Base de la démontration automatique : Résolution au premier ordre

Benjamin Wack

Université Grenoble Alpes

Avril 2025

Plan

Plan

Idée

La skolémisation donne des formules sans quantificateur.

On en cherche ensuite des instances insatisfaisables (intuitivement ou par énumération).

Aujourd'hui, généralisation au premier ordre de la résolution :

- Mise en forme clausale des formes de Skolem
- Résolution sur des clauses avec variables
- Unification pour obtenir des littéraux contradictoires

Plan

Littéral, clause au premier ordre

Définition 5.2.19

Un littéral positif est une formule atomique. Ex : $P(x, y)$

Un littéral négatif est la négation d'une formule atomique. Ex : $\overline{Q(a)}$

Une clause est une somme de littéraux. Ex : $P(x, y) + \overline{Q(a)}$

Forme clausale d'une formule

Définition 5.2.20

La forme clausale d'une formule fermée A est obtenue en 2 étapes :

1. skolémiser A (on obtient une forme normale sans quantificateur)
2. distribuer les $\vee$ sur les $\wedge$ pour obtenir un ensemble de clauses Γ

Propriété 5.2.21

$\forall(\Gamma)$ a un modèle si et seulement A a un modèle. Plus précisément :

- $\forall(\Gamma)$ a pour conséquence A
- si A a un modèle alors $\forall(\Gamma)$ en a un aussi

Preuve : On le sait déjà pour la forme de Skolem.
Or cette dernière est équivalente à la forme clausale.

Forme clausale d'un ensemble de formules

Définition 5.2.22

Soit $\Gamma = A_1, \ldots, A_n$ un ensemble de formules fermées.
La forme clausale de Γ est l'union des formes clausales de $A_1, \ldots, A_n$, en prenant soin d'éliminer chaque $\exists$ à l'aide d'un nouveau symbole.

Corollaire 5.2.23

Soient Γ un ensemble de formules fermées et Δ sa forme clausale :

- $\forall(\Delta)$ a pour conséquence Γ
- si Γ a un modèle alors $\forall(\Delta)$ a un modèle.

Adaptation du théorème de Herbrand aux formes clausales

Théorème 5.2.24

Soient Γ un ensemble de formules fermées et Δ sa forme clausale :

Γ est insatisfaisable

si et seulement si

il existe un ensemble fini insatisfaisable d'instances des clauses de Δ.

Preuve.

- La skolémisation préserve la satisfaisablité.
- Puis application du théorème de Herbrand à $\forall(\Delta)$.

□

Exemple 5.2.25

Soit $A = \exists y \forall z \Big(P(z,y) \Leftrightarrow \neg \exists x \big(P(z,x) \wedge P(x,z) \big) \Big)$. Calculons sa forme clausale.

1-4. Les 4 étapes de Skolémisation donnent la forme :

$$\Big(\neg P(z,a) \vee \neg P(z,x) \vee \neg P(x,z) \Big) \ \wedge \ \Big(P(z,f(z)) \wedge P(f(z),z) \ \vee \ P(z,a) \Big)$$

5. Forme clausale :
   - $C_1 = \overline{P(z,a)} + \overline{P(z,x)} + \overline{P(x,z)}$
   - $C_2 = P(z,f(z)) + P(z,a)$
   - $C_3 = P(f(z),z) + P(z,a)$

On recherche un ensemble fini insatisfaisable d'instances de C_1, C_2, C_3.

On instancie :
- C_1 avec $x := a, z := a$ d'où : $C_1' = \overline{P(a,a)}$
- C_2 avec $z := a$ d'où : $C_2' = P(a,f(a)) + P(a,a)$
- C_3 avec $z := a$ d'où : $C_3' = P(f(a),a) + P(a,a)$
- C_1 avec $x := a, z := f(a)$ d'où : $C_1'' = \overline{P(f(a),a)} + \overline{P(a,f(a))}$

L'ensemble C_1', C_1'', C_2', C_3' est insatisfaisable, donc A est insatisfaisable.

Comment procéder en pratique

Soit Γ un ensemble de clauses. On veut montrer que $\forall(\Gamma)$ n'a pas de modèle :

- Comment choisir judicieusement les instances ?
- Comment démontrer leur insatisfaisabilité ?

Le système de « factorisation, copie, résolution binaire » est un système formel permettant de déduire $\bot$ de Γ.

