo, per ciascun $v\in V$.

Proprietà dei potenziali

Dato un potenziale y, sottraendo y_r a ciascun y_v, $v \in N$, si ottiene un nuovo potenziale per cui $y_r = 0$.

Un qualsiasi potenziale (con $y_r = 0$) fornisce limitazioni inferiori delle lunghezze dei cammini minimi:

Lemma. Sia y potenziale e $P = r = v_0, a_1, v_1, a_2, \ldots, a_k, v_k = v$ un cammino da r a v.

Allora $l(P) \geq y_v$

Dimostrazione. Vale la seguente disuguaglianza:

$$l(P) = \sum_{i=1}^{k} l_{a_i} \geq \sum_{i=1}^{k} \left(y_{v_i} - y_{v_{i-1}} \right) = y_{v_k} - y_{v_0} = y_v$$

□

Relazione min-max

Teorema (max potenziale – min lavoro).

Dato un grafo orientato $G = (N, A)$, una coppia di suoi nodi r e v ed una qualunque funzione lunghezza l: $A \to R$,

la massima differenza di potenziale $y_v - y_r$ è pari alla minima lunghezza di un cammino da r a v.

Inoltre, se le lunghezze sono intere, tale potenziale è intero.

Relazione min-max

Dimostrazione

Dal lemma precedente, sappiamo che il potenziale fornisce una limitazione inferiore della lunghezza del cammino minimo da r a v, $v \in N$.

Inoltre, ponendo y_v pari alla lunghezza di un cammino minimo da r a v, si ottiene un potenziale. Tale potenziale è quindi tale da massimizzare $y_v - y_r$ per ogni $v \in N$.

Algoritmo di Ford

Idea chiave

Calcolare un potenziale e contemporaneamente un cammino P da r a v di lunghezza pari a y_v, $v \in N$ (dal teorema precedente deriva che P è un cammino da r a v di lunghezza minima).

Algoritmo di “discesa duale”

In una generica iterazione si mantiene un vettore di distanze y che non è un potenziale. In particolare, y_v è la lunghezza di un qualche cammino da r a v.

Si usa un vettore dei predecessori p, in cui p_v contiene il predecessore di v nel cammino di lunghezza y_v individuato.

Algoritmo di Ford

Inizializzazione

$y_r = 0,\ y_v = \infty$ per ogni $v \in V,\ v \neq r$.

while (esiste un arco per cui $y_v - y_u > l_{uv}$){

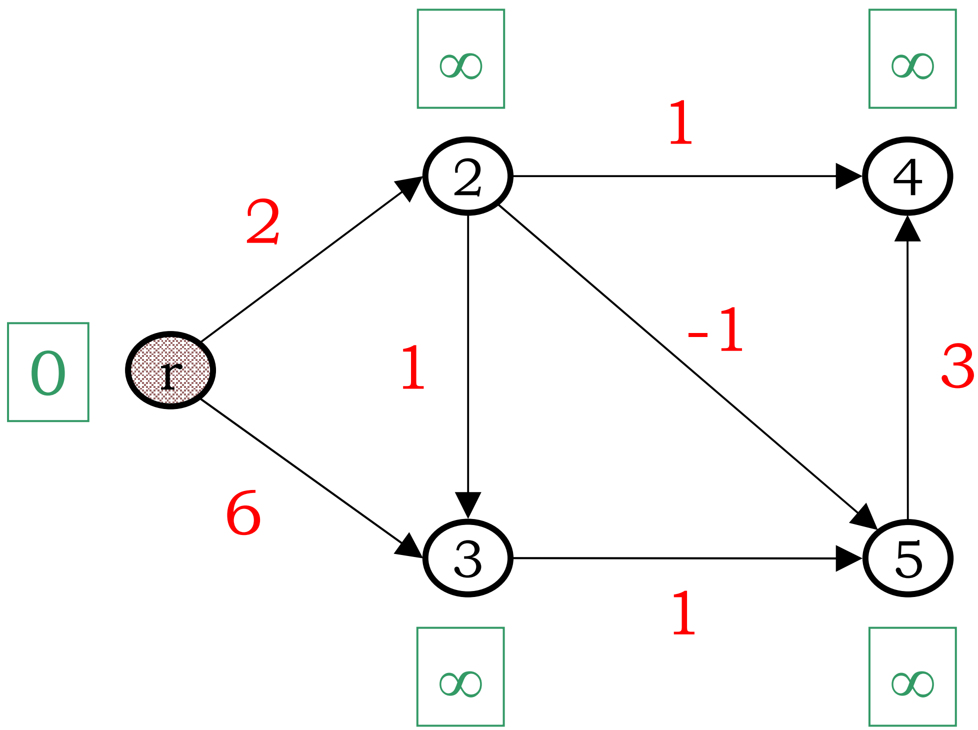
$y_v := y_u + l_{uv}$;

$p_v := u$;

}

Esempio

Inizializzazione

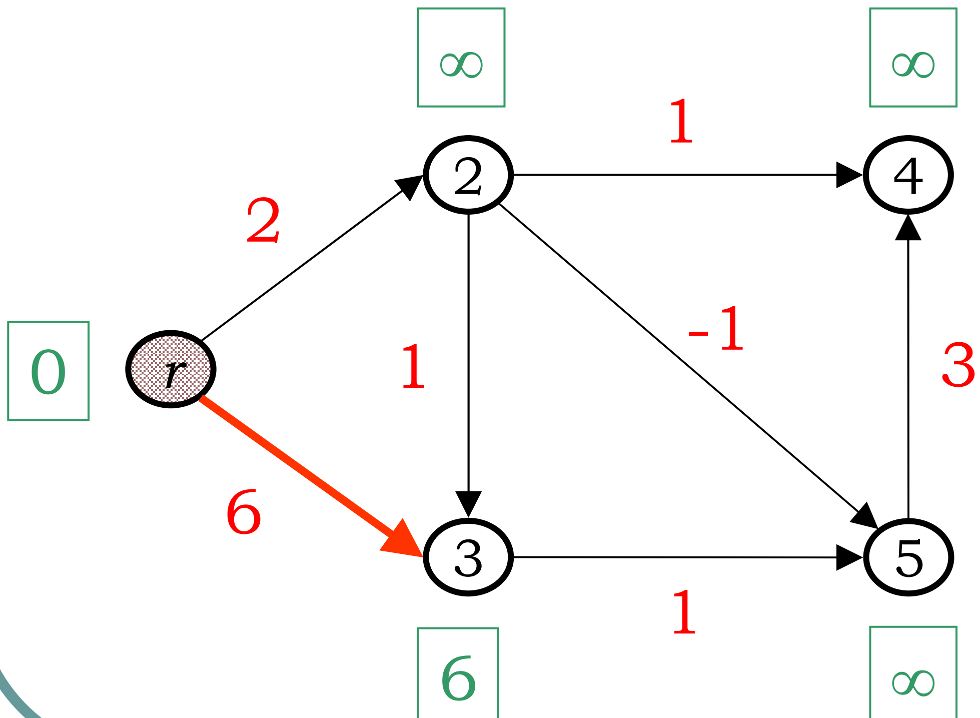


	y	p
r	0	0
2	∞	-1
3	∞	-1
4	∞	-1
5	∞	-1

Esempio

Iterazione 1

arco scelto: $(r,3)$

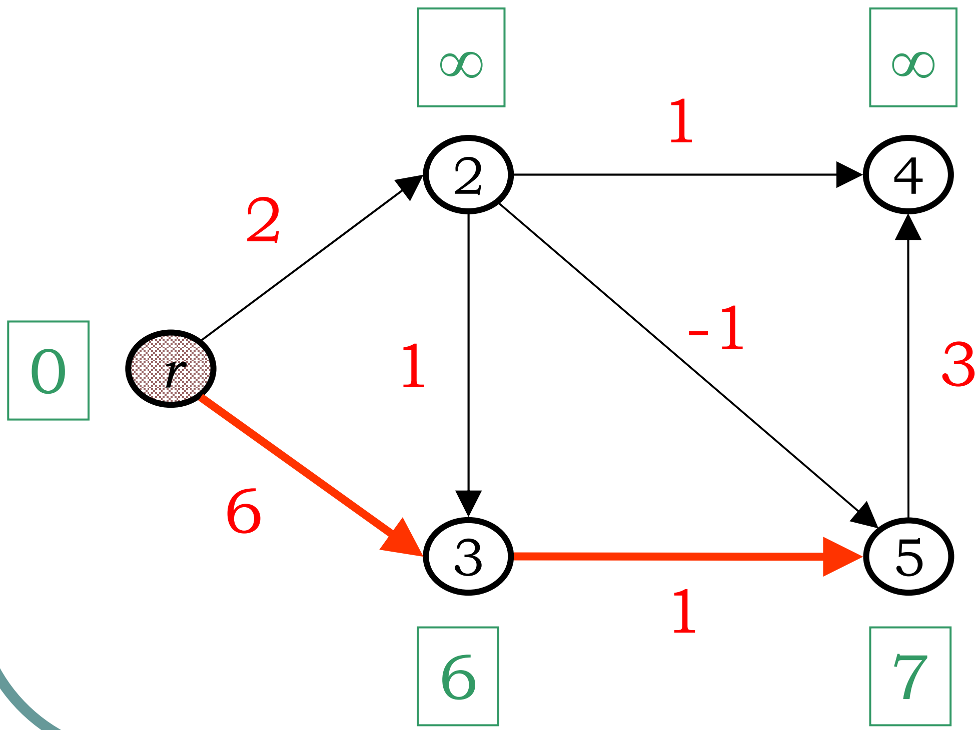


	y	p
r	0	0
2	∞	-1
3	6	r
4	∞	-1
5	∞	-1

Esempio

Iterazione 2

arco scelto: (3,5)

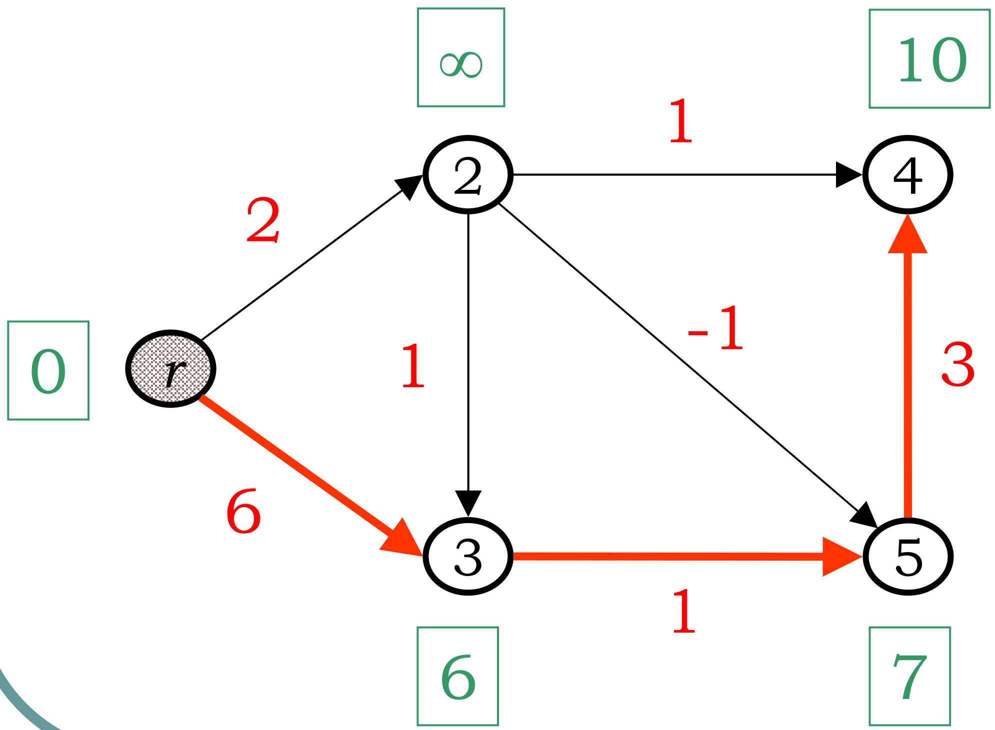


	y	p
r	0	0
2	∞	-1
3	6	r
4	∞	-1
5	7	3

Esempio

Iterazione 3

arco scelto: (5,4)

