Le plan va être recopié →

919 Unification : Algorithmes et applications

I Unification [Baa] IV 6-7-8

1 Cadre

Soit $\mathcal{L} = (\mathcal{V}, \mathcal{F} = \bigcup_{n \geq 0} \mathcal{F}_n)$ un langage contenant des symboles de variables, et de fonctions de toute arité. On va travailler sur l'ensemble $T_{\mathcal{L}}$ des termes bien formés de $\mathcal{L}$.

Ex : $\mathcal{L}_0 = \{\{(x_i)_{i\in\mathbb{N}}\}, \{f^{(2)}\}\}$ $\quad \mathcal{L}_1 = \{\{x, y, z\}, \{a^{(0)}, h^{(1)}, f^{(2)}, g^{(2)}\}\}$

$\mathcal{L}_2 =$

Def 1 : Une fonction $\sigma : \mathcal{V} \to T_{\mathcal{L}}$ tq $Dom(\sigma) = \{x, \sigma(x) \neq x\}$ est fini est appelée substitution

Ex : $(x \to y)$ $\quad (x \to f(x, y),\ y \to g(z))$

Prop 2 : une substitution s'étend canoniquement en une fonction $\hat{\sigma} : T_{\mathcal{L}} \to T_{\mathcal{L}}$ (qu'on confondra parfois avec σ)

Def 3 : La composition de σ par τ est la substitution :
$\sigma \circ \tau : x \mapsto \hat{\sigma}(\tau(x))$

Prop 4 : l'ensemble des substitutions est muni d'une relation $\leq$ réflexive transitive $\quad \sigma \leq \sigma'$ ssi $\exists \theta^{\text{subst}}$ tq $\sigma = \theta \circ \sigma'$

2 Problème d'unification

Entrée : $E = (t_i, t'_i)_{1 \leq i \leq n}$ un ensemble fini de couples de termes

Sortie : Pour $\mathcal{U}(E) = \{\sigma \text{ tq } \forall i, \sigma(t_i) = \sigma(t'_i)\}$ l'ensemble des unificateurs de E :

Si $\mathcal{U}(E) = \emptyset$: Echec

Sinon, un élément de $\mathcal{U}(E)$

Ex : $(x \to h(a),\ y \to a,\ z \to h(a))$ est un unificateur de (t_1, t'_1) cf annexe

- $(x \to g(z),\ y \to z)$ est un unificateur de $(x, g(y))$
- $(x_2 \to x_3)$ est un unificateur de (x_1, x_1) $\quad \bullet\ \mathcal{U}((a, b)) = \emptyset$

$\bullet\ \mathcal{U}(f(x, x), h(a)) = \emptyset$

3 Algorithmes d'unification

Théorème 5 : L'algorithme d'unification de Robinson termine, et

- si $\mathcal{U}(E) = \emptyset$, renvoie Echec
- Sinon, renvoie $\sigma \in \mathcal{U}(E)$ tq $\forall \sigma' \in \mathcal{U}(E)$, $\sigma \leq \sigma'$

$\boxed{\text{DEV 1}}$

Corollaire : Si $\mathcal{U}(E) \neq \emptyset$, $\mathcal{U}(E)$ possède un unificateur le plus général (mgu), c'est à dire une borne inférieure pour $\leq$.

Implémentations : Naïve → Complexité spatiale exponentielle

Avec des DAGs → Complexité spatiale linéaire
Complexité temporelle exponentielle $\quad$ Fig 2

Améliorations

① Diviser pour régner : Classes d'équivalences sur les sous-termes (noeuds du DAG)

- Lorsqu'on unifie deux termes, on fait l'union des classes
- Les appels récursifs se font sur les représentants de chaque classe

→ Complexité temporelle quadratique

② Implementation des classes d'équivalence avec Union-Find

- Occurs-check seulement à la fin (on vérifie que le graphe obtenu est acyclique)

→ Complexité temporelle quasi-linéaire

⊕ Occurs-check sans répétition : on ne passe au plus qu'une fois par noeud [illegible]

