Problèmes NP-complets

Dire qu'un problème est NP-complet, c'est à dire qu'il est dans NP (borne supérieure) et qu'il est difficile pour cette classe (borne inférieure). Le plus souvent, le premier point est facile alors que le second est plus délicat. Pour le montrer, on effectue souvent une réduction à un autre problème dont on sait déjà qu'il est NP-difficile. Les exercices suivants permettent de se familiariser avec cette technique.

Problèmes de logique

Considérez les deux problèmes suivants :

k-SAT NAE
Données : un ensemble U de variables $\{u_1, u_2, \ldots, u_\ell\}$ et une formule logique $L = C_1 \wedge \cdots \wedge C_\ell$ avec $C_i = (y_{i,1} \vee y_{i,2} \vee \cdots \vee y_{i,k})$ où $y_{i,j}$ est égal soit à l'un des u_k ou soit à l'un des $\overline{u_k}$
Question : Existe-t-il une fonction $t : U \to \{0, 1\}$ telle que t satisfait L et telle que les variables de chaque clause ne sont pas toutes de la même valeur ?

2-SAT
Données : un ensemble U de variables $\{u_1, u_2, \ldots, u_\ell\}$ et une formule logique $L = C_1 \wedge \cdots \wedge C_\ell$ ayant des clauses de 2 littéraux
Question : Existe-t-il une fonction $t : U \to \{0, 1\}$ telle que t satisfait ϕ ?

Exercice 1 Montrer que 4-SAT NAE est NP-complet sachant que 3-SAT est NP-complet. *Indication : Introduire une nouvelle variable z et l'insérer dans toutes les clauses*

Exercice 2 Montrer que 3-SAT NAE est NP-complet sachant que 4-SAT NAE est NP-complet. 3-SAT est NP-complet. *Indication : Utiliser la technique pour la transformation de SAT en 3-SAT.*

Exercice 3 2-SAT et 3-SAT

1. Donner une fonction t qui satisfait $\phi_1 : \phi_1(u_1, u_2, u_3) = (u_2 \vee \neg u_3) \wedge (\neg u_2 \vee u_3) \wedge (u_2 \vee \neg u_1)$.
2. Que se passe-t-il pour $\phi_2 : \phi_2(u_1, u_2) = (u_2 \vee u_3) \wedge (\neg u_2 \vee u_3) \wedge (\neg u_3 \vee u_1) \wedge (\neg u_3 \vee \neg u_1)$?
3. Montrer que $u \vee v = (\neg u \to v) \wedge (\neg v \to u)$.
   A partir d'une instance (U, ϕ) de 2-SAT, nous construisons un graphe orienté $G_\phi = (V, E)$ tel que
   – un sommet par un littéral de U
   – un arc par implication (en transformant chaque clause par deux implications)
4. Dessiner le graphes G_{ϕ_1} et G_{ϕ_2}

5. Montrer qu'il existe une variable u de U tel que G_ϕ contient un cycle entre u vers $\neg u$ dans G, si et seulement si ϕ n'est pas satisfiable.
6. Montrer que 2-SAT est solvable par un algorithme en temps polynomial.
7. En déduire la classe de complexité de 2-SAT.

Problèmes liés aux cycles hamiltoniennes

Exercice 4 Chaine hamiltonienne.

Considérez les trois problèmes suivants.

Cycle Hamiltonien
Données : un graphe non-orienté G.
Question : G contient-il un cycle hamiltonien ?

Chaine Hamiltonienne
Données : un graphe non-orienté G, deux sommets u et v distincts de G.
Question : G contient-il une chaine hamiltonienne entre u et v ?

Chaine
Données : un graphe non-orienté G de n sommets, deux sommets u et v distincts de G.
Question : G contient-il une chaine de longueur $n/2$ entre u et v ?

Montrer que

1. les problèmes Chaine Hamiltonienne et Chaine sont dans NP.
2. Chaine Hamiltonienne et Chaine sont NP-complets.

