TD 8 : Algorithmes gloutons

Jeudi 22 novembre 2017

1 Emploi du temps optimal sans conflit

On cherche à résoudre le problème suivant : connaissant l'emploi du temps hebdomadaire d'une liste de n cours, on cherche à trouver le sous-ensemble de cours de taille maximale auxquels il est possible de s'inscrire sans qu'il y ait de conflit d'emploi du temps. On dispose d'un tableau D de taille n représentant l'horaire de début du chaque cours, et d'un tableau F de taille n représentant l'horaire de fin du chaque cours. Pour simplifier, on suppose que la date est représentée par un nombre réel positif, et que

$$\forall i \in \{1 \dots n\}, 0 \leq D[i] < F[i] \leq M.$$

De façon plus formelle, on cherche donc un sous-ensemble $G \subseteq \{1 \dots n\}$ tel que

$$\forall i, j \in G, i \neq j \implies F[i] < D[j] \text{ ou } F[j] < D[i].$$

Un exemple de problème d'emploi du temps est illustré à la Figure 1 et décrit par les tableaux D et F suivants :

n	1	2	3	4	5	6	7	8	9	10	11
D	2	7	16	1	6	14	3	12	0	9	20
F	5	13	19	4	11	18	10	21	8	15	22

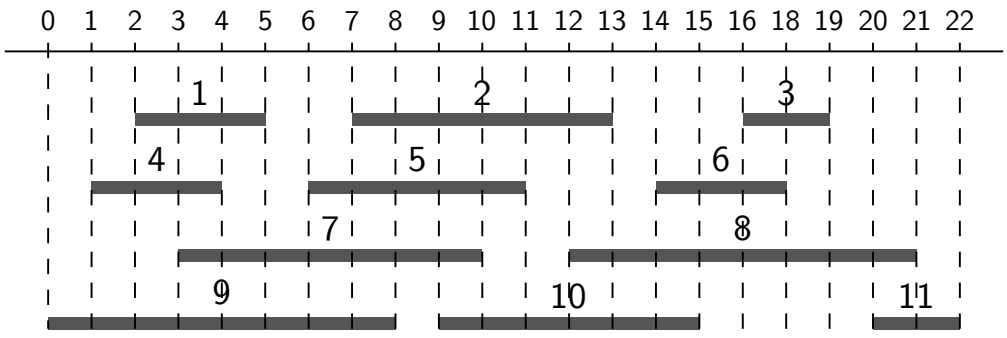

FIGURE 1 – Un problème d'emploi du temps à 11 cours.

QUESTION 1 – Écrire un algorithme récursif « naïf » qui calcule la taille maximale d'un emploi du temps. Donner sa complexité dans le pire cas.

L'objectif des questions suivantes est de trouver une stratégie plus efficace que cette approche récursive pour résoudre le problème de l'emploi du temps.

QUESTION 2 – Montrer qu'il existe au moins un emploi du temps sans conflit maximal qui contient le cours qui se termine le plus tôt.

QUESTION 3 – En déduire que la stratégie gloutonne, qui consiste à construire incrémentalement la solution en ajoutant systématiquement le cours sans conflit avec les cours déjà choisis qui se termine le plus tôt, donne un emploi du temps sans conflit maximal.

QUESTION 4 – Écrire l'algorithme glouton ; quelle est la complexité de la solution gloutonne ?

2 Codages de Huffman

On s'intéresse au problème d'obtenir un encodage compact pour un texte. Dans cet exercice, on qualifiera donc d'alphabet un ensemble de symboles $(s_i)_i$ auxquels sont associés leurs fréquences $(f(s_i))_i$ d'apparition dans le texte.

On appelle *codage binaire* (simplement nommé codage dans la suite) d'un alphabet $\mathcal{A}$ une fonction $c : \mathcal{A} \to \{0,1\}^*$. Le codage est dit *de longueur fixe* s'il existe $n \in \mathbb{N}$ tel que $c : \mathcal{A} \to \{0,1\}^n$.

QUESTION 5 – Évaluer la taille du codage d'un fichier de n caractères par un code de longueur fixe.

Nous cherchons naturellement à faire mieux en utilisant un *codage de longueur variable*. Nous allons néanmoins ne considérer que des codages dit *préfixes* : il n'existe pas deux mots m_1 et m_2 tels que $c(m_1)$ soit un préfixe de $c(m_2)$.

QUESTION 6 – Quel intérêt voyez-vous aux codages préfixes ?

Un codage va être représenté par un arbre dont les feuilles sont les caractères de l'alphabet : le chemin de la racine à une feuille définit son encodage selon la convention qu'un déplacement à gauche équivaut à un 0, un déplacement à droite équivaut à un 1. Nous identifierons dans la suite le codage à son arbre.

QUESTION 7 – Représenter le codage suivant sous forme d'arbre :

$$\begin{array}{lcllcl} a & = & 0 & b & = & 101 \\ c & = & 100 & d & = & 111 \\ e & = & 1101 & e & = & 1100 \end{array}$$

On peut ainsi définir le coût d'un codage comme suit, où $d_c(s)$ est la profondeur de la feuille contenant le caractère s dans l'arbre de c :

$$B(c) = \sum_{s \in \mathcal{A}} f(s) * d_c(s)$$

Un codage est ainsi dit *optimal* si son coût est minimal. On dit d'un arbre binaire qu'il est *complet* si tout nœud interne (c'est-à-dire qui n'est pas une feuille) a deux fils non vides.

QUESTION 8 – Montrer que tout codage optimal est un arbre binaire *complet*.

QUESTION 9 – Proposer un algorithme glouton permettant de construire l'arbre d'un codage optimal d'un alphabet.

On souhaite à présent prouver la correction de notre algorithme, c'est à dire que notre algorithme renvoie un codage optimal.

QUESTION 10 – Soient x et y deux caractères d'un alphabet $\mathcal{A}$ ayant une fréquence minimale dans $\mathcal{A}$. Montrer qu'il existe un codage préfixe optimal c pour $\mathcal{A}$ dans lequel $c(x)$ et $c(y)$ ont même longueur et ne diffèrent que par le dernier bit.

QUESTION 11 – Soient x et y deux caractères d'un alphabet $\mathcal{A}$ ayant une fréquence minimale dans $\mathcal{A}$. Soit $\mathcal{A}'$ l'alphabet obtenu en remplaçant x et y de $\mathcal{A}$ par un nouveau caractère z de fréquence $f(z) = f(x) + f(y)$.

Soit c' un codage optimal pour $\mathcal{A}'$, montrer que le codage c obtenu en remplaçant le nœud feuille de z par un nœud interne ayant x et y pour enfants est optimal pour $\mathcal{A}$.

QUESTION 12 – Conclure.

QUESTION 13 – Quelle est le codage de Huffman d'un alphabet dont les fréquences décrivent les termes de la suite de Fibonacci ?