Application : Pattern Matching (t, p) : cas simplifié d'unification
- Les variables des deux termes sont considérées comme séparées
- Les variables du terme gauche sont considérées comme des constantes

→ Algo en complexité temporelle linéaire (Version un peu modifiée de Robinson)

II Réécriture [Baa] 4.2

Def 6: Un <u>système de réécriture</u> est un ensemble $R = \{(l_i \rhd r_i) \quad i \in I\}$ de règles, où les l_i et les r_i sont des termes

Ex : $R_1 = \{0 + x \rhd x \ ; \ S(x)+y \rhd S(x+y)\}$

$R_2 = \{a_i \rhd a_{i+1} \ ; \ a_i \rhd a \ , \ i \in \mathbb{N}\}$

$R_3 = \{a \rhd b \ ; \ b \rhd a \ ; \ a \rhd c \ ; \ b \rhd d\}$

$R_4 = \left\{\begin{array}{l} f(g(x)) \rhd h(g(x)) \\ g(x) \rhd h(x) \end{array}\right\}$

Def 7 Une position d'un terme est un noeud de l'arbre représentant ce terme.
Pour $p \in Pos(t)$, t_p est le sous terme de t de racine p. et $t[s]_p$ le terme t où t_p a été remplacé par s.

Def 8 On définit une relation binaire sur T_Σ, notée $\longrightarrow_R$ (ou $\longrightarrow$) par :

$t_1 \longrightarrow_R t_2$ ssi $\exists\, l \rhd r \in R$, $\exists p \in Pos(t_1)$, $\exists \sigma$ subst tq $t_1|_p = \sigma l$ et $t_2 = t_1[\sigma r]_p$

$\boxed{t_1 \text{ se réduit en } t_2}$

Def 9 Un terme t est dit irréductible ssi $\forall u$, $t \not\longrightarrow u$
Deux termes u et v sont dits joignables ssi $\exists t$, $u \longrightarrow^* t$, $v \longrightarrow^* t$. On note : $u \downarrow v$

On note $\longrightarrow^*$ la fermeture réflexive transitive de $\longrightarrow$
$\longrightarrow^+$ transitive de $\longrightarrow$.

~~Def~~ Prop 10 d'un système de réécriture.
- <u>Terminaison</u> : Toute suite de réductions est finie
- <u>Confluence</u> : $\forall t, t_1, t_2$ tq $t \longrightarrow^* t_1$, $t \longrightarrow^* t_2$, $t_1 \downarrow t_2$
- <u>Confluence locale</u> : $\forall t, t_1, t_2$ tq $t \longrightarrow t_1$, $t \longrightarrow t_2$, $t_1 \downarrow t_2$

<u>Prop 11</u> Si R est terminant et confluent, tout terme possède une unique forme normale : $\forall t, \exists u$ tq u irred et $t \longrightarrow^* u$

Ex : R_1 est terminant et confluent
R_2 est confluent mais ne termine pas
R_3 est localement confluent, mais pas confluent (ni terminant)

<u>Théorème 12</u> Les deux problèmes de décision suivant sont indécidables :
Entrée : Un système de réécriture R
Sortie$_1$: Oui ssi R est terminant [Baa] ~~VI~~ V
Sortie$_2$: Oui ssi R est confluent

Décider la confluence (avec terminaison)

Lemme 13 de Newman : Si R est terminant, alors la confluence locale implique la confluence globale.

Def 14 Paire Critique : Soient $(l_1 \rhd r_1)$ et $(l_2 \rhd r_2) \in R$ (où les variables de chaque paire ont été séparées)
$p \in Pos(l_1)$ tq $l_1|_p$ n'est pas une variable
Si $l_1|_p$ et l_2 unifiables, alors $(\theta r_1, \theta l_1[\theta l_2])$ est une <u>paire critique</u> (pour θ mgu)

On trouve σ grâce à l'algo de Pattern-Matching

Ex : Dans R_4, on a
$f(g(x)) \longrightarrow h(g(x))$
$ \longrightarrow f(h(x))$
Paire critique (non joignable)

Théorème 15 Un système de réécriture terminant est confluent ssi toutes les paires critiques sont joignables [Baa VI]

Théorème 16 La confluence d'un système terminant et fini (nb fini de règles) est décidable

Q° : Est-ce que ... (paires critiques) marche si on a un unificateur mais pas minimal ?

