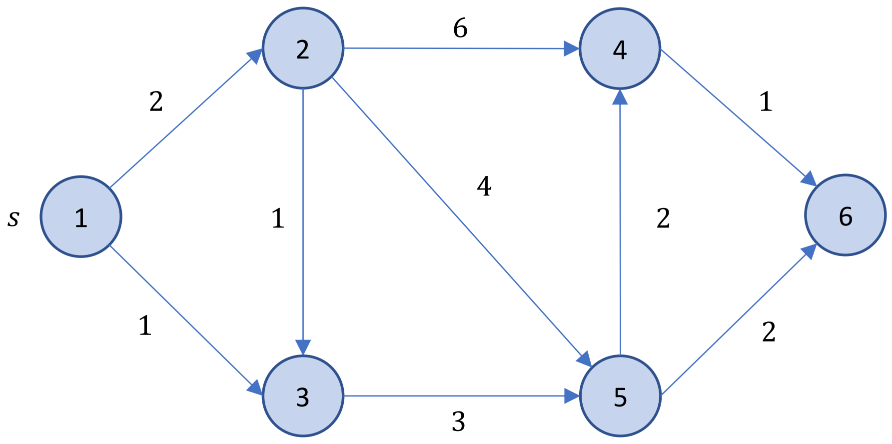
Teoria dei Grafi - Parte 1

Esercizi

1) Si consideri il seguente grafo orientato $G(V, A)$:



Su ogni arco (i, j) è riportato il costo c_{ij}. Rispondere ai seguenti quesiti:

a) Determinare i cammini minimi dal vertice $s = 1$ a tutti gli altri vertici.

Per determinare i cammini minimi da s a tutti gli altri vertici, siccome tutti i costi sono positivi, possiamo applicare l'algoritmo di Dijkstra:

```
Algoritmo Cammini di Costo Minimo: Dijkstra

Il vertice 1 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 2: label=2, pred=1
  Aggiornamento etichetta vertice 3: label=1, pred=1

Il vertice 3 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 5: label=4, pred=3

Il vertice 2 diventa permanente e viene espanso:
  Vertice 3 non viene aggiornato
  Aggiornamento etichetta vertice 4: label=8, pred=2
  Vertice 5 non viene aggiornato

Il vertice 5 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 4: label=6, pred=5
  Aggiornamento etichetta vertice 6: label=6, pred=5

Il vertice 4 diventa permanente e viene espanso:
  Vertice 6 non viene aggiornato

Il vertice 6 diventa permanente e viene espanso:
  Non ha successori

Ripepilogo Cammini Minimi:
  Cammino da 1 a 2 (costo=2):  0  1
  Cammino da 1 a 3 (costo=1):  0  2
  Cammino da 1 a 4 (costo=6):  0  2  4  3
  Cammino da 1 a 5 (costo=4):  0  2  4
  Cammino da 1 a 6 (costo=6):  0  2  4  5
```

b) Determinare se il grafo è fortemente connesso.

Per determinare se un grafo è fortemente connesso, in generale sarebbe necessario determinare se esiste un cammino per ogni coppia di vertici.

In questo caso è facile vedere che il vertice 1 ha solo archi uscenti, così come il vertice 6 ha solo archi entranti; quindi, il grafo non è sicuramente fortemente connesso. Infatti, non è possibile trovare un cammino che raggiunga il vertice 1 da tutti gli altri e non è possibile partire dal vertice 6 e raggiungere gli altri vertici.

c) Determinare se il grafo è aciclico.

Per determinare se il grafo è aciclico è sufficiente, per esempio, verificare se s riesce a dare un ordinamento "topologico" ai vertici applicando il seguente algoritmo:

Passo 1. Si inizializzano l'insieme V_0 con tutti i vertici del grafo (i.e., $V_0 = V$), l'insieme A_0 con tutti gli archi del grafo (i.e., $A_0 = A$) e $k = 0$.

Passo 2. Se $V_k = \emptyset$, allora il grafo è aciclico; STOP.

Passo 3. Si determinano il sottoinsieme di vertici $V'_k \subseteq V_k$ che non hanno archi entranti in A_k. Se $V'_k = \emptyset$ e $V_k \neq \emptyset$, allora il grafo non è aciclico; STOP.