Résolution au premier ordre ?

$$\frac{Q(x) + P(x,a) \qquad \overline{P(b,y)} + R(f(y))}{Q(b) + R(f(a))} \quad \text{pourvu que } x := b,\ y := a$$

Pour trouver les instances contradictoires des clauses, les règles utilisent l'unification.

Plan

John Alan Robinson (1930-2016)

- à l'origine du **principe de résolution**
- algorithme d'**unification** (1965)
  - rend efficace la recherche d'instances de clauses contradictoires
  - cas particulier du *filtrage* utilisé dans les langages fonctionnels
- Fondateur de la *programmation logique*

```
parent(pascal, mathilde).
frere(stephane, pascal).
oncle(X,Y) :- parent(Z,Y),  frere(X,Z).
?- oncle(stephane, mathilde).
  true.
```

(Prolog, Colmerauer & Roussel, 1972)

Unification : expression, solution

Définition 5.3.1

- Un terme ou un littéral est une **expression**.
- Une substitution σ est **solution** de l'**équation** $e_1 = e_2$ si $e_1\sigma$ et $e_2\sigma$ sont syntaxiquement identiques.
- Une substitution est solution d'un ensemble d'équations si elle est solution de chaque équation de l'ensemble.

Unification : exemple 5.3.4

L'équation $P(x, f(y)) = P(g(z), z)$ a pour solution :

$$x := g(f(y)), z := f(y)$$

Le système d'équations $x = g(z),\ \ f(y) = z$ a pour solution :

$$x := g(f(y)), z := f(y)$$

Unification : composition de substitutions

Définition 5.3.5

- Soient σ et τ deux substitutions, on note $\sigma\tau$ la substitution définie par $x\sigma\tau = (x\sigma)\tau$ pour toute variable.
- La substitution $\sigma\tau$ est une instance de σ.
- Deux substitutions sont équivalentes si chacune d'elles est une instance de l'autre.

Unification : exemple 5.3.6

Considérons les substitutions

- $\sigma_1 =< x := g(z), y := z >$
- $\sigma_2 =< x := g(y), z := y >$
- $\sigma_3 =< x := g(a), y := a, z := a >$

On a les relations suivantes entre ces substitutions :

σ_1 et σ_2 sont équivalentes (par renommage de y en z).

σ_3 est une instance de σ_1 ou de σ_2, mais ne leur est pas équivalente.

Unification : définition de la solution la plus générale

Définition 5.3.7 (mgu)

Une solution d'un système d'équations est la plus générale
si toute autre solution en est une instance.
Remarque : deux solutions « les plus générales » sont équivalentes.

Exemple 5.3.8

Considérons l'équation $f(x, g(z)) = f(g(y), x)$.

- $\sigma_1 = < x := g(z), y := z >$
- $\sigma_2 = < x := g(y), z := y >$
- $\sigma_3 = < x := g(a), y := a, z := a >$

sont 3 solutions.

σ_1 et σ_2 sont les plus générales.

Unificateur

Définition 5.3.2

Soit E un ensemble d'expressions et $E\sigma = \{t\sigma \mid t \in E\}$.
σ est un unificateur de E si et seulement si $E\sigma$ est réduit à un élément.

Si $E = \{e_1, \ldots, e_n\}$ cela revient à dire que

σ est solution du système d'équations $\begin{cases} e_1 = e_2 \\ \ldots \\ e_{n-1} = e_n \end{cases}$

On définit de même un unificateur le plus général (ou principal).

Unification : l'algorithme (plan)

L'algorithme distingue :

- les équations à résoudre, notées par un $=$
- les équations résolues, notées par un $:=$

Initialement, il n'y a pas d'équations résolues.

L'algorithme procède par systèmes équivalents et s'arrête quand :

- il n'y a plus d'équations à résoudre,
  alors les équations résolues donnent la solution la plus générale ;
- ou il a prouvé que le système n'a pas de solution.

