Problema del commesso viaggiatore (TSP)

Istanza: Un grafo G indiretto, completo e pesato con pesi interi e positivi.

Problema: Trovare un tour in G di costo minimo.

Complessità di TSP

- TSP può essere **verificato** in tempo polinomiale: dato un tour, calcolo il suo costo sommando il costo di tutti i suoi archi e verifico se è minore o uguale a k. Quindi TSP $\in$ **NP**.
- E' dimostrabile che tutti i problemi in **NP** possono essere **ridotti** a TSP, quindi TSP è **NP-completo**.

Fino a oggi, nessuno ha mai trovato un algoritmo che risolve TSP in tempo polinomiale.

Problemi difficili

Di fronte a problemi difficili (NP-completi) possiamo procedere in uno dei seguenti modi:

- identificare casi particolari del problema per i quali esiste una soluzione polinomiale;
- risolvere il problema nella sua generalità solo su istanze piccole;
- usare un algoritmo che velocemente restituisce una soluzione parziale (**algoritmo incompleto**) o una soluzione prossima a quella ottima (**algoritmo di approssimazione**).

Approx-TSP

Vediamo un algoritmo di approssimazione per TSP chiamato Approx-TSP. Un algoritmo incompleto per TSP con soglia si ottiene facilmente a partire dall'algoritmo di approssimazione.

Sia $G = (V, E, w)$ un grafo indiretto completo con pesi non negativi. Facciamo l'ipotesi che la funzione peso w soddisfi la **disuguaglianza triangolare**: per ogni $u, v, w \in V$,

