Algorithme recherche locale

***Question 1 : Donner le pseudo-code de l’algorithme recherche locale ..***

**Require : une instance SAT F**

**Ensure : Renvoyer une valuation qui satisfait F si l’algorithme termine**

**Loop**

**T :=une valuation tirée aléatoirement**

**For i := 1 to Maxiter do**

**if T satisfait F then**

**return T**

**else**

**C := une clause de F aléatoirement choisie non satisfaite par T**

**if rand(0, 1) < p then**

**Flipper dans T une variable aléatoirement choisie dans C**

**Else**

**Flipper dans T une variable dans C qui minimise le nombre de clauses rendues fausses après le flip**

**end if**

**end if**

**end for**

**end loop**

***Question 2 : Expliquer l’utilité des différents paramètres de l’algorithme ?***

L’algorithme commence par choisir aléatoirement une interprétation complète T ,if T satisfait donc il va retourner T sinon, il choisit aléatoirement une clause de F c non satisfaite par T .Un mécanisme de random restart est ajouté à la boucle donnée ,une variable ∈ c qu’il flippe de manière à améliorer le nombre de clauses satisfaites jusqu’à ce qu’une solution (une interprétation satisfaisant toutes les clauses) soit trouvée ou qu’un nombre maximum donné de flippes, soit atteint.

***Question 3 : Comment peut-on appliquer la recherche locale pour résoudre votre problème d’optimisation?***

Ce processus est répété pour un nombre donné de relances (Maxiter). Ce système de relances peut être vu comme un mécanisme de diversification dans le sens où il réduit l’impact de l’interprétation aléatoire initiale sur l’efficacité de la recherche.

Les variantes de WalkSat diffèrent principalement de choix des variables à flipper se basent sur deux compteurs : break et make qui calculent, pour une variable, le nombre de clauses qui seront respectivement falsifiées et satisfaites si le variable ∈ c est flippée. La version originale de WalkSat, une fois la clause falsifiée c sélectionnée, choisit la variable à flipper aléatoirement parmi les variables de c qui ont une valeur de break nulle s’il en existe. Sinon, elle choisit aléatoirement une variable de c avec la plus petite valeur de break.

Donc on conclure que le walksat commencent par attribuer une valeur aléatoire à chaque variable de la formule. Si l'affectation satisfait toutes les clauses, l'algorithme se termine et renvoie l'affectation. Sinon, une variable est inversée et ce qui précède est ensuite répété jusqu'à ce que toutes les clauses soient satisfaites.

Expression du random restart