Lezioni di Ricerca Operativa

Corso di Laurea in Informatica

Università di Salerno

Lezione n° 19

- Algoritmo di Kruskal
- Algoritmo di Prim

Prof. Cerulli – Dott.ssa Gentili – Dott. Carrabs

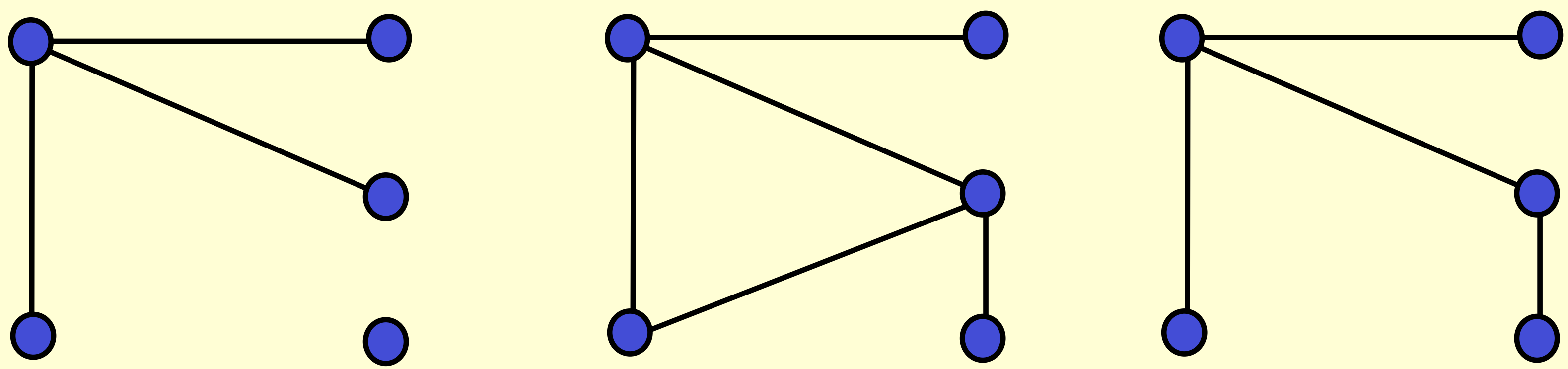
Alberi

Un grafo è **aciclico** se non contiene cicli (orientati o non)