Exercice 5 Chevaliers de la table ronde

Etant donnés n chevaliers, et connaissant toutes les paires de féroces ennemis parmi eux, est-il possible de les placer autour d'une table circulaire de telle sorte qu'aucune paire de féroces ennemis ne soit côte à côte ?

Problèmes de graphe

Nous supposerons que le problème Couverture de sommets est NP-complet
Couverture de sommets
Données : un graphe non-orienté G et un entier k.
Question : G contient-il une *couverture de sommets* de cardinalité au plus k : c'est-à-dire un ensemble $\mathcal{S}$ de sommets tel que toutes les arêtes de G sont incidentes à au moins un sommet de $\mathcal{S}$?

Exercice 6 Le problème du stable maximum.

STABLE
Données : un graphe non-orienté $G = (V, E)$ et un entier k.
Question : Existe-t-il un sous-ensemble $V' \subseteq V$, avec $|V'| = k$, tel que $u, v \in V' \Longrightarrow (u, v) \notin E$?

Question 6.1 Montrer que

1. si le graphe a une couverture sommet de cardinalité k, alors il a un stable de cardinalité $n - k$ où n est le nombre de sommets.
2. et réciproquement

Question 6.2 Montrer que le problème STABLE est NP-complet.

Exercice 7 Le problème de la clique maximum.

Considérons le problème de décision CLIQUE :
Données : un graphe non-orienté $G = (V, E)$ et un entier k.
Question : Existe-t-il une clique de taille k (un sous graphe complet de k sommets) ?

Question 7.1 Nous noterons par G^c le complémentaire du graphe G.

Montrer que G a une clique de taille k si et seulement si G^c a une couverture sommet de taille $n - k$.

Question 7.2 Montrer que le problème CLIQUE est NP-complet.

Nous allons travailler sur une restriction du problème CLIQUE en considérant uniquement les graphes dans lequels tous leurs sommets sont de degré au plus 3. Nous le noterons 3-CLIQUE.

Question 7.3 Montrer que 3-CLIQUE est dans NP.

Question 7.4 Trouver l'erreur dans le raisonnement suivant : *Nous savons que le problème* CLIQUE *est NP-complet, il suffit donc de présenter une réduction de* 3-CLIQUE *à* CLIQUE. *Étant donné un graphe G dont les sommets sont de degré inferieur à* 3*, et un entier k, la réduction laisse inchanger le graphe et le paramètre k : clairement le résultat de la réduction est une entrée possible pour le problème CLIQUE. Par ailleurs, la réponse aux deux problèmes sont identiques. Cela prouve la justesse de la réduction et, par conséquent, la NP-complétude de la* 3-CLIQUE.

Question 7.5 Donner un algorithme polynomial en $O(|V|^4)$ pour le problème 3-CLIQUE.

Le problème du k-centre

Dans cette partie, nous allons nous concentrer sur le problème du k-centre : étant donné un ensemble de villes dont les distances sont spécifiées, choisir k villes afin d'installer des entrepôts de façon à minimiser la distance maximale d'une ville à l'entrepôt le plus proche. Un tel ensemble de k villes est appelé k-centre.

Le problème associé de décision est le suivant k-CENTRE
Données : un graphe complet $K = (V, E)$ muni d'une fonction de poids w sur les arêtes, et des entiers strictement positifs k et b.
Question : Existe-il un ensemble S de sommets tel que $|S| = k$ et tel que tout sommet v de V satisfait la condition suivante

$$min\{w(v, u) : u \in S\} \leq b$$

NP-complétude du problème k-centre

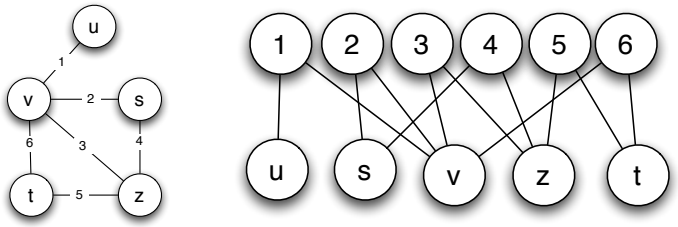
Exercice 8 Ensemble dominant

Rappellons les définitions suivantes :

