**Problema dell’albero ricoprente di peso minimo, Kruscal e Prim-Dijkstra**

1. Un **albero ricoprente** di un grafo connesso è un albero che collega tutti i nodi del grafo; si vuole individuare un albero ricoprente tale

che la somma dei pesi degli archi appartenenti all’albero sia minimizzata.

2. **Taglio fondamentale**: dato un grafo connesso ed un suo albero ricoprente si può notare che la rimozione di un qualunque arco

dall’albero ricoprente generi due sottoalberi. Un albero contiene n-1 archi e quindi ci saranno n-1 tagli fondamentali. Aggiungendo un arco appartenente al taglio si ottiene nuovamente un albero ricoprente.

**Condizioni di ottimalità sui tagli**: sia T\* un albero ricoprente di costo minimo, un arco e apparterrà all’albero ricoprente di costo minimo se e solo se esiste un taglio fondamentale in G tale che l’arco e minimizzi il costo degli archi tagliati.

e: archi appartenenti all’albero (ce =costo arco).

f: archi non appartenenti all’albero (cf = costo arco).

Dimostrazione:

Supponiamo per assurdo che esista un arco e che minimizza un taglio, ma che tale arco non appartenga all’albero ricoprente di costo minimo. Aggiungendo all’albero ricoprente l’arco e si otterrà un ciclo fondamentale, e ci sarà almeno un altro arco che appartiene al taglio, sia questo arco f. Varrà la condizione cf > ce, quindi rimuovendo f dall’albero di costo minimo si avrebbe un albero a costo inferiore segue l’assurdo.

Dato T\* dobbiamo mostrare come sia possibile costruire un taglio tale che l’arco e sia l’arco di costo minimo tra gli archi appartenenti al taglio. A tal fine consideriamo nell’insieme S una delle due componenti connesse che si verrebbe a generare a seguito della rimozione dell’arco e. L’arco e quindi appartiene ad un taglio e necessariamente sarà il minimo in quel taglio altrimenti l’albero T\* non sarebbe una soluzione ottima.

**Condizioni di ottimalità sui cammini**: un albero ricoprente T\* è

minimo se e solo se soddisfa ce <= cf per ogni arco f di G e per ogni arco e contenuto nel cammino che connette i due nodi terminali di f.

Supponiamo per assurdo che T\* sia ottimo e che esista un arco e nel cammino tra i nodi terminali dell’arco f che non soddisfi la

condizione. Se vale ce >= cf allora introdurre l’arco f nell’albero ricoprente al posto dell’arco e produrrebbe un albero a costo inferiore di T\*, contraddicendo l’ipotesi iniziale. Siano S e S generati dal taglio legato alla rimozione di e, sia l’arco f nel taglio [S, V − S]. Visto che T\* contiene un unico cammino che unisce le estremità di f, tale cammino dovrà passare per l’arco e. L’ipotesi implica ce <= cf. Considerando che questa condizione deve essere valida per ogni arco f nel taglio [S, S] , allora T\* soddisfa le condizioni di ottimalità dei tagli da cui T\* deve essere un albero a costo minimo.

3. **Algoritmo di Kruscal**

a. Albero ricoprente vuoto.

b. Si ordinano gli archi in ordine crescente di lista.

c. Si scorre la lista prendendo ad ogni passo l’arco con peso minore:

i. Se l’arco aggiunto genere un ciclo, non si aggiunge un albero ricoprente.

ii. Se l’arco congiunge due componenti a costo minimo, apparterrà all’albero.

d. Si ottiene un albero ricoprente di costo minimo.

La complessità dell’algoritmo è O(n log(n)+n2).

4. **Algoritmo di Prim-Dijkstra**

a. Albero ricoprente vuoto.

b. Si considera un taglio fondamentale differente.

c. Si cerca l’arco che minimizza il peso tra gli archi appartenenti al taglio corrente.

d. Si aggiorna il taglio corrente con l’arco selezionato nell’albero ricoprente.

e. L’iterazione verrà ripetuta finché non si è costruito l’intero albero.

La complessità dell’algoritmo è di O(n3).

**Problema di cammino minimo e Floyd-Warshall**

1. Il **problema di cammino minimo** consiste nel trovare una sequenza senza ripetizioni di vertici ed archi in un grafo orientato e pesato tale che sia minimizzato il costo della sequenza.

