Aix*Marseille université

Année 2020-2021

Domaine Sciences et Technologies
LICENCE INFORMATIQUE

Algorithmique 1 : TP Dijkstra
Code UE : S04IN4A2

Vous allez implémenter l'algorithme de Dijkstra pour le calcul des plus courts chemins dans un graphe orienté à partir d'un sommet donné. L'objectif est de comparer expérimentalement différentes représentations de la file de priorité.

Génération des graphes

Définissez la classe `Graph` pour représenter des graphes orientés. Cette représentation devra permettre d'accéder facilement aux voisins d'un sommet donné (voir cours). Vous implémenterez un constructeur pour la classe `Graph` prenant en paramètre le nombre de sommets n et la valeur de la *densité* θ qui construira de façon aléatoire un graphe à n sommets dont la proportion d'arcs présents par rapport au graphe complet est θ. La génération aléatoire consiste, tant que le nombre désiré d'arcs n'est pas atteint, à choisir un arc (u, v) au hasard. Attention, avant d'ajouter l'arc (u, v) au graphe, il faut vérifier qu'il n'y est pas déja.

Algorithme de Dijkstra

Reprenez l'algorithme de Dijkstra vu en cours. Dans un premier temps vous représenterez la file de priorité F de la façon la plus simple, avec un tableau contenant les sommets de F, et en effectuant une recherche séquentielle pour trouver le sommet qui a la distance la plus petite (voir cours, EXTRAIRE_MIN(F) : parcours du tableau).

Dans un deuxième temps vous implémenterez la file de priorité F à l'aide d'un tas binaire (PriorityQueue en Java). Dans ce cas la file de priorité doit contenir non pas des sommets mais des arcs, pondérés par la distance obtenue lors du relachement de cet arc. La méthode reste la même, sauf qu'à chaque étape on extrait un arc (u, v) et une distance l. Si la distance $d[v]$ est meilleure que l, il n'y a rien à faire. Sinon il faut relacher les arcs sortants de v et insérer dans la file ceux qui ont permis de diminuer la distance.

Comparez les performances de l'algorithme de Dijkstra en fonction de la densité du graphe et de l'implémentation de la file de priorité F. Evaluez expérimentalement la densité au delà de laquelle il est préférable d'effectuer une recherche séquentielle pour des graphes à 100 sommets.

Calcul de tous les plus courts chemins

On veut maintenant calculer les plus courts chemins entre toutes les paires de sommets. Implémentez l'algorithme de Floyd-Warshall vu en cours en utilisant une matrice pour représenter les distances.

Une autre approche consiste à appliquer l'algorithme de Dijkstra à partir de chaque sommet du graphe. Comparez les deux approches. En fonction de la densité du graphe, peut-on dire que l'une est meilleure que l'autre ?