

# Elementi di Teoria dei Grafi

## Argomenti lezione:

- Il problema del percorso orientato ottimo (*“shortest path problem”*)
- Algoritmo di Floyd–Warshall
- Esercizi
- Pianificazione di Progetti

# Percorso orientato di costo minimo (1)

Problema affrontato: Identificare il percorso orientato ottimo tra tutte le coppie di nodi in un *digrafo pesato*

Approccio visto: Applicare ripetutamente l'algoritmo di Dijkstra tra un nodo e tutti gli altri nodi del digrafo

Il problema di individuare il percorso di costo minimo tra ogni coppia di nodi necessita  $n$  esecuzioni di Dijkstra a partire da ogni nodo del digrafo ...

=> Complessità della procedura complessiva:  $O(n^3)$

# Percorso orientato di costo minimo (2)

Nuovo approccio di complessità  $O(n^3)$  :

L'algoritmo di Floyd–Warshall può essere applicato a digrafi che abbiano anche pesi negativi sugli archi, ammesso sempre che non esistano cicli a peso negativo!

Presenza di cicli a peso negativo nel digrafo:

L'algoritmo di Floyd–Warshall individua uno dei cicli e termina l'esecuzione senza aver risolto il problema  
=> in tal caso la *correttezza* dell'algo non è garantita

# Percorso orientato di costo minimo (3)

Esempio: Si noti che il problema del percorso orientato ottimo da  $s$  a  $t$  per il digrafo in figura è illimitato inferiormente e, pertanto, non ammette percorsi orientati ottimi.

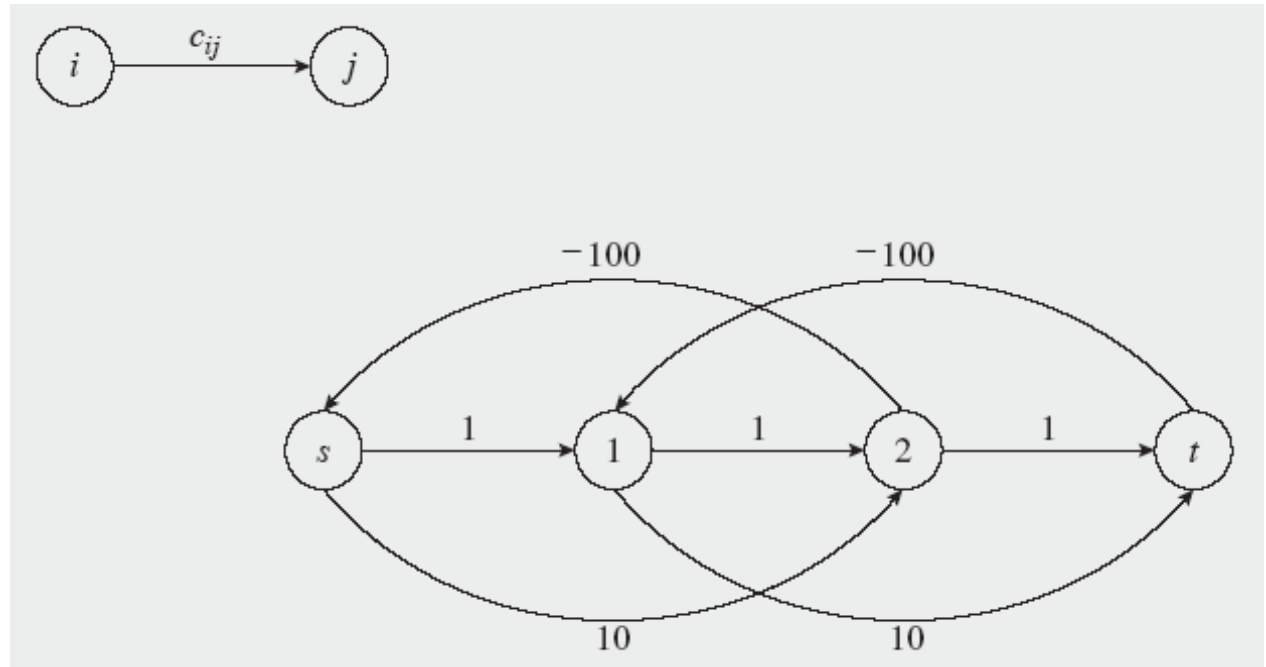
Il cammino orientato di costo minimo :

da  $s$  a  $t$  è  $\{s, 1, 2, t\}$ ;

da  $s$  a  $1$  è  $\{s, 2, t, 1\}$ ;

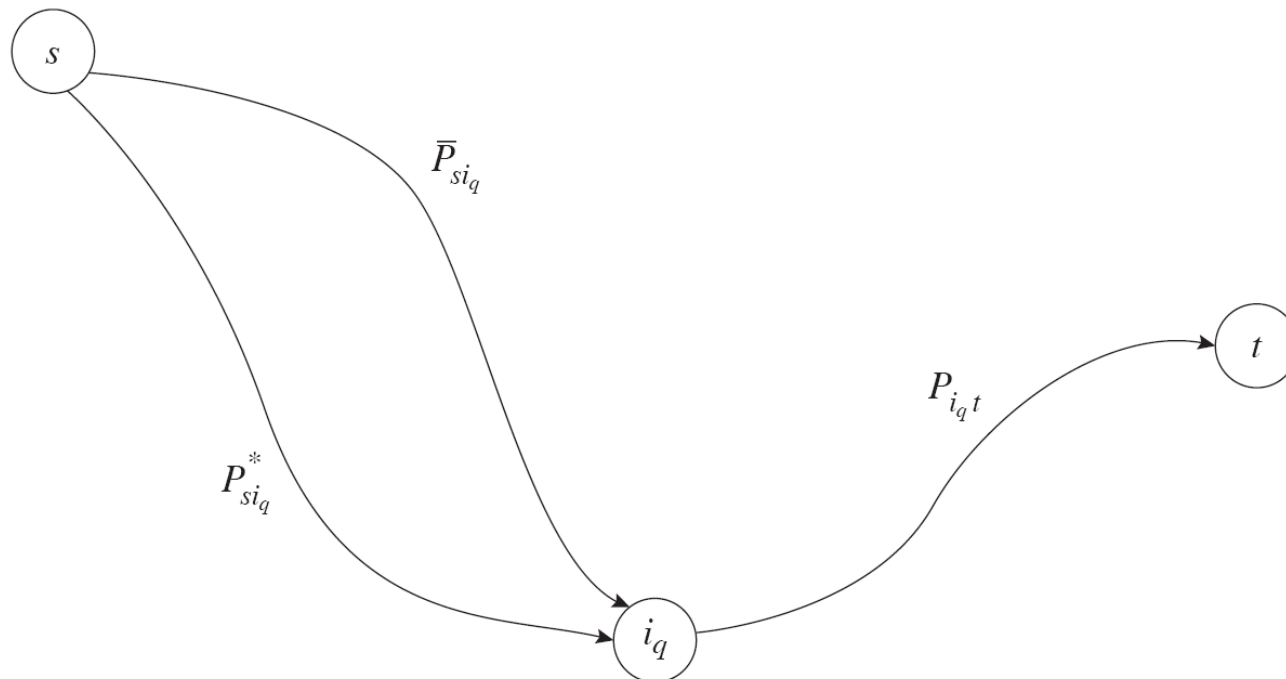
da  $2$  a  $t$  è  $\{2, s, 1, t\}$ .

Mentre non è possibile calcolare il percorso orientato di costo minimo  $s$ - $t$ .



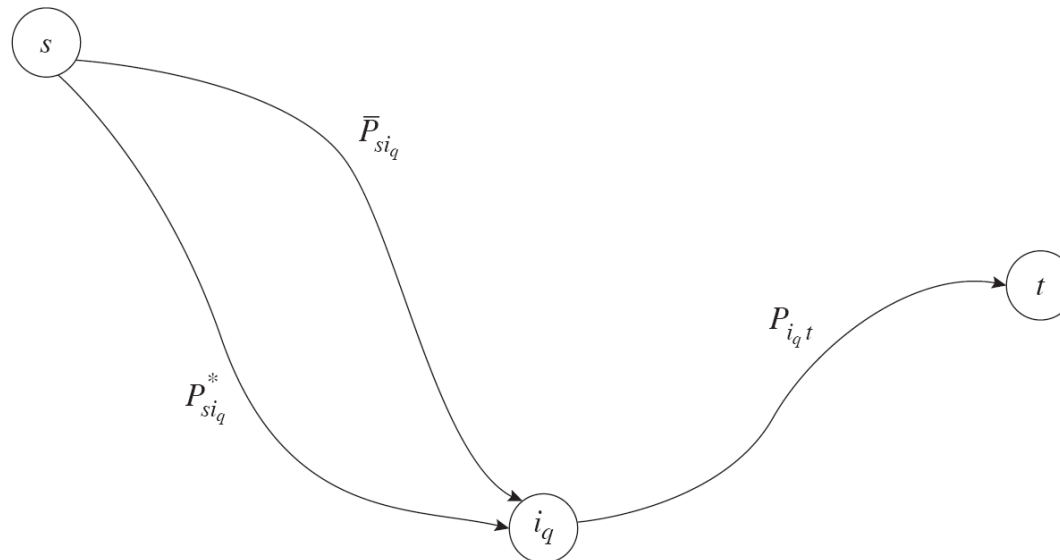
# Percorso orientato di costo minimo (4)

**Proprietà:** Se il percorso orientato  $P_{st}^* = (s = i_1, i_2, \dots, i_r = t)$ , con  $r \geq 3$ , è un percorso orientato di costo minimo da  $s$  a  $t$ , allora per ogni  $q = 2, \dots, r-1$  il sotto-percorso orientato  $P_{s i_q}^* = (s = i_1, i_2, \dots, i_q)$  è un percorso orientato minimo da  $s$  a  $i_q$ .



# Percorso orientato di costo minimo (5)

**Dimostrazione:** Sia  $P_{iq t}^*$  il sotto-percorso orientato di  $P_{st}^*$  da  $i_q$  a  $t$  e si assuma per assurdo l'esistenza di un percorso orientato  $\underline{P}_{s i_q}$  di costo minore di  $P_{s i_q}^*$ . In tal caso il percorso orientato ottenuto dalla concatenazione di  $\underline{P}_{s i_q}$  con  $\underline{P}_{i_q t}^*$  definisce un percorso orientato dal nodo  $s$  al nodo  $t$  di costo inferiore rispetto a quello del percorso orientato  $P_{st}^*$  contraddicendo l'ipotesi di ottimalità del percorso  $P_{st}^*$ .



# Percorso orientato di costo minimo (6)

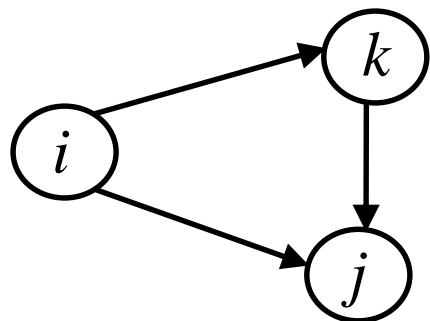
L'idea alla base dell'algoritmo di Floyd-Warshall (anno 1962) sta nella determinazione del percorso orientato ottimo  $P_{ij}^{(k)}$  dal nodo  $i$  al nodo  $j$  nel digrafo  $(N, A)$  con  $|N| = n$  nodi e  $|A| = m$  archi, costituito esclusivamente da nodi intermedi appartenenti all'insieme  $\{1, 2, \dots, k\}$ ,  $0 \leq k \leq n$ .

Sia  $d_{[i,j]}^{(k)}$  il costo di  $P_{ij}^{(k)}$ , si dimostra che il costo  $d_{[i,j]}^*$  del percorso orientato ottimo  $P_{ij}^*$  dal nodo  $i$  al nodo  $j$  è pari a  $d_{[i,j]}^{(n)}$ .

Nel caso in cui il digrafo non sia privo di cicli orientati di costo negativo, il problema del percorso orientato ottimo è “illimitato” per almeno una coppia di nodi origine-destinazione.

# Condizioni di ottimalità Floyd-Warshall (1)

**Teorema:** Il costo  $d_{[i,j]}^{(k)}$ ,  $k = 1, \dots, n$ , del percorso orientato ottimo  $P_{ij}^{(k)}$  dal nodo  $i$  al nodo  $j$  nel digrafo  $(N, A)$  con nodi intermedi appartenenti esclusivamente all'insieme  $\{1, 2, \dots, k\}$ , è pari a

$$d_{[i,j]}^{(k)} = \min\{d_{[i,j]}^{(k-1)}, d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}\} .$$


Esempio di “triangolarizzazione”:

Per la coppia di nodi  $i, j$  si controlla se per andare da  $i$  a  $j$  conviene passare per il nodo  $k$  ovvero:

Se  $(d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)} < d_{[i,j]}^{(k-1)})$

allora  $d_{[i,j]}^{(k)} = d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$



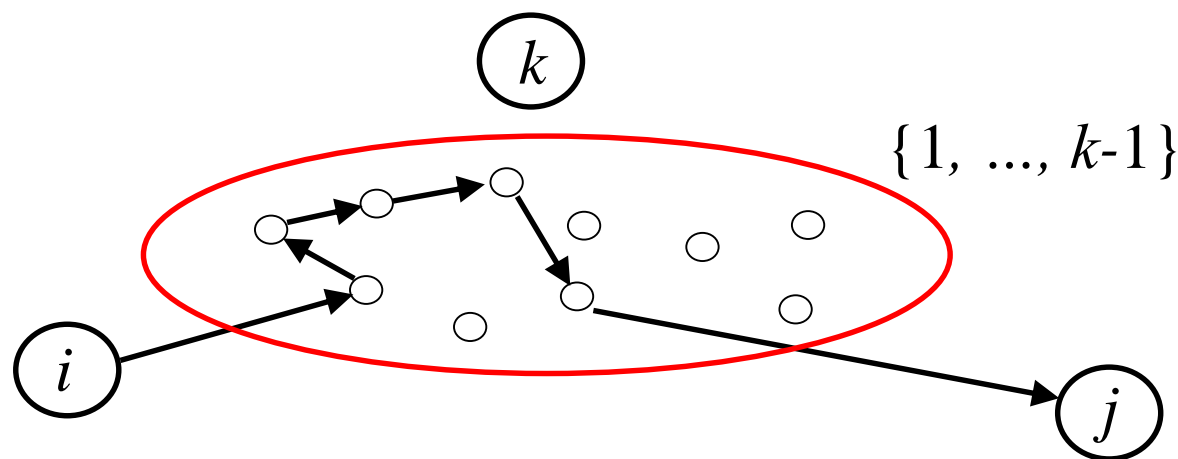
# Condizioni di ottimalità Floyd-Warshall (2)

## Dimostrazione (1):

Dato un percorso orientato di costo minimo  $P_{ij}^{(k)}$  da  $i$  a  $j$  che contiene esclusivamente nodi intermedi appartenenti all'insieme  $\{1, 2, \dots, k\}$ , si possono verificare solo questi due casi:

1)  $P_{ij}^{(k)}$  non include il nodo  $k$ , quindi coincide con  $P_{ij}^{(k-1)}$ .

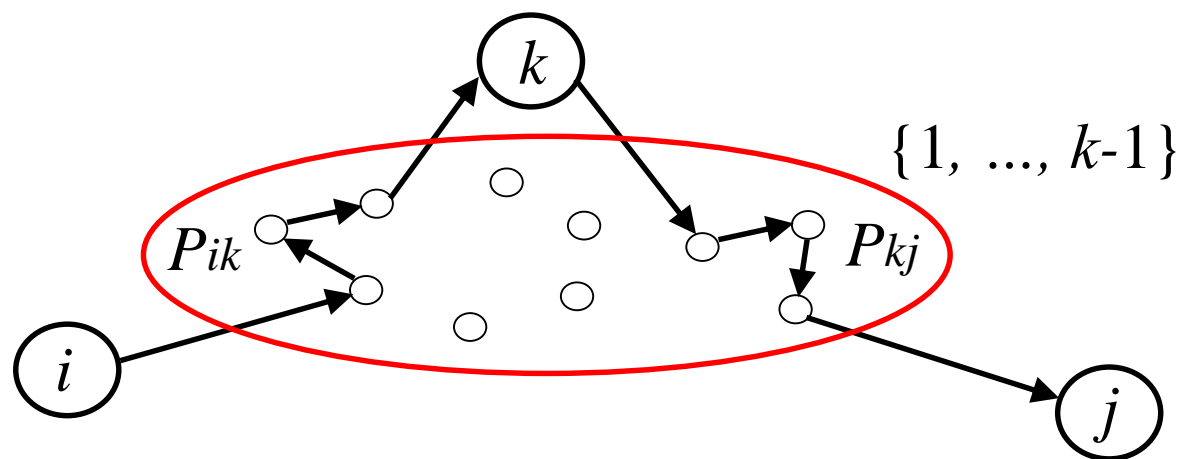
In tal caso,  $d_{[i,j]}^{(k)} = d_{[i,j]}^{(k-1)} = \min\{ d_{[i,j]}^{(k-1)}, d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)} \}$ .



# Condizioni di ottimalità Floyd-Warshall (3)

## Dimostrazione (2):

2)  $P_{ij}^{(k)}$  include il nodo  $k$ . In questo caso,  $P_{ij}^{(k)}$  può decomporre in due sottopercorsi orientati disgiunti che, non avendo  $k$  come nodo intermedio, coincidono rispettivamente con i percorsi orientati ottimi  $P_{ik}^{(k-1)}$  e  $P_{kj}^{(k-1)}$  che hanno rispettivamente costo  $d_{[i, k]}^{(k-1)}$  e  $d_{[k, j]}^{(k-1)}$ . Pertanto, in questo caso si ha  $d_{[i, j]}^{(k)} = d_{[i, k]}^{(k-1)} + d_{[k, j]}^{(k-1)} = \min\{d_{[i, j]}^{(k-1)}, d_{[i, k]}^{(k-1)} + d_{[k, j]}^{(k-1)}\}$ .



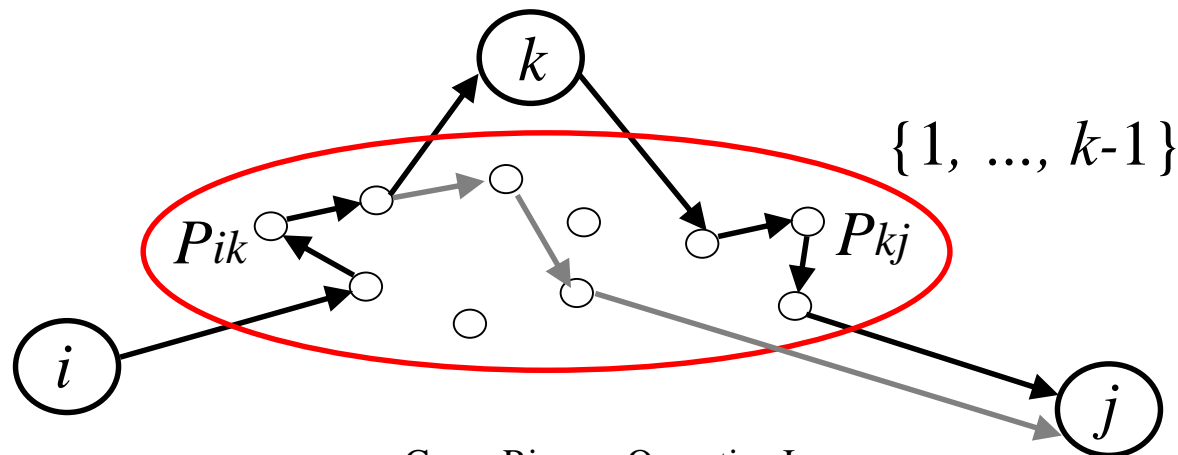
# Algoritmo di Floyd–Warshall (1)

## Notazioni:

Indicando con  $pred_{[i,j]}^{(k)}$  il nodo predecessore di  $j$  nel percorso orientato minimo  $P_{ij}^{(k)}$ , si ha che:

- 1)  $pred_{[i,j]}^{(k)} = pred_{[i,j]}^{(k-1)}$  ( $P_{ij}^{(k)}$  non include il nodo  $k$ );
- 2)  $pred_{[i,j]}^{(k)} = pred_{[k,j]}^{(k-1)}$  ( $P_{ij}^{(k)}$  include il nodo  $k$ ).

Indichiamo, inoltre, con  $pred_{[i,j]}^*$  il nodo predecessore di  $j$  nel percorso orientato minimo da  $i$  a  $j$ .

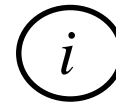


# Algoritmo di Floyd–Warshall (2)

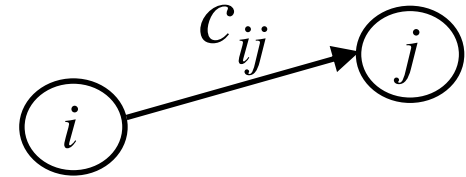
## Inizializzazione:

L'algoritmo di Floyd-Warshall pone inizialmente per  $k = 0$ :

$$d_{[i, i]}^{(0)} = 0, \text{ } pred_{[i, i]}^{(0)} = i, \text{ } i \in N;$$



$$d_{[i, j]}^{(0)} = c_{ij}, \text{ } pred_{[i, j]}^{(0)} = i, \text{ } (i, j) \in A \text{ e } i \neq j;$$



$$d_{[i, j]}^{(0)} = +\infty, \text{ } pred_{[i, j]}^{(0)} = j, \text{ } (i, j) \notin A \text{ e } i \neq j.$$



# Algoritmo di Floyd–Warshall (3)

## Passo iterativo:

Dopo l’inizializzazione, l’algoritmo itera  $n$  volte. All’iterazione  $k$ ,  $k = 1, \dots, n$  viene eseguita la cosiddetta «triangolarizzazione» relativa al nodo  $k$ .

Dall’iterazione precedente, per ogni  $[i, j] \in N \times N$ , abbiamo i percorsi orientati minimi  $P_{ij}^{(k-1)}$  con nodi intermedi nell’insieme  $\{1, 2, \dots, k-1\}$ . Conosciamo le etichette  $d_{[i,j]}^{(k-1)}$  dei costi (ottimi) e i nodi  $pred_{[i,j]}^{(k-1)}$ .