Passo 4. Si aggiorna $A_{k+1} = A_k \backslash \{(i,j) \in A_k : i \in V'_k, j \in V_k \backslash V'_k\}$, poi $V_{k+1} = V_k \backslash V'_k$ e $k = k + 1$. Si torna al Passo 2.

Nel caso dell'esercizio abbiamo:

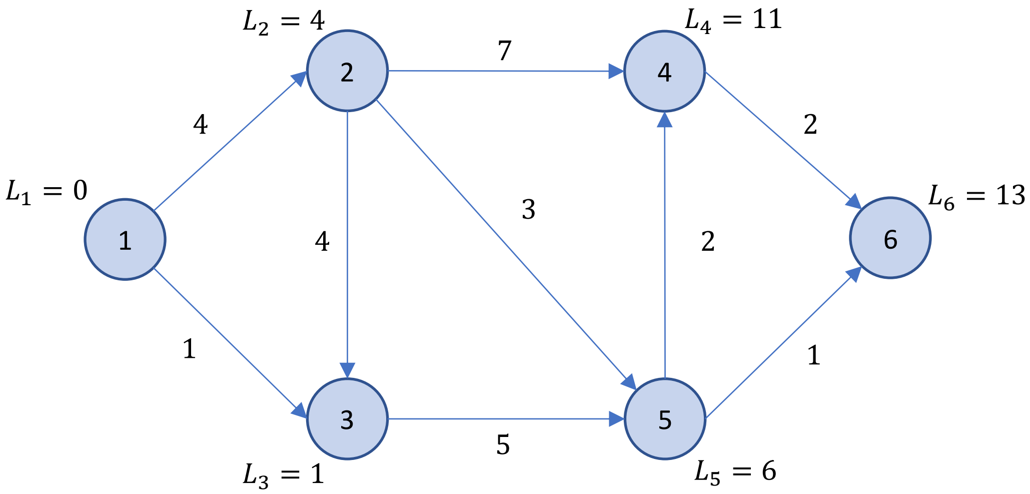
1. $V_0 = \{1,2,3,4,5,6\}$, $A_0 = \{\{1,2\},\{1,3\},\{2,3\},\{2,4\},\{2,5\},\{3,5\},\{4,6\},\{5,4\},\{5,6\}\}$ e $V'_0 = \{1\}$;
2. $V_1 = \{2,3,4,5,6\}$, $A_1 = \{\{2,3\},\{2,4\},\{2,5\},\{3,5\},\{4,6\},\{5,4\},\{5,6\}\}$ e $V'_1 = \{2\}$;
3. $V_2 = \{3,4,5,6\}$, $A_2 = \{\{3,5\},\{4,6\},\{5,4\},\{5,6\}\}$ e $V'_2 = \{3\}$;
4. $V_3 = \{4,5,6\}$, $A_3 = \{\{4,6\},\{5,4\},\{5,6\}\}$ e $V'_3 = \{5\}$;
5. $V_4 = \{4,6\}$, $A_4 = \{\{4,6\}\}$ e $V'_4 = \{4\}$;
6. $V_5 = \{6\}$, $A_4 = \emptyset$ e $V'_4 = \{6\}$;
7. $V_6 = \emptyset$, per cui il grafo a aciclico.

d) Si consideri il grafo non orientato ottenuto sostituire ogni arco (i,j) con un lato non orientato $\{i,j\}$ dello stesso costo c_{ij} e determinare l'albero di copertura di costo minimo.

```
Alberi di Copertura di Costo Minimo: Algoritmo Kruskal

Selezione dei lati:
 1) Inserimento del lato {1,3}
 2) Inserimento del lato {2,3}
 3) Inserimento del lato {4,6}
    Il lato {1,2} genera un ciclo e viene scartato
 4) Inserimento del lato {5,4}
    Il lato {5,6} genera un ciclo e viene scartato
 5) Inserimento del lato {3,5}
 Costo = 8
```

```
Alberi di Copertura di Costo Minimo: Algoritmo Prim

 1) Inserimento del lato {1, 3}
 2) Inserimento del lato {3, 2}
 3) Inserimento del lato {3, 5}
 4) Inserimento del lato {5, 4}
 5) Inserimento del lato {4, 6}
 Costo = 8
```

2) Si consideri il seguente grafo orientato $G(V, A)$:



Su ogni arco (i, j) è riportato il costo c_{ij} e su ogni vertice i la "distance label" L_i. Rispondere ai seguenti quesiti:

a) Le distance label corrispondono a cammini di costo minimo dal vertice 1 a tutti gli altri vertici? Giustificare la risposta.