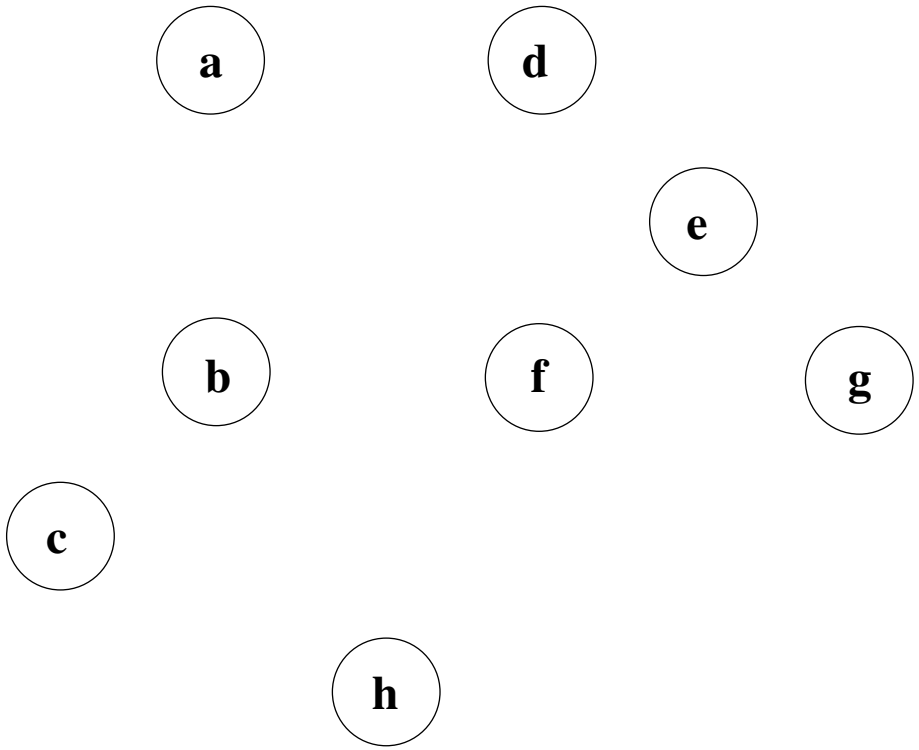
$$w(u, w) \leq w(u, v) + w(v, w)$$

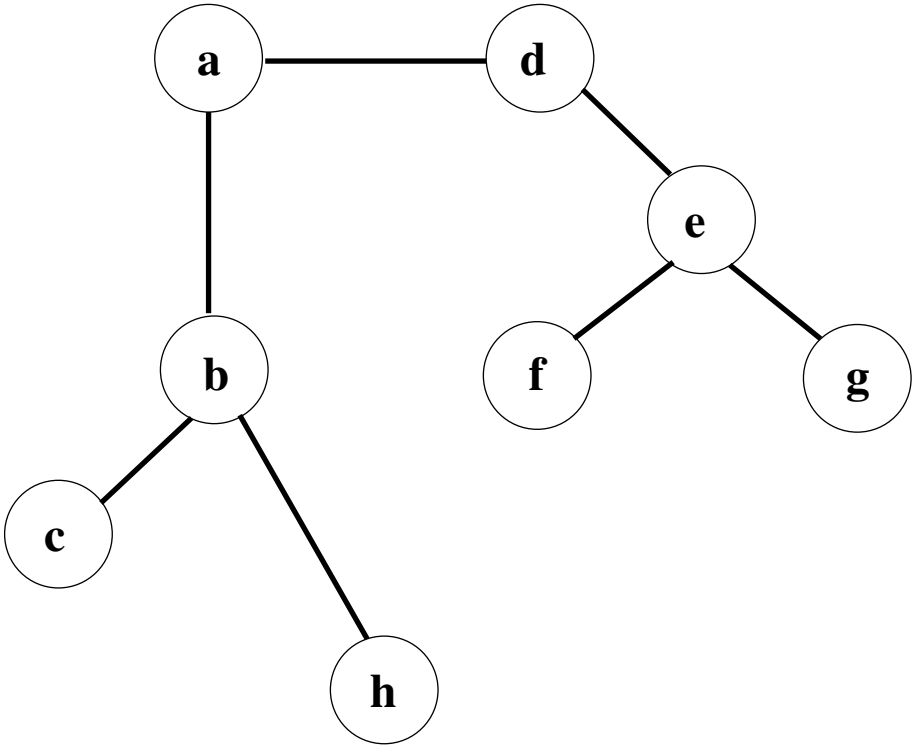
Se w è una metrica (per esempio la **distanza Euclidea**), allora w soddisfa la disuguaglianza triangolare.

Approx-TSP

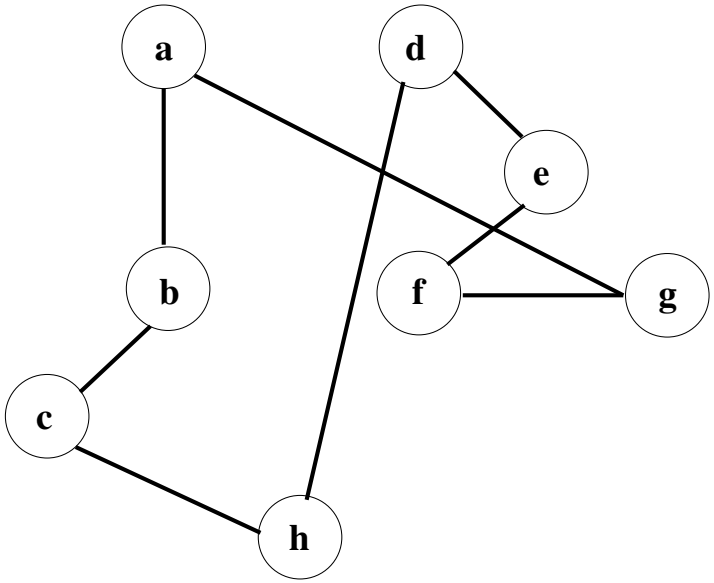
Approx-TSP è il seguente algoritmo di approssimazione per TSP:

1. scegliere a caso una radice $r \in V$;
2. calcolare un albero T di supporto di costo minimo per G con radice r usando l'algoritmo di Prim;
3. visitare i nodi di T in ordine anticipato e inserire i nodi in una lista C;
4. appendere la radice r alla fine della lista C e restituire C.