Un **albero** è un grafo connesso ed aciclico

Ogni grafo aciclico è in generale l'unione di alberi e viene detto **foresta**

Esempio



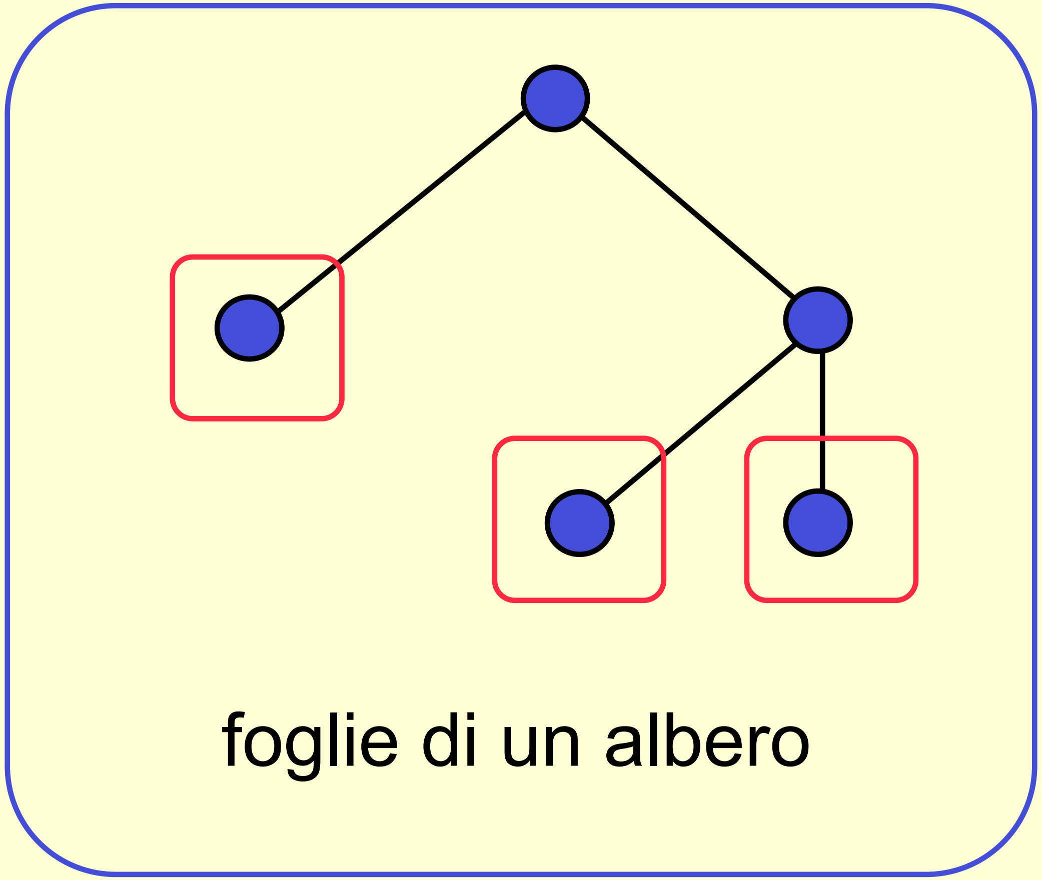
grafo aciclico (foresta) — grafo non aciclico — albero

Dato G=(V,E), le seguenti affermazioni sono equivalenti:

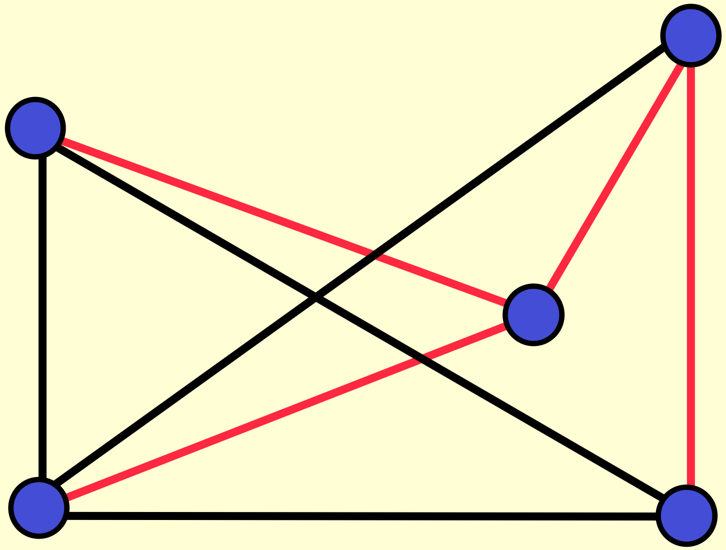
- G è un albero
- ogni coppia di nodi di G è connessa da un unico cammino
- G è aciclico e $|E|=|V|-1$
- G è connesso e $|E|=|V|-1$

Dato un albero $T=(V,E)$, si definisce foglia di T un qualsiasi nodo $v \in V$ tale che $|\delta(v)|=1$.

Se $|V| \geq 2$ allora esistono almeno due foglie.