III La méthode de résolution et la programmation logique

1) Méthode de résolution [CL] p267 [Ste] p220

Pour $\mathcal{C}$ une clause du 1er ordre (conjonction de littéraux), on notera $\forall \mathcal{C}$ sa clôture universelle (nommée clause universelle)

Déf 13: Soient $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}$ des clauses, avec $\mathcal{C}_1$ et $\mathcal{C}_2$ sans variable libre commune.
$\forall \mathcal{C}$ est une résolution de $\forall \mathcal{C}_1$ et $\forall \mathcal{C}_2$ s'il existe $\mathcal{S}_1 \subset \mathcal{C}_1$, $\mathcal{S}_2 \subset \mathcal{C}_2$, et σ une substitution tels que :
- σ unifie $\mathcal{S}_1$ et $\mathcal{S}_2$
- $\mathcal{C} = \sigma((\mathcal{C}_1 \setminus \mathcal{S}_1) \cup (\mathcal{C}_2 \setminus \mathcal{S}_2))$

Une preuve par résolution de $\forall \mathcal{C}$ à partir de $\forall \Gamma$ (ensemble de clauses) est un arbre dont la racine est $\forall \mathcal{C}$ et les feuilles sont dans $\forall \Gamma$ et chaque nœud est résolution de ses fils.

Thm 14: Toute théorie du 1er ordre est équisatisfiable à un ensemble de clauses universelles.

Thm 15: La résolution est complète
càd pour $\forall \mathcal{E}$ un ensemble de clauses et $\forall F$ une clause, il existe une preuve par résolution de $\forall F$ à partir de $\forall \mathcal{E}$ ssi tout modèle de $\forall \mathcal{E}$ est modèle de $\forall F$

$\boxed{\text{DEV 2}}$: Sens $\Rightarrow$ à l'aide des lemmes :
Notations : Pour $\mathcal{L}$ un langage, on note $\mathcal{T}$ l'ensemble des termes, $\mathbb{P}$ l'ensemble des formules sans quantif., $\mathcal{J}$ l'ensemble des formules atomiques,

FAUX

Lemme 16: Il existe une bijection entre les $\mathcal{L}$-structures et les valuations (propositionnelles) sur $\mathcal{P}$ vérifiant :
$\forall F \in \mathcal{J}, \quad \mathcal{M} \models F \iff \beta(\mathcal{M})(F) = 1$

Lemme 17: Soit $\mathcal{E}$ un ensemble de clauses,
$E := \{\sigma(\mathcal{C}), \sigma \text{ une substitution}, \mathcal{C} \in \mathcal{E}\}$
On suppose qu'il existe un arbre de preuve par coupure de C à partir de E.
Alors il existe une substitution τ, une clause $\mathcal{C}$ et un arbre de preuve par résolution de $\mathcal{C}$ à partir de $\mathcal{E}$, avec $C = \tau(\mathcal{C})$

2) Programmation logique

On cherche à créer un programme prenant en entrée une théorie T et donnant en sortie si elle est contradictoire.

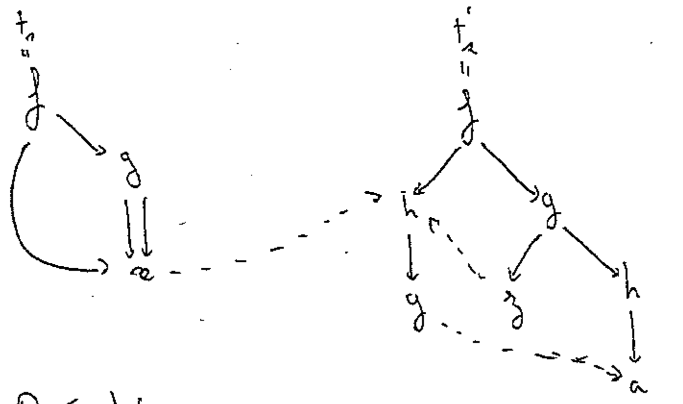
On n'utilise en pratique que des clauses de Horn (avec au plus un littéral positif), l'algorithme cherche donc des unificateurs pour tenter d'appliquer la méthode de résolution.

