TSP

In questa documentazione c’è la descrizione della classe TSP programmata per essere eseguita esclusivamente nel compito di ricerca operativa.

La classe tsp ha un vettore di dimensione 10, con in più un albero da allocare, che sarà di 3 piani.

Il vettore è di 10 dimensioni, perché sappiamo con certezza che i nodi sono 5 e che il tsp è simmetrico, per cui ci bastano 10 celle.

L’albero binario ci servirà per il tsp e per stampare i risultati.

La classe tsp ha molte funzioni di utilità, ma poche funzioni pubbliche, tra cui:

* Costruttore tsp(), che alloca in memoria un vettore di 10 celle, i cui valori saranno stabiliti dall’utente e ogni cella corrisponde alla combinazione (i,j) con riga i e colonna j, disposta in ordine lessicografico Xij. Ci sarà possibilità di controllare con una stampa diretta se la digitazione sia stata corretta e verrà chiesto di confermare con 1 la combinazione o di ripeterla con 0 e correggerla, indicando il numero di errori, e poi per ciascun di essi di indicare rispettivamente il valore corretto e la posizione in cui si sarebbe verificato l’errore.  
  L’albero è vuoto
* Distruttore ~tsp, che distrugge l’albero
* Branch\_Bound, funzione che si comporta come segue e con l’ordine così descritto:

1. Viene chiesto di digitare rispettivamente il nodo per il k\_albero e il nodo più vicino a k;
2. Poi la funzione chiama la funzione che calcola il k\_albero, indicando i nodi scelti e in cui non è collegato il nodo k scelto(l’albero di copertura), e poi gli archi in cui è coinvolto il nodo k. Viene poi stampato la valutazione inferiore (Vi) che è la somma dei pesi degli archi;
3. Poi viene chiamata la funzione che stampa gli archi scelti in ordine secondo quanto previsto dal nodo più vicino a k e la valutazione superiore(Vs);
4. Poi chiede di digitare per 3 volte di fila i nodi che partono da i e arrivano a j (i,j), cioè secondo le convenzioni del corso di ricerca operativa dei nodi (i,j), per poter allocare l’albero completo;
5. Chiama la funzione branch, una funzione di utilità che svolgerà i conti necessari.

Branch

La funzione Branch è la funzione ricorsiva che svolge i conti. Come funziona? Dato l’albero binario allocato, come descritto nel paragrafo precedente, crea un vettore fittizio che ci terrà in mente quali nodi includere ed escludere nell’albero, fa una visita anticipata dell’albero e:

* Nel caso in cui fossimo al nodo al piano 0, allora si stampa il seguente messaggio: “Testa albero: [Vi,Vs] = [(valore Vi),(valore Vs)]” e non fa nessun conto, in quanto questo è già avvenuto in precedenza;
* Nel caso in cui non fossimo al piano 0, si può seguire le indicazioni fornite dall’albero stesso che ci indicherà quale posizione della cella è da includere o se da escludere e poi includerà o escluderà quel nodo
* Durante l’inclusione del nodo ci si assicura che: che non vengano inclusi nodi che generino un ciclo disgiunto o 3 archi nel nodo k. In caso contrario, viene stampato “Vuoto” e si esce dalla chiamata
* Durante l’esclusione del nodo ci si assicura che tutti gli archi da collegare ad un certo nodo riga o colonna dell’albero abbia almeno 2 nodi collegabili. Se a quel nodo vengono vietati i collegamenti con altri 3 nodi, la soluzione sarà vuota e per cui verrà stampato “Vuoto” e si esce dalla chiamata
* Calcola il k\_albero del nodo corrente, considerando eventuali nodi tolti da aggiornare nel vettore, o già scelti, anch’essi da aggiornare nel vettore della classe
* Controlla se la valutazione inferiore non sia più grande di quello superiore: nel caso in cui lo fosse viene restituita la funzione e viene stampato il seguente messaggio : “ Taglio Vi = (inserire valore) che è > (valore di Vs)”
* Controlla se il ciclo è hamiltoniano o se Vi = Vs. Per entrambi i casi viene specificato la motivazione del taglio, con conseguente aggiornamento di Vs e ritorno della funzione. Nel primo caso viene stampato: “Taglio Vs = (inserire valore Vi) = (inserire valore Vs), in quanto ciclo hamiltoniano” o “Taglio Vs = (inserire valore Vi) = (inserire valore Vs), in quanto Vs e Vi hanno stesso valore”
* Superati i controlli, si stampa il seguente messaggio: “[[Vi,Vs] = [(valore Vi),(valore Vs)]]”

