**Minimum Spanning Tree**

Dato un grafo orientato pesato, si vuole determinare tra tutti gli alberi di supporto quello che ha il peso, o costo, minore.

Il passo successivo è quello di identificare una procedura in grado di risolvere il problema.

L’algoritmo più semplice è **greedy per MST**

Una particolare classe di algoritmi, caratterizzato per la scelta della migliore opzione per quella particolare interazione (non garantisce di trovare la migliore soluzione, ma per alcuni problemi l’algoritmo greedy è in grado di trovare una soluzione ottima).

L’algoritmo greedy prevede diverse fasi:

**INIZIALIZZAZIONE:** Si ordinano tutti gli archi in ordine crescente rispetto al peso.

Successivamente si eseguono i seguenti passi.

1. Verifichiamo se il numero di archi è pari al numero dei nodi – 1, se questa condizione è vera allora si termina e si restituisce l’albero come soluzione, altrimenti si prosegue con il PASSO 2.
2. Identifichiamo l’attuale arco (arco k) da esaminare e verifichiamo se forma cicli.

Se l’arco forma dei cicli allora viene scartato, altrimenti viene aggiunto a et.

1. Si pone k = k + 1. (Si passa all’arco successivo).

Quando scriviamo un algoritmo dobbiamo valutare la correttezza e la complessità, l’output consiste in un albero di supporto a peso minimo (il peso minimo non è una condizione banale e quindi va dimostrata).

DIMOSTRAZIONE

Supponiamo per assurdo che esista un albero di supporto T’ a peso minimo con peso inferiore a T ovvero l’albero fornito come output dall’algoritmo greedy.

Indichiamo con eh l’arco con peso più piccolo tra ET e ET’.

Aggiungiamo eh ad ET’, in tal caso si forma un ciclo che contiene esattamente un arco er∈ ET (gli archi in ET formano un albero di supporto e quindi non possono generare cicli), allora necessariamente il peso di er è >= di eh altrimenti l’algoritmo greedy lo avrebbe selezionato.

Iterando questo procedimento con tutti gli archi otteniamo nuovamente l’albero T con un peso non superiore a T’, il quale contraddice la condizione iniziale.

Complessità algoritmo greedy

L’operazione più costosa in questo algoritmo consiste nella proceduta di inizializzazione, ovvero di ordinamento degli archi in base al peso, inoltre, la complessità fa riferimento alla densità del grafo.

La complessità dell’algoritmo greedy è O(| E | log(| E |)).

(non è richiesta la dimostrazione perché fa uso di strutture dati particolari)

Il numero di archi che si hanno in un grafo completo è: [n\*(n-1)]/2.

Non tutti i grafi sono densi, infatti se si considera un grafo a griglia, il numero di archi che incide su ogni nodo è dell’ordine di grandezza di n e non di n^2.

RISULTATO

Chiamiamo foresta di supporto di un grafo G un grafo parziale F = (V, EF ) di G privo di cicli. In particolare, un albero di supporto è una foresta con una sola componente connessa.

TEOREMA

Teorema Indichiamo con (V1, E1), . . . ,(Vk, Ek) le componenti connesse di una foresta di supporto F = (V, EF ) del grafo G. Sia (u, v) un arco a peso minimo tra quelli con un solo estremo in V1. Allora, tra tutti gli alberi di supporto a peso minimo tra quelli contenenti ∪ki=1Ei, ce ne è almeno uno che contiene (u, v).

*DIMOSTRAZIONE*

Per assurdo si supponga che ci sia un albero di supporto T = (V, ET ) con ET ⊇ ∪ki=1Ei e di peso minore rispetto a tutti quelli che contengono (u, v), dove ipotizziamo u ∈ V1 e v 6∈ V1. Aggiungiamo (u, v) a ET . In tal caso si forma esattamente un ciclo che oltre a (u, v), deve contenere anche un altro arco (u′, v′) con u′ ∈ V1 e v′ 6∈ V1 (altrimenti il ciclo che parte da u ∈ V1 non potrebbe chiudersi). Per come è definito (u, v), si deve avere che wuv ≤ wu′v′ . Se ora togliamo (u′, v′) otteniamo un albero di supporto a peso non superiore a T, che contiene tutti gli archi in ∪ki=1Ei (quindi anch’esso a peso minimo tra gli alberi che contengono tali archi) e che contiene anche (u, v), il che contraddice l’ipotesi iniziale.