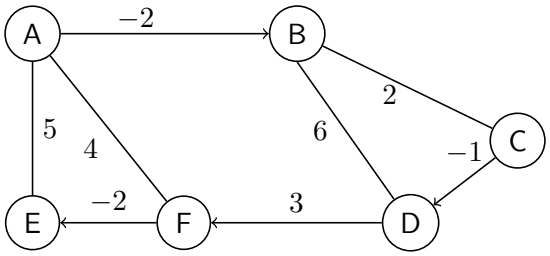
TD 7 : Programmation dynamique

Jeudi 08 novembre 2018

1 Plus courts chemins avec poids négatifs



QUESTION 1 – Appliquer l'algorithme de Bellman-ford en partant du sommet A.

2 Emploi du temps sans conflit avec pondération

Connaissant l'emploi du temps hebdomadaire de n cours attribuant un nombre de crédits fixé, on cherche à trouver le sous-ensemble de cours maximisant le nombre de crédits obtenus auxquels il est possible de s'inscrire sans qu'il n'y ait de conflit.

Note : Il existe des manières plus intelligentes de choisir ses cours en pratique...

On dispose de trois tableaux D, F, et W, de taille n, représentant respectivement les horaires de début, de fin et les poids (nombre de crédits) des cours. Pour simplifier, on suppose que la date est représentée par un nombre réel positif, que F est trié par ordre croissant 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

$$\begin{cases} \forall i, j \in G, i \neq j \implies F[i] \leq D[j] \text{ ou } F[j] \leq D[i] \\ w(G) = \Sigma_{j \in G} W[i] \text{ est maximal.} \end{cases}$$

Le poids d'un tel sous ensemble optimal utilisant au plus les k premiers cours est noté $OPT(k)$.

Rappel : il existe un algorithme glouton permettant de résoudre ce problème si tous les cours attribuent le même nombre de crédits (cf. TD 6).

QUESTION 2 – Donner un exemple illustrant que l'algorithme glouton proposé dans le cas du problème d'emploi du temps optimal sans poids n'est pas correct avec poids.

Pour tout $i \in \{1 \dots, n\}$, on note $p(i)$ le plus grand indice $j \leq i$ compatible avec i i.e. tel que $[\ D[i], F[i]\ ] \cap [\ D[j], F[j]\ ] = \emptyset$ s'il existe, 0 sinon.

QUESTION 3 – Donner une relation de récurrence liant $OPT(k)$, $p(k)$ et $W[k]$.
QUESTION 4 – Écrire l'algorithme correspondant.

3 Justification de paragraphes

En typographie, la *justification* d'un paragraphe consiste à répartir les mots entre les lignes de façon à ce que l'ensemble soit harmonieux. Pour ce problème, on utilise une police dont les caractères ont une largeur fixe. Formellement, on souhaite agencer les mots sur des lignes de longueur M. On dispose en entrée d'une séquence de n mots, de longueurs $l_1, l_2, \dots, l_n$ (tous de taille inférieure à M). Étant donnée une ligne contenant les mots de i à j (inclus), le nombre d'*espaces résiduels* à la fin de la ligne est

$$e_{i,j} = M - \sum_{k=i}^{j} l_k - (j - i).$$

Le coût d'une telle ligne est défini comme $e_{i,j}^3$, sauf pour la dernière ligne dont le coût est nul. On cherche à trouver un découpage de coût total minimal.

QUESTION 5 – On note c_j le coût minimal d'un paragraphe contenant les j premiers mots (où toutes les lignes comptent). Proposer une relation de récurrence pour c_j.
QUESTION 6 – En déduire une méthode utilisant la programmation dynamique pour le calcul du coût optimal.

4 Allocation de skis

On s'intéresse au problème d'allouer m paires de skis de longueurs $s_1, \dots, s_m$, respectivement, à n skieurs $(m \geq n)$ de tailles $h_1, \dots, h_n$. On appelle *allocation* une fonction (injective) $f : \{1, \dots, n\} \to \{1, \dots, m\}$; une allocation f est dite optimale lorsqu'elle minimise $w(f) = \sum_{k=1}^{n} |s_{f(k)} - h_k|$.

QUESTION 7 – Donner la complexité de la solution naïve qui consiste à explorer toutes les possibilités.
QUESTION 8 – Proposer une formule de récurrence pour une allocation optimale des skis. Indice : on pourra supposer que les longueurs des skis et les tailles des skieurs sont rangées dans l'ordre croissant.
QUESTION 9 – En déduire un algorithme efficace permettant de calculer la valeur optimale $w(f)$ et calculer sa complexité.

Question 10 – Expliquer comment modifier l'algorithme pour qu'il renvoie également l'allocation optimale.