Projet SINF1250
« Tous les chemins mènent à Rome ... »

Professeur	Marco Saerens marco.saerens@uclouvain.be
Téléphone	010 47 92 46
Bureau	b.102
Adresse	Université catholique de Louvain Place des Doyens 1 1348 Louvain-La-Neuve Belgique
Assistants	Bertrand Lebichot bertrand.lebichot@uclouvain.be Mathieu Zen mathieu.zen@uclouvain.be
Bureau	b.105;b.106
Date	8 Novembre 2016

Objectif : Le but est d'implémenter et de tester deux algorithmes résolvant le problème du plus court chemin sur des graphes. Le premier algorithme est celui de Dijkstra et le second sera laissé à votre libre choix. Vous travaillerez par groupe de deux étudiants et l'implémentation se fera en Julia - `http://julialang.org/`. Voici un pseudocode de l'algorithme de Dijkstra tiré de votre livre de référence (Rosen) :

ALGORITHM 1 Dijkstra's Algorithm.

procedure *Dijkstra*(G: weighted connected simple graph, with
    all weights positive)
{G has vertices $a = v_0, v_1, \ldots, v_n = z$ and lengths $w(v_i, v_j)$
    where $w(v_i, v_j) = \infty$ if $\{v_i, v_j\}$ is not an edge in G}
for $i := 1$ **to** n
    $L(v_i) := \infty$
$L(a) := 0$
$S := \emptyset$
{the labels are now initialized so that the label of a is 0 and all
    other labels are ∞, and S is the empty set}
while $z \notin S$
    $u :=$ a vertex not in S with $L(u)$ minimal
    $S := S \cup \{u\}$
    for all vertices v not in S
        if $L(u) + w(u, v) < L(v)$ **then** $L(v) := L(u) + w(u, v)$
        {this adds a vertex to S with minimal label and updates the
        labels of vertices not in S}
return $L(z)$ {$L(z) =$ length of a shortest path from a to z}

Il faut suivre rigoureusement cet algorithme (pas de fantaisies svp).

Nous vous demandons d'ajouter une fonctionnalité à votre fonction Julia qui permet, en plus de connaître le coût total du plus court chemin entre deux noeuds, de savoir par *quels noeuds ce chemin est passé*. S'il y a plusieurs chemins les plus courts, vous pouvez renvoyer n'importe lequel d'entre eux. Vous pouvez trouver la description de cette procédure dans tout bon ouvrage d'algorithmique (par exemple l'ouvrage *Introduction to Algorithms* de Cormen, Leiserson, Rivest et Stein).

Fonction Julia :

- **Input :** Une matrice d'adjacence **A** et deux entiers i et j qui sont les indices des noeuds du graphe G entre lesquels on cherche à calculer le plus court chemin.
- **Output :** coût$(i,j) = c_{ij}$, le coût du plus court chemin entre i et j, ainsi qu'un vecteur de nombres comprenant, dans l'ordre, la liste des noeuds par lesquels passent le plus court chemin.

En plus, nous vous demandons d'écrire un script "main" en Julia (dans le même fichier que la fonction qui calcule le plus court chemin) qui charge le fichier de données (la matrice d'adjacence) et qui appelle les fonctions calculant la distance et le chemin le plus court entre les deux noeuds, et les imprime à l'écran. Ces fonctions seront exécutées pour vérifier que votre programme fonctionne bien.

Données : Vous pouvez tester vos algorithmes sur des données de graphes disponibles en ligne - https://networkdata.ics.uci.edu/index.php. Ces données contiennent essentiellement des matrices d'adjacence de graphes représentant divers types de réseaux.
Attention, les pondérations a_{ij} de la matrice d'adjacence peuvent représenter des affinités entres noeuds. Il faudra donc les transformer en coûts c_{ij} en utilisant la formule $c_{ij} = 1/a_{ij}$. Ceci revient a convertir une conductance (affinité) en résistance (coût).

Rapport : Le rapport est un fichier PDF qui se compose de :

- Une courte explication de (1) vos fonctions Julia, (2) de l'algorithme de Dijkstra et de l'autre algorithme choisi ainsi que (3) la méthode utilisée pour déterminer le trajet le plus court.
- Une description des données utilisées et les résultats apportés par votre algorithme sur ces données.
- Une comparaison en termes de complexité et performance des deux algorithmes.
- Le code complet en annexe – n'oubliez pas de bien commenter le code !

Langage de programmation : L'implémentation devra être codée en Julia.

Evaluation et consignes : Le projet est à réaliser par groupes de deux étudiants. L'évaluation portera sur le contenu du rapport et le code (lisibilité, structure, **commentaires**,...) et comptera pour trois points sur 20. Les fichiers, c'est-à-dire l'unique fichier de code source et le rapport pdf, sont à remettre sur Moodle au plus tard le vendredi 22 décembre (dernier jour de cours) avant 23h59. La note sera la même pour tous les membres du groupe.

Bon courage !