Unification : l'algorithme (les règles)

Choisir une équation à résoudre puis :

1. **Supprimer l'équation** si ses 2 membres sont identiques.
2. **Décomposer**
   - $\neg A = \neg B$ devient $A = B$
   - $f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)$ devient $s_1 = t_1, \ \ldots, \ s_n = t_n$
     (rien si f est une constante)
3. **Échec de décomposition**
   Si l'équation est $f(s_1, \ldots, s_n) = g(t_1, \ldots, t_p)$ avec $f \neq g$
   alors il n'y a pas de solution.
   (en particulier si l'équation est $\neg A = B$ avec B un littéral positif)

Unification : l'algorithme (les règles)

4. **Orienter**
   Si l'équation est $t = x$ avec t un (vrai) terme et x une variable, on la remplace par $x = t$.
5. **Élimination d'une variable**
   Si l'équation est $x = t$ avec x une variable et t un terme ne contenant pas x :
   - l'enlever des équations à résoudre
   - remplacer x par t partout (équations à résoudre et résolues)
   - ajouter $x := t$ aux équations résolues
6. **Échec de l'élimination**
   Si l'équation est $x = t$ avec x une variable et t contenant x alors il n'y a pas de solution.

Unification : l'algorithme (exemple 5.3.11)

1. Résoudre $f(x, g(z)) = f(g(y), x)$.

Décomposition	$x = g(y)$,	$g(z) = x$	
Élimination de x	$x := g(y)$,	$g(z) = g(y)$	
Décomposition	$x := g(y)$,	$z = y$	
Élimination de z	$x := g(y)$,	$z := y$	solution

2. Résoudre $f(x, x, a) = f(g(y), g(a), y)$.

Décomposition	$x = g(y)$,	$x = g(a)$,	$a = y$
Élimination du (premier) x	$x := g(y)$,	$g(y) = g(a)$,	$a = y$
Décomposition	$x := g(y)$,	$y = a$,	$a = y$
Élimination de y	$x := g(a)$,	$y := a$,	$a = a$
Suppression	$x := g(a)$,	$y := a$	solution

Unification : l'algorithme (exemple 5.3.11)

3. Résoudre $f(x,x,x) = f(g(y),g(a),y)$.

Décomposition	$x = g(y),$	$x = g(a),$	$x = y$
Élimination de x	$x := g(y),$	$g(y) = g(a),$	$g(y) = y$
Orientation	$x := g(y),$	$g(y) = g(a),$	$y = g(y)$
Échec d'élimination	il n'y a pas de solution		

Remarque : L'algorithme est correct et se termine pour tout système (preuves dans le poly).

Plan

Trois règles (exemples)

1. **Factorisation**

$$\frac{P(x,x) + P(y,a) + Q(y)}{P(a,a) + Q(a)} \quad \text{unification}$$

2. **Copie**

$$\frac{P(x,y)}{P(u,v)}$$

3. **Résolution binaire**

$$\frac{Q(x) + P(x,a) \qquad \overline{P(b,y)} + R(f(y))}{Q(b) + R(f(a))} \quad \text{unification}$$

Factorisation

Définition 5.4.2

La clause C' est un facteur de la clause C si :

- $C' = C$
- ou $C' = C\sigma$
  avec σ l'unificateur le plus général d'au moins 2 littéraux de C

Exemple 5.4.3

La clause **P(x)** $+ Q(g(x,y)) +$ **P(f(a))** a deux facteurs :

- elle-même
- $P(f(a)) + Q(g(f(a),y))$ obtenu en appliquant $x := f(a)$

Propriété 5.4.4

Soit C' un facteur de la clause C : alors $\forall(C) \models \forall(C')$.
Preuve : En fait $\forall(A) \vDash \forall(A\sigma)$ pour toute formule A et toute substitution σ.

Copie

Définition 5.4.5

Soit σ une substitution qui :

- transforme toutes les variables en variables ;
- est une bijection.

La clause $C\sigma$ est une copie de la clause C.

σ est aussi appelée un renommage de C.

Exemple 5.4.7

Soit $\sigma = < x := u, y := v >$.
Le littéral $P(u, v)$ est une copie de $P(x, y)$.

Notons que $P(x, y)$ est aussi une copie de $P(u, v)$
par le renommage $\tau = < u := x, v := y >$ inverse de σ.

Copie

Propriété 5.4.8

Si σ est un renommage de C, alors C est aussi une copie de $C\sigma$.

Preuve.

On montre facilement que σ^{-1} est un renommage de $C\sigma$. □

Propriété 5.4.9

Si C et C' sont copies l'une de l'autre, alors $\forall(C) \equiv \forall(C')$.

Preuve.

