

Tutoraggio Ricerca Operativa 2020/2021

7. Teoria dei Grafi:
Nozioni base, alberi ricoprenti
e cammini minimi

Aurora Rossi, Alice Raffaele, Romeo Rizzi

Università degli Studi di Verona

8 aprile 2021

- **Grafo:** configurazione formata da un insieme di punti (*vertici* o *nodi*) e un insieme di linee (*lati* o *archi*) che uniscono coppie di nodi;
 - **Indiretto:** quando i *lati* non sono orientati; si indica con $G = (V, E)$;
 - **Diretto:** quando gli *archi* sono orientati; si indica con $G = (V, A)$.
- **Grafo pesato:** a ogni lato/arco è associato un valore, detto *peso*; esiste quindi una funzione $f : E \rightarrow R$ che associa a ogni arco e un valore p ;
- **Grafo connesso:** quando tutti i vertici sono collegati tra loro (i.e., esiste un percorso che collega tutte le coppie di nodi);
- **Grafo completo:** quando ogni vertice è direttamente collegato con un lato a tutti gli altri;
- **Vertici adiacenti:** se esiste un lato/arco che li collega direttamente;
- **Ciclo:** sottografo i cui vertici formano un ciclo (un percorso chiuso).
- **Taglio:** dato un insieme $S \subseteq V$, il taglio $\delta(S)$ è l'insieme dei lati/archi del grafo che hanno esattamente una estremità in S ;
- **Albero:** grafo connesso e aciclico.

Il problema dell'albero ricoprente di costo minimo (Minimum Spanning Tree)

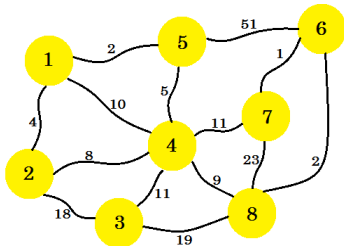
Consideriamo un grafo $G = (V, E, w)$ indiretto, connesso e pesato.

Un *albero ricoprente* di G è un sottografo $T = (V, E' \subseteq E)$ di G tale che:

- T è un albero;
- T contiene tutti i vertici di G .

Il minimo albero ricoprente (MST) è un albero ricoprente a costo minimo.

Esempio: consideriamo 8 case e i loro possibili collegamenti con i loro relativi costi; come possiamo collegarle in modo che il costo complessivo sia il minimo?



L'algoritmo di Kruskal

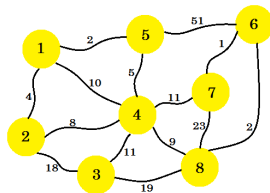
Algoritmo proposto dal matematico americano **Joseph Kruskal** nel 1956.

Procedimento:

- ① Ordinare i lati in base al loro peso, in modo non decrescente;
- ② Prendere i primi $n - 1$ lati tali da:
 - Ottenere un sottografo connesso;
 - Non avere cicli.

Strategia: mantiene un sottografo non per forza connesso di un MST (all'inizio tutti i vertici del grafo e nessun lato).

Risoluzione esercizio con Kruskal (I)



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

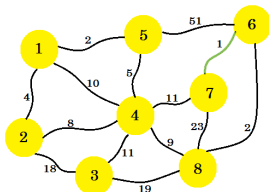
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (II)

Inserisco in soluzione il lato (6, 7) → **Costo parziale: 1**



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

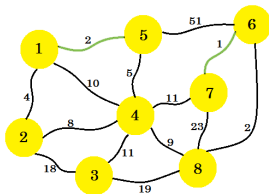
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (III)

Inserisco in soluzione il lato (1, 5) \rightarrow **Costo parziale:** 3



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

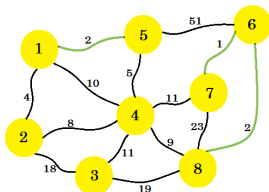
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (IV)