- une couverture de sommets S du graphe G est un sous-ensemble de sommets tel que toutes les arêtes de G sont incidentes à au moins un sommet de S.
- un ensemble dominant C du graphe G est un sous-ensemble de sommets tel que tout sommet est soit dans C soit voisin d'un sommet de C.

Soit G un graphe $G = (V, E)$. Nous allons construire un graphe $G' = (V', E')$ à partir de G tel que

- $V' = V \cup E$;
- $E' = E \cup \{(v, a) | v \in V, a \in E, v$ est extremité de l'arête a dans $G\}$



Graphe G Graphe G'

Question 8.1 Montrer que si S est une couverture de sommets du graphe G, alors S est un ensemble dominant de G'.

Question 8.2 Exprimer le problème de minimisation de l'ensemble dominant sous forme de problème de décision.

Question 8.3 Montrer que ce problème est dans NP.

Question 8.4 Montrer que ce problème est dans NP-complet.

Exercice 9 Complexité du problème k-centre

Question 9.1 Montrer que k-CENTRE est dans NP.

Question 9.2 Montrer que k-CENTRE est NP-complet sachant que DOMINANT est NP-complet.

Approximation et in-approximation

Exercice 10 Approximation du problème

Nous allons nous focaliser sur le problème de k-CENTRE MÉTRIQUE. Ici, la fonction de poids respecte l'inégalité triangulaire.

Question 10.1 Le *carré* d'un graphe G, noté G^2, est défini comme le graphe qui contient une arête (u, v) pour chaque couple de sommets u et v reliés par un chemin de longueur au plus 2 dans G.

Montrer que si le sous-ensemble de sommets I est un stable de G^2 et alors pour tout ensemble dominant D de G, on a $|I| \leq |D|$.

Question 10.2 Dans la suite de l'exercice, nous supposerons que les arêtes de K sont triés par poids croissant, c'est-à-dire $w(e_1) \leq w(e_2) \leq \cdots \leq w(e_m)$ et nous notons $K_i = (V, E_i)$, où $E_i = \{e_1, e_2, \ldots, e_i\}$.

Montrer que trouver un k-centre estéquivalent à trouver le plus petit index i tel que G_i admet un dominant de taille au plus k

Question 10.3 Voici l'algorithme $\mathcal{A}$ du k-centre :

1. Construire $(G_1)^2, (G_2)^2, \ldots, (G_m)^2$.
2. Calculer un stable maximal, M_i, pour chaque $(G_i)^2$.
3. Calculer le plus petit indice i tel que $|M_i| \leq k$. Notons-le j.
4. Renvoyer M_j.

Donner la complexité de cet algorithme.

Notons OPT le coût d'un k-centre optimal. Notons aussi par i^* le plus petit indice, $w(e_i*) = OPT$ et tel que G_i* contient un k-centre optimal.

- Montrer que si j est l'indice calculé par l'algorithme, alors $w(e_j) \leq OPT$.
- Montrer que l'algorithme $\mathcal{A}$ est une 2-approximation.

Question 10.4 Considérons le graphe complet de $n + 1$ sommets possèdant un sommet singulier u. Chaque arête incidente à u est de poids 1 et les autres sont de poids 2.

Appliquer l'algorithme sur ce graphe en considérant la pire solution. Qu'en déduisez-vous ?

Exercice 11 In-approximation du problème

Question 11.1 Montrer que si $P \neq NP$, quel que soit $\epsilon > 0$, il n'existe pas de $(2 - \epsilon)$-approximation pour le problème du k-CENTRE MÉTRIQUE.

