ISIMA - Recherche Operationnelle

TP1 - Minimum Weight Spanning Trees

Renaud Chicoisne & Annegret Wagler *20 Janvier 2023*

L'objectif de ce TP est d'implementer l'algorithme de Prim en C (rappelé dans l'Algorithme 1 ci-dessous) et de trouver un arbre couvrant de poids minimum dans un graphe donné.

Algorithm 1: Prim

Data: A connected graph $G = (V, E)$ with edge weights $(c_e)_{e \in E}$

Result: A minimum weight spanning tree $T = (V, E')$ of G

Choose a start node $x \in V$, initialize a tree $T = (V', E')$ with $V' := \{x\}$ and $E' := \emptyset$;

while $|V'| < |V|$ **do**

  Choose an edge $e = (u, v) \in E$ with minimal weight c_e from the set $\{(u, v) \in E : u \in V', v \in V | V'\}$;

  $E' \leftarrow E' \cup \{e\}$, $V' \leftarrow V' \cup \{v\}$;

return T;

Votre code source devra comporter deux fichiers principaux : `TP1.C` et `TP1Functions.c` que vous compilerez avec la commande `gcc TP1.c -o TP1`, ce qui génèrera un fichier executable que vous executerez avec la commande `./TP1`.

Votre code devra lire un fichier d'instance `TP1instance.csv` qui contient toutes les informations du graphe considéré sous le format suivant :

1. La premiere ligne `n,m` contient le nombre de noeuds $n = |V|$ et d'aretes $m = |E|$
2. Chacune des m lignes suivantes `i,j,c` contient l'information d'une arete $(i, j) \in E$ et son poids $c_{ij} =$`c`

Votre code devra retourner les informations relatives a l'arbre trouve dans un fichier `TP1solution.csv` contenant en premiere ligne le poids total de l'arbre et a chacune des $n - 1$ lignes suivantes l'index des aretes composant votre arbre. Testez votre implémentation sur le réseau suivant :

	Aachen	Bonn	Dusseldorf	Frankfurt	Koeln	Wuppertal
Aachen	-	91	80	-	70	-
Bonn	91	-	-	175	27	84
Dusseldorf	80	-	-	-	47	29
Frankfurt	-	175	-	-	189	-
Koeln	70	27	47	189	-	55
Wuppertal	-	84	29	-	55	-

Executez votre programme six fois en selectionnant un noeud initial x different a chaque execution.