Inserisco in soluzione il lato (6, 8) → **Costo parziale: 5**



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

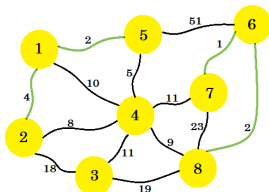
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (V)

Inserisco in soluzione il lato (1, 2) → **Costo parziale:** 9



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

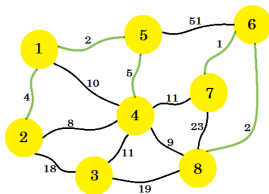
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (VI)

Inserisco in soluzione il lato (4, 5) → **Costo parziale:** 14



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

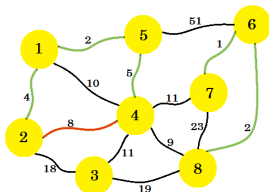
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (VII)

Inserisco in soluzione il lato (2, 4) → **NO!** Si formerebbe il ciclo 1-2-4-5.



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

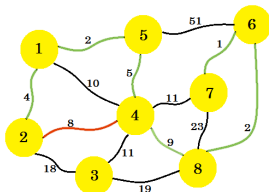
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (VIII)

Inserisco in soluzione il lato (4, 9) → **Costo parziale:** 23



❶ (6, 7) 1

❷ (1, 5) 2

❸ (6, 8) 2

❹ (1, 2) 4

❺ (4, 5) 5

❻ (2, 4) 8

❼ (4, 8) 9

❽ (1, 4) 10

❾ (3, 4) 11

❿ (4, 7) 11

⓫ (2, 3) 18

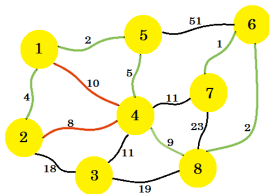
⓬ (3, 8) 19

⓭ (7, 8) 23

⓮ (5, 6) 51

Risoluzione esercizio con Kruskal (IX)

Inserisco in soluzione il lato (1, 4) → **NO!** Si formerebbe il ciclo 1-4-5.



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

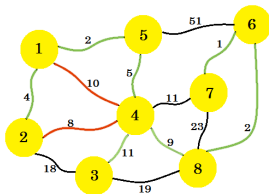
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal (X)

Inserisco in soluzione il lato (3, 4) → **Costo parziale:** 34



① (6, 7) 1

② (1, 5) 2

③ (6, 8) 2

④ (1, 2) 4

⑤ (4, 5) 5

⑥ (2, 4) 8

⑦ (4, 8) 9

⑧ (1, 4) 10

⑨ (3, 4) 11

⑩ (4, 7) 11

⑪ (2, 3) 18

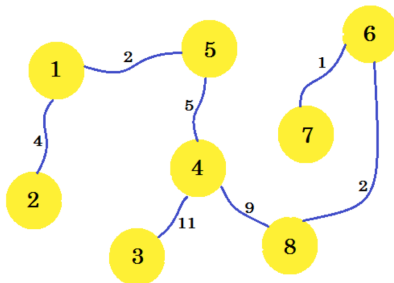
⑫ (3, 8) 19

⑬ (7, 8) 23

⑭ (5, 6) 51

Risoluzione esercizio con Kruskal - Soluzione

In soluzione ci sono $n - 1 = 7$ lati \rightarrow **Costo totale MST: 34**



Note:

- Se al posto del lato $(3,4)$ avessimo inserito il lato $(4,7)$, si sarebbe formato il ciclo $4-5-6-7$;
- L'algoritmo di Kruskal è *ottimo*: trova sempre la soluzione migliore;

Complessità: si può dimostrare che sia $O(|E| \log |E|)$ o $O(|E| \log |V|)$.

L'algoritmo di Prim

Fu originariamente sviluppato nel 1930 dal matematico ceco **Vojtěch Jarník** e indipendentemente dall'informatico **Robert C. Prim** nel 1957. Nel 1959 venne riscoperto da **Edsger Dijkstra**.