Question 11.2 Montrer que quel que soit $\alpha(n)$ calculable en temps polynomial, il n'existe pas de $\alpha(n)$-approximation pour k-CENTRE, à moins que $P = NP$.

Le problème du 2-partition.

Complexité du problème 2-partition.

Exercice 12 NP-complétude du problème 2-partition.

2-PARTITION
Données : un ensemble de n entiers $A = \{a_1, \ldots, a_n\}$.
Question : Existe-t-il un ensemble $A_1 \subset A$ tel que $\sum_{a \in A_1} a = \sum_{a \notin A_1} a$?

Question 12.1 Donner la taille de l'instance du problème 2-PARTITION.

Question 12.2 Montrer que le problème 2-PARTITION est NP-complet.

Exercice 13 Programmation dynamique
Nous considérons un entier $\alpha = \frac{\sum_{a \in A} a}{2}$ et un tableau de booléen

$$T : [1, \ldots n] \times [1, \ldots \alpha] \times [1, \ldots \alpha].$$

L'objectif est de réaliser un l'algorithme qui remplit ce tableau de telle façon que $T[i, w_1, w_2] = true$ si et seulement si il existe une partition des i premiers élements de A telle que la somme du premier (resp. second) sous-ensemble est égale à w_1 (resp. w_2).

Question 13.1 Donner la formule de récurrence.

Question 13.2 Donner l'algorithme utilisant la programmation dynamique qui remplit ce tableau.

Question 13.3 Donner la complexité de cet algorithme.

Approximation du problème 2-partition.

Le problème de décision 2-partition est NP-complet. Considérons le problème d'optimisation associé à 2-Partition :
Données : un ensemble de n entiers $A = \{a_1, \ldots, a_n\}$.
Objectif : un ensemble $A_1 \subset A$ tel que $max(\sum_{a \in A_1} a, \sum_{a \notin A_1} a)$ soit minimal.

Exercice 14 Approximation gloutonne.

On propose l'algorithme glouton suivant : Considérer les entiers dans l'ordre $a_1, \ldots, a_n$, et assigner à chaque étape le sous-ensemble le plus $\ll$ petit $\gg$ (c'est-à-dire en considérant la somme des entiers).

Soit $X = \sum_{j=1}^{n} a_j$ et soit OPT la valeur de solution optimale.

Question 14.1 Soit 3 entiers 1, 1 et 2. Quelle est la solution optimale et celle donnée par l'algorithme glouton ?

Question 14.2 Donner des minorants de OPT en fonction de X et des a_i ?

Soit S_1 et S_2 les sous-ensembles de $a_1, \ldots, a_n$ fournis par l'algorithme glouton en supposant que $\sum_{a \in S_1} a \geq \sum_{a \in S_2} a$. Supposons que a_j est le dernier entier inséré dans S_1 par l'algorithme glouton.

Question 14.3 Montrer que $\sum_{a \in S_1} a \leq Opt + Opt/2$.

Question 14.4 Que déduisez-vous ?

Exercice 15 Amélioration de l'approximation gloutonne.

A partir de maintenant, nous supposons que les entiers sont au départ triés de façon décroissants.

Question 15.1 Montrer que

1. si $a_j \leq OPT/3$, alors $\sum_{a \in S_1} a \leq \frac{7}{6} OPT$
2. si $a_j > OPT/3$, alors $j \leq 4$

Question 15.2 Montrer que pour tout ensemble de quatre entiers triés de façon décroissant, l'algorithme glouton calcule une solution optimale.

Question 15.3 Soit 5 entiers $3, 3, 2, 2, 2$. Quelle est la solution optimale et celle donnée par l'algorithme glouton dans ce contexte ?

Question 15.4 Que déduisez-vous ?

Le problème du Sac-à-Dos.

