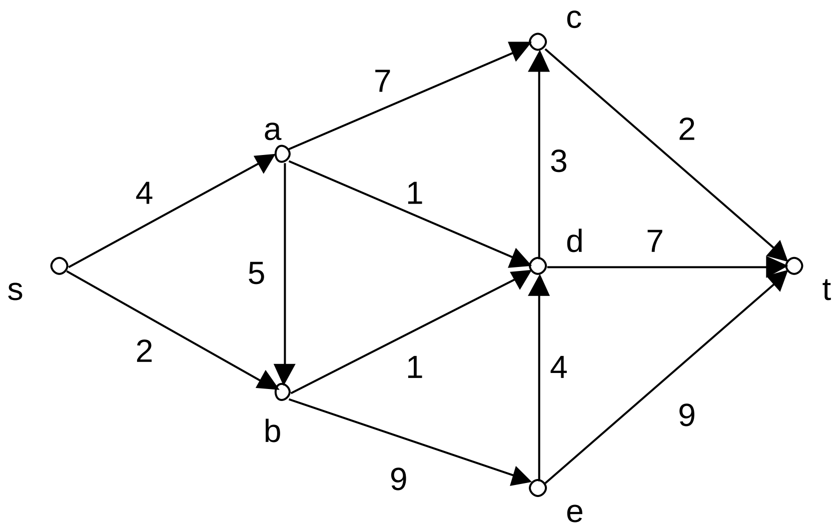
Esercizio su cammino di costo minimo

Calcolare mediante il metodo di Djikstra un cammino da s a t di costo minimo nel seguente grafo orientato, in cui per ogni arco (i, j) è indicato il costo associato q_{ij}.



<u>*Riferimento*</u>: M. Fischetti, "Lezioni di Ricerca Operativa" pag. 139-144, oppure
R. Baldacci, M. Dell'Amico, "Fondamenti di Ricerca Operativa" pag. 129-132.

Soluzione

Il problema è determinare un cammino di costo minimo da s a t, dove il costo di un cammino è dato dalla somma dei costi associati agli archi del cammino (ad esempio, il cammino $s \rightarrow a \rightarrow d \rightarrow t$ ha costo 4 + 1 + 7 = 12). Il metodo di soluzione è basato sul seguente risultato:

Teorema. Sia W un insieme di vertici w per cui è noto il costo L_w di un cammino di costo minimo da s a w (in particolare, $s \in W$). Sia $(w^*, v^*) = \text{argmin} \{ L_w + q_{wv} : w \in W, v \notin W\}$. Allora $L_{w^*} + q_{w^*v^*}$ è il costo di un cammino di costo minimo da s a v^* (cioè v^* può essere inserito in W). □

Il metodo costruisce iterativamente tale insieme W, fino ad inserire t in W.

<u>Inizializzazione</u>

$W := \{s\}$. Costruisci il vettore z, che ha una componente per ogni vertice v del grafo, con $z(v)$ uguale: a q_{sv} se esiste l'arco sv, a ∞ altrimenti.

s	*a*	*b*	*c*	*d*	*e*	*t*
0	4	2	∞	∞	∞	∞

Passo generico

- Scegli vertice x non in W tale che $z(x)$ è minimo
- Inserisci x in W
- Aggiorna z ponendo $z(v) := \min \{z(v), z(x) + q_{xv}\}$ per ogni vertice v non in W

Passo 1

- Scegli b
- $W := W \cup \{b\}$
- Aggiorna z come di seguito

s	*a*	*b*	*c*	*d*	*e*	*t*
0	4	**2**	∞	3	11	∞

Passo 2

- Scegli d
- $W := W \cup \{d\}$
- Aggiorna z come di seguito

s	*a*	*b*	*c*	*d*	*e*	*t*
0	4	**2**	6	**3**	11	10

Passo 3

- Scegli a
- $W := W \cup \{a\}$
- Aggiorna z come di seguito

s	*a*	*b*	*c*	*d*	*e*	*t*
0	**4**	**2**	6	**3**	11	10

Passo 4

- Scegli c
- $W := W \cup \{c\}$
- Aggiorna z come di seguito

s	a	b	c	d	e	t
0	**4**	**2**	**6**	**3**	11	8

Passo 5

- Scegli t
- $W := W \cup \{t\}$
- Il metodo termina dato che t è stato inserito in W

Il numero $z(t) = 8$ indica il costo di un cammino di costo minimo da s a t. Per individuare tale cammino, si può procedere andando a ritroso da t verso s in questo modo:

$z(t)$ è stato aggiornato l'ultima volta quando c è entrato in W;
$z(c)$ è stato aggiornato l'ultima volta quando d è entrato in W;
$z(d)$ è stato aggiornato l'ultima volta quando b è entrato in W;
$z(b)$ è stato aggiornato l'ultima volta quando s è entrato in W.
Si ottiene il cammino: $s \to b \to d \to c \to t$ (con costo = 8)