Procedimento:

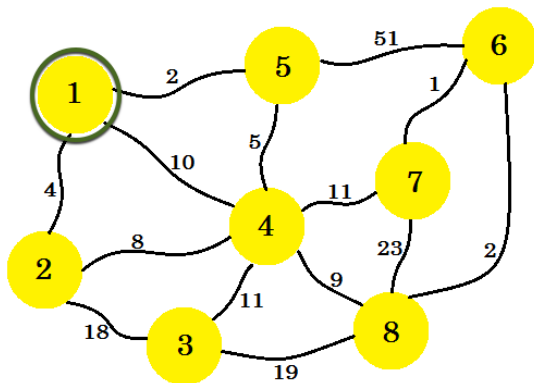
- 1 Selezionare un nodo di origine e inserirlo in un insieme S ;
- 2 Aggiungere il lato (u, v) di costo minimo adiacente a un nodo in S (i.e., $u \in S, v \in V \setminus S$);
- 3 Inserire v in S e ripetere dallo Step 2, finché la cardinalità di S non è pari a n .

Strategia: costruisce un albero connesso e aciclico (all'inizio fa parte della soluzione solo il nodo di origine).

Complessità: $O(|E| \log |V|)$.

Risoluzione esercizio con Prim (I)

Risolviamo lo stesso esercizio con l'algoritmo di Prim.



Selezioniamo come nodo origine il vertice 1 e inseriamolo in S .

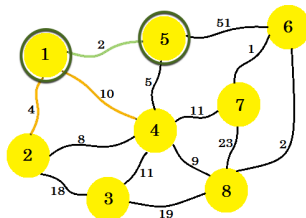
Risoluzione esercizio con Prim (II)

Vertici già visitati: $S = \{1\}$;

Lati selezionabili:

$\{(1, 2), (1, 4), (1, 5)\}$;

Costo parziale: 0.



Scegliendo il lato $(1, 5)$, inseriamo il vertice 5 in S e aggiungiamo i lati a esso adiacenti tra quelli selezionabili:

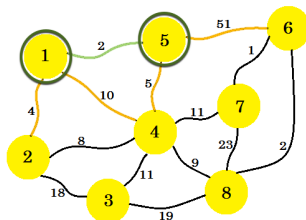
Vertici già visitati: $S = \{1, 5\}$;

Lati in soluzione: $\{(1, 5)\}$

Lati selezionabili:

$\{(1, 4), (1, 5), (4, 5), (5, 6)\}$;

Costo parziale: 2.



Risoluzione esercizio con Prim (III)

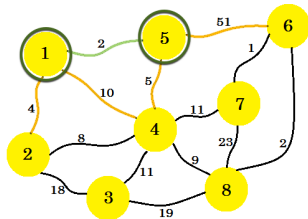
Vertici già visitati: $S = \{1, 5\}$;

Lati in soluzione: $\{(1, 5)\}$

Lati selezionabili:

$\{(1, 2), (1, 5), (4, 5), (5, 6)\}$;

Costo parziale: 2.



Selezioniamo il lato (1, 2) e aggiungiamo a S il vertice 2:

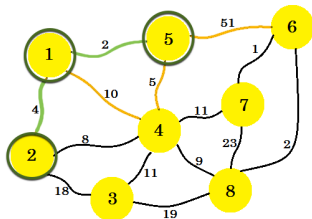
Vertici già visitati: $S = \{1, 5, 2\}$;

Lati in soluzione: $\{(1, 5), (1, 2)\}$

Lati selezionabili:

$\{(1, 4), (4, 5), (5, 6)\}$;

Costo parziale: 6.



