HUGO3

Algorithmique

TD n°4 : programmation dynamique

Exercice 1 – Sous-séquence de plus grande somme

Soit T un tableau de nombres réels de taille n. On cherche à déterminer la suite de cellules consécutives de ce tableau dont la somme est maximale. Par exemple, si $T = \boxed{5 \mid 15 \mid -25 \mid 10 \mid -5 \mid 30 \mid 25}$, la plus grande sous-séquence est $\langle 10, -5, 30, 25 \rangle$ dont la somme est 60.

Q 1.1 Soit $c(i)$ une fonction qui renvoie la somme maximale de toutes les sous-séquences se terminant en $T[i]$. Déterminez les équations de récurrence permettant de relier les $c(i)$. Quel est le cas de base de cette récurrence ?

Q 1.2 Déduisez-en un algorithme pour déterminer la sous-séquence dont la somme est maximale.

Q 1.3 Indiquez la complexité de votre algorithme.

Exercice 2 – Les dominos

Un domino est un rectangle qui contient 2 lettres, par exemple $\boxed{\text{A} \mid \text{H}}$ (il ne peut pas être retourné). Une séquence de dominos est une chaîne si les lettres voisines coïncident sur chaque paire de dominos consécutifs. Par exemple : $\boxed{\text{A} \mid \text{H}}\ \boxed{\text{H} \mid \text{E}}\ \boxed{\text{E} \mid \text{S}}\ \boxed{\text{S} \mid \text{W}}$.

On considère qu'on dispose de n pièces de dominos $D_1 = \boxed{x_1 \mid y_1}, \ldots, D_n = \boxed{x_n \mid y_n}$. On souhaite trouver la plus longue sous-séquence de tous ces dominos qui forme une chaîne en développant un algorithme de type programmation dynamique.

Par exemple, si on a les pièces suivantes : $\boxed{\text{A} \mid \text{H}}\ \boxed{\text{A} \mid \text{Z}}\ \boxed{\text{K} \mid \text{A}}\ \boxed{\text{H} \mid \text{K}}\ \boxed{\text{Z} \mid \text{V}}$, alors une plus longue sous-séquence est : $\boxed{\text{A} \mid \text{H}}\ \boxed{\text{H} \mid \text{K}}$.

Q 2.1 Soit $c(i)$ une fonction de $\mathbb{N}$ dans $\mathbb{N}$ qui donne la longueur de la chaîne la plus longue qui est une sous-séquence de $D_1, D_2, \ldots, D_i$ et qui se termine par la pièce D_i. Donnez les équations de récurrence pour cette fonction sans oublier les cas de base.

Q 2.2 Écrivez un algorithme efficace itératif pour calculer c à partir d'un tableau D contenant les n dominos. On utilisera une fonction `compatible`(j, i) qui renvoie vrai si D_j est compatible avec D_i.

Q 2.3 En sachant calculer la fonction choisie c, comment trouver la longueur de la sous-chaîne la plus longue parmi $D_1, D_2, \ldots, D_i$?

Q 2.4 Analysez la complexité de cet algorithme.

Q 2.5 Appliquez l'algorithme sur les pièces : $\boxed{\text{A} \mid \text{H}}\ \boxed{\text{A} \mid \text{Z}}\ \boxed{\text{K} \mid \text{A}}\ \boxed{\text{H} \mid \text{K}}\ \boxed{\text{Z} \mid \text{V}}$.

Exercice 3 – Sac à dos avec répétitions

Vous souhaitez partir en randonnée et vous préparez votre sac à dos. Afin de ne pas trop vous fatiguer durant la randonnée, vous vous imposez de ne pas mettre plus de P_max kilos dans ce sac (on suppose que P_max est un nombre entier). Vous avez également estimé l'intérêt d'emporter chacun des objets x_i que vous pouvez mettre dans votre sac et, pour cela, vous leur avez affecté un score entier $v(x_i)$ (plus le score est élevé, plus il apparaît intéressant d'emporter l'objet). Vous pouvez emporter un nombre entier de chaque objet. Les objets x_i, leurs poids p_i et leurs scores $v(x_i)$ sont les suivants :

objet	poids	score
x_0	4	20
x_1	2	12
x_2	3	18
x_3	5	23

Q 3.1 Donnez la formule de récurrence permettant, pour un poids donné p, $p \leq P\_max$, d'obtenir la somme maximale $S(p)$ des scores des objets que l'on peut placer dans le sac de telle sorte que le poids du sac soit d'au plus p.

Q 3.2 Déduisez-en un algorithme pour déterminer le score maximum que peut contenir le sac à dos.

Q 3.3 Quelle est la complexité de l'algorithme ?

Q 3.4 Modifiez votre algorithme pour déterminer le contenu du sac à dos optimal.

Q 3.5 Quelle est la complexité de l'algorithme ?

Exercice 4 – Monter un escalier en sautant

On considère un escalier de m marches ($m \geq 0$) que l'on peut gravir à l'aide de différents sauts de α marches, $\alpha \in [\alpha_1, \alpha_2, \ldots, \alpha_p]$, $\alpha_i > 0$. On suppose que les α_i sont rangés par ordre croissant. Une façon de gravir m marches en exactement s sauts, $s \geq 0$, correspond alors à une suite $(\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_s})$. On veut calculer $N(s, m)$ le nombre de façons différentes de gravir m marches en exactement s sauts, c'est-à-dire le nombre de suites distinctes à s éléments de $[\alpha_1, \alpha_2, \ldots, \alpha_p]$ dont la somme des éléments vaut m.