Per ogni  $[i, j] \in N \times N$ , a valle della triangolarizzazione relativa al nodo  $k$ , si determinano le etichette  $d_{[i,j]}^{(k)}$  e i nodi  $pred_{[i,j]}^{(k)}$  associati ai  $P_{ij}^{(k)}$ .

Al termine dell’algoritmo, per ogni  $[i, j] \in N \times N$ , si ha:

le etichette  $d_{[i,j]}^{(n)}$  sono il costo  $d_{[i,j]}^*$  del percorso orientato minimo  $P_{ij}^*$ , e i nodi  $pred_{[i,j]}^{(n)}$  sono i predecessori  $pred_{[i,j]}^*$  in tale percorso.

# Algoritmo di Floyd–Warshall (4)

Richiamo notazioni:

$G = (N, A)$  : digrafo

con  $n = |N|$  e  $m = |A|$

$c_{ij}$  : il costo dell'arco  $[i, j]$

$d_{[i,j]}^{(k)}$  : è la distanza tra  
il nodo  $i$  e il nodo  $j$   
all'iterazione  $k$ -esima

$pred_{[i,j]}^{(k)}$  : è il nodo che  
precede il nodo  $j$  nel  
percorso che va da  $i$  a  $j$   
all'iterazione  $k$ -esima ( $P_{ij}^{(k)}$ )

```

for each  $[i, j] \in \mathcal{N} \times \mathcal{N}$  do
     $d_{[i,j]}^{(0)} := +\infty$ ;  $pred_{[i,j]}^{(0)} := j$ ;
end_for
for each  $i \in \mathcal{N}$  do  $d_{[i,i]}^{(0)} := 0$ ;  $pred_{[i,i]}^{(0)} := i$ ; end_for
for each  $(i, j) \in \mathcal{A}$  do
     $d_{[i,j]}^{(0)} := c_{ij}$ ;  $pred_{[i,j]}^{(0)} := i$ ;
end_for
for  $k := 1$  to  $n$  do
    for each  $[i, j] \in \mathcal{N} \times \mathcal{N}$  do
        if  $d_{[i,j]}^{(k-1)} > d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$  then
             $d_{[i,j]}^{(k)} := d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$ ;
             $pred_{[i,j]}^{(k)} := pred_{[k,j]}^{(k-1)}$ ;
        else
             $d_{[i,j]}^{(k)} := d_{[i,j]}^{(k-1)}$ ;
             $pred_{[i,j]}^{(k)} := pred_{[i,j]}^{(k-1)}$ ;
        end_if
    end_for
end_for
end_for
    
```

# Algoritmo di Floyd–Warshall (5)

Inizializzazione:  
costi dei percorsi  
predecessori

**for each**  $[i, j] \in \mathcal{N} \times \mathcal{N}$  **do**

$d_{[i,j]}^{(0)} := +\infty$ ;  $pred_{[i,j]}^{(0)} := j$ ;

**end\_for**

**for each**  $i \in \mathcal{N}$  **do**  $d_{[i,i]}^{(0)} := 0$ ;  $pred_{[i,i]}^{(0)} := i$ ; **end\_for**

**for each**  $(i, j) \in \mathcal{A}$  **do**

$d_{[i,j]}^{(0)} := c_{ij}$ ;  $pred_{[i,j]}^{(0)} := i$ ;

**end\_for**

**for**  $k := 1$  **to**  $n$  **do**

**for each**  $[i, j] \in \mathcal{N} \times \mathcal{N}$  **do**

**if**  $d_{[i,j]}^{(k-1)} > d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$  **then**

$d_{[i,j]}^{(k)} := d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$ ;

$pred_{[i,j]}^{(k)} := pred_{[k,j]}^{(k-1)}$ ;

**else**

$d_{[i,j]}^{(k)} := d_{[i,j]}^{(k-1)}$ ;

$pred_{[i,j]}^{(k)} := pred_{[i,j]}^{(k-1)}$ ;

**end\_if**

**end\_for**

**end\_for**

**Obiettivo:**  
trovare il percorso  
orientato minimo  
origine-multiplo  
destinazione-multiplo

# Algoritmo di Floyd–Warshall (6)

L'algoritmo esegue  $n$  iterazioni e ad ogni iterazione controlla tutti gli elementi di una matrice con  $n$  righe ed  $n$  colonne, per cui complessivamente l'algo di Floyd–Warshall termina in  $O(n^3)$ .

```
for  $k := 1$  to  $n$  do
  for each  $[i, j] \in \mathcal{N} \times \mathcal{N}$  do
    if  $d_{[i,j]}^{(k-1)} > d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$  then
       $d_{[i,j]}^{(k)} := d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$ ;
       $pred_{[i,j]}^{(k)} := pred_{[k,j]}^{(k-1)}$ ;
    else
       $d_{[i,j]}^{(k)} := d_{[i,j]}^{(k-1)}$ ;
       $pred_{[i,j]}^{(k)} := pred_{[i,j]}^{(k-1)}$ ;
    end_if
  end_for
end_for
```



# Algoritmo di Floyd–Warshall (7)

```
for  $k := 1$  to  $n$  do
  for each  $[i, j] \in \mathcal{N} \times \mathcal{N}$  do
    if  $d_{[i,j]}^{(k-1)} > d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$  then
       $d_{[i,j]}^{(k)} := d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$  ;
       $pred_{[i,j]}^{(k)} := pred_{[k,j]}^{(k-1)}$  ;
    else
       $d_{[i,j]}^{(k)} := d_{[i,j]}^{(k-1)}$  ;
       $pred_{[i,j]}^{(k)} := pred_{[i,j]}^{(k-1)}$  ;
    end_if
  end_for
end_for
```

Per ogni  $[i, j] \in N \times N$ ,  
si effettua la triangolarizzazione  
relativa alla  $k$ -esima iterazione,  
fissando le etichette  $d_{[i,j]}^{(k)}$   
e i nodi  $pred_{[i,j]}^{(k)}$  associati a  $P_{ij}^{(k)}$ .

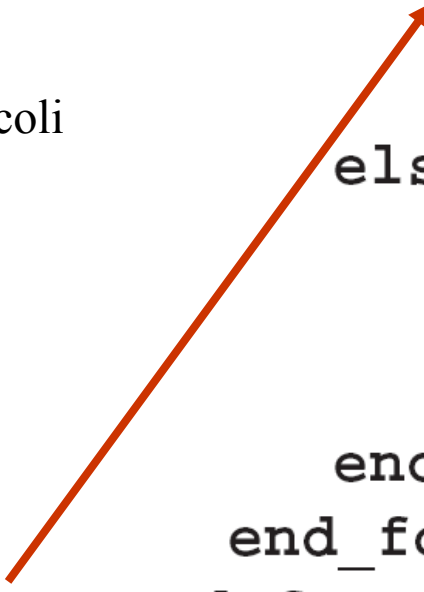
# Algoritmo di Floyd–Warshall (8)

Se il digrafo non contiene cicli di costo negativo, allora l'algoritmo termina restituendo la matrice corretta dei costi dei cammini minimi.

Se il digrafo contiene cicli di costo negativo, non è più garantito che l'algoritmo calcoli correttamente i valori delle etichette  $d_{[i,j]}^{(k)}$ .

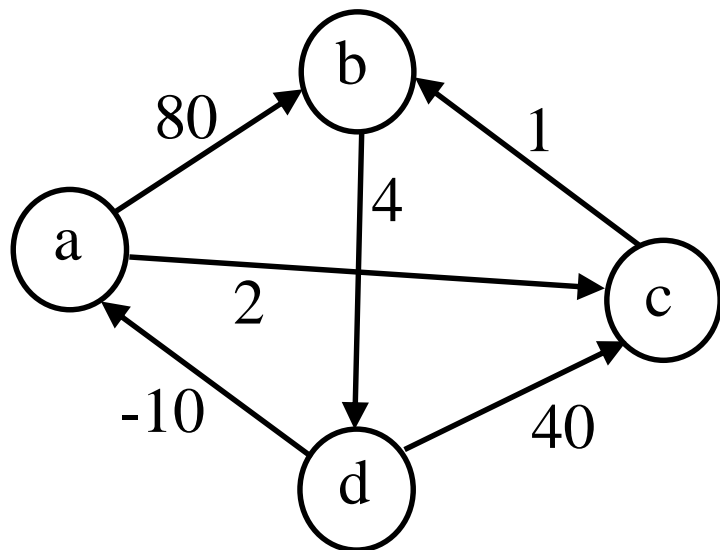
L'algoritmo è in grado di rilevare l'eventuale presenza di un ciclo orientato di costo negativo quando  $d_{[h,h]}^{(k)} < 0$ , per un nodo  $h$  all'iterazione  $k$ .

```
for  $k := 1$  to  $n$  do
  for each  $[i, j] \in \mathcal{N} \times \mathcal{N}$  do
    if  $d_{[i,j]}^{(k-1)} > d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$  then
       $d_{[i,j]}^{(k)} := d_{[i,k]}^{(k-1)} + d_{[k,j]}^{(k-1)}$ ;
       $pred_{[i,j]}^{(k)} := pred_{[k,j]}^{(k-1)}$ ;
    else
       $d_{[i,j]}^{(k)} := d_{[i,j]}^{(k-1)}$ ;
       $pred_{[i,j]}^{(k)} := pred_{[i,j]}^{(k-1)}$ ;
    end_if
  end_for
end_for
```



# Floyd–Warshall: I Esempio (1)

Inizializzazione



MATRICE COSTI DEI PERCORSI

<b>D\A→</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>
<b>a</b>	0	80	2	inf
<b>b</b>	inf	0	inf	4
<b>c</b>	inf	1	0	inf
<b>d</b>	-10	inf	40	0

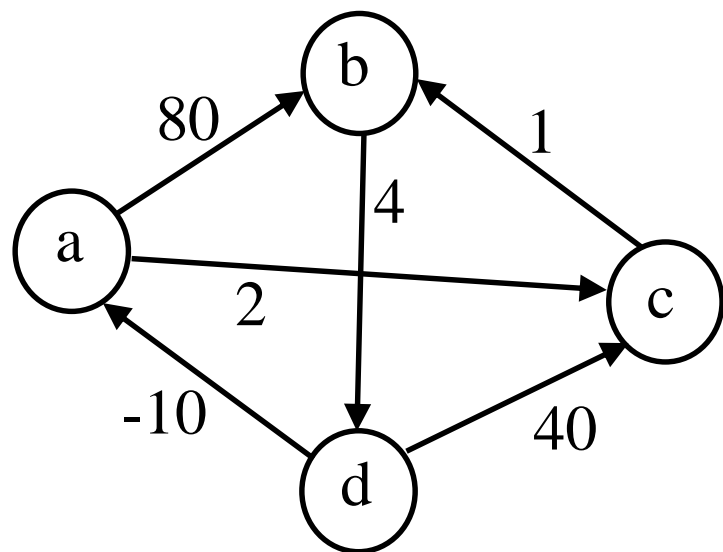
MATRICE PREDECESSORI

<b>D\A→</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>
<b>a</b>	a	a	a	d
<b>b</b>	a	b	c	b
<b>c</b>	a	c	c	d
<b>d</b>	d	b	d	d

Sulla diagonale principale della matrice dei percorsi iniziali ho tutti valori nulli

# Floyd–Warshall: I Esempio (2)

Prima iter. :  $k = a$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	inf	40	0

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	b	d	d

Percorso tra  $i$  e  $j$  passante solo per  $k = a$

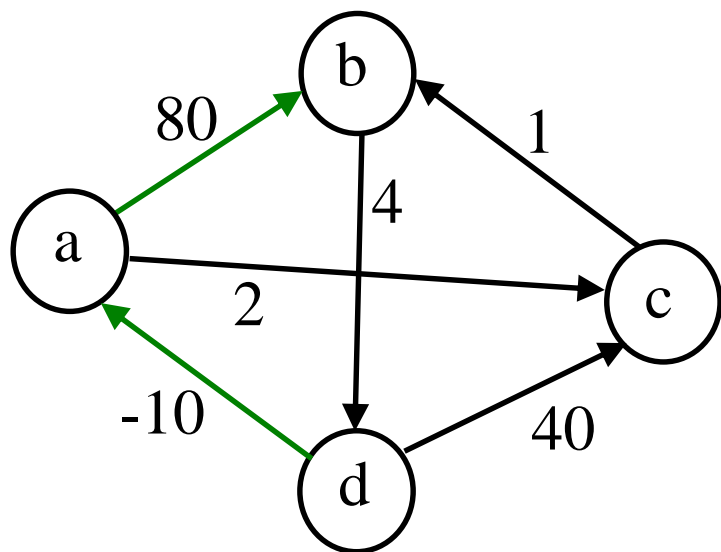
passando	k	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d

Osservazione 1: Se  $(i = k$  oppure  $j = k)$  allora non effettuo aggiornamenti

Osservazione 2: Se  $(i = j \ \&\& \ i \neq k)$  allora cerco possibili aggiornamenti

# Floyd–Warshall: I Esempio (3)

Prima iter. :  $k = a$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	40	0

MATRICE PREDECESSORI

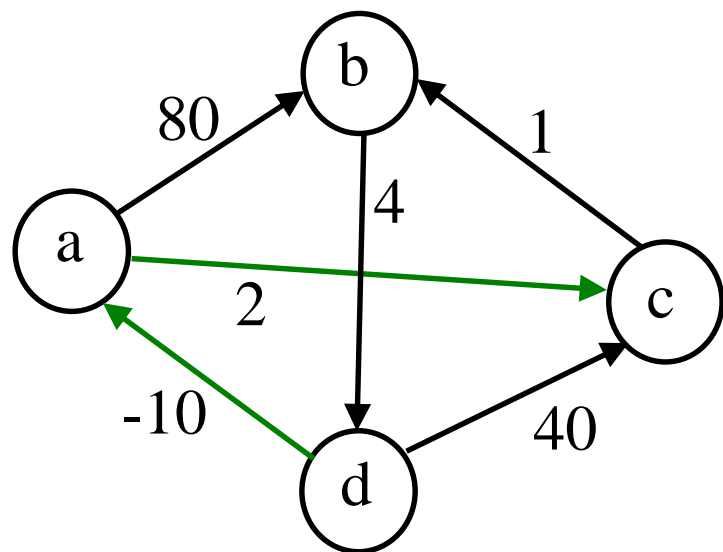
D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	a	d	d

Percorso tra  $i$  e  $j$  passante solo per  $k = a$

passando	k	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost															70		
old cost															inf		

# Floyd–Warshall: I Esempio (4)

Prima iter. :  $k = a$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	-8	0

MATRICE PREDECESSORI

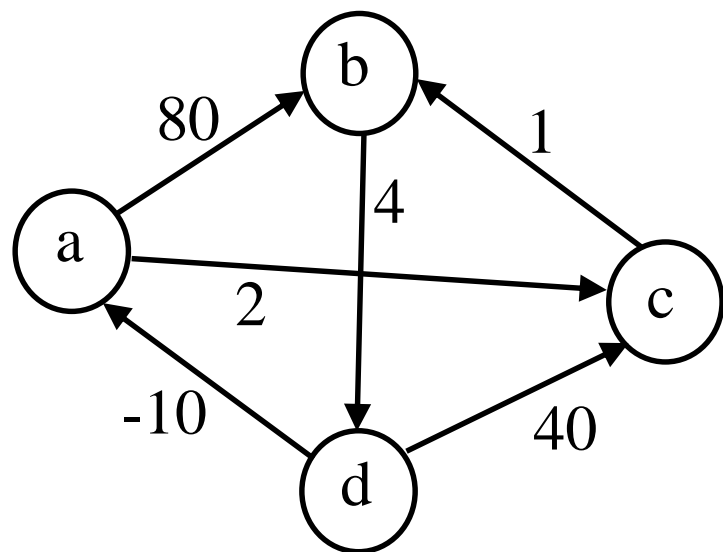
D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	a	a	d

Percorso tra  $i$  e  $j$  passante solo per  $k = a$

passando	k	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost															70	-8	
old cost															inf	40	

# Floyd–Warshall: I Esempio (5)

Prima iter. :  $k = a$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	-8	0

MATRICE PREDECESSORI

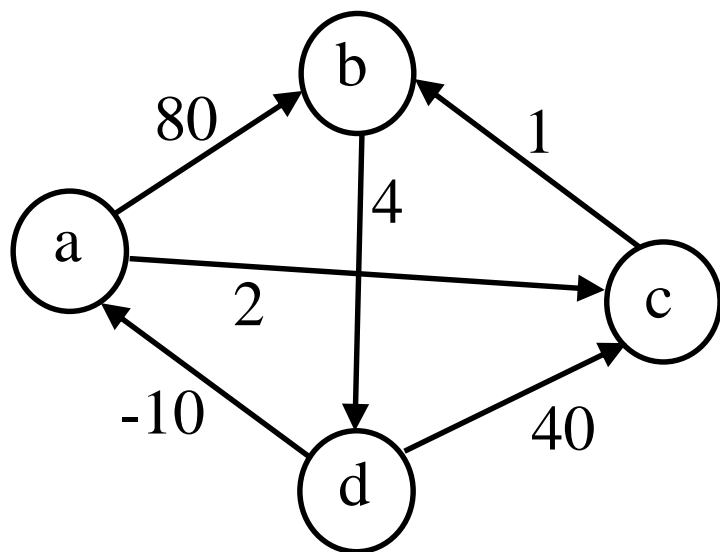
D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	a	a	d

Percorso tra  $i$  e  $j$  passante solo per  $k = a$

passando	k	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost															70	-8	
old cost															inf	40	

# Floyd–Warshall: I Esempio (6)

Seconda iter. :  $k = b$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
↓ a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	-8	0

MATRICE PREDECESSORI

D\A→	a	b	c	d
↓ a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	a	a	d

N.B. Posso passare solo per  $a, b$

passando	k	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d

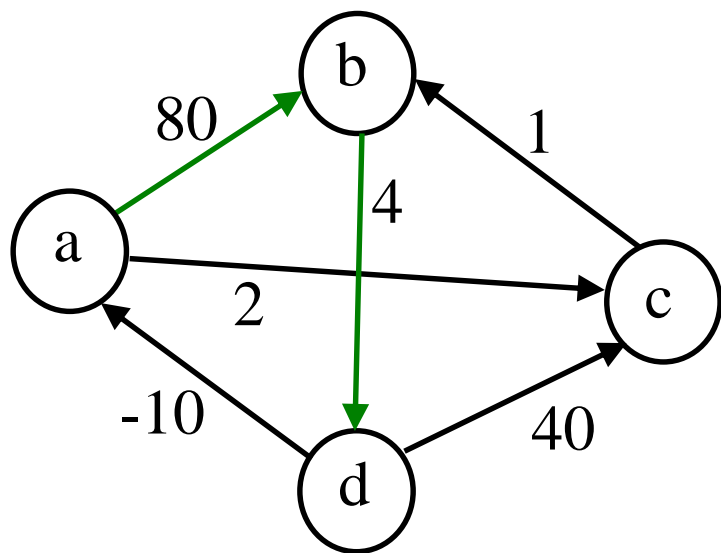
Osservazione 1: Se  $(i = k \text{ oppure } j = k)$  allora non effettuo aggiornamenti

Osservazione 2: Se  $(i = j \text{ \&\& } i \neq k)$  allora cerco possibili aggiornamenti



# Floyd–Warshall: I Esempio (7)

Seconda iter. :  $k = b$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	-8	0

MATRICE PREDECESSORI

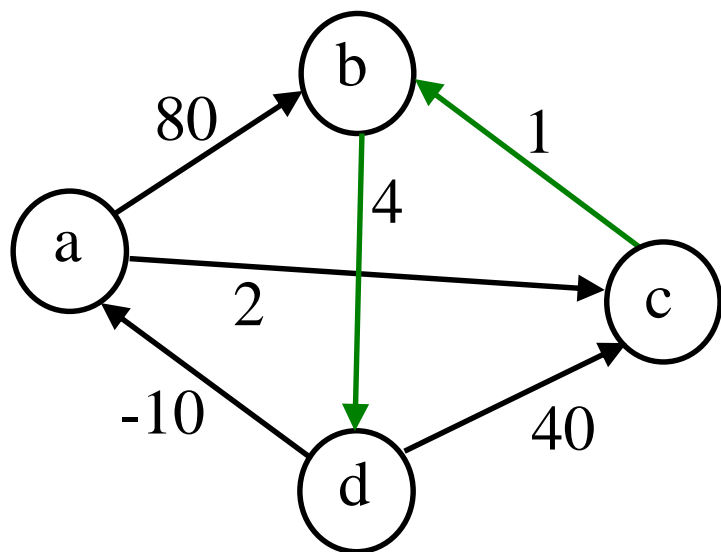
D\A→	a	b	c	d
a	a	a	a	b
b	a	b	c	b
c	a	c	c	d
d	d	a	a	d

N.B. Posso passare solo per  $a, b$

passando	k	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost					84												
old cost					inf												

# Floyd–Warshall: I Esempio (8)

Seconda iter. :  $k = b$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

MATRICE PREDECESSORI

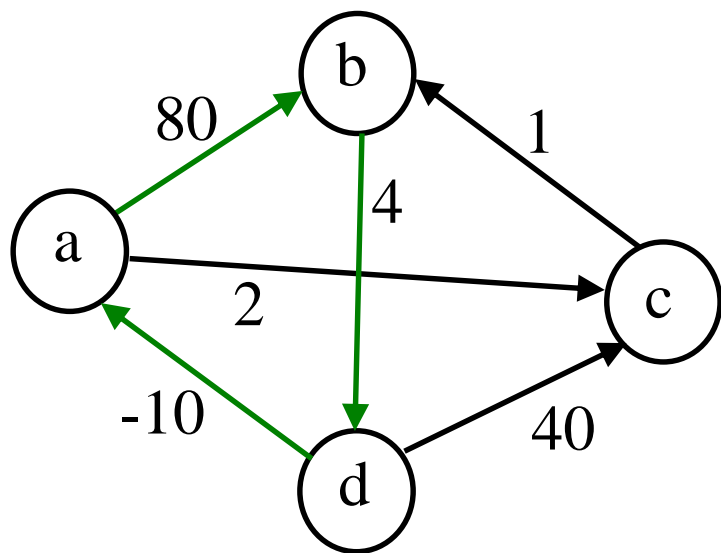
D\A→	a	b	c	d
a	a	a	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