[CL] : Cori-Lascar, Logique mathématique
[Ste] : Stern, Fondements mathématique de l'info.
[Baa] : Baader, Term Rewriting and All That.
[Dow] : Dowek, Les démonstrations et les algorithmes.

[Bauder]

$t_1 = f(x, g(x,x))$

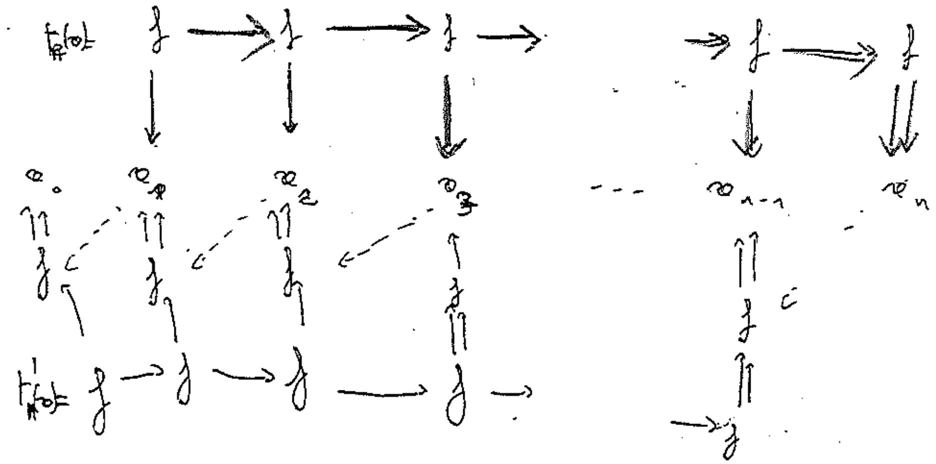
$t'_1 = f(h(y), g(z, h(a)))$



Représentation DAG

Antiunification générée par gen

Fig 1



$x_i \to$ arbre binaire complet de hauteur i

Fig 2

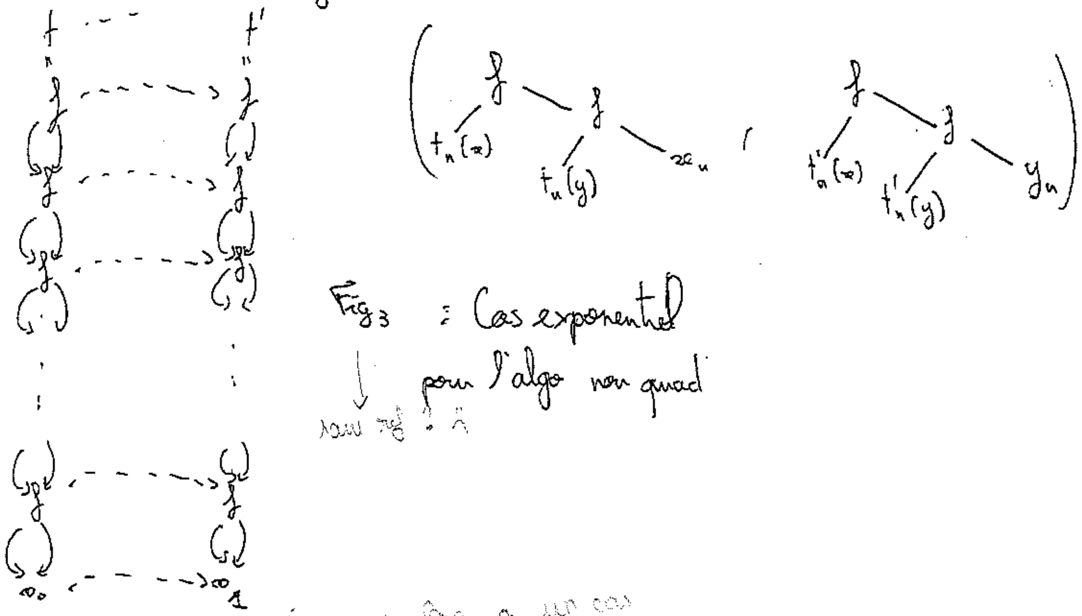
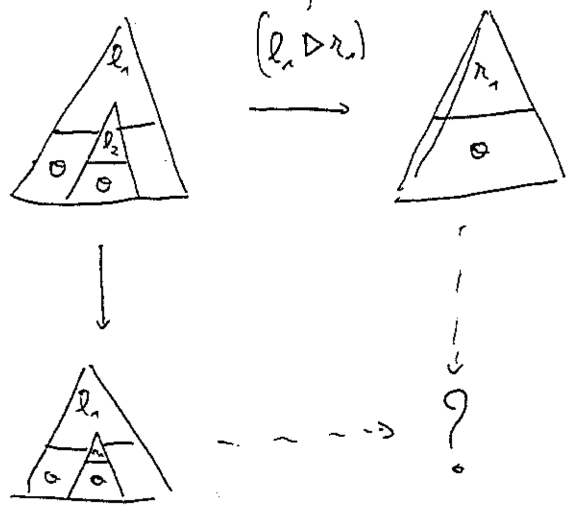