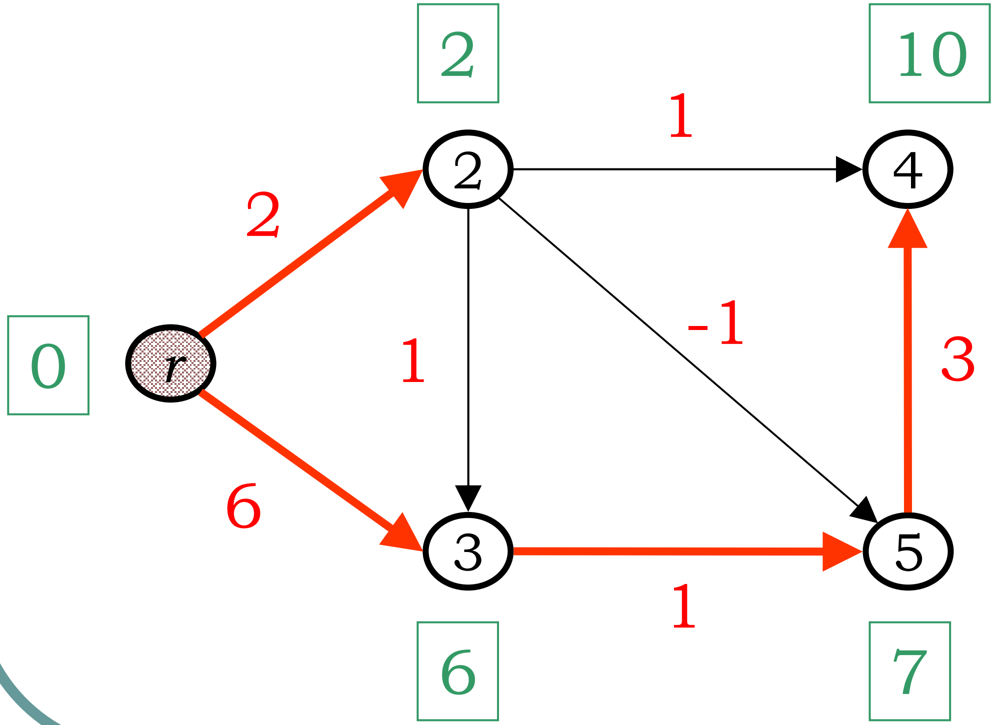


	y	p
r	0	0
2	∞	-1
3	6	r
4	10	5
5	7	3

Esempio

Iterazione 4

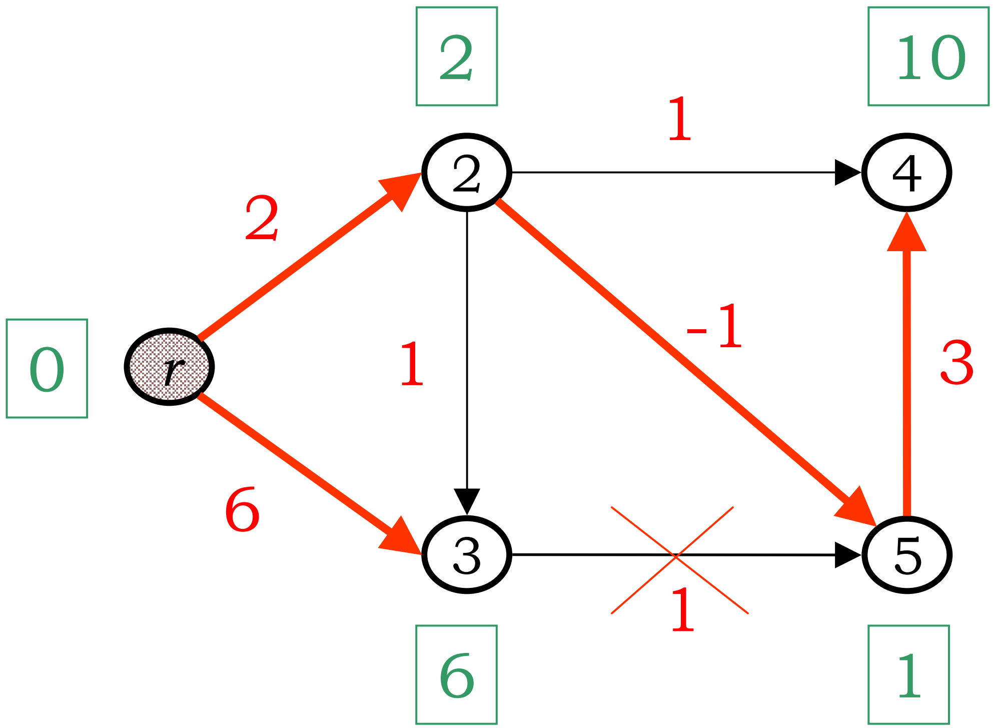
arco scelto: $(r,2)$



	y	p
r	0	0
2	2	r
3	6	r
4	10	5
5	7	3

Esempio

Iterazione 5 arco scelto: (2,5)



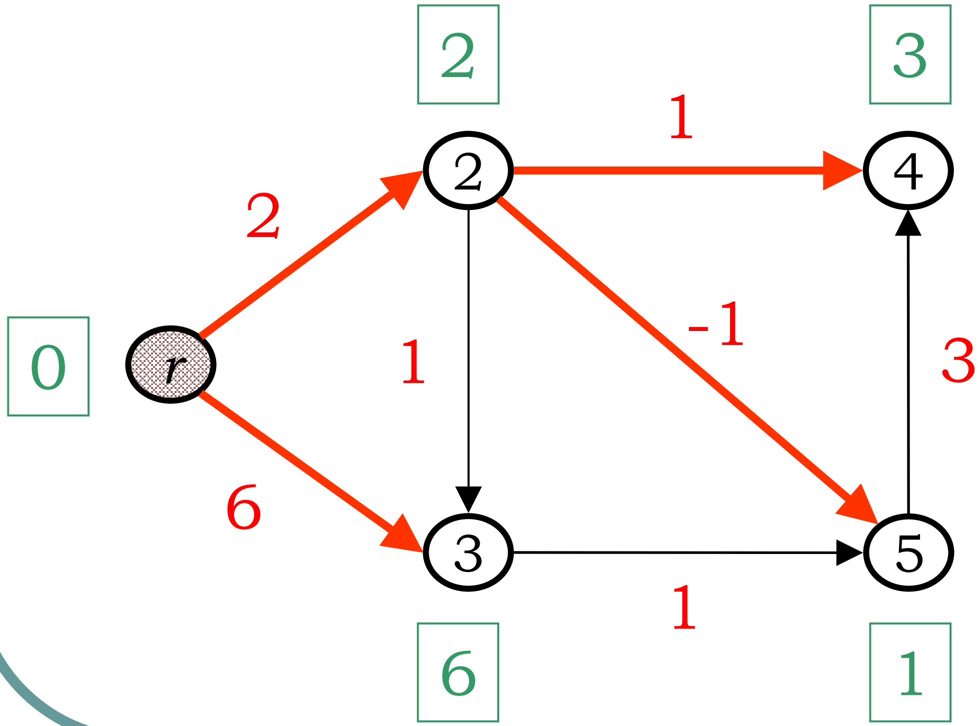
	y	p
r	0	0
2	2	r
3	6	r
4	10	5
5	1	2

Osservazione l'arco (3,5) esce dall'albero ed è sostituito da (2,5), mentre l'arco (5,4) diventa violato (!!)

Esempio

Iterazione 6

arco scelto: (2,4)

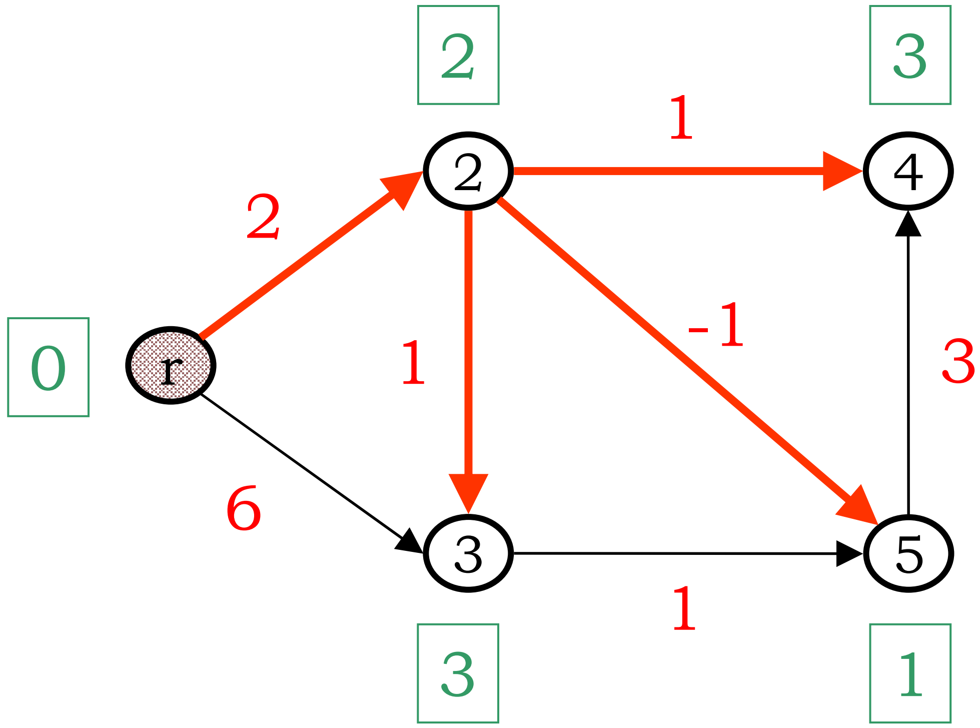


	y	p
r	0	0
2	2	r
3	6	r
4	3	2
5	1	2

Esempio

Iterazione 7

arco scelto: (2,3)



	y	p
r	0	0
2	2	r
3	3	2
4	3	2
5	1	2

Stop: non esistono archi violati

Analisi

Proposizione 1. Per ogni arco del grafo dei predecessori G_p, risulta

$$y_v - y_u \geq l_{uv}$$

Dim:
Quando un arco (u,v) viene inserito, si esegue $y_v = y_u + l_{uv}$.
Nelle iterazioni successive:

Se y_u diminuisce, allora $y_v - y_u > l_{uv}$
(Es: arco (5,4) all'iter. 3)

Se, invece, diminuisce y_v si ha che (u,v) esce da G_p

$\square$

Analisi

Proposizione 2. Ad una generica iterazione il grafo G_p, contiene un cammino P_v da r a ciascun v per cui $y_v \neq \infty$ di lunghezza $l(P_v) \leq y_v$

Infatti, sia P_v il cammino $r = v_0, a_1, v_1, a_2, \ldots, a_k, v$. Dalla Proposizione 1 si ha che:

$$l(P_v) = \sum_{i=1}^{k} l_{a_i} \leq \sum_{i=1}^{k} \left(y_{v_i} - y_{v_{i-1}} \right) = y_{v_k} - y_{v_0} = y_v$$

$\square$

Complessità

Se la funzione lunghezza $l: A \to Z$ è intera, l'algoritmo di Ford converge in $O(n^2 L)$.

Infatti:

la distanza di un nodo da r è limitata superiormente da nL, in quanto un cammino contiene al più n-1 archi ciascuno di lunghezza non superiore a L. Analogamente, la minima distanza di un nodo da r è $-nL$.

Data l'ipotesi di interezza, la distanza si riduce almeno di una unità ad ogni aggiornamento. Quindi il massimo numero di aggiornamenti per un nodo è $2nL$ e, complessivamente, $2n^2L$.

Dato che ogni iterazione esegue un aggiornamento, l'algoritmo converge in $O(n^2L)$ iterazioni

Validità

Teorema

L'algoritmo di Ford termina restituendo l'albero dei cammini minimi da r a v, $v \in N$.

La terminazione dell'algoritmo è dimostrata. La condizione di arresto e la Proposizione 1 implicano che l'algoritmo restituisce un grafo dei predecessori G_p in cui ciascun arco soddisfa $y_v - y_u = l_{uv}$.

Quindi, G_p, contiene un cammino P_v da r a v per cui di lunghezza $l(P_v) = y_v$

Il teorema del max potenziale-min lavoro implica l'ottimalità di tali cammini

□

Individuazione di cicli negativi

Cosa succede se G può contenere cicli orientati di lunghezza negativa?

L'algoritmo di Ford può essere modificato in modo da arrestarsi restituendo un ciclo negativo se il grafo ne contiene almeno uno.

Esistono due criteri per l'individuazione:

1. Basato sulle distanze y
2. Basato sulla struttura del grafo dei predecessori

Criterio 1

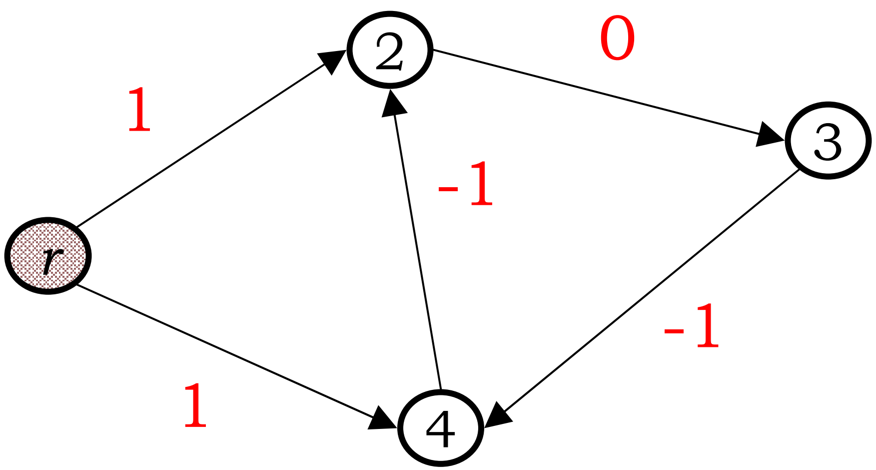
Se G contiene un ciclo orientato di lunghezza negativa, allora non esiste un potenziale. Quindi, l'algoritmo di Ford continua a diminuire le distanze indefinitamente.

Ma $-nL$ è una limitazione inferiore della lunghezza di un qualunque cammino quando G non contiene cicli negativi.

Quindi, se una distanza y_v diventa minore di $-nL$ esiste un ciclo negativo. Esso è contenuto nel grafo dei predecessori G_p.