Le problème du Sac-à-Dos est un problème classique en informatique. Il modélise une situation analogue au remplissage d'un sac. Une personne veut remplir un sac à dos ne pouvant pas supporter plus d'un certain poids $C \in \mathbb{N}$, et elle dispose de n objets (On note l'ensemble des objectspar $\mathcal{O} = \{1, \ldots, n\}$). Chaque objet i a une valeur $v_i \in \mathbb{N}\backslash\{0\}$ et un poids $p_i \in \mathbb{N}\backslash\{0\}$. Le problème est de trouver un ensemble d'objets tel que

- tous les objets de cet ensemble puissent être mis dans le sac.
- la somme des valeurs de ces objets soit maximale.

Exercice 16 Programmation dynamique

Nous allons construire un tableau T dans lequel les lignes seront indexées par les objets et les colonnes par les valeurs. L'élément $T[i, j]$ représentera la valeur maximale pour un sac-à-dos de capacité j à l'aide des i premiers objets.

Question 16.1 Donner la formule de récurrence.

Question 16.2 Donner l'algorithme utilisant la programmation dynamique.

Question 16.3 Donner la complexité de cet algorithme.

Programmation Dynamique

Exercice 17 Le chemin le plus long dans un graphe orienté et ordonné

Soit $G = (V, E)$ un graphe orienté avec $V = \{v_1, \ldots, v_n\}$. On dit que G est ordonné si il vérifie les deux propriétés suivantes :

1. Chaque arc de ce graphe est de la forme $(i \to j)$ si $i < j$
2. Tous les sommets sauf le sommet v_n ont au moins un arc sortant.

L'objectif est de trouver la longueur du chemin le plus long commençant par v_1 finissant par v_n (si il n'en existe pas, la valeur retournée doit être égale à zéro).

Question 17.1 Montrer que l'algorithme suivant ne résoud pas correctement le problème.

1. $w \leftarrow v_1$;
2. $L \leftarrow 0$;
3. tant qu'il existe un arc sortant du sommet w
   (a) choisir l'arc $(w \to v_j)$ tel que j est le plus petit possible
   (b) $w \leftarrow v_j$;
   (c) $L \leftarrow L + 1$;
4. retourner L

Question 17.2 Donner un algorithme qui retourne la longueur du chemin le plus long commençant par v_1 finissant par v_n.

Exercice 18 Planning

Considérons un chef de projet qui doit gérer une équipe en lui affectant un projet qui dure une semaine. Le chef de projet doit choisir si il prend un projet « stressant » ou « non-stressant » pour la semaine.

1. Si le chef de projet choisit le projet qui n'est pas « stressant » durant la semaine i, alors l'entreprise reçoit un revenu ℓ_i.
2. Si le chef de projet choisit le projet qui est « stressant » durant la semaine i, alors l'entreprise reçoit un revenu h_i et l'équipe ne travaille pas durant la semaine $i - 1$

Exemple : Si chef de projet choisit de se reposer à la semaine 1, puis de prendre le projet « stressant » à la semaine 2, puis de prendre les projets « non-stressant » à la semaine 3 et 4. Le revenu total est de 70 et il correspond au maximun.

	semaine 1	semaine 2	semaine 3	semaine 4
ℓ	10	1	10	10
h	5	50	5	1

Question 18.1 Montrer que l'algorithme suivant ne résoud pas correctement le problème.

1. pour chaque itération $i = 1, \ldots, n$
   (a) si $h_{i+1} > \ell_i + \ell_{i+1}$ alors
      i. Choisir « Ne pas travailler à la semaine i »
      ii. Choisir « le projet « stressant » à la semaine i+1 »
      iii. Continuer à l'itération $i + 2$
   (b) sinon
   (a) Choisir « le projet « non-stressant » à la semaine i »
   (b) Continuer à l'itération $i + 1$

Question 18.2 Donner un algorithme qui retourne le revenu maximal que peut obtenir le chef de projet.