Fig 3 : Cas exponentiel pour l'algo non quad

non réf ?!

(Rq : les Bou a un cas exponentiel ?)

Paires critiques :



Complétude de la méthode de résolution.
Références :
J.Stern "Fondements mathématiques de l'informatique", *Ediscience* ; p220
R.Cori, D.Lascar "Logique mathématiques vol 1" *Dunod*, p245
Le David-Nour-Raffali présente en 2 pages la méthode et essaie de l'optimiser, sans trop justifier pourquoi ni comment elle fonctionne
Le Dehornoi présente lui aussi en 2 pages et ne démontre pas le théorème principal
On conseillera aussi la lecture du document suivant, qui complète le Stern pour ce développement : `http://minerve.ens-rennes.fr/images/Dvt_resolution.pdf`

Introduction :

Prérequis : substitution, unificateur, unificateur principal, équisatisfiabilité, (clause), VL (ensemble des variables libres d'une formule), preuve par coupure (ainsi que sa complétude dans la logique propositionnelle)
Notations : Pour C une clause (du 1er ordre, et pas forcément close), on notera $\forall C$ sa clôture universelle : c'est à dire qu'on ajoutera un $\forall x$ pour chaque variable x libre dans C. On appellera "clause universelle" une clause close universellement.
Pour $C = A_1 \vee A_2 \vee \ldots \vee A_a \vee \neg B_1 \vee \neg B_2 \vee \ldots \vee \neg B_b$ une clause et σ une substitution, on notera $\sigma(C) = \sigma(A_1) \vee \sigma(A_2) \vee \ldots \sigma(\vee A_a) \vee \neg\sigma(B_1) \vee \neg\sigma(B_2) \vee \ldots \vee \neg\sigma(B_b)$.

Définition :

Soient $\forall \mathcal{C}_0$, $\forall \mathcal{C}_1$, $\forall \mathcal{C}_2$ des clauses universelles du 1^{er} ordre,
$\mathcal{C}_1 = A_1 \vee A_2 \vee \ldots \vee A_a \vee \neg B_1 \vee \neg B_2 \vee \ldots \vee \neg B_b$
$\mathcal{C}_2 = C_1 \vee C_2 \vee \ldots \vee C_d \vee \neg D_1 \vee \neg D_2 \vee \ldots \vee \neg D_e$
$\forall \mathcal{C}_0$ est une résolution de $\forall \mathcal{C}_1$ et $\forall \mathcal{C}_2$ si $VL(\mathcal{C}_1) \bigcap VL(\mathcal{C}_2) = \varnothing$ et s'il existe $X \subset \{1 \ldots j\}$, $Y \subset \{1 \ldots k\}$ et σ une substitution tels que :
σ soit un unificateur principal[1] de $\{B_x, x \in X\}$ et $\{D_y, y \in Y\}$ et

$$\mathcal{C}_0 = \left(\bigvee_{i=1}^{a} \sigma(A_i)\right) \vee \left(\bigvee_{j \notin X} \sigma(B_j)\right) \vee \left(\bigvee_{k \notin Y} \sigma(C_k)\right) \vee \left(\bigvee_{l=1}^{e} \sigma(D_l)\right)$$