N.B. Posso passare solo per  $a, b$

passando	k	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost					84								5				
old cost					inf								inf				

# Floyd–Warshall: I Esempio (9)

Seconda iter. :  $k = b$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

MATRICE PREDECESSORI

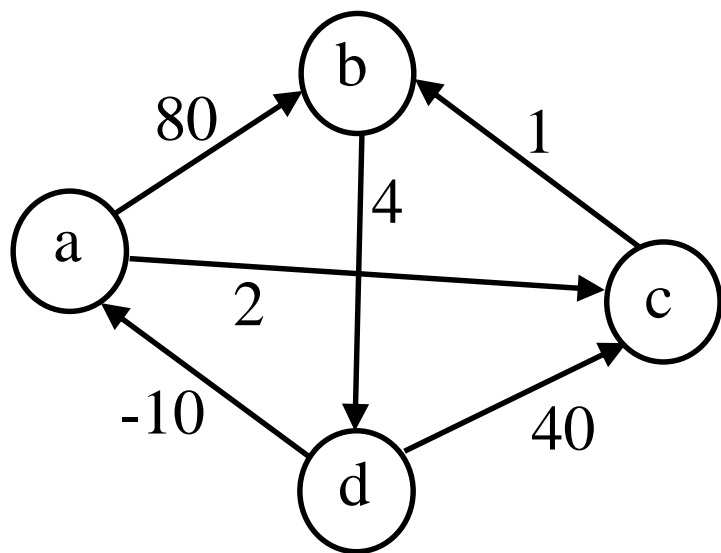
D\A→	a	b	c	d
a	a	a	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

N.B. Posso passare solo per  $a, b$

passando	k	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost					84								5				74
old cost					inf								inf				0

# Floyd–Warshall: I Esempio (10)

Seconda iter. :  $k = b$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

MATRICE PREDECESSORI

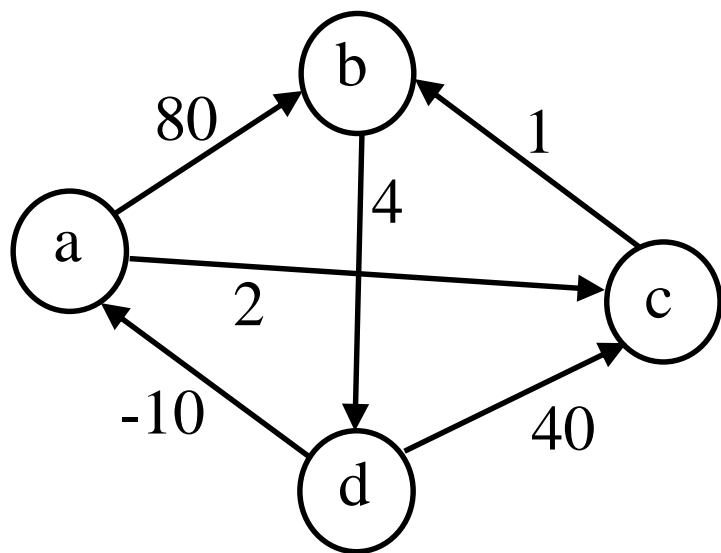
D\A→	a	b	c	d
a	a	a	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

N.B. Posso passare solo per  $a, b$

passando	k	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost					84								5				74
old cost					inf								inf				0

# Floyd–Warshall: I Esempio (11)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D/A →	a	b	c	d
↓ a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

MATRICE PREDECESSORI

D/A →	a	b	c	d
↓ a	a	a	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

N.B. Posso passare solo per  $a, b, c$

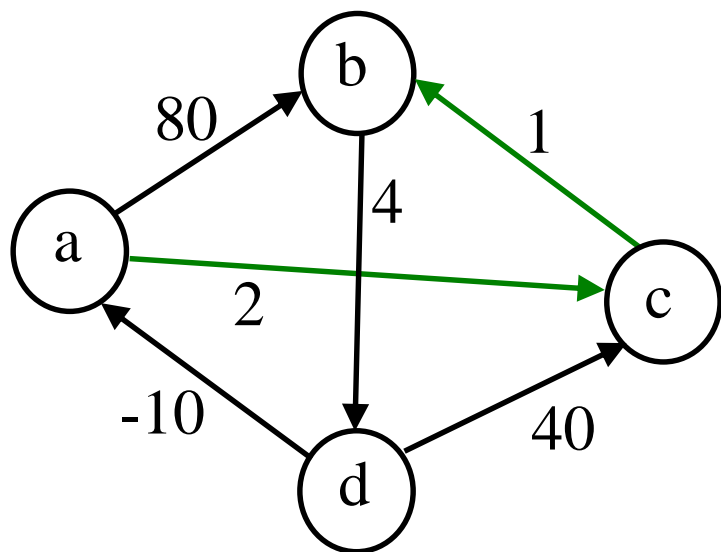
passando	k	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d

Osservazione 1: Se  $(i = k \text{ oppure } j = k)$  allora non effettuo aggiornamenti

Osservazione 2: Se  $(i = j \text{ \&\& } i \neq k)$  allora cerco possibili aggiornamenti

# Floyd–Warshall: I Esempio (12)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	3	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

MATRICE PREDECESSORI

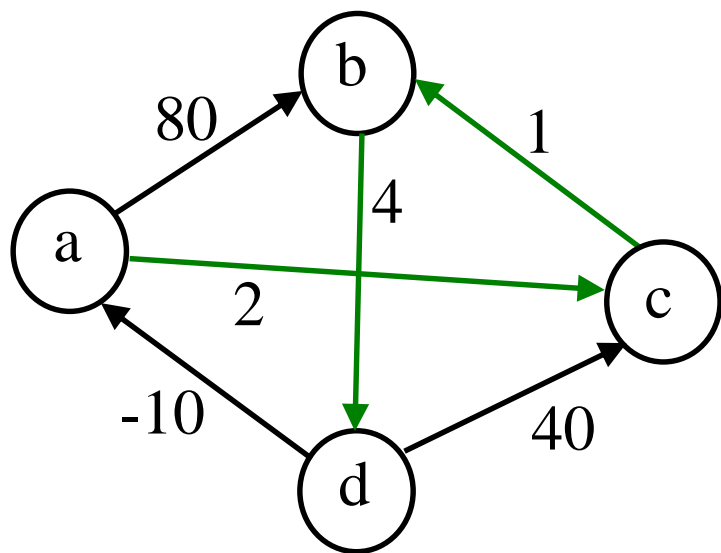
D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

N.B. Posso passare solo per  $a, b, c$

passando	k	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c
new cost			3													
old cost			80													

# Floyd–Warshall: I Esempio (13)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

MATRICE PREDECESSORI

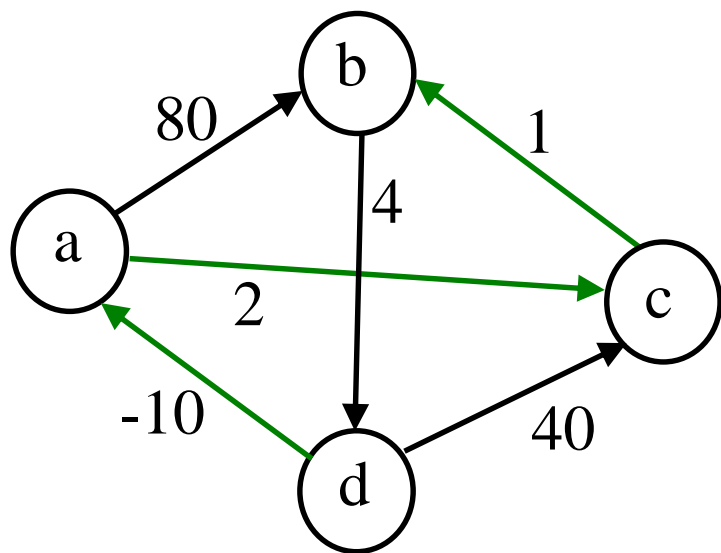
D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

N.B. Posso passare solo per  $a, b, c$

passando	k	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c
new cost			3		7											
old cost			80		84											

# Floyd–Warshall: I Esempio (14)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	0

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	d

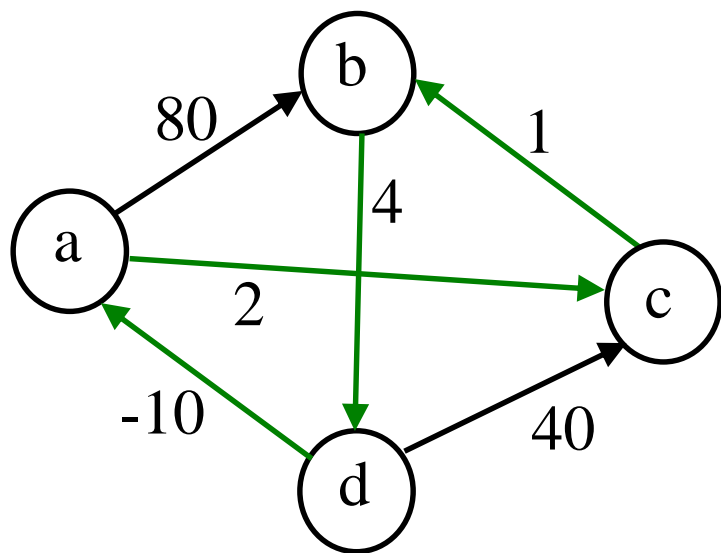
N.B. Posso passare solo per  $a, b, c$

passando	k	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost			3		7										-7		
old cost			80		84										70		



# Floyd–Warshall: I Esempio (15)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	-3

MATRICE PREDECESSORI

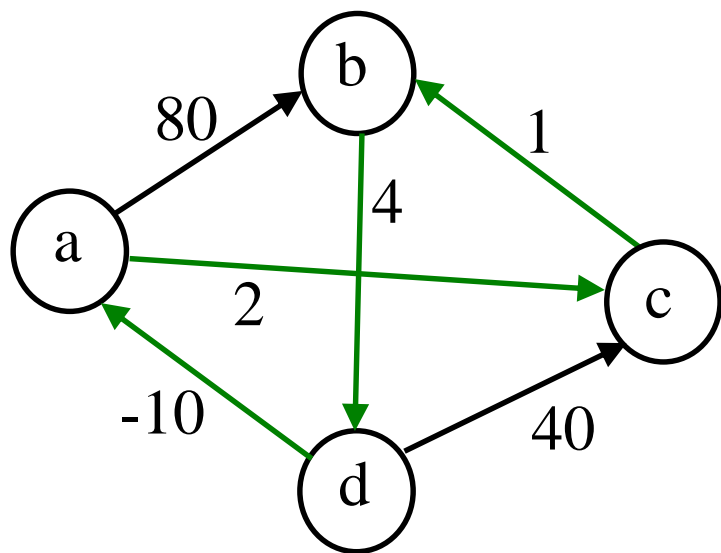
D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	<b>b</b>

N.B. Posso passare solo per  $a, b, c$

passando	k	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost			3		7										-7		-3
old cost			80		84										70		0

# Floyd–Warshall: I Esempio (16)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
↓ a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	-3

MATRICE PREDECESSORI

D\A→	a	b	c	d
↓ a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	b

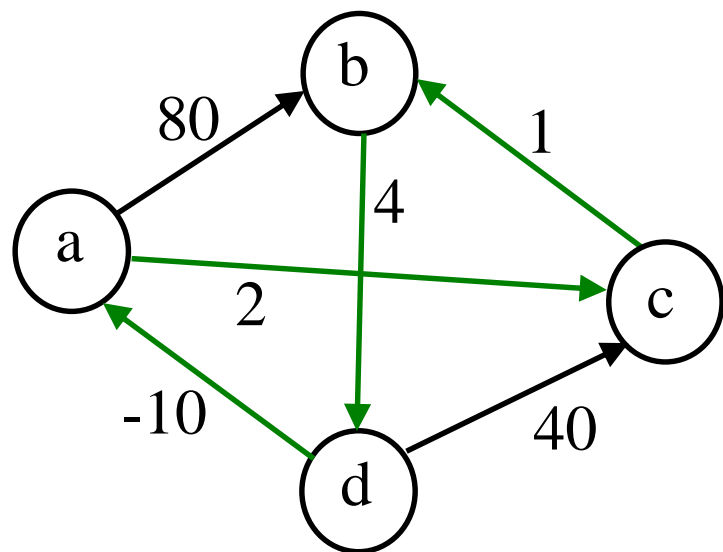
Sulla diagonale principale ho un valore  $< 0$

$\Rightarrow$  ciclo a peso negativo

Nell'esempio: ciclo passante per i nodi d, a, c, b, d di peso -3

# Floyd–Warshall: I Esempio (17)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
↓ a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	-3

MATRICE PREDECESSORI

D\A→	a	b	c	d
↓ a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	b

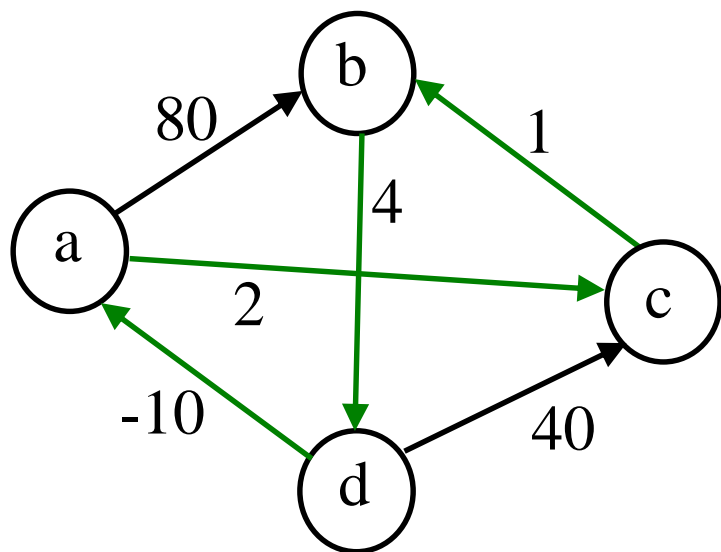
Sulla diagonale principale ho un valore  $< 0$

$\Rightarrow$  ciclo a peso negativo

Nell'esempio: ciclo passante per i nodi **d**, a, c, b, **d** di peso -3

# Floyd–Warshall: I Esempio (18)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	-3

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	b

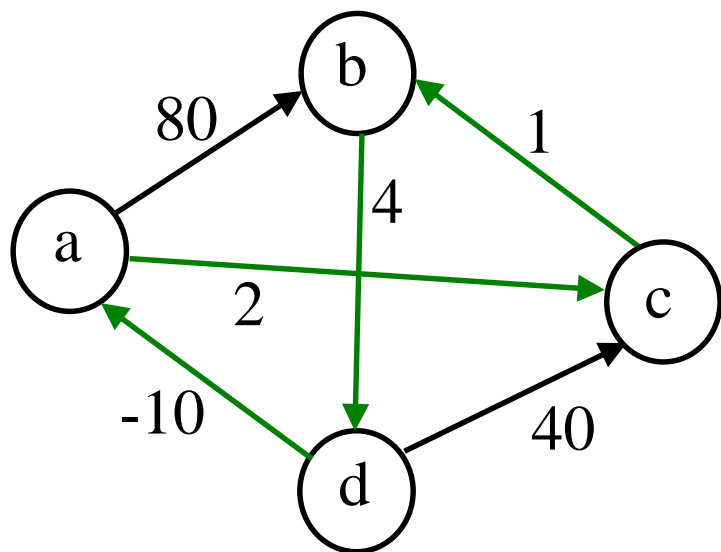
Sulla diagonale principale ho un valore  $< 0$

$\Rightarrow$  ciclo a peso negativo

Nell'esempio: ciclo passante per i nodi **d**, a, c, **b**, d di peso -3

# Floyd–Warshall: I Esempio (19)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
↓ a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	-3

MATRICE PREDECESSORI

D\A→	a	b	c	d
↓ a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	b

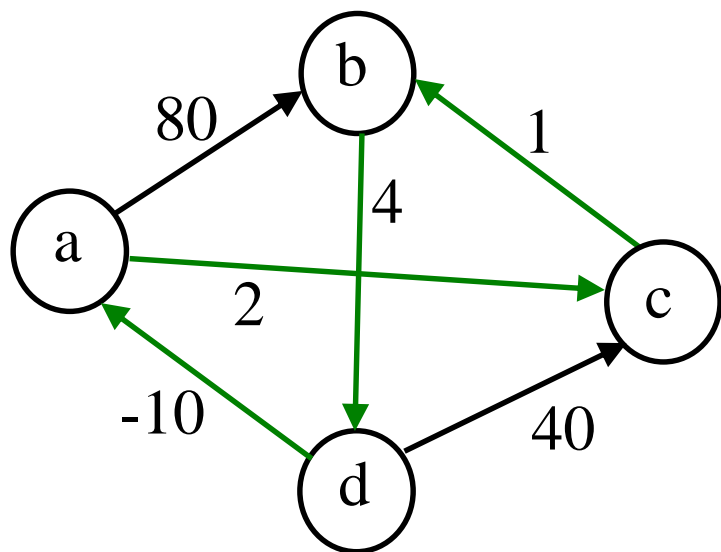
Sulla diagonale principale ho un valore  $< 0$

$\Rightarrow$  ciclo a peso negativo

Nell'esempio: ciclo passante per i nodi **d**, a, **c**, b, d di peso -3

# Floyd–Warshall: I Esempio (20)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	3	2	7
b	inf	0	inf	4
c	inf	1	0	5
d	-10	-7	-8	-3

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	b

Sulla diagonale principale ho un valore  $< 0$

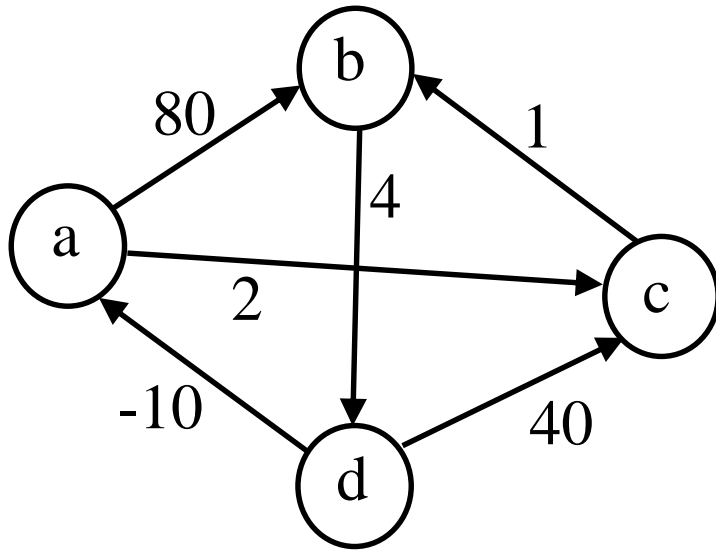
=> ciclo a peso negativo

Nell'esempio: ciclo passante per i nodi **d**, **a**, c, b, d di peso -3

# Floyd–Warshall: I Esempio (21)

## \*Metodo rapido di soluzione

Inizializzazione\*



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	inf	40	0

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	b	d	d

Con questo metodo non sarà necessario guardare il digrafo nei passi iterativi...