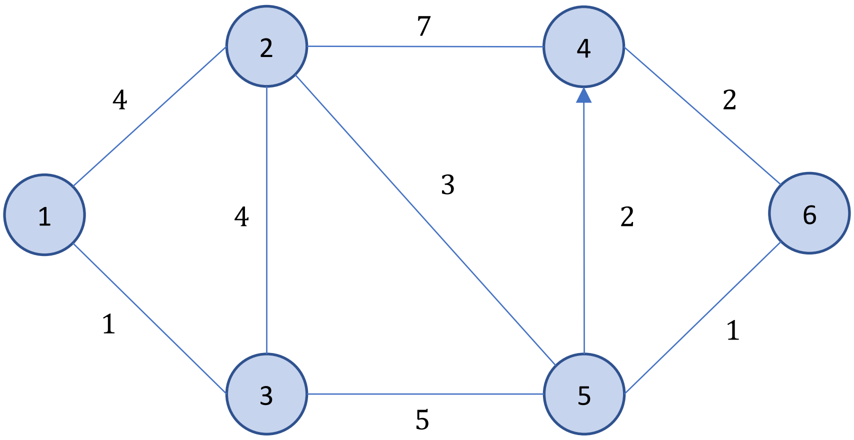
Per rispondere a questa domanda si può risolvere il problema di cammino minimo dal vertice 1 a tutti gli altri vertici. In alternativa, si poteva verificare che le label corrispondono al costo di un cammino e che le condizioni di ottimalità siano soddisfatte, come qui di seguito:

- $L_1 = 0$, per cui corrisponde al costo minimo per un cammino che parte dal vertice 1;
- $L_2 \leq L_1 + c_{12} \Longrightarrow 4 \leq 0 + 4$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 1;
- $L_3 \leq L_1 + c_{13} \Longrightarrow 1 \leq 0 + 1$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 1;
- $L_3 \leq L_2 + c_{23} \Longrightarrow 1 \leq 4 + 4$;
- $L_4 \leq L_2 + c_{24} \Longrightarrow 11 \leq 4 + 7$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 2;
- $L_4 \leq L_5 + c_{54} \Longrightarrow 11 \leq 6 + 2$ che viola le condizioni di ottimalità, quindi sicuramente le label non corrispondono a cammini di costo minimo dal vertice 1 a tutti gli altri vertici.

b) Determinare i cammini di costo minimo dal vertice 1 a tutti gli altri vertici, indicando sia il costo che i cammini.

```
Algoritmo Cammini di Costo Minimo: Dijkstra

Il vertice 1 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 2: label=4, pred=1
  Aggiornamento etichetta vertice 3: label=1, pred=1

Il vertice 3 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 5: label=6, pred=3

Il vertice 2 diventa permanente e viene espanso:
  Vertice 3 non viene aggiornato
  Aggiornamento etichetta vertice 4: label=11, pred=2
  Vertice 5 non viene aggiornato

Il vertice 5 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 4: label=8, pred=5
  Aggiornamento etichetta vertice 6: label=7, pred=5

Il vertice 6 diventa permanente e viene espanso:
  Non ha successori

Il vertice 4 diventa permanente e viene espanso:
  Vertice 6 non viene aggiornato

Ripepilogo Cammini Minimi:
  Cammino da 1 a 2 (costo=4):  1  2
  Cammino da 1 a 3 (costo=1):  1  3
  Cammino da 1 a 4 (costo=8):  1  3  5  4
  Cammino da 1 a 5 (costo=6):  1  3  5
  Cammino da 1 a 6 (costo=7):  1  3  5  6
```

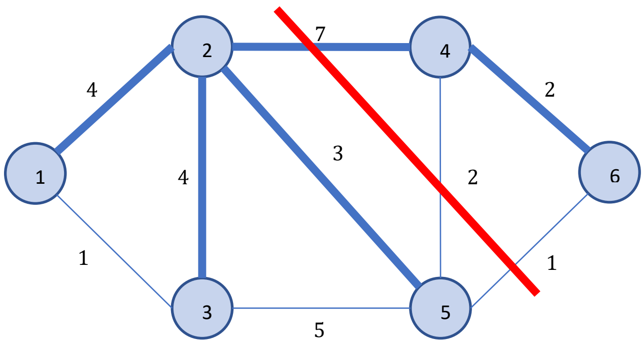
3) Si consideri il seguente grafo non orientato $G(V,E)$:



Su ogni lato $\{i,j\}$ è riportato il costo c_{ij}. Rispondere ai seguenti quesiti:

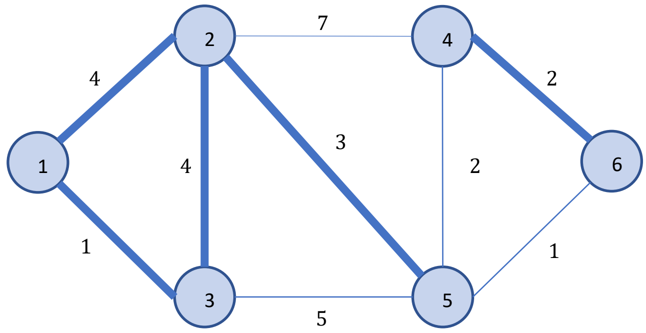
a) Si consideri l'insieme di lati $T = \{\{1,2\},\{2,3\},\{2,4\},\{2,5\},\{4,6\}\}$. Verificare se è un albero di copertura di costo minimo.

Evidenziamo i lati in T e possiamo notare che se, per esempio, eliminiamo il lato $\{2,4\}$ di costo 7 possiamo trovare nel taglio così generato il lato $\{4,5\}$ che ha costo 2. Quindi, possiamo trovare un nuovo albero di copertura di costo più basso, quindi T non è di costo minimo.



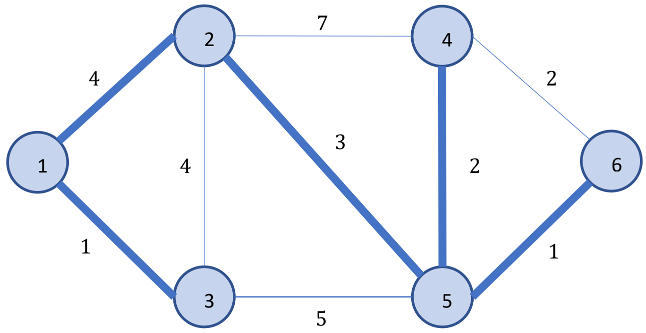
b) Si consideri l'insieme di lati $T = \{\{1,2\},\{1,3\},\{2,3\},\{2,5\},\{4,6\}\}$. Verificare se è un albero di copertura di costo minimo.

Evidenziamo i lati in T e possiamo notare che c'è un ciclo che coinvolge i vertici 1, 2 e 3, quindi il sottografo T non può essere un albero. Si noti, inoltre, che non è neppure connesso.



c) Si consideri l'insieme di lati $T = \{\{1,3\},\{5,6\},\{5,4\},\{5,2\},\{1,2\}\}$. Dimostrare che T è un albero di copertura di costo minimo. Se modifichiamo il costo del lato $\{1,3\}$, in modo che $c_{13} = 5$ lasciando invariati gli altri costi, come si può calcolare il nuovo albero di copertura di costo minimo?

Evidenziamo i lati in T e notiamo che se proviamo ad eliminare ciascun lato e consideriamo i lati del corrispondente taglio non troviamo mai un lato più conveniente. Quindi, T è un albero di copertura di costo minimo.



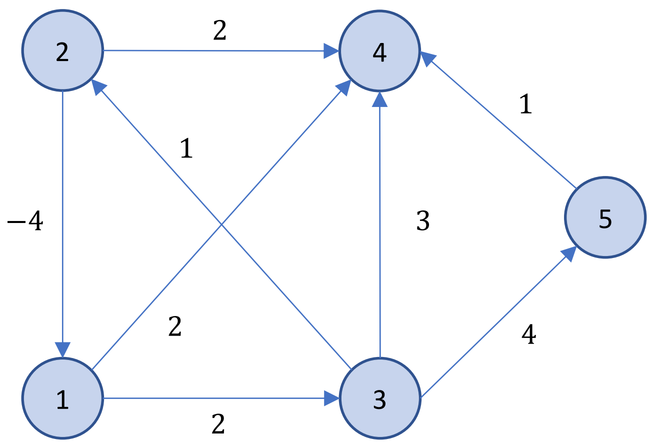
Se modifichiamo il costo del lato $\{1,3\}$ in modo che $c_{13} = 5$, lasciando invariati gli altri costi, per "riottimizzare" basta eliminare il lato $\{1,3\}$ e verificare se nel corrispondente taglio c'è un lato più conveniente. In pratica, si verifica se tra i lati $\{\{1,3\},\{2,3\},\{3,5\}\}$ c'è un'alternativa a $\{1,3\}$ più conveniente e la troviamo nel lato $\{2,3\}$ che costa 4.