Dato G=(V,E), si dice **albero ricoprente** (**spanning tree**) di G un albero T=(V',E') con V'=V ed E'⊆E (è un sottografo di G)



un albero ricoprente

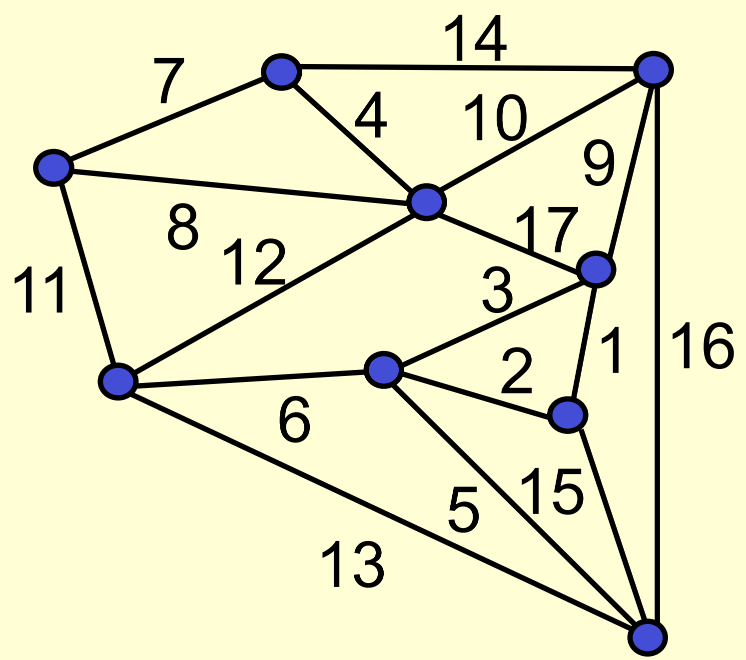
Il Problema del Minimo Albero Ricoprente (Minimum Spanning Tree Problem)

Si considera un grafo G=(V,E)

Ad ogni arco e_i, i=1,...,n di G è associato un costo c_i, i=1,...,m

