Corrigé de la PC 6

6 mars 2007

2. Recouvrement d'ensembles et de mots

I.1. La version de décision consiste à chercher un J tel que $\cup_{j\in J} S_j = E$ et $\sum_{j\in J} w(S_j) \leq w$.

I. 2. Il faut chosir S_v égal à l'ensemble des arêtes incidentes à v. A un recouvrement de E par une famille d'ensembles S_v on associe l'ensemble des sommets v qui apparaissent comme indice d'un S_v de recouvrement ; toute arête du graphe est clairement incidente à l'un de ceux-ci et on a un recouvrement par sommets de G. La réciproque est aussi immédiate.

II. 3. A une instance de COUVERTURE PAR SOMMETS on a asoocié une instance de COUVERTURE PAR ENSEMBLE en temps polynomial (les ensembles S_v sont représentés par les listes de voisins de chaque sommet dans le graphe) ; l'un admet une solution si et seulement si c'est le cas pour l'autre. C'est une réduction polynomiale.

II. 1. Soit $S_{i_1}, \ldots S_{i_k}$ les sous-ensembles de la solution optimale. Si pour tout j on a :

$$\frac{w(S_{i_j})}{|S_{i_j}|} > \frac{W_0}{n}$$

alors $w(S_{i_j}) > \frac{W_0|S_{i_j}|}{n}$ en sommant on obtient (comme on a un recouvrement de E : $\sum |S_{i_j}| \geq n$) $W_0 > \frac{W_0 n}{n}$ Qui est une contradiction.

II. 2. L'algoritme glouton chosit T_1 tel que $\frac{w(T_1)}{|T_1|}$ soit minimum, il est en particulier inférieur ou égal aux ratios des sous-ensembles intervenant dans la solution optimum, d'où le résultat en appliquant II. 1.

II. 3. A l'étape i on note $\overline{F} = E \setminus F$, on peut restreindre le problème à $\overline{F}$, considérer les $S'_i = S_i \setminus F$ et supposer que l'on est au début de l'algorithme. Dans cette sitaution $n - j + 1$ est un minorant de $|\overline{F}|$, d'où le résultat en utilisant la question précédente.

II. 4. Il y a au pire n étapes, et en sommant les inégalités de la question précédente, le coût obtenu par l'algorithme est inférieur ou égal à $W_0 H_n$. L'algorithme est donc H_n approché par définition de l'approximation.

II. 5 . On peut prendre comme poids $1, 1/2, 1/3 \ldots 1/n$ et $((1+\varepsilon)/n)$, l'algorithme chosisit S_n puis S_{n-1} et ainsi de suite jusqu'à S_1 obtenant un poids H_n ; alors que la solution optimale est S_{n+1} de poids $((1+\varepsilon)/n)$. Noter que l'énoncé imposait des poids entiers, il convient donc de multiplier tous les poids par $n!$ pour satisfaire cette condition.

III. 1. σ_{12k} est égal à $ab^k c$ et le f optimal est $ab^{k+1}c$.

III. 2. L'algorithme à égalité de longueur de chevauchements pour l'exemple précédent peut choisir σ_{12k} qui donne un ensemble $S = \{ab^k c, b^{k+1}\}$ composé de deux mots qui n'ont qu'un 0-chevauchement d'où la solution obtenue $ab^k c, b^{k+1}$, celle-ci est de longeur $2k+3$ alors que la solution optimale est de longeur $k+2$ le seuil est supérieur ou égal à $\frac{2k+3}{k+2}$ qui aussi proche que l'on veut de 2.

III. 3. Il suffit de prendre pour f la concaténation des mots $\theta_1, \theta_2, \dots \theta_p$. Comme les S_{θ_i} recouvrent S, tout f_j est élément d'un des S_{θ_i}, donc facteur du θ_i correspondant. Le poids W_0 de la solution optmale au problème Q est égal à la somme des longueurs des mots θ_i, mais c'est exactement la longuer de f.

III. 4. Comme on suppose que les f_i ne sont pas facteurs l'un de l'autre, on obtient que chaque f_j est facteur d'un segment de f situé entre le début d'un f_{b_i} et la fin d'un f_{e_i} (voir figure). Mais ce segment est justement égal à $\pi_i = \sigma_{b_i e_i k_i}$; ainsi tout f_j est facteur d'un π_i et il appartient au S_{π_i} correspondant. Les S_{π_i} constituent donc un recouvrement de S ; la somme de leurs poids est $\sum_{i=1}^{t} \ell(\pi_i)$

figure manquante

III. 5. Il suffit de montrer que π_1 et π_3 ne s'intersectent pas ou ce qui revient au même que tel n'est pas le cas pour f_{e_1} et f_{b_3}. Si f_{e_1} et f_{b_3} s'intersectent alors comme on le voit sur la figure ci-dessous, f_{b_2} serait facteur de f_{e_1} et l'énoncé exclut qu'un mot f_i soit facteur d'un f_j.

2eme figure manquante

III.6. On considère l'algorithme qui remplace le problème P par le problème Q correspondant, puis qui résout Q par l'algorithme glouton de la partie II, enfin qui concatène les mots u correspondants aux S_u de la solution trouvée. On perd un facteur 2 en remplaçant P par Q, et un facteur H_n en se contentant de l'algorithme glouton. On a donc un algorithme $2H_n$ approché.