Correction TD 9

Exercice 1 - Complexité de SUBSET-SUM

SUBSET-SUM est évidemment NP.

Pour montrer qu'il est NP-complet nous allons suivre l'indication et effectuer une réduction à partir de 3-SAT.
Pour ce faire nous allons commencer en transformant les données de 3-SAT en chiffre.

Soit $C_0, \ldots, C_{m-1}$ les clauses et $x_0, \ldots, x_{n-1}$ les variables de 3-SAT. On construit v_i et v'_i de la manière suivante:
$v_i = 10^{m+i} + \sum_{j=0}^{m-1}(b_{ij}10^j)$ et $v'_i = 10^{m+i} + \sum_{j=0}^{m-1}(b'_{ij}10^j)$, $1 \leq i \leq n-1$ avec

$$b_{ij} = \begin{cases} 0 & \text{si } x_i \text{ apparaît dans } C_j \\ 1 & \text{sinon} \end{cases} \quad \text{et } b'_{ij} = \begin{cases} 0 & \text{si } \overline{x_i} \text{ apparaît dans } C_j \\ 1 & \text{sinon} \end{cases}$$

Pour mieux comprendre ce qu'on fait, on peut imaginer qu'on crée des (m+n)-upplets dont les n premières cases il n'y à que des zéros sauf à un endroit i qui permet de savoir de quelle variable on parle, que ce soit x_i ou $\overline{x_i}$, et les m-suivant permettent de savoir si la variable est dans les clauses correspondantes.

Désormais on va sommer certains v_i et certains v'_i, en fait on somme n entier, une par littéral, $T = \sum_{j=0}^{n-1}(w_i)$ où w_i vaut soit x_i soit $\overline{x_i}$. T est un nombre qui commence par n "1" et est suivit de m chiffres compris entre 0 et 3 (comme on est parti de 3-SAT il y a au plus 3 variables par clause et ces m chiffres correspondres exactment au nombre de littéral qui met cet clause à vrai). Si on part d'une solution du problème 3-SAT, et qu'on prend v_i si x_i est vrai et v'_i si x_i est faux, on remarque que c'est m chiffres sont compris entre 1 et 3.

A partir de là on se demande comment faire la différence entre les sous-ensembles donnant un somme qui contient un "0" de ceux donnant une somme sans "0" qui sont exactment les sous-ensembles correspondant à un certificat du problème 3-SAT par la bijection évident $v_i \in$ S $\leftrightarrow x_i$ est vrai.
C'est à cela que servent les $s_j = 10^j$ et $s'j = 2.10^j$. En prenant $t = 1 \ldots 14 \ldots 4$ et en ne réfléchissant que sur les m chiffres de droite de T on voit que:
-à partir de 3 on peut avoir 4,3 + 1
-à partir de 2 on peut avoir 4,2 + 2
-à partir de 1 on peut avoir 4,1 + 1 + 2
-à partir de 0 on ne peut avoir 4.
(Cette réflection est la même qui si on considère les nombres comme des n+m-upplets)

Le problème 3-SAT est donc réduit en SUBSET-SUM avec $S = \{v_i | 1 \leq i \leq n-1\} \cup \{v'_i | 1 \leq i \leq n-1\} \cup \{s_j | 1 \leq j \leq m-1\} \cup \{s_j | 1 \leq j \leq m-1\}$ et $t = 1 \ldots 14 \ldots 4$

On a ainsi, que si on trouve une solution du problème 3-SAT, on a une solution du problème SUBSET-SUM en prenant la bijection precédente, et en completant avec les $s_j = 10^j$ et $s'j = 2.10^j$. Réciproquement, si on trouve un sous-ensemble S' de S tels que si $\sum_{x \in S'}(x) = t = 1 \ldots 14 \ldots 4$, en prenant $\tilde{\text{S}} = S \cap (\{v_i | 1 \leq i \leq n-1\} \cup \{v'_i | 1 \leq i \leq n-1\})$ on a nécessairement $T = \sum_{x \in S'}(x)$ qui est nombre commencant par n "1" et suivit de m chiffre compris entre 1 et 3 (ni 0 car sans les "s" on perd au plus 3 car 4 ne sont plus car en sans les "s", on atteind au mieux 3), ce qui montre que comme les aucun "0" n'apparrait, les instances de 3-SAT peuvent toutes êtres mises à vrai, trouvé à l'aide de la bijection citée plus haut, donc à partir du certificat de ce problème SUBSET-SUM on retrouve un certificat du problème 3-SAT de départ.