Esempio

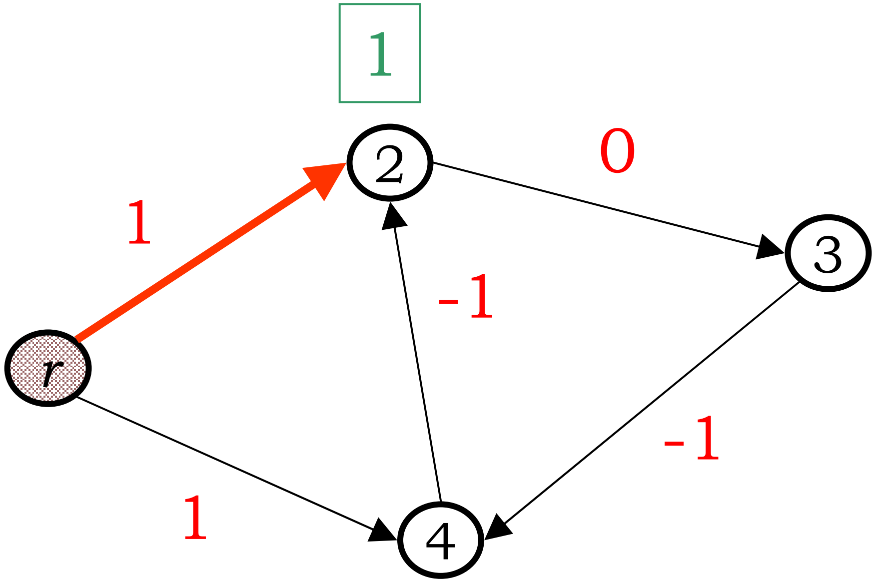
Applichiamo l'algoritmo al grafo ed alla funzione lunghezza di figura



	y	p
r= 1	0	0
2	∞	-1
3	∞	-1
4	∞	-1

Esempio

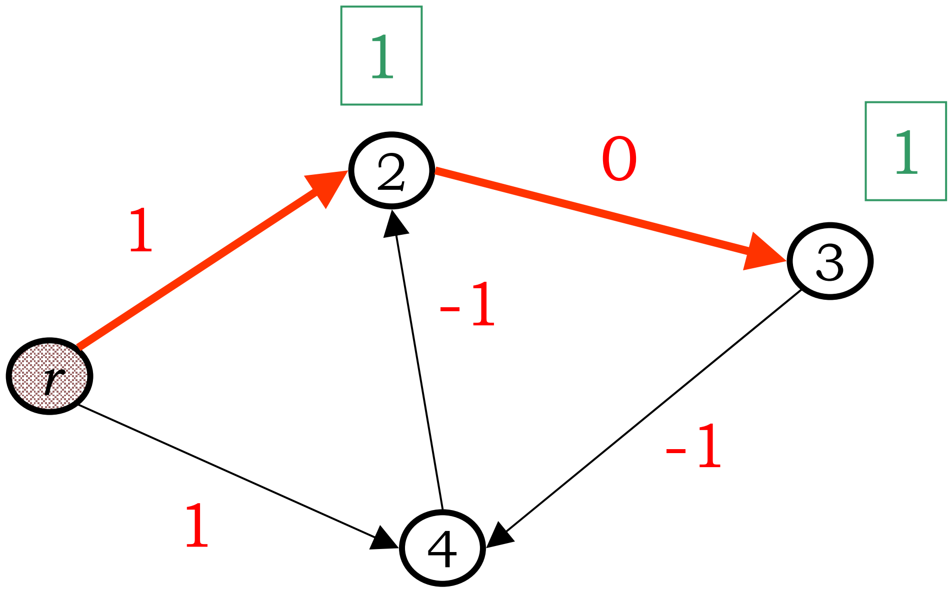
Iter 1. arco scelto $(r,2)$



	y	p
r= 1	0	0
2	1	r
3	∞	-1
4	∞	-1

Esempio

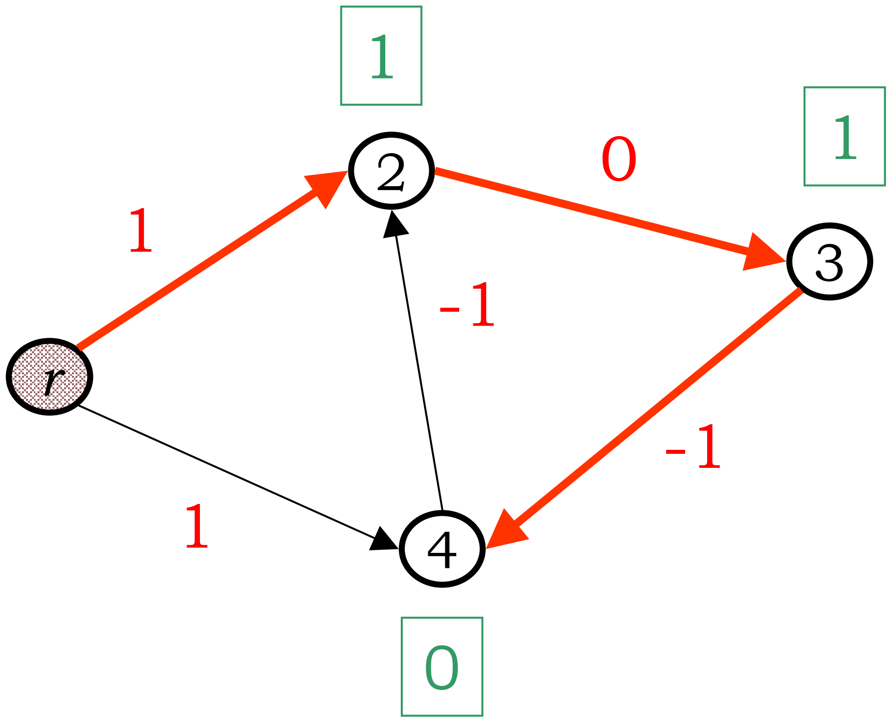
Iter 2. arco scelto $(2,3)$



	y	p
r= 1	0	0
2	1	r
3	1	2
4	∞	-1

Esempio

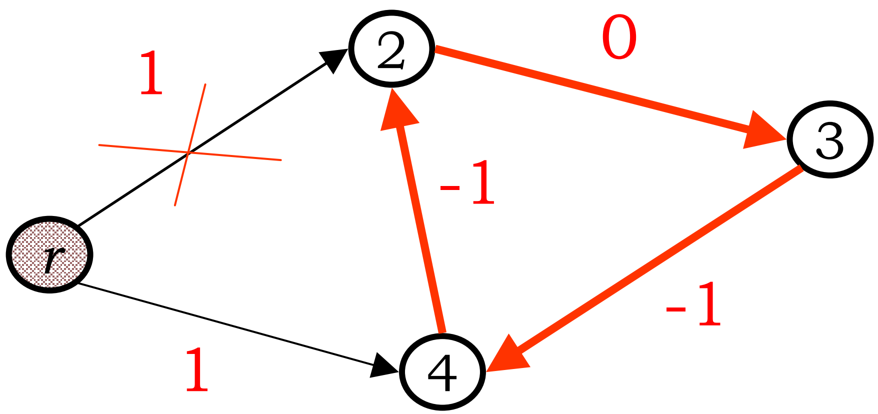
Iter 3. arco scelto (3,4)



	y	p
r= 1	0	0
2	1	r
3	1	2
4	0	3

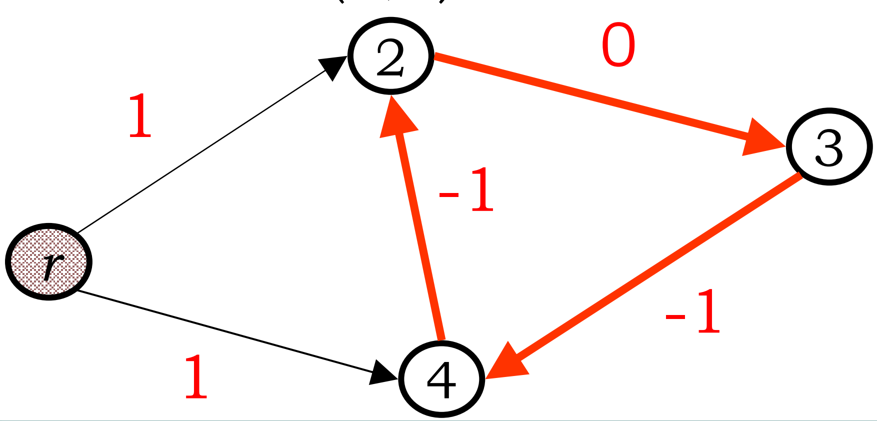
Esempio

Iter 4. arco scelto $(4,2)$



	y	p
r= 1	0	0
2	-1	4
3	1	2
4	0	3

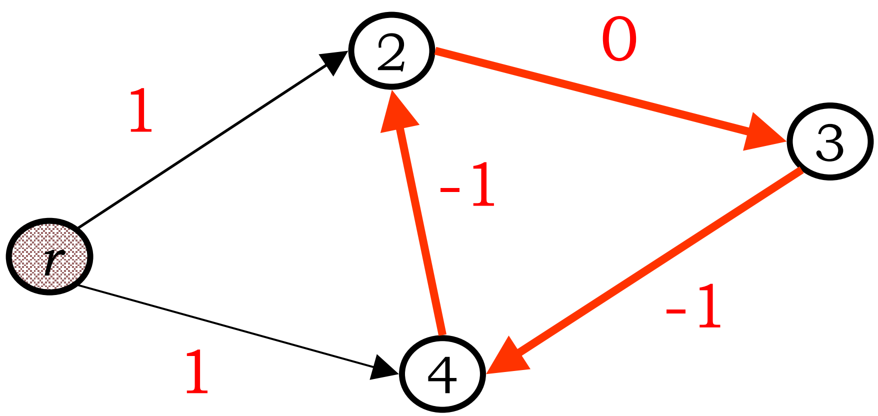
Iter 5. arco scelto $(2,3)$



	y	p
r= 1	0	0
2	-1	4
3	-1	2
4	0	3

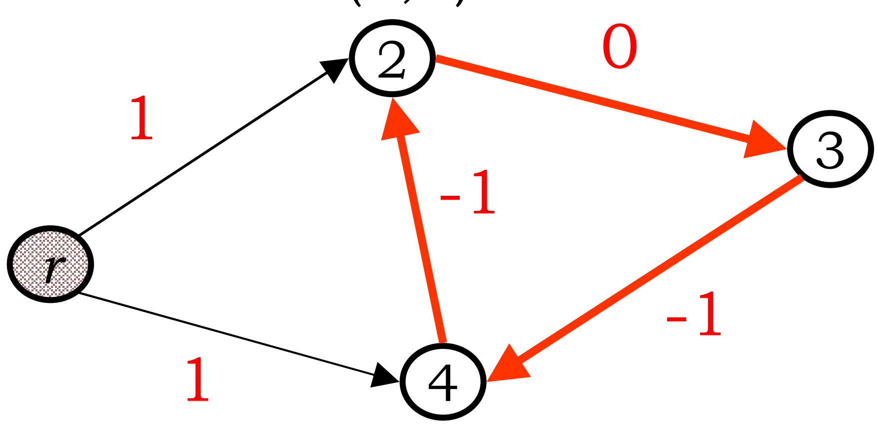
Esempio

Iter 6. arco scelto (3,4)



	y	p
r= 1	0	0
2	-1	4
3	-1	2
4	-2	3

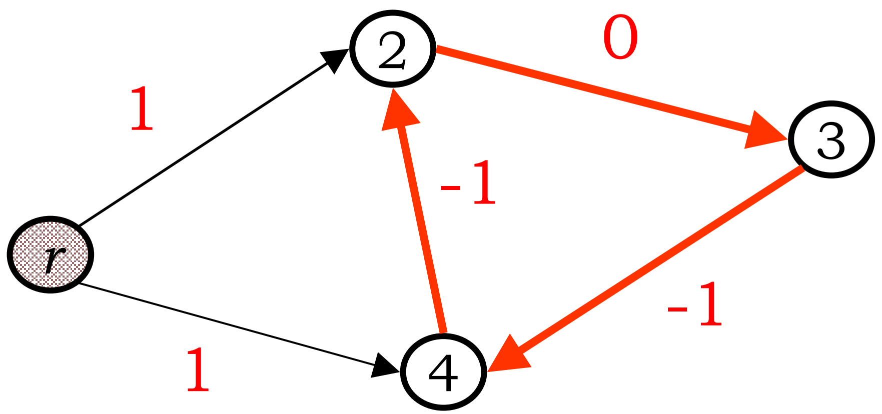
Iter 7. arco scelto (4,2)



	y	p
r= 1	0	0
2	-3	4
3	-1	2
4	-2	3

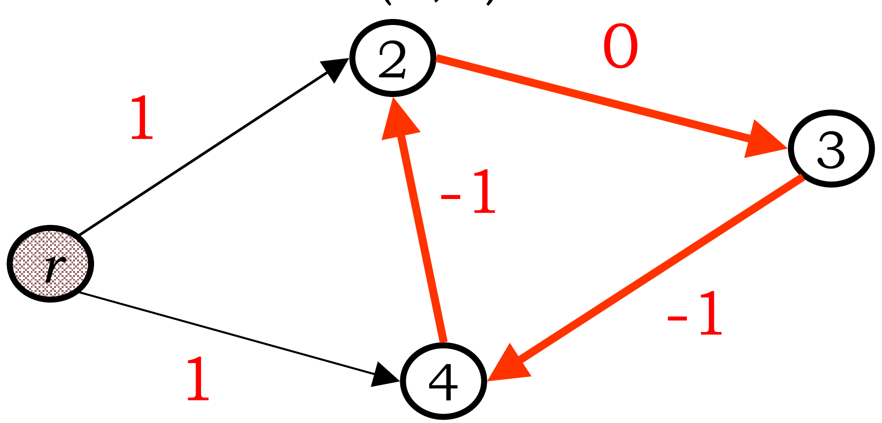
Esempio

Iter 8. arco scelto (2,3)