# Floyd–Warshall: I Esempio (22)

## \*Metodo rapido di soluzione

Prima iter. \*:  $k = a$

- La tabella risultante avrà la prima riga (in rosso) e la prima colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	inf	40	0

$-10 + 80 = 70$

$-10 + 2 = -8$

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	b	d	d



# Floyd–Warshall: I Esempio (23)

## \*Metodo rapido di soluzione

Prima iter. \*:  $k = a$

- La tabella risultante avrà la prima riga (in rosso) e la prima colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	inf	40	0

$-10 + 80 = 70$

$-10 + 2 = -8$

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	a	a	d

# Floyd–Warshall: I Esempio (24)

## \*Metodo rapido di soluzione

Seconda iter. \*:  $k = b$

- La tabella risultante avrà la seconda riga (in rosso) e la seconda colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	-8	0

$80 + 4 = 84$      $1 + 4 = 5$

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	a	a	d
b	a	b	c	b
c	a	c	c	d
d	d	a	a	d

# Floyd–Warshall: I Esempio (25)

## \*Metodo rapido di soluzione

Seconda iter. \*:  $k = b$

- La tabella risultante avrà la seconda riga (in rosso) e la seconda colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	inf
b	inf	0	inf	4
c	inf	1	0	inf
d	-10	70	-8	0

$80 + 4 = 84$      $1 + 4 = 5$

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	a	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

# Floyd–Warshall: I Esempio (26)

## \*Metodo rapido di soluzione

Terza iter. \*:  $k = c$

- La tabella risultante avrà la terza riga (in rosso) e la terza colonna (in rosso) uguali a quelle della tabella precedente
- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

$$2+1=3$$

$$-8+1=-7$$

$$2+5=7$$

$$-8+5=-3$$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	a	a	b
b	a	b	c	b
c	a	c	c	b
d	d	a	a	d

# Floyd–Warshall: I Esempio (27)

## \*Metodo rapido di soluzione

Terza iter.\*:  $k = c$

- La tabella risultante avrà la terza riga (in rosso) e la terza colonna (in rosso) uguali a quelle della tabella precedente
- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
a	0	80	2	84
b	inf	0	inf	4
c	inf	1	0	5
d	-10	70	-8	0

$$2+1=3$$

$$-8+1=-7$$

$$2+5=7$$

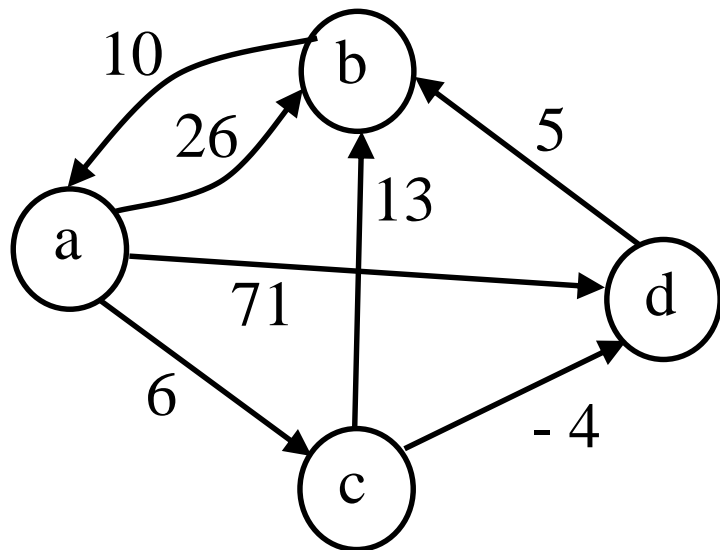
$$-8+5=-3$$

MATRICE PREDECESSORI

D\A→	a	b	c	d
a	a	c	a	b
b	a	b	c	b
c	a	c	c	b
d	d	c	a	b

# Floyd–Warshall: II Esempio (1)

Inizializzazione



MATRICE COSTI DEI PERCORSI

<b>D\A→</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>
<b>a</b>	0	26	6	71
<b>b</b>	10	0	inf	inf
<b>c</b>	inf	13	0	-4
<b>d</b>	inf	5	inf	0

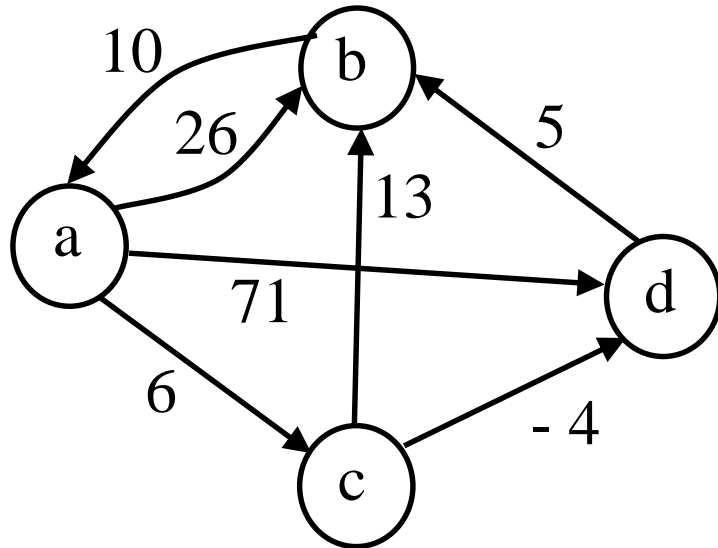
MATRICE PREDECESSORI

<b>D\A→</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>
<b>a</b>	a	a	a	a
<b>b</b>	b	b	c	d
<b>c</b>	a	c	c	c
<b>d</b>	a	d	c	d

Sulla diagonale principale della matrice dei percorsi iniziali ho tutti valori nulli

# Floyd–Warshall: II Esempio (2)

Prima iter. :  $k = a$



MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
↓ a	0	26	6	71
b	10	0	16	81
c	inf	13	0	-4
d	inf	5	inf	0

MATRICE PREDECESSORI

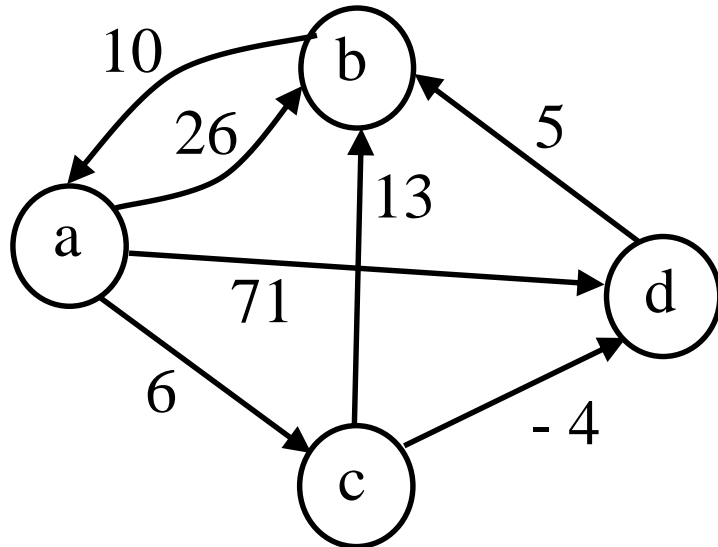
D\A →	a	b	c	d
↓ a	a	a	a	a
b	b	b	a	a
c	a	c	c	c
d	a	d	c	d

N.B. Posso passare solo per  $a$

passando	k	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost							36	16	81								
old cost							10	inf	inf								

# Floyd–Warshall: II Esempio (3)

Seconda iter. :  $k = b$



MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
↓ a	0	26	6	71
b	10	0	16	81
c	23	13	0	-4
d	15	5	21	0

MATRICE PREDECESSORI

D\A →	a	b	c	d
↓ a	a	a	a	a
b	b	b	a	a
c	b	c	c	c
d	b	d	a	d

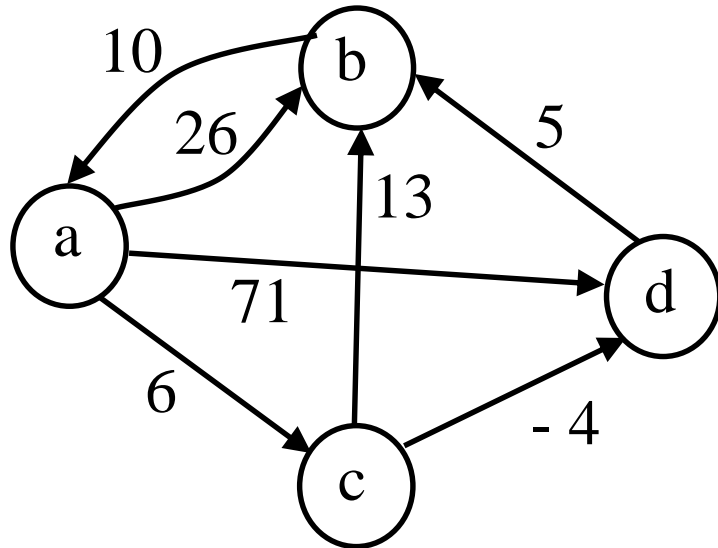
N.B. Posso passare solo per  $a, b$

passando	k	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost		36								23		29	94	15		21	86
old cost		0								inf		0	-4	inf		inf	0



# Floyd–Warshall: II Esempio (4)

Terza iter. :  $k = c$



MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
↓ a	0	19	6	2
b	10	0	16	12
c	23	13	0	-4
d	15	5	21	0

MATRICE PREDECESSORI

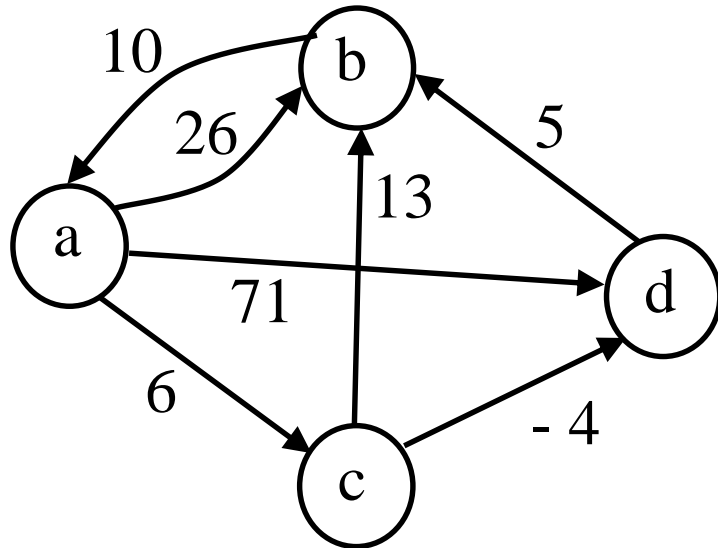
D\A →	a	b	c	d
↓ a	a	c	a	c
b	b	b	a	c
c	b	c	c	c
d	b	d	a	d

N.B. Posso passare solo per  $a, b, c$

passando	k	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
new cost		29	19		2		29		12								17
old cost		0	26		71		0		81								0

# Floyd–Warshall: II Esempio (5)

Quarta iter. :  $k = d$



MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
↓ a	0	7	6	2
b	10	0	16	12
c	11	1	0	-4
d	15	5	21	0

MATRICE PREDECESSORI

D\A →	a	b	c	d
↓ a	a	d	a	c
b	b	b	a	c
c	b	d	c	c
d	b	d	a	d

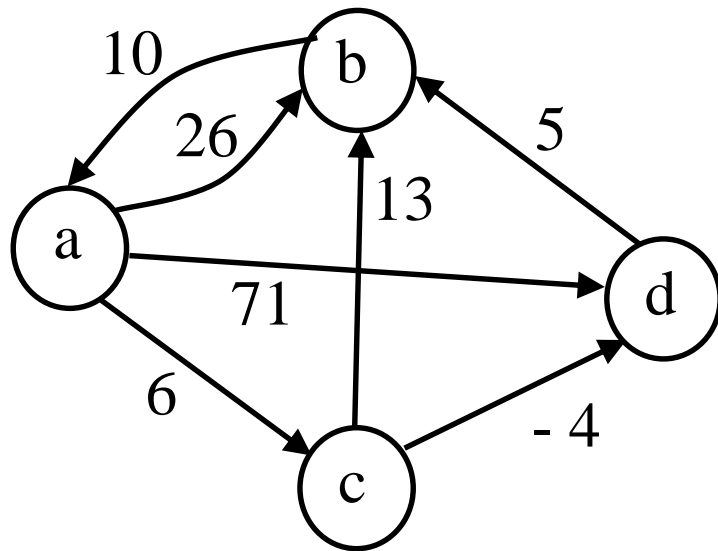
N.B. Posso passare per  $a, b, c, d$

passando	k	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d
da	i	a	a	a	a	b	b	b	b	c	c	c	c	d	d	d
a	j	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c
new cost		17	7				17			11	1	17				
old cost		0	19				0			23	13	0				

# Floyd–Warshall: II Esempio (6)

## \*Metodo rapido di soluzione

Inizializzazione\*



MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d
↓ a	0	26	6	71
b	10	0	inf	inf
c	inf	13	0	-4
d	inf	5	inf	0

MATRICE PREDECESSORI

D\A→	a	b	c	d
↓ a	a	a	a	a
b	b	b	c	d
c	a	c	c	c
d	a	d	c	d

Con questo metodo non sarà necessario guardare il digrafo nei passi iterativi...

# Floyd–Warshall: II Esempio (7)

## \*Metodo rapido di soluzione

Prima iter. \*:  $k = a$

- La tabella risultante avrà la prima riga (in rosso) e la prima colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	26	6	71
b	10	0	inf	inf
c	inf	13	0	-4
d	inf	5	inf	0

$10+6=16$

$10+71=81$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	a	a	a
b	b	b	c	d
c	a	c	c	c
d	a	d	c	d

# Floyd–Warshall: II Esempio (8)

## \*Metodo rapido di soluzione

Prima iter. \*:  $k = a$

- La tabella risultante avrà la prima riga (in rosso) e la prima colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	26	6	71
b	10	0	inf	inf
c	inf	13	0	-4
d	inf	5	inf	0

$10 + 6 = 16$        $10 + 71 = 81$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	a	a	a
b	b	b	a	a
c	a	c	c	c
d	a	d	c	d

# Floyd–Warshall: II Esempio (9)

## \*Metodo rapido di soluzione

Seconda iter. \*:  $k = b$

- La tabella risultante avrà la seconda riga (in rosso) e la seconda colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	26	6	71
b	10	0	16	81
c	inf	13	0	-4
d	inf	5	inf	0

$13 + 10 = 23$        $5 + 10 = 15$        $5 + 16 = 21$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	a	a	a
b	b	b	a	a
c	a	c	c	c
d	a	d	c	d

# Floyd–Warshall: II Esempio (10)

## \*Metodo rapido di soluzione

Seconda iter. \*:  $k = b$

- La tabella risultante avrà la seconda riga (in rosso) e la seconda colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	26	6	71
b	10	0	16	81
c	inf	13	0	-4
d	inf	5	inf	0

$13 + 10 = 23$        $5 + 10 = 15$        $5 + 16 = 21$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	a	a	a
b	b	b	a	a
c	b	c	c	c
d	b	d	a	d

# Floyd–Warshall: II Esempio (11)

## \*Metodo rapido di soluzione

Terza iter. \*:  $k = c$

- La tabella risultante avrà la terza riga (in rosso) e la terza colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	26	6	71
b	10	0	16	81
c	23	13	0	-4
d	15	5	21	0

$6 + 13 = 19$        $6 - 4 = 2$        $16 - 4 = 12$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	a	a	a
b	b	b	a	a
c	b	c	c	c
d	b	d	a	d



# Floyd–Warshall: II Esempio (12)

## \*Metodo rapido di soluzione

Terza iter.\*:  $k = c$

- La tabella risultante avrà la terza riga (in rosso) e la terza colonna (in rosso) uguali a quelle della tabella precedente

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	26	6	71
b	10	0	16	81
c	23	13	0	-4
d	15	5	21	0

$6 + 13 = 19$        $6 - 4 = 2$        $16 - 4 = 12$

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	c	a	c
b	b	b	a	c
c	b	c	c	c
d	b	d	a	d

# Floyd–Warshall: II Esempio (13)

## \*Metodo rapido di soluzione

Quarta iter. \*:  $k = d$

- La tabella risultante avrà la quarta riga (in rosso) e la quarta colonna (in rosso) uguali a quelle della tabella precedente

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	19	6	2
b	10	0	16	12
c	23	13	0	-4
d	15	5	21	0

$-4 + 15 = 11$        $-4 + 5 = 1$        $2 + 5 = 7$

- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	c	a	c
b	b	b	a	c
c	b	c	c	c
d	b	d	a	d

# Floyd–Warshall: II Esempio (14)

## \*Metodo rapido di soluzione

Quarta iter.\*:  $k = d$

- La tabella risultante avrà la quarta riga (in rosso) e la quarta colonna (in rosso) uguali a quelle della tabella precedente

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d
a	0	19	6	2
b	10	0	16	12
c	23	13	0	-4
d	15	5	21	0

$-4 + 15 = 11$        $-4 + 5 = 1$        $2 + 5 = 7$

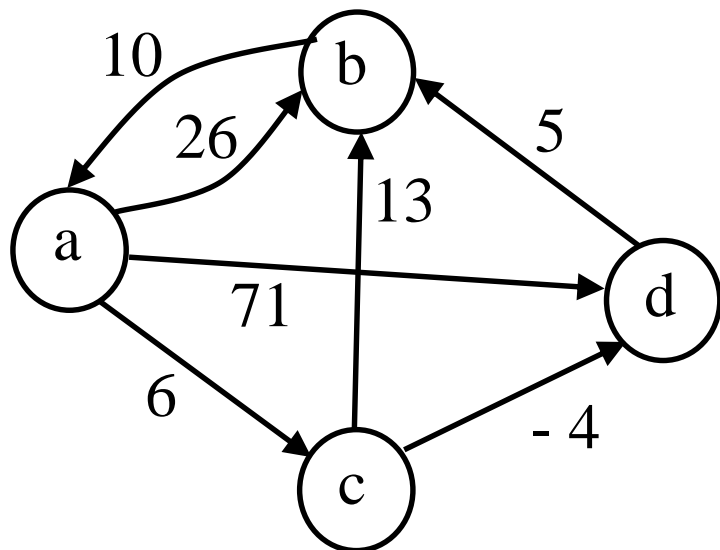
- Per tutti gli altri elementi basta fare il minimo tra l'elemento considerato e la somma delle proiezioni sulla riga rossa e sulla colonna rossa

MATRICE PREDECESSORI

D\A →	a	b	c	d
a	a	d	a	c
b	b	b	a	c
c	b	d	c	c
d	b	d	a	d

# Floyd–Warshall: II Esempio (15)

$k = d$



MATRICE COSTI DEI PERCORSI

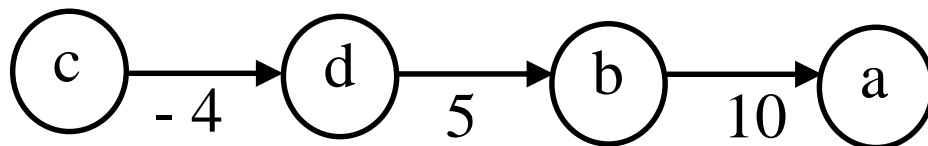
<b>D\A →</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>
<b>a ↓</b>	0	7	6	2
<b>b</b>	10	0	16	12
<b>c</b>	11	1	0	-4
<b>d</b>	15	5	21	0

MATRICE PREDECESSORI

<b>D\A →</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>
<b>a ↓</b>	a	d	a	c
<b>b</b>	b	b	a	c
<b>c</b>	b	d	c	c
<b>d</b>	b	d	a	d

Domanda: Mostrare l'albero dei cammini minimi centrato nel nodo c.

Risposta: L'albero dei cammini minimi dal nodo c è il seguente (di peso 11):



- Da **c** ad **a** passo per **b**
- Da **c** ad **b** passo per **d**
- Da **c** ad **d** passo per **c**

# Floyd–Warshall: III Esempio (1)

Si parte dalla fine della terza iterazione con le matrici:

MATRICE COSTI DEI PERCORSI

<b>D\A→</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>
<b>a</b>	0	inf	2	5	1
<b>b</b>	inf	0	1	4	inf
<b>c</b>	inf	inf	0	3	inf
<b>d</b>	1	inf	3	0	2
<b>e</b>	inf	2	3	6	0

MATRICE PREDECESSORI

<b>D\A→</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>
<b>a</b>	a	a	a	c	a
<b>b</b>	b	b	b	c	b
<b>c</b>	c	c	c	c	c
<b>d</b>	d	d	a	d	a
<b>e</b>	e	e	b	c	e

Domanda (1): Effettuare i rimanenti passi dell'algo (se possibile)

Domanda (2): Mostrate i cammini minimi da **e** a **d** e da **c** a **a**

# Floyd–Warshall: III Esempio (2)

Punto di partenza dell'esercizio:

MATRICE COSTI DEI PERCORSI

D\A→	a	b	c	d	e
a	0	inf	2	5	1
b	inf	0	1	4	inf
c	inf	inf	0	3	inf
d	1	inf	3	0	2
e	inf	2	3	6	0

MATRICE PREDECESSORI

D\A→	a	b	c	d	e
a	a	b	a	c	a
b	a	b	b	c	e
c	a	b	c	c	e
d	d	b	a	d	a
e	a	e	b	c	e

Fine della terza iterazione dell'algo:

*effettuate le iterazioni  $k = a$ ,  $k = b$ ,  $k = c$*

# Floyd–Warshall: III Esempio (3)

Quarta iterazione ( $k = d$ ):

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
a	0	inf	2	5	1
b	inf	0	1	4	inf
c	inf	inf	0	3	inf
d	1	inf	3	0	2
e	inf	2	3	6	0

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
a	0	inf	2	5	1
b	5	0	1	4	6
c	4	inf	0	3	5
d	1	inf	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
a	a	b	a	c	a
b	a	b	b	c	e
c	a	b	c	c	e
d	d	b	a	d	a
e	a	e	b	c	e

MATRICE PREDECESSORI

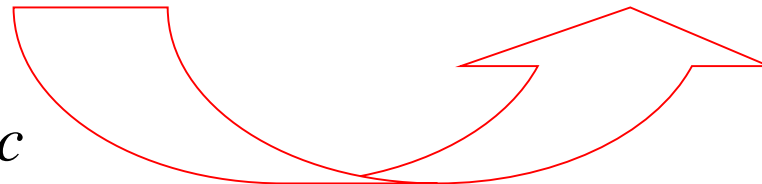
D\A →	a	b	c	d	e
a	a	b	a	c	a
b	d	b	b	c	a
c	d	b	c	c	a
d	d	b	a	d	a
e	d	e	b	c	e

Terzo passo:

$k = a, k = b, k = c$

Quarto passo:

$k = d$



# Floyd–Warshall: III Esempio (4)

Quinta iterazione ( $k = e$ ):

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
a	0	inf	2	5	1
b	5	0	1	4	6
c	4	inf	0	3	5
d	1	inf	3	0	2
e	7	2	3	6	0

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
a	0	3	2	5	1
b	5	0	1	4	6
c	4	7	0	3	5
d	1	4	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
a	a	b	a	c	a
b	d	b	b	c	a
c	d	b	c	c	a
d	d	b	a	d	a
e	d	e	b	c	e

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
a	a	e	a	c	a
b	d	b	b	c	a
c	d	e	c	c	a
d	d	e	a	d	a
e	d	e	b	c	e

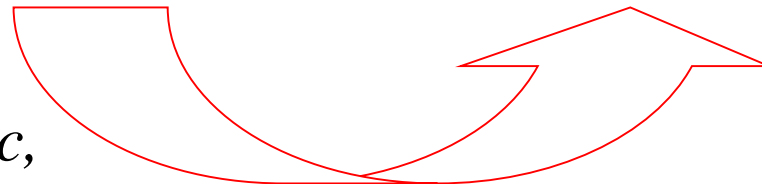
Quarto passo:

$k = a, k = b, k = c,$

$k = d$

Quinto passo:

$k = e$





# Floyd–Warshall: III Esempio (5)

Cammini orientati minimi nella soluzione ottima:

Domanda (2):

Mostrate i

cammini

orientati

minimi da

**e a d**

e da

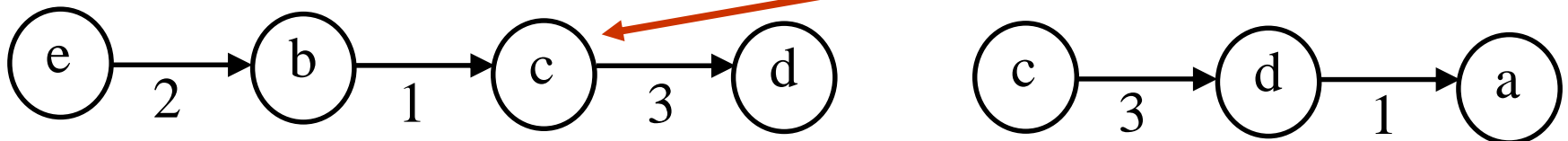
**c a a**

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
↓ a	0	3	2	5	1
b	5	0	1	4	6
c	4	7	0	3	5
d	1	4	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
↓ a	a	e	a	c	a
b	d	b	b	c	a
c	d	e	c	c	a
d	d	e	a	d	a
e	d	e	b	c	e



