

Algorithmes et Complexité des Problèmes de Satisfaction de Contraintes (cours n° 2)

Nicolas (Miki) Hermann
Manuel Bodirsky

LIX, École Polytechnique

`hermann@lix.polytechnique.fr`

Clauses particulières

Une clause est dite

- **Horn** si elle contient **au plus un littéral positif** ;
- **dual Horn** si elle contient **au plus un littéral négatif** ;
- **bijonctive** si elle contient **au plus deux littéraux** ;
- **affine** si elle peut s'écrire en forme d'une **équation linéaire modulo 2**.

Formules particulières

Une formule $\varphi = c_1 \wedge \cdots \wedge c_p$ est dite **Horn**, **dual Horn**, **bijonctive** ou **affine** si chaque clause c_i est **Horn**, **dual Horn**, **bijonctive** ou **affine**, respectivement.

Attention

Les formules Horn, dual Horn, bijonctives et affines sont toujours en CNF.

Différentes types de clauses

- $c_1 = (\neg x \vee \neg y \vee z)$ est une clause de **Horn**, mais elle n'est pas dual Horn, ni bijonctive, ni affine ;
- $c_2 = (\neg x \vee y \vee z)$ est **dual Horn** ;
- $c_3 = (\neg x \vee y)$ est **bijonctive**, **Horn** et **dual Horn** ;
- $c_4 = (x \vee y)$ est **bijonctive** et **dual Horn**, mais pas Horn ;
- $c_5 = (x + y + z = 1) \bmod 2$ est **affine** ;
- $c_6 = (x \equiv y)$ est **affine**, **bijonctive**, **Horn** et **dual Horn** ;
- $c_7 = (x \not\equiv y)$ est **affine** et **bijonctive**

Problème HORN SAT

Entrée: Un ensemble de variables V et une formule Horn φ sur V .

Question: La formule φ est-elle satisfaisable ?

Theorem

HORN SAT est décidable en temps polynomial.

Algorithme pour HORN SAT

- 1 Tant que la formule Horn φ contient des clauses unitaires.
 - 1 Soit $\varphi = l \wedge \varphi'$ où l une clause unitaire.
 - 2 Si $l = x$ alors $m(x) := 1$ sinon $m(x) := 0$.
 - 3 Poser $\varphi := \varphi'[x \leftarrow m(x)]$ et simplifier φ , i.e., appliquer $0 \vee x \rightarrow x$, $1 \vee x \rightarrow 1$, $0 \wedge x \rightarrow 0$ et $1 \wedge x \rightarrow x$.
 - 4 Si $\varphi = 0$ alors réponse 0.
- 2 Pour chaque variable x de φ , poser $m(x) := 0$ et réponse 1.

Remarque 1

Notez qu'à la fin de la boucle la formule φ ne contient plus de clauses unitaires et, par conséquent, chaque clause c de φ contient au moins un littéral négatif.

Remarque 2

Cette méthode s'appelle **résolution unitaire** + **propagation**.

- Chaque formule Horn est **héréditaire**. Après la substitution de valeur 0 ou 1 pour une variable x , suivie par la simplification, la formule résultante est toujours Horn.
- Une clause unitaire $c = l$, composée d'un seul littéral $l = x$ ou $l = \neg x$, détermine la valuation $m(x)$ pour la variable x .
- Si une formule Horn $\varphi = c_1 \wedge \dots \wedge c_p$ ne contient pas de clauses unitaires, la valuation $m(x) = 0$ pour chaque variable x **satisfait** φ , car chaque clause c_i contient ou moins un littéral négatif.

Satisfaisabilité des formules dual Horn

Problème DUALHORNSAT

Entrée: Un ensemble de variables V et une formule dual Horn φ sur V .

Question: La formule φ est-elle satisfaisable ?

Lemma

La formule $\varphi(x_1, \dots, x_k)$ est **Horn** si et seulement si la formule $\varphi(\neg x_1, \dots, \neg x_k)$ est **dual Horn**.

Theorem

DUALHORNSAT est décidable en temps polynomial.

Démonstration.

Le vecteur $m = (m[1], \dots, m[k])$ satisfait la formule $\varphi(x_1, \dots, x_k)$ si et seulement si $\neg m = (\neg m[1], \dots, \neg m[k])$ satisfait $\varphi(\neg x_1, \dots, \neg x_k)$. \square

Problème 2SAT

Entrée: Un ensemble de variables V et une formule bijonctive φ sur V .

Question: La formule φ est-elle satisfaisable ?

Theorem

2SAT est décidable en temps polynomial.