	y	p
r= 1	0	0
2	-3	4
3	-3	2
4	-2	3

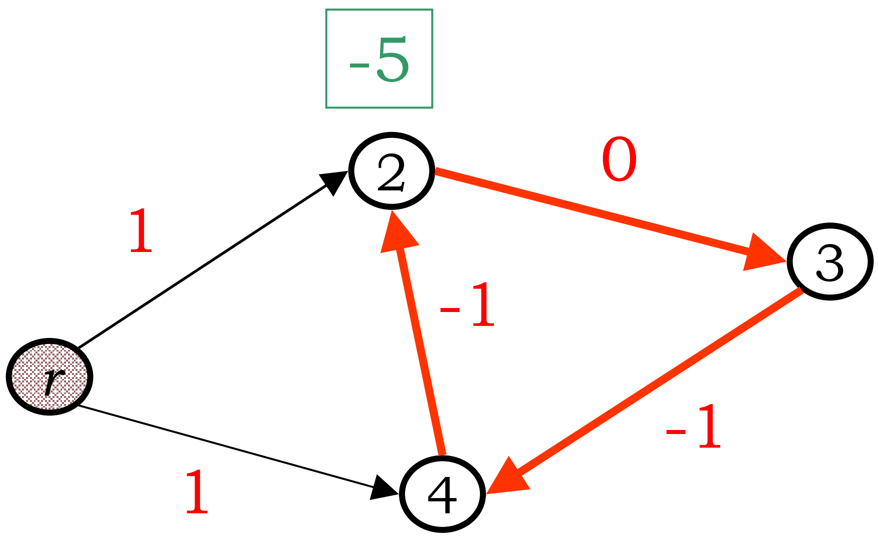
Iter 9. arco scelto (3,4)



	y	p
r= 1	0	0
2	-3	4
3	-3	2
4	-4	3

Esempio

Iter 10. arco scelto (4,2)



	y	p
r= 1	0	0
2	-5	4
3	-3	2
4	-2	3

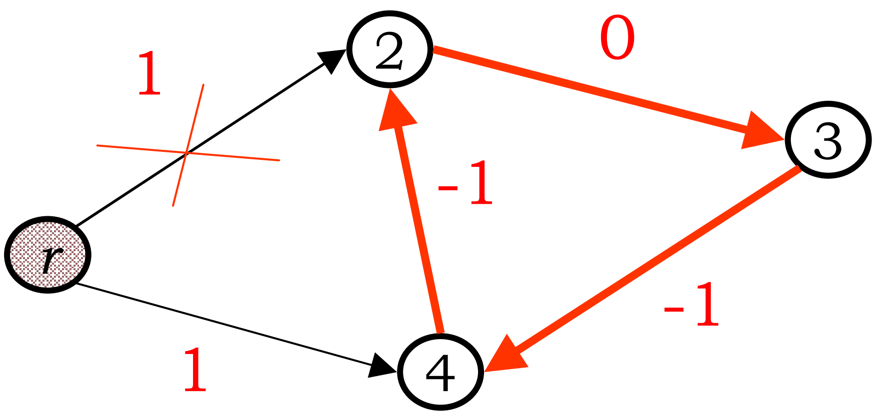
$y_2 = -5 < -nL$ e G_p restituisce il ciclo negativo.

Criterio 2

Il numero di iterazioni dipende da L, quindi la convergenza del criterio 1 è in genere molto lenta.

All'iterazione 4, l'arco $(r,2)$ viene sostituito dall'arco $(4,2)$ generando un ciclo in G_p.

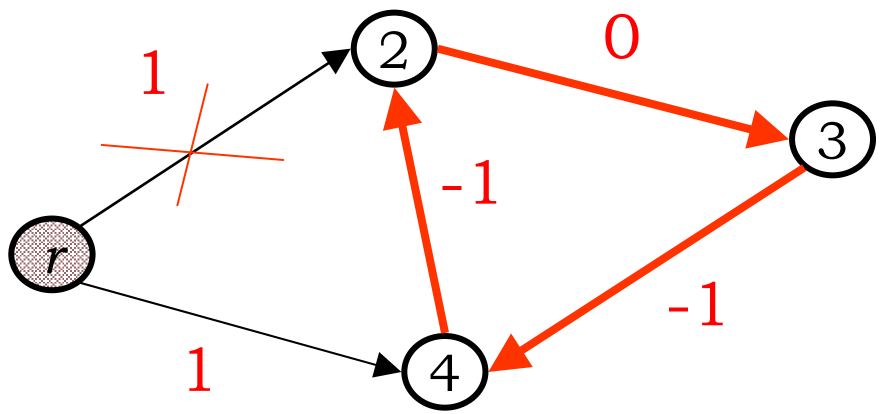
Iter 4. arco scelto $(4,2)$



In generale, questo può accadere se e solo se G contiene un ciclo orientato di lunghezza negativa.

Criterio 2

Quindi, un secondo criterio di individuazione di cicli negativi in G consiste nel verificare (con frequenza fissata) se G_p contiene un ciclo.



Dopo l'aggiornamento di un arco (u,v) [(2,4)] si percorrono i predecessori di u[4]. Se si incoltra v[2] allora G_p contiene un ciclo.

Analisi

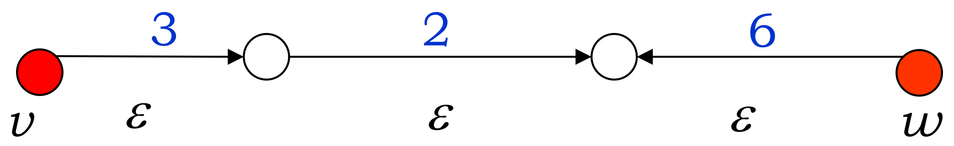
Il secondo criterio è, in generale, molto più efficiente del primo. Nell'esempio, un ciclo in G_p appare alla iterazione 4, mentre il criterio 1 non individua il ciclo negativo prima di 10 iterazioni.

Percorrere a ritroso i predecessori di u ha complessità lineare.

La complessità asintotica dell'algoritmo di Ford non cambia se ad ogni passo, si aggiunge la visita a ritroso dei predecessori del nodo u.

Circuiti incrementanti di costo negativo

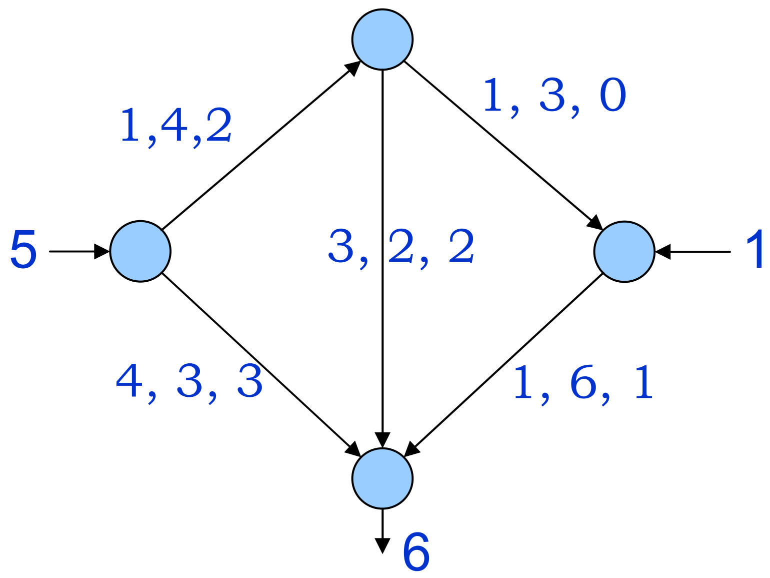
Consideriamo in G un cammino (non necessariamente orientato) incrementante:

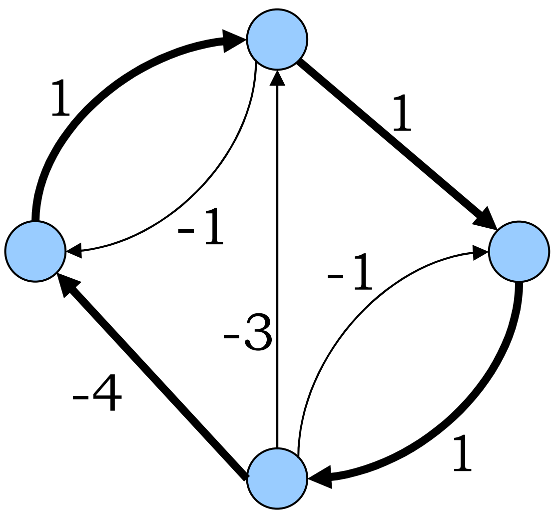


Se aumentiamo il flusso di ε lungo il cammino (v, w) il costo varia del valore $3\varepsilon + 2\varepsilon - 6\varepsilon$. Il valore (3+2-6) = -1 si dice costo del cammino.

Se esiste in G un circuito x-incrementante di costo negativo posso, incrementando di ε il flusso lungo il circuito, diminuire il valore della soluzione corrente.

Esempio





Notazione: (c_{ij}, u_{ij}, x_{ij})

Grafo ausiliario $G(x)$

Un circuito x-incrementante di costo negativo corrisponde ad un circuito orientato di peso negativo sul grafo ausiliario

Teorema (ottimalità per assenza di circuiti di costo negativo)

Una soluzione ammissibile x è ottima se e solo se non esistono circuiti x-incrementanti di costo negativo.

Dimostrazione

Se $G(x)$ ammette un ciclo negativo, ovviamente x non è ottima.

Dimostriamo il viceversa. Sia $G'(x)$ il grafo ottenuto aggiungendo a $G(x)$ un nodo r e collegando r a tutti i nodi in $i \in N$, con costo $c'_{ri} = 0$. Cerchiamo un cammino minimo da r a tutti gli altri nodi. Se non esiste un ciclo negativo in $G'(x)$, l'algoritmo di Ford ritorna un potenziale che soddisfa il seguente sistema

Dimostrazione

di disequazioni

$$y'_j - y_i' \leq c'_{ij} \qquad \forall (i, j) \in A(G'(x))$$

Ora, ponendo $y_i = -y'_i$ e ricordando che se $x_{ij} = u_{ij}$ si ha $c'_{ji} = -c_{ij}$ si ottiene:

$$c_{ij} - y_i + y_j \geq 0 \text{ se } x_{ij} \geq 0$$

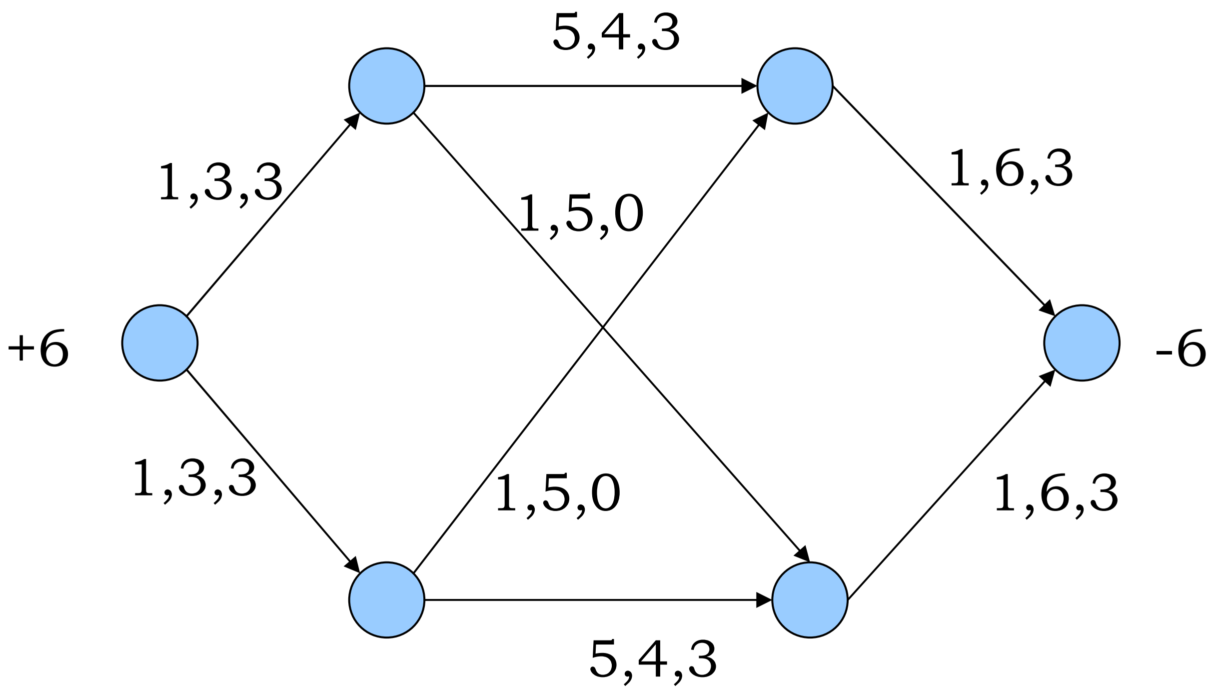
$$c_{ij} - y_i + y_j \leq 0 \text{ se } x_{ij} = u_{ij}$$

Dalle condizioni di ottimalità segue la tesi.

