Devoir : $\neq$SAT

Remy Detobel

18 décembre 2017

1 Introduction

Nous devons prouver que $\neq$SAT est NP-Complet.

1.1 Définition de $\neq$SAT

Prenons ϕ comme étant une formule sous la forme normale conjonctive de trois littéraux. Une assignation des variables de ϕ est correcte si deux littéraux ont deux valeurs différentes. En d'autres mots, les trois littéraux ne peuvent pas avoir la même valeur. $\neq$SAT est une collection de ce genre de formule. Nous appellerons ces formules sous forme normale conjonctive : "$\neq$assignement".

1.2 Définition de NP-Complet

Un langage B est "NP-complet" si et seulement si :
— $B \in NP$
— Pour chaque $A \in NP, A \leq_p B$
Dans le cas présent, nous devons donc prouver que $\neq$SAT est dans NP et également que pour un problème A étant lui même NP on peut écrire : $A \leq_p \neq$SAT. C'est à dire qu'il existe une fonction f s'exécutant en un temps polynomial tel que :

$$\forall w, w \in A \Leftrightarrow f(w) \in B$$

2 Réduction

Nous allons donc réduire 3SAT en $\neq$SAT.

2.1 Réduction de 3SAT

Pour se faire nous allons modifier les clauses du problème 3SAT. Notons $x_{i,j}$ un littéral appartenant à la $i^{\text{ème}}$ clause et étant le $j^{\text{ème}}$ élément de cette clause (variant de 1 à 3 pour 3SAT). La clause i peut donc être écrite comme étant :

$$(x_{i,1} \vee x_{i,2} \vee x_{i,3})$$

Comme indiqué, nous voulons éviter que chaque littéral soit égal à $True$. Pour ce faire nous allons rajouter une variable C_i et une constante $False$. Nous pouvons donc écrire :

$$\begin{gathered}(x_{i,1} \vee x_{i,2} \vee C_i)\\ \wedge\ (x_{i,3} \vee C_i \vee False)\end{gathered}$$

Où $C_i = \big(x_{i,1} \vee x_{i,2} \vee x_{i,3}\big) \wedge \big(\overline{x_{i,1}} \vee \overline{x_{i,2}}\big)$.
En effet, on veut que l'assignation soit fausse si $x_{i,1}, x_{i,2}$ et $x_{i,3}$ sont faux car cette assignation serait fausse également pour le problème 3SAT. Cette première phrase justifie la condition dans la première parenthèse. En effet, on peut écrire :

$$\begin{aligned}\overline{C_i} &= (\overline{x_{i,1}} \wedge \overline{x_{i,2}} \wedge \overline{x_{i,3}})\\ \Leftrightarrow C_i &= x_{i,1} \vee x_{i,2} \vee x_{i,3}\end{aligned}$$

La seconde condition permet de vérifier que lorsque $x_{i,1}$ et $x_{i,2}$ sont à $True$, C_i soit bien à $False$ pour éviter que les trois littéraux ne soient à $True$, on peut donc écrire :

$$\begin{aligned}\overline{C_i} &= x_{i,1} \wedge x_{i,2}\\ \Leftrightarrow C_i &= \overline{x_{i,1}} \vee \overline{x_{i,2}}\end{aligned}$$

2.1.1 Exemple de réduction d'un problème 3SAT

Un problème 3SAT peut donc être écrit de la manière suivante :

$$\begin{aligned} &(x_1 \vee x_2 \vee x_3) \\ \wedge\ &(\overline{x_1} \vee x_2 \vee x_3) \\ \wedge\ &(\overline{x_1} \vee \overline{x_2} \vee x_3) \end{aligned}$$

Une solution possible est : $x_1 = True, x_2 = True$ et $x_3 = True$. Appliquons donc la réduction :

$$\begin{aligned} &(x_1 \vee x_2 \vee C_1) \\ \wedge\ &(x_3 \vee C_1 \vee False) \\ \wedge\ &(\overline{x_1} \vee x_2 \vee C_2) \\ \wedge\ &(x_3 \vee C_2 \vee False) \\ \wedge\ &(\overline{x_1} \vee \overline{x_2} \vee C_3) \\ \wedge\ &(x_3 \vee C_3 \vee False) \end{aligned}$$

Où $C_i = (x_{i,1} \vee x_{i,2} \vee x_{i,3}) \wedge (\overline{x_{i,1}} \vee \overline{x_{i,2}})$.
Donc pour $x_1 = True, x_2 = True$ et $x_3 = True$, on aura :

$$\begin{aligned} C_1 &= (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2}) \\ &= (True \vee True \vee True) \wedge (\overline{True} \vee \overline{True}) \\ &= (True \vee True \vee True) \wedge (False \vee False) \\ &= False \end{aligned}$$

$$\begin{aligned} C_2 &= (\overline{x_1} \vee x_2 \vee x_3) \wedge (\overline{(\overline{x_1})} \vee \overline{x_2}) \\ &= (False \vee True \vee True) \wedge (\overline{False} \vee \overline{True}) \\ &= (False \vee True \vee True) \wedge (True \vee False) \\ &= True \end{aligned}$$

$$\begin{aligned} C_3 &= (\overline{x_1} \vee \overline{x_2} \vee x_3) \wedge (\overline{(\overline{x_1})} \vee \overline{(\overline{x_2})}) \\ &= (False \vee False \vee True) \wedge (\overline{False} \vee \overline{False}) \\ &= (False \vee False \vee True) \wedge (True \vee True) \\ &= True \end{aligned}$$

On a donc $C_1 = False, C_2 = True$ et $C_3 = True$.

Si l'on considère une autre solution : $x_1 = False,\ x_2 = True$ et $x_3 = False$, on aura : $C_1 = True,\ C_2 = False$ et $C_3 = True$.

2.2 $\neq$SAT est un problème NP

Un problème appartient à NP s'il existe un algorithme permettant de vérifier les solutions en un temps polynomial par rapport aux nombres d'entrées. Pour vérifier qu'une solution assignant n variables soit à T, soit à F, il suffit de calculer chacune des clauses. Le nombre étant fixe pour un problème donné, cette vérification se fait en $O(n)$. Il s'agit donc bien d'un problème appartenant à NP.

3 Conclusion

En prouvant qu'il existe une réduction de 3SAT vers $\neq$SAT en un temps polynomial et en montrant que $\neq$SAT était bien un problème appartenant à NP, on a bien montré que $\neq$SAT est NP-Complet.