Algorithmique - ROB3
TD8 - Plus courts chemins

Exercice 1 (Arbre couvrant de poids minimum et plus courtes chaînes)

On considère un graphe connexe non-orienté $G = (V, E)$, où chaque arête e est valuée par un poids $w_e \geq 0$. On suppose qu'on a déterminé un arbre couvrant T de poids minimum et qu'on a également déterminé les plus courtes chaînes d'un sommet particulier s à tous les autres sommets. On suppose maintenant que le poids de chaque arête est incrémenté de 1 : les nouveaux poids sont donc $w'_e = w_e + 1$.

Question 1.1 *L'arbre couvrant T reste-t-il un arbre couvrant de poids minimum ? Prouver que c'est vrai ou donner un exemple où ce n'est pas le cas.*

Question 1.2 *Les plus courtes chaînes le restent-elles ? Prouver que c'est vrai ou donner un exemple où ce n'est pas le cas.*

Exercice 2 (Calcul du nombre de chemins)

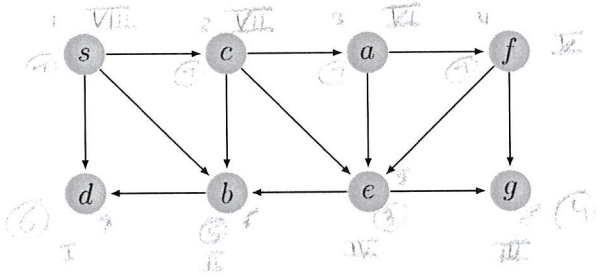
On considère un graphe $G = (V, E)$ orienté sans circuit, de racine s. L'objet de cet exercice est de concevoir une procédure de programmation dynamique qui, pour chaque sommet $v \in V$, calcule le nombre de chemins *de s à v dans G.*

Question 2.1 *Soit $nb(v)$ le nombre de chemins de s à v dans G. A la manière de la relation de récurrence à l'origine de l'algorithme de Bellman, donner la relation de récurrence permettant de déterminer $nb(v)$.*

Question 2.2 *Comment initialiser la récurrence ?*

Question 2.3 *Ecrire l'algorithme "du bas vers le haut" mettant en œuvre cette récurrence. Analyser sa complexité. On prendra soin de préciser la structure de données utilisée pour obtenir cette complexité.*

Question 2.4 *Appliquer cet algorithme pour déterminer les différentes valeurs $nb(v)$ ($v \in \{s, a, b, c, d, e, f, g\}$) dans le graphe ci-dessous. On prendra soin d'indiquer l'ordre sur les sommets qui a permis de déterminer les valeurs $nb(v)$.*



Exercice 3 (Déplacements d'un robot)

On considère un graphe $G = (V, E)$ non-orienté, avec $V = \{0, \ldots, n-1\}$ et $|E| = m$. Un robot est positionné initialement (au pas de temps 0) sur le sommet 0. Puis, entre chaque pas de temps k et $k+1$ ($k \in \{0, \ldots, T-1\}$), il se déplace aléatoirement dans G en suivant la règle suivante : depuis le sommet $v \in V$ sur lequel il est positionné au pas de temps k, il se déplace vers un voisin v' de v, selon une loi de probabilité uniforme (il sera donc positionné sur le sommet v' au pas de temps $k+1$). Par exemple, sur le graphe de la figure 1, si le robot est positionné au sommet 4 au pas de temps k, la probabilité que le robot soit positionné au sommet 1 (resp. au sommet 3) au pas de temps $k+1$ est 1/2 (resp. 1/2). Précisons que rien n'empêche le robot de se replacer sur un sommet déjà visité (le robot se déplace toutefois toujours entre chaque pas de temps).

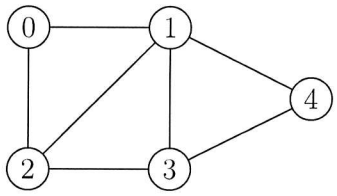


FIGURE 1 – Un exemple de graphe non-orienté.