□

Esempio

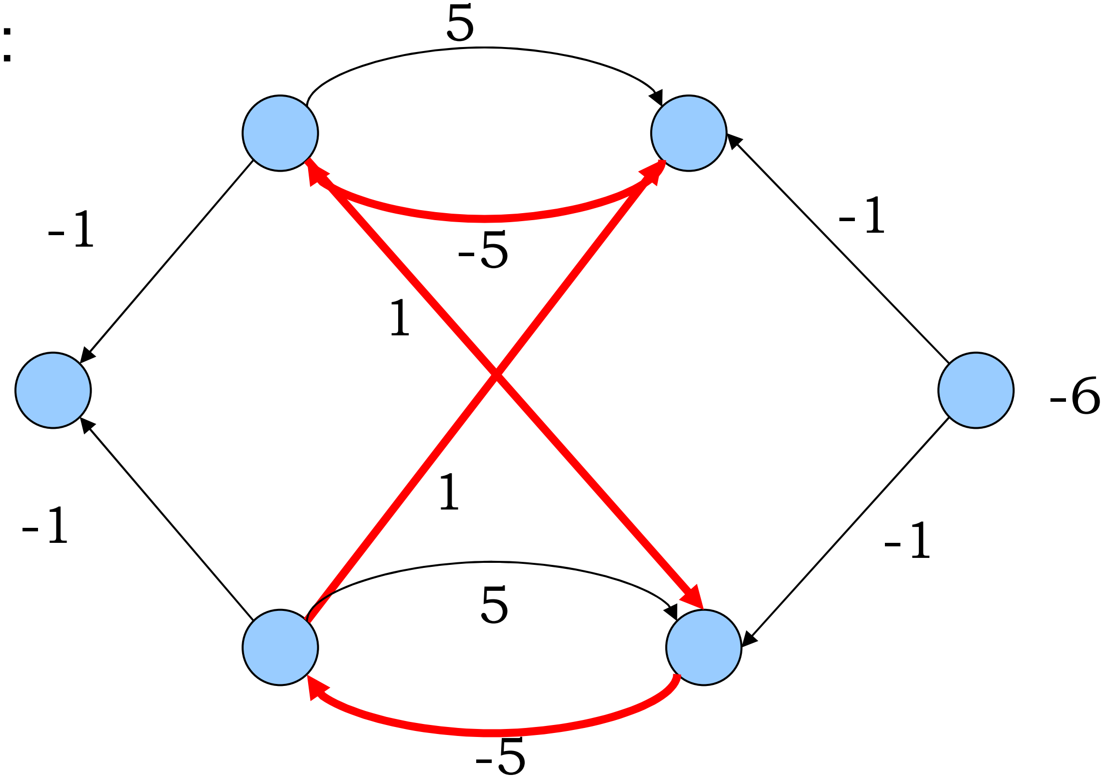
Trovare il flusso di costo minimo sul grafo di figura a partire dalla soluzione ammissibile riportata (di valore 42)



Notazione: $(c_{ij},\ u_{ij},\ x_{ij})$

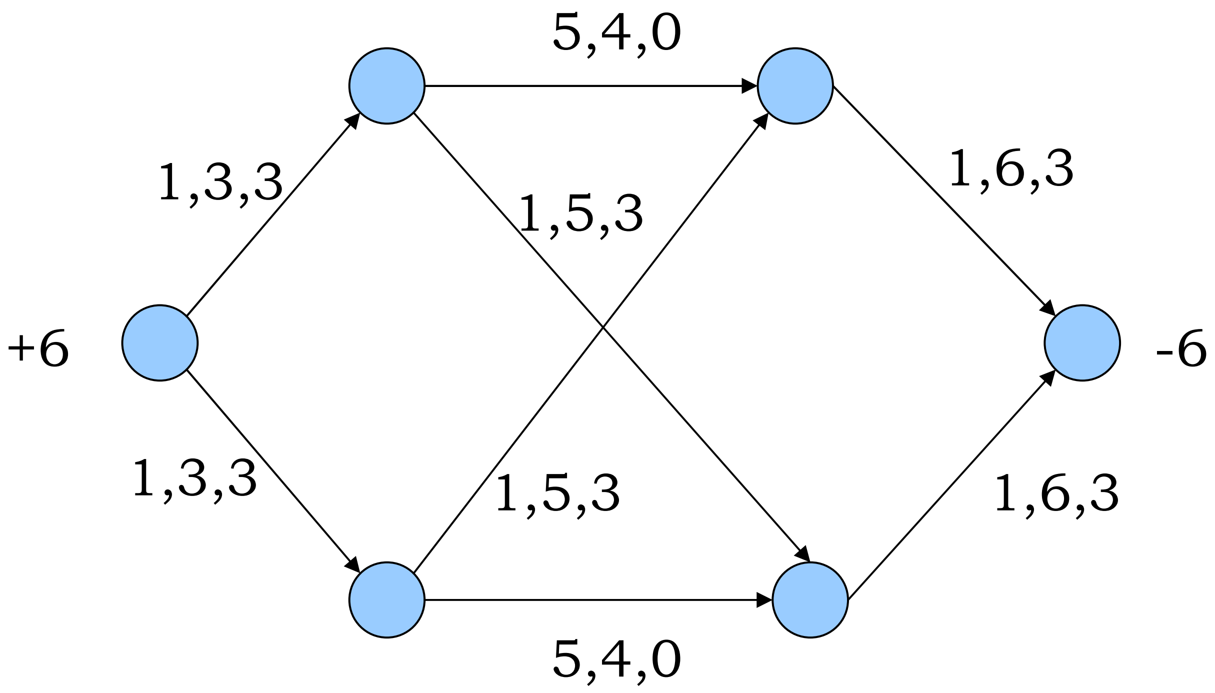
Esempio

Il grafo ausiliario $G'(x)$ contiene un ciclo negativo di valore -8:



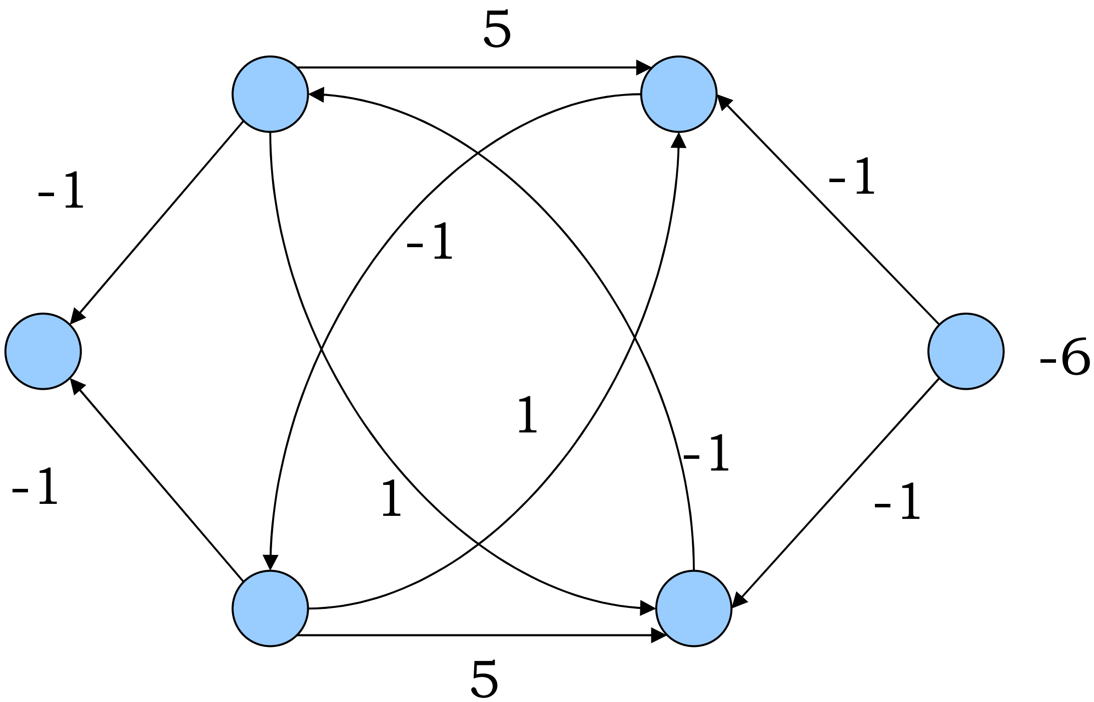
Esempio

Posso variare di 3 unità il flusso lungo questo ciclo, ottenendo la soluzione seguente, di valore 18



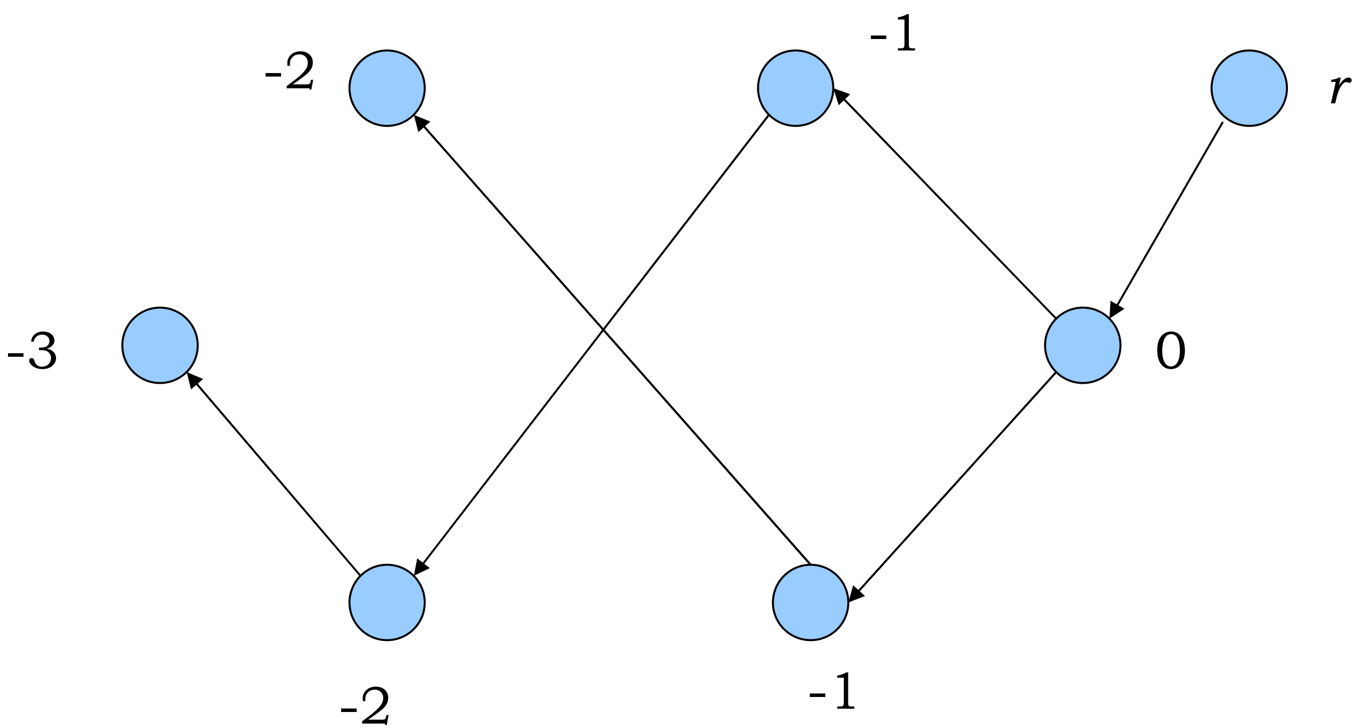
Esempio

Il grafo ausiliario diventa:



Questo grafo ammette il seguente potenziale:

Esempio



Pertanto, per il teorema precedente la soluzione di valore 18 è ottima

Un'applicazione del min cost flow

Consideriamo il seguente problema di PL:

$$\min cx$$

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix} x \geq \begin{bmatrix} 4 \\ 10 \\ 14 \\ 7 \end{bmatrix}$$

$$x \geq 0$$

La matrice dei A dei coefficienti ha le seguenti proprietà:

1. I coefficienti sono 0 o 1
2. Gli 1 appaiono sulle colonne in modo consecutivo (consecutive 1's property)

Osservazione la proprietà 2 va verificata permutando eventualmente righe e colonne

Consecutive 1's property

Portiamo il problema in forma standard e introduciamo una riga ridondante $0x+0y=0$

$$\min cx$$

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 14 \\ 7 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 \end{bmatrix}$$

$$x \geq 0$$

A questo punto, sottraiamo la riga i dalla riga $i+1$, per $i = m, m-1, \ldots, 1$, ottenendo il nuovo problema

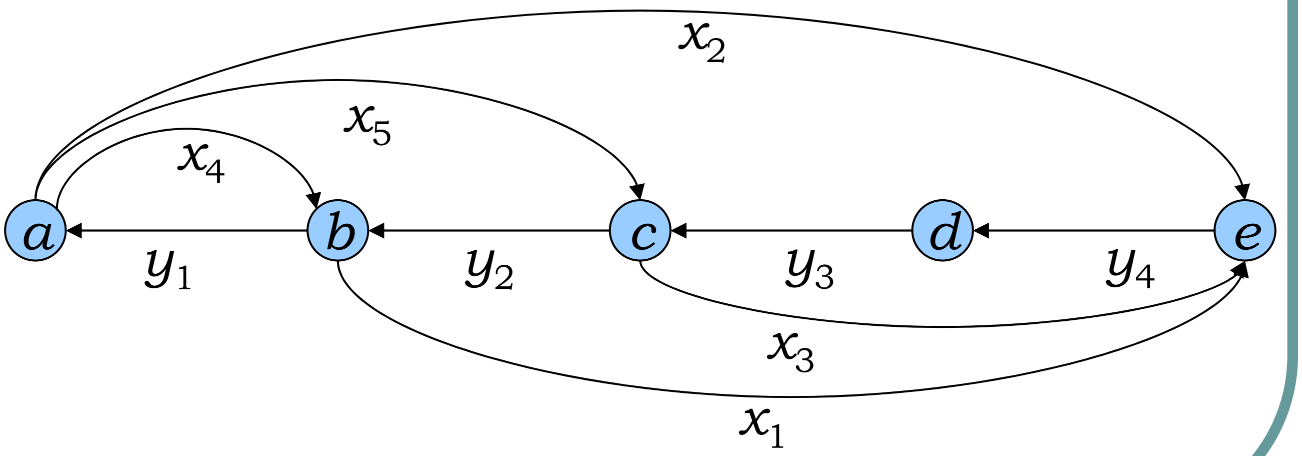
Consecutive 1's property

$$\min cx$$

$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array}\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \\ 4 \\ -7 \\ -7 \end{bmatrix}$$

$$x \geq 0$$

Risolvere il problema di PL equivale a determinare il flusso a costo minimo sul grafo di figura



Simplesso su reti

Consideriamo il problema di flusso a costo minimo

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = b(i) \qquad \forall i \in N \qquad \text{(P)}$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in A$$

su $G = (N, A)$, connesso.