Avec de plus les $\sigma(A_i)$ différents les uns des autres, les $\sigma(A_i)$ différents des $\sigma(B_j)$, les $\sigma(B_j)$ différents les uns des autres, et de même pour les $\sigma(C_k)$ et les $\sigma(D_l)$.
Une reformulation d'une clause $\forall \mathcal{C}$ est une clause de la forme $\forall\sigma(\mathcal{C})$, avec σ une substitution.[2]
Une preuve par résolution de $\forall \mathcal{C}$ (clause) à partir de Γ (ensemble de clauses universelles) est un arbre (fini) dont la racine est $\forall \mathcal{C}$, les feuilles sont dans

[1] Le principal n'est pas nécessaire, mais il est plus intéressant algorithmiquement
[2] On peut voir ça comme, pour F une formule sans quantificateurs, une résolution de $\forall(\mathcal{C} \vee F)$ et de $\forall(\mathcal{C} \vee \neg F)$

Γ et chaque noeud est résolution de ses deux fils ou reformulation de son fils.

Théorème :
La résolution est complète
càd Pour T un ensemble de clauses (universelles) sur $\mathcal{L}$ et F une clause, il existe une preuve par résolution de F à partir de T
ssi tout modèle de T est modèle de F

Ce théorème se reformule aisément en :
T a un modèle ssi il n'existe pas de preuve par résolution de $\perp$ à partir de T.
Ou de manière équivalente :
Il exsite une preuve par résolution de $\perp$ à partir de T ssi T n'a pas de modèle.

Le sens de gauche à droite s'appelle la correction de la méthode de résolution, et il ne s'agit pas ici de le montrer (la démonstration se fait en vérifiant que la résolution préserve les résultats des valuations, c'est un peu moins évident pour les closes universelles que pour juste les clauses, mais ça ne présente cependant pas de difficulté technique majeure).
On va montrer l'autre sens en s'appuyant sur la complétude de la preuve par coupure dans la logique propositionnelle que l'on admet.

Avant tout, il est important de remarquer que l'on ne perd pas de généralités en prenant T un ensemble de clauses et pas une théorie. Cela est dû à la proposition suivante, que l'on admettra pour un oral :

Proposition :
Toute théorie est équisatisfiable à un certain un ensemble de clauses universelles.

La démonstration se fait par une mise en forme prénexe des formules de la théorie (la nouvelle théorie est équivalente) On obtient une formule de forme $Qx_1 \dots Qx_n F[x_1, \dots x_n]$ avec Q des quantificateurs, et F sans quantificateurs. On peut alors mettre F sous forme de conjonction de clauses. On fait alors passer les conjonctions avant les quantificateurs, et l'on sépare chacune des formules en différentes formules. (la nouvelle théorie est un ensemble de clauses quantifiés qui est toujours équivalente à la théorie de départ). Ensuite, on met sous forme de Skolem (équisatisfiable, mais plus équivalente) pour enlever les quantificateurs existentiels. Reste à traiter le cas des quantificateurs universels...
Plusieurs choix s'offrent alors à nous :
- Définir la résolution sur l'ensemble des clauses universelles, comme fait dans le Cori-Lascar, et comme on a choisi de le faire.

- Dire qu'une clause universelle admet un modèle si et seulement si l'ensemble de ses particularisations admet un modèle, (une particularisation étant la supression de tous les quantificateurs puis le remplacement des variables libérées par des termes clos du langage). Ce qui est fait via le théorème de Herbrand dans le Stern.

Soit $\mathcal{L}$ un langage. On note $\mathcal{T}$ l'ensemble de ses termes, et $\mathcal{P}$ l'ensemble de ses formules atomiques.
On peut construire une logique propositionnelle sur $\mathcal{P}$. On note $\mathbb{P}$ l'ensemble des formules propositionnelles dont les variables sont dans $\mathcal{P}$ (c'est aussi l'ensemble des formules du 1^{er} ordre sans quantificateurs sur $\mathcal{L}$). Se donner une valuation (propositionnelle) sur $\mathcal{P}$, c'est s'en donner une sur $\mathbb{P}$.