2. Un algoritmo per risolvere il caso in cui siano presenti archi di peso negativo (ammesso sempre che non esistono cicli a peso negativo) è l’algoritmo di Floyd-Warshall.

**Enunciato**: il cammino minimo da un nodo i ad un nodo j è composto dal cammino ottimo da i a k e dal cammino ottimo da k a j. L’algoritmo è corretto perché ad ogni iterazione aggiunge un nodo al sotto-grafo indotto considerato e valuta l’operazione triangolare a partire da quel nodo verso tutte le altre coppie di nodi già considerate; al crescere del sotto-grafo,aggiorno i cammini). Al termine avremo un ciclo a peso negativo o la matrice corretta dei costi dei cammini minimi.

3. **Dimostrazione**:

a. L’operazione triangolare: per ogni coppia di nodi i,j si vede se per passare da i a j, conviene passare per k.

b. Si procede per induzione:

i. Passo base: per k=0, c[i,j] è il costo effettivo del cammino da i a j nel sotto-grafo indotto da {i,j}.

ii. Passo induttivo: supponiamo che la proprietà sia verificata all’iterazione k -1 e si considera un cammino minimo Pij da i a j nel sottografo indotto da {1, …, k} U {i, j} e sia c(Pij) il suo costo. A questo punto si possono verificare due casi:

1. Pij non passa dal vertice k, allora c(Pij) = c[i, j] per l’ipotesi induttiva;

2. Pij passa dal vertice k, allora Pij = Pik U Pkj.

c(Pij) = C[i,k] + c[k,j] -> c(Pij) = min {c[i,j], c[i,k] + c[k,j]} e dopo l’operazione triangolare su k si ha che: c[i,j] = c(Pij).

**Problema di massimo flusso e Ford-Fulkerson**

1. Il **problema di massimo flusso** consiste nel trovare il modo migliore per trasferire entità da un punto ad un altro in una rete; ogni arco è

caratterizzato da una capacità massima, ovvero da un limite massimo di entità che possono attraversarlo. Nei problemi di flusso il grafo è

capacitato, ovvero gli archi hanno una portata massima che non può essere superata, mentre nei problemi di cammino minimo gli archi

hanno un loro peso.

2. **Cammino aumentante**: è un cammino da S a P in cui gli archi diretti non sono saturi e gli archi inversi non sono scarichi. Flusso massimo corrisponde capacità minima di taglio.

**Rete residua**: è un grafo con n nodi, e per ogni arco (i,J) del grafo originale definisco:

• Arco diretto: (i,j) con capacità residua Kij - Xij.

• Arco inverso: (i,j) con capacità residua Xij.

Se nella rete residua c’è un cammino aumentante è possibile aumentare il flusso di una quantità pari alla minima capacità residua.

Se non esiste un cammino aumentante, esiste un taglio che separa la sorgente dal pozzo.

Tutti gli archi del taglio diretto sono saturi.

Tutti gli archi del taglio inverso sono scarichi.

3. **Dimostrazione**: per assurdo, dico che:

il taglio non esiste -> posso raggiungere un nuovo nodo. Così via fino a raggiungere il pozzo; quanto l’ho raggiunto ho trovato

un cammino aumentante. Ma il cammino aumentante non può esistere dal momento che ho un taglio.

**Problema di flusso di costo minimo**

1. Il **problema di flusso di costo minimo** consiste nel far giungere il “prodotto” realizzato nei nodi sorgente ai nodi destinazione facendolo viaggiare attraverso la rete e cercando di spendere il meno possibile per il trasporto. Ciò che viene prodotto nei nodi sorgente è esattamente pari a ciò che viene consumato nei nodi destinazione.

2. Un algoritmo per risolverlo è il simplesso su reti.

HP: il grafo è connesso con n nodi (ne potrei eliminare 1). Base di A <-> Albero ricoprente del grado. Imposto il duale:

a. Trovo u risolvendo: uj-ui = Cij, per ogni (i,j) in base e fissando u1 = 0.

b. Verifico l’ammissibilità duale di u: uj-ui <= Cij per ogni (i,j) fuori base. Se uj-ui > Cij allora xij entra in base, esce l’arco discorde con il

minimo flusso presente nel ciclo formato da xij entrante.