d) Determinare i cammini di costo minimo dal vertice 1 a tutti gli altri vertici, indicando sia il costo che i cammini.

```
Algoritmo Cammini di Costo Minimo: Dijkstra

Il vertice 1 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 2: label=4, pred=1
  Aggiornamento etichetta vertice 3: label=1, pred=1

Il vertice 3 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 5: label=6, pred=3

Il vertice 2 diventa permanente e viene espanso:
  Vertice 3 non viene aggiornato
  Aggiornamento etichetta vertice 4: label=11, pred=2
  Vertice 5 non viene aggiornato

Il vertice 5 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 4: label=8, pred=5
  Aggiornamento etichetta vertice 6: label=7, pred=5
```

```
Il vertice 6 diventa permanente e viene espanso:
  Non ha successori

Il vertice 4 diventa permanente e viene espanso:
  Vertice 6 non viene aggiornato

Ripepilogo Cammini Minimi:
  Cammino da 1 a 2 (costo=4):  1  2
  Cammino da 1 a 3 (costo=1):  1  3
  Cammino da 1 a 4 (costo=8):  1  3  5  4
  Cammino da 1 a 5 (costo=6):  1  3  5
  Cammino da 1 a 6 (costo=7):  1  3  5  6
```

4) Si consideri il seguente grafo orientato $G(V,A)$:



Su ogni arco (i,j) è riportato il costo c_{ij}. Rispondere ai seguenti quesiti:

a) Determinare se il grafo è aciclico.

Si può applicare lo stesso metodo utilizzato nell'esercizio 1.c, oppure basta trovare un ciclo per escludere che possa essere aciclico. In questo caso è molto facile trovare un ciclo, ad esempio il ciclo generato dagli archi $(2,1)$, $(1,3)$ e $(3,2)$. Per cui, il grafo non può essere aciclico.

b) Determinare i cammini minimi dal vertice $s = 1$ a tutti gli altri vertici e determinare se eventualmente esiste un ciclo di costo negativo.

```
Algoritmo Cammini di Costo Minimo: Bellman-Ford

Iterazione 1:
  L'arco (1,3) viola la condizione: Label[3]=100000 > Label[1]=0 + Costo(1,3)=2
    ==> Label[3]=2 e pred[3]=0
  L'arco (1,4) viola la condizione: Label[4]=100000 > Label[1]=0 + Costo(1,4)=2
    ==> Label[4]=2 e pred[4]=0
  L'arco (3,5) viola la condizione: Label[5]=100000 > Label[3]=2 + Costo(3,5)=4
    ==> Label[5]=6 e pred[5]=2

Iterazione 2:
  L'arco (3,2) viola la condizione: Label[2]=100000 > Label[3]=2 + Costo(3,2)=1
    ==> Label[2]=3 e pred[2]=2

Iterazione 3:
  L'arco (2,1) viola la condizione: Label[1]=0 > Label[2]=3 + Costo(2,1)=-4
    ==> Label[1]=-1 e pred[1]=1
  L'arco (1,3) viola la condizione: Label[3]=2 > Label[1]=-1 + Costo(1,3)=2
    ==> Label[3]=1 e pred[3]=0
  L'arco (1,4) viola la condizione: Label[4]=2 > Label[1]=-1 + Costo(1,4)=2
    ==> Label[4]=1 e pred[4]=0
  L'arco (3,5) viola la condizione: Label[5]=6 > Label[3]=1 + Costo(3,5)=4
    ==> Label[5]=5 e pred[5]=2
```

```
Iterazione 4:
  L'arco (3,2) viola la condizione: Label[2]=3 > Label[3]=1 + Costo(3,2)=1
    ==> Label[2]=2 e pred[2]=2

Ciclo di costo negativo: l'arco (2,1) viola la condizione: Label[1]=-1 > Label[2]=2 + Costo(2,1)
```

c) Determinare se il grafo è fortemente connesso.

