Exercices “Plus courts chemins” : Correction

19 octobre 2016

1.
2.
3. Il suffit d'appliquer la fonction log sur les poids.
4. L'algorithme reste le même, mais on met à jour l'étiquette d'un sommet v en faisant $\lambda(v) := \max(\lambda(u), w(uv))$. La preuve est identique. Noter que pour ce problème, le fait que la fonction max soit croissante permet de travailler avec des poids de signe quelconque.
5. — On considère le graphe orienté D dont l'ensemble des sommets V est la collection d'intervalles $\mathcal{C}$ (on identifie alors V et $\mathcal{C}$) et dans lequel on a un arc (I, J) si l'extrémité supérieure de I est strictement inférieure à l'extrémité inférieure de J. C'est un graphe acircuitique, et les sommets des chemins élémentaires sont précisement les sous-ensembles d'intervalles deux à deux disjoints. En mettant comme poids sur tout arc (I, J) la quantité $w(I)$, et en ajoutant un dernier sommet X et tous les arcs (I, X) avec $I \in \mathcal{C}$ que l'on pondère avec $w(I)$, chercher le sous-ensemble d'intervalles de $\mathcal{C}$ deux à deux disjoints de poids maximal revient à chercher le plus long chemin de D pour la pondération w, ce qui se fait par un algorithme de programmation dynamique (le graphe étant acircuitique). Pour se ramener totalement au cas du cours, on peut également ajouter un sommet Y et tous les arcs (Y, I) avec $I \in \mathcal{C}$ que l'on pondère avec 0, et chercher le chemin de Y à X de plus grand poids.
   — Une demande de location se modélise par un intervalle de la droite réelle dont les extrémités sont les debut et fin de la location, pondéré par la gain qui serait obtenue en satisfaisant cette demande. Maximiser le gain revient alors à chercher le sous-ensemble d'intervalles disjoints (les locations ne pouvant se recouvrir) de plus grand poids.
6. On définit
$$\pi(v, k) := \text{coût minimal d'un } s\text{-}v \text{ chemin possédant exactement } k \text{ arcs.}$$
   On peut alors écrire l'équation de programmation dynamique
$$\pi(v, k+1) = \min\left(\pi(u, k) + c((u, v))\right)$$
   où le minimum est pris sur les sommets u antécédents de v. On répond à la question en comparant les valeurs de $\pi(t, k)$ pour k variant de 0 à b. Chacune de ces valeurs se calcule en temps polynomial $O(|V|^2)$, et la dernière se fait en $b + 1$ opérations, ce qui est polynomial car $b \leq |A|$.
7. — Pour chaque poutre p, on achète une poutre de type $i(p)$, où $i(p)$ est le plus petit indice i tel que $S_i \geq \sigma_p$.
   — Sur un arc (s, i), on met un coût égal à $q_j g_i$, où g_i est le nombre de poutres $p \in \mathcal{P}$ telles que $\sigma_p \in]0, S_i]$. Sur un arc (i, j) on met un coût égal à $q_j f_{ij}$, où f_{ij} est le nombre de poutres $p \in \mathcal{P}$ telles que $\sigma_p \in]S_i, S_j]$. Sur l'arc (x, t), on met un coût égal à 0. On pose $b = k + 1$.
   Les types sélectionnés dans une solution optimale du problème constituent avec s et t un s-t chemin dans ce graphe, utilisant au plus b arcs. Noter que le type x est forcément utilisé dans une solution optimale. On peut vérifier que le coût du chemin et le coût de la solution coïncident. Réciproquement, tout s-t chemin donne une solution réalisable de même coût.

— S'il existe une poutre p telle que $\sigma_p \in]S_i, S_j]$ et $(1+\theta)\sigma_p < S_j$, on supprime l'arc (i,j) du graphe D. De même pour les arcs (s,i). Dans ce nouveau graphe, la méthode de la question précédente permet encore de résoudre le problème.

8. On construit un graphe $\mathcal{D} = (\mathcal{V}, \mathcal{A})$ dont les sommets $\mathcal{V}$ sont s^{bas}, s^{haut} et tous les couples d'arcs (a, a') tels que la tête de a est la queue de a'. Les arcs de $\mathcal{A}$ sont de la forme :
   - $\big((s^{bas}, (a,a')\big)$ pour tout $(a,a') \in \mathcal{V}$ tel que $a \in \delta^+(s^{bas})$.
   - $\big((a,a'), s^{haut}\big)$ pour tout $(a,a') \in \mathcal{V}$ tel que $a' \in \delta^-(s^{haut})$.
   - $((a,a'),(a',a''))$ pour tout $(a,a'),(a',a'') \in \mathcal{V}$.