<u>Il problema</u>: determinare l'albero ricoprente di G con il minimo costo associato.

Esempio



Esempi di applicazioni:

- determinare la rete di comunicazione più affidabile
- determinare la connessione tra n centri a costo minimo (e.g., distribuzione del gas)

Due algoritmi:

- l'algoritmo di Kruskal (Greedy Algorithm)
- l'algoritmo di Prim

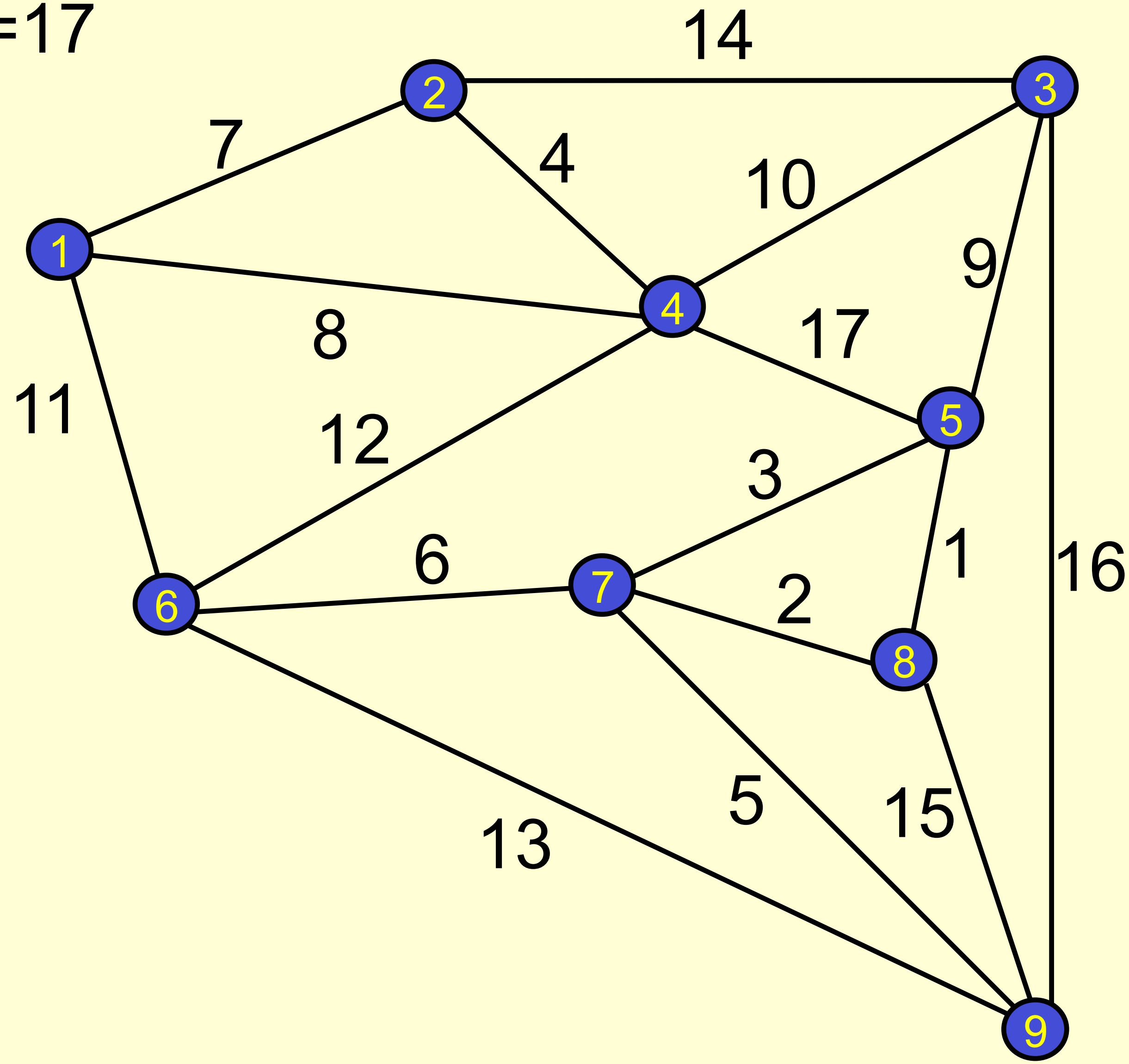
Algoritmo di Kruskal (Minimum Spanning Tree)

