Boolean Satisfiability and the DPLL Algorithm

Léo Leesco

Jeudi 5 février 2026

Table des matières

SAT Solver

Choix de modélisation

Pour des raisons purement pratiques, je préfère utiliser les idées du sujet sans utiliser directement les implémentations proposées. Ce faisant, je "généralise" un peu les notations, en espérant qu'elles me permettent de prouver les résultats attendus :

- je ne modifie pas en place l'`assignment`, ce qui me permet d'adopter une solution purement fonctionnelle, que j'espère plus simple à prouver
- on ne restreint plus explicitement les clauses à exactement 3 littéraux :
  - ceci permet de terminer immédiatement l'algorithme lorsqu'on a une clause vide ; on doit satisfaire une union vide de litéraux, et étant donné que l'élément neutre pour l'union est $\bot$
  - lorsqu'on n'a plus de clauses à traiter, de même que dans le cas de l'union, on se trouve dans une intersection vide, qui est donc vraie.

DIMACS Format

J'adapte le parser proposé, qui me fournit un résultat extrêmement proche de celui que j'attends : je produis simplement un `list clause` au lieu d'un `array clause`.

DPLL Algorithm

L'algorithme est légèrement différent que celui proposé. Évidemment, je me fonde sur les mêmes idées.

Élimination d'un `literal` : `subst`

Dans l'algorithme "de base", on teste toutes les assignations possibles, avec quelques court-circuits. Cela signifie que, pour un `literal` donné, on va essayer de trouver un `assignment` (récursivement) qui satisfait le reste de la formule une fois substitué le `literal` par `true` ou `false`.

Dans chaque `clause`, on regarde s'il existe le `literal` qui est de la bonne positivité [1] :

- soit il est de même positivité et alors la clause est trivialement satisfaite et on peut simplement la supprimer de la liste
- soit elle est de positivité opposée, et on retire le litéral de la clause (car il contribue, dans cet `assignment`, nécessairement comme `false`)

Backtracking : `dpll`

Pour récupérer l'`assignment` correspondant, on retourne récursivement l'`assignment` convenant après substitution, complémentée par la valeur du `literal` qui convient.

Naturellement, si `dpll` échoue, l'`assignment` retourné n'a pas de sens (comme attendu).

L'`assignment` correspond alors à une `list` contenant les `literal` avec la positivité calculée. Il y a plusieurs invariants associés à l'`assignment` résultant :

- il n'y a pas de doublon, c'est-à-dire qu'on n'a jamais `i` et `-i`
- l'`assignment` satisfait la `list clause` passée en argument

Pour simplifier les choses, `dpll` renvoie un `option assignment`. Assez naturellement, `None` est renvoyé qui un `assignment` ne peut pas satisfaire la `list clause` passée en argument, et `Some assignment` sinon.

1. par exemple, si on regarde x_1, x_1 est de même positivité que `true`, et de positivité opposée à `false` ; réciproquement $\neg x_1$ est de même positivité que `false` et de positivité opposée à `true`