Risoluzione esercizio con Prim (IV)

Aggiungiamo i lati adiacenti al vertice 2:

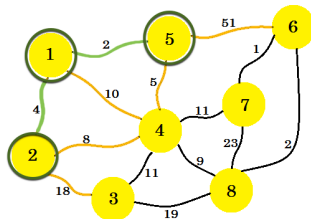
Vertici già visitati: $S = \{1, 5, 2\}$;

Lati in soluzione: $\{(1, 5), (1, 2)\}$

Lati selezionabili:

$\{(1, 5), (4, 5), (5, 6), (2, 3), (2, 4)\}$;

Costo parziale: 6.



Selezioniamo il lato (4, 5) e aggiungiamo a S il vertice 4:

Vertici già visitati: $S = \{1, 5, 2, 4\}$;

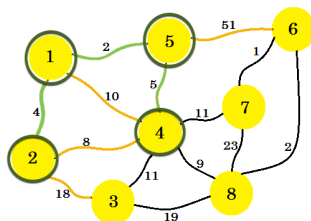
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5)\}$ **Lati**

selezionabili:

$\{(1, 5), (5, 6), (2, 3), (2, 4)\}$;

Costo parziale: 11.



Risoluzione esercizio con Prim (V)

Aggiungiamo i lati adiacenti al vertice 4:

Vertici già visitati: $S = \{1, 5, 2, 4\}$;

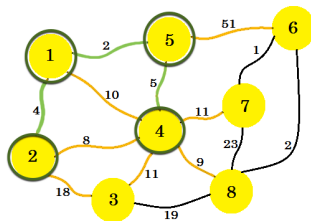
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5)\}$

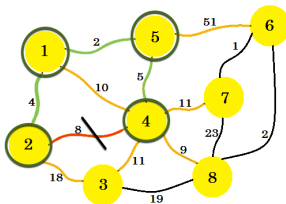
Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (2, 4), (3, 4), (4, 7),$

Costo parziale: 11.



Non possiamo selezionare il lato $(2, 4)$ perché si formerebbe un ciclo tra i nodi 1-2-4-5.



Risoluzione esercizio con Prim (VI)

Vertici già visitati: $S = \{1, 5, 2, 4\}$;

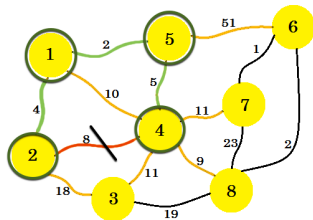
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7), (4, 8)\}$

Costo parziale: 11.



Selezioniamo il lato $(4, 8)$ e aggiungiamo a S il vertice 8:

Vertici già visitati:

$S = \{1, 5, 2, 4, 8\}$;

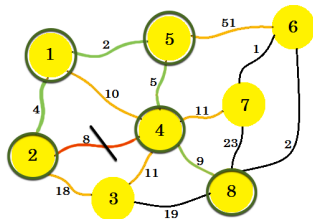
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7)\}$;

Costo parziale: 20.



Risoluzione esercizio con Prim (VII)

Aggiungiamo i lati adiacenti al vertice 8:

Vertici già visitati:

$S = \{1, 5, 2, 4, 8\};$

Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7), (6, 8),$

Costo parziale: 20.

Selezioniamo il lato $(6, 8)$ e aggiungiamo a S il vertice 6:

Vertici già visitati:

$S = \{1, 5, 2, 4, 8, 6\};$

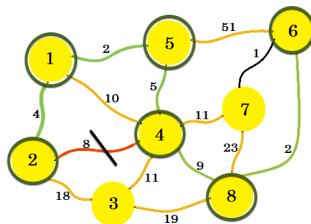
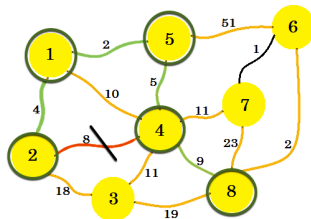
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8), (6, 8)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7), (7, 8)\}$

Costo parziale: 22.