Algorithme pour 2SAT (méthode de résolution binaire)

- 1 Soit C l'ensemble de clauses de la formule φ .
- 2 Tant que l'ensemble C n'est pas saturé, appliquer les règles suivantes

$$\begin{array}{lll} [res] & (l \vee x) \wedge (\neg x \vee l') & \rightarrow (l \vee l') \\ [fct] & & (l \vee l) \rightarrow l \\ [rft] & & x \wedge \neg x \rightarrow \perp \end{array}$$

et ajouter les nouvelles clauses à l'ensemble C .

- 3 Si $\perp \notin C$ alors φ est **satisfaisable**, sinon φ est **invalide** ($\perp \in C$ signifie la réfutation de φ).

- Sur un ensemble de variables V il existe seulement $4|V|^2$ clauses bijonctives. Donc il y a seulement $O(|V|^2)$ répétitions de la boucle de saturation possibles.
- La résolution (règle *res*) appliquée sur les clauses bijonctives ne produit que des clauses bijonctives.

Satisfaisabilité des formules affines

Problème AFFINESAT

Entrée: Un ensemble de variables V et une formule affine φ sur V .

Question: La formule φ est-elle satisfaisable ?

Reformulation du Problème AFFINESAT

Entrée: Un système d'équations linéaires affines $A\vec{x} = b$ sur \mathbb{Z}_2 (noté aussi $\mathbb{Z}/2\mathbb{Z}$ ou \mathbb{F}_2).

Question: Le système $A\vec{x} = b$ possède-t-il une solution ?

Theorem

AFFINESAT est décidable en temps polynomial.

Démonstration.

La solution du système affine $A\vec{x} = b$ est déterminée par l'élimination gaussienne. □

Problème 2_{COL}

Entrée: Un graphe $G = (V, E)$ avec les sommets V et les arêtes E .

Question: Existe-t-il une fonction $f: V \rightarrow \{0, 1\}$ telle que $f(x) \neq f(y)$ si $(x, y) \in E$?

Analyse

Relation $2_{\text{col}} = \{01, 10\}$ donne lieu à une contrainte affine

$$2_{\text{col}}(x, y) = (x + y = 1) \bmod 2$$

Donc pour chaque arête $(x, y) \in E$ nous avons la contrainte $(x + y = 1)$. Ceci donne lieu à un système affine d'équations linéaires binaires $A\vec{x} = b$ sur \mathbb{Z}_2 , dont la solution est déterminée en temps polynomial par l'élimination gaussienne.

Récapitulation de la satisfaisabilité des formules

Complexité de la satisfaisabilité booléenne

3SAT est NP-complet.

HORN SAT, DUALHORN SAT, 2SAT et AFFINE SAT sont dans P.

Question 1

Y-a-t-il d'autres cas polynomiaux ?

Réponse

OUI, mais c'est difficile à voir au niveau des formules.

Question 2

Y-a-t-il des cas dans NP qui ne sont ni polynomiaux, ni NP-complets, à condition que $P \neq NP$?

Réponse

NON, mais nous n'avons pas encore les outils pour le voir.

Convention

Nous n'allons plus faire la différence entre la **relation** $R \subseteq \{0, 1\}^k$ et la **contrainte** correspondante $R(x_1, \dots, x_k)$, car elles représentent deux aspects de la même structure.

Construction de relations (1)

Nous avons vu que

$$\begin{aligned} or_0 &= [x \vee y \vee z] &= \{0, 1\}^3 \setminus \{000\} \\ or_3 &= [\neg x \vee \neg y \vee \neg z] &= \{0, 1\}^3 \setminus \{111\} \\ nae &= &= \{0, 1\}^3 \setminus \{000, 111\} \end{aligned}$$

Construction de relations (2)

Nous pouvons donc écrire

$$nae(x, y, z) = or_0(x, y, z) \wedge or_3(x, y, z)$$

au niveau des contraintes ou

$$nae = [x \vee y \vee z] \cap [\neg x \vee \neg y \vee \neg z]$$

au niveau des relations.

Comment construire la négation (1)

Question

Si nous avons l'ensemble de contraintes $S = \{or_0, or_3\}$, comment construire les contraintes or_1 et or_2 ?

Connaissances

Nous avons les identités

$$or_1(x, y, z) = or_0(x, y, \neg z)$$

$$or_2(x, y, z) = or_3(\neg x, y, z)$$

mais nous n'avons ni la négation disponible dans S , ni la possibilité de la placer tout simplement dans l'argument d'une autre contrainte.