Lemme :
Pour $\mathcal{M}$ une $\mathcal{L}$-structure, il existe une valuation $\delta_{\mathcal{M}} : \mathcal{P} \to \{0,1\}$ telle que pour toute formule atomique F, $\mathcal{M} \models F \iff \delta_{\mathcal{M}}(F) = 1$.
Réciproquement, pour δ une valuation sur $\mathcal{P}$, il existe une structure $\mathcal{M}_\delta$ telle que pour toute formule atomique F, $\delta(F) = 1 \iff \mathcal{M}_\delta \models F$

Proof. $\Rightarrow$ Si $\mathcal{M}$ est une $\mathcal{M}$-structure, on définit $\delta : \begin{array}{ccc} \mathcal{P} & \to & \{0,1\} \\ F & \mapsto & 1 \text{ si } \mathcal{M} \models F \\ & & 0 \text{ si } \mathcal{M} \not\models F \end{array}$
Cela définit bien une valuation propositionnelle sur $\mathcal{P}$ (et donc sur $\mathbb{P}$)

$\Leftarrow$ Si δ est une valuation sur $\mathcal{P}$, on crée la structure $\mathcal{M}$ suivante :
Son ensemble sous-jacent est $\mathcal{T}$ l'ensemble des termes de $\mathcal{L}$
Pour c symbole de constante, $c^{\mathcal{M}} := c$
Pour f symbole de fonction, $f^{\mathcal{M}} := \begin{array}{ccc} \mathcal{T}^n & \to & \mathcal{T} \\ t_1, \dots, t_n & \mapsto & ft_1 \dots t_n \end{array}$
Pour R symbole de relation, $R^{\mathcal{M}} := \{t_1, \dots, t_n \in \mathcal{T}^n / \delta(\underbrace{Rt_1 ... t_n}_{\in \mathcal{P}}) = 1\}$
Cette structure est communément appelée modèle de Herbrand. Nous préférons cependant l'appelation <u>structure de Herbrand</u>[3].
Si $\delta(F) = 1$, alors par la valuation du 1er ordre $\rho : \begin{array}{ccc} \mathbb{V} & \to & \mathcal{T} \\ v_i & \mapsto & v_i \end{array}$,
on obtient un modèle de f. □

Lemme :
Soit T un ensemble de clauses (avec variables libres, mais non quantifiées). On suppose qu'il existe un arbre de preuve par coupure dont les feuilles sont

[3]Il arrive (notamment dans le Stern) que cette structure soit définie sur l'ensemble des termes clos et pas sur l'ensemble des termes. Il faut alors que le langage ait au moins un symbole de constante pour que la structure ne soit pas vide. Si le langage a un nombre infini de constantes, alors ces deux structures sont isomorphes.

dans l'ensemble $\Gamma = \{\sigma(\mathcal{C}), \sigma$ une substitution$, \mathcal{C} \in T\}$, et dont la racine est $\mathcal{C}_0$. Alors il existe une substitution τ, une clause $\mathcal{E}$ et une preuve par résolution de $\forall\mathcal{E}$ à partir de $\forall T = \{\forall\mathcal{C}, \mathcal{C} \in T\}$, avec de plus $\tau(\mathcal{E}) = \mathcal{C}_0$ [4]

Proof. On raisonne par induction sur l'arbre de coupure.
Feuille : Si $\sigma(\mathcal{C})$ est dans Γ, alors on crée une reformulation de $\mathcal{C}$ vers $\sigma(\mathcal{C})$
Si la racine de l'arbre est $\mathcal{C}$ et celle des fils sont $\mathcal{C}_1$ et $\mathcal{C}_2$:
Par hypothèse d'induction, il existe des substitutions τ_1 et τ_2, des clauses $\mathcal{E}_1$ et $\mathcal{E}_2$ tels que $\tau_1(\mathcal{E}_1