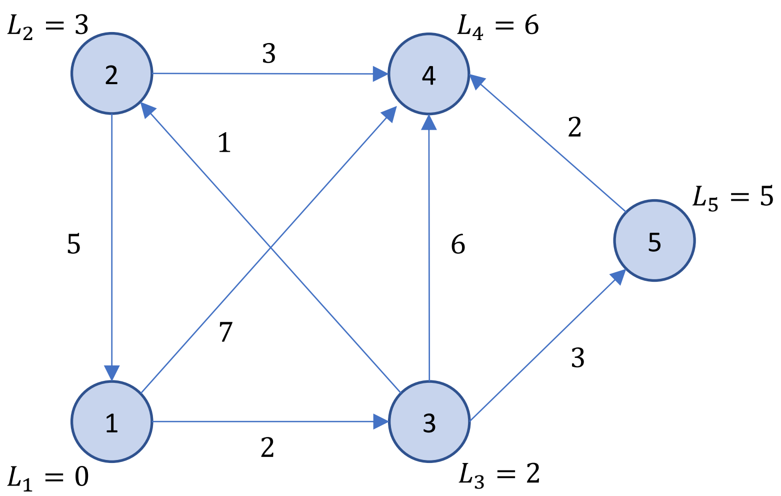
Per determinare se un grafo è fortemente connesso, in generale sarebbe necessario determinare se esiste un cammino per ogni coppia di vertici.

In questo caso è facile verificare che il vertice 4 ha solo archi entranti per cui nessun cammino che parte dal vertice 4 può raggiungere gli altri vertici; quindi, il grafo non è sicuramente fortemente connesso.

d) Si consideri il grafo non orientato ottenuto sostituire ogni arco (i, j) con un lato non orientato $\{i, j\}$ dello stesso costo c_{ij} e determinare l'albero di copertura di costo minimo.

```
Alberi di Copertura di Costo Minimo: Algoritmo Kruskal

Selezione dei lati:
 1) Inserimento del lato {2,1}
 2) Inserimento del lato {3,2}
 3) Inserimento del lato {5,4}
    Il lato {1,3} genera un ciclo e viene scartato
 4) Inserimento del lato {1,4}
 Costo = 0

Alberi di Copertura di Costo Minimo: Algoritmo Prim

 1) Inserimento del lato {1, 2}
 2) Inserimento del lato {2, 3}
 3) Inserimento del lato {1, 4}
 4) Inserimento del lato {4, 5}
 Costo = 0
```

5) Si consideri il seguente grafo orientato $G(V, A)$:



Su ogni arco (i, j) è riportato il costo c_{ij} e su ogni vertice i la "distance label" L_i. Rispondere ai seguenti quesiti:

a) Le distance label corrispondono a cammini di costo minimo dal vertice 1 a tutti gli altri vertici? Giustificare la risposta.

Per rispondere a questa domanda si può risolvere il problema di cammino minimo dal vertice 1 a tutti gli altri vertici. In alternativa, si poteva verificare che le label corrispondono al costo di un cammino e che le condizioni di ottimalità siano soddisfatte, come qui di seguito:

- $L_1 = 0$, per cui corrisponde al costo minimo per un cammino che parte dal vertice 1;
- $L_1 \leq L_2 + c_{21} \Rightarrow 0 \leq 3 + 5$;
- $L_2 \leq L_3 + c_{32} \Rightarrow 3 \leq 2 + 1$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 3;
- $L_3 \leq L_1 + c_{13} \Rightarrow 2 \leq 0 + 2$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 1;
- $L_4 \leq L_1 + c_{14} \Rightarrow 6 \leq 0 + 7$;
- $L_4 \leq L_2 + c_{24} \Rightarrow 6 \leq 3 + 3$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 2;
- $L_4 \leq L_3 + c_{34} \Rightarrow 6 \leq 2 + 6$;
- $L_4 \leq L_5 + c_{54} \Rightarrow 6 \leq 5 + 2$;
- $L_5 \leq L_3 + c_{35} \Rightarrow 5 \leq 2 + 3$ ed essendo saturata la disuguaglianza corrisponde al costo del cammino che arriva dal vertice 3.