La réduction précédente est polynomiale car on passe de n variables et m clauses à 2n+2m entiers de m chiffres, SUBSET-SUM est donc NP-complet.

Remarques: - On peut faire le même travail dans toutes les autres bases qui n'introduisent pas de problèmes dues aux phénomènes de retenue.
-On peut aussi remplacer les "4" dans t par nimporte quel autre chiffre supérieur à 4, à condition de rajouter des "s" en nombre suffisant. -On peut aussi faire des $\tilde{s_i} = 10^{m+i}$ et sans changer t, ça remplace les variables dont pour lesquels peut choisir n'importe que booléen.

Exercice 2 - Variantes de 3-SAT
1. 3-SAT NAE

Evidemment 3-SAT NAE est NP.

Pour montrer qu'il est NP-complet, et contrairement à toute attente, nous allons effectuer une reduction à partir de 3-SAT de la manière suivante:

Pour chaque clause, $a \wedge b \wedge c$ du problème 3-SAT, on créé les instances $a \wedge b \wedge x$ et $c \wedge \overline{x} \wedge f$ où f est global (ie f est la même pour chaque clause créée ainsi)

S'il existe une instanciation des variables qui met toutes les clauses à vrai, il en existe aussi une pour sa réduction, pour cela, il suffit de prendre $x = \overline{a \cup b}$, avec les notations ci-dessus, de plus si a et b sont à vrai, x sera à faux, et donc pour la seconde clause sera mis a vrai par $\overline{x}$, on fini en prenant f toujours à faux, on a donc une bonne instanciation pour 3-SAT NAE.

Réciproquement, si on a un bonne instanciation du problème 3-SAT NAE (réduction du problème 3-SAT) il existe un bonne instanciation pour le problème 3-SAT d'origine. En effet si f est à faux soit c est à vrai et donc la clause $a \wedge b \wedge c$ est vrai, soit c est à faux donc $\overline{x}$ est à vrai, x est donc à faux, d'où soit a soit b est à vrai et donc la clause $a \wedge b \wedge c$ est à vrai, par contre si f est à vrai, on prend toutes les variables du problème NAE et on prend leur négation. Cette instanciation, met toujours NAE à vrai, car comme “not all equal” une variable fausse au moins par clause qui est à vrai, et on recommence comme ci-dessus.

Ce qui précède montre que tout problème 3-SAT peut se réduire polynomialement (on multiplie le nombre de clause par deux, et on rajoute une variable plus une par clauses de départ) en un problème 3-SAT NAE équivalent, 3-SAT NAE est donc NP-complet.

2. 3-SAT OIT

On fait comme le cas précédent, sauf qu'à partir de $a \wedge b \wedge c$ on créé les trois clauses $a \wedge x \wedge y$, $\overline{b} \wedge x \wedge x'$ et $\overline{c} \wedge y \wedge y'$

Pour commencer on va montrer que s'il existe une bonne instanciation pour le problème 3-SAT de départ, il en existe une pour sa réduction à 3-SAT OIT.

C'est assez simple à montrer une fois qu'on connait la transformation, par contre elle est un peu longue à trouver. En effet, pour chaque clause de $a \wedge b \wedge c$ 3-SAT , si a est à vrai on met les variables x et y (avec les notations précédentes) à faux, et on donne la valeur de b à x' et celle de c à y' (a, b et c restent inchangées d'un problème à l'autre); et si a est à faux, alors soit b, soit c est à vrai. Comme la construction est symétrique, on supposera que b est à vrai, x et x' sont donc mis à faux, donc y doit être mis à vrai, et y' à faux. On a alors exactement une variable à vrai dans les trois instances, ça se passe comme ça dans chaque instance, et vu qu'on ne touche à aucun des variables directment associées à celle du problème 3-SAT, il n'y a pas de problème.

