Devoir d'Algorithmique - Maitrise d'Informatique

Décembre 1999

Nous considérons des graphes dont les arcs sont étiquetés par des poids entiers peuvant être positifs ou négatifs. Dans ce qui suit, nous entendons donc par graphe un triplet $G = (S, A, p)$ où S est un ensemble de sommets, A est un ensemble d'arcs, et $p : A \to \mathbb{Z}$ est une fonction associant à chaque arc un poids dans l'ensemble des entiers.

Nous voulons alors résoudre le problème suivant:

> *étant donné un graphe G défini comme ci-dessus et un sommet $s \in S$, calculer les poids minimums de chemins partant de s vers chacuns des autres sommets du graphe.*

Le poids d'un chemin est défini comme étant la somme des poids des arcs le constituant. Ce poids peut être positif ou négatif. Pour tout sommet u, nous notons $\delta(s, u)$ le poids minimum d'un chemin allant de s à u.

Question 1: Considérer le graphe de la figure 1. Quel est un chemin de poids minimum entre les sommets a et f, a et e ?

Nous considérons que, pour tout sommet u tel qu'il existe un circuit de poids négatif sur un chemin allant de s à u, nous avons $\delta(s, u) = -\infty$.

Question 2: Donner un exemple simple de graphe (avec des arcs ayant des poids négatifs) pour lequel l'algorithme de Dijkstra produit une réponse incorrecte. Dire pourquoi la preuve de l'algorithme de Dijkstra ne peut pas être

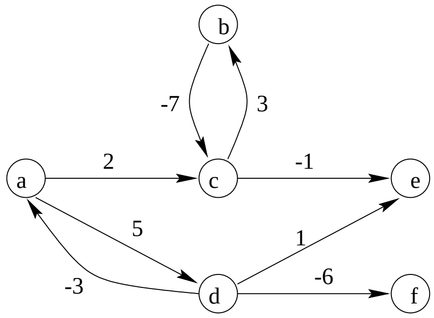

Figure 1:

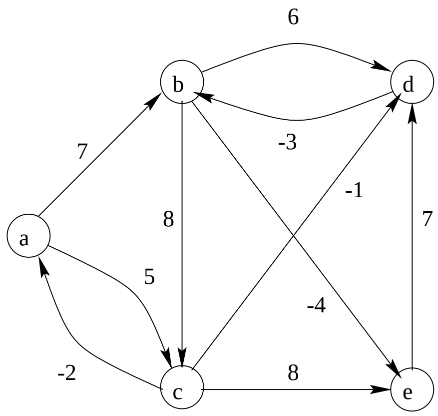

Figure 2:

appliquée dans le cas où l'on permet des arcs de poids négatifs.

Considérons l'algorithme suivant:

$Initialiser(s)$

pour $i = 1$ **à** $|S| - 1$ **faire**

  pour tout $(u, v) \in A$ **faire** $Relaxer(u, v)$

pour tout $(u, v) \in A$ **faire**

  si $d[v] > d[u] + p(u, v)$ **alors** **retourner**$(faux)$

retourner$(vrai)$

Avec:

$Initialiser(s) =$

$d[s] := 0$

pour tout $u \in S \setminus \{s\}$ **faire** $d[u] := \infty$

$Relaxer(u, v) =$

si $d[v] > d[u] + p(u, v)$ **alors** $d[v] := d[u] + p(u, v)$

Notre but est de montrer la correction de cet algorithme: si le graphe ne contient pas de circuit négatif, alors l'algorithme répond "*vrai*" et dans ce cas nous avons $\forall u \in S$, $d[u] = \delta(s, u)$, sinon (si le graphe contient un circuit de poids négatif), l'algorithme répond "*faux*".

Question 3: Appliquer l'algorithme ci-dessus au graphe donné en figure 2.

Nous rappelons les faits suivants (vus en cours) concernant les suites d'actions $Relaxer$:

Lemme 0.1 *La propriété* $\forall u \in S,\ d[u] \geq \delta(s,u)$ *est invariante pour toute séquence de relaxations. De plus, si* $d[u] = \delta(u,v)$ *devient vraie, alors elle le reste après toute autre séquence de relaxations.*

Lemme 0.2 *Soit* $s \rightsquigarrow u \to v$ *un chemin de poids minimum de* s *vers* u *et* v*. Si au cours de l'execution d'une séquence d'actions de relaxation nous avons* $d[u] = \delta(s,u)$*, alors nous aurons* $d[v] = \delta(u,v)$ *après l'execution de* $Relaxer(u,v)$*.*

Question 4: Supposons que G soit sans circuit de poids négatif accessible à partir de s. Soit u un sommet *accessible* à partir de s et $c = u_0, u_1, \ldots, u_n$ un chemin de poids minimum entre s et u, où $s = u_0$ et $u_n = u$. Montrer que pour tout $i \in \{0, \ldots, n\}$, l'égalité $d[u_i] = \delta(s, u_i)$ est vraie après la i^{eme} étape (passage sur les arcs du graphe) de l'algorithme. Que peut-on en déduire comme propriété sur $d[u]$ à la terminaison de l'algorithme ? Est-ce qu'il est possible que l'on ait $d[u] = \infty$?

Que dire de $d[v]$ à la terminaison de l'algorithme si v n'est pas accessible à partir de s dans G ?

Question 5: Montrer que si G n'a pas de circuit de poids négatif accessible à partir de s, l'algorithme répond "*vrai*".

Supposons maintenant que G ait un circuit de poids négatif accessible à partir de s. Soit $c = u_0, u_1, \ldots, u_n$ un tel circuit. Noter que $u_0 = u_n$ (c est un circuit), et que $\sum_{i=1}^{n} p(u_{i-1}, u_i) < 0$ (c est de poids négatif). Nous devons montrer que, dans ce cas, l'algorithme répond "*faux*".

Question 6: Supposons que l'algorithme répond "*vrai*". Peut-on en déduire une relation entre $d[u_i]$ et $d[u_{i-1}]$ pour tout $i \in \{1, \ldots, n\}$? En déduire une contradiction avec le fait que le poids du circuit c est négatif.

Question 7: Quelle est la complexité en temps de cet algorithme ?