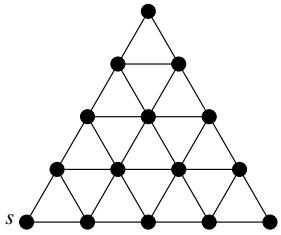
- Examen -

- Durée : 2 heure - Aucun document n'est autorisé -
Janvier 2016

Le barême est indicatif. Les exercices peuvent être traités dans un ordre quelconque.
Toutes les réponses doivent être claires et justifiées.

- Exercice 1 - Parcours - *(4 points)*

La grille triangulaire T_5 est le graphe dessiné ci-dessous. On souhaite effectuer des parcours de cette grille.



En cas de choix sur un possible successeur d'un sommet x au cours du parcours on choisira le premier sommet libre dans le sens direct (anti-horaire) en partant de $pere(x)$. Pour partir de la racine s on choisira en premier le voisin de s situé à sa droite.

a. Effectuer un parcours en profondeur sur T_5 issu de s en respectant la règle ci-dessus. On ne représentera que l'arbre de parcours obtenu (sur le graphe original) en indiquant l'ordre de découverte.

b. Effectuer un parcours en largeur sur T_5 issu de s en respectant la règle ci-dessus. On ne représentera que l'arbre de parcours obtenu (sur le graphe original) en indiquant l'ordre de découverte.

c. Pour n quelconque, calculer le niveau maximal obtenu lors d'un parcours en largeur de T_n, calculé comme précédemment.

d. Pour n quelconque, calculer le niveau maximal obtenu lors d'un parcours en profondeur de T_n, calculé comme précédemment.

- Exercice 2 - Plus longs chemins dans les graphes orientés acycliques - *(9 points)*

a. On considère un graphe orienté D acyclique (c'est-à-dire sans cycle orienté). Rappeler la définition d'un tri-topologique de D.

b. Soit D_1 le graphe orienté de sommets $\{a, b, c, d, e, f, g, h\}$ et d'arcs $\{ab, ae, be, bd, ca, cd, ed, fa, gc, gh, he\}$. Calculer un tri-topologique de D_1 en expliquant la méthode utilisée.

c. On veut calculer la longueur d'un plus long chemin orienté dans graphe orienté acyclique. Pour cela on considère l'algorithme PLC détaillé ci-dessous.

Faire tourner l'algorithme PLC sur le graphe D_1 de la question précédente.

d. On a vu en TD un algorithme permettant de calculer un tri topologique d'un graphe acyclique D en temps $O(n + m)$ où n et m désignent respectivement le nombre de sommets et d'arcs de D. En supposant que cet algorithme est utilisé à la ligne 1 de l'algorithme PLC, calculer, en justifiant, la complexité de PLC.

e. Prouver la validité de l'algorithme PLC (on pourra montrer par récurrence sur i que après l'étape i de la boucle ligne 2, $l(v_i)$ contient la longueur d'un plus long chemin de D terminant en v_i).

Algorithme : PLC

Données : Un graphe orienté acyclique D sur n sommets codé par listes de voisins entrants.
Résultat : La longueur d'un plus long chemin orienté de D.

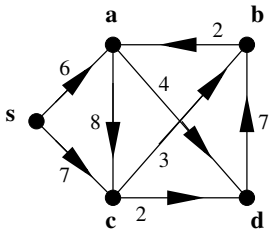
```
1 Calculer v_1, ..., v_n un tri topologique de D;
2 pour i de 1 à n faire
3 |   l(v_i) ← 0;
4 |   pour v_j voisin entrant de v_i (avec j < i donc) faire
5 |   |   si l(v_j) + 1 > l(v_i) alors
6 |   |   |   l(v_i) ← l(v_j) + 1;
7 retourner max{l(v_i) : i = 1, ..., n};
```

f. Modifier l'algorithme PLC pour faire afficher un plus long chemin orienté du graphe acyclique donné en entrée.

g. *Modélisation : Matriochka !* Une *boîte* $B(l, L, h)$ est un parallélépipède rectangle dont les dimensions sont l,L et h, avec par convention $l \leq L \leq h$. La boîte $B(l_1, L_1, h_1)$ *s'emboîte* dans la boîte $B(l_2, L_2, h_2)$ si $l_1 < l_2$, $L_1 < L_2$ et $h_1 < h_2$. Étant donnée un ensemble de n boîtes, donner un moyen algorithmique de calculer la plus longue séquence de boîtes pouvant s'emboîter.

- Exercice 3 - Plus courts chemins - *(7 points)*

On considère le graphe orienté et pondéré ci-dessous :



a. *Question de cours :* rappeler l'algorithme de Dijkstra : ce qu'il prend en entrée, produit en sortie, son pseudo-code et sa complexité.

b. Utiliser cet algorithme sur le graphe ci-dessus afin de déterminer une arborescence des plus courts chemins issue de s. Indiquer brièvement les étapes de calcul.

c. La longueur de l'arc cb augmente pour passer à 10 unités de longueur. En expliquant, recalculer une arborescence des plus courts chemins issue de s dans ce nouveau graphe sans relancer l'algorithme de Dijsktra.

d. Plus généralement, écrire un algorithme qui :
- prend en entrée un graphe orienté D codé par listes de voisins entrants et valué positivement sur les arcs par une fonction l, un sommet s de D, une arborescence T des plus courts chemins de D issue de s et la fonction de distance d associée, y une feuille de T de père x et un réel positif p.
- retourne une arborescence des plus courts chemins issue de s dans le graphe D où la longueur de l'arc xy a augmenté de p.
- s'exécute en temps $O(n)$ (donc ne relance pas l'algorithme de Dijsktra...).

e. Prouver la validité de l'algorithme précédent.