Dans l'autre sens, on va montrer que si on a une instanciation qui permet de répondre oui au problème 3-SAT OIT obtenue à partir d'un problème 3-SAT par la contruction explicitée plus haut alors on en a un pour ce problème 3-SAT. En effet on ne peut avoir aucune clause $a \wedge b \wedge c$ à faux car cela signifie que a, b et c sont faux, ce qui signifie que dans les clauses du 3-SAT OIT équivalentes on a x et x' à faux à cause de $\overline{b}$ (dans la deuxième clause), et de même, on a y et y' à faux à cause de $\overline{c}$ (dans la troisième clause), ce qui donne que la première clause est fausse, il y a donc contradiction.

On a donc équivalence entre entre les deux problèmes. Il y a un certificat pour un problème 3-SAT si et seulement il y en a une pour sa réduction vue précédement, en un problème 3-SAT OIT. De plus la réduction est polynomiale (on multiplie le nombre de clause par trois, et on rajoute six variables par clauses de départ).
Ce qui précède montre que tout problème 3-SAT peut se réduire polynomialement en un problème 3-SAT OIT équivalent, 3-SAT OIT est donc un problème NP-complet.

Remarque: une fois qu'on à montré que 3-SAT OIT est NP-complet, le problème 1 deviens plus facile, car on démontre que SUBSET-SUM est NP-complet en faisant la même construction, sans les s_i et s_i', et en prenat t=1...11 ...1

Exercice 3 - SAT-n
1. SAT-3

SAT-3 est trivialement NP.

Pour montrer qu'il est NP-complet, on d'un problème de SAT et pour chaque littéral x_i apparaissant k fois avec k avec $k > 3$, on applique l'algorithme suivant :
on remplace les occurences de x_i par k nouveaux littéraux $y_{i_1}, \ldots, y_{i_k}$, et on rajoute les clauses $y_{i_1} \wedge \overline{y_{i_2}}, \ldots, y_{i_{k-1}} \wedge \overline{y_{i_k}}$ et $y_{i_k} \wedge \overline{y_{i_1}}$.

Les nouveaux littéraux apparaissent alors 3 fois chacun (une fois à la place de x_i, et deux fois dans les nouvelles clauses). De plus les nouvelles clauses entrainent que les y_{i_j} prennent tous la même valeur : si y_{i_j} est vrai, alors $y_{i_{j-1}}$ egalement, et inversement, si y_{i_j} est faux, alors $y_{i_{j+1}}$ doit l'être aussi (en toute rigueur on devrait rajouter des modulos k), et comme les clauses sont cycliques, on obtient que tous ces littéraux ont la même valeur.

Enfin, si n est le nombre de littéraux, et m le nombre de clauses, le nouveau problème obtenu comporte au plus nm littéraux et (m+nm) clauses, donc une instance de SAT se réduit polynomialement en une instance SAT-3 équivalente, et comme SAT-3 est NP, SAT-3 est NP-complet.

Et comme pour tout $n \geq 3$, SAT-3 est un cas particulier de SAT-n, SAT-n est NP-complet.

2. SAT-2

Soit x une variable d'un système de clauses F de SAT-2, on transforme F selon les cas de sorte que le système obtenue soit satisfiable si et seulement si le système initial l'était :

- si x (ou $\overline{x}$) n'apparait qu'une seule fois, on supprime la clause dans laquelle il apparait, car on est libre de mettre x (ou $\overline{x}$), donc toute la clause à vrai.

- si x (ou bien $\overline{x}$) apparait 2 fois dans une même clause, ou dans 2 clauses distinctes, on supprime la ou les clauses, pour la même raison.

- si x et $\overline{x}$ apparaissent dans la même clause, on supprime la clause car elle est toujours à vrai.

- si x et $\overline{x}$ apparaissent dans 2 clauses distinctes, $x \vee C_1$ et $\overline{x} \vee C_2$, alors le système de clause est satisfiable si et seulement si au moins des deux clause C_1 ou C_2 peut être mis à vrai en même temps que les autres clause (pour l'autre on choisit x pour compléter), ainsi:

$\rightarrow$ si $C_1 = C_2 = \emptyset$ alors le système n'est pas satisfiable.

$\rightarrow$ sinon on remplace les 2 clauses par la nouvelle clause $C_1 \vee C_2$.

On a ainsi défini un algorithme en n étapes (où n est le nombre de littéraux), pour lequel on décide à chaque étape de la valeur d'un littéral avec une complexité m (où m est le nombre de clauses) : <u>on peut donc résoudre SAT-2 en $\bigcirc(nm)$</u>.