3. **Teorema**: se un grafo G è connesso, la base di A è un albero ricoprente di G.

Dimostrazione: una base di A contiene necessariamente (n-1) colonne di A; (n-1) colonne di A che non formino una base dell’albero ricoprente devono necessariamente contenere un ciclo. Le colonne di A sono linearmente dipendenti, ovvero che esiste una combinazione lineare delle colonne con coefficienti non tutti nulli, a somma 0. Dato

un ciclo, i coefficienti della combinazione lineare possono ottenersi semplicemente percorrendo il ciclo in un verso qualsiasi e fissando il coefficiente dell’arco (i,j) del ciclo:

• 1 se l’arco è concorde al verso di percorrenza del ciclo;

• -1 se l’arco è discorde al verso di percorrenza.

Sommando le colonne si ottiene una colonna nulla, il che dimostra che queste tre colonne sono linearmente dipendenti e quindi non

possono far parte contemporaneamente di una base di A.

4. **Teorema**: un grafo G è connesso se e solo se il rango di A è uguale a n-1. Dimostrazione: albero ricoprente =>base di A.

Sempre per le motivazioni di prima, essendo le colonne linearmente dipendenti, il rango non può essere massimo, ma solo n-1, ma dal

fatto che la disponibilità totale di tutti i nodi sorgente è uguale alla domanda totale di tutti i nodi pozzo, il problema ammetterà sempre

soluzione e una delle equazioni può essere cancellata.

**Definizione base ammissibile:**

Ponendo che la soluzione di un sistema Ax=b sia x= =

La soluzione base (e, per estensione, la base B stessa) si dice soluzione base ammissibile, o SBA , se xB=B-1b≥0.

**Definizione vertice**:

-Un punto x di un poliedro P si dice punto di estremo o vertice di P se non può essere espresso come una combinazione convessa stretta di altri due punti del poliedro, cioè non esistono y.z ϵ P, y≠z e λϵ(0,1) tali che x=λy+(1-λ)z.

-Un punto x ϵ P è un vertice del poliedro non vuoto P: ={x≥0 : Ax=b} se e solo se x è una soluzione base ammissibile del sistema Ax=b.

Dimostrazione: (una soluzione ammissibile di un problema di PL in forma standard è un vertice del poliedro delle soluzioni ammissibili se e solo se è una soluzione base ammissibile)

Dimostriamo prima l’implicazione x SBA→ x vertice. Supponiamo per assurdo che una soluzione x ϵ P sia una SBA e non un vertice P. Senza perdita di generalità possiamo raggruppare le componenti positive di x e quelle nulle, ovvero assumiamo: x=[x1,…,xK,0,…,0]T positive dove k rappresenta il numero di componenti non nulle(cioè positive) di x. Ne consegue che le colonne A1,…,Ak devono fare parte di una qualsiasi base B associata alla SBA x, insieme eventualmente ad altre colonne(SOLUZIONE DEGENERE).

-Se x non è un vertice di P, esistono due punti : y=[y1,…,yK,0,…0]T ϵ P ; z=[z1,…,zK,0,…0]T ϵ P con y≠z, tale che x=λy+(1-λ)z, per un qualche λϵ(0,1).

Si noti che y e z devono necessariamente avere le ultime componenti a zero, altrimenti la loro combinazione convessa non potrebbe dare x. Per ipotesi si ha allora: y ϵ P→Ay=b → A1y1+…+AKyK=b ; z ϵ P→Az=b → A1z1+…+AKzK=b

sottraendo la seconda equazione dalla prima si ottiene (y1-z1)A1+…+(yK-zK)AK=α1A1+…+ αKAK=0 , ove si è posto αi =yi-zi, i=1,…,k. Esistono quindi scalari αi ,…, αk non tutti nulli(dato che y≠z) tale che , , pertanto le colonne A1,…,AK sono linearmente dipendenti, e non possono fare parte di una fase, contraddicendo l’ipotesi x SBA.

Dimostriamo ora l’implicazione x vertice→x SBA.