Visto che tutte le label corrispondono a costi di cammini e le condizioni di ottimalità sono soddisfatte per tutti gli archi, allora le distance label corrispondono a cammini di costo minimo dal vertice 1 a tutti gli altri vertici.

b) Determinare i cammini di costo minimo dal vertice 1 a tutti gli altri vertici, indicando sia il costo che i cammini.

```
Algoritmo Cammini di Costo Minimo: Dijkstra

Il vertice 1 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 3: label=2, pred=1
  Aggiornamento etichetta vertice 4: label=7, pred=1

Il vertice 3 diventa permanente e viene espanso:
  Aggiornamento etichetta vertice 2: label=3, pred=3
  Aggiornamento etichetta vertice 5: label=5, pred=3

Il vertice 2 diventa permanente e viene espanso:
  Vertice 1 non viene aggiornato
  Aggiornamento etichetta vertice 4: label=6, pred=2

Il vertice 5 diventa permanente e viene espanso:
  Vertice 4 non viene aggiornato

Il vertice 4 diventa permanente e viene espanso:
  Non ha successori

Ripepilogo Cammini Minimi:
  Cammino da 1 a 2 (costo=3):  1  3  2
  Cammino da 1 a 3 (costo=2):  1  3
  Cammino da 1 a 4 (costo=6):  1  3  2  4
  Cammino da 1 a 5 (costo=5):  1  3  5
```

c) Determinare i cammini di costo minimo per ogni coppia di vertici, indicando il costo. Inoltre, indicare i cammini che partono dal vertice 3 a tutti gli altri vertici.

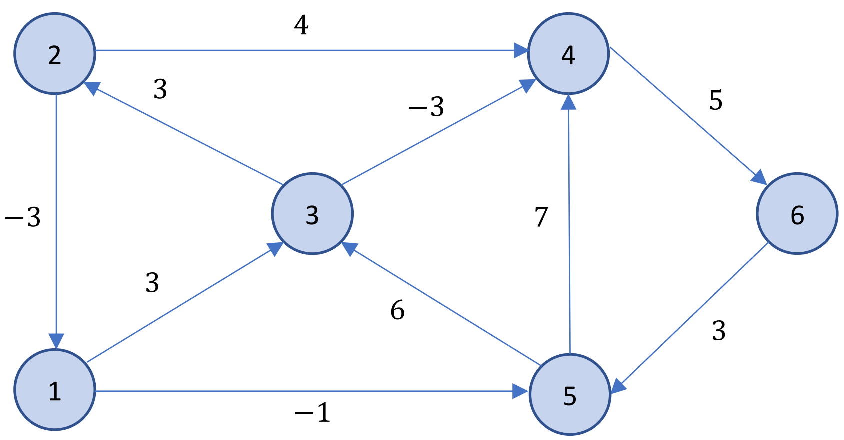
```
Algoritmo Cammini di Costo Minimo: Floyd-Warshal

Iterazione 0:

Matrice u_ij:
    0  1000     2     7  1000
    5     0     7     3  1000
 1000     1     0  1000     3
 1000  1000  1000     0  1000
 1000  1000  1000     2     0

Matrice Pred_ij:
    1     1     1     1     1
    2     2     1     2     2
    3     3     3     3     3
    4     4     4     4     4
    5     5     5     5     5

Iterazione 1:

Matrice u_ij:
    0  1000     2     7  1000
    5     0     7     3  1000
    6     1     0     4     3
 1000  1000  1000     0  1000
 1000  1000  1000     2     0

Matrice Pred_ij:
    1     1     1     1     1
    2     2     1     2     2
    2     3     3     2     3
    4     4     4     4     4
    5     5     5     5     5

Iterazione 2:

Matrice u_ij:
    0     3     2     6     5
    5     0     7     3    10
    6     1     0     4     3
 1000  1000  1000     0  1000
 1000  1000  1000     2     0

Matrice Pred_ij:
    1     3     1     2     3
    2     2     1     2     3
    2     3     3     2     3
    4     4     4     4     4
    5     5     5     5     5

Iterazione 3:

Matrice u_ij:
    0     3     2     6     5
    5     0     7     3    10
    6     1     0     4     3
 1000  1000  1000     0  1000
 1000  1000  1000     2     0
```

```
Matrice Pred_ij:
    1     3     1     2     3
    2     2     1     2     3
    2     3     3     2     3
    4     4     4     4     4
    5     5     5     5     5

Iterazione 4:

Matrice u_ij:
    0     3     2     6     5
    5     0     7     3    10
    6     1     0     4     3
 1000  1000  1000     0  1000
 1000  1000  1000     2     0

Matrice Pred_ij:
    1     3     1     2     3
    2     2     1     2     3
    2     3     3     2     3
    4     4     4     4     4
    5     5     5     5     5
```

