Mise sous forme clausale

On chercher à mettre une formule φ sous forme clausale, c'est-à-dire à trouver ψ sous forme clausale, t.q. $\psi \equiv \varphi$. Pour cela, on suit l'algorithme suivant, en répétant chaque étape tant que l'on peut le faire :

1. On élimine les implications à l'aide de la substitution $p \Rightarrow q \rightarrow \neg p \vee q$
2. Supprimer les négations aux niveaux les plus hauts à l'aide des substitutions
   - $\neg\neg p \rightarrow p$
   - $\neg(p \vee q) \rightarrow \neg p \wedge \neg q$
   - $\neg(p \wedge q) \rightarrow \neg p \vee \neg q$
3. Transformer une FND en FNC, en utilisant la substitution $p \vee (q \wedge r) \rightarrow (p \vee q) \wedge (p \vee r)$

Exemple

Table of Contents

Définition

Le **problème SAT** est le problème de décision qui consiste à déterminer si une formule propositionnelle φ donnée en entrée admet ou non un modèle.

Définition

Le **problème SAT** est le problème de décision qui consiste à déterminer si une formule propositionnelle φ donnée en entrée admet ou non un modèle.

Définition

Un problème de décision est **décidable** s'il existe un algorithme qui le décide (répond par oui ou non à la question formulée par le problème en un nombre fini d'étapes).

Définition

Le **problème SAT** est le problème de décision qui consiste à déterminer si une formule propositionnelle φ donnée en entrée admet ou non un modèle.

Définition

Un problème de décision est **décidable** s'il existe un algorithme qui le décide (répond par oui ou non à la question formulée par le problème en un nombre fini d'étapes).

Proposition

Définition

Le **problème SAT** est le problème de décision qui consiste à déterminer si une formule propositionnelle φ donnée en entrée admet ou non un modèle.

Définition

Un problème de décision est **décidable** s'il existe un algorithme qui le décide (répond par oui ou non à la question formulée par le problème en un nombre fini d'étapes).

Proposition

Le problème SAT est décidable

Définition

Le **problème SAT** est le problème de décision qui consiste à déterminer si une formule propositionnelle φ donnée en entrée admet ou non un modèle.

Définition

Un problème de décision est **décidable** s'il existe un algorithme qui le décide (répond par oui ou non à la question formulée par le problème en un nombre fini d'étapes).

Proposition

Le problème SAT est décidable

Algorithme de Quine

On remarque que l'ensemble des clauses C_i et la formule $\bigcap C_i$ admettent les mêmes modèles. Un algorithme de résolution du problème SAT peut ainsi manipuler une formule sous forme normale conjonctive comme sous forme d'ensemble de clauses.

Algorithme de Quine

On remarque que l'ensemble des clauses C_i et la formule $\bigcap C_i$ admettent les mêmes modèles. Un algorithme de résolution du problème SAT peut ainsi manipuler une formule sous forme normale conjonctive comme sous forme d'ensemble de clauses. L'algorithme de Quine parcourt l'arbre de Herbrand, arbre binaire dont les feuilles sont les valuations, qui a chaque étage, associe une valeur à l'une des variables, différente, selon si le noeud est fils gauche ou droit. Dès qu'une variable est affectée à la valeur $x \in \{0, 1\}$, le calcul se fait récursivement avec $C[p \leftarrow \perp]$ si $x = 0$ et avec $C[p \leftarrow \top]$ si $x = 1$.
L'algorithme prend en entrée un ensemble de clauses C et renvoie **vrai** si C est satisfaisable, et **faux** sinon.

Algorithme de Quine

- simplification de l'ensemble des clauses
- si $C = \emptyset$ renvoyer **vrai**
- si C contient la clause $\perp$ renvoyer **faux**
- choisir une variable propositionnelle p apparaissant dans une clause
- simplifier C :
  - si une clause disjonctive contient $\top$, la supprimer
  - si une clause subsume une autre (inclusion des littéraux de la première dans la seconde), supprimer la seconde
  - si une clause contient les littéraux p et $\neg p$, la remplacer par $\top$
  - si une clause contient une répétition du même littéral, on n'en laisse qu'une occurrence

  - si Quine($C[p \leftarrow \perp]$) =**vrai**, renvoyer **vrai**
  - sinon renvoyer Quine($C[p \leftarrow \top]$)

Problème $n - SAT$

La classe des problèmes SAT contient les sous-classes des problèmes $n - SAT$, lesquels sont ceux de la satisfaisabilité d'une formule sous forme clausale dont les clauses sont d'ordre n (chaque clause est une disonction d'au plus n littéraux).
Le problème $2 - SAT$ est résoluble en temps polynomial ; le problème $3 - SAT$ est aussi "difficile" que SAT.
Le problème HORN-SAT s'intéresse à la satisfaisabilité des disjonctions de clauses comportant au plus un littéral positif (clauses de Horn). Il est résolvable en temps polynomial.

Algorithme DPLL

L'**algorithme de Davis-Putnam-Logemann-Loveland** procède, comme l'algorithme de Quine, à une recherche par **backtracking**, en tirant profit de deux règles supplémentaires :

- Si une clause est unitaire (ne contient qu'un littéral), la valeur de la variable propositionnelle concernée est imposée
- Si un littéral n'apparaît dans l'ensemble des clauses que positivement (ou que négativement), on dit qu'il est pur. On peut ici aussi choisir la valuation de la variable concernée, et donc supprimer des clauses ce littéral.

Les littéraux à traiter à chaque étape sont déterminer par des **heuristiques**.
Par exemple, l'heuristique "Maximum Occurrences in Clauses of Minimum Size" (MOM) sélectionne la variable ayant le plus d'occurrences dans les clauses de plus petites tailles, par maximisation de la fonction
$(f*(x)+f*(\neg x))\times 2^k+f*(x)\times f*(\neg x)$, ou $f*(x)$ est le nombre d'occurrences du littéral x dans les clauses les plus courtes non encore satisfaites, avec k lui-même choisi de manière heuristique. Le choix de cette variable favorise la propagation unitaire.