Aucun document autorisé.

1 listes d'adjacences : ajout d'un arc

Un graphe orienté $G(S, A)$ est donné par liste d'adjacences (tableaux Tete[0..n] et Succ[1..m], avec Tete[0]=0 tels que les successeurs de i sont rangés dans Succ[Tete[i-1]+1..Tete[i]]). Donnez le pseudo-code d'une procédure `Add(Tete[],Succ[],n,u,v)` qui ajoute l'arc `(u,v)` au graphe. (On supposera qu'il y a la place nécessaire dans le tableau, i.e. que $MAX < m$). On supposera que les sommets u et v sont déjà dans le graphe, mais que l'arc (u, v) n'y est pas. Peut-on effectuer cet ajout en temps $O(n)$?

2 Les composantes connexes ont au moins, au plus un sommet noir

G est un graphe non-orienté. Certains sommets sont noirs, les autres sont blancs. Un tableau `Noir[sommet] de booleens` indique pour chaque sommet sa couleur.

Ecrire le pseudo-code d'un algorithme qui rend vrai ssi toute composante connexe contient au MOINS un sommet noir.

Ecrire le pseudo-code d'un algorithme qui rend vrai ssi toute composante connexe contient exactement un sommet noir.

3

Soit G un graphe non orienté pondéré. Soit v le poids minimal des arêtes. Soit T un arbre couvrant minimal.

Montrez que toutes les arêtes de T sont de poids v ssi le graphe contenant les sommets de G mais seulement l'ensemble des arêtes de poids v; est connexe.

Une partie de cette preuve est très facile. L'autre peut se faire soit en utilisant les propriétés des algorithmes vus en cours, soit sans les utiliser. Faire si possible les deux preuves.

4 Mise a jour des distances

G est un graphe orienté valué par des poids positifs, s est un sommet de G. On a éterminé les poids de s à tout sommet.

Soit x et y deux sommets, soit w le poids de l'arête de x à y (éventuellement, l'arc n'existe pas et donc $w = \infty$), soit $\overline{w}$ un réel positif inférieur à w. On construit le graphe $\overline{G}$ en recopiant G, puis en remplaçant le poids w de l'arc de x à y, par $\overline{w}$.

(a) Donner un test rapide qui dit si les distances depuis s en sont modifiées.

(b)Donner un algorithme efficace calculant $\overline{Dist}$ table des distance de s à tout sommet à partir de x, y, $\overline{w}$, G, $\overline{G}$, $Dist$

On rappelle l'algo de Dijkstra :

```
pour tout sommet t, d[t] <- infini
d[s] <- 0
E <- S

faire n fois :
   soit y dans E tq d[y] minimise { d[w] | w dans E }
   E <- E - { y }
   pour tout successeur z de y faire
      si d[z] > d[y] + w(y,z)
      alors   d[z] <- d[y] + w(y,z)
```

On a un sommet s dans le graphe valué G. Pour tout sommet x, on note $d[x]$ la distance de s à x.

(1) G est non orienté et on a une arête de poids nul entre deux sommets x et y. A-t-on $d[x] = d[y]$?

(2) G est orienté et on a un arc de poids nul du sommet x vers le sommet y. A-t-on $d[x] = d[y]$?

Justifiez.

6 Qui est qui ?

N personnages sont dessinés, on a N prénoms. On donne des caractéristiques en fonction des prénoms (Jean est blond, il a une moustache et une jambe de bois) et il faut attribuer un prénom à chacun des personnages.

Exemple : Le personnage 1 est blond sans moustache, le personnage 2 est blond avec moustache, le personnage 3 est brun avec moustache. Exemple 1 : Bernard et Claude sont bruns (il n'y a pas de solution). Exemple 2 : Alfred est brun et Claude a une moustache (une unique solution). Exemple 3 : Alfred a une moustache et Bernard est blond (trois solutions)

Existe-t-il un algo polynomial qui permet de savoir s'il y a une solution ? Et si elle est unique ?

7 Flots avec coûts

G est un graphe orienté avec deux sommets s et t, et deux valuations $Capacite$ et $CoutParUnite$ (positives ou nulles) des arêtes Le coût d'un flot est la somme sur toutes les arêtes de la valeur du flot passant par cette arête multiplié par le coût par unité de cette arête. k est inférieur ou égale à la valeur du flot maximal de G. On souhaite trouver un flot de valeur k de coût minimal (parmi les flots de valeur k).

(a) Soit $G = (\{s, t, u, v\}, \{(s, u), (s, v), (u, v), (u, t), (v, t)\})$. Tous les arcs sont de capacité 2 sauf (u, v) qui est de capacité 1. Les arcs $(s, u), (u, v), (v, t)$ sont de coût 1, (s, v) est de coût 3, (u, t) est de coût 4. Donnez les cinq flots de valeurs 0,1,2,3 et 4 de coût mininal.

(b) Comment adapter Ford-Fulkerson pour trouver un flot de valeur k et de coût minimal ?

8 On considère le graphe ci-dessous. Dans le dessin de gauche, les nombres désignent les capacités. Dans le dessin de droite, les nombres représentent les valeurs d'un flot ϕ. Le flot ϕ est-il maximal ? Si non, déterminer le flot maximal en utilisant un algorithme du cours, en partant du flot ϕ. Donner les graphes d'écart et les chemins améliorants. Dessiner toutes les coupes minimales.