I cammini che partono da 3 sono i seguenti:

```
Ripepilogo Cammini Minimi che partono da 3:
  Cammino da 3 a 1 (costo=6):  3  2  1
  Cammino da 3 a 2 (costo=1):  3  2
  Cammino da 3 a 4 (costo=4):  1  3  2  4
  Cammino da 3 a 5 (costo=3):  1  3  5
```

6) Si consideri il seguente grafo orientato $G(V, A)$:



Su ogni arco (i, j) è riportato il costo c_{ij}. Rispondere ai seguenti quesiti:

a) Determinare i cammini di costo minimo dal vertice 1 a tutti gli altri vertici, indicando sia il costo che i cammini e determinare se eventualmente esiste un ciclo di costo negativo.

```
Algoritmo Cammini di Costo Minimo: Bellman-Ford

Iterazione 1:
  L'arco (1,2) viola la condizione: Label[2]=100000 > Label[1]=0 + Costo(1,2)=100000
    ==> Label[2]=100000 e pred[2]=0
  L'arco (1,3) viola la condizione: Label[3]=100000 > Label[1]=0 + Costo(1,3)=3
    ==> Label[3]=3 e pred[3]=0
```

```
  L'arco (1,4) viola la condizione: Label[4]=100000 > Label[1]=0 + Costo(1,4)=100000
    ==> Label[4]=100000 e pred[4]=0
  L'arco (3,4) viola la condizione: Label[4]=100000 > Label[3]=3 + Costo(3,4)=-3
    ==> Label[4]=0 e pred[4]=2
  L'arco (1,5) viola la condizione: Label[5]=100000 > Label[1]=0 + Costo(1,5)=-1
    ==> Label[5]=-1 e pred[5]=0
  L'arco (1,6) viola la condizione: Label[6]=100000 > Label[1]=0 + Costo(1,6)=100000
    ==> Label[6]=100000 e pred[6]=0
  L'arco (4,6) viola la condizione: Label[6]=100000 > Label[4]=0 + Costo(4,6)=5
    ==> Label[6]=5 e pred[6]=3
Iterazione 2:
  L'arco (3,2) viola la condizione: Label[2]=100000 > Label[3]=3 + Costo(3,2)=3
    ==> Label[2]=6 e pred[2]=2
Iterazione 3:
Iterazione 4:
Iterazione 5:
Ripepilogo Cammini Minimi:
  Cammino da 1 a 2 (costo=6):  1  3  2
  Cammino da 1 a 3 (costo=3):  1  3
  Cammino da 1 a 4 (costo=0):  1  3  4
  Cammino da 1 a 5 (costo=-1):  1  5
  Cammino da 1 a 6 (costo=5):  1  3  4  6
```

Non ci sono cicli di costo negativo.

b) Si consideri il grafo non orientato ottenuto sostituire ogni arco (i, j) con un lato non orientato $\{i, j\}$ dello stesso costo c_{ij} e determinare l'albero di copertura di costo minimo.

```
Alberi di Copertura di Costo Minimo: Algoritmo Kruskal

Selezione dei lati:
 1) Inserimento del lato {2,1}
 2) Inserimento del lato {3,4}
 3) Inserimento del lato {1,5}
 4) Inserimento del lato {1,3}
    Il lato {3,2} genera un ciclo e viene scartato
 5) Inserimento del lato {6,5}
 Costo = -1


Alberi di Copertura di Costo Minimo: Algoritmo Prim

 1) Inserimento del lato {1, 2}
 2) Inserimento del lato {1, 5}
 3) Inserimento del lato {1, 3}
 4) Inserimento del lato {3, 4}
 5) Inserimento del lato {5, 6}
 Costo = -1
```

c) Determinare se il grafo è fortemente connesso.

Per determinare se un grafo è fortemente connesso, in generale sarebbe necessario determinare se esiste un cammino per ogni coppia di vertici. In questo caso per ogni vertice è sempre possibile trovare un cammino verso ogni altro vertice. Per determinarlo, per esempio, possiamo usare un algoritmo di etichettatura simile a quello di Ford-Fulkerson per il flusso massimo, visto che ci serve solo sapere se c'è un cammino e non quello di costo minimo.