L'implementazione della libreria grafo è stata realizzata in modo da essere ottimale nel caso in cui si abbiano dati sparsi e che rispetti le seguenti operazioni con la rispettiva complessità (con "n" numero di nodi o di archi):

- Creazione di un grafo vuoto – O(1)
- Aggiunta di un nodo – O(1)
- Aggiunta di un arco – O(1)
- Verifica se il grafo è diretto – O(1)
- Verifica se il grafo contiene un dato nodo – O(1)
- Verifica se il grafo contiene un dato arco – O(1)
- Cancellazione di un nodo – O(n)
- Cancellazione di un arco – O(1)
- Determinazione del numero di nodi – O(1)
- Determinazione del numero di archi – O(n)
- Recupero dei nodi del grafo – O(n)
- Recupero degli archi del grafo – O(n)
- Recupero nodi adiacenti di un dato nodo – O(1)
- Recupero etichetta associata a una cSoppia di nodi – O(1)

Il grafo può essere sia diretto che non diretto, a discrezione dell'utente che decide tramite linea di comando quale tipologia utilizzare. Sia il tipo dei nodi, sia le etichette degli archi sono implementate genericamente.

Per determinare la foresta minima ricoprente si è implementato l'algoritmo di Kruskal che utilizza la struttura dati "UFSnode.java" implementata nell'Esercizio 3.
La struttura UFSnode :

- **FIND(x):** accede alla foglia a corrispondente all'elemento x. Da tale nodo segue il puntatore al padre, che è la radice dell'albero, e restituisce il nome memorizzato in tale radice.
- **MAKESET(x):** crea un nuovo albero, composto da due nodi: una radice ed un unico figlio (foglia). Memorizza x sia nella foglia dell'albero che come nome della radice.
- **UNION(A,B):** considera l'albero A corrispondente all'insieme di nome a, e l'albero B corrispondente all'insieme di nome b. Sostituisce tutti i puntatori delle foglie di B alla radice di B con puntatori alla radice di A. Cancella la vecchia radice di B.

Algoritmo di Kruskal è un algoritmo ottimo utilizzato per calcolare gli alberi di supporto minimi di un grafo non orientato e con gli archi con costi non negativi.
Si consideri un grafo non orientato e connesso dove V rappresenta il numero di vertici (o nodi) ed E il numero di spigoli (o archi). Ad ogni spigolo è associato un peso (o distanza): lo scopo dell'algoritmo è quello di trovare un albero ricoprente di peso minimo, cioè quello in cui la

somma dei pesi sia minima. L’algoritmo può essere applicato solo se si dispone di due o più vertici.
L’algoritmo di Kruskal si basa sulla seguente semplice idea: ordiniamo gli archi in ordine crescente di costo e successivamente li analizziamo singolarmente, inseriamo l’arco nella soluzione se non forma cicli con gli archi precedentemente selezionati. Notiamo che ad ogni passo, se abbiamo più archi con lo stesso costo, è indifferente quale viene scelto.

Eseguendo l’algoritmo di Kruskal sui dati contenuti nel file “italian_dist_graph.csv” i risultati ottenuti sono i seguenti:

```
il numero di nodi  18640
>>>>Il numero di arco minimi generati  18637<<<<
89939.91258557337
```