Esercizio su programmazione di progetti

Calcolare mediante il metodo PERT il tempo minimo di completamento, le attività critiche e i massimi slittamenti ammessi (per ogni attività) nel seguente progetto:

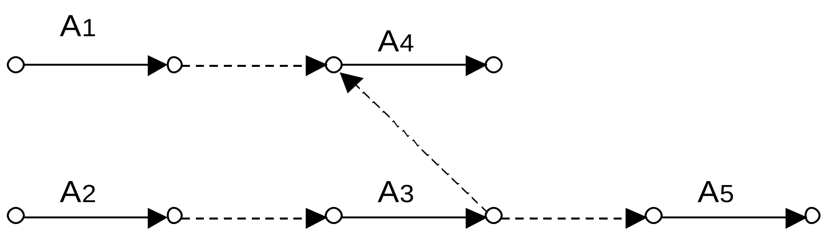
le attività sono 5: $A_1, \ldots, A_5$; ogni attività A_i ha durata d_i, dove $d = [5, 7, 2, 4, 3]$;

le relazioni di precedenza sono: $A_1 < A_4$; $A_2 < A_3$; $A_3 < A_4$; $A_3 < A_5$.

<u>*Riferimento*</u>: M. Fischetti, "Lezioni di Ricerca Operativa" pag. 147-152.

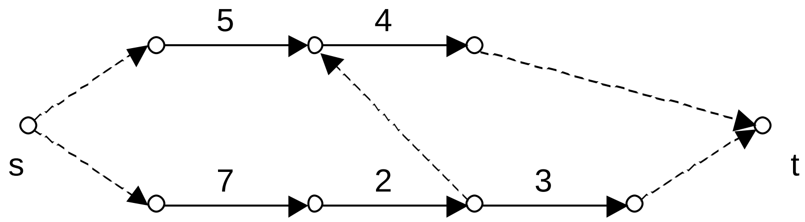
Soluzione

- Costruisci un grafo orientato G associando ad ogni attività un arco e introducendo le precedenze con archi fittizi.

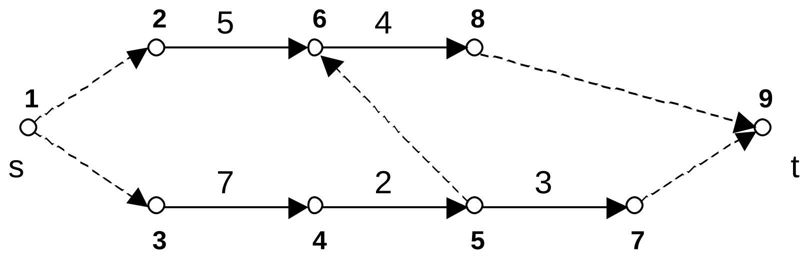


- Costruisci un grafo G' modificando il grafo G come segue: (i) elimina gli archi fittizi che sono obsoleti – tale operazione non deve né togliere né aggiungere precedenze; (ii) aggiungi un vertice s e gli archi fittizi sv per ogni vertice v di G in cui non entra alcun arco; (iii) aggiungi un vertice t e gli archi fittizi vt per ogni vertice v di G da cui non esce alcun arco; (iv) associa ad ogni attività A_i il valore d_i (per $i = 1, \ldots, 5$); (v) associa ad ogni arco fittizio il valore 0 (non riportato in figura).

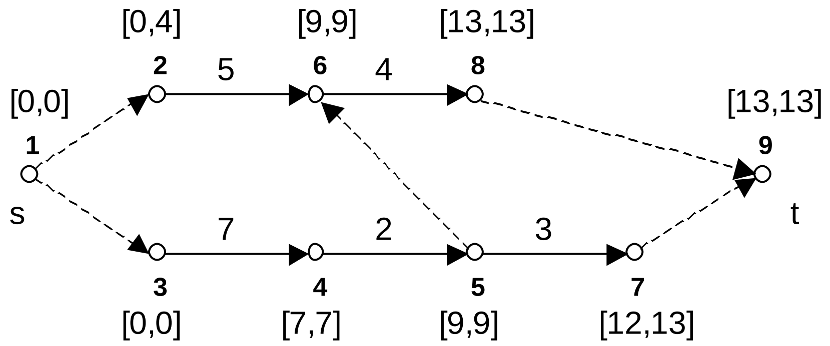
Osservazione: i valori d_i delle attività e i valori 0 degli archi fittizi possono essere visti come "costi" q_{ij} associati agli archi di G'; si osservi che, in tal modo, il costo di un cammino di costo massimo da s a t in G' è pari al tempo minimo di completamento del progetto.



- Ordina topologicamente gli archi di *G'*, cioè associa una numerazione progressiva ai nodi in modo che se il numero associato a un nodo *x* è minore di quello associato a un nodo *y* allora non esiste un cammino in *G'* da *y* a *x*.