(1) E' dato il grafo $G=(V,E)$ con n nodi ed m archi.
Si ordinano gli archi $e_1, e_2, \ldots, e_m$ in modo che i costi associati non siano decrescenti ($c_1 \leq c_2 \leq \ldots \leq c_m$).
Si pone $E^0=\emptyset$, $k=1$ ed il grafo $ST^0=(V, \emptyset)$

(2) Se $(V, E^{k-1} \cup \{e_k\})$ è un grafo aciclico allora $ST^k=(V,E^k)$ con $E^k=E^{k-1} \cup \{e_k\}$, altrimenti $E^k=E^{k-1}$ e $ST^k= ST^{k-1}$

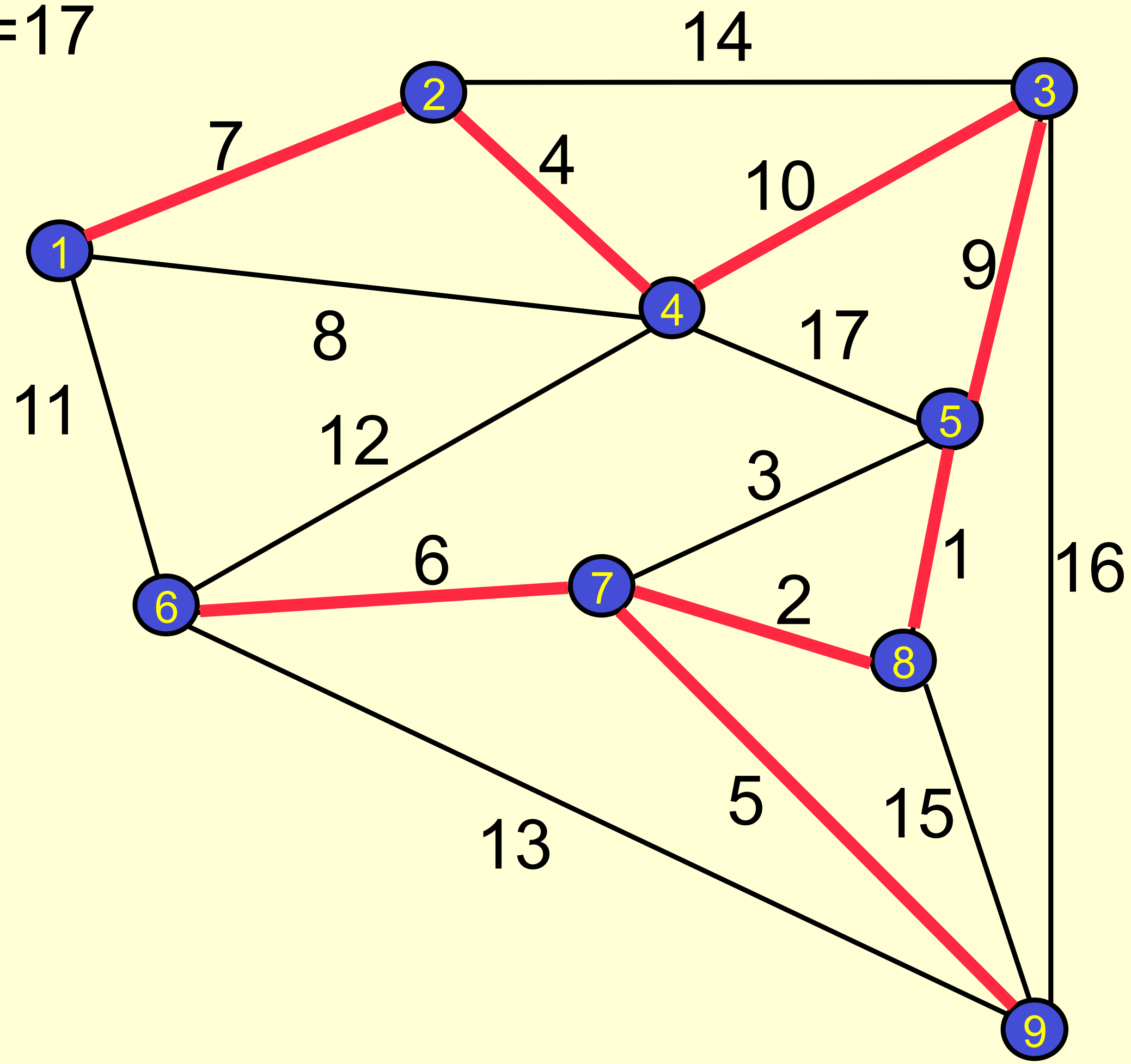
(3) Se $|E^k|=n-1$ l'algoritmo si ferma ed ST^k è l'albero ricoprente cercato, altrimenti $k=k+1$ e continuare col passo (2).

