Esercitazione 9
Minimum Spanning Tree

Corso di Fondamenti di Informatica II

BIAR2 (Ing. Informatica e Automatica) e BSIR2 (Ing. dei Sistemi)

A.A. 2013/2014

13 Dicembre 2013

Sommario

Scopo di questa esercitazione è calcolare il minimo albero ricoprente di un grafo pesato.

1 Minimo albero ricoprente

Il problema del *minimum spanning tree* (MST) consiste nel calcolare, per un dato grafo pesato, un sottografo albero a peso minimo che ricopra tutti i nodi del grafo.

Uno degli algoritmi per il calcolo del MST è l'algoritmo di Kruskal. Basandosi sul materiale di supporto, si vuole implementare una classe `Kruskal` per il calcolo del MST.

L'algoritmo di Kruskal è a sua volta basato su una struttura dati che mantiene una *partizione* dei nodi del grafo. Prima di poter implementare la classe `Kruskal` sarà quindi necessario implementare una classe `Partition` che implementi tale struttura dati.

L'algoritmo di Kruskal utilizza anche una *coda di priorità* per ordinare gli archi del grafo. Tale struttura dati è già disponibile, in quanto implementata precedentemente nel corso.

Materiale di supporto. Viene fornita la classe `Graph` che rappresenta un grafo semplice non orientato, a pesi interi (con nodi aventi tipo `Integer`). È inclusa la classe `Edge` per rappresentare archi del grafo.

Viene anche già fornita la classe `MinHeap<V>` (assieme alla sua classe ausiliaria `HeapEntry<V>`) che realizza una coda di priorità di oggetti di tipo `V`.

Viene infine fornita una classe `Test` che contiene un `main` di prova per testare sia `Partition` che `Kruskal`.

Programma Java. Realizzare una classe `Partition` contenente i seguenti metodi.

```
public Partition(Collection<Integer> S)
```

Crea una partizione in cui ogni elemento di `S` forma un insieme a se stante. Il costo del metodo deve essere $O(n)$.

```
public List<Integer> find(int key)
```

Recupera la lista contenente la chiave `key`. Il costo del metodo deve essere $O(1)$.

```
public void union(int u, int v)
```

Fonde le liste contenenti le due chiavi u e v, aggiornandone i riferimenti. Il costo del metodo deve essere $O(\min(n_u, n_v))$, dove n_u, n_v sono il numero di elementi degli insiemi cui fanno capo u e v, rispettivamente.

Realizzare inoltre una classe `Kruskal` contenente il metodo statico

```
public static Graph computeKruskal(Graph G)
```

che restituisce l'albero minimo ricoprente di `G` calcolato secondo l'algoritmo di Kruskal.

Riferimenti bibliografici

[1] M. T. Goodrich and R. Tamassia. *Strutture dati e algoritmi in Java*. Zanichelli, 2007.