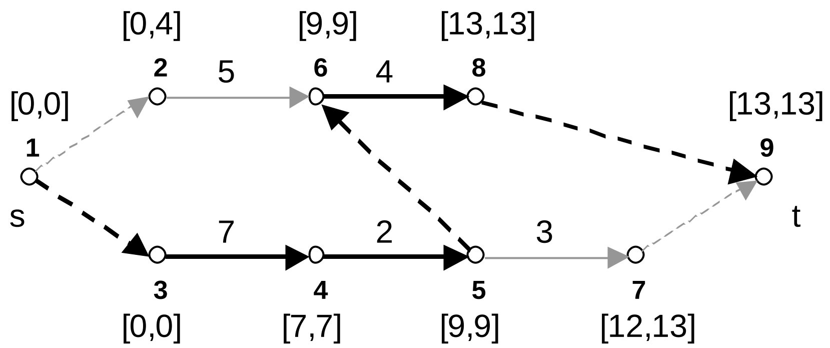


- Associa ad ogni vertice una doppia etichetta [T_{min} , T_{max}], cioè per ogni vertice *h* di *G'* calcola due valori T_{min} (*h*) e T_{max} (*h*) come segue: (i) $T_{min}(1) = 0$; (ii) per $h = 2, \ldots, 9$, poni $T_{min}(h) = \max \{T_{min}(i) + q_{ih} : \text{esiste arco } (i, h)\}$; (iii) $T_{max}(9) = T_{min}(9)$; (iv) per $h = 8, \ldots, 1$, poni $T_{max}(h) = \min \{T_{max}(i) + q_{hi} : \text{esiste arco } (h, i)\}$.



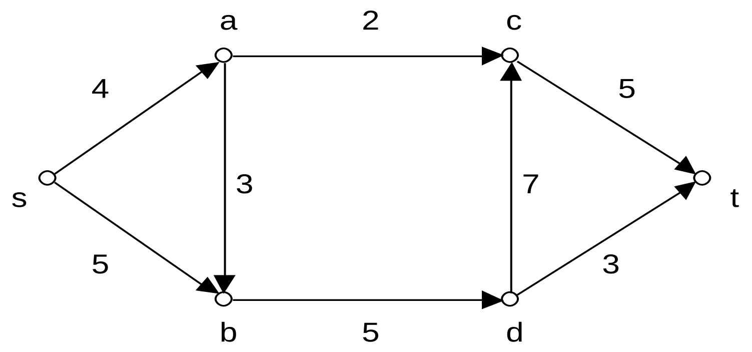
Il tempo minimo di completamento del progetto è dato da $T_{min}(9) = 13$.

Le attività critiche (cioè gli archi (*i*, *j*) tali che: $T_{min}(i) = T_{max}(i)$; $T_{min}(j) = T_{max}(j)$; $T_{max}(j) - T_{max}(i) = q_{ij}$) sono A_2, A_3, A_4. Tali attività sono quelle il cui eventuale ritardo di attuazione genera direttamente un ritardo sul tempo minimo di completamento del progetto. Esse individuano almeno un *cammino critico* (insieme con archi fittizi corrispondenti ad attività critiche fittizie) come evidenziato in figura sottostante.



Esercizio su massimo flusso

Calcolare mediante l'algoritmo di Ford-Fulkerson un flusso da *s* a *t* di valore massimo nella seguente rete, in cui accanto ad ogni arco è indicata la corrispondente capacità.



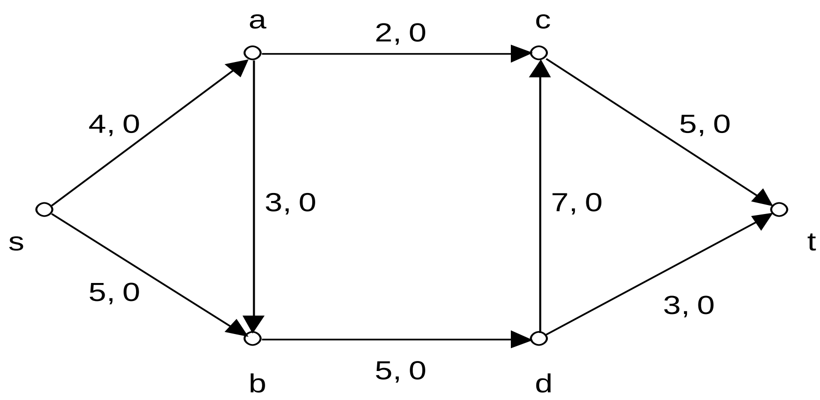
<u>*Riferimento*</u>: R. Baldacci, M. Dell'Amico, "Fondamenti di Ricerca Operativa" pag. 89-93.