Q 4.1 Donnez une formule de récurrence pour $N(i, j)$, $i = 1 \ldots, s$, $j = 1 \ldots m$.

Q 4.2 Étant donné $p = 3$, $\alpha_1 = 2$, $\alpha_2 = 3$, $\alpha_3 = 5$ et $m = 12$, remplissez le tableau $N(1...6, 1...12)$.

Q 4.3 Écrivez l'algorithme correspondant.

Q 4.4 À quoi correspond $n(m) = N(., m)$? Donnez n pour $m \leq 12$.

Q 4.5 Donnez une formule de récurrence pour le calcul de $n(j)$, $j = 1 \ldots m$.

Exercice 5 – Plus court chemin entre tout couple de sommets

Dans cet exercice, on souhaite concevoir un algorithme de programmation dynamique permettant de calculer les valeurs $d(i, j)$ des plus courts chemins de tous les couples de sommets (i, j) dans un graphe orienté. On suppose que les sommets sont numérotés de 1 à n. De plus, chaque arc est muni d'une longueur l_{ij}.

Q 5.1 Soit $d(i, j, k)$ la longueur d'un plus court chemin allant du sommet i au sommet j et dont tous les sommets intermédiaires appartiennent à l'ensemble $\{1, \ldots, k\}$. Pour tout couple (i, j), quelle est la valeur de $d(i, j, 0)$?

Q 5.2 Donnez l'équation de récurrence sur k caractérisant $d(i, j, k)$.

Q 5.3 Donnez la formule permettant de calculer $d(i, j)$ à partir de $d(i, j, k)$.

Q 5.4 Déduisez-en un algorithme par programmation dynamique pour calculer les longueurs des plus courts chemins pour tous les couples de sommets.

Q 5.5 Quelle est la complexité de votre algorithme ?

Exercice 6 – Plus longue sous-séquence commune

Une mesure de similarité entre deux séquences d'ADN est la longueur de leur plus longue sous-séquence commune (PLSC). Ainsi, étant donné deux séquences $X = (x_1, \ldots, x_n)$ et $Y = (y_1, \ldots y_m)$ (où les x_i et y_i appartiennent à l'alphabet $\{A, C, G, T\}$), le problème consiste à trouver une sous-séquence (suite extraite) commune à X et Y de longueur maximale.

Par exemple, la PLSC entre $X = (A, G, T, T, A, C, G)$ et $Y = (C, G, A, T, A, A, G)$ est (A, T, A, G) et sa longueur est 4.

Pour cela, on introduit la notion suivante : on définit le i-ème préfixe de X, pour $i = 0, 1, \ldots, n$, par $X_i = (x_1, x_2, \ldots, x_i)$. Par exemple, si $X = (A, B, C, B, D, A, B)$, alors $X_4 = (A, B, C, B)$ et X_0 représente la séquence vide.

Q 6.1 Pour résoudre ce problème par la programmation dynamique, on utilise le lemme suivant. Soit deux séquences $X = (x_1, \ldots, x_n)$ et $Y = (y_1, \ldots y_m)$, et soit $Z = (z_1, \ldots, z_k)$ une PLSC de X et Y.

1. si $x_n = y_m$ et $z_k = x_n = y_m$ alors Z_{k-1} est une PLSC de X_{n-1} et Y_{m-1}
2. si $x_n \neq y_m$ et $z_k \neq x_n$ alors Z est une PLSC de X_{n-1} et Y
3. si $x_n \neq y_m$ et $z_k \neq y_m$ alors Z est une PLSC de X et Y_{m-1}

Prouvez ce lemme.

Q 6.2 La caractérisation de ce lemme montre qu'une PLSC de deux séquences contient une PLSC pour des préfixes respectifs X' et Y' des deux séquences X et Y. Déduisez-en la formule récursive sur laquelle débouche la sous-séquence optimale.

Q 6.3 Appliquez cette définition récursive sur les séquences $X = (A, G, A, T, A, C, A, T, G)$ et $Y = (G, A, T, T, A, C, A)$.

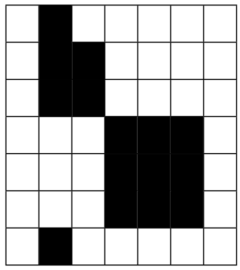
Q 6.4 Écrivez un algorithme de type programmation dynamique en temps $\mathbf{O}(nm)$ renvoyant la longueur de la PLSC.

Exercice 7 – Plus grand sous-carré mono-chromatique

Étant donné un tableau M de dimension $n \times m$ dont les entrées sont des couleurs (représentées par des entiers), un sous-carré monochromatique de côté k est donné par $i \leq n - k$ et $j \leq m - k$ tels que tous les $M(a, b)$ sont égaux pour $i \leq a < i + k$, $j \leq b < j + k$.

Q 7.1 Donnez la formule récursive sur laquelle débouche la sous-structure optimale.

Q 7.2 Appliquez la formule sur le motif suivant :



Q 7.3 Écrivez l'algorithme qui, étant donné M, renvoie un plus grand sous-carré monochromatique.

Q 7.4 Quelle est la complexité de l'algorithme ?