SAT: le problème de la satisfaisabilité

Lionel Vaux Auclair

I2M, université d'Aix-Marseille

Logique HUGO@Polytech, 2021–2022

Formes normales

Si on prend les connecteurs $\wedge$, $\vee$ et $\neg$, on peut *pousser les négations jusqu'aux atomes*.

Théorème

Toute formule propositionnelle est équivalente à une formule de la forme :

$$P, Q ::= X \mid \neg X \mid P \wedge Q \mid P \vee Q$$

Démonstration: Il suffit de vérifier (inductivement) que si P est de cette forme, alors $\neg P$ peut s'y ramener (par les lois de De Morgan). □

On dit que P est une forme normale.

Formes normales

Si on prend les connecteurs $\wedge$, $\vee$ et $\neg$, on peut *pousser les négations jusqu'aux atomes*.

Théorème

Toute formule propositionnelle est équivalente à une formule de la forme :

$$P, Q ::= X \mid \neg X \mid P \wedge Q \mid P \vee Q$$

Démonstration: Il suffit de vérifier (inductivement) que si P est de cette forme, alors $\neg P$ peut s'y ramener (par les lois de De Morgan). □

On dit que P est une forme normale.

$\rightsquigarrow$ Essayez avec $\neg((A \wedge (B \vee C)) \vee \neg(D \vee E))$?

Formes normales

Si on prend les connecteurs $\wedge$, $\vee$ et $\neg$, on peut *pousser les négations jusqu'aux atomes*.

Théorème

Toute formule propositionnelle est équivalente à une formule de la forme :

$$P, Q ::= X \mid \neg X \mid P \wedge Q \mid P \vee Q$$

Démonstration: Il suffit de vérifier (inductivement) que si P est de cette forme, alors $\neg P$ peut s'y ramener (par les lois de De Morgan). □

On dit que P est une forme normale.

$\rightsquigarrow$ Essayez avec $\neg((A \wedge (B \vee C)) \vee \neg(D \vee E))$?

$\rightsquigarrow$ Est-ce que ça change beaucoup la taille de la formule ?

Clauses

- Un littéral est un atome X ou un atome nié $\neg X$.
- Une clause disjonctive est une forme normale sans $\wedge$:
  autrement dit, c'est une disjonction de littéraux.
  Ex : $X \vee \neg Y$.
- Une clause conjonctive est une forme normale sans $\vee$:
  autrement dit, c'est une conjonction de littéraux.
  Ex : $\neg X \wedge Y$.
- Il y a des formules qui ne sont équivalentes à aucune clause : par ex. $X \vee (Y \wedge Z)$.

Formes normales disjonctives (FND)

- Une forme normale disjonctive est une disjonction de clauses conjonctives.
  Ex : $X \vee (Y \wedge Z)$.

Théorème

Toute formule propositionnelle peut être mise en FND.

Démonstration: Par distributivité. □

$\rightsquigarrow$ Essayez avec $(A \vee (\neg B \wedge \neg C)) \wedge (D \vee E)$?

Formes normales conjonctives (FNC)

- Une forme normale conjonctive est une conjonction de clauses disjonctives.
  Ex : $(X \vee Y) \wedge (X \vee Z)$.

Théorème

Toute formule propositionnelle peut être mise en FNC.

Démonstration: Par distributivité. □

$\rightsquigarrow$ Essayez avec $(A \vee (\neg B \wedge \neg C)) \wedge (D \vee E)$?

Formes normales conjonctives (FNC)

- Une forme normale conjonctive est une conjonction de clauses disjonctives.
  Ex : $(X \vee Y) \wedge (X \vee Z)$.

Théorème

Toute formule propositionnelle peut être mise en FNC.

Démonstration: Par distributivité. □

$\rightsquigarrow$ Essayez avec $(A \vee (\neg B \wedge \neg C)) \wedge (D \vee E)$?

Ça peut faire grossir les formules : à quel point ?

Facile

- Tester si une FND est satisfaisable (= tester si l'une des clauses l'est)
- Tester si une FNC est valide (= tester si chaque clause l'est)

$\rightsquigarrow$ Comment ?

Facile

- Tester si une FND est satisfaisable (= tester si l'une des clauses l'est)
- Tester si une FNC est valide (= tester si chaque clause l'est)

$\rightsquigarrow$ Comment ?

- Tester si une FNC est falsifiable (= tester si l'une des clauses l'est)
- Tester si une FND est contradictoire (= tester si chaque clause l'est)

$\rightsquigarrow$ Comment ?