Soluzione

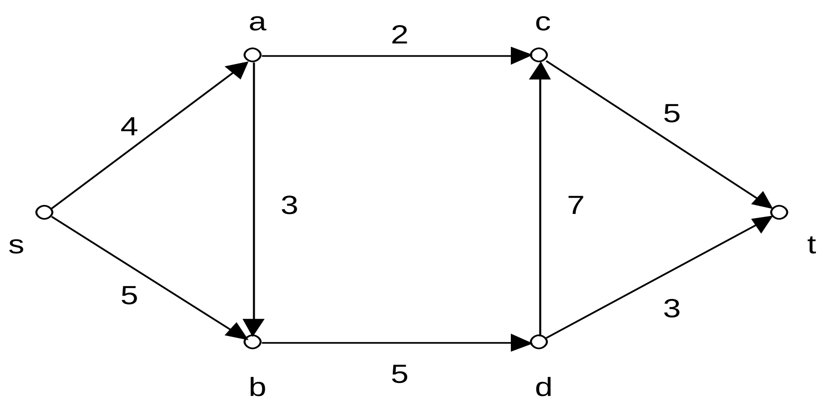
Applichiamo l'algoritmo di Ford-Fulkerson, nelle pagine seguenti.

Inizializzazione

- poni pari a 0 ogni componente del flusso su ogni arco

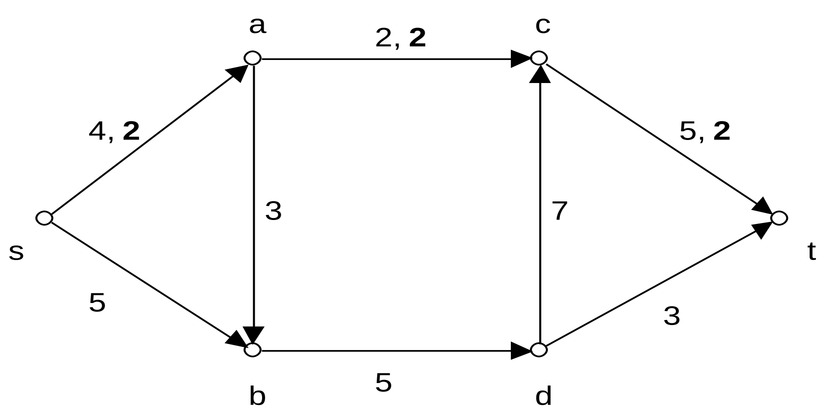


- costruisci rete incrementale rispetto al flusso



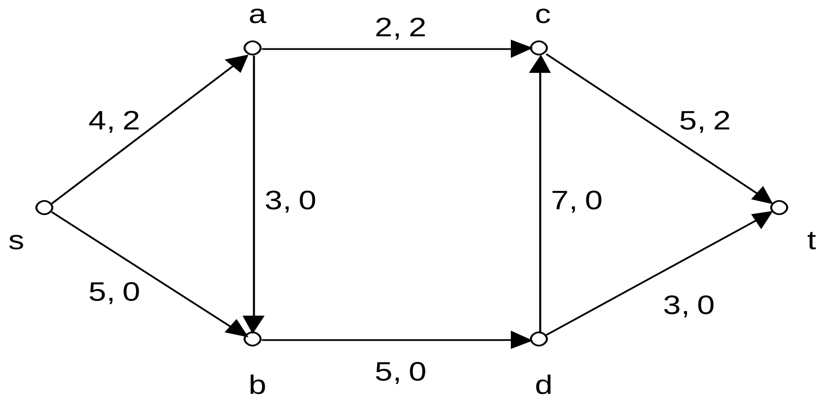
- individua cammino da *s* a *t* nella rete incrementale, e saturalo; un cammino è *s*, *a*, *c*, *t*;

lo si satura “inviando” più flusso possibile, cioè 2 (l’arco *ac* è collo di bottiglia)

