SAE 2.02
Partie 2 : Plus Court Chemin

Contact : Mikal.Ziane@u-paris.fr

Modification du 09/04/2022

Etant donné qu'en mathématiques vous avez vu une version simplifiée de l'algorithme de Bellman, vous pouvez supposer dans la méthode estOK de PCCBellman que le graphe doit être sans cycle et non pas sans cycle absorbant. Un graphe qui n'est pas OK doit lever une exception si on appelle la méthode pcc (plus court chemin entre deux nœuds) qui sera une sous classe IllegalArgumentException.

Que ce soit pour Dijkstra ou pour Bellman, estOK n'a pas à vérifier que le graphe est connexe mais s'il n'y a pas de chemin entre les deux nœuds pour lesquels un plus court chemin est demandé alors il faut lever une exception NoPathEx qui sera une sous classe de IllegalArgumentException .

Introduction

Dans cette seconde partie de la SAE vous devez garder le même groupe que pour la première partie. Le thème de cette seconde partie porte sur le calcul du plus court chemin entre deux nœuds d'un graphe dont les arêtes portent une valeur numérique qui pourra ou non être négative selon les exercices.

Exercice 1 Algorithme de Dijskstra

Ecrivez une classe PCCDijkstra qui calcule un plus court chemin entre deux nœuds donnés d'un graphe. Dans le cas où plusieurs plus courts chemins existent, un seul, quelconque, sera retourné. Les nœuds sont indiqués par leur numéro (un entier positif) et le résultat sera une liste de numéros de nœuds. On supposera qu'il existe au plus un seul arc entre deux nœuds donnés.

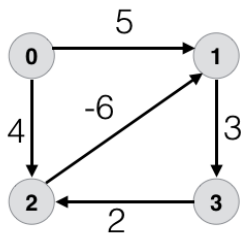
Votre algorithme doit être **générique** : il ne doit **pas dépendre** de la représentation du graphe (Matrice ou listes d'adjacence). Pour cela, vous définirez une interface IGraph qui sera implémentée par les classes GrapheMa et GrapheLA de la première partie de la SAE après les avoir modifiées pour prendre en compte des valuations sur les arcs. Pour cet exercice une méthode booléenne estOK(Igraph) vérifiera que les conditions exigées par l'algorithme sont bien respectées.

Testez votre algorithme sur les exemples 3.1 1 et 3.2 et 3.6.1 de votre polycopié de mathématiques de la ressource R2-07. Pour le graphe 3.6.1 le test exigera qu'une exception de type ArcNEégatifEx est levée (classe que vous définirez comme une sous-classe de IllegalArgumentException).

Notez bien qu'aucune dépendance sur les classes implémentant un graphe ne doit être présente dans la classePCCDijkstra. Cette classe en particulier ne fera aucune création d'une instance de graphe mais prendra en paramètre une instance de IGraph. Cette instance sera créée par la classe de test.

Exercice 2 Algorithme de Bellman

Il s'agit du même exercice que le précédent mais avec l'algorithme de Bellman. Votre classe s'appellera PCCBellman. Vous la testerez en plus sur le graphe suivant (crédit dû à Wikipédia) qui devra être rejeté avec une exception CircuitAbsorbantEx (classe que vous définirez comme une sous-classe de IllegalArgumentException) ce que vérifiera le test.


5
0
1
-6
4
3
2
3
2