TP N°6
Le problème SAT
Illustration du Non-déterminisme Polynomial

Le but de ce TP est de montrer que le Problème SAT (SATisfiabilité d'une expression booléenne) appartient à la classe de complexité NP, la classe des problèmes de décision non déterministes polynômiaux. Il faut pour ce faire trouver un algorithme de validation de complexité polynômiale pour ce problème SAT.

Le problème SAT est défini comme suit :

- Une proposition atomique est une variable booléenne, c'est-à-dire prenant ses valeurs dans l'ensemble BOOL= {VRAI, FAUX}.
- Un littéral est une proposition atomique ou la négation d'une proposition atomique.
- Une proposition atomique est aussi appelée littéral positif ; et la négation d'une proposition atomique est appelé littéral négatif.
- Une clause est une disjonction (somme logique ou (or)) de littéraux.

Etant données m propositions atomiques $p_1, \ldots, p_m$, une instanciation du m-uplet ($p_1, \ldots$, pm) est un élément de {VRAI, FAUX}. Une instanciation ($e_1, \ldots, e_m$) de (p_1, …, pm) satisfait une clause c (notée (e_1, …, em) c) si et seulement si l'une des conditions suivantes est satisfaite :

1- il existe i Є {1, …, m} tel que (e_i = VRAI) et (p_i occurre ou existe dans c) ;
2- il existe i Є {1, …, m} tel que (e_i = FAUX) et ($\neg p_i$ occurre ou existe dans c).

Une instanciation satisfait une conjonction (produit logique et (and)) de clauses si et seulement si elle satisfait chacune de ses clauses. Une conjonction de clauses est satisfiable si et seulement s'il existe une instanciation la satisfaisant. Une instanciation satisfaisant une conjonction est dite solution ou modèle de la conjonction.

Le problème SAT est maintenant défini comme suit :

- **Entrée** : une conjonction C de n clauses construites à l'aide de m propositions atomiques p_1, …, pm ;
- **Sortie** : la conjonction C est-elle satisfiable ?

1. Proposer une structure de données permettant de représenter une instance du problème SAT.

2. Développer l'algorithme de validation, noté validation, sous forme d'une fonction booléenne dont il est important d'expliquer les paramètres.
3. Calculer le nombre d'opérations élémentaires de l'algorithme dans le pire cas.
4. Montrer que la complexité de l'algorithme est polynômiale.
5. Développer l'algorithme de résolution du problème SAT qui cherche une solution pour ce problème.
6. Calculer la complexité de cet algorithme.
7. Développer les programmes correspondants et mesurer les temps d'exécutions pour un échantillon de valeurs de n et m.