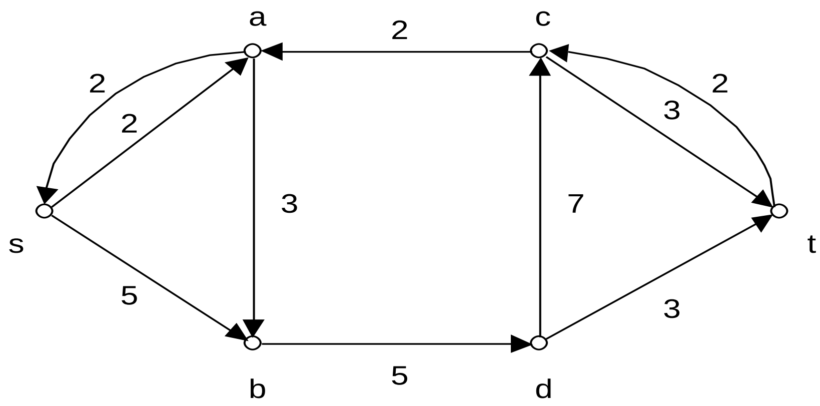


<u>Iterazione 1</u>

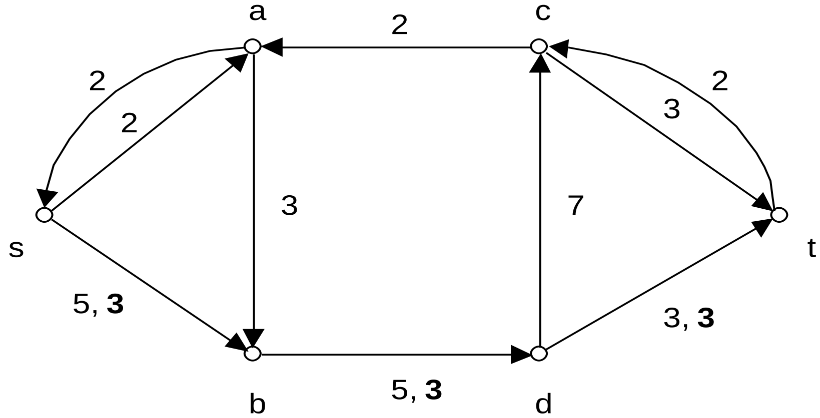
- aggiorna il flusso (in base al cammino aumentante trovato)



- costruisci la rete incrementale relativa al flusso

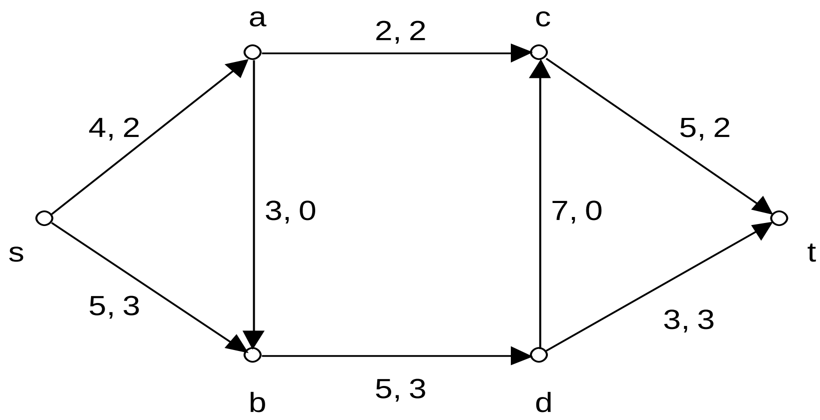


- individua cammino da *s* a *t* nella rete incrementale, e saturalo; un cammino è *s*, *b*, *d*, *t*; lo si satura "inviando" più flusso possibile, cioè 3 (l'arco *dt* è collo di bottiglia)

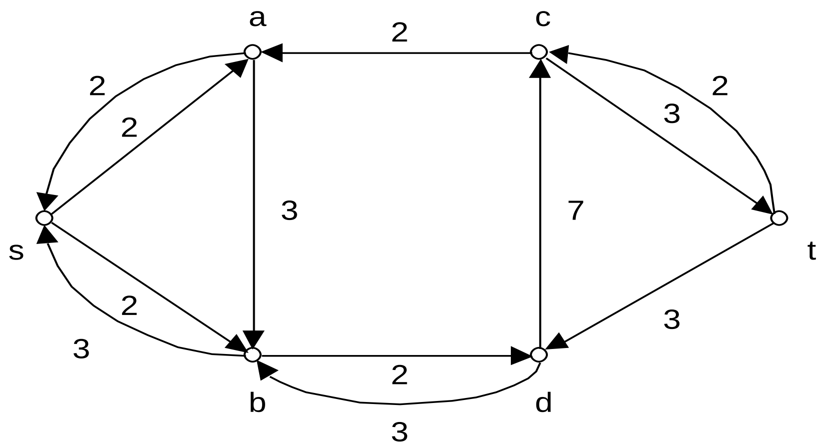


Iterazione 2

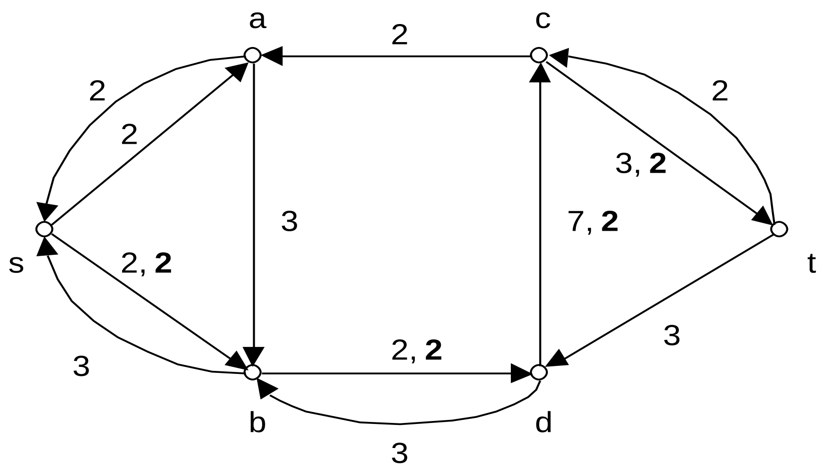
- aggiorna il flusso (in base al cammino aumentante trovato)