Esempio: n=9 m=17



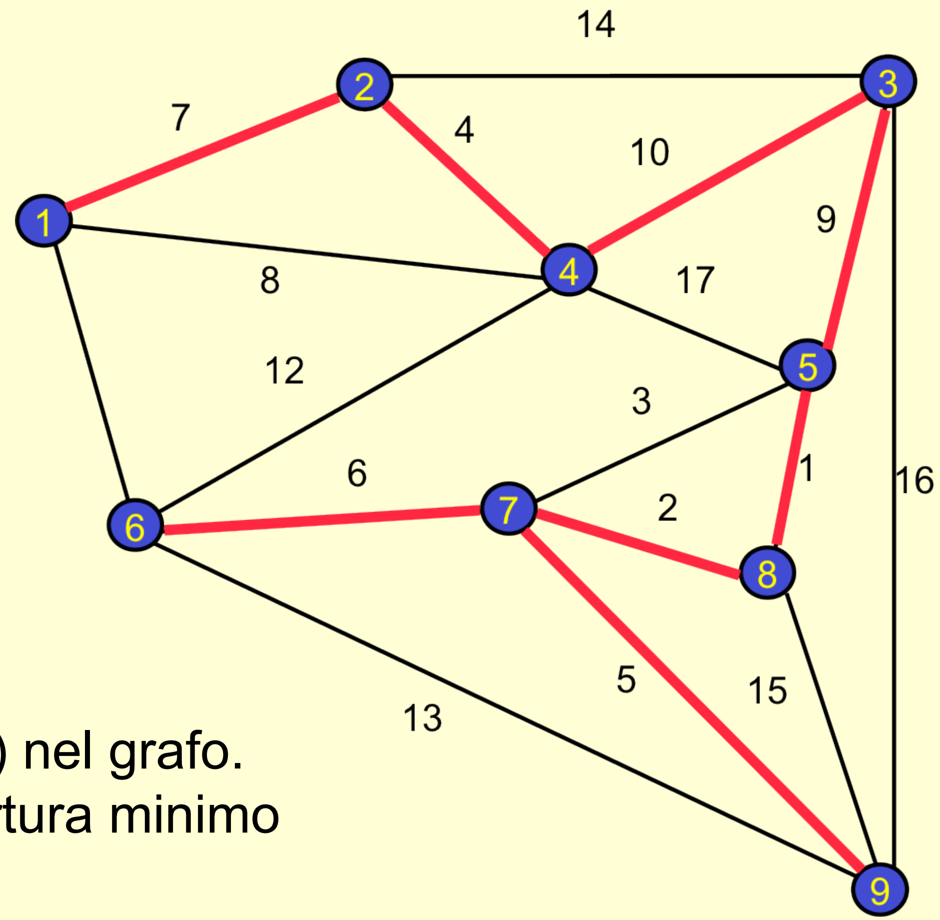
(5,8)	(7,8)	(5,7)	(2,4)	(7,9)	(6,7)	(1,2)	(1,4)	(3,5)	(3,4)	(1,6)	(4,6)	(6,9)	(2,3)	(8,9)	(3,9)	(4,5)
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Esempio: n=9 m=17



~~(5,8)~~	~~(7,8)~~	~~(5,7)~~	~~(2,4)~~	~~(7,9)~~	~~(6,7)~~	~~(1,2)~~	~~(1,4)~~	~~(3,5)~~	~~(3,4)~~	(1,6)	(4,6)	(6,9)	(2,3)	(8,9)	(3,9)	(4,5)
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Esempio: n=9 m=17



- Sia 4+k il nuovo peso dell'arco (2,4) nel grafo. Per quali valori di k l'albero di copertura minimo non cambia?
- Sia 8+k il nuovo peso dell'arco (1,4) in G. Per quali valori di k l'arco (1,4) verrà inserito nella soluzione ottima?

~~(5,8)~~	~~(7,8)~~	~~(5,7)~~	~~(2,4)~~	~~(7,9)~~	~~(6,7)~~	~~(1,2)~~	~~(1,4)~~	~~(3,5)~~	~~(3,4)~~	(1,6)	(4,6)	(6,9)	(2,3)	(8,9)	(3,9)	(4,5)
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Algoritmo di Prim (Minimum Spanning Tree)

(1) E' dato il grafo $G=(V,E)$ con n nodi ed m archi.
Si sceglie un vertice arbitrario di G, $V^0=\{v_s\}$, si pone $E^0=\emptyset$ e $k=1$