   On définit le coût d'un arc $(\,\cdot\,,(a,a'))$ de $\mathcal{D}$ comme étant $f(a,a')$ et celui d'un arc $\big((a,a'), s^{haut}\big)$ comme étant nul. Un s^{haut}-s^{bas} chemin dans D induit un s^{haut}-s^{bas} de $\mathcal{D}$ et réciproquement. N'importe quel algorithme de calcul de plus court chemin dans un graphe orienté avec des poids positifs convient.

9.

10. Ce problème se résout par la programmation dynamique. Les états sont les âges possibles de la machine en début d'année. Les périodes sont les années. On s'intéresse aux quantités

$$\pi(t,k) = \text{profit maximal possible en terminant la } t\text{ème année en possédant une machine d'âge } k.$$

On souhaite trouver $\max_{k\in\{1,2,3\}}(\pi(5,k)+p_k)$.
$\pi(t,k)$ satisfait les relations suivantes :

$$\pi(1,k) = \begin{cases} 0 & \text{pour tout } k \in \{2,3\} \\ b_0 - c_0 & \text{pour } k = 0. \end{cases}$$

$$\pi(t+1,k+1) = \pi(t,k) + b_k - c_k \quad \text{si } k \in \{1,2\}$$

$$\pi(t+1,1) = \max_{k\in\{1,2,3\}}(\pi(t,k) + p_{k-1} - 100000 + b_0 - c_0) \text{ si } t \in \{1,2,3,4\}$$

On peut alors remplir le tableau des valeurs de π

k \ t	1	2	3	4	5
1	70000	90000	110000	90000	150000
2	0	116000	136000	156000	136000
3	0	0	124000	144000	164000

Le profit maximal de l'entreprise est donc $\max(150000+50000, 136000+24000, 164000+10000) = 200000$. Une stratégie optimale consiste à garder la machine deux ans, puis la vendre et en racheter une nouvelle, encore une fois pour deux ans.

11. On pose

$$\pi(i,x) := \text{coût optimal d'une stratégie terminant sur le mois } i \text{ avec } x \text{ employés,}$$

et

$$\pi(0,x) := \begin{cases} 0 & \text{si } x = 0 \\ +\infty & \text{si } x > 0. \end{cases}$$

Noter qu'il n'ai jamais intéressant d'avoir strictement plus de 11 employés sur un mois quelconque. Pour $i \geq 1$ on a alors en notant $d(i)$ la demande en intérimaires sur le mois i :

$$\pi(i,x) = \begin{cases} \min\limits_{y\in\{0,1,\ldots,11\}}(\pi(i-1,y) + c(y,x)) & \text{si } x \in \{d(i),\ldots,11\} \\ +\infty & \text{sinon.} \end{cases}$$

avec

$$c(y,x) := \begin{cases} 1600x + 800(x-y) & \text{si } y \leq x \\ 1600x + 1200(y-x) & \text{sinon.} \end{cases}$$

On calcule alors aisément de proche en proche les valeurs de $\pi(i, x)$ pour tout i et tout x. La stratégie optimale s'obtient en calculant la valeur de $\pi(5, 11)$. (Le coût total optimal est $\pi(5, 11) + 11 \times 1200$ euros, en supposant que l'on doit mettre fin aux contrats des 11 intérimaires du mois 5.)

12.

13. — On définit

$$\pi(X, v) := \text{coût minimum d'une } s\text{-}v \text{ chaîne hamiltonienne du sous-graphe induit par } X \cup \{s, v\}.$$

On a alors pour $X \neq \varnothing$:

$$\pi(X, v) = \min_{u \in X} (\pi(X \setminus \{u\}, u) + c(uv))$$

et le cycle hamiltonien de plus petit coût est alors obtenu en comparant les valeurs de $\pi(V \setminus \{v\}, v) + c(sv)$ pour tout $v \in V \setminus \{s\}$.

— Le calcul de $\pi(X, v)$ est en $O(n)$. On fait $O(n2^n)$ fois ce calcul. La complexité d'un tel algorithme est donc $O(n^2 2^n)$, à comparer à la complexité $O(n!)$ de l'algorithme naïf. La programmation dynamique fait donc beaucoup mieux, mais cela reste exponentiel.