Final RO03 **Printemps 2016**

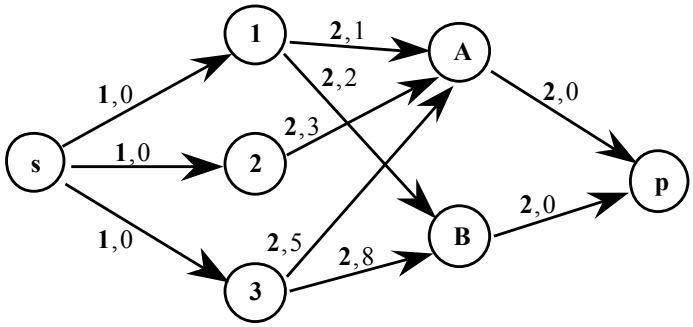
Le polycopié et les notes de cours et Td sont admis.

Rédigez vos réponses sur des copies différentes pour chaque partie.

Partie I

Exercice 1 : (3 points)

Faire tourner l'algorithme de ROY sur le graphe ci-dessous afin de déterminer un flot maximal à coût minimal. On rapportera les graphes d'écart et les chemins améliorants successifs, ainsi que le flot final et son coût.



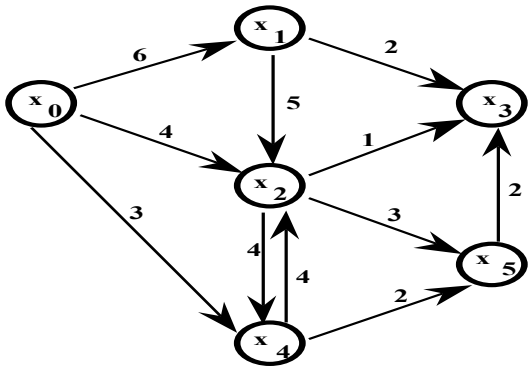
Notation (**capacité**, coût)

Exercice 2 (7 points)

Soit G=(X,U,c) un graphe dont les arcs sont valués par une valuation strictement positive représentant les capacités des arcs. On rappelle que la capacité d'un chemin est la plus **petite** valuation des arcs de ce chemin.

Question 1. Modifier l'algorithme de Dijkstra (rappelé ci-dessous) pour calculer les chemins de capacité **maximale** issus du sommet x0. Utiliser les mêmes notations que pour l'algorithme de Dijkstra.

Question 2. Appliquer l'algorithme sur le graph G et calculer les chemins de capacité maximale issus de x0. On rapportera le tableau analogue à celui de DIJKSTRA. Déterminer une arborescence de chemins de capacités maximales. Y-a-t-il unicité des chemins de capacités maximales ? Justifier.



Question 3. Proposer une méthode générale qui calcule le chemin le plus court parmi les chemins de capacité maximale et qui utilise l'algorithme de capacité maximale trouvé dans la Question 1.

Rappel : Algorithme de Dijkstra

(i) Poser S ={x_0}, $\lambda_0 = 0$, $\lambda_i = v_{0i}$ si (x_0 x_i) $\in$ U, et $\lambda_i = \infty$ sinon.
(ii) Tant que S $\neq$ X faire
(a) Choisir $x_i \in$ X - S de λ_i minimum.
(b) Poser S = S + { x_i }.
(c) Pour tout $x_j \in$(X - S), successeur de x_i, poser : $\lambda_j = \min(\lambda_i + v_{ij}, \lambda_j)$.

PARTIE II

Exercice 3. Le problème du Bin-Packing (10 points)

Dans ce problème, on a n articles u_1, u_2, …, u_n de taille $s(u_1)$, $s(u_2)$, …, $s(u_n)$ comprise entre 0 et 1. Il s'agit de mettre ces articles dans un nombre minimum de boites de capacité 1. Mathématiquement, il faut déterminer une partition U_1, U_2, U_k de l'ensemble des articles telle que :

- Les contraintes de capacités soient satisfaites
  $\sum_{u \in Uj} s(u) \leq 1$ (j=1, .., k)
- Le nombre k de boites soit minimum.

Etude de l'Algorithme FIRST FIT

{on suppose que l'on dispose d'un nombre infini de boites de capacité 1 ; ces boites sont numérotées par les entiers naturels : on met l'article i dans la boite admissible de plus petite indice.}

Algorithme FIRST FIT

DEBUT {Les articles sont indicés selon une liste quelconque}

Pour i=1 à n faire

Début

Déterminer le plus petit numéro de boite j pour lequel la place restante disponible est supérieure ou égale à $s(u_i)$;

Affecter l'article u_i à la boite j.

Fin.

FIN.

Question 1. (4 points)

1) Appliquer l'algorithme FIRST FIT à la donnée suivante : on a 10 articles de tailles : $s(u_1)$=0,5 ; $s(u_2)$=0,1 ; et ainsi de suite 0,3 ; 0,4 ; 0,7 ; 0,2 ; 0,6 ; 0,6 ; 0,1 et le dernier $s(u_{10})$=0,5.
2) Evaluer la complexité de l'algorithme en fonction de n. Toute réponse non justifiée sera considérée comme fausse.
3) Que proposez-vous pour améliorer le résultat de l'algorithme FIRST FIT ? Appliquer l'amélioration que vous proposez sur l'exemple ci-dessus.

Question 2. (4 points) On se place dans le cas général. D dénote une donnée quelconque de FIRST FIT. On veut démontrer que FF(D) < 2 F*(D), où FF(D) est le nombre de boites non vides utilisés par FIRST FIT, et F*(D) le nombre optimal de boites.

1) Démontrez qu'au cours de l'algorithme le nombre de boites non vides, dont le contenu est inférieur où égal à 0,5, vaut au plus 1.

Dans la suite, on note *a* le contenu de la boite la moins chargée.

2) Cas $a > 1/2$
   Justifier $FF(D)/2 < \sum_{i=1}^{n} s(u_i)$ et $F^*(D) \geq \sum_{i=1}^{n} s(u_i)$, et en déduire le résultat.
3) Cas $a \leq 1/2$
   Utiliser les résultats démontrés en 1) et 2) pour démontrer le résultat.

Question 3. Proposer un model PL pour ce problème. (2 points)

Vous pouvez utiliser les notations suivantes :

n – nombre d'articles ;

$s(u_i)$: taille de l'article i, $1 \leq i \leq n$.

x_{ij}= 1 si article i est placé dans la boite j et 0 sinon.

y_j = 1 si la boite j est utilisé et 0 sinon.