Série 6

1 Trouver deux nombres identiques consécutifs dans une liste

Soit $L = (L(1), L(2), \dots, , L(n))$ une liste de n nombres entiers qui satisfait la propriété suivante:

$$\text{pour tout } 1 < i < n, \quad \frac{L(i-1) + L(i+1)}{2} \geq L(i) \tag{1}$$

Un exemple de liste L satisfaisant cette propriété est la suivante (ce que vous êtes bienvenus de vérifier par vous-mêmes!): soit $1 < k < n$ fixé; on pose

$$L(i) = i\,(i-k), \quad 1 \leq i \leq n$$

On cherche un algorithme qui prenne en entrée la liste L et sa taille n, et dont la sortie soit oui si et seulement s'il existe $1 \leq i \leq n-1$ tel que $L(i+1) = L(i)$.

a) Essayez tout d'abord de répondre à la question pour l'exemple ci-dessus sans écrire formellement d'algorithme.

b) Ecrivez maintenant un algorithme qui réponde à la question demandée.

c) Quel est la complexité temporelle de votre algorithme? (utiliser la notation de Landau $\mathcal{O}(\cdot)$)

d) Ecrivez un algorithme dont la sortie soit identique à l'algorithme de la partie **b)** et dont la complexité temporelle soit $\mathcal{O}(\log_2(n))$.

Indication: Il est utile pour ça de puiser l'inspiration dans l'algorithme de dichotomie vu au cours.

2 Remplir son sac à dos

On considère tout d'abord le problème suivant: Soient L une liste de n nombres entiers positifs, représentant les poids de n objets, et C un autre nombre entier positif C, représentant la capacité maximale d'un sac à dos[1]. On suppose de plus que $L(i) < C$ pour tout $i \in \{1, \dots, n\}$ et que $\sum_{i=1}^{n} L(i) > C$. Existe-t-il un sous-ensemble $S \subset \{1, \dots, n\}$ tel que

$$\sum_{i \in S} L(i) = C \ ?$$

a) Le problème ci-dessus fait-il partie de la classe P, de la classe NP?

Toujours sous les mêmes hypothèses, on considère maintenant une version légèrement modifiée du problème ci-dessus: trouver un sous-ensemble $S \subset \{1, \dots, n\}$ tel que

$$\sum_{i \in S} L(i) \quad \text{est maximum}$$

tout en respectant la contrainte $\sum_{i \in S} L(i) \leq C$.

b) Ce nouveau problème fait-il partie de la classe NP? Est-il NP-difficile?

c) Ecrivez un algorithme de complexité temporelle polynomiale en n permettant de sélectionner un sous-ensemble $S \subset \{1, \dots, n\}$ tel que $\frac{C}{2} \leq \sum_{i \in S} L(i) \leq C$ (garantissant ainsi de remplir au moins la moitié du sac).

[1]On ne tient aucunement compte ici de la forme des objets ou du sac.

3 Exercice de révision

Soit n un nombre entier positif et A un tableau de dimensions $2 \times n$ rempli de nombres entiers positifs tels que *dans chacune des 2 lignes du tableau, tous les nombres sont différents.* Par exemple, si $n = 7$, un tel tableau pourrait être:

$$A = \begin{pmatrix} 4 & 3 & 11 & 2 & 12 & 1 & 5 \\ 5 & 2 & 12 & 13 & 15 & 4 & 3 \end{pmatrix}$$

a) Ecrire un algorithme dont les entrées soient un tableau A de ce type et n le nombre de colonnes de A, et dont la sortie soit oui s'il existe un numéro de colonne $i_0 \in \{1, \ldots, n\}$ tel que

$$A(1, i_0) \geq A(1, i) \quad \text{et} \quad A(2, i_0) \geq A(2, i) \quad \text{pour tout } i \in \{1, \ldots, n\}$$

et non dans le cas contraire (dans l'exemple ci-dessus, la sortie doit donc être oui).

b) Quel est la complexité temporelle de votre algorithme? (utiliser la notation de Landau $\mathcal{O}(\cdot)$)

c) Ecrire un algorithme dont la sortie soit identique à l'algorithme de la partie **a)** et *dont la complexité temporelle soit $\mathcal{O}(n)$.* [seulement si votre algorithme en a) n'a pas déjà cette complexité temporelle]

4 Exercice de révision

On considère les deux algorithmes suivants:

machin
entrée : *nombre entier positif n* sortie : *???*
Si $n = 1$ **Sortir:** 1 **Sortir :** $2^{(\textbf{bidule}(n-1)+1)}$

bidule
entrée : *nombre entier positif n* sortie : *???*
Si $n = 1$ **Sortir:** 0 **Sortir :** $\log_2(\textbf{machin}(n-1))$

a) Quelle est la sortie de l'algorithme **bidule** lorsque $n = 4$ en entrée?

b) Quelle est la complexité temporelle de l'algorithme **bidule**? (utiliser la notation de Landau $\mathcal{O}(\cdot)$)

c) Réécrire l'algorithme **machin** de façon récursive, mais sans faire appel à **bidule**.

d) En déduire quelle est la sortie de **machin** pour une valeur quelconque de $n \geq 1$ en entrée.

5 Exercice de révision

Soient n, C deux nombres entiers positifs et L une liste de n nombres entiers positifs telle que $0 < L(i) < C$ pour tout $i \in \{1, \ldots, n\}$ et $\sum_{i=1}^{n} L(i) \geq C$.

a) Ecrire un algorithme qui permette de trouver *en temps polynomial en n* le plus petit nombre $k \in \{2, \ldots, n\}$ tel qu'il existe un sous-ensemble $S \subset \{1, \ldots, n\}$ contenant k éléments et satisfaisant la relation $\sum_{i \in S} L(i) \geq C$.

Indication: Pour résoudre cet exercice, vous avez le droit d'utiliser un algorithme de tri vu au cours.

b) Quel est la complexité temporelle de votre algorithme? (utiliser la notation de Landau $\mathcal{O}(\cdot)$)