Comment construire la négation (1)

Constructions intermédiaires

Construisons d'abord les contraintes binaires suivantes :

$$\begin{aligned}bor_0(x, y) &= or_0(x, y, y) = (x \vee y) \\bor_2(x, y) &= or_3(x, y, y) = (\neg x \vee \neg y)\end{aligned}$$

Inégalité

Nous pouvons maintenant construire la contrainte d'inégalité

$$\begin{aligned}neq(x, y) &= bor_0(x, y) \wedge bor_2(x, y) \\&= (x \vee y) \wedge (\neg x \vee \neg y) = (x \neq y)\end{aligned}$$

Combinaison

Nous pouvons maintenant combiner les contraintes neq_2 et or_0 , ou neq_2 et or_3 , pour construire or_1 ou or_2 :

$$\begin{aligned}or_1(x, y, z) &= or_0(x, y, \neg z) = \exists v neq(z, v) \wedge or_0(x, y, v) \\or_2(x, y, z) &= or_3(\neg x, y, z) = \exists v neq(x, v) \wedge or_3(v, y, z)\end{aligned}$$

Les deux transparents précédents prouvent la proposition suivante.

Theorem

$$\text{CSP}(or_0, or_3) = \text{CSP}(or_0, or_1, or_2, or_3) = 3\text{SAT}$$

Corollary

$\text{CSP}(or_0, or_3)$ est NP-complet.

Comment construire la négation (2)

Construction intermédiaire

Supposons que nous avons la relation

$$nae = \{001, 010, 011, 100, 101, 110\}.$$

Nous pouvons facilement créer la contrainte

$$neq(x, y) = nae(x, y, y)$$

Construction nouvelles

Nous pouvons maintenant créer les contraintes suivantes

$$\begin{aligned} nae_1(x, y, z) &= nae(x, y, \neg z) &= \exists v neq(z, v) \wedge nae(x, y, v) \\ nae_2(x, y, z) &= nae(x, \neg y, \neg z) &= \exists v neq(y, v) \wedge nae_1(x, v, z) \\ nae_3(x, y, z) &= nae(\neg x, \neg y, \neg z) &= \exists v neq(x, v) \wedge nae_2(v, y, z) \end{aligned}$$

Problème NAE3SAT

Entrée: Un ensemble de variables V et une formule $\varphi = c_1 \wedge \dots \wedge c_p$ en CNF sur V avec exactement trois littéraux par clause.

Question: Existe-t-il une valuation m de φ qui évalue dans chaque clause c_i au moins un littéral à 1 et au moins un littéral à 0?

Theorem (Schaefer 1978)

NAE3SAT est NP-complet.

Corollary

CSP(*nae*) est NP-complet.

Règles de construction des contraintes

Soit S un ensemble fini de relations pas forcément de la même arité.
L'ensemble $\langle S \rangle$ contient toutes les contraintes qu'on peut construire à partir des relations dans S . L'ensemble $\langle S \rangle$ est construit par **saturation** :

Règles de saturation de $\langle S \rangle$

Introduction : Si $R \in S$ et $ar(R) = k$ alors $R(x_1, \dots, x_k) \in \langle S \rangle$.

Permutation : Si $R(x_1, \dots, x_k) \in \langle S \rangle$ et π est une permutation de $\{1, \dots, k\}$ alors $R(x_{\pi(1)}, \dots, x_{\pi(k)}) \in \langle S \rangle$.

Diagonalisation : Si $R(x_1, \dots, x_{k-1}, x_k) \in \langle S \rangle$ et
 $R'(x_1, \dots, x_{k-1}) = R(x_1, \dots, x_{k-1}, x_{k-1})$ alors
 $R'(x_1, \dots, x_{k-1}) \in \langle S \rangle$.

Conjonction : Si $R_1(\vec{x}), R_2(\vec{y}) \in \langle S \rangle$ alors $R_1(\vec{x}) \wedge R_2(\vec{y}) \in \langle S \rangle$.

Quantification : Si $R(x_1, \dots, x_{k-1}, x_k) \in \langle S \rangle$ et
 $R'(x_1, \dots, x_{k-1}) = \exists x_k R(x_1, \dots, x_{k-1}, x_k)$ alors
 $R'(x_1, \dots, x_{k-1}) \in \langle S \rangle$.

Les contraintes sont construites à partir d'un ensemble de relations S par conjonction et quantification existentielle. Cette construction possède une dénomination.

Definition (Formule primitive positive)

Une formule $\exists y_1, \dots, y_n \varphi(x_1, \dots, x_k, y_1, \dots, y_n)$ construite à partir d'un ensemble de relations S , la conjonction \wedge et la quantification existentielle \exists s'appelle **primitive positive**, aussi nommée **pp-formule**.

C'est tout pour aujourd'hui.
Avez-vous des questions ?