Risoluzione esercizio con Prim (VIII)

Aggiungiamo i lati adiacenti al vertice 6:

Vertici già visitati: $S = \{1, 5, 2, 4, 8, 6\}$;

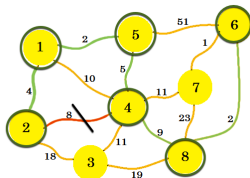
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8), (6, 8)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7), (7, 8), (6, 7)\}$;

Costo parziale: 22.



Selezioniamo il lato $(6, 7)$ e aggiungiamo a S il vertice 7:

Vertici già visitati:

$S = \{1, 5, 2, 4, 8, 6, 7\}$;

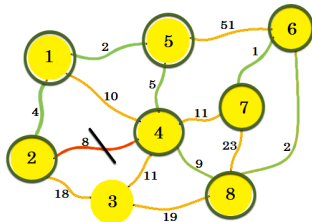
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8), (6, 8), (6, 7)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7), (7, 8)\}$;

Costo parziale: 23.



Risoluzione esercizio con Prim (IX)

Tutti i lati adiacenti al vertice 7 sono già tra i selezionabili:

Vertici già visitati:

$S = \{1, 5, 2, 4, 8, 6, 7\};$

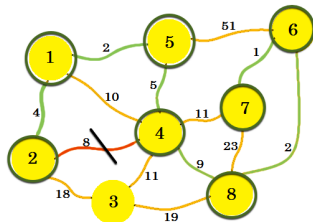
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8), (6, 8), (6, 7)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (3, 4), (4, 7), (7, 8)\};$

Costo parziale: 22.



Non possiamo selezionare il lato $(1, 4)$ altrimenti otterremmo il ciclo 1-4-5.

Selezioniamo il lato $(3, 4)$ e aggiungiamo a S il vertice 3:

Vertici già visitati:

$S = \{1, 5, 2, 4, 8, 6, 7, 3\};$

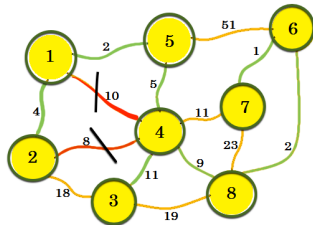
Lati in soluzione:

$\{(1, 5), (1, 2), (4, 5), (4, 8), (6, 8), (6, 7), (3, 4)\}$

Lati selezionabili:

$\{(1, 5), (5, 6), (2, 3), (4, 7), (7, 8)\};$

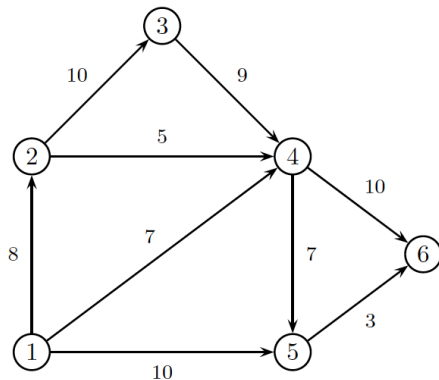
Costo totale: 34.



Il problema dei cammini minimi (*Shortest Path*)

Consideriamo un grafo $G = (V, A, w)$ diretto, connesso e pesato (pesi non negativi).

Dato un vertice di partenza s , si vuole calcolare la lunghezza (il costo) di tutti i cammini minimi per raggiungere gli altri vertici.



L'algoritmo di Dijkstra (II)

Indichiamo con $d[u]$ il valore del cammino minimo da s al nodo u ; segniamo il nodo di provenienza tra parentesi (e.g., se il parent di 4 è 1, allora scriveremo $d[4] (1)$).

Procedimento:

① Inizializzazione:

- selezionare un nodo di origine s , inserirlo in un insieme S e impostare $d[s] := 0$;
- per ogni nodo $u \neq s$, impostare $d[u] := \infty$ (*nil*).

② Finché S non è vuoto:

- visitare (i.e., rimuovere) il nodo u in S avente distanza minore;
- inserire i nodi raggiungibili da u in S e aggiornare le loro distanze, calcolando $\min\{d[v], d[u] + w(u, v)\}$, dove v è raggiungibile da u .