- costruisci la rete incrementale relativa al flusso

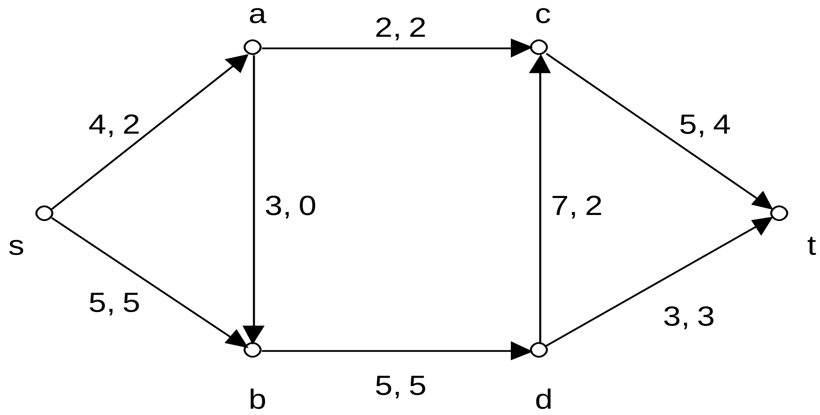


- individua cammino da *s* a *t* nella rete incrementale, e saturalo; un cammino è *s*, *b*, *d*, *c*, *t*; lo si satura "inviando" più flusso possibile, cioè 2 (l'arco *sb* è collo di bottiglia)

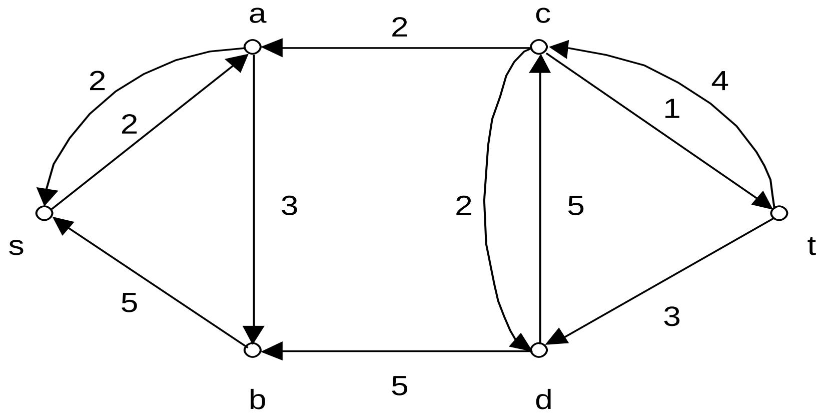


Iterazione 3

- aggiorna il flusso (in base al cammino aumentante trovato)

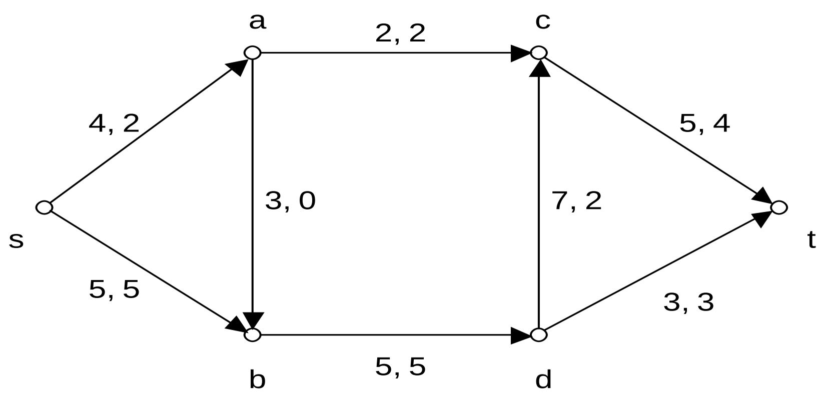


- costruisci la rete incrementale relativa al flusso



- individua cammino da *s* a *t* nella rete incrementale, e saturalo.

in questo caso non esiste alcun cammino da *s* a *t* nella rete incrementale; quindi il flusso corrente (cioè quello relativo all'ultimo aggiornamento – nella figura sotto) è ottimo; il suo valore è dato dalla somma delle componenti del flusso relative agli archi uscenti da *s*, cioè, 2 + 5 = 7.



Esercizio su programmazione della produzione

Un bene deve essere prodotto su un orizzonte temporale composto da 4 periodi. Per ogni periodo t = 1,…, 4, è nota la domanda d_t da soddisfare nel periodo t. All'inizio del periodo 1 non ci sono giacenze in magazzino e non si vuole che ce ne siano alla fine del periodo 4.