# Floyd–Warshall: III Esempio (6)

Cammini orientati minimi nella soluzione ottima:

Domanda (2):

Mostrate i

cammini

orientati

minimi da

**e a d**

e da

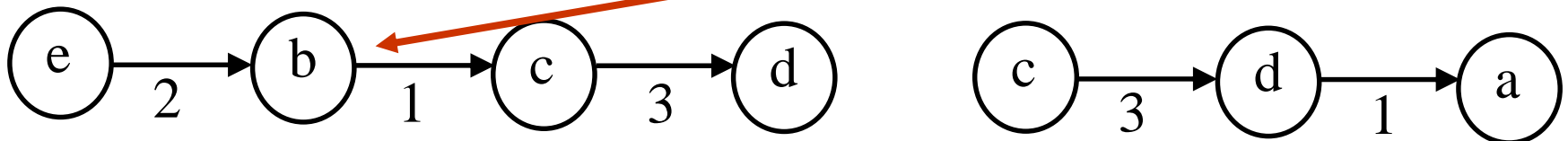
**c a a**

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
↓ a	0	3	2	5	1
b	5	0	1	4	6
c	4	7	0	3	5
d	1	4	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
↓ a	a	e	a	c	a
b	d	b	b	c	a
c	d	e	c	c	a
d	d	e	a	d	a
e	d	e	b	c	e



# Floyd–Warshall: III Esempio (7)

Cammini orientati minimi nella soluzione ottima:

Domanda (2):

Mostrate i

cammini

orientati

minimi da

**e a d**

e da

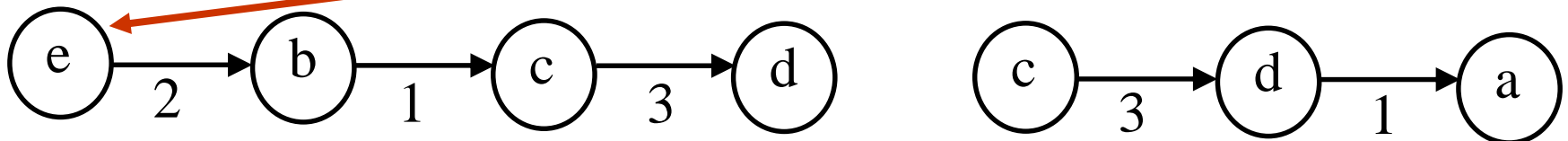
**c a a**

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
↓ a	0	3	2	5	1
b	5	0	1	4	6
c	4	7	0	3	5
d	1	4	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
↓ a	a	e	a	c	a
b	d	b	b	c	a
c	d	e	c	c	a
d	d	e	a	d	a
e	d	e	b	c	e



# Floyd–Warshall: III Esempio (8)

Cammini orientati minimi nella soluzione ottima:

Domanda (2):

Mostrate i

cammini

orientati

minimi da

**e a d**

e da

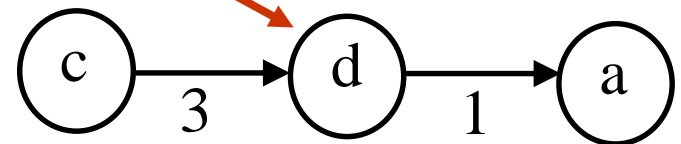
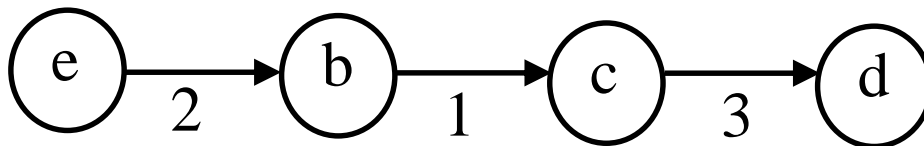
**c a a**

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
↓ a	0	3	2	5	1
b	5	0	1	4	6
c	4	7	0	3	5
d	1	4	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
↓ a	a	e	a	c	a
b	d	b	b	c	a
c	d	e	c	c	a
d	d	e	a	d	a
e	d	e	b	c	e



# Floyd–Warshall: III Esempio (9)

Cammini orientati minimi nella soluzione ottima:

Domanda (2):

Mostrate i

cammini

orientati

minimi da

**e a d**

e da

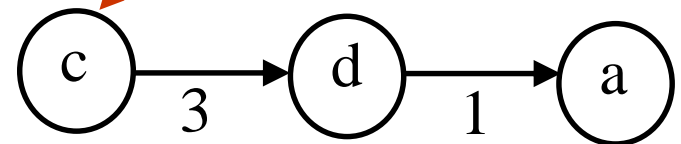
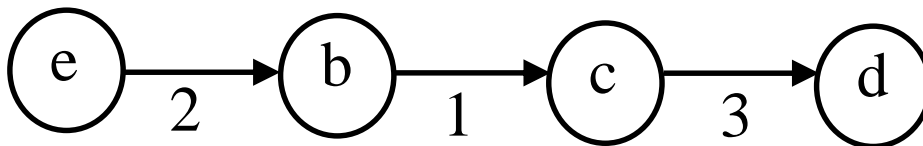
**c a a**

MATRICE COSTI DEI PERCORSI

D\A →	a	b	c	d	e
↓ a	0	3	2	5	1
b	5	0	1	4	6
c	4	7	0	3	5
d	1	4	3	0	2
e	7	2	3	6	0

MATRICE PREDECESSORI

D\A →	a	b	c	d	e
↓ a	a	e	a	c	a
b	d	b	b	c	a
c	d	e	c	c	a
d	d	e	a	d	a
e	d	e	b	c	e



# Pianificazione di progetti (1)

- Un progetto complesso può essere visto come un insieme di *attività*  $\alpha_i$ , tra loro correlate, per raggiungere un determinato scopo
- Tra attività sono presenti delle *relazioni (vincoli) di precedenza* che impongono che alcune attività non possano cominciare prima che altre attività siano state terminate

Esempio: se l'attività  $\alpha_j$  può essere eseguita dopo il completamento dell'attività  $\alpha_i$ , questo verrà indicato come  $\alpha_j > \alpha_i$

- Ad ogni  $\alpha_i$  è associata una durata *di (tempo di processamento)* nota a priori e deterministica.
- Per ogni  $\alpha_i$  è possibile associare un *tempo di inizio*  $t_i$  e un *tempo di completamento (terminazione)*  $T_i$

# Pianificazione di progetti (2)

Formalmente:

Sia un progetto  $P$  definito da un insieme di attività (nodi)  $N$  e un insieme di relazioni di precedenza (archi orientati)  $A$ , il digrafo  $G = (N, A)$  sarà allora una rappresentazione del progetto come una *rete con attività su nodi*.  $G$  è un modello del progetto  $P$ .

Le durate delle attività  $d_i$  sono rappresentate come i pesi sugli archi.

Tutti gli archi uscenti dal nodo che rappresenta  $\alpha_i$  hanno peso  $d_i$   
 $\Rightarrow$  ovvero  $c_{ij} = d_i$ , per ogni arco  $(i, j)$  in  $\delta^+(i)$

Funzione obiettivo: Determinare l'istante di inizio di ogni attività e quindi determinare il tempo di complemento del progetto

# Pianificazione di progetti (3)

Definizioni basilari:

*Attività fittizie*: Un'attività fittizia  $j$  viene rappresentata da un nodo  $j$  con durata nulla  $d_j = 0$ . Le attività fittizie sono utili per poter rappresentare, ad esempio, l'inizio e il termine di un progetto.

Indicheremo convenzionalmente con:

- 0 : l'attività di inizio progetto, chiamata in  $G$  *nodo sorgente*
- \* : l'attività di fine progetto, chiamata in  $G$  *nodo pozzo*

Per il nodi 0 (\*) non è preceduto (seguito) da alcun nodo

$t[*]$  è la durata del progetto, ovvero il suo tempo di completamento



# Pianificazione di progetti (4)

Riassumendo:

- Scopo della pianificazione dei progetti è quello di stabilire un tempo minimo di inizio e un tempo massimo di inizio per ogni attività in modo tale da minimizzare il tempo di completamento complessivo del progetto
- Questi tempi di inizio minimo e massimo di ogni attività devono essere tali da rispettare (1) le relazioni di precedenza tra le attività e (2) le durate delle attività stesse
- In altre parole se una attività  $\alpha_i$  richiede una durata  $d_i$  allora dovrà valere la disuguaglianza  $t_j \geq t_i + d_i$  per ogni  $\alpha_j$  tale che  $\alpha_j > \alpha_i$  (ovvero  $\alpha_j$  segue  $\alpha_i$ )

# Pianificazione di progetti (5)

## Esempio:

Progetto di sviluppo software composto da 4 attività:

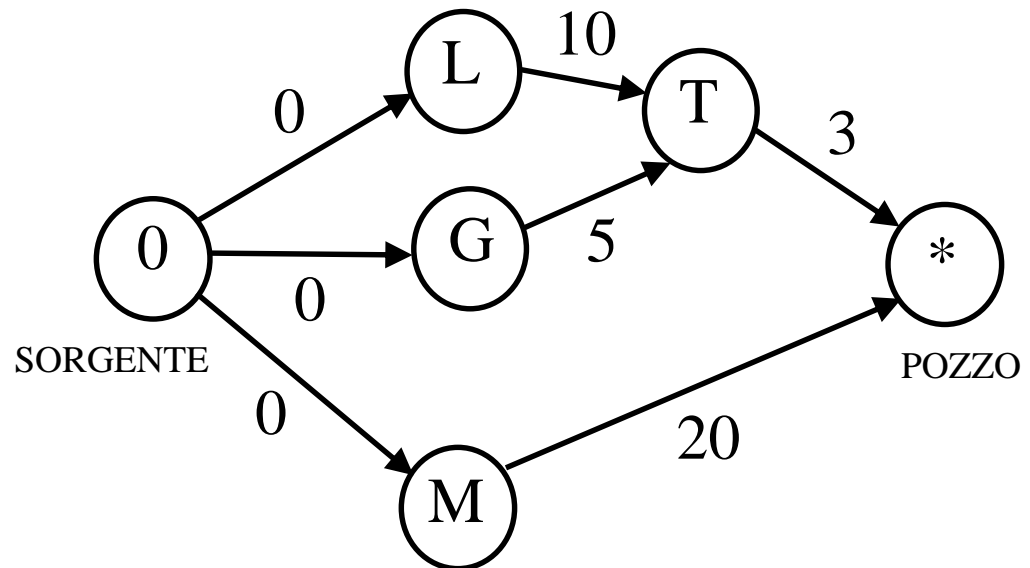
Logica, GUI (Graphical User Interface), Test, Manuale

Le attività hanno la seguente durata: 10, 5, 3, 20

Esistono dei vincoli di precedenza: Logica e GUI precedono Test

Obiettivo: minimizzare il tempo di completamento del progetto

Digrafo corrispondente  
al problema da risolvere:



Situazione nodo  $T$ :

$$t_T \geq t_L + 10$$

$$t_T \geq t_G + 5$$

# Pianificazione di progetti (6)

Ulteriori definizioni:

Earlier Starting Time ( $EST_i$ ) : il minimo istante d'inizio per  $\alpha_i$

Latest Starting Time ( $LST_i$ ): l'ultimo istante utile in cui  $\alpha_i$  può iniziare senza procurare un ritardo nella durata del progetto

Attività critica : se non può essere rallentata senza causare un ritardo nel progetto (ovvero se vale  $EST_i = LST_i$ )

$EST_i - LST_i$  : lo slittamento o slack ( $si$ ) di  $\alpha_i$ , ovvero quanto è possibile ritardare l'inizio dell'attività senza causare rallentamenti

Cammino critico : cammino orientato di costo massimo da 0 a \*.  
Il cammino critico è composto da attività (critiche) con slack nullo.

In generale, un progetto è sempre individuato uno (o più) cammini critici che congiungono il nodo sorgente al nodo pozzo.

# Il problema del cammino critico (1)

Al fine di risolvere un problema di pianificazione di progetti è necessario essere in grado di riuscire a calcolare su un digrafo che modella il progetto in esame il cammino critico, ovvero il cammino orientato di costo massimo tra il nodo sorgente e il nodo pozzo.

Si osservi come un digrafo pesato  $G$  che modella la pianificazione del progetto, sia necessariamente aciclico, ovvero il digrafo pesato non può contenere **cicli** (i.e. attività che precedono se stesse), indipendentemente dalla loro lunghezza.

Infatti un ciclo rappresenterebbe una situazione non ammissibile e non realizzabile in pratica.

# Il problema del cammino critico (2)

Calcolare il cammino orientato di costo massimo in un digrafo  $G$  è equivalente a calcolare il cammino orientato minimo in un digrafo dove tutte le durate delle attività siano state moltiplicate per  $-1$ .

In generale, il digrafo  $G$  può avere archi di peso sia positivo che negativo, per cui non sarà sempre possibile risolverlo applicando l'algoritmo di Dijkstra.

Mentre l'algoritmo di Floyd–Warshall è in grado di risolvere il problema (essendo il grafo aciclico) con una complessità di  $O(n^3)$ .

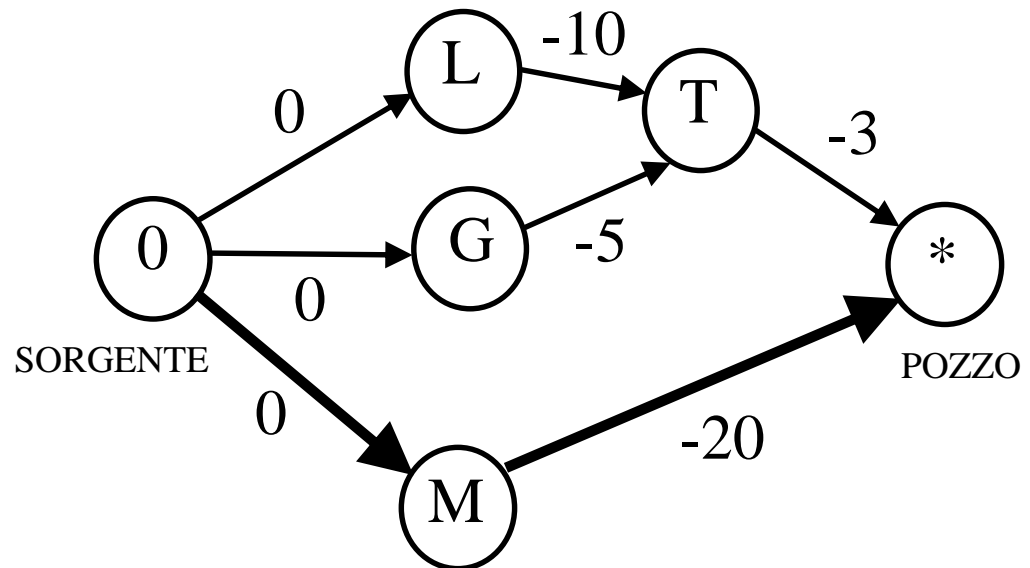
Un altro approccio possibile, e maggiormente efficiente, permette di trovare in un digrafo aciclico il cammino orientato minimo, eseguendo  $O(m)$  operazioni. Infatti, in un digrafo aciclico è sempre possibile individuare un ordinamento topologico dei nodi.

# Il problema del cammino critico (3)

Calcolare il cammino orientato di costo massimo in un digrafo  $G$  è equivalente a calcolare il cammino orientato minimo in un digrafo dove tutte le durate delle attività siano state moltiplicate per  $-1$ .

Risolviamo l'esempio precedente con Floyd-Warshall:

Cammino orientato  
minimo:  
[0, M, \*] di peso -20



# Il problema del cammino critico (4)

Ordinamento topologico: è un ordinamento dei nodi tale che se il nodo  $i$  precede il nodo  $j$ , allora non esiste in  $G$  un percorso orientato che congiunge  $i$  con  $j$ .

Dato un digrafo aciclico, è possibile calcolare più ordinamenti topologici.

Algoritmo OT assegna un indice

di ordinamento a ogni nodo.

A ogni passo si individua un

nodo senza archi entranti

( $\delta^-(i) = 0$ ) assegnandogli il

valore corrente di ordinamento.

Poi si incrementa l'ordinamento

e si rimuovono tutti gli archi

uscenti da quel nodo del digrafo.

Questo passo viene ripetuto fino

all'ordinamento di tutti i nodi.

**Algoritmo di Ordinamento Topologico**

$O[i] = 0, \forall i \in V;$

for (  $i = 1; i \leq n; i++$  )

{

    Trova  $k$  tale che  $\delta^-(k) = 0;$

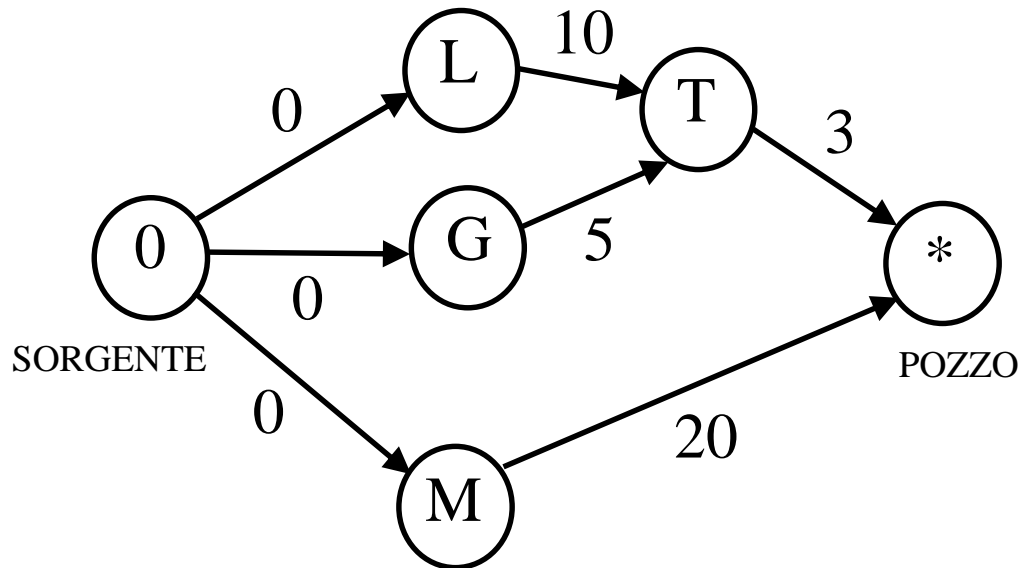
$O[k] = i;$

$G = G - \{k\};$

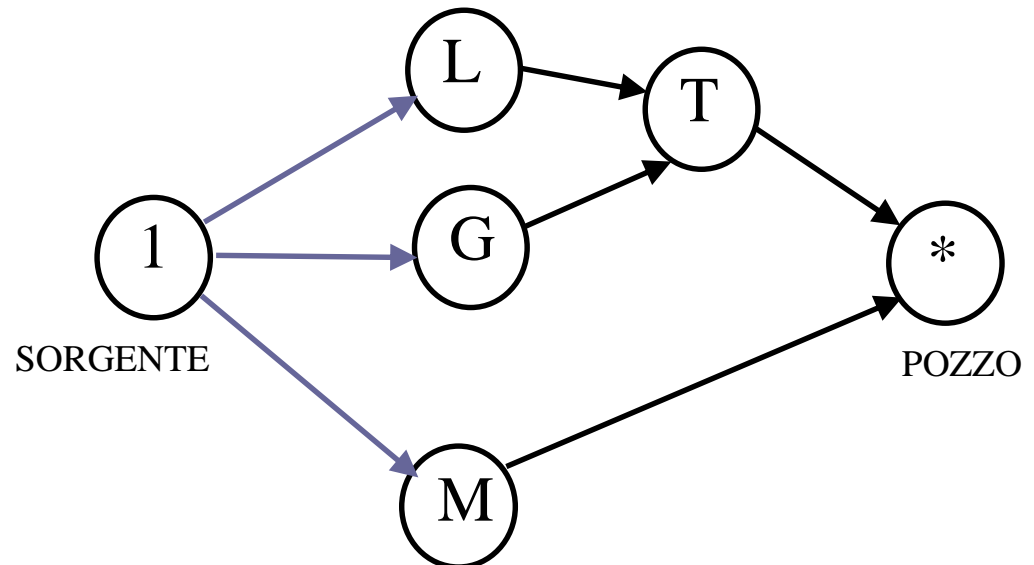
}

# Il problema del cammino critico (5)

Calcoliamo un ordinamento topologico per l'esempio precedente:



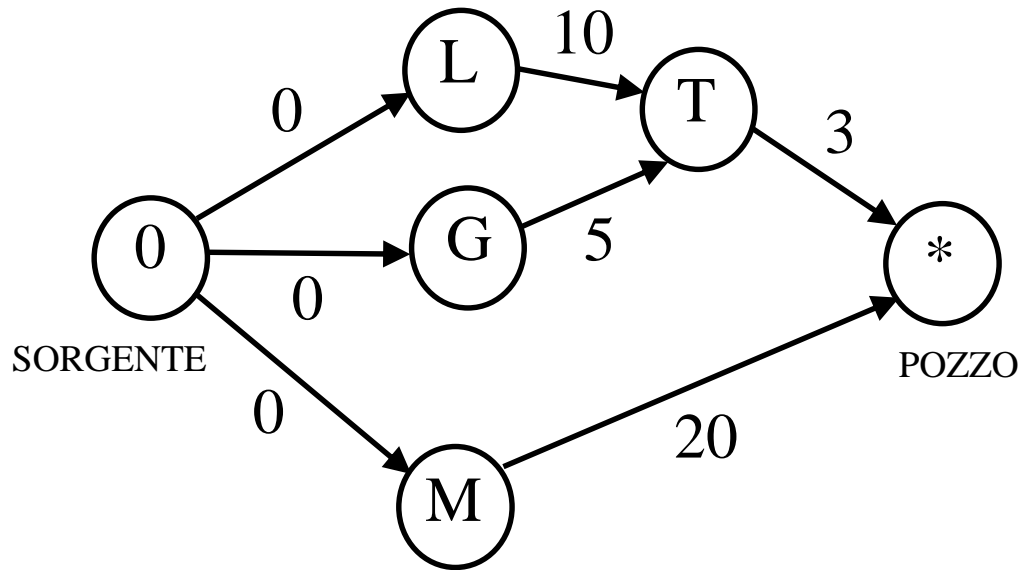
- Scelgo il nodo 0 perché è l'unico senza ha archi entranti
- Gli assegno il primo indice
- Elimino gli archi uscenti dal nodo sorgente