Strategia: anche Dijkstra costruisce un albero connesso e aciclico (all'inizio fa parte della soluzione solo il nodo di origine s).

Complessità: $\Theta(|E| + |V| \log |V|)$, usando come strutture dati heap di Fibonacci, adatte alle code di priorità.

Risoluzione esercizio con Dijkstra (I)

Impostiamo $s = 1$ e costruiamo una tabella che, iterazione dopo iterazione, aggiorneremo:

NODO	ITER 0 DIST. (PADRE)	ITER 1 DIST. (PADRE)	ITER 2 DIST. (PADRE)	ITER 3 DIST. (PADRE)	ITER 4 DIST. (PADRE)	ITER 5 DIST. (PADRE)
2	∞ (nil)					
3	∞ (nil)					
4	∞ (nil)					
5	∞ (nil)					
6	∞ (nil)					

All'inizio impostiamo per tutti i nodi la distanza ∞ e come nodo padre *nil* (i.e., nulla).

Risoluzione esercizio con Dijkstra (II)

Nodi visitati: {1}.

I nodi raggiungibili dal nodo 1 sono 2, 4 e 5 → Aggiorniamo i loro valori nella tabella:

NODO	ITER 0 DIST. (PADRE)	ITER 1 DIST. (PADRE)	ITER 2 DIST. (PADRE)	ITER 3 DIST. (PADRE)	ITER 4 DIST. (PADRE)	ITER 5 DIST. (PADRE)
2	∞ (nil)	8 (1)				
3	∞ (nil)	∞ (nil)				
4	∞ (nil)	7 (1)				
5	∞ (nil)	10 (1)				
6	∞ (nil)	∞ (nil)				

Ora che abbiamo usato il nodo 1, selezioniamo il nodo più vicino, ossia il nodo 4. Dovremo:

- considerare i nodi raggiungibili da 4;
- eventualmente aggiornare le loro distanze.

Risoluzione esercizio con Dijkstra (III)

Nodi visitati: $\{1, 4\}$.

I nodi raggiungibili dal nodo 4 sono 5 e 6 \rightarrow Aggiorniamo i loro valori nella tabella:

NODO	ITER 0 DIST. (PADRE)	ITER 1 DIST. (PADRE)	ITER 2 DIST. (PADRE)	ITER 3 DIST. (PADRE)	ITER 4 DIST. (PADRE)	ITER 5 DIST. (PADRE)
2	∞ (nil)	8 (1)	8 (1)			
3	∞ (nil)	∞ (nil)	∞ (nil)			
4	∞ (nil)	7 (1)	7 (1)			
5	∞ (nil)	10 (1)	10 (1)			
6	∞ (nil)	∞ (nil)	17 (4)			

Nota:

- Per aggiornare il valore del nodo 5, abbiamo calcolato $\min\{d[5], w(4, 5) + d[4]\} = \min\{10, 7 + 7\} = 10 \rightarrow$ Conviene ancora partire da 1 per arrivare a 5.

Iteriamo ancora selezionando il nodo più vicino, ossia il nodo 2.

Come prima:

- Consideriamo i nodi raggiungibili da 2;
- Eventualmente aggiorniamo le loro distanze.

Risoluzione esercizio con Dijkstra (IV)

Nodi visitati: $\{1, 4, 2\}$.

I nodi raggiungibili dal nodo 2 sono 3 e 4 \rightarrow 4 è già stato visitato, quindi aggiorniamo solo il valore relativo al 3:

NODO	ITER 0 DIST. (PADRE)	ITER 1 DIST. (PADRE)	ITER 2 DIST. (PADRE)	ITER 3 DIST. (PADRE)	ITER 4 DIST. (PADRE)	ITER 5 DIST. (PADRE)
2	∞ (nil)	8 (1)	8 (1)	8 (1)		
3	∞ (nil)	∞ (nil)	∞ (nil)	18 (2)		
4	∞ (nil)	7 (1)	7 (1)	7 (1)		
5	∞ (nil)	10 (1)	10 (1)	10 (1)		
6	∞ (nil)	∞ (nil)	17 (4)	17 (4)		

Continuiamo selezionando ora il nodo 5.

