Faculté des Sciences
Aix*Marseille Université

Année universitaire 2020/2021

Site : ☒ Luminy ☐ St-Charles ☐ St-Jérôme ☐ Cht-Gombert ☐ Aix-Montperrin ☐ Aubagne-SATIS
Sujet de : ☒ 1er semestre ☐ 2ème semestre ☐ Session 2 Durée de l'épreuve : 1h30
Examen de : L3 Nom du diplôme : Licence d'Informatique
Code du module : SIN5U03 Libellé du module : Algorithmique 2
Calculatrices autorisées : NON Documents autorisés : OUI, notes de cours/TD/TP

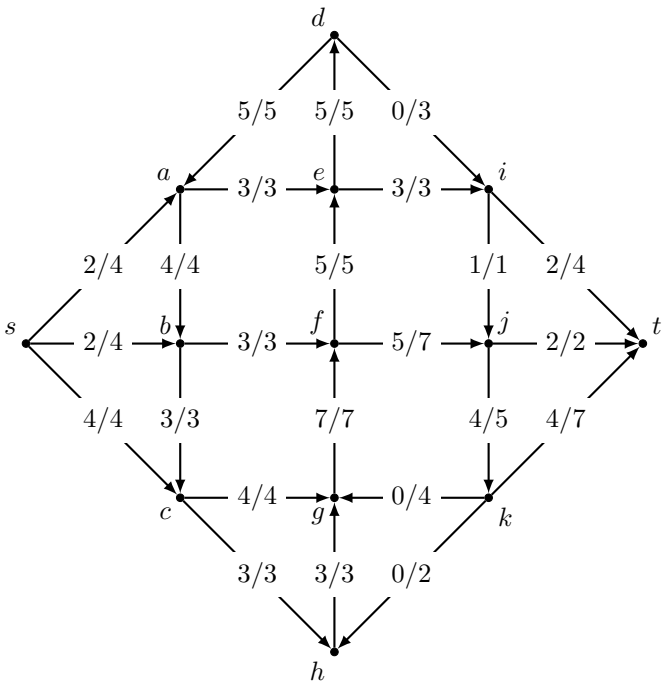
Ce sujet contient quatre exercices indépendants sur 14 points (12 points plus 2 points de bonus). Les exercices peuvent donc être traités dans n'importe quel ordre.

1 Calcul d'un flot maximum *(3 points)*

Question 1. A partir du flot représenté sur la figure ci-dessous, où x/y représente un flot de valeur x sur un arc de capacité y, exécuter autant d'itérations de l'algorithme de Ford-Fulkerson que nécessaire pour obtenir un flot maximum. Pour chacune des itérations, vous indiquerez la chaîne augmentante que vous avez utilisée et la valeur de l'augmentation du flot.

Question 2. Identifier la liste des sommets atteignables par une chaîne augmentante par rapport au flot maximum et donc une st-coupe de capacité minimum. Vérifier que la capacité de cette st-coupe est égale à la valeur du flot maximum.

Question 3. Est-ce que la valeur du flot change si la capacité de l'arc (k, t) passe de 7 à 5 ? Est-ce qu'elle change si la capacité de cet arc passe de 7 à 4 ? Dans chacun des cas, donner une justification.



2 Le plus grand sous-cube monochromatique *(3 points)*

La nouvelle tablette graphique 3d *Wacom* est un cube $n \times n \times n$ constitué de n^3 cubes unitaires $1 \times 1 \times 1$, appelés *voxels*. Chaque voxel est défini par les coordonnées (i, j, k) de son centre. Aussi un voxel peut avoir une couleur (représentée par un entier). Une image sur la tablette est une coloration de chaque voxel.

L'entrée du problème est donc une image, définie par un tableau M de dimension $n \times n \times n$ dont les entrées $M[i, j, k]$ sont des entiers représentant les couleurs des voxels. Un *sous-cube monochromatique de côté* ℓ est donné par un triplet (i, j, k) tel que $i \leq n - \ell, j \leq n - \ell$ et $k \leq n - \ell$ tels que tous les $M[a, b, c]$ sont égaux pour $i \leq a \leq i+\ell, j \leq b \leq j+\ell$ et $k \leq c \leq k+\ell$. Le but de cet exercice est d'utiliser la programmation dynamique pour trouver le plus grand sous-cube monochromatique.

Question 1. Formuler les sous-problèmes. Quel est le nombre de sous-problèmes ?

Question 2. Donner une formule récursive pour résoudre les sous-problèmes. Justifier cette formule.

Question 3. Ecrire le pseudo-code de l'algorithme et sa complexité.

3 Diviser-pour-régner *(4 points)*

(a) Trouver les complexités asymptotiques induites par les relations de récurrence suivantes :

$$T(n) = 2T\left(\tfrac{n}{2}\right) + \sqrt{n} \quad ; \quad T(n) = 4T\left(\tfrac{n}{2}\right) + O(n^3) \quad ; \quad T(n) = T\left(\tfrac{n}{2}\right) + O(1)$$

(b) Soit T un tableau trié d'entiers deux à deux distincts. On veut déterminer s'il existe un entier i tel que $T[i] = i$. Proposer et justifier un algorithme en $O(\log n)$ pour résoudre ce problème.

4 Bin Packing avec des objets de tailles $0,3$ et $0,4$ *(4 points)*

Rappelons que dans le problème de *Bin Packing* (vu en TD), on se donne un ensemble de N objets de tailles $s_1, s_2, \ldots, s_N$ où $0 < s_i < 1$ pour tout $i = 1, 2, \ldots, N$. Le but est de ranger les objets dans un nombre minimum de boîtes de taille 1. On connaît déjà les algorithmes suivants :

NextFit On prend les objets un après l'autre et on place l'objet courant dans la dernière boîte si on peut, sinon l'objet est placé dans une nouvelle boîte.

FirstFit On prend les objets un après l'autre et on place l'objet courant dans la première boîte qui peut l'accueillir, sinon on place l'objet dans une nouvelle boîte.

FirstFitDecreasing (FFD) Trier les objets dans l'ordre décroissant des tailles et sur la liste résultante exécuter FirstFit.

Dans cet exercice, nous supposons que chaque objet est de taille $0,3$ ou $0,4$. Notamment, supposons qu'il y a n objets de taille $0,3$ et m objets de taille $0,4$.

Question 1. Trouver un exemple où NEXT FIT se trompe (ne retourne pas un remplissage optimal).

Question 2. Quel est le nombre de boîtes utilisées par FFD en fonction de n et m ? Trouver un exemple où FFD se trompe.

Question 3. Formuler un principe de remplissage qui vous semble adapté pour les instances de Bin Packing avec des objets de tailles $0,3$ et $0,4$. Appelons votre algorithme ALG.

Montrer que ALG est toujours meilleur ou aussi bien que FFD. Pour cela trouver le nombre de boites utilisées par ALG dans les deux cas suivants : (a) $n \leq 2m$ et (b) $n \geq 2m$. Comparer dans chaque cas le nombre de boîtes de ALG avec celui de FFD.

Indication : Pour simplifier les choses et ne pas avoir affaire aux parties entières ($\lfloor x \rfloor$ et $\lceil x \rceil$), supposons que n et m sont divisibles par 2, 3, et 4.