RELAZIONE ESAME 31/01/2019 (18 punti)

Francesco Cheinasso S239345

STRUTTURA DATI

Per la memorizzazione dei dati è stata usata una struttura a grafo sfruttando le librerie ST.h Graph.h e Edge.h

In particolare graph_t sfrutta una matrice di adiacenza e una di incidenza per memorizzare i legami tra i vari nodi.

La tabella di simboli è stata inserita all'interno della struttura graph_t . Essa salva i nomi dei nodi e li associa agli indici della tabella sfruttando il quasi adt node_t, definito anch'esso in ST.h.

Notare l'utilizzo nella funzione load della funzione itemsetvoid() che ritorna una variabile globale settata con valore nullo, per considerare i casi in cui una ricerca produce un risultato vuoto. Tale variabile viene allocata una sola volta e restituita per evitare sprechi di memoria.

VERIFICA KERNEL

Tale verifica è fatta utilizzando la funzione iskernel().

Essa riceve in input un vettore v di lunghezza g->v contenente 1 o 0 ad indicare una partizione dei nodi del grafo (1 = nodo $\in$ K, 0 = nodo $\in$ V-K), in quanto gli indici del vettore corrispondono alla posizione dei nodi nella tabella di simboli.

La funzione itera lungo v analizzando 2 casi:

- V[i]=0 : si verifica l' incidenza di almeno un nodo considerato $\in$ K in V[i]
- V[i]=1 : si verifica la non adiacenza di V[i] nei nodi considerati $\in$ K

Entrambi i casi vengono gestiti tramite un'iterazione sulla matrice di adiacenza o d'incidenza.

VERIFICA FILE

Per verificare il file viene generato un vettore di input, utilizzando i nomi del file per ottenere i nodi dalla tabella di simboli e di conseguenza settare a 1 i loro indici corrispondenti in d[i]. Infine viene ritornato il risultato della iskernel() utilizzando d come input.

KERNEL MASSIMO

Per trovare il kernel con cardinalità massima è stato utilizzato un powerset che genera tutte le possibili partizioni di cardinalità 2 utilizzando i blocchi 0 e 1.

La funzione wrapper kernelMax() si limita a inizializzare il vettore sol a 0, ad allocare solmax e a stampare su file il risultato ottenuto.

combR() ricorre creando le partizioni.

Nella condizione di terminazione viene verificato se sol è un kernel usando iskernel() e, se sol contiene più 1 rispetto al massimo finora trovato, sol viene salvato in solmax.

CAMMINO MASSIMO SUI NODI DEL KERNEL

E’ stato utilizzato come input il kernel massimale trovato al punto precedente, tanto che viene fatto un controllo per accettarsi che solmax contenga il risultato corretto.

In seguito si cerca il cammino massimo ottenuto partendo da ciascun nodo del grafo usando pathkernelR(). Questa funzione non è altro che una dfs (con backtrack) che conta il numero di nodi kernel attraversati e salva il numero massimo di nodi attraversati insieme al numero di passi.

PRINCIPALI CAMBIAMENTI DALLA VERSIONE CARTACEA AL CODICE CONSEGNATO

- In load si è utilizzato l’indice della variabile nulla piuttosto che la variabile stessa, per effettuare il confronto;
- stput() ritorna l’indice aggiunto per evitare una ricerca lineare inutile;
- Cambiata la condizione di verifica in iskernel() nel caso v[i]=0 da conta != 1 a conta < 1;
- stgetName() non utilizza solmax[i], ma i;
- Aggiunta stampa in kernelpath();
- Eliminata variabile E in graph_t;
- Errore nell’acquisizione: tentativo di acquisire il peso del grafo (risolto modificando la fscanf e inserendo archi con peso 1);
- In combR aggiunti return nella fase di terminazione e seconda chiamata ricorsiva;
- Aggiunto controllo su solmax in kernelpath();