Risoluzione esercizio con Dijkstra (V)

Nodi visitati: {1, 4, 2, 5}.

Il nodo 5 può raggiungere solo il nodo 6 → Aggiorniamone il valore.

NODO	ITER 0 DIST. (PADRE)	ITER 1 DIST. (PADRE)	ITER 2 DIST. (PADRE)	ITER 3 DIST. (PADRE)	ITER 4 DIST. (PADRE)	ITER 5 DIST. (PADRE)
2	∞ (nil)	8 (1)	8 (1)	8 (1)	8 (1)	
3	∞ (nil)	∞ (nil)	∞ (nil)	18 (2)	18 (2)	
4	∞ (nil)	7 (1)	7 (1)	7 (1)	7 (1)	
5	∞ (nil)	10 (1)	10 (1)	10 (1)	10 (1)	
6	∞ (nil)	∞ (nil)	17 (4)	17 (4)	13 (5)	

Ora toccherebbe allo stesso nodo 6, ma non può raggiungere niente quindi lo riteniamo visitato e passiamo all'ultimo nodo rimasto, ossia il 3.

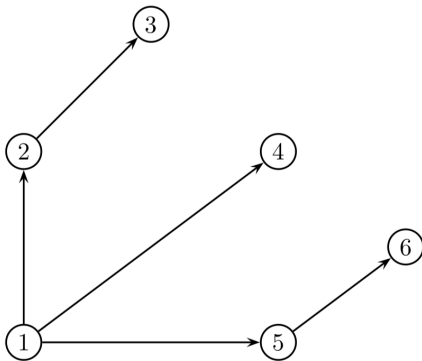
NODO	ITER 0 DIST. (PADRE)	ITER 1 DIST. (PADRE)	ITER 2 DIST. (PADRE)	ITER 3 DIST. (PADRE)	ITER 4 DIST. (PADRE)	ITER 5 DIST. (PADRE)
2	∞ (nil)	8 (1)	8 (1)	8 (1)	8 (1)	8 (1)
3	∞ (nil)	∞ (nil)	∞ (nil)	18 (2)	18 (2)	18 (2)
4	∞ (nil)	7 (1)	7 (1)	7 (1)	7 (1)	7 (1)
5	∞ (nil)	10 (1)	10 (1)	10 (1)	10 (1)	10 (1)
6	∞ (nil)	∞ (nil)	17 (4)	17 (4)	13 (5)	13 (5)

Nodi visitati: {1, 4, 2, 5, 6, 3}.

Risoluzione esercizio con Dijkstra (VI)

Nodi visitati: $\{1, 4, 2, 5, 6, 3\}$.

Albero dei cammini minimi:



- Wikipedia, *Algoritmo di Kruskal*,
https://it.wikipedia.org/wiki/Algoritmo_di_Kruskal
- Wolfram MathWorld, *Kruskal's Algorithm*,
<https://mathworld.wolfram.com/KruskalsAlgorithm.html>
- RIP Tutorial, *Introduzione all'algoritmo di Prim*,
<https://riptutorial.com/it/algorithm/example/24246/introduzione-all-algoritmo-di-prim>
- it-swarm-dev, *Qual è la differenza tra l'algoritmo di Dijkstra e Prim?*,
<https://www.it-swarm.dev/it/algorithm/qual-e-la-differenza-tra-lalgoritmo-di-dijkstra-e-prim/1070017448/>
- Università di Pisa, *Esercizi di PL su grafi* http://groups.di.unipi.it/~a006137/esercizi1_pl_grafi.pdf