L'objet de cet exercice est de concevoir une procédure de programmation dynamique dont le but final est, pour chaque sommet $v \in V$, de calculer la probabilité que le robot soit positionné en v au pas de temps T.

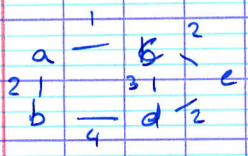
Question 3.1 *Soit $p_k(v)$ la probabilité que le robot soit positionné au sommet v au pas de temps k. A la manière de la relation de récurrence à l'origine de l'algorithme de Bellman-Ford, donner la relation de récurrence permettant de déterminer $p_k(v)$ connaissant les probabilités $p_{k-1}(u)$ ($u \in \{0, \ldots, n-1\}$), en notant $d(u)$ le degré d'un sommet $u \in V$. Comment initialiser la récurrence ?*

Question 3.2 *Analyser la complexité de l'algorithme de programmation dynamique qui met en œuvre cette récurrence (on ne demande pas d'écrire l'algorithme). On prendra soin de préciser la structure de données utilisée pour obtenir cette complexité.*

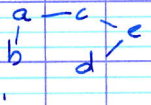
Question 3.3 *Appliquer cet algorithme au graphe de la figure 1 pour déterminer les différentes valeurs $p_k(v)$ ($v \in \{0, \ldots, 4\}$) pour $k \in \{0, 1, 2\}$.*

TD 8

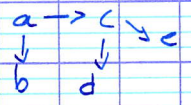
Exercice 1



Arbre couvrant de poid minimum



Arborescence des plus court chemin depuis a.



Remarque: Chaque chaîne dans un ACPT est de coût min et minimaxe (càd. minimise la valeur de la plus grande arête de la chaîne).

1/

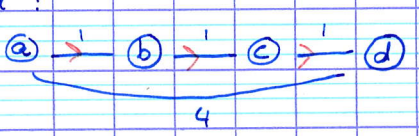
On a $w'(T) = \sum_{e \in T} w_e' = \sum_{e \in T} (w_e + 1) = w(T) + (n-1)$

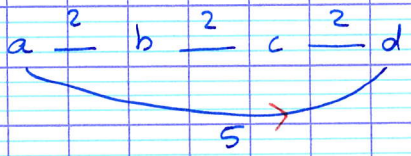
car T a $(n-1)$ arête.

Donc a arbre couvrant de poid minimum le reste.

2/

les plus courtes chaînes ne le restent pas nécessairement:





Exercice 2

1)

$nb(v) = \sum_{u \in Pred(v)} nb(u)$

2)

$nb(s) = 1$

3/

Soit $L = (v_1, v_2, \dots, v_n)$ une liste topologique des sommets du graphe.

$nb[s] = 1$

pour i allant de 1 à n.

$nb[v_i] = 0$.

~~pour~~ pour u dans $Pred(v_i)$ faire

$nb[v_i] = nb[u] + nb[v_i]$

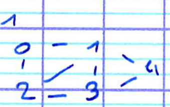
(marge :) $O(n+m)$ si le graphe est repré. par une liste de successeur.

(marge :) $O(1)$; $O(\deg(v_i))$; $O(n+m)$ si liste des prédécesseurs

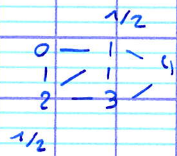
Exercice 3

3/

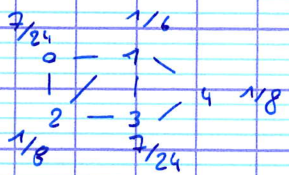
$K = 0$



$K = 1$



$K = 2$



1/

$$p_K(v) = \sum_{u \in Pred(v)} p_{K-1}(u) \times \frac{1}{\deg(u)}$$

$p_0(0) = 1 \quad p_0(v) = 0$ si $v \neq 0$.

2/

Chaque itération est en $O(n+m)$ si le graphe est représenté par des listes d'adjacences.

Donc $O(T(n+m))$ pour calculer les $p_T(v)$.