Difficile

- Tester si une FNC est satisfaisable
- Tester si une FND est valide
- Tester si une FND est falsifiable
- Tester si une FNC est contradictoire

Difficile

- Tester si une FNC est satisfaisable
- Tester si une FND est valide
- Tester si une FND est falsifiable
- Tester si une FNC est contradictoire

C'est le problème **SAT**.

La classe *NP*

SAT est un problème très *générique*.
Si on savait le résoudre efficacement (= en temps polynomial), on résoudrait aussi efficacement tout problème de la forme

> *Étant donnée une entrée x de taille n existe-t-il une donnée a de taille $p(n)$ telle que $Q(x, a)$?*

où p est un polynome et Q se calcule en temps polynomial.

C'est la classe *NP* : on dit que **SAT** est *NP*-complet.

La classe *NP*

SAT est un problème très *générique*.
Si on savait le résoudre efficacement (= en temps polynomial), on résoudrait aussi efficacement tout problème de la forme

> *Étant donnée une entrée x de taille n existe-t-il une donnée a de taille $p(n)$ telle que $Q(x, a)$?*

où p est un polynome et Q se calcule en temps polynomial.

C'est la classe *NP* : on dit que **SAT** est *NP*-complet.
La question $P = NP$? est une des questions principales de l'informatique théorique depuis des décennies.

Exemples de problèmes dans *NP*

- étant donné un graphe, est-ce qu'on peut le colorier avec 3 couleurs (deux sommets voisins étant de couleurs différentes) ?
- étant données une liste de cours, chacun avec une durée et un enseignant attribuer, et une liste de salles, est-ce qu'on peut construire un emploi du temps ?
- étant donnés un nombre de processeurs et une liste de tâches, chacune prenant un temps fixé, peut-on ordonnancer les tâches pour les réaliser en un temps donné ?
- étant donnés une liste de logiciels à installer, leurs dépendances et incompatibilités, peut-on résoudre les contraintes ?

Résoudre **SAT** : *à la main*

Pour une formule à n variables :

En testant toutes les dvv

- il y en a 2^n
- sur un processeur, on fait de l'ordre de $10^9 = 2^{30}$ opérations/seconde
- dans une journée il y a entre 2^{16} et 2^{17} secondes
- avec 50 variables, il faut plusieurs jours

Résoudre **SAT** : *à la main*

Pour une formule à n variables :

En testant toutes les dvv

- il y en a 2^n
- sur un processeur, on fait de l'ordre de $10^9 = 2^{30}$ opérations/seconde
- dans une journée il y a entre 2^{16} et 2^{17} secondes
- avec 50 variables, il faut plusieurs jours

⇝ Tester

Résoudre **SAT** : Davis–Putnam–Logemann–Loveland

Soit F une FNC, on démarre avec une dvv d inconnue :

- s'il n'y a plus de clauses ($F \equiv \top$) : on renvoie la valeur courante de d ;
- si une clause est vide : F est contradictoire ;
- si on prend une variable x sur laquelle d n'est pas encore définie :
  - on fixe $d(x) = 0$ et on recommence ;
  - si aucun résultat n'a été retourné, on fixe $d(x) = 1$ et on recommence.

Résoudre **SAT** : Davis–Putnam–Logemann–Loveland

Soit F une FNC, on démarre avec une dvv d inconnue :

- s'il n'y a plus de clauses ($F \equiv \top$) : on renvoie la valeur courante de d ;
- si une clause est vide : F est contradictoire ;
- si on prend une variable x sur laquelle d n'est pas encore définie :
  - on fixe $d(x) = 0$ et on recommence ;
  - si aucun résultat n'a été retourné, on fixe $d(x) = 1$ et on recommence.

Étonnamment c'est efficace *en pratique*.

$\rightsquigarrow$ `http://www.satcompetition.org/`

Résoudre **SAT** : Davis–Putnam–Logemann–Loveland

Soit F une FNC, on démarre avec une dvv d inconnue :

- s'il n'y a plus de clauses ($F \equiv \top$) : on renvoie la valeur courante de d ;
- si une clause est vide : F est contradictoire ;
- si on prend une variable x sur laquelle d n'est pas encore définie :
  - on fixe $d(x) = 0$ et on recommence ;
  - si aucun résultat n'a été retourné, on fixe $d(x) = 1$ et on recommence.

Étonnamment c'est efficace *en pratique*.

$\rightsquigarrow$ `http://www.satcompetition.org/`

$\rightsquigarrow$ Essayons