Soluzioni albero

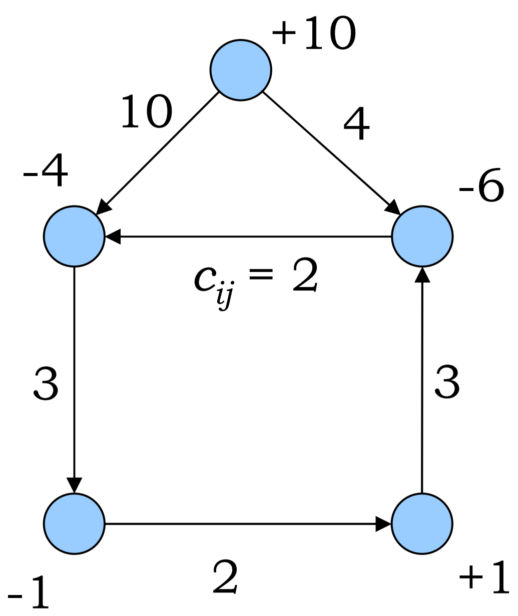
Poiché G è connesso, ammette un albero ricoprente T

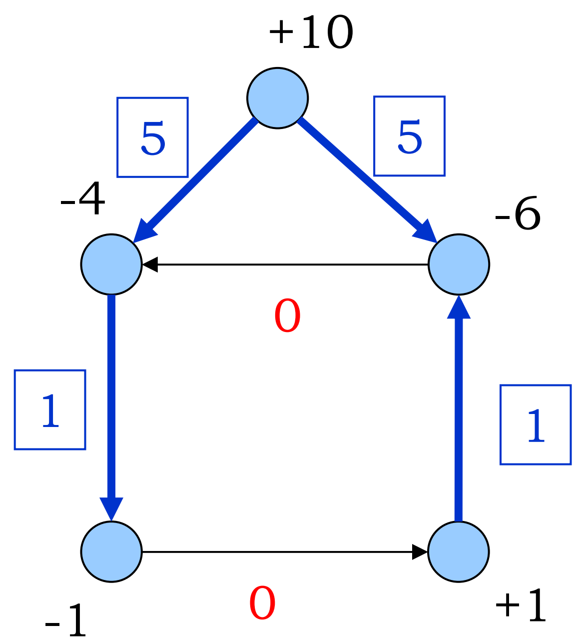
Definizione

Una soluzione ammissibile x per P si dice soluzione albero se, dato un albero ricoprente T di G si ha:

$$\begin{cases} \sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = b(i) & \forall i \in N \\ x_{ij} = 0 & \forall (i,j) \notin T \end{cases}$$

Esempio





Il grafo di sinistra ha i costi sugli archi. Invece, gli archi blu sul grafo di destra rappresentano una soluzione albero (i valori sugli archi sono i flussi).

Corrispondenza alberi/soluzioni

Dato un albero T, possiamo scegliere un suo nodo r come radice e riferire T come albero con radice in r.

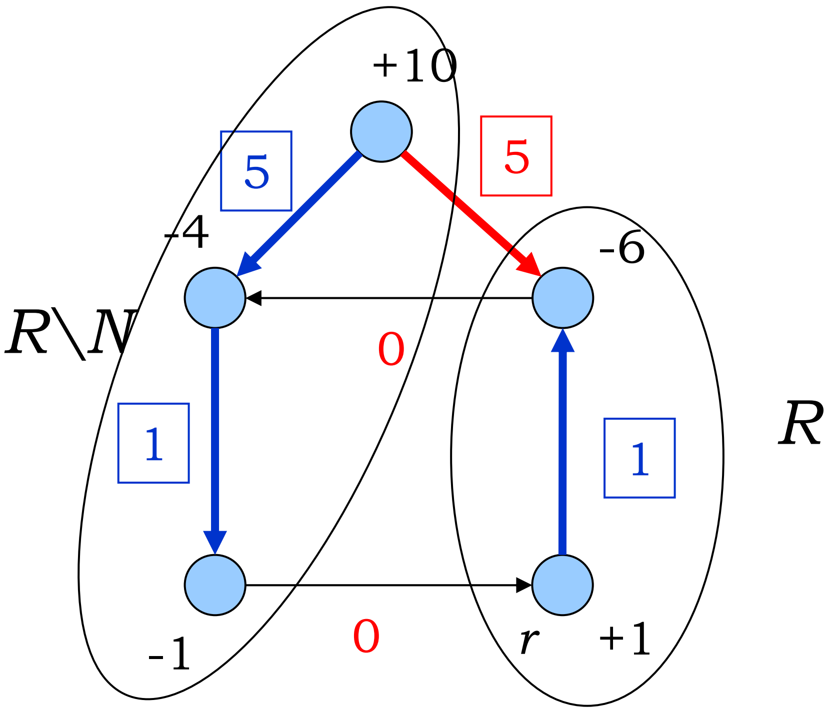
Data una soluzione albero, se T ha radice r e scegliamo un arco (i, j) di T, si determina una partizione dei nodi di G in due insiemi R e $R \backslash N$, tali che il flusso tra R e $R \backslash N$ è supportato totalmente dall'arco (i, j) (nel verso concorde con il verso dell'arco).

In altre parole, ogni arco dell'albero T ha un valore di flusso univocamente determinato. Pertanto:

Lemma

Ad un albero T è associata un'unica soluzione albero.

Esempio



L'arco rosso supporta tutto il flusso da $R \setminus N$ verso R.

Corrispondenza alberi/soluzioni

Osservazione

Ad una soluzione ammissibile, in generale, possono corrispondere diverse soluzioni albero.

Esempio

Sia $b(i) = 0$, per ogni $i \in N$. La soluzione $\boldsymbol{x} = \mathbf{0}$ è ammissibile e ogni albero ricoprente T di G è una soluzione albero associata alla soluzione $\boldsymbol{x} = \mathbf{0}$.

Corrispondenza alberi/soluzioni

Teorema

Se P ammette una soluzione ammissibile, allora ha una soluzione albero ammissibile.

Se P ha soluzione ottima, allora ha una soluzione albero ottima

Dimostrazione

Parte 1

Sia $\boldsymbol{x}$ una soluzione ammissibile per P. Se $\boldsymbol{x}$ non è una soluzione albero, significa che esiste almeno un ciclo C con la proprietà che $x_{ij} > 0$ per ogni $(i, j) \in C$. C può essere sempre orientato in modo che contenga almeno un arco "reverse".

Corrispondenza alberi/soluzioni

Sia $\varepsilon = \min x_{ij}$ tale che (i,j) è un arco "reverse" di C. La soluzione $\boldsymbol{x}'$ ottenuta da $\boldsymbol{x}$ con la seguente tecnica:

$x'_{ij} = x_{ij} + \varepsilon$ se (i,j) è un arco forward di C

$x'_{ij} = x_{ij} - \varepsilon$ se (i,j) è un arco reverse di C

è ammissibile e non contiene più C. Pongo $\boldsymbol{x} = \boldsymbol{x}'$ e reitero finché $\boldsymbol{x}$ è priva di cicli.

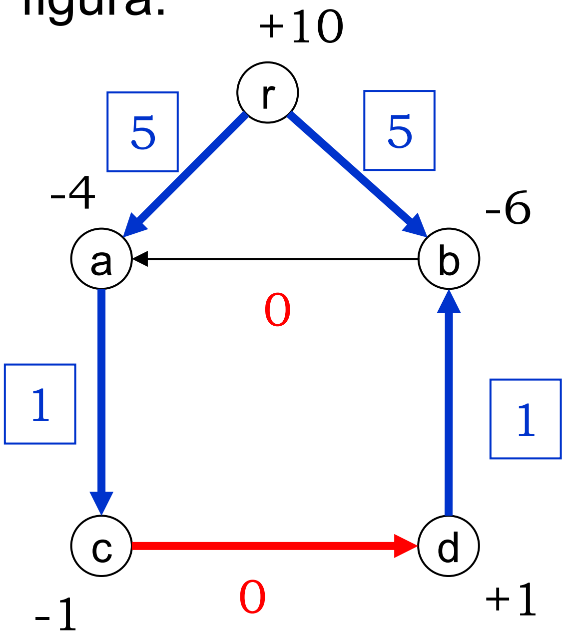
Parte 2

Sia x^* una soluzione ottima. Se x^* contiene un circuito, questo ha costo zero. Difatti, x^* non può contenere circuiti C di costo negativo (altrimenti x^* non sarebbe ottima). Potendo scegliere l'orientazione di C, significa che se x^* contiene cicli, questi hanno costo zero. È sufficiente, quindi, applicare la tecnica precedente per ottenere una soluzione ottima albero. □

Conseguenza

Il teorema precedente ci consente di poter limitare la ricerca della soluzione ottima alle sole soluzioni albero.

Una opportuna tecnica consente di passare da una soluzione albero ad una nuova soluzione albero. Consideriamo la soluzione di figura:



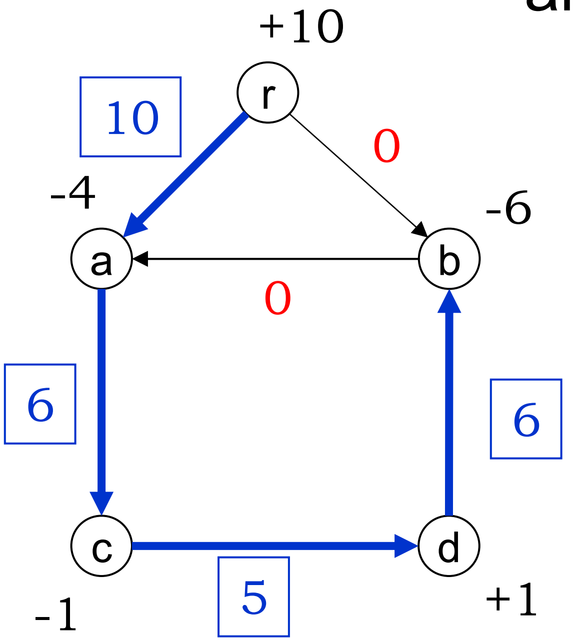
se aggiungiamo all'albero in blu l'arco (c, d), otteniamo un ciclo C.

Ora, scegliendo come orientazione del ciclo quella dell'arco (c, d), posso inviare 5 unità di flusso lungo l'arco (c, d), svuotando l'arco (r, b) e ottenendo una nuova soluzione albero.

Conseguenza

Il valore del flusso inviato lungo il nuovo arco è pari al minimo del flusso sugli archi “reverse” del ciclo *C*.

La nuova soluzione albero è, per costruzione, ammissibile.



Potenziali di un nodo

Data una soluzione albero T con radice in r, definisco potenziale di un nodo la seguente quantità:

$$\begin{cases} y_r = 0 \\ y_h = -\sum_{(i,j)\text{ arco forward di } P} c_{ij} + \sum_{(j,i)\text{ arco reverse di } P} c_{ji} \quad \forall h \neq r \end{cases}$$

ove P è il cammino (unico) in T da r ad h.

Osservazione

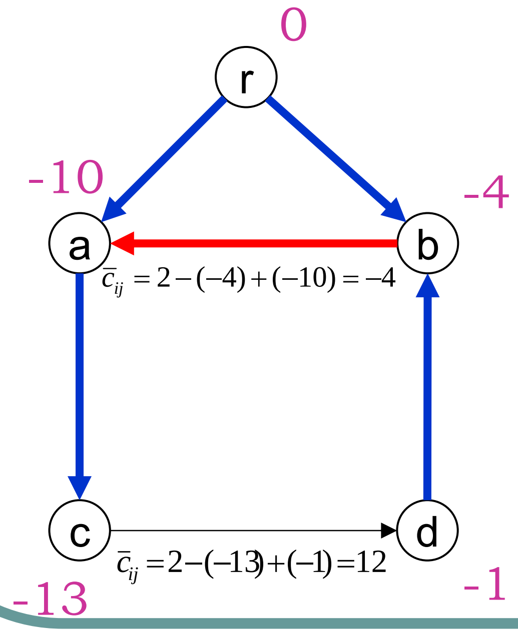
Il costo ridotto è stato definito come $\bar{c}_{ij} = c_{ij} - y_i + y_j$

Dalla definizione segue che il costo ridotto di un arco di T è pari a zero.

Esempio

Dalle condizioni di ottimalità per il min cost flow, sappiamo che se il costo ridotto è maggiore o uguale a zero per ogni arco del grafo $\notin T$, siamo in presenza di una soluzione ottima.

