### 03MNO ALGORITMI E PROGRAMMAZIONE CORSO DI LAUREA IN INGEGNERIA INFORMATICA A.A. 2018/19

# Commento al Laboratorio n. 6

#### Esercizio n.1: Vertex cover

Si crea una struct arco con 2 campi interi u e v per memorizzare un arco tramite i 2 vertici su cui esso insiste e un vettore di archi a di E elementi.

Il problema richiede di generare tutti i sottoinsiemi degli N vertici e di verificare se tutti gli archi del grafo hanno almeno uno dei vertici su cui insistono nel sottoinsieme corrente. Il modello per generare tutti i sottoinsiemi è quello del powerset, l'implementazione con divide et impera, disposizioni ripetute o combinazioni semplici è indifferente. Se si fosse dovuto risolvere un problema di ottimizzazione, ad esempio ritornare il vertex cover di cardinalità minima, allora l'implementazione del powerset con combinazioni semplici sarebbe stata conveniente in quanto, grazie all'iterazione che fa crescere la dimensione della combinazione corrente, ci si sarebbe potuti fermare alla prima soluzione valida, che era per costruzione anche a cardinalità minima.

Visto che le scelte sono i vertici e visto che i vertici sono identificati da interi nell'intervallo 0..N-1, non è necessario registrare le scelte in un vettore val. Si presentano 2 soluzioni:

- 1. powerset costruito con le disposizioni ripetute: la soluzione sol è un vettore di N elementi, ciascuno dei quali indica se il vertice corrispondente all'indice fa o no parte della soluzione. Nella condizione di terminazione si chiama una funzione check per validare la soluzione corrente: essa è scartata se e solo se per almeno uno degli archi del grafo entrambi i vertici non appartengono al sottoinsieme corrente
- 2. powerset costruito con le combinazioni semplici: nel main si itera la chiamata alla funzione delle combinazioni semplici con dimensione k crescente da 1 a N. La soluzione sol è un vettore di k elementi, ciascuno dei quali è un vertice della soluzione. Una funzione check verifica se per ogni arco esiste nella soluzione corrente almeno un vertice su cui l'arco insiste.

# Esercizio n.2: Anagrafica con liste

**Menu:** come nell'es. 4 del Lab. 4, invece di utilizzare un tipo definito per enumerazione per i comandi, nella soluzione proposta si utilizzano esplicitamente gli interi come indici di un vettore di stringhe che li contiene.

### **Dati:** si presentano 2 soluzioni per il tipo Item:

- 1. versione con Item con dati composti per valore in cui i campi stringa sono vettori di caratteri sovrallocati. Per gestire i casi particolari si introduce un Item vuoto (caratterizzato da codice vuoto), creato dalla funzione ItemSetVoide riconosciuto dalla funzione ItemCheckVoid. La funzione leggiItem chiama al suo interno la scomponiData per trasformare la data da stringa a struct con campi interi peri giorno, mese ed anno. Tutte le funzioni che ricevono o ritornano Item lo passano per valore, facendo sempre quindi copie dei dati. Alcune funzioni, ad esempio quella di stampa, potrebbero in alternativa ricevere puntatori ad Item.
- 2. versione con Item con dati composti per riferimento in cui i campi stringa sono vettori di caratteri allocati dinamicamente. All'item si accede unicamente tramite puntatore. La funzione ItemNew alloca un nuovo item e lo inizializza con i dati passati come parametri. La funzione leggiItem alloca un item mediante ItemNew e ne trasferisce il possesso al programma chiamante, che si occupa di deallocarlo quando non più necessario mediante la ItemFree. Non è più necessario gestire con ItemSetVoid e ItemCheckVoid il caso di item vuoto, in quanto basta usare puntatori NULL.



# 03MNO ALGORITMI E PROGRAMMAZIONE CORSO DI LAUREA IN INGEGNERIA INFORMATICA A.A. 2018/19

Vista la semplicità dell'item non si introducono funzioni di accesso o confronto sulle chiavi.

Collezione di dati: il nodo della lista è definito secondo la modalità 3 (*Puntatori e strutture dati dinamiche 4.1.1*). La creazione di un nuovo nodo è fatta come in *Puntatori e strutture dati dinamiche 4.1.3*. La funzione insertOrdinato (*Puntatori e strutture dati dinamiche 4.1.3*) trasferisce alla lista la proprietà dell'item ricevuto. La ricerca ricercaCodice, essendo per codice mentre l'ordinamento è per data, è una ricerca su lista non ordinata e quindi non può sfruttare l'ordinamento per un'interruzione anticipata in caso negativo. Le funzioni di eliminazione elimina e eliminaTraDate ritornano l'item al programma chiamante, cui viene trasferita la proprietà e la responsabilità di deallocazione.