Tutorat de Complexité - 1

La totalité des problèmes posés sont tirés des références suivantes :

- Le cours *Introduction à la complexité* de Paul Rozière :
- Le cours *Fondements de l'Informatique : Logique, modèles, calculs* d'Olivier Bournez :
- *Computational Complexity* de Christos Papadimitriou

1 Réductions

Définition 1.1 (`MAXSAT`). On s'intéresse au problème :

- Entrée : Soit $\mathcal{C}$ un ensemble de clauses et $n \in \mathbb{N}$ avec $n \leqslant \text{Card}(\mathcal{C})$.
- Question : Existe-t-il une valuation satisfaisant au moins n clauses de $\mathcal{C}$?

Question 1. Montrer que `MAXSAT` est NP-complet.

> On va réduire le problème `SAT` à `MAXSAT`. On définit la fonction f qui à un ensemble $\mathcal{C}$ de clauses associe le couple $(\mathcal{C}, |\mathcal{C}|)$. La fonction f se calcule en temps polynomial.
>
> Si $\mathcal{C}$ est satisfaisable, il est possible de trouver une valuation satisfaisant au moins $|\mathcal{C}|$ clauses, il suffit pour cela de prendre la valuation satisfaisant $\mathcal{C}$.
>
> De même, s'il est possible de satisfaire $|\mathcal{C}|$ clauses de $\mathcal{C}$, c'est que $\mathcal{C}$ est satisfaisable.

Définition 1.2 (`NAESAT`). On s'intéresse au problème :

- Entrée : Soit $\mathcal{C}$ un ensemble de clauses.
- Question : Existe-t-il une valuation v telle que v et $\bar{v}$ satisfont $\mathcal{C}$

Question 2. Montrer que `NAESAT` est NP-complet.

> On va réduire `SAT` à `NAESAT`. Soit ϕ la fonction qui a une clause et une variable associe une clause telle que $\phi(\bigvee_j l_j, z) = \bigvee_j l_j \vee \neg z$. On définit f par $f(\bigwedge_i c_i) = \bigwedge_i \phi(c_i, z)$, où z est une variable n'apparaissant pas dans $\bigwedge_i c_i$. La fonction f se calcule en temps polynomial. Soit $\mathcal{C} = \{c_i\}$ un ensemble de clauses.
>
> Si $\mathcal{C}$ est satisfait par une valuation v, alors $v \cup \{z \mapsto 1\}$ satisfait $f(\mathcal{C})$

puisque toutes les clauses apparaissant dans $f(\mathcal{C})$ sont des sur-clauses de celles apparaissant dans $\mathcal{C}$ et $\bar{v} \cup z \mapsto 0$ satisfait $f(\mathcal{C})$ puisque $\neg z$ apparait dans toutes les clauses.

Réciproquement, si $f(\mathcal{C})$ admet une valuation v qui la satisfait, telle que $\bar{v}$ la satisfait aussi, alors une deux deux valuations, associe 1 à z et elle satisfait alors $\mathcal{C}$.

Définition 1.3 (2-clause). Une *2-clause* est une clause comptant au plus 2 littéraux.

Définition 1.4 (`MAX2SAT`). On s'intéresse à la variante de `MAXSAT` suivante :

- Entrée : Soit $\mathcal{C}$ un ensemble de 2-clauses et $n \in \mathbb{N}$ avec $n \leqslant \text{Card}(\mathcal{C})$.
- Question : Existe-t-il une valuation satisfaisant au moins n clauses de $\mathcal{C}$?

Question 3. Soit v une valuation. En fonction des images de x, y et z par v, étudier le nombre de 2-clauses satisfaisables parmi

$$\{x, y, z, w, \neg x \vee \neg y, \neg x \vee \neg z, \neg y \vee \neg z, x \vee \neg w, y \vee \neg w, z \vee \neg w\}$$

On commence par remarquer que x, y et z ont des rôles symétriques.

Si $x = y = z = 0$ alors au plus 6 des 10 clauses sont satisfaisables. Dans tous les autes cas, il est possible de satisfaire 7 des 10 clauses. Il n'est en aucun cas possible d'en satisfaire plus que 7.

Question 4. Montrer que `MAX2SAT` est NP-complet.