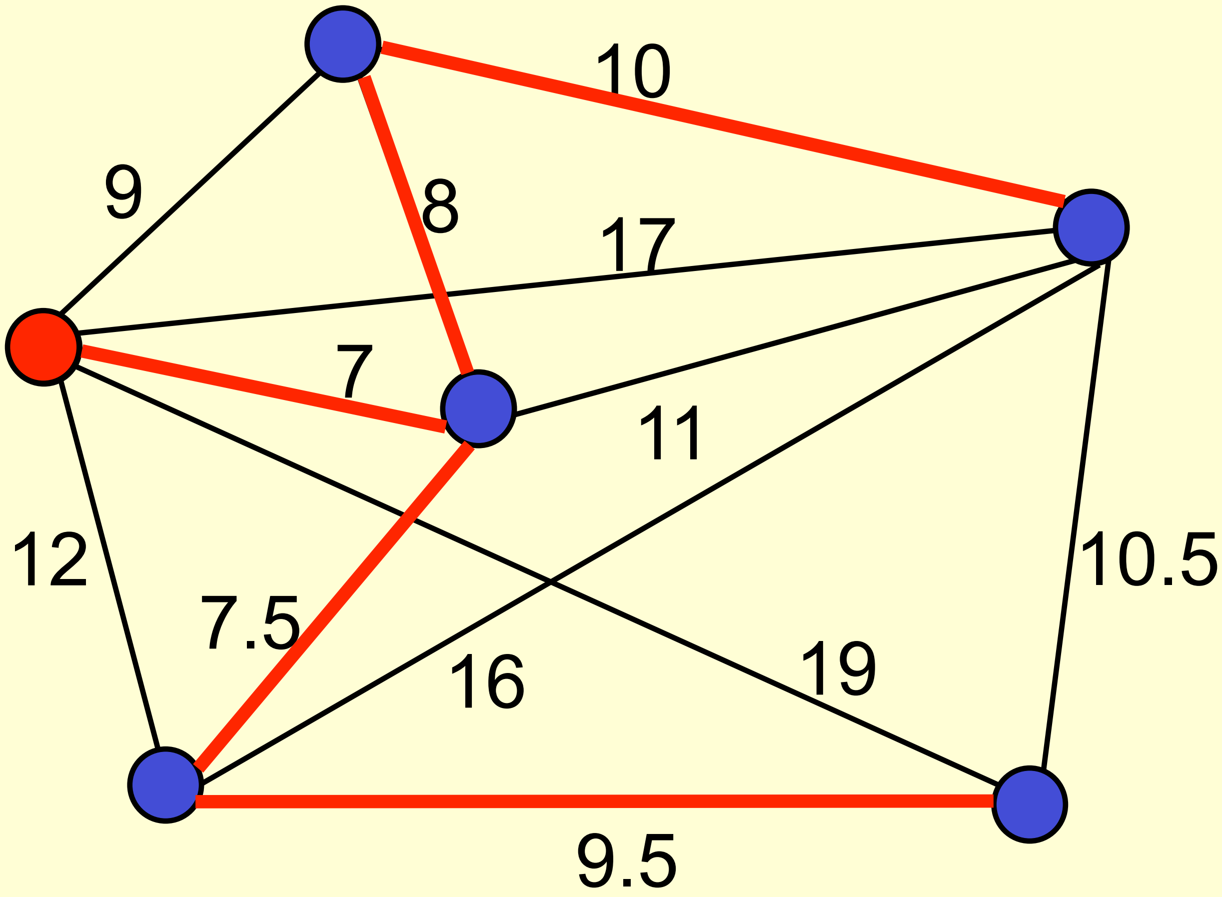
(2) Si connette un nodo $v_i \in V^{k-1}$ con un nodo $v_h \in V - V^{k-1}$ tale che il costo dell'arco (v_i,v_h) sia

$$c(v_i,v_h) = \min_{\substack{v_j \in V^{k-1}, v_e \in V - V^{k-1} \\ (v_j,v_e) \in E}} \left[c(v_j,v_e) \right]$$

e si pone $V^k=V^{k-1} \cup \{v_h\}$ e $E^k=E^{k-1} \cup \{(v_i,v_h)\}$

(3) Se $|E^k|=n-1$ l'algoritmo si ferma e $ST=(V^k,E^k)$ è l'albero ricoprente cercato, altrimenti $k=k+1$ e continuare col passo (2).

Esempio: n=6 m=12



L'algoritmo di Prim O(|E|log|V|) è più efficiente di quello di Kruskal O(|E|log|E|).