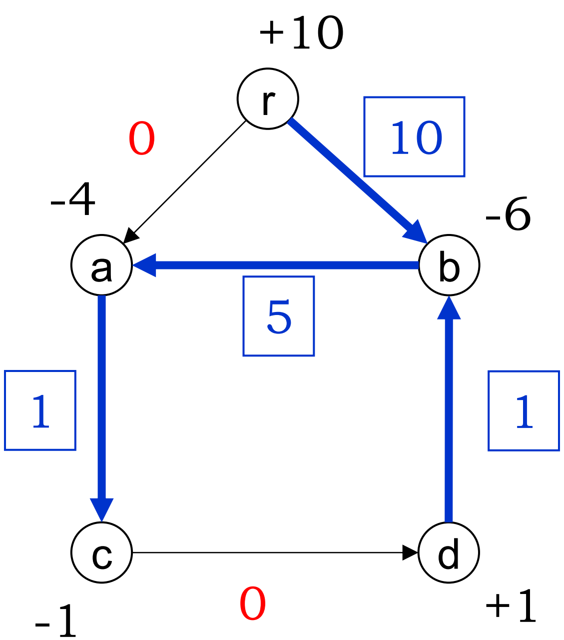
Se la soluzione non è ottima, devo selezionare come arco da far entrare nella soluzione albero un arco con costo ridotto negativo.

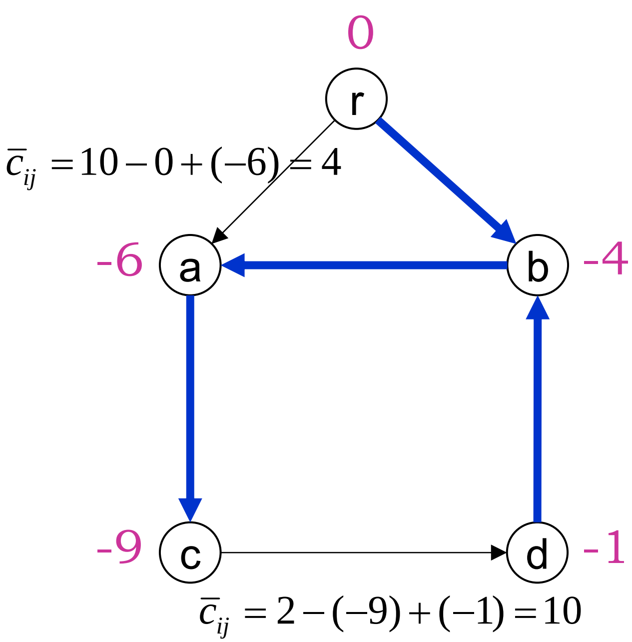


In questa soluzione, di valore 5*5 + 4*10 + 1*3 + 1*3 = 71, l'arco selezionato è l'arco (b, a).

Esempio

La nuova soluzione vale 10*4 + 5*2 + 1* 3 + 1*3 = 56. I potenziali ai nodi valgono:





Poiché la nuova soluzione ha costi ridotti positivi per gli archi $\notin T$, essa è ottima.

Algoritmo del simplesso su reti

```
Trova una soluzione albero ammissibile, T ;

Calcola i potenziali dei nodi y ;

while esiste un arco (i, j) ∉ T tale che c_ij – y_i + y_j < 0 {
   sia C il ciclo (orientato secondo (i, j)) che si ottiene
   aggiungendo (i, j) a T;
   if (C non contiene archi reverse) then stop;
   else {
        ε = min {x_hk, t.c. (h, k) è un arco reverse di C};
        scegli un arco reverse (l, m) di C t.c. x_lm = ε;
        aumenta x di ε lungo il ciclo C ;
        T = T ∪ (i, j)\(l, m);
        aggiorna y ;
        }
}
```

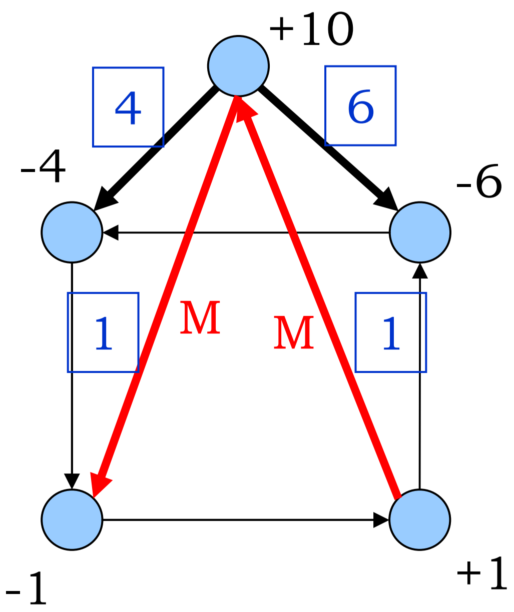
Osservazioni

Come si determina una soluzione albero iniziale?

Metodo 1 (o degli archi artificiali)

Scelgo un nodo r. Per ogni nodo $i \neq r$, se $b(i) \geq 0$, inserisco in T l'arco (i, r), se $b(i) < 0$ inserisco in T l'arco (r, i). Se qualcuno di questi archi non esiste in G, li aggiungo al grafo con un costo M.

Se la soluzione ottima non contiene archi di costo M, allora il problema originario era ammissibile.

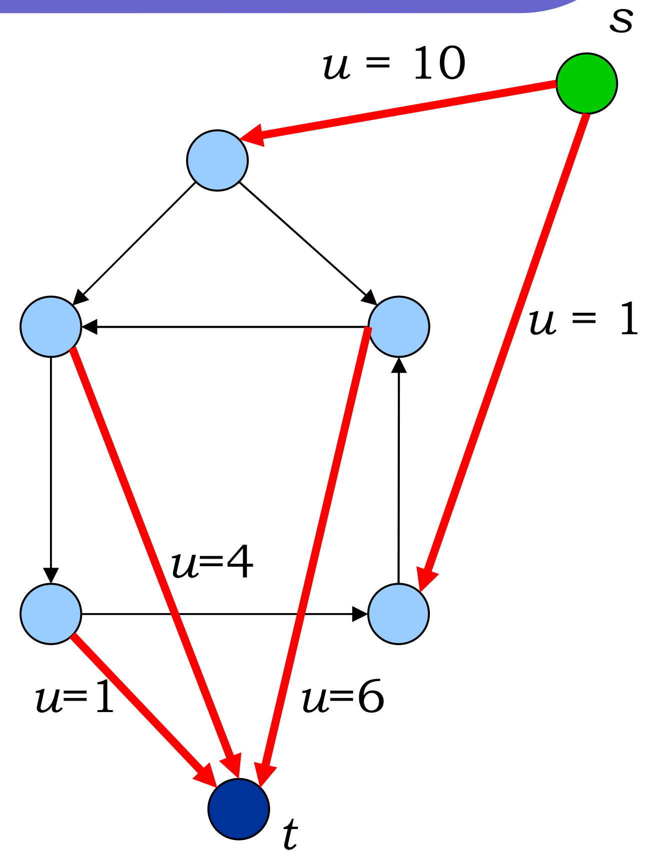


Osservazioni

Metodo 2

Aggiungo due nodi s e t e li collego, rispettivamente ai nodi con $b(i)>0$ e ai nodi con $b(k)<0$. La capacità di un arco (s, i) è pari a $b(i)$, la capacità di un arco (k, t) è pari a $b(k)$.

Quindi, calcolo l's-t flusso massimo. Se il flusso satura tutti gli archi aggiunti, il problema originario è ammissibile.



Terminazione dell'algoritmo

Per quanto riguarda la convergenza in un numero finito di iterazioni del metodo, si rimanda al paragrafo del libro di testo (pagg. 108/109)

Simplesso su reti

Come si adatta l'algoritmo precedente quando reintroduciamo le capacità sugli archi?

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = b(i) \qquad \forall i \in N \qquad \text{(P)}$$

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

Soluzione albero

Definizione

Una soluzione ammissibile x per P si dice soluzione albero se, dato un albero ricoprente T di G e una partizione L, U degli archi di G non appartenenti a T, si ha:

$$\begin{cases} \displaystyle\sum_{(i,j)\in\delta^+(i)} x_{ij} - \sum_{(j,i)\in\delta^-(i)} x_{ji} = b(i) & \forall i \in N \\ x_{ij} = 0 & \forall (i,j) \in \mathrm{L} \\ x_{ij} = u_{ij} & \forall (i,j) \in \mathrm{U} \end{cases}$$

Una soluzione albero si indica con (T, L, U).

Simplesso su reti

Anche in questo caso si può dimostrare il seguente

Teorema

Se P ammette una soluzione ammissibile, allora ha una soluzione albero ammissibile.

Se P ha soluzione ottima, allora ha una soluzione albero ottima □

Pertanto, ci si può limitare a esplorare soluzioni albero (T, L, U).

Simplesso su reti

Osservazioni

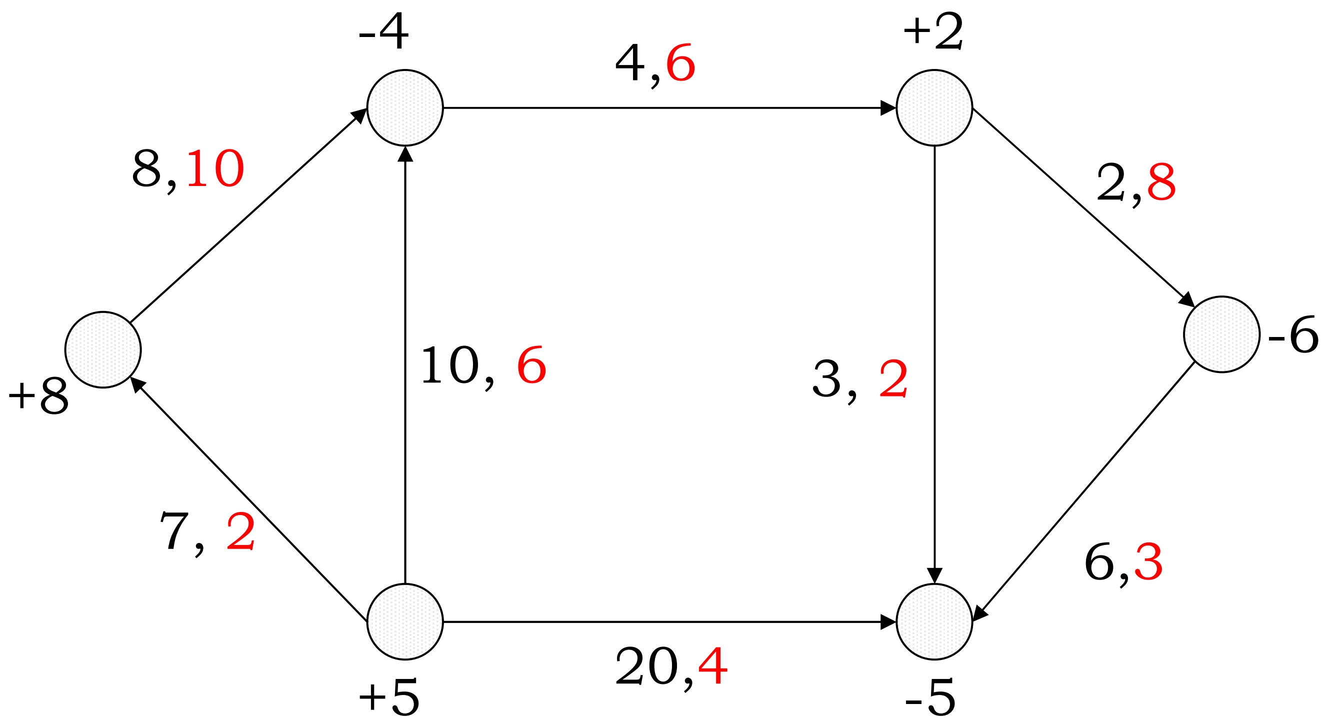
1. La presenza di capacità sugli archi modifica la modalità di aggiornamento della soluzione albero
2. Per verificare l'ottimalità della soluzione bisogna considerare le condizioni di ottimalità anche per gli archi in U

L'algoritmo modificato è il seguente:

