Théorie des Graphes

5

RECHERCHE DES « PLUS COURTS » CHEMINS

Sommaire

5.1 Introduction

G=(S,A) un graphe valué.

5.1.1 *Objectif*

Longueur d'un chemin = somme des valeurs associées aux arcs formant le chemin.
Exemples :

- le moins de km à parcourir entre deux villes
- le trajet le plus rapide entre deux stations de métro
- le coût minimum (nombre ou coût d'actions à effectuer) pour transformer une situation d'un état initial à un état solution
- …

5.1.2 *Définitions*

5.1.2.1 *Coût / Poids d'un arc*

Valeur associée à l'arc

5.1.2.2 *Coût cumulé / Coût d'un chemin / Longueur d'un chemin*

Somme cumulée des coûts des arcs constituant un chemin

5.1.3 *Circuits*

Soit un circuit cx = (x,y1,…,yn,x)
Soit un chemin c = (x1,x2,…,x,…,xn)
Soit ci un chemin de x1 à xn qui contienne i fois le circuit cx

Soit coût(cx) > 0 coût (c0) < coût (ci), quelle que soit la valeur de i > 0
donc c0 est le plus court chemin de x1 à xn

Soit coût(cx) = 0 coût (ci) est constant, quel que soit la valeur de i
il y a donc une infinité de « plus court » chemin ci

Soit coût(cx) < 0 coût (ci) < coût (cj) si i > j
donc on ne peut pas trouver de plus court chemin

cx est alors dit « circuit absorbant » (coût négatif dans une recherche de plus court chemin).

La recherche du plus court chemin nécessite l'absence de circuit absorbant

5.1.4 *Résultats des algorithmes*

Selon la méthode utilisée, on calculera :

1. le plus court chemin entre deux sommets donnés
2. le plus court chemin d'un sommet donné à tous les autres
3. les plus courts chemins de tout sommet x à tout autre sommet y

5.2 DIJKSTRA

Recherche des plus courts chemins d'un sommet initial x à tout autre sommet du graphe.

Contraintes :

- Pas de circuit absorbant (contrainte générale au problème)
- Les coûts des arcs sont tous positifs ou nuls

5.2.1 *Principe de fonctionnement*

G = (S , A)

On partitionne l'ensemble des sommets en deux parties :

CC : ceux pour lesquels le « plus court chemin » est connu de façon définitive
S – CC : ceux pour lesquels on peut avoir une longueur déjà calculée, mais celle-ci ne correspond pas nécessairement au plus court chemin

Initialisation :

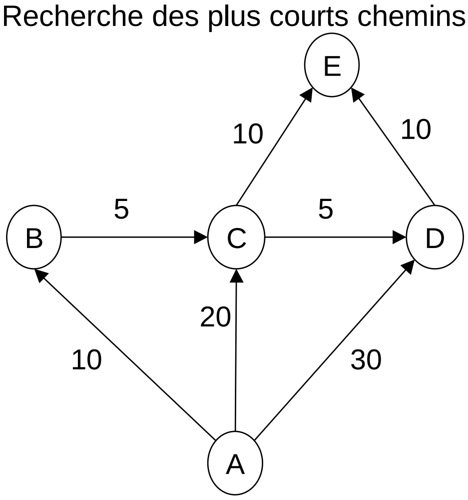
- CC ← { init } ensemble des sommets pour lesquels le plus court chemin est déterminé
- coût chemin (init) ← 0
- pour tout successeur s de init : coût chemin (s) ← coût (init , s) (coût de l'arc)
- pour tout autre sommet x : coût chemin (x) ← infini

Itération jusqu'à ce que CC = S :

- sélectionner x dans S-CC tel que coût chemin (x) soit la valeur minimale de celles associées aux éléments de S-CC
- CC ← C + x (L'algorithme et ses conditions d'utilisation font que la valeur actuelle est celle du plus court chemin)
- pour tout successeur y de x encore dans S-CC :
  coût chemin (y) ← MIN (coût chemin (y) , coût chemin (x) + coût (x , y))

5.2.2 *Exemple*

Recherche des plus courts chemins en partant de A



CC	A	B Min(A+10)	C Min(A+20 , B+5)	D Min(A+30 , C+5)	E Min(C+10 , D+10)
A	0	--	--	--	--
	0	10	20	30	--
B	0	10	A+20 / B+5 0+20 / 10+5 15	A+30 / C+5 0+30 / 20+5 25	C+10 / D+10 20+10 / 30+10 30
C	0	10	15	A+30 / C+5 0+30 / 15+5 20	C+10 / D+10 15+10 / 25+10 25
D	0	10	15	20	C+10 / D+10 15+10 / 20+10 25
E	0	10	15	20	25

5.3 BELLMAN

Recherche du plus court chemin d'un sommet donné à tout autre sommet

Contraintes :

- Pas de circuit absorbant (contrainte générale au problème)

Note : Contrairement à l'algorithme de Dijkstra, celui-ci autorise les arcs à valeur négative.

5.3.1 Principe

Calcul de 'init' à tout y

Soit x1, x2, … les prédécesseurs de y
Le plus court chemin de init à y a pour coût :

coût chemin (y) = MIN (coût chemin (xi) + coût (xi , y))

L'idée est d'initialiser les coûts des chemins à une valeur majorante, et de les diminuer progressivement en appliquant cette formule pour tout arc x→y, en fonction de la valeur précédente associée à ses prédécesseurs.
Le mécanisme s'arrête lorsqu'aucune modification n'intervient.

Initialisation :

coût chemin (init) = 0
pour tout autre sommet x : coût chemin (x) = infini

Itération jusqu'à stabilité :

pour tout sommet y :
 pour tout prédécesseur x de y :
  coût chemin (y) ← min (coût chemin (y) , coût chemin (x) + coût (x , y))