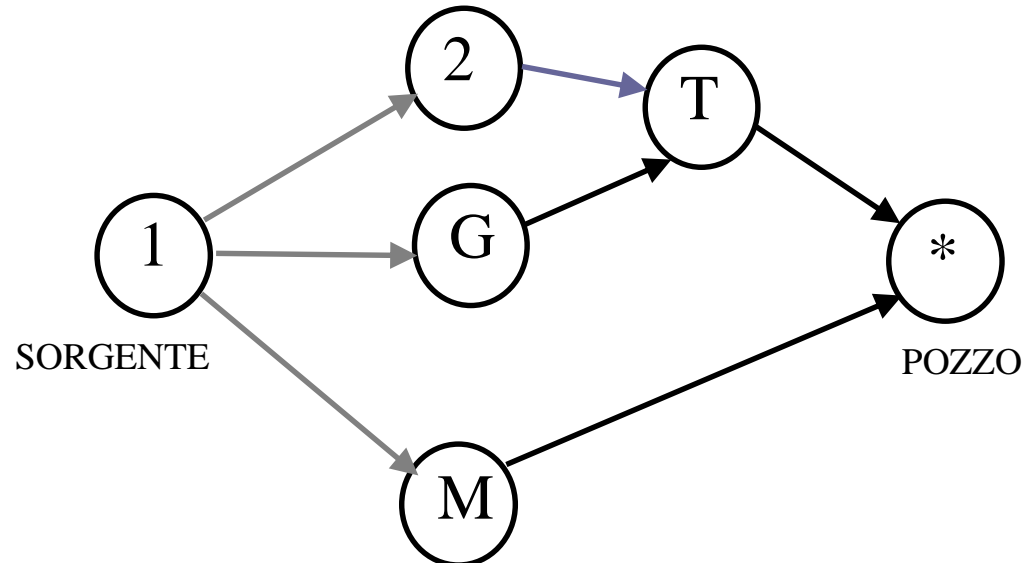


# Il problema del cammino critico (6)

Calcoliamo un ordinamento topologico per l'esempio precedente:

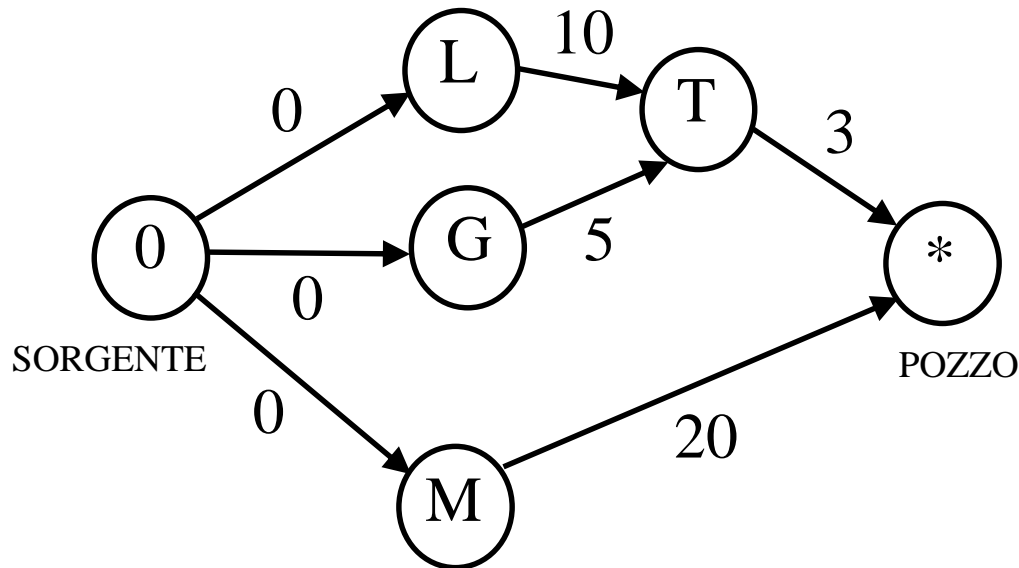


- Scelgo il nodo L=2 perché è non ha archi entranti (chiaramente, avrei potuto scegliere G o M al posto di L)
- Elimino l'arco uscente da L

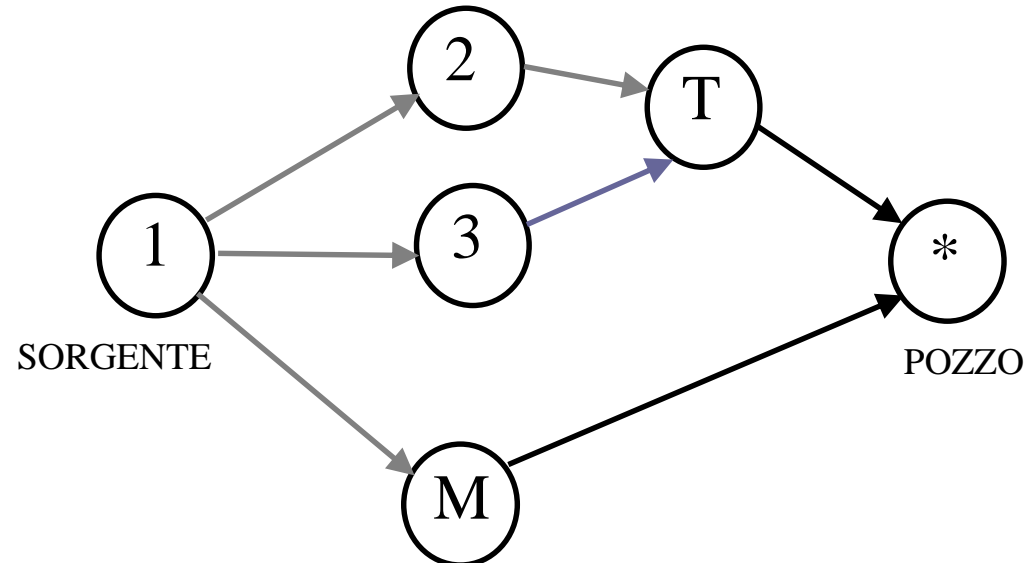


# Il problema del cammino critico (7)

Calcoliamo un ordinamento topologico per l'esempio precedente:

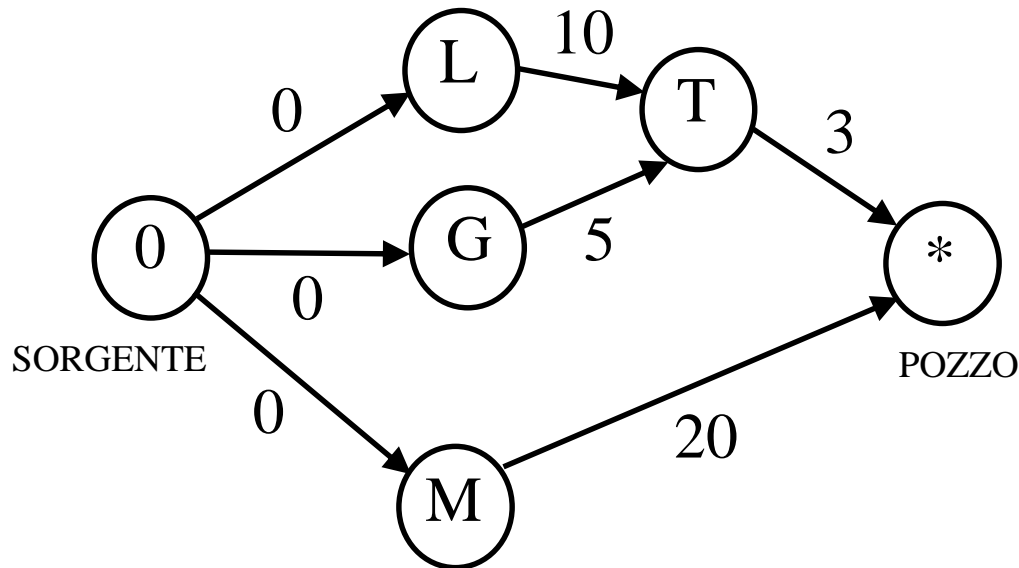


- Scelgo il nodo G=3 perché è non ha archi entranti (chiaramente, avrei potuto scegliere M al posto di G)
- Elimino l'arco uscente da G

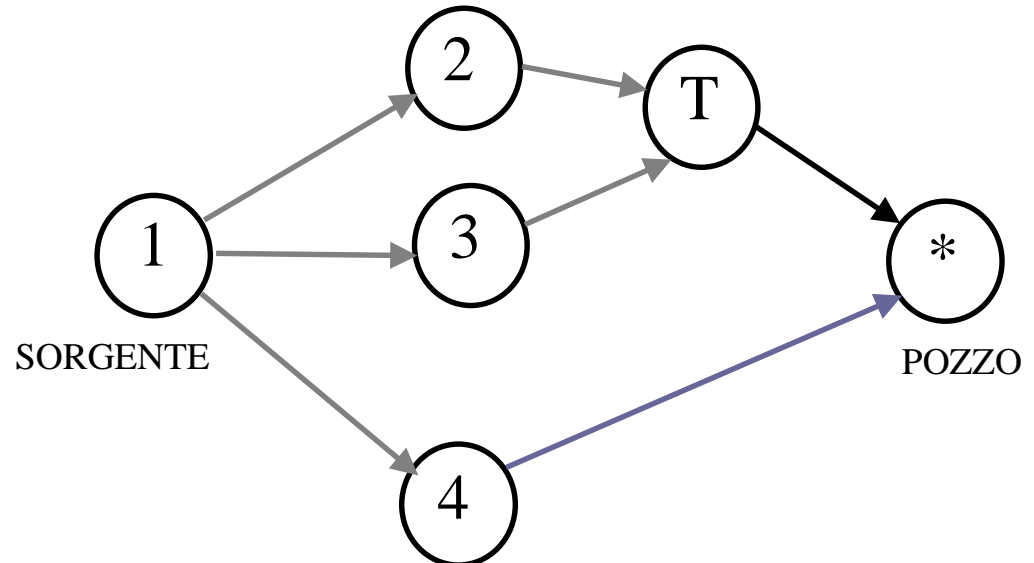


# Il problema del cammino critico (8)

Calcoliamo un ordinamento topologico per l'esempio precedente:

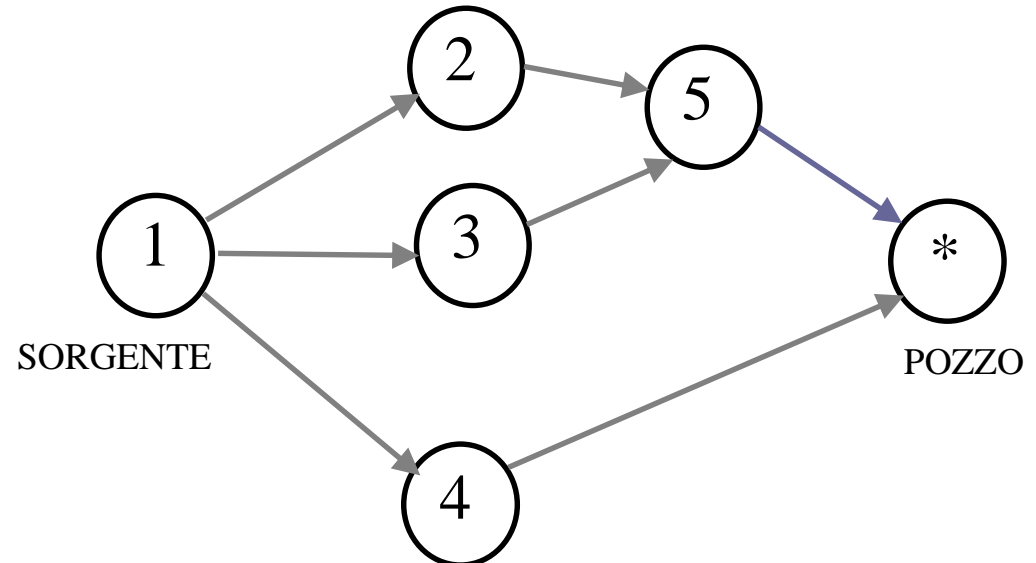
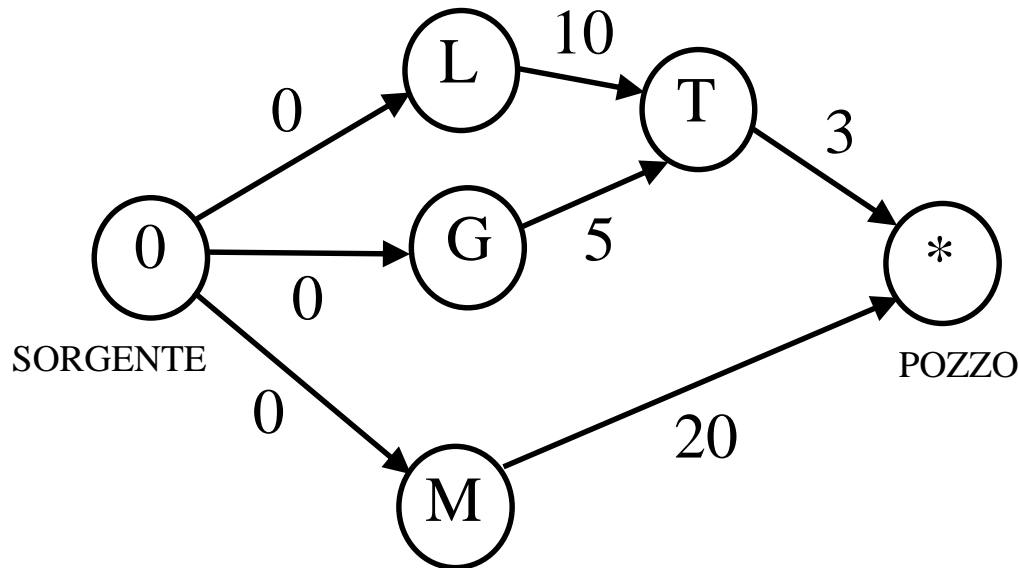


- Scelgo il nodo M=4 perché è non ha archi entranti (chiaramente, avrei potuto scegliere T al posto di M)
- Elimino l'arco uscenti da M



# Il problema del cammino critico (9)

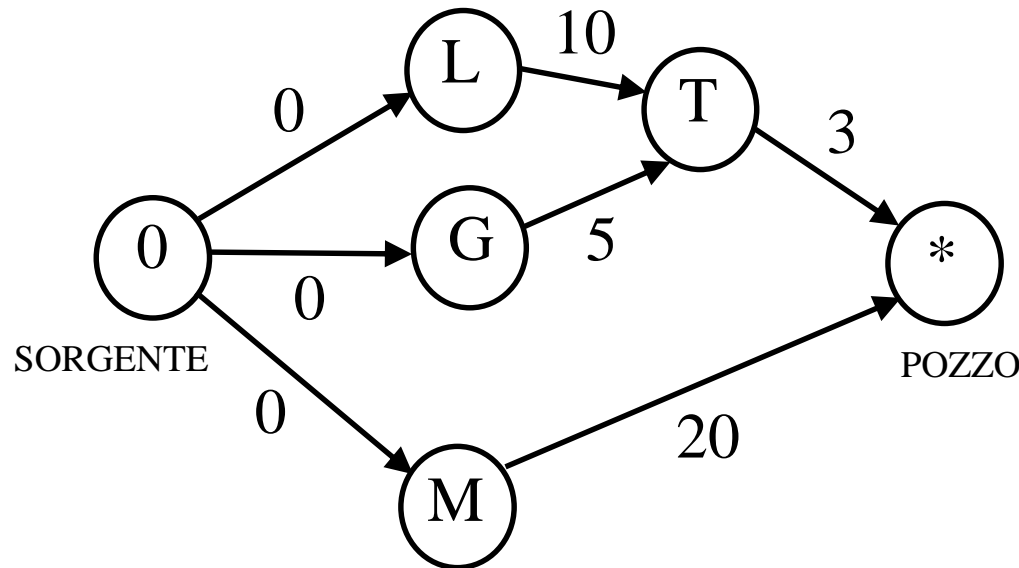
Calcoliamo un ordinamento topologico per l'esempio precedente:



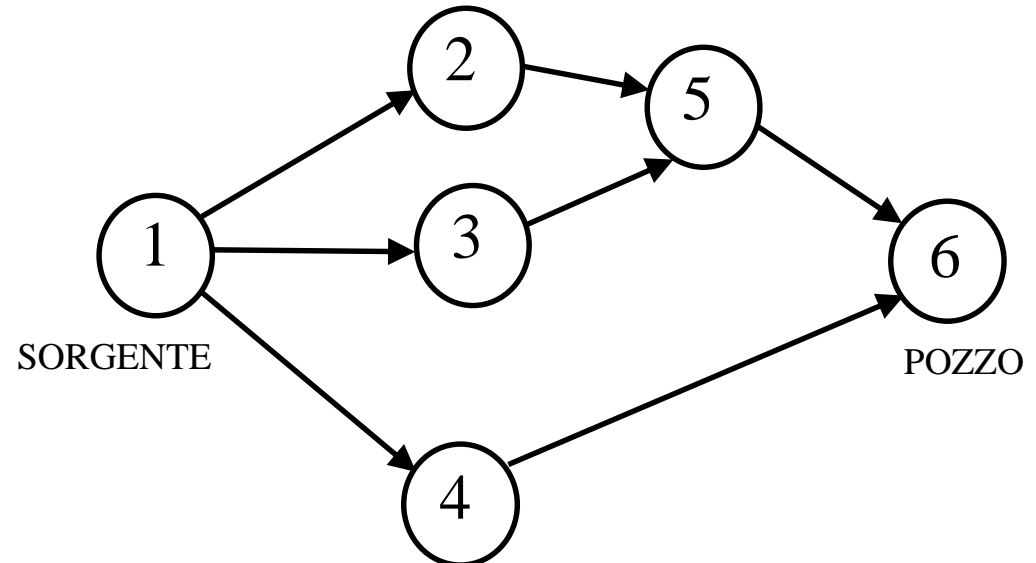
- Scelgo il nodo T=5 perché è non ha archi entranti
- Elimino l'arco uscente da T

# Il problema del cammino critico (10)

Calcoliamo un ordinamento topologico per l'esempio precedente:

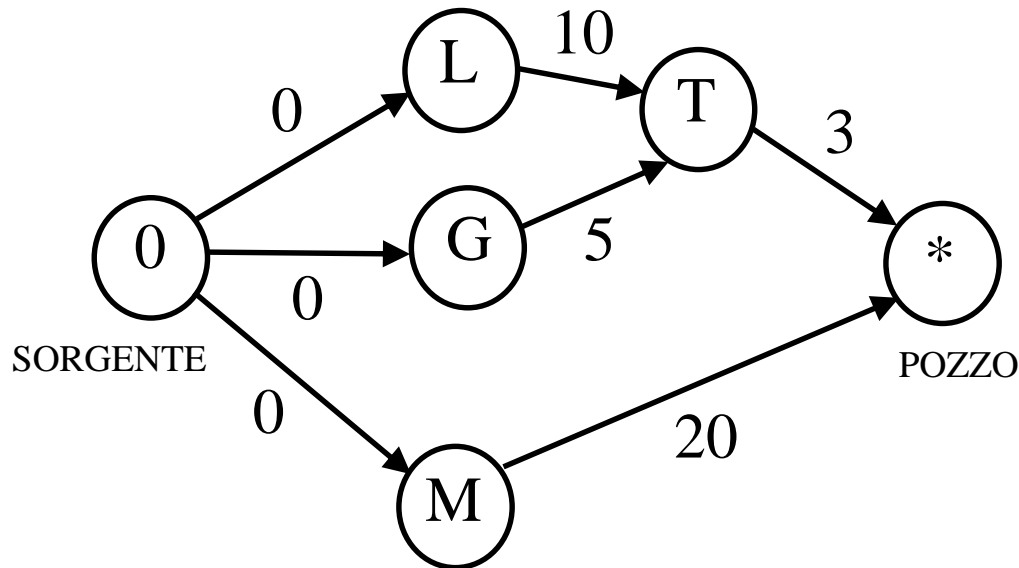


- Ho trovato un ordinamento topologico dei nodi

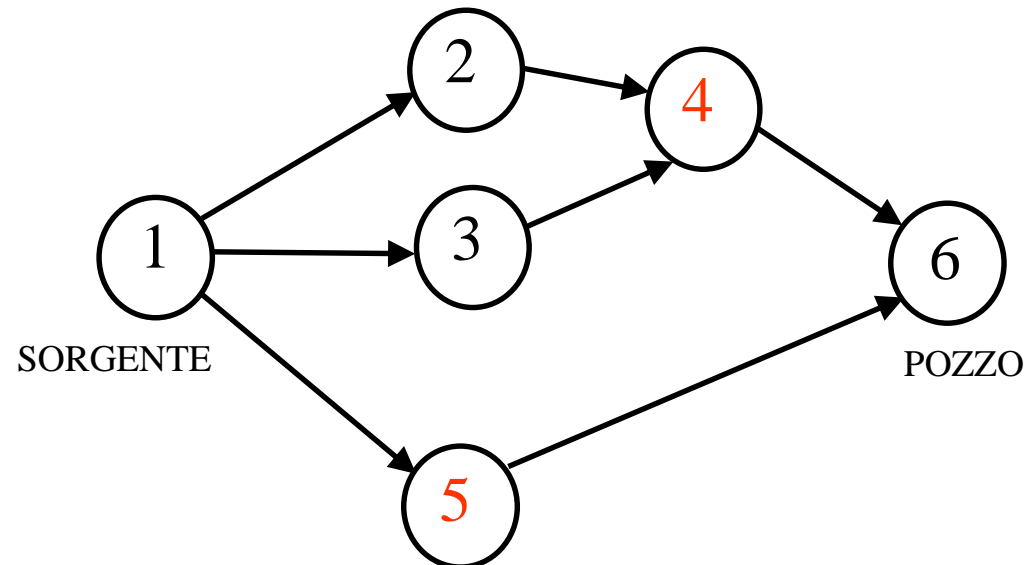


# Il problema del cammino critico (11)

Calcoliamo un ordinamento topologico per l'esempio precedente:

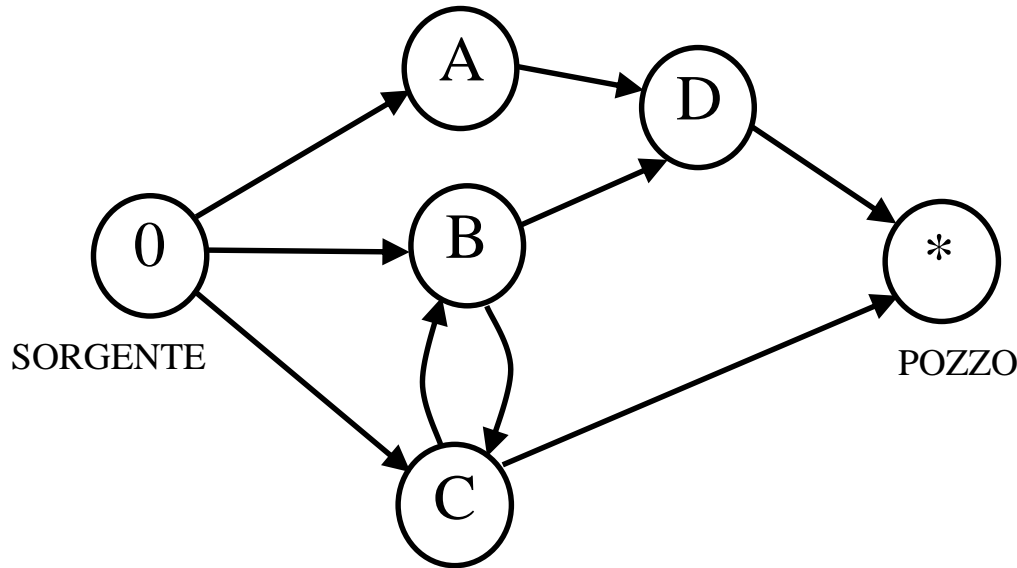


- In questo caso esiste più di un ordinamento topologico dei nodi, ad esempio:

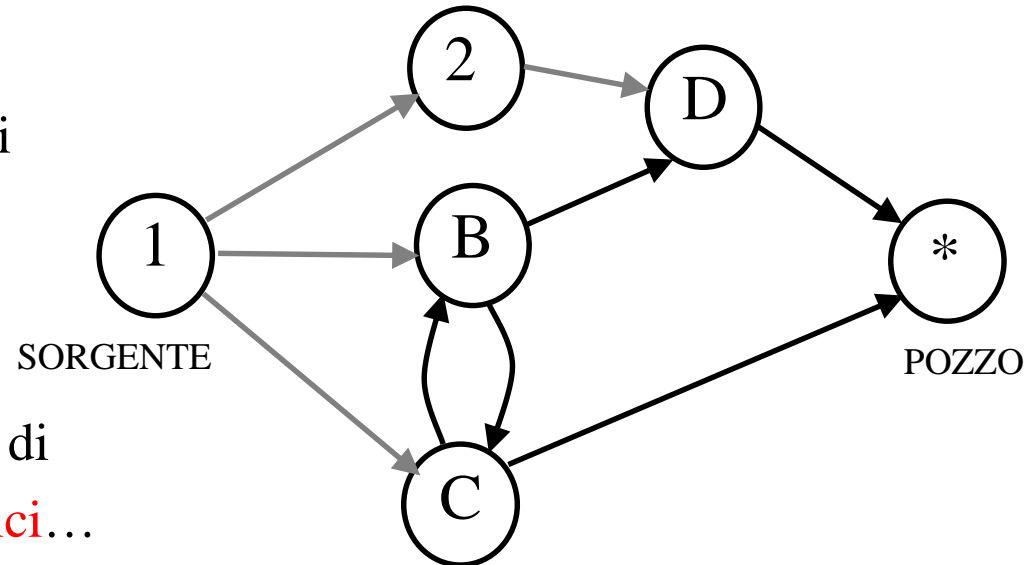


# Il problema del cammino critico (12)

Non esiste un ordinamento topologico se ci sono cicli nel digrafo:



- In questo **nuovo esempio** tutti i nodi non ancora appartenenti all'ordinamento topologico hanno almeno un arco entrante
- Anche se nella pianificazione di progetti si lavora con **digrafi aciclici**...



# Il problema del cammino critico (13)

Algoritmo Critical Path Method (Ver.  $O(m)$ )

// Inizializzazione

Ordina topologicamente i vertici;

// Calcolo dei  $EST$  (minimo istante di inizio dell'attività  $\alpha_i$ )

$EST[0] = 0$ ;

for (  $i = 1; i \leq n; i++$  ) (stella archi  
entranti nel nodo  $i$ )  
{  
     $EST[i] = \max\{EST[h] + c_{hi} : (h, i) \in \delta^-(i)\}$ ;  
}

$LST[\star] = EST[\star]$ ; ( $EST[*]$  esprime la durata del progetto)

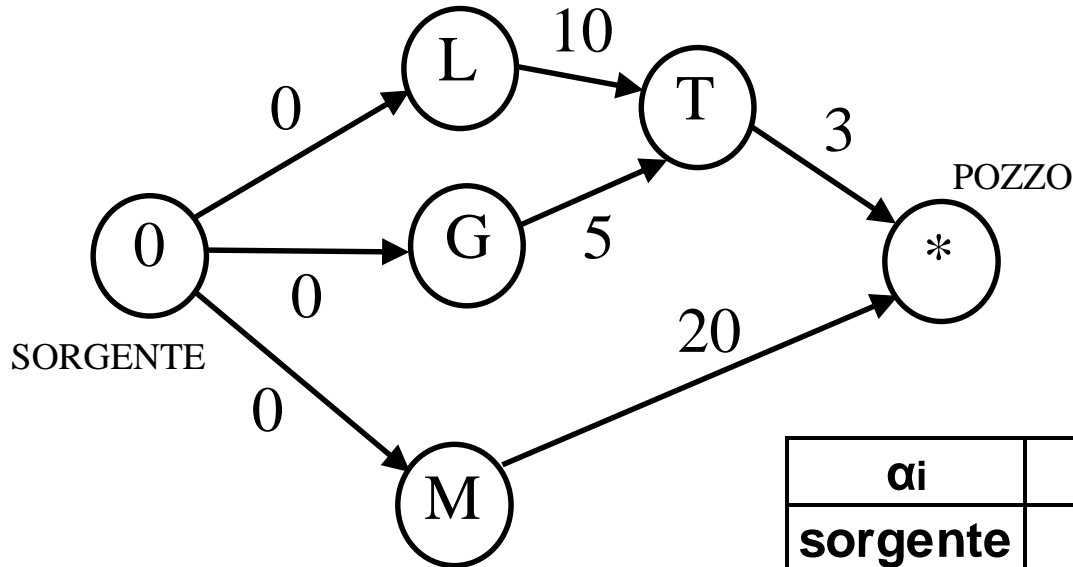
// Calcolo dei  $LST$  (massimo istante di inizio dell'attività  $\alpha_i$ )

for (  $i = n; i \geq 1; i--$  ) (stella archi  
uscenti dal nodo  $i$ )  
{  
     $LST[i] = \min\{LST[h] - c_{ih} : (i, h) \in \delta^+(i)\}$ ;  
}




# Il problema del cammino critico (14)

Calcoliamo EST e LST per ogni attività nel precedente esempio:



- Attività critiche ( $si = 0$ ):  
[sorgente, M, pozzo]
- Minimo tempo di completamento: 20

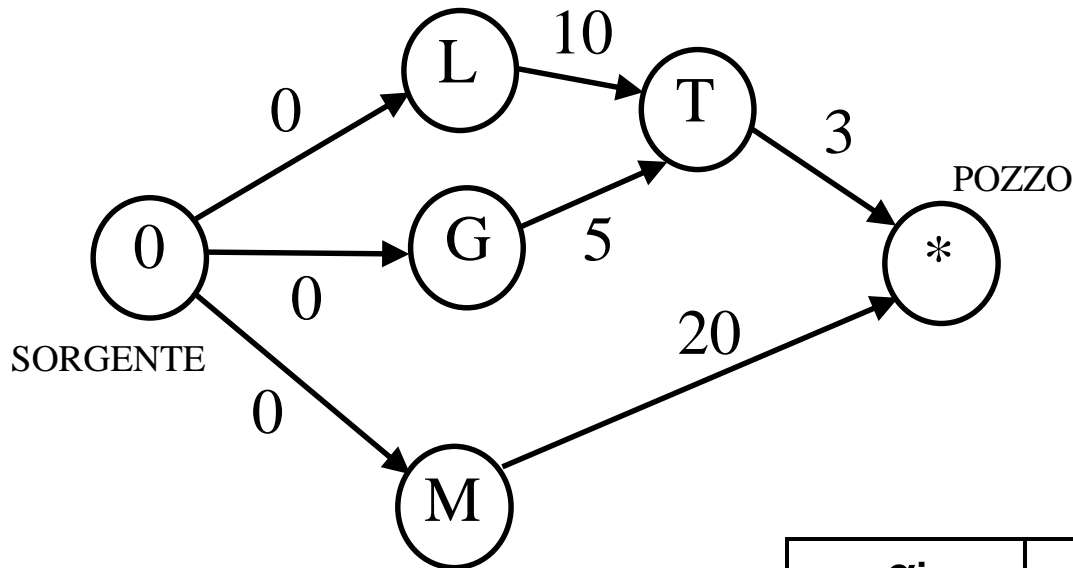


$\alpha_i$	OT <sub>i</sub>	EST <sub>i</sub>	LST <sub>i</sub>	Si
<b>sorgente</b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>L</b>	<b>2</b>	<b>0</b>	<b>7</b>	<b>7</b>
<b>G</b>	<b>3</b>	<b>0</b>	<b>12</b>	<b>12</b>
<b>M</b>	<b>4</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>T</b>	<b>5</b>	<b>10</b>	<b>17</b>	<b>7</b>
<b>pozzo</b>	<b>6</b>	<b>20</b>	<b>20</b>	<b>0</b>

$$EST[i] = \max\{EST[h] + c_{hi} : (h, i) \in \delta^-(i)\}$$

# Il problema del cammino critico (15)

Calcoliamo EST e LST per ogni attività nel precedente esempio:



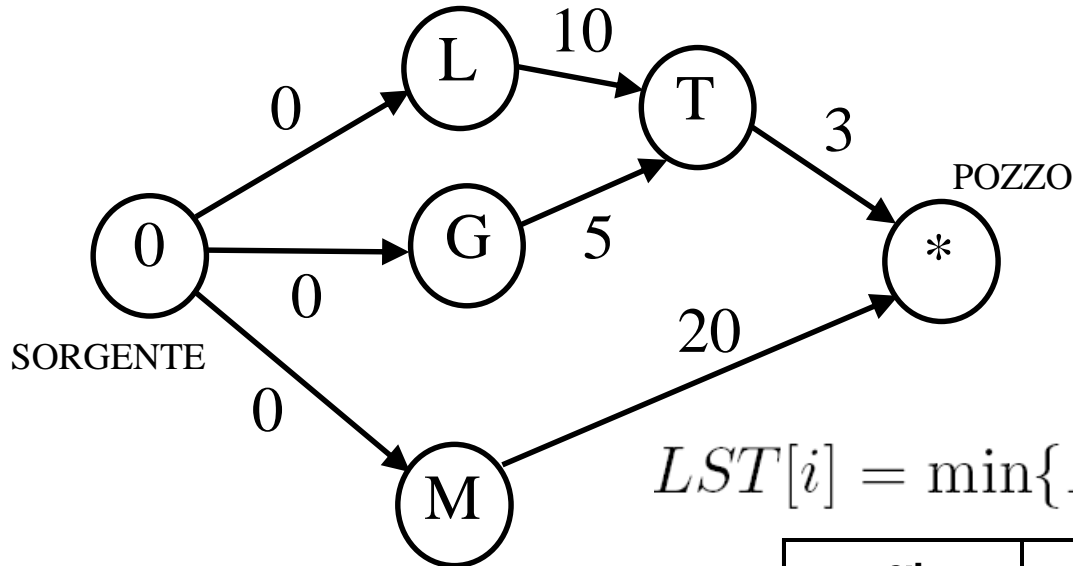
$$LST[\star] = EST[\star]$$

- Attività critiche ( $si = 0$ ):  
[sorgente, M, pozzo]
- Minimo tempo di completamento: 20

$\alpha_i$	$OT_i$	$EST_i$	$LST_i$	$si$
<b>sorgente</b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>L</b>	<b>2</b>	<b>0</b>	<b>7</b>	<b>7</b>
<b>G</b>	<b>3</b>	<b>0</b>	<b>12</b>	<b>12</b>
<b>M</b>	<b>4</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>T</b>	<b>5</b>	<b>10</b>	<b>17</b>	<b>7</b>
<b>pozzo</b>	<b>6</b>	<b>20</b>	<b>20</b>	<b>0</b>

# Il problema del cammino critico (16)

Calcoliamo EST e LST per ogni attività nel precedente esempio:



$$LST[i] = \min\{LST[h] - c_{ih} : (i, h) \in \delta^+(i)\}$$

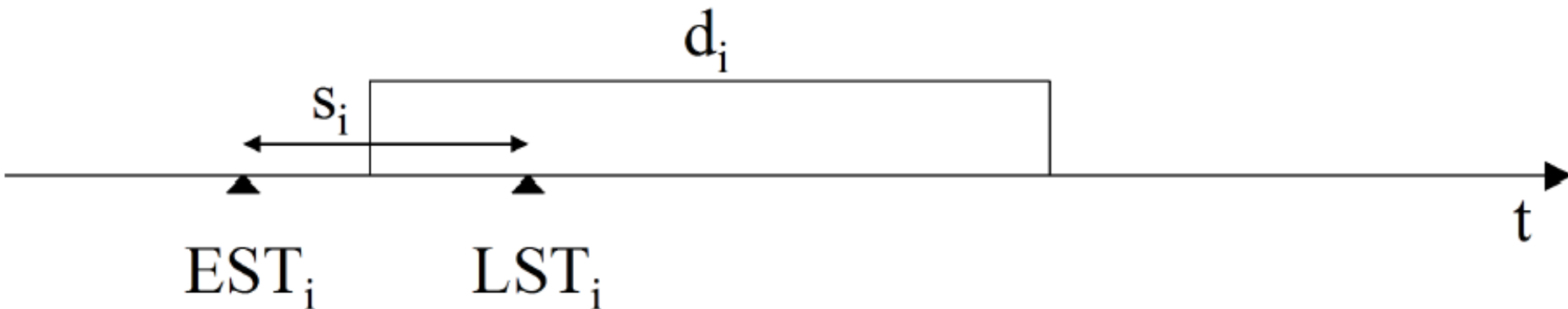
- Attività critiche ( $si = 0$ ):  
[sorgente, M, pozzo]
- Minimo tempo di completamento: 20

$\alpha_i$	OT <sub>i</sub>	EST <sub>i</sub>	LST <sub>i</sub>	si
<b>sorgente</b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>L</b>	<b>2</b>	<b>0</b>	<b>7</b>	<b>7</b>
<b>G</b>	<b>3</b>	<b>0</b>	<b>12</b>	<b>12</b>
<b>M</b>	<b>4</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>T</b>	<b>5</b>	<b>10</b>	<b>17</b>	<b>7</b>
<b>pozzo</b>	<b>6</b>	<b>20</b>	<b>20</b>	<b>0</b>



# Introduzione ai diagrammi di Gantt

- I diagrammi di Gantt sono un'utile strumento per la rappresentazione grafica delle durate di un progetto,
- Su di essi si rappresentano gli istanti di inizio e fine di ogni attività ( $\alpha_i$ ) evidenziando i cammini critici, il minimo ed il massimo istante d'inizio ( $EST_i$  e  $LST_i$ ) e lo slittamento ( $s_i$ ),
- Nei diagrammi di Gantt l'asse orizzontale rappresenta il tempo, ogni attività  $\alpha_i$  è rappresentata da una barra orizzontale la cui lunghezza esprime la sua durata  $d_i$ .



# Esempio di progetto (1)

Si supponga di voler tinteggiare una stanza.

Le attività da svolgere sono le seguenti:

1. Acquisto della vernice. Per l'acquisto della vernice servono 2 ore.
2. Acquisto dei pennelli. Questa attività richiede 1 ora.
3. Spostare i mobili della prima parete. Richiede 10 minuti.
4. Spostare i mobili della seconda parete. Richiede 30 minuti.
5. Spostare i mobili della terza parete. Richiede 20 minuti.
6. Spostare i mobili della quarta parete. Richiede 5 minuti.
7. Prima verniciatura della prima parete. Richiede 30 minuti.
8. Prima verniciatura della seconda parete. Richiede 40 minuti.
9. Prima verniciatura della terza parete. Richiede 30 minuti.
10. Prima verniciatura della quarta parete. Richiede 40 minuti.

# Esempio di progetto (2)

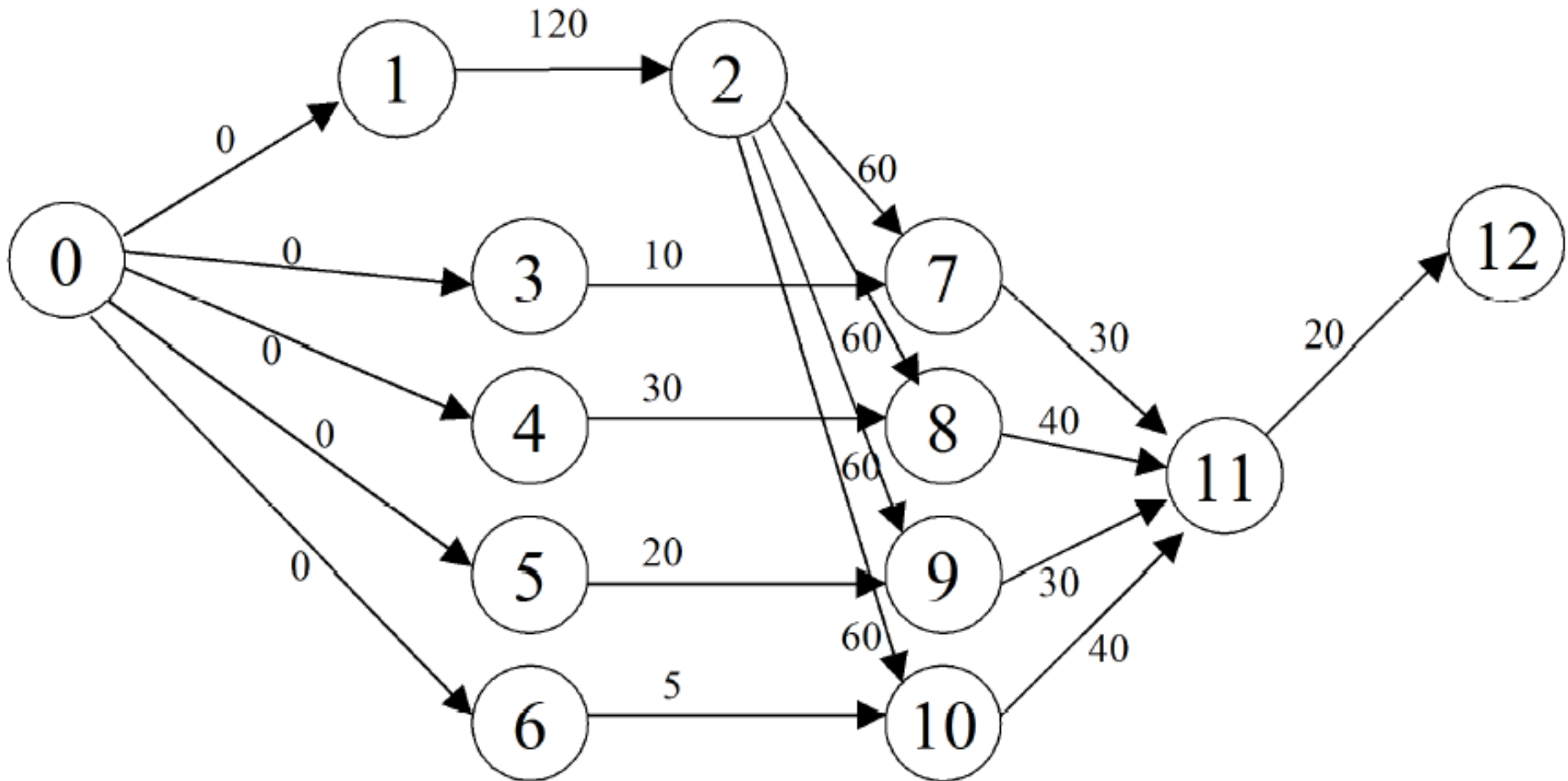
Inoltre valgono le seguenti precedenze:

- Non è possibile acquistare pennelli senza aver acquistato prima la vernice, per cui  $2 > 1$
- Non si potrà verniciare una parete se non si hanno i pennelli e la vernice, e se non si sono spostati i mobili dalla parete
- La fase dei ritocchi finali richiede che tutte le pareti siano state tinteggiate
- Si suppone che le quattro pareti siano indipendenti e che non ci siano vincoli di capacità, ovvero che si abbia a disposizione un numero sufficiente di operai