Simplesso su reti

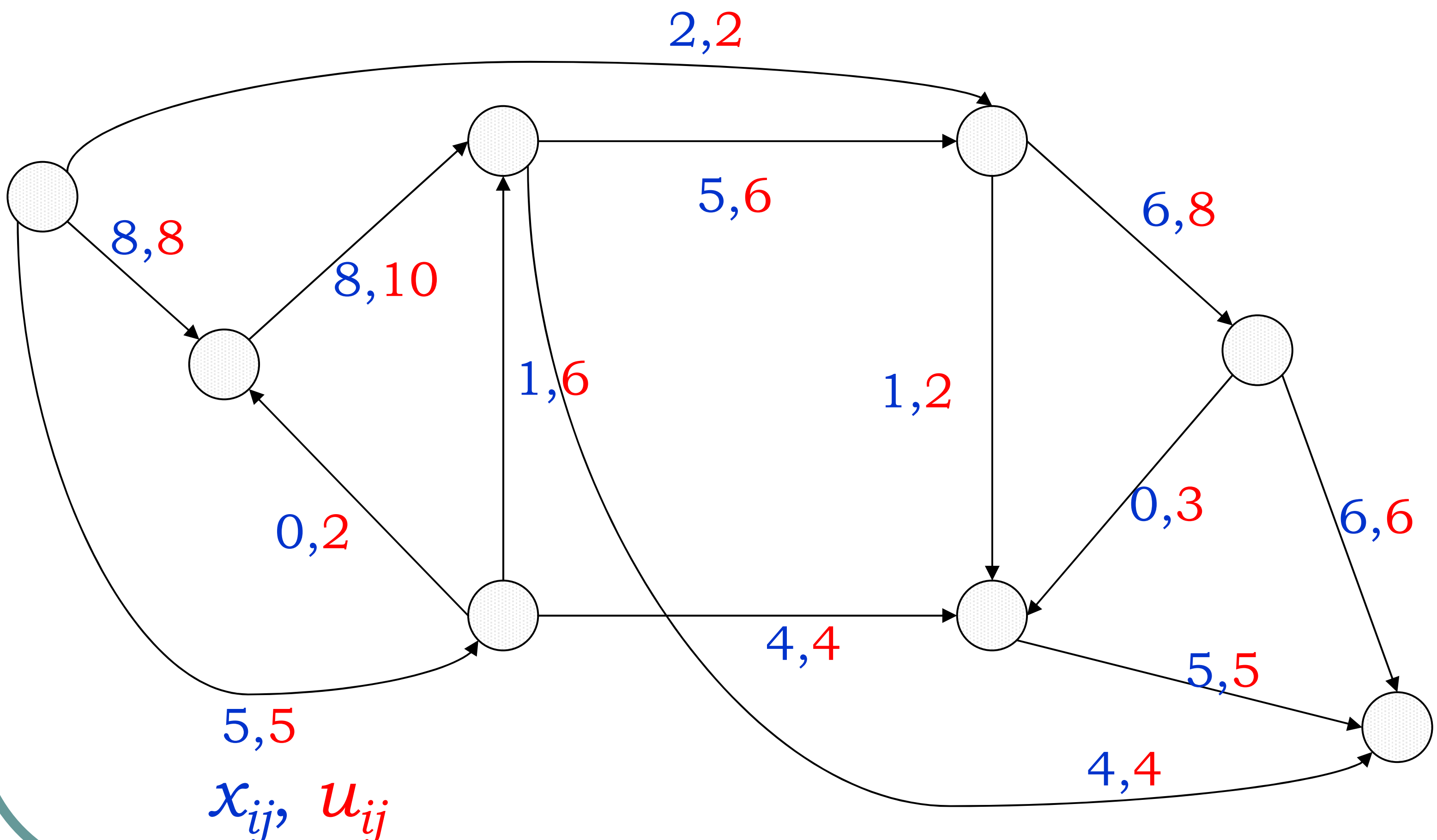
```
Trova una soluzione albero ammissibile, (T, L, U);
Calcola i potenziali dei nodi y;
while esiste un arco (i, j) ∈ L tale che c_ij − y_i + y_j < 0 or
esiste un arco (i, j) ∈ U tale che c_ij − y_i + y_j > 0 {
        sia C il ciclo orientato in cui (i, j) è forward se (i, j) ∈ L oppure è
        reverse se (i, j) ∈ U;
        if (C non contiene archi reverse or C non contiene archi forward
                di capacità finita) then stop;
        else {ε1 = min {x_hk, t.c. (h, k) è un arco reverse di C};
              ε2 = min {u_hk - x_hk, t.c. (h, k) è un arco forward di C};
              ε = min {ε1, ε2};
              scegli un arco reverse (l, m) di C t.c. x_lm = ε
                oppure un arco forward (l, m) di C t.c. u_lm - x_lm = ε;
                aumenta x di ε lungo il ciclo C;
                T = T ∪ (i, j)\(l, m);
                aggiorna y;
                }
}
```

Esempio



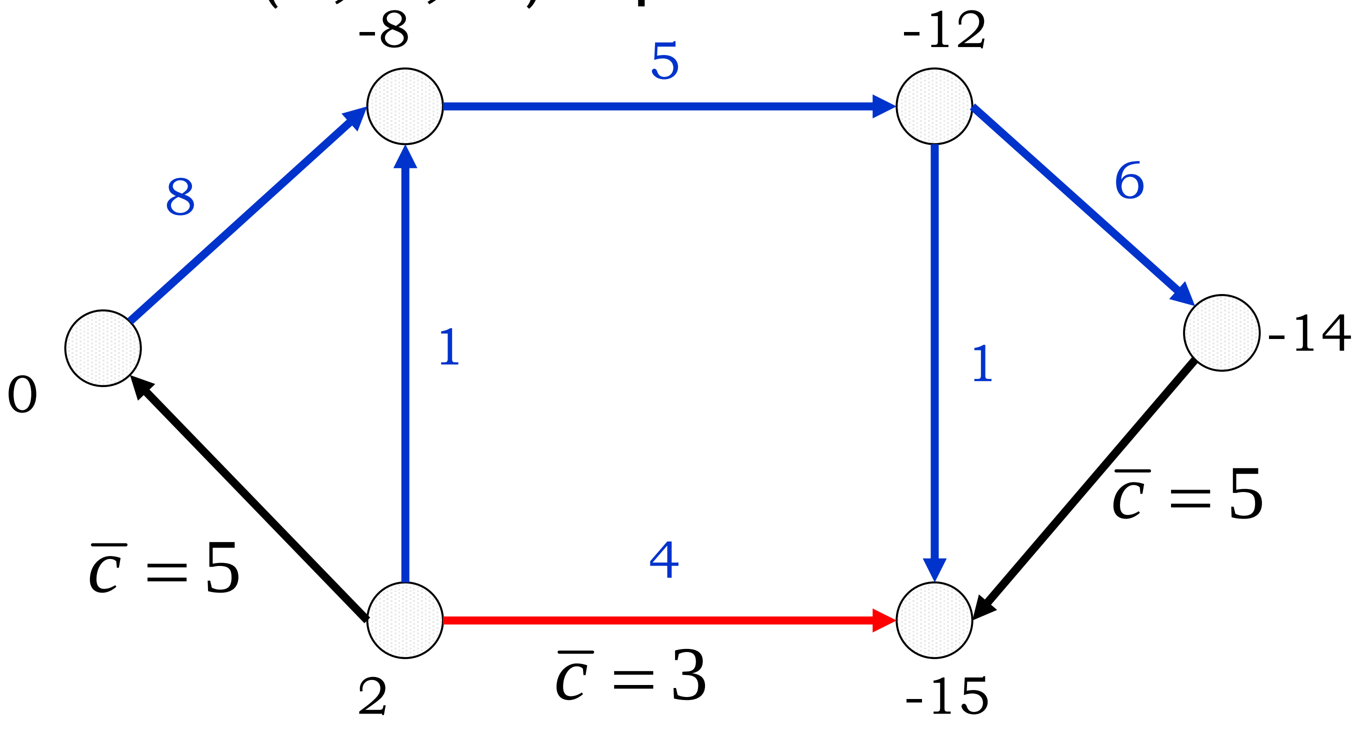
c_{ij}, u_{ij}

Esempio



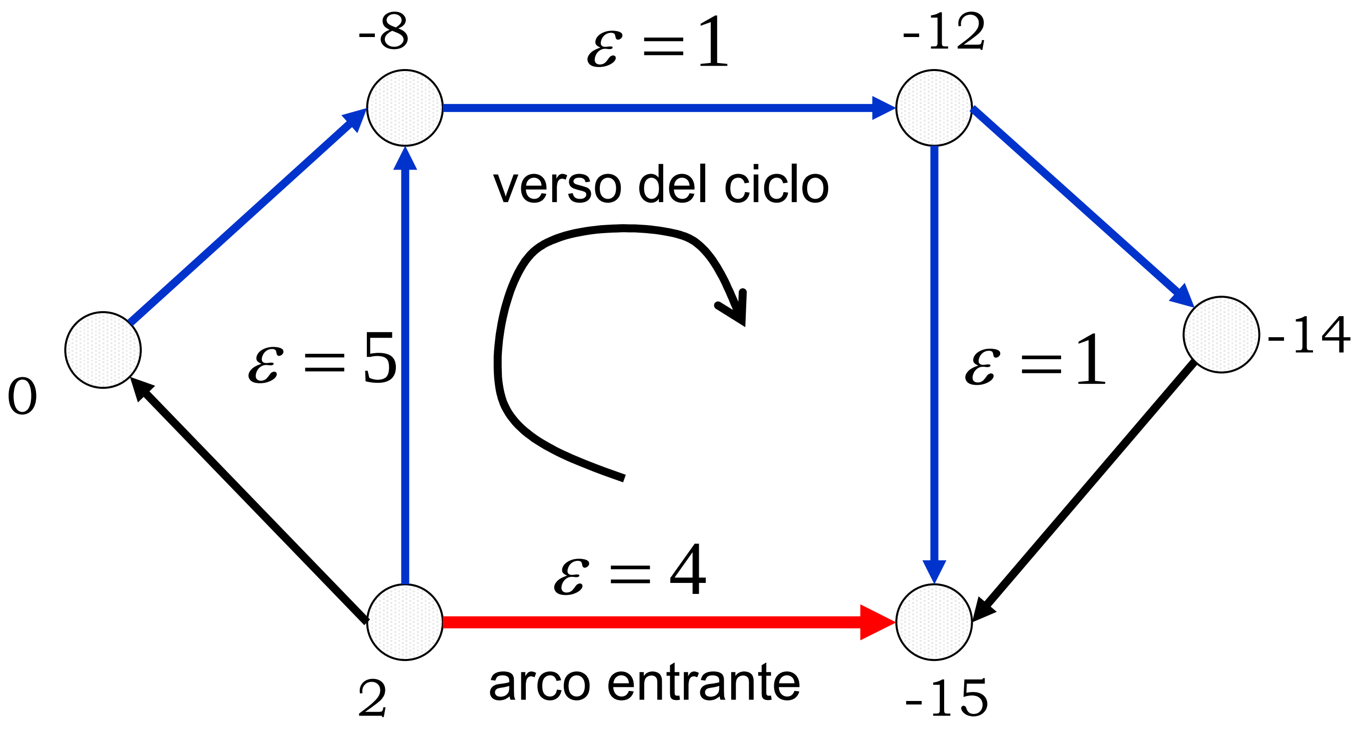
Esempio

Struttura (***T***, ***L***, ***U***) e potenziali ai nodi

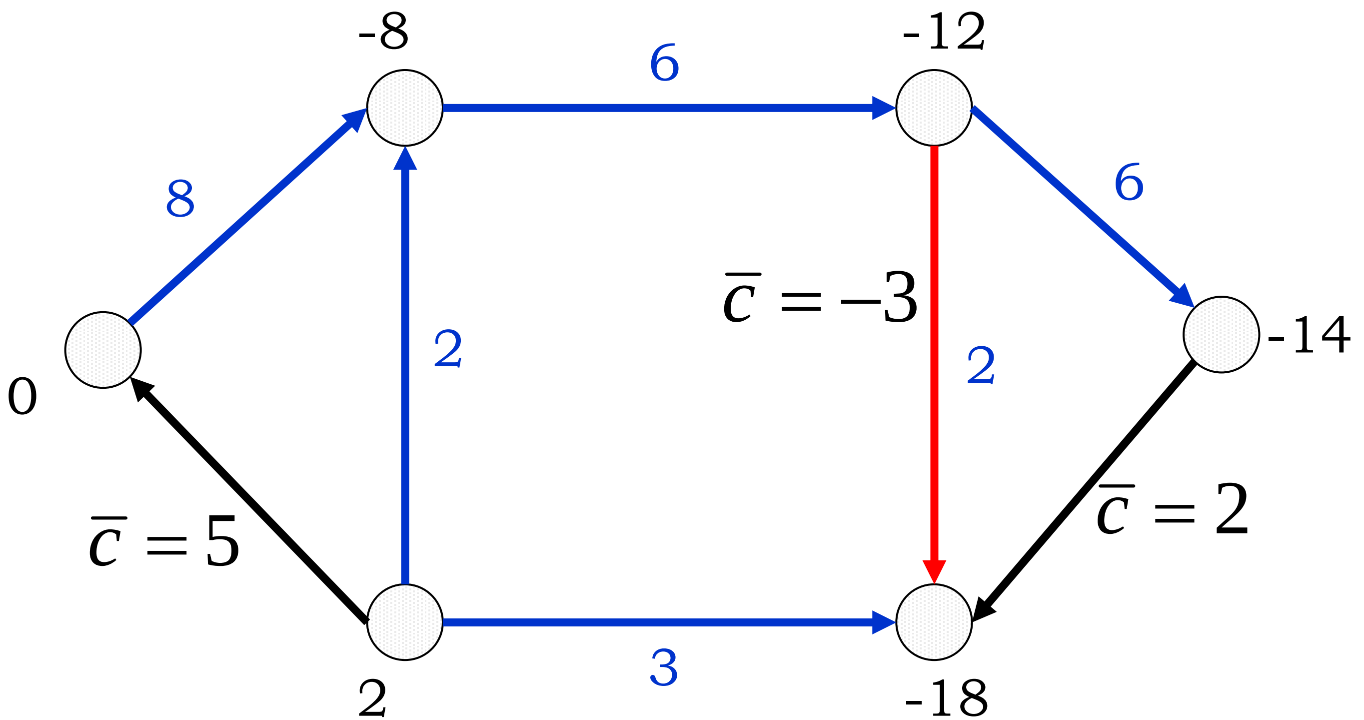


Valore della soluzione: 189

Esempio


-8
$\varepsilon = 1$
-12
verso del ciclo
0
$\varepsilon = 5$
$\varepsilon = 1$
-14
$\varepsilon = 4$
2
arco entrante
-15

Esempio



Soluzione ottima di valore 166