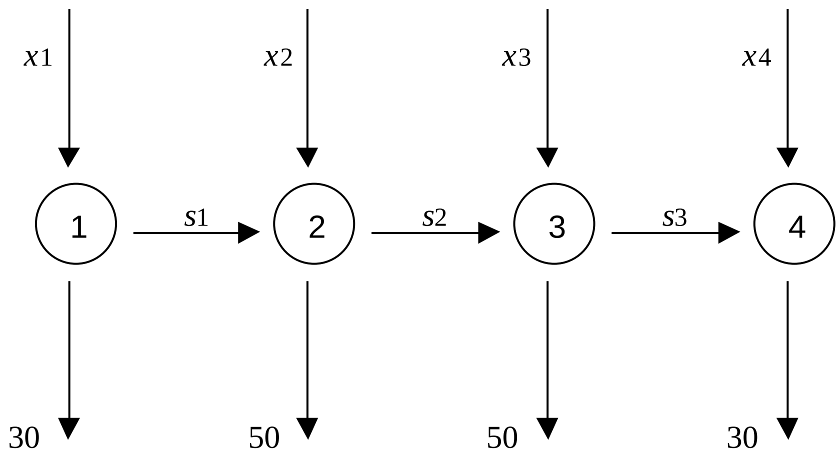
Applicando il metodo di Wagner-Whitin, determinare per t = 1,…, 4, la quantità x_t prodotte nel periodo t e le quantità s_t in giacenza nel periodo t, in modo da minimizzare i costi totali e da soddisfare le domande al momento in cui vengono effettuate, sapendo che la funzione costo di produzione è $C_t = A_t w(x_t) + c_t x_t$, dove: $w(x_t) = 1$ se $x_t > 0$, e $w(x_t) = 0$ altrimenti; la funzione costo di stoccaggio è $H_t = h_t s_t$; inoltre in tabella sono riportati i dati relativi a quanto sopra.

Periodo	Domanda (d)	A	c	h
1	30	40	2	1
2	50	20	3	2
3	50	10	4	2
4	30	20	2	

Riferimento: A. Sassano, "Modelli e algoritmi della Ricerca Operativa" pag. 327-340.

Soluzione

La situazione può essere schematizzata come nella figura di seguito, in cui ad ogni periodo i sono associate le quantità d_i di domanda, x_i di produzione e s_i di stoccaggio (ciò che avanza a fine periodo). Quindi, note le quantità d_i e le funzioni costo, il problema è determinare le quantità x_i per ogni periodo i = 1, …, 4 e le quantità s_i per ogni periodo i = 1, …, 3.



<u>Fase 1</u> : calcola i valori $M(i, j)$ per ogni coppia di periodi (i, j) con $i \leq j$

Il valore $M(i, j)$ è il costo totale che si dovrebbe pagare per soddisfare le domande dei periodi i, i+1, …, j, producendo solo nel periodo i (partendo da giacenza nulla e lasciando giacenza nulla).

$M(1,1) = 40 + 2 \cdot 30 = 100$

$M(1,2) = 40 + 2 \cdot (30+50) + 1 \cdot 50 = 250$

$M(1,3) = 40 + 2 \cdot (30+50+50) + 1 \cdot (50+50) + 2 \cdot 50 = 500$

$M(1,4) = 40 + 2 \cdot (30+50+50+30) + 1 \cdot (50+50+30) + 2 \cdot (50+30) + 2 \cdot 30 = 710$

$M(2,2) = 20 + 3 \cdot 50 = 170$

$M(2,3) = 20 + 3 \cdot (50+50) + 2 \cdot 50 = 420$

$M(2,4) = 20 + 3 \cdot (50+50+30) + 2 \cdot (50+30) + 2 \cdot 30 = 630$

$M(3,3) = 10 + 4 \cdot 50 = 210$

$M(3,4) = 10 + 4 \cdot (50+30) + 2 \cdot 30 = 390$

$M(4,4) = 20 + 2 \cdot 30 = 80$

<u>Fase 2</u> : calcola ricorsivamente F_k per k = 1, …, 4

Il valore F_k è il valore di una soluzione ottima del problema ristretto ai primi k periodi. Quindi F_4 è il valore di una soluzione ottima cercata. Il calcolo avviene ricorsivamente, ponendo $F_0 = 0$ e applicando la formula $F_k = \min_{1 \leq j \leq k} \{F_{j-1} + M(j, k)\}$.