On va réduire `3SAT` à `MAX2SAT`. On définit ϕ par

$$\begin{aligned}\phi\left(\bigwedge_i (l_1^i \vee l_2^i \vee l_3^i)\right) = \bigwedge_i (&l_1^i \wedge l_2^i \wedge l_3^i \wedge w_i \wedge \\ &(\neg l_1^i \vee \neg l_2^i) \wedge (\neg l_1^i \vee \neg l_3^i) \wedge (\neg l_2^i \vee \neg l_3^i) \wedge \\ &(l_1^i \vee \neg w_i) \wedge (l_2^i \vee \neg w_i) \wedge (l_3^i \vee \neg w_i))\end{aligned}$$

où les w_i sont des variables n'apparaissant pas dans $\bigwedge_i (l_1^i \vee l_2^i \vee l_3^i)$. Et f par $f(\mathcal{C}) = (\phi(\mathcal{C}), 7\,|\mathcal{C}|)$. f se calcule en temps polynomial.

S'il existe une valuation v satisfaisant un ensemble $\mathcal{C}$ de 3-clauses, alors en étendant v avec les bonnes valeurs sur les w_i, il est possible d'après la question 3 de satisfaire au moins $7\,|\mathcal{C}|$ clauses de $\phi(\mathcal{C})$.

Réciproquement, s'il existe une valuation v satisfaisant au moins $7\,|\mathcal{C}|$ clauses de $\phi(\mathcal{C})$, comme d'après la question 3 il est impossible de satisfaire plus de 7 2-clauses par groupe de 10 correspondant à l'image par ϕ d'une 3-clause de $\mathcal{C}$, on en déduit que dans chacun de ces groupes, exactement 7 2-clauses sont satisfaites, ce qui signifie que la restriction de v satisfait toutes les clauses de $\mathcal{C}$.

Définition 1.5 (`TABLERONDE`). On s'intéresse au problème :

- Entrée : Soit E un ensemble de chevaliers et l une liste de paires de chevaliers qui se détestent.
- Question : Le Roi Arthur (qui fait partie de E) peut-il faire un plan de table tel que deux chevaliers se détestant ne sont pas côte-à-côte ?

Question 5. Montrer que `TABLERONDE` est NP-complet.

> On constate que résoudre `TABLERONDE` revient à trouver un cycle hamiltonien sur le graphe complémentaire de celui définit par la liste des détestations. Comme il est possible de construire en temps polynomial le complémentaire d'un graphe, `CYCLEHAMILTONIEN` se réduit à `TABLERONDE`.

2 Clauses de Horn

Notations. On notera la clause $p_1 \vee \cdots \vee p_n \vee \neg q_1 \vee \cdots \vee \neg q_m$ par le couple de deux ensembles finis de variables propositionnelles

$$\{q_1, \dots q_m\} \to \{p_1, \dots, p_n\}$$

La clause vide, $\to$, s'interprète sémantiquement par l'absurde.

Définition 2.1 (Partie positive et négative). La *partie positive* de $\Gamma \to \Delta$ désigne Δ, la *partie négative* Γ.

Définition 2.2 (Clauses de Horn). Une clause de Horn est une clause dont la partie positive contient au plus une variable propositionnelle.

Définition 2.3 (Coupures). La règle de *coupure* associe à deux clauses une nouvelle clause selon le schéma suivant :

$$\frac{\Gamma_1 \cup \{p\} \to \Delta_1 \qquad \Gamma_2 \to \Delta_2 \cup \{p\}}{\Gamma_1 \cup \Gamma_2 \to \Delta_1 \cup \Delta_2}$$

Définition 2.4 (Déduction et réfutation par coupures). On définit inductivement la *déduction par coupures*. Une clause c se *déduit par coupures* d'un ensemble de clauses S si et seulement si $c \in S$ ou c est obtenue par une règle de coupure à partie de clauses qui se déduisent par coupures de S.

Un ensemble de clauses est *réfutable par coupures* signifie que la clause vide se déduit de S.

Question 6. Montrer que si un ensemble de clauses n'est pas satisfaisable il contient nécessairement une clause positive.

> Si toutes les clauses contiennent une variable niée, la valuation associant 0 à toutes les variables satisfait toutes les clauses.

Question 7. Montrer que le fait d'être une clause de Horn est stable par coupures.