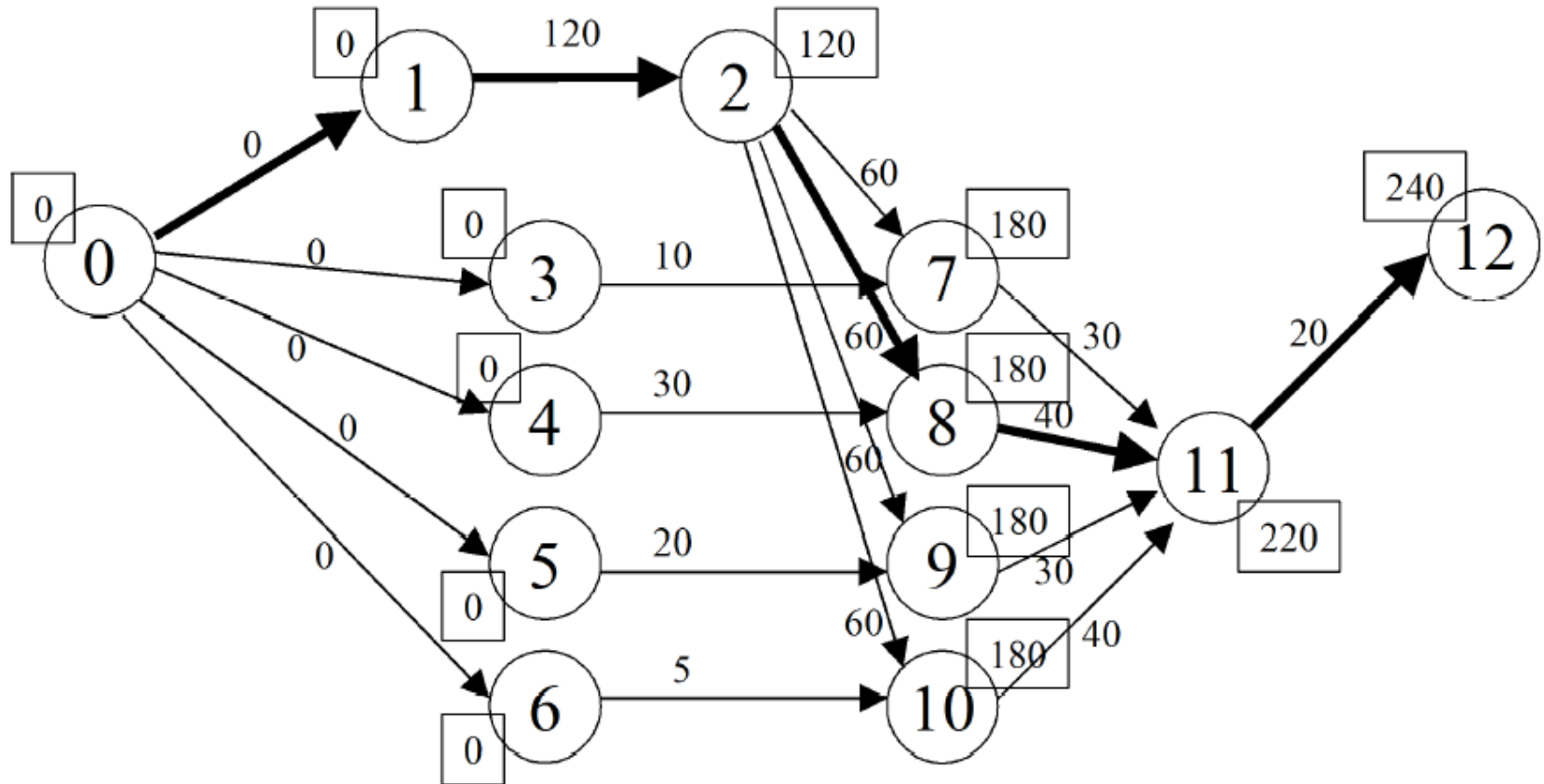
Rappresentiamo le precedenze relative alla prima parete che saranno  $7 > 2$ ,  $7 > 3$  e  $11 > 7$ , le precedenze relative alle altre pareti si ottengono similmente

# Esempio di progetto (3)

Il digrafo che rappresenta questo progetto è mostrato in figura:



# Esempio di progetto (4)

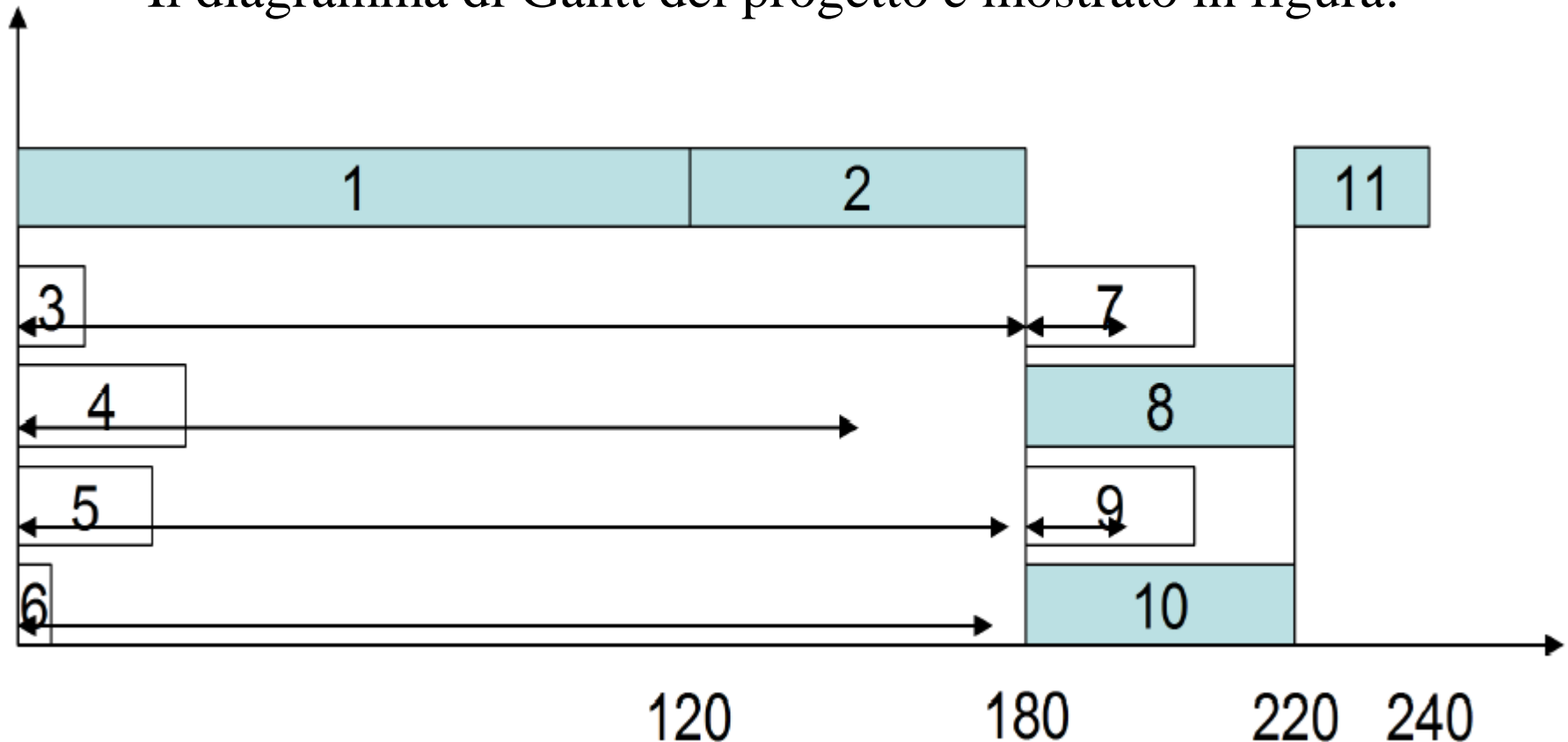


La durata del progetto sarà almeno di 240 minuti e il **cammino critico** attraversa le attività 1, 2, 8 e 11, per cui se non si vorrà introdurre un ritardo nel tempo di completamento del progetto, l'acquisto di vernice e pennelli, e la tinteggiatura della parete due e i ritocchi finali sono le attività critiche.



# Esempio di progetto (5)

Il diagramma di Gantt del progetto è mostrato in figura:



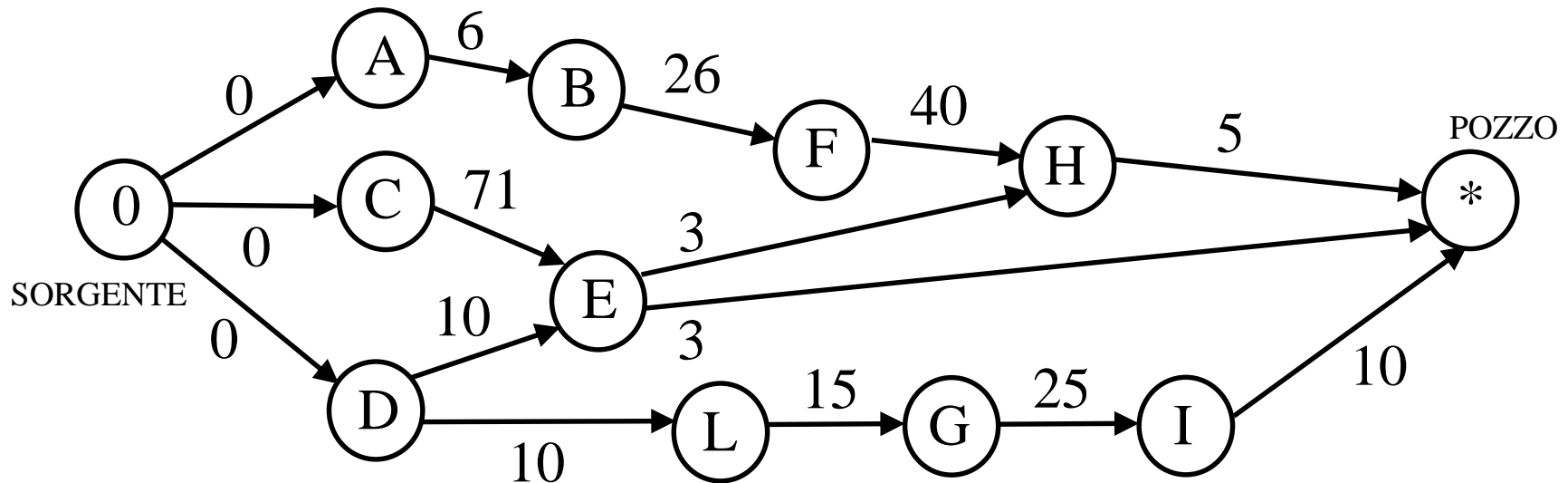
Nel diagramma sono state evidenziate le attività critiche e per le attività non critiche è stato evidenziato il valore dello slittamento, si noti in particolare come esistano due diversi cammini critici (notare le operazioni con slittamento nullo).

# Esercizi sulla pianificazione di progetti (1)

I Es. In tabella sono riportate le attività di un progetto.

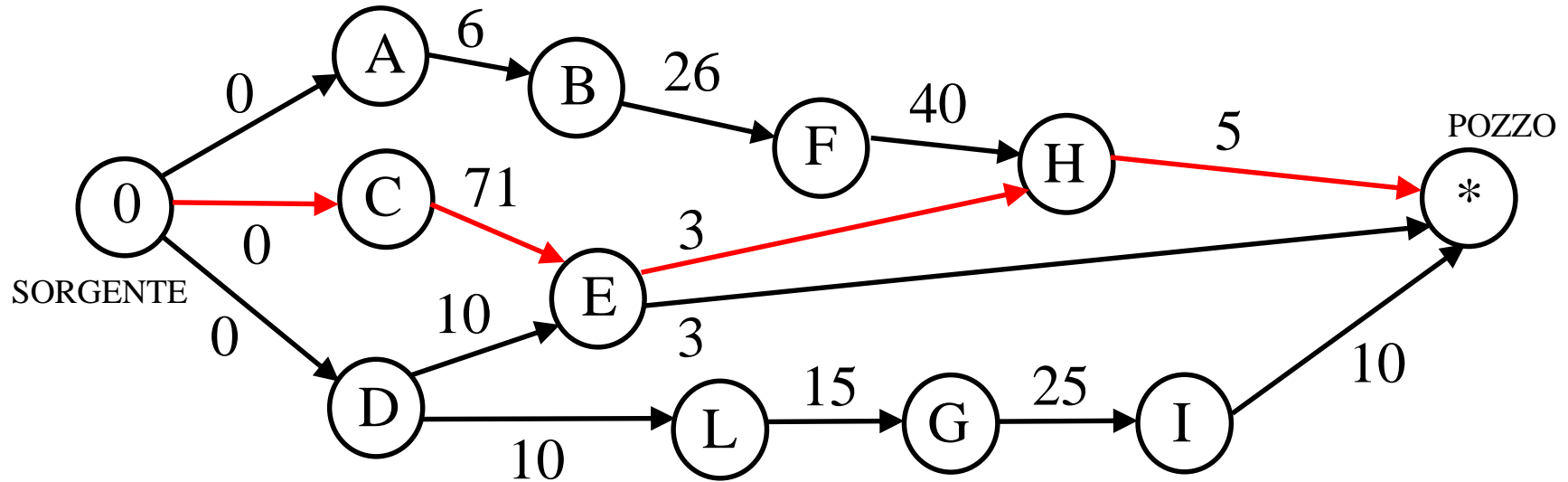
Domande: (1) Individuare la durata minima del progetto, (2) il cammino critico e (3) tutte le attività critiche.

Attività	A	B	C	D	E	F	G	H	I	L
Predecessori	-	A	-	-	C,D	B	L	F,E	G	D
Durata	6	26	71	10	3	40	25	5	10	15



# Esercizi sulla pianificazione di progetti (2)

I Es. Soluzione: Durata minima = 79; Attività critiche: 0, C, E, H, \*



$\alpha_i$	OT <sub>i</sub>	EST <sub>i</sub>	LST <sub>i</sub>	si
<b>sorgente</b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>A</b>	<b>2</b>	<b>0</b>	<b>2</b>	<b>2</b>
<b>B</b>	<b>3</b>	<b>6</b>	<b>8</b>	<b>2</b>
<b>F</b>	<b>4</b>	<b>32</b>	<b>34</b>	<b>2</b>
<b>C</b>	<b>5</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>D</b>	<b>6</b>	<b>0</b>	<b>20</b>	<b>20</b>

$\alpha_i$	OT <sub>i</sub>	EST <sub>i</sub>	LST <sub>i</sub>	si
<b>E</b>	<b>7</b>	<b>71</b>	<b>71</b>	<b>0</b>
<b>H</b>	<b>8</b>	<b>74</b>	<b>74</b>	<b>0</b>
<b>L</b>	<b>9</b>	<b>10</b>	<b>30</b>	<b>20</b>
<b>G</b>	<b>10</b>	<b>25</b>	<b>45</b>	<b>20</b>
<b>I</b>	<b>11</b>	<b>50</b>	<b>69</b>	<b>19</b>
<b>pozzo</b>	<b>12</b>	<b>79</b>	<b>79</b>	<b>0</b>

# Esercizi sulla pianificazione di progetti (3)

Il Es. In tabella sono riportate le 8 attività di un progetto, con le durate, i vincoli di precedenza tra le attività, minimo tempo d'inizio e lo slittamento di ciascuna attività  $i$  (indicata con  $A_i$  invece di  $\alpha_i$ ).

Attività	$A_1$	$A_2$	$A_3$	$A_4$	$A_5$	$A_6$	$A_7$	$A_8$
Durata	4	4	6	3	6	4	2	5
Predecessori	-	$A_1$	-	$A_1$ $A_3$	$A_3$	$A_2$ $A_4$	$A_5$ $A_6$	$A_4$ $A_7$
Min inizio	0	4	0	6	6	9	13	15
slittamento	1	1	0	0	1	0	0	0

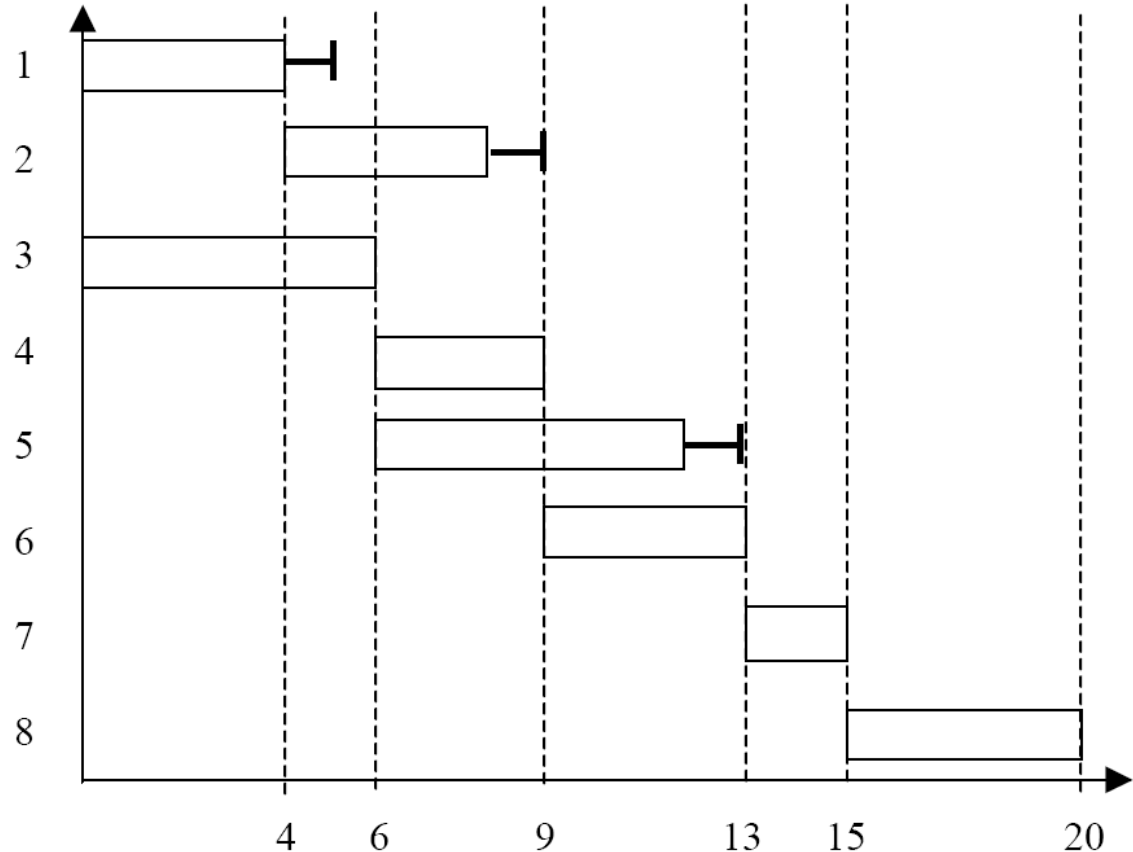
Rappresentare il diagramma di Gantt del progetto evidenziando le attività critiche e gli slittamenti delle attività non critiche.

# Esercizi sulla pianificazione di progetti (4)

Attività	A <sub>1</sub>	A <sub>2</sub>	A <sub>3</sub>	A <sub>4</sub>	A <sub>5</sub>	A <sub>6</sub>	A <sub>7</sub>	A <sub>8</sub>
Durata	4	4	6	3	6	4	2	5
Predecessori	-	A <sub>1</sub>	-	A <sub>1</sub> A <sub>3</sub>	A <sub>3</sub>	A <sub>2</sub> A <sub>4</sub>	A <sub>5</sub> A <sub>6</sub>	A <sub>4</sub> A <sub>7</sub>
Min inizio	0	4	0	6	6	9	13	15
slittamento	1	1	0	0	1	0	0	0

## Il Es. Soluzione:

- Minimo tempo di completamento: 20
- Le attività critiche sono: A<sub>3</sub>, A<sub>4</sub>, A<sub>6</sub>, A<sub>7</sub>, A<sub>8</sub>



# Esercizi sulla pianificazione di progetti (5)

Attività	A <sub>1</sub>	A <sub>2</sub>	A <sub>3</sub>	A <sub>4</sub>	A <sub>5</sub>	A <sub>6</sub>	A <sub>7</sub>	A <sub>8</sub>	A <sub>9</sub>	A <sub>10</sub>
Durata	4	13	6	17	8	3	6	4	14	19
Predecessori	-	-	-	A <sub>1</sub>	A <sub>2</sub>	A <sub>3</sub>	A <sub>5</sub>	A <sub>6</sub>	A <sub>7</sub>	A <sub>8</sub>
EST	0	0	0	4	13	6	21	27	27	31
LST	29	0	7	33	13	24	21	27	36	31
Slittamento	29	0	7	29	0	18	0	0	9	0

## III Es. Considerazioni:

- Minimo tempo di completamento: 50 ( $EST_{10} + d_{10}$ )
- Le attività critiche sono A<sub>2</sub>, A<sub>5</sub>, A<sub>7</sub>, A<sub>8</sub>, A<sub>10</sub>
- Nel caso in cui l'attività A<sub>4</sub> dovesse aumentare di 29 giorni allora la durata del progetto non cambierebbe, poiché A<sub>4</sub> ha uno slittamento di 29, ma gli slittamenti di A<sub>4</sub> e A<sub>1</sub> diverrebbero entrambi nulli, ovvero diventerebbero attività critiche.
- In questo caso il cammino critico diverrebbe un cammino orientato multiplo, ovvero un altro cammino critico sarebbe A<sub>1</sub>, A<sub>4</sub>.

# Esercizi sulla pianificazione di progetti (6)

Attività	A <sub>1</sub>	A <sub>2</sub>	A <sub>3</sub>	A <sub>4</sub>	A <sub>5</sub>	A <sub>6</sub>	A <sub>7</sub>	A <sub>8</sub>
Durata	4	16	14	10	10	20	4	14
Predecessori	-	A <sub>1</sub>	A <sub>1</sub>	A <sub>1</sub>	A <sub>2</sub> A <sub>3</sub> A <sub>4</sub>	A <sub>4</sub>	A <sub>2</sub> A <sub>3</sub> A <sub>4</sub> A <sub>5</sub> A <sub>6</sub>	A <sub>2</sub> A <sub>3</sub> A <sub>4</sub> A <sub>6</sub>
Min inizio	0	4	4	4	20	14	34	34
slittamento	0	14	16	0	14	0	10	0

## IV Es. Considerazioni:

- Min tempo di completamento: 48
- Le attività critiche sono: A<sub>1</sub>, A<sub>4</sub>, A<sub>6</sub>, A<sub>8</sub>
- Poiché lo slittamento di A<sub>7</sub> è di soli 10 giorni, nel caso in cui l'attività A<sub>7</sub> dovesse richiedere 16 giorni invece di 4 allora la durata del progetto cambierebbe e con essa il cammino critico.
- Riapplicando l'algoritmo si ha che la nuova durata diventa 50 e il cammino critico è A<sub>1</sub>, A<sub>4</sub>, A<sub>6</sub>, A<sub>7</sub>.