C et C' sont instances l'une de l'autre.
Donc $\forall(C) \models \forall(C')$ et réciproquement. □

Résolvant binaire (RB)

Définition 5.4.10

Soient C et D deux clauses n'ayant pas de variable commune.
S'il y a deux littéraux :

- $C = C' + L$
- $D = D' + \overline{M}$
- tels que L et M sont unifiables
- σ est la solution la plus générale de l'équation $L = M$

$E = (C' + D')\sigma$ est un résolvant binaire de C et D.

Résolvant binaire

Exemple 5.4.11

Soient $C = P(x,y) + P(y,k(z))$ et $D = \overline{P(a,f(a,y_1))}$.

$< x := a, y := f(a,y_1) >$ est la solution la plus générale de $P(x,y) = P(a,f(a,y_1))$.
Le (seul) résolvant binaire est $P(f(a,y_1),k(z))$.

Propriété 5.4.12

Soit E un résolvant binaire des clauses C et D : $\forall(C), \forall(D) \models \forall(E)$.

Résolution

Définition 5.4.13

Une preuve de C à partir de Γ est une suite de clauses toutes :

- élément de Γ,
- ou facteur d'une clause précédente,
- ou copie d'une clause précédente,
- ou résolvant binaire de 2 clauses précédentes,

se terminant par C.

On note $\Gamma \vdash_{1fcb} C$ s'il y a une preuve de C à partir de Γ.

Propriété 5.4.14 : cohérence

Si $\Gamma \vdash_{1fcb} C$ alors $\forall(\Gamma) \models \forall(C)$

Par récurrence, en se basant sur la cohérence des 3 règles.

Résolution : Exemple 5.4.15

Soient les deux clauses

1. $C_1 = P(x,y) + P(y,x)$
2. $C_2 = \overline{P(u,z)} + \overline{P(z,u)}$

Montrons par résolution que $\forall(C_1, C_2)$ n'a pas de modèle.

1.	$P(x,y) + P(y,x)$	Hyp C_1
2.	$\overline{P(y,y)}$	Facteur de 1 $< x := y >$
3.	$\overline{P(u,z)} + \overline{P(z,u)}$	Hyp C_2
4.	$\overline{P(z,z)}$	Facteur de 3 $< u := z >$
5.	$\bot$	Résolvant Binaire 2, 4 $< y := z >$

Cet exemple montre que la résolution binaire seule est incomplète : sans la factorisation on ne peut pas déduire la clause vide.

Résolution : Exemple 5.4.16

1. $C_1 = \overline{P(z,a)} + \overline{P(z,x)} + \overline{P(x,z)}$
2. $C_2 = P(z,f(z)) + P(z,a)$
3. $C_3 = P(f(z),z) + P(z,a)$

1.	$\overline{P(z,a)} + \overline{P(z,x)} + \overline{P(x,z)}$	Hyp C_1
2.	$P(z,f(z)) + P(z,a)$	Hyp C_2
3.	$P(v,f(v)) + P(v,a)$	Copie 2 $< z := v >$
4.	$\overline{P(f(v),a)} + \overline{P(f(v),v)} + P(v,a)$	RB 1(3), 3(1) $< z{:=}f(v)\,;\, x{:=}v >$
5.	$\overline{P(f(a),a)} + P(a,a)$	Fact 4 $< v := a >$
6.	$P(f(z),z) + P(z,a)$	Hyp C_3
7.	$P(a,a)$	RB 5(1), 6(1) $< z := a >$
8.	$\overline{P(a,a)}$	Fact 1 $< x := a; z := a >$
9.	$\bot$	RB 7, 8

Plan

Théorème du relèvement (intuition)

Si on note :

- $\Gamma \vdash_p C$: preuve par résolution propositionnelle (sans substitution)
- $\Gamma \vdash_{1fcb} C$: preuve par factorisation, copie et résolution binaire.

Théorème 5.4.21

Soit Γ un ensemble de clauses et Δ des instances de ces clauses.

Soit une preuve par résolution propositionnelle $\Delta \vdash_p C$.

Alors il existe une preuve similaire par résolution au 1^{er} ordre $\Gamma \vdash_{1fcb} D$ en remplaçant chaque clause de Δ par celle dont elle est l'instance.

On dit que la preuve de C est relevée en une preuve au premier ordre.

Exemple 5.4.23

$\Gamma = \{P(f(x)) + P(u), \overline{P(x)} + Q(z), \overline{Q(x)} + \overline{Q(y)}\}$.

$\forall(\Gamma)$ est insatisfaisable et nous le montrons de trois manières.