Per dimostrare l’implicazione è sufficiente che x vertice→x soluzione base. Il fatto che la soluzione base sia anche ammissibile deriva infatti dall’ipotesi x ϵ P. Supponiamo per assurdo che x sia un vertice di P, ma non una soluzione base del sistema Ax=b. Ipotizzando come prima x=[x1,…,xK,0,…,0]T , con x1,…, xK >0, si ha che x ϵ P → Ax=b → A1x1+…+ AKxK=b, le colonne A1 ,…,AK sono linearmente dipendenti, e quindi esistono k coefficienti α1,…, αK non tutti nulli tali che: α1A1+…+ αKAK=0. Sommando la prima equazione e la seconda moltiplicate per ε>0 si ottiene: A1(α1+ ε α1)+…+AK(αK+ ε αK)=b.

***Definizione direzione estrema****:*

Un vettore d ϵ Rn di norma unitaria(cioè tale che ||d||=1 ) si dice direzione di un poliedro P se u≥0, x ϵ P x+ud ϵ P.

Una direzione d ϵ Rn di un poliedro P si dice direzione estrema di P se non può essere espressa come una combinazione conica stretta di altre due direzioni di P.

***Teorema di Minkowski-Weyl:***

Ogni punto di un poliedro dotato di almeno un vertice si può ottenere come somma di una combinazione convessa dei suoi vertici e di una combinazione conica delle sue direzioni estreme.

Teorema: Dato un PL min{CTx : x ϵ P}, con P poliedro contenente almeno un vertice, se esiste una soluzione ottima del problema, esiste un vertice di P ottimo.

Dimostrazione: Siano x1,…,xK i vertici di P e siano d1,…,dh le sue direzioni estreme. Sia infine z\*=min{CT xi : i=1,…,k}.

Per dimostrare la tesi del teorema basta dimostrare che, dato un qualunque y ϵ P, si ha cTy≥z\*. Dal lemma si ha che cTdi≥0, per i=1,…,h. Devono esistere moltiplicatori u1,…uh≥0 e λ1,…, λK≥0, , tali che y= . Si ha allora cTy=cT .

***Definizione minimo globale****:*

Una soluzione x\* ϵ X si dice punto di minimo globale per f(x), o soluzione ottima, se: f(x\*)≤f(x) x ϵ X. In questo caso f(x\*) si dice minimo globale di f(x) in X. Un punto di minimo globale è stretto se f(x\*)<fx) x ϵ X, x≠x\*.

***Definizione minimo locale:***

Una soluzione ϵ X si dice punto di minimo locale per f(x) se: ϵ>0 : f( )≤f(x) x ϵ X : ||x- ||<ϵ. Un punto di minimo locale è stretto se f(x\*)<f(x) x ϵ X : ||x- ||<ϵ, x≠ .

***Definizione di insieme convesso e funzione convessa:***

L’intersezione di k insiemi convessi X1,…,XK Rn è un insieme convesso. Un insieme X Rn si dice convesso se:

X, λ ϵ [0,1] z=λx+(1-λ)y ϵ X.

Dati un insieme convesso X e una funzione f:x→Rn, si dice che f è una funzione convessa su X se comunque presi due punti x,y ϵ X e uno scalare λ ϵ [0,1] e detto z=λx+(1-λ)y, si ha che: f(z)≤λf(x)+(1-λ)f(y).

***Definizione di Programmazione Convessa:***

Un PM si dice problema di Programmazione convessa se l’insieme ammissibile x è convesso e la funzione obiettivo f(x) è convessa su x. Un punto di minimo locale è anche detto di minimo globale(solo nella programmazione convessa) : Sia un punto di minimo locale, ovvero tale che ε>0 : f( )≤f(x) x ε X: ||x- ||<ε. Sia y ε X una generica soluzione ammissibile. Dalla convessità di x discende il fatto che λ ε [0,1] il punto z=λ +(1-λ)y ε X. E’ sempre possibile scegliere un valore di x sufficientemente vicino a 1 tale che sia verificata la condizione ||z- ||<ε, il che implica f( )≤f(x), la convessità di f implica: f(z)≤λf( )+(1-λ)f(y). Unita alla precedente: f( )≤λf( )+(1-λ)f(y). Portando λf( ) a primo membro e dividendo per (1-λ) segue la tesi.