$F_0 = 0$

$F_1 = \min \{F_0 + M(1,1)\} = 100$

$F_2 = \min \{F_0 + M(1,2), F_1 + M(2,2)\} = \{250, 270\} = 250$

$F_3 = \min \{F_0 + M(1,3), F_1 + M(2,3), F_2 + M(3,3)\} = \{500, 520, 460\} = 460$

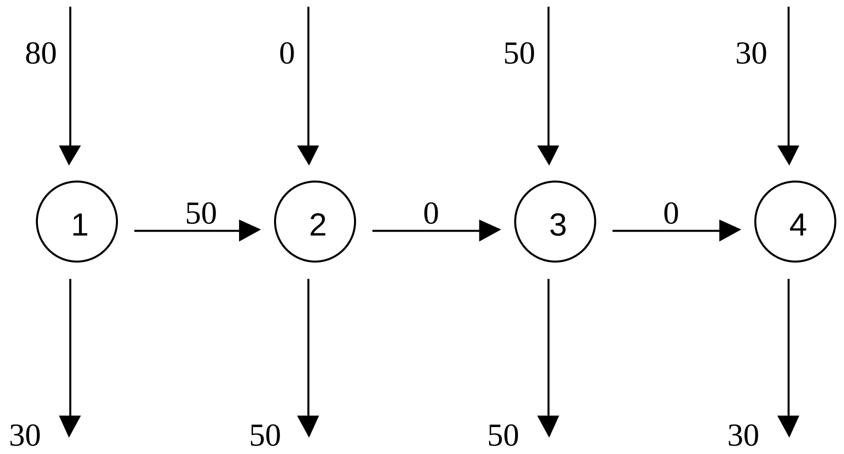
$F_4 = \min \{F_0 + M(1,4), F_1 + M(2,4), F_2 + M(3,4), F_3 + M(4,4)\} = \{710, 730, 640, 540\} = 540$

Fase 3 : evidenzia la soluzione ottima trovata

Per fare ciò si parte da F_4 andando a ritroso, sostituendo (in base ai calcoli appena svolti) iterativamente i valori F_k fino ad arrivare a F_0, come di seguito:

$F_4 = F_3 + M(4,4) = F_2 + M(3,3) + M(4,4) = F_0 + M(1,2) + M(3,3) + M(4,4)$.

Si ottiene che la soluzione ottima è individuata da $M(1,2) + M(3,3) + M(4,4)$, cioè si produce: nel periodo 1 (soddisfacendo le domande dei periodi 1, 2), nel periodo 3 (soddisfacendo le domande del periodo 3), nel periodo 4 (soddisfacendo le domande del periodo 4). I valori ottimi di $x_1, \ldots, x_4$ e $s_1, \ldots, s_3$ possono quindi essere dedotti dal loro significato e dalle domande, come indicato in figura.



Esercizio su localizzazione di impianti

Bisogna attivare un certo numero di impianti, da scegliere fra m potenziali impianti, che dovranno rifornire n clienti. Attivare il potenziale impianto j costa f_j. Ogni impianto si intende abbia capacità di rifornimento illimitata. Ogni cliente sarà collegato ad un unico impianto; in particolare collegare un cliente i ad un impianto j comporta un costo c_{ij}. Problema: scegliere quali impianti attivare in modo da minimizzare i costi complessivi di attivazione e di assegnazione ai clienti.

Calcolare una soluzione "greedy" di tale problema per la seguente istanza:

matrice c_{ij}

		impianti				
		1	2	3	4	5
	1	0	1	7	0	4
	2	1	0	3	0	3
	3	3	7	1	4	2
clienti	4	1	7	4	0	7
	5	1	0	3	2	4
	6	4	4	1	4	3
	7	7	1	2	1	0
vettore f		7	9	5	4	7

Riferimento: A. Sassano, “Modelli e algoritmi della Ricerca Operativa” pag. 269-274.

Soluzione

Inizializzazione

$i = 1$; $T_0 = \emptyset$; $Z(T_0) = \infty$.

Iterazione 1

$Z(\{1\})$	$Z(\{2\})$	$Z(\{3\})$	$Z(\{4\})$	$Z(\{5\})$
24	29	26	15	30

Ad esempio $Z(\{1\}) = (0 + 1 + 3 + 1 + 1 + 4 + 7) + 7 = 24$.

Il minimo è $Z(\{4\}) = 15$;

$15 < \infty \;\Rightarrow\; i = 2,\ T_1 = \{4\}$.

<u>Iterazione 2</u>

$Z(\{4,1\})$	$Z(\{4,2\})$	$Z(\{4,3\})$	$Z(\{4,5\})$
20	22	14	18

Ad esempio $Z(\{4,1\}) = (0 + 0 + 3 + 0 + 1 + 4 + 1) + (7 + 4) = 20$.

Il minimo è $Z(\{4,3\}) = 14$;

$14 < 15 \;\Rightarrow\; i = 3,\ T_2 = \{4,3\}$.

<u>Iterazione 3</u>

$Z(\{4,3,1\})$	$Z(\{4,3,2\})$	$Z(\{4,3,5\})$
22	21	20

Ad esempio $Z(\{4,3,1\}) = (0 + 0 + 1 + 0 + 1 + 1 + 1) + (7 + 5 + 4) = 22$.

Il minimo è $Z(\{4,3,5\}) = 20$;

$20 \geq 14 \;\Rightarrow$ STOP

La soluzione greedy è $T_2 = \{4,3\}$, con $Z(\{4,3\}) = 14$.