L3 Informatique **Année 2015-2016**

Méthodes Algorithmiques

Examen du 22 Avril 2016
Responsable : Sophie Pinchinat

Documents non autorisés
Si les algorithmes ne sont pas **impeccablement** *écrits, ils risquent de ne pas être corrigés.*

Exercice 1 Un rabat-joie est chargé de partitionner $2n$ ($n \in \mathbb{N}$) joueurs en deux équipes A et B de n joueurs chacune. Chaque joueur est doté d'une notation qui mesure à quel point ce joueur est fort au jeu. Le rabat-joie cherche à diviser l'ensemble des joueurs de la manière la **moins équitable** possible, de manière à créer un déséquilibre le plus grand possible entre les équipes A et B. Montrer que le rabat-joie peut faire ce travail en temps $O(n \log n)$.

Exercice 2 Soit une tige d'origine de longueur n (unités de longueur) qui peut être découpée en tiges de longueurs entières. Pour chaque tige de longueur $1 \leq i \leq n$, on connaît son prix de vente p_i.

Exemple 1.

longueur	1	2	3	4	5	6	7
prix de vente	1	5	8	9	19	17	17

On cherche à décrire un algorithme qui calcule la meilleure manière de découper une tige de longueur n du point de vue la somme des prix de vente de toutes les tiges obtenues, ce qu'on apppellera dans la suite le *bénéfice*, noté $b(n)$.

1) Décrire toutes les manières possibles de découper une tige de longueur $n = 1, 2, 3$. Pour chacune de ces découpages, indiquer le bénéfice $b(n)$ obtenu en vous référant aux valeurs de l'Exemple 1.

2) De manière générale, quel est l'ordre de grandeur du nombre de manières possibles de découper une tige de longueur n ? Justifier.

On peut constater que si le découpage optimal contient k tiges de longueurs respectives $i_1, i_2, \ldots, i_k$ (donc $n = i_1 + i_2 + \cdots + i_k$), alors $b(n) = p_{i_1} + p_{i_2} + \cdots + p_{i_k}$. De plus, cette solution optimale qui consiste à effectuer le premier découpage de la tige d'origine à la longueur i_1 induit un découpage optimal du reste de la tige de longueur $n - i_1$; c'est une application du principe d'optimalité. Le découpage optimal s'obtient alors comme le meilleur entre celui où l'on ne découpe rien, et ceux où la tige d'origine est coupée à gauche d'abord en une tige de longueur i (que l'on ne découpera plus), et un découpage optimal de la tige restante de longueur $n - i$.

3) Utiliser ce principe d'optimalité pour écrire les équations qui caractérisent $b(n)$.

4) En déduire un algorithme de Programmation Dynamique pour calculer $b(n)$.

5) Quelle est la complexité de votre algorithme ?

Exercice 3 On suppose donnée un tableau $T[1..n]$ trié de n entiers naturels distincts. Décrire un algorithme de complexité $O(\log n)$ en temps (à justifier) qui retourne un indice i tel que $T[i] = i$ s'il existe. Par exemple, pour le tableau $[-10, -3, 3, 5, 7]$, $T[3] = 3$, en revanche pour le tableau $[2, 3, 4, 5, 6, 7]$, il n'y a pas de tel i.

Exercice 4 Une société de réseautage (*networking company*) utilise une techique de compression pour coder les messages avant de les transmettre à travers le réseau. Supposons que le message à transmettre contient les caractères suivants avec leur fréquence :

Caractère	a	b	c	d	e	f
Fréquence	5	9	12	13	34	45

Si la technique de compression utilisée est le codage de Huffman, expliquer en détail combien de bits[1] seront économisés dans le message envoyé.

[1] on suppose que chaque caractère est originellement codé sur un octet, soit 8 bits.