a
d
e
b
f
g
c
h

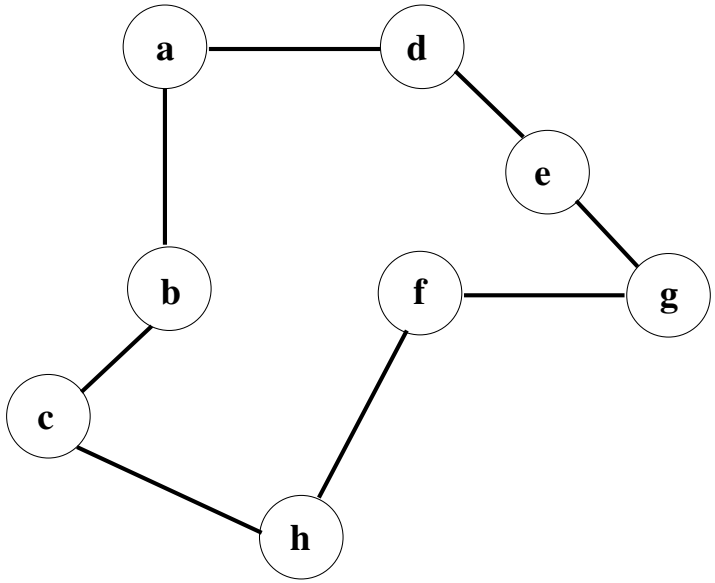


C = a, b, c, h, d, e, f, g, a



C = a, b, c, h, d, e, f, g, a

w(C) = 19,074



C* = a, b, c, h, f, g, e, d, a

w(C*) = 14,715

Complessità di Approx-TSP

Approx-TSP ha **complessità polinomiale**: il passo 1 ha costo costante, il passo 2 ha costo $\Theta(m \log n)$, il passo 3 ha costo $\Theta(n)$ e il il passo 4 ha costo costante.

Dato che il grafo è completo, $m = \Theta(n^2)$, dunque il costo di Approx-TSP risulta

$$\Theta(n^2 \log n)$$

Grado di approssimazione di Approx-TSP

Teorema *Nel caso di grafi pesati che soddisfano la disuguaglianza triangolare, la soluzione fornita da Approx-TSP ha costo minore o uguale al* ***doppio*** *del costo della soluzione ottima.*

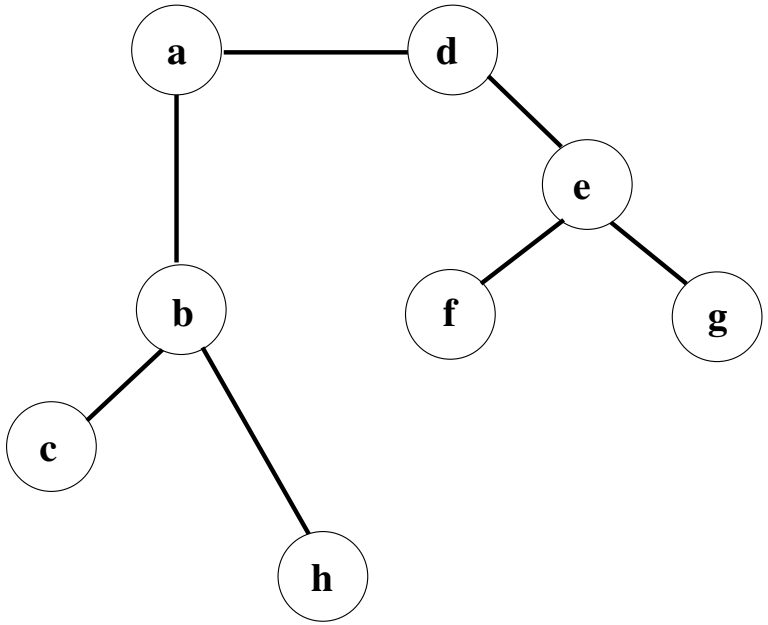
Sia C il tour fornito da Approx-TSP e C^* il tour di costo minimo. Occorre dimostrare che

$$w(C) \leq 2w(C^*)$$

Sia T l'albero di supporto di costo minimo. Dato che togliendo un arco qualsiasi a C^* ottengo un albero di supporto, allora

$$w(T) \leq w(C^*)$$

Sia P il cammino ottenuto seguendo la visita di T in ordine anticipato e inserendo in P ogni nodo quando incontrato:



P = a, b, c, b, h, b, a, d, e, f, e, g, e, d, a

Si noti che:

- P passa esattamente due volte per ogni arco di T. Dunque $w(P) = 2w(T)$;
- il tour C è ottenuto elimimando da P le occorrenze ripetute di ogni nodo. Dunque, per la disuguaglianza triangolare, $w(C) \leq w(P)$.

Quindi:

$$w(C) \leq w(P) = 2w(T) \leq 2w(C^*)$$