1. Instanciation sur le domaine de Herbrand $a, f(a), f(f(a)), \ldots$:

$$\begin{array}{rll} P(f(x)) + P(u) & \text{est instanciée en} & P(f(a)) \\ \overline{P(x)} + Q(z) & \text{est instanciée en} & \overline{P(f(a))} + Q(a) \\ \overline{Q(x)} + \overline{Q(y)} & \text{est instanciée en} & \overline{Q(a)} \end{array}$$

Preuve par $\vdash_p$ que cet ensemble d'instances est insatisfaisable :

$$\dfrac{\dfrac{P(f(a)) \qquad \overline{P(f(a))} + Q(a)}{Q(a)} \qquad \overline{Q(a)}}{\bot}$$

Exemple 5.4.23

$$\Gamma = \{P(f(x)) + P(u), \overline{P(x)} + Q(z), \overline{Q(x)} + \overline{Q(y)}\}$$

1. Preuve par instanciation et $\vdash_p$

$$\dfrac{\dfrac{P(f(a)) \qquad \overline{P(f(a))} + Q(a)}{Q(a)} \qquad \overline{Q(a)}}{\bot}$$

2. On relève cette preuve en remplaçant chaque clause par celle dont elle est l'instance :

$$\dfrac{\dfrac{P(f(x)) + P(u) \qquad \overline{P(x)} + Q(z)}{Q(z)} \qquad \overline{Q(x)} + \overline{Q(y)}}{\bot}$$

Exemple 5.4.23

2.

$$\dfrac{\dfrac{P(f(x))+P(u) \qquad \neg P(x)+Q(z)}{Q(z)} \qquad \neg Q(x)+\neg Q(y)}{\bot}$$

3. Chaque étape peut être prouvée par factorisation, copie et résolution binaire :

$$\dfrac{\dfrac{\dfrac{P(f(x))+P(u)}{P(f(x))}\,fact \qquad \dfrac{\overline{P(x)}+Q(z)}{\overline{P(y)}+Q(z)}\,copie}{Q(z)}\,rb \qquad \dfrac{\overline{Q(x)}+\overline{Q(y)}}{\overline{Q(x)}}\,fact}{\bot}\,rb$$

Complétude réfutationnelle de la résolution au 1er ordre

Théorème 5.4.24

Si (1) $\forall(\Gamma) \models \bot$

alors (2) il existe σ tel que $\Gamma\sigma \vdash_p \bot$

et donc (3) $\Gamma \vdash_{1fcb} \bot$.

Démonstration.

- (1 $\Rightarrow$ 2). Supposons que $\forall(\Gamma)$ est insatisfaisable.
  D'après Herbrand, on peut instancier Γ en un $\Gamma\sigma$ insatisfaisable.
  Par complétude de la résolution propositionnelle, on a $\Gamma\sigma \vdash_p \bot$.
- (2 $\Rightarrow$ 3) La preuve $\Gamma\sigma \vdash_p \bot$ se relève en une preuve $\Gamma \vdash_{1fcb} \bot$.

□

Preuves automatiques

Pour produire automatiquement des preuves en résolution binaire, il est possible d'utiliser le logiciel (principe similaire à la *stratégie complète*) :

http://teachinglogic.univ-grenoble-alpes.fr/ResBinSc/

Si l'ensemble de clauses est insatisfaisable, le logiciel est théoriquement capable de déduire la clause vide (en un temps illimité).

Que peut-on en conclure ?

- si le logiciel affirme qu'il a déduit la clause vide :
  - les clauses sont effectivement insatisfaisables
  - il en fournit une preuve
- si le logiciel affirme qu'il ne peut pas prouver la clause vide ou s'il atteint sa limite de temps :
  - on ne peut rien conclure

Plan

Aujourd'hui

- L'unification permet de déterminer des instanciations judicieuses de clauses avec variables
- La résolution au premier ordre intègre dans un même système déductif la **recherche d'instances insatisfaisables** et la **preuve d'insatisfaisabilité** d'un ensemble de clauses
- La résolution au premier ordre est **correcte** et **complète**.

Plan du Semestre

- Logique propositionnelle
- Résolution propositionnelle
- Déduction naturelle propositionnelle

PARTIEL

- Logique du premier ordre
- Base de la démonstration automatique (“résolution au premier ordre”) *
- Déduction naturelle au premier ordre

EXAMEN

La prochaine fois

Déduction naturelle au premier ordre

- Règles
- Exemples