Ogni salto è una chiamata ricorsiva e vengono gestiti gli aggiornamenti. Nelle funzioni di utilità gli input sono sanitizzati e provvedono a chiudere forzatamente l’esclusione con il codice di uscita 1.

Attenzione: la visita dell’albero sono anticipate, per cui verrà stampato il risultato secondo l’ordine previsto dalla stampa anticipata

Attenzione: Il problema di questo algoritmo è che la visita è anticipata, per cui consiglio di controllare gli aggiornamenti di Vs dal nodo sinistro, per eventuali errori di questa natura, cioè non sbaglia a calcolare i nodi, ma aggiorna con ordine di visita anticipata, per cui dobbiamo fare alcuni accorgimenti e sistemare eventualmente l’albero. Per maggiori informazioni, consultare la sezione “Bug” dove vengono descritti i bug della funzione e varie considerazioni da attuare, in modo da utilizzare in totale sicurezza l’algoritmo

Facciamo un esempio:

15 26 66 47 99 58 58 12 9 15

La tabella e' la seguente:

2 3 4 5

1 15 26 66 47

2 - 99 58 58

3 - - 12 9

4 - - - 15

Digitare 1 per confermare la combinazione e 0 per correggere l'errore

1

Digitare rispettivamente gli indici per il k\_albero e il nodo vicino

5 1

Archi non collegati a 5:(3,4)

Archi non collegati a 5:(1,2)

Archi non collegati a 5:(1,3)

Archi collegati a 5: (3,5),(4,5)

Vi = 77

I nodi violati sono:

Nodo: 2 con equazione: X12 + X23 + X24 + X25 = 2

Nodo: 3 con equazione: X13 + X23 + X34 + X35 = 2

Archi: (1-2),(2-4),(3-4),(3-5),(1-5),Vs = 141

Scrivere quali sono i nodi interessati per il Branch e Bound per 3 volte di fila, rispettando l'ordine in cui si desidera istanziarle:

5 1

2 3

1 2

Digitare 1 per confermare la combinazione,0 per ripeterla

0

3 5

3 4

2 3

Digitare 1 per confermare la combinazione,0 per ripeterla

1

Testa albero: [Vi,Vs] = 77,141

[Vi,Vs] = [115,141]

Taglio: Vi = 161 che e' > 141

[Vi,Vs] = [115,141]

[Vi,Vs] = [115,141]

Taglio: Vi = 188 che e' > 141

[Vi,Vs] = [77,141]

Taglio: Vs = 123 = 123, in quanto ciclo hamiltoniano.

[Vi,Vs] = [77,123]

[Vi,Vs] = [77,123]

Vuoto

Qui noterete che Vs è anticipato in ritardo, per cui state attenti e controllate che ogni nodo riceva l’aggiornamento corretto. Il rischio sarà quello di avere per lo più nodi di troppo, piuttosto che conti in più da fare

Bug

I bug rilevati sono i seguenti:

1. Aggiornamento Vs = Vi corretto se il ciclo hamiltoniano o Vi = Vs a sinistra e non sempre a destra.

Ci aspetteremmo che i nodi, a partire da una certa altezza h, fino a scendere alle foglie, siano tutti aggiornati, almeno che siano già stati tagliati, anche se l’aggiornamento accade in quel punto, ma a causa della visita anticipata questo non accade sempre, per cui viene stampato un Vs che sarebbe da aggiornare, oltre al fatto che vengano considerati un Vs diverso.

1. Bug Grafico: a volte appare un “+” di troppo nelle equazioni dei nodi che hanno violato il vincolo di grado