Reprenons la règle de coupure :

$$\frac{\Gamma_1 \cup \{p\} \to \Delta_1 \qquad \Gamma_2 \to \Delta_2 \cup \{p\}}{\Gamma_1 \cup \Gamma_2 \to \Delta_1 \cup \Delta_2}$$

Dans le cas d'une clause de Horn, Δ_2 est vide et Δ_1 contient au plus un élément, donc $\Delta_1 \cup \Delta_2$ contient au plus un élément.

Soit S un ensemble de clauses de Horn. On suppose que pour toute variable p, p n'apparaît à la fois positivement et négativement dans aucune clause de S.

Remarquons que comme les clauses contenant simultanément une variable et sa négation sont trivialement satisfaisables, cette restriction n'en est pas une.

Question 8. Soit $\to \{p\}$ une clause positive de S. On note S_p l'ensemble des clauses de S contenant p. Soit $\text{res}^+(S,p)$ l'ensemble des clauses obtenues par une seule règle de coupure entre $\to \{p\}$ et une clause de S qui contient p en position négative. Montrer que S est satisfaisable ssi $(S \setminus S_p) \cup \text{res}^+(S,p)$ est satisfaisable.

Soit v une valuation satisfaisant S. Elle saisfait $(S \setminus S_p)$ car cet ensemble ne contient que des clauses de S. Elle satisfait également $\text{res}^+(S,p)$. En effet, $\text{res}^+(S,p)$ contient des clauses de la forme $\Gamma_1 \to \Delta_1$ où $\Gamma_1 \cup \{p\} \to \Delta_1$ et $\to \{p\}$ apparaissent dans $\mathcal{C}$. Comme v satisfait $\mathcal{C}$, elle satisfait $\to \{p\}$, $v(p) = 1$ et donc, comme v satisfait $\Gamma_1 \cup \{p\} \to \Delta_1$, v satisfait $\Gamma_1 \to \Delta_1$.

Réciproquement, une valaution v' satisfaisant $(S \setminus S_p) \cup \text{res}^+(S,p)$ lorsqu'elle est étendu par $p \mapsto 1$ satisfait S. En effet, les clauses de S sont de 3 formes :

- celles dans $S \setminus S_p$ qui sont dans $(S \setminus S_p) \cup \text{res}^+(S,p)$ donc satisfaitent par v',
- celles où p apparaît positivement, qui sont satisfaitent par $p \mapsto 1$,
- celles où p apparaît négativement, qui sont satisfaitent par $v' \cup \{p \mapsto 1\}$ car se sont des sur-clauses de clauses de $\text{res}^+(S,p)$.

Remarquons que l'hypothèse qu'aucune variable n'apparaît à la fois positivement et négativement dans une clause est utilisée ici, puisque c'est elle qui nous garantit que p n'apparaît pas dans $\text{res}^+(S,p)$ et donc que l'image de p n'est pas fixé par v' et peut être choisie librement.

Question 9. En déduire un algorithme pour la satisfaisabilité d'un ensemble de clauses de Horn $\mathcal{C}$ en temps quadratique en fonction de l'entrée.

On parcourt $\mathcal{C}$ à la recherche d'une clause positive. S'il y en a une, on l'efface, puis on efface toutes ses occurrences négatives et toutes les clauses où elle apparaît positivement. On itère jusqu'à ce qu'il n'y ait plus de clauses positives (au plus une itération par variable). À ce moment là, trois situations peuvent

se présenter :

- L'une des clauses est vide, alors $\mathcal{C}$ n'est pas satisfaisable.
- Il n'y a plus de clauses, alors $\mathcal{C}$ ne contenait pas de clause négative et la valuation constante associant 1 à toutes les variables le satisfait.
- Il reste des clauses, mais aucune n'est positive, $\mathcal{C}$ est satisfait par la valuation associant 1 à toutes les variables sur lesquelles on a coupé durant l'algorithme et 0 à toutes les autres.

Remarquons que si l'on veut vérifier la condition qu'aucune variable n'apparaît à la fois positivement et négativement dans une clause, cela peut se faire avant l'algorithme en temps quadratique également.

La résolution par coupures donne donc un algorithme quadratique pour la satisfaisabilité des clauses de Horn (alors qu'elle donne un algoithme exponentielle pour la satifaisabilité de clauses quelconques).