ENSTA ParisTech -3e année - Cours C9-3 **2013**
Durée : 2 h.

Examen du module complexité & Graphes

Exercice 1 : Problème de *voyageur de commerce*

Montrer que le problème du voyageur de commerce ayant une métrique respectant l'inégalité triangulaire est dans NP-complet.

Correction

Rapellons le problème suivant :
Données : un graphe complet $K = (V, V \times V)$ muni d'une fonction de coût positive $C : V \times V \to \mathbb{N}$ sur les arêtes, respectant l'inégalité triangulaire, un entier k.
Question : Existe-il un cycle passant par chaque sommet exactement une fois et ayant un coût inférieur à k ?

Le problème *voyageur de commerce* ayant une métrique respectant l'inégalité triangulaire est dans NP car étant donné un cycle, on peut vérifier en temps polynomial si

- le cycle passe par chaque sommet exactement une fois
- son coût est inférieur à k.

Nous allons faire la réduction à partir du problème Cycle Hamiltonien. Soit $\mathcal{I} =< G = (V, E), >$ une instance du problème Cycle Hamiltonien.

Nous transformons cette instance en une instance $\mathcal{I}'$ du problème *voyageur de commerce* de la façon suivante. Nous affectons $|V|$ à k et nous construisons un graphe $K = (V, V \times V)$ muni de la fonction de coût sur les arêtes suivantes :

$$C(u,v) = \begin{cases} 1 & si \quad (u,v) \in E \\ 2 & si \quad (u,v) \notin E \end{cases}$$

La fonction de coût respecte bien l'inégalité triangulaire. Cette transformation se fait en temps polynomial car nous avons

- copié le graphe G et rajouté des arêtes en au pire $\mathcal{O}(|V|^2)$ opérations ;
- construit la fonction de coût C en $\mathcal{O}(|V|^2)$ opérations.

Il est facile de prouver l'assertion suivante :

il existe un cycle hamiltonnienne dans G si et seulement si il existe un cycle du voyage de commerce de coût n.

□

Exercice 2 : Problème du *Set Cover*

Rappellons la définition suivante : une *couverture de sommets* S d'un graphe G est un sous-ensemble de sommets tel que toutes les arêtes de G sont incidentes à au moins un sommet de S.

Nous allons considérer le problème (voir figure 1 pour illustration) : Set Cover
Données : un ensemble U de u éléments, un ensemble $\mathcal{S}$ composé de sous-ensembles $S_1, \ldots, S_t$ de U, et un entier k.
Question : Existe-il au plus k éléments de $\mathcal{S}$ tels que leur union est égale à U ?

Question 2.1 Montrer que ce problème est dans NP-complet.

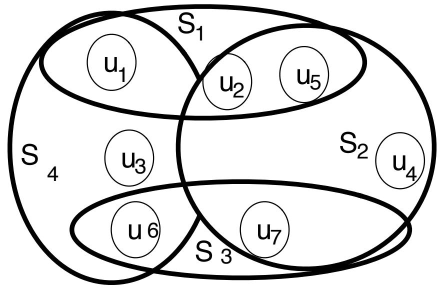

FIGURE 1 – $U = \{u_1, \dots, u_7\}$, S_1, S_2, S_4 est un *set cover* de U et de cardinalité 3.

Correction

Le problème du *Set Cover* est dans NP car étant donné un ensemble $\mathcal{S}'$ composé d'éléments de $\mathcal{S}$ on peut vérifier en temps polynomial si

- $\mathcal{S}'$ possède k éléments.
- chaque l'élément du U est contenu dans un des éléments de $\mathcal{S}'$.

Nous allons faire la réduction à partir du problème du VERTEX COVER. Rapellons la définition : VERTEX COVER

Données : un graphe $G = (V, E)$ et un entier j.

Question : Existe-il une couverture sommet C de cardinalité inférieure à égale à j ?

Soit $\mathcal{I} =< G = (V, E), k >$ une instance du problème du VERTEX COVER. Nous affectons j à k et nous construisons l'ensemble U et un ensemble $\mathcal{S}$ de la façon suivante :

- $U = E$
- $\mathcal{S} = \bigcup_{v \in V} S_v$ où $S_v = \{e \in E | v \text{ est incident à l'arête } e\}$

Cette transformation se fait en temps polynomial car

- créer l'ensemble U nécessite au pire $\mathcal{O}(|V|^2)$ opérations ;
- Pour chaque sommet v, créer l'ensemble S_v nécessite au pire $\mathcal{O}(d_v)$ opérations où d_v est le degré du sommet v ;
- Au total, créer tous les ensembles S_v nécessite au pire $\mathcal{O}(|V|^2)$ opérations.

Il reste à prouver les deux assertions suivantes :

1. Si C est une couverture sommet de G de cardinalité inférieur ou égale à j, alors

$$\mathcal{S} = \bigcup_{v \in C} S_v \text{ est un set cover de cardinalité inférieur ou égale à } j.$$

2. Si $\mathcal{S}$ est un set cover de cardinalité inférieur ou égale à j, alors

$$C = \{v | S_v \in \mathcal{S}\} \text{ est une couverture sommet de } G \text{ de cardinalité inférieur ou égale à } j.$$

$\square$

Exercice 3 : Ensemble de la coupe maximum

Nous allons considérer un graphe non-orienté $G = (V, E)$ ayant une fonction de poids sur les arêtes $w : E \to \mathbb{N}$. Une coupe d'un graphe non-orienté $G = (V, E)$ est un ensemble d'arêtes qui partagent G en deux sous-ensembles disjoints et distincts (S et $V \backslash S$). Dans la suite de l'exercice, la coupe se définit par aussi les deux sous-ensembles de sommets disjoints.

Cet exercice se traite le problème d'optimisation COUPE MAX : nous avons en entrée un graphe non orienté $G = (V, E)$ avec un poids $w(e)$ sur chaque arête e, et nous voulons partager les sommets en deux ensembles S et $V \backslash S$ afin que le poids total des arêtes entre les deux ensembles est aussi grand que possible.

Notation : Soit A et B deux ensembles de sommets disjoints. Nous noterons

$$w(A, B) = \sum_{u \in A, v \in B, (u,v) \in E} w(u, v) \text{ et } W(A) = \sum_{u \in A, v \in A, (u,v) \in E} w(u, v) \tag{1}$$

Rappellons que $\Gamma_G(u)$ est l'ensemble des sommets de G adjacents à u. Nous dirons qu'un sommet v est α-*content* pour la coupe $(A,\ B)$ dans le graphe G si et seulement si

- $w(A \backslash \{u\}, B \cup \{u\}) \leq \alpha w(A, B)$ si $u \in A$
- ou $w(A \cup \{u\}, B \backslash \{u\}) \leq \alpha w(A, B)$ si $u \in B$

Un peu d'entrainement.

Considérons le graphe biparti complet de 8 sommets (voir la figure 1). Toutes les arêtes de ce graphe ont un poids égal à 1 : i.e. $\forall e \in E$, $w(e) = 1$.

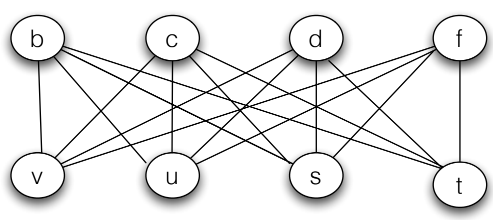

FIGURE 2 – Graphe biparti complet de huit sommets. Toutes ses arêtes ont un poids égal à 1.

Question 3.1 Compléter les trois tableaux (recopier les sur votre copie) :

A	$B = V \backslash A$	$w(A, V \backslash A)$
$\{b\}$	$\{c, d, f, v, u, s, t\}$	4
$\{b, v\}$		
$\{b, c, v\}$		
$\{b, c, d, f\}$		

A	B	$A \backslash \{b\}$	$w(A \backslash \{b\}, B \cup \{b\})$	$w(A, B)$
$\{b, c, d, f\}$	$V \backslash A$	$\{c, d, f\}$	12	16
$\{b, c, d, v\}$	$V \backslash A$			
$\{b, u, s, t\}$	$V \backslash A$			

A	B	$W(A)$	$W(B)$	b est 1-content
$\{b, c, d, f\}$	$V \backslash A$	0	0	oui car ...
$\{b, c, d, v\}$	$V \backslash A$			
$\{b, u, s, t\}$	$V \backslash A$			

Correction

A	$B = V\backslash A$	$w(A, V\backslash A)$
$\{b\}$	$\{c, d, f, v, u, s, t\}$	4
$\{b, v\}$	$\{c, d, f, u, s, t\}$	6
$\{b, c, v\}$	$\{d, f, u, s, t\}$	2+3+3
$\{b, c, u\}$	$\{d, f, v, s, t\}$	8
$\{b, v, u\}$	$\{c, d, f, s, t\}$	8
$\{b, c, v, u\}$	$\{d, f, s, t\}$	8
$\{b, c, d, f\}$	$\{v, u, s, t\}$	16

A	B	$A\backslash\{b\}$	$w(A\backslash\{b\}, B \cup \{b\})$	$w(A, B)$
$\{b, c, d, f\}$	$V\backslash A$	$\{c, d, f\}$	12	16
$\{b, c, d, v\}$	$V\backslash A$	$\{c, d, v\}$	8	10
$\{b, c, u, v\}$	$V\backslash A$	$\{c, u, v\}$	8	8
$\{b, u, s, t\}$	$V\backslash A$	$\{u, s, t\}$	12	10

A	B	$W(A)$	$W(B)$	b est 1-content
$\{b, c, d, f\}$	$V\backslash A$	0	0	oui car $w(A\backslash\{b\}, B \cup \{b\}) \leq \alpha w(A, B)$
$\{b, c, d, v\}$	$V\backslash A$	3	3	oui car $w(A\backslash\{b\}, B \cup \{b\}) \leq \alpha w(A, B)$
$\{b, c, u, v\}$	$V\backslash A$	4	4	oui car $w(A\backslash\{b\}, B \cup \{b\}) \leq \alpha w(A, B)$
$\{b, u, s, t\}$	$V\backslash A$	3	3	non car $w(A\backslash\{b\}, B \cup \{b\}) > \alpha w(A, B)$

□

Le problème COUPE MAX est trouver une coupe (A, B) qui maximise $w(A, B)$ sur l'ensemble des sous-ensembles de V. Considérons l'algorithme suivant pour le COUPE MAX :

Algorithme Max-Cut-Local

Entrée : Un graphe $G = (V, E)$ et une fonction poids $w : E \to \mathbb{N}$

Sortie : Deux sous-ensembles de sommets A et B.

1. Choisir une partition arbitraire de sommets (A, B) de V.
2. Tant qu'il existe un sommet v qui n'est pas α-*content*
   – Si v est dans A, alors $A \leftarrow A\backslash\{v\}$ et $B \leftarrow B \cup \{v\}$
   sinon $B \leftarrow B\backslash\{v\}$ et $A \leftarrow A \cup \{v\}$
3. retourner les deux ensembles (A, B)

Question 3.2 Exécuter l'algorithme ayant en entrée le graphe de la figure 1 sachant que $\alpha = 1$ et que la partition initiale est la suivante

1. $A = \{b, c, d\}$ et $B = \{v, u, s, t, f\}$
2. $A = \{v, u, b, c\}$ et $B = \{d, f, s, t\}$

Correction

Pour l'exécution sur la partition initiale $A = \{v, u, b, c\}$ et $B = \{d, f, s, t\}$: Tous les sommets sont contents. L'algorithme retourne cette partition.

Pour l'exécution sur la partition initiale $A = \{b, c, d\}$ et $B = \{v, u, s, t, f\}$: seul le sommet f n'est pas content. L'algorithme modifie la partition. Elle devient $A = \{b, c, d, f\}$ et $B = \{v, u, s, t\}$. Ici, dans la nouvelle partition, tous les sommets sont contents. L'algorithme retourne cette partition.

□

Par la suite, la solution retournée par l'algorithme Max-Cut-Local est notée (A, B) tandis une partition correspondant à une coupe optimale est notée (A^*, B^*).

Question 3.3 Donner une relation entre $W(E)$, $W(A)$, $W(B)$, $w(A, B)$.

Correction

$$W(E) = W(A) + W(B) + w(A, B).$$

□

Question 3.4 Montrer que pour tout sommet $u \in A$, on a $w(\{u\}, A) \leq w(\{u\}, B) + (\alpha - 1)w(A, B)$. *Indication : u est α-content.*

Correction

Par définition on a $w(A\backslash\{u\}, B\cup\{u\}) = w(A, B) + w(\{u\}, A) - w(\{u\}, B)$. Comme u est α-content, on a $w(A\backslash\{u\}, B\cup\{u\}) \leq \alpha w(A, B)$ si $u \in A$. En combinant les deux équations, on obtient :

$$w(A, B) + w(\{u\}, A) - w(\{u\}, B) \leq \alpha w(A, B)$$

□

Cas où $\alpha = 1$

Question 3.5 Montrer que pour tout sommet $u \in A$, on a $w(\{u\}, B) \geq \frac{w(\{u\},\Gamma_G(u))}{2}$.

Correction

D'après la question précédente et le fait que $\alpha = 1$, on a

$$w(\{u\}, A) \leq w(\{u\}, B) + 0 \times w(A, B)$$

Commme $w(\{u\}, \Gamma_G(u)) = w(\{u\}, A\backslash\{u\}) + w(\{u\}, B)$, on a

$$w(\{u\}, \Gamma_G(u)) \leq 2 * w(\{u\}, B)$$

□

Question 3.6 Montrer que $2w(A, B) \geq W(E)$ et que $2w(A, B) \geq w(A^*, B^*)$.

Correction

Comme une arête a deux extrémités on a

– pour A : $w(A, B) = \sum_{u\in A} w(\{u\}, B) \geq \sum_{u\in A} \frac{w(\{u\},\Gamma_G(u))}{2}$

– pour B : $w(A, B) = \sum_{u\in B} w(\{u\}, A) \geq \sum_{u\in B} \frac{w(\{u\},\Gamma_G(u))}{2}$

En sommant, on obtient $2w(A, B) \geq \sum_{u\in V} \frac{w(\{u\},\Gamma_G(u))}{2}$.

Il suffit de constater que $W(E) \geq w(A^*, B^*)$

□

Question 3.7 Quelle est la complexité de l'algorithme ? (*indication : évaluer au pire le nombre d'itérations de l'algorithme en fonction du nombre d'arêtes*)

Correction

Nombre d'instructions

– L'instruction (1) peut se faire en $\mathcal{O}(n)$ opérations

– Le nombre de fois qu'on peut rentrer dans la boucle "tant que" est majoré par $W(E)$.

– La condition du “tant que” peut se faire en $\mathcal{O}(W(E))$ opérations (Mais cela peut se réaliser plus rapidement)

La complexité de l'algorithme est égale à $\mathcal{O}(|E|W(E))$ □

Question 3.8 Déduisez que l'algorithme Max-Cut-Local est d'un algorithme polynomial d'approximation avec un rapport 2 pour le problème COUPE MAX.

Correction

La question précédente, l'algorithme calcule cette partition en temps polynomial. Et de plus comme $2w(A, B) \geq w(A^*, B^*)$, la solution calculée est à distance 2 de l'optimale. □

Question 3.9 Que pouvez-vous déduire des questions 4.7 et 4.8 ?

Correction

Nous ne pouvons pas réduire cet interval, puisque cet algorithme peut retourner une solution à distance 2 de l'optimal. Il suffit de considérer l'exemple de la partie d'entrainement. Supposons que l'algorithme choisit lors de l'instruction 1 comme ensemble $A = \{b, c, v, u\}$ et $B = \{d, f, s, t\}$. Comme tous les sommets sont α-contents, alors l'algorithme retourne la coupe (A, B) avec $w(A, B) = 8$. La solution optimale (A^*, B^*) est la suivante :$A^* = \{b, c, d, f\}$ et $B^* = \{u, v, s, t\}$. Donc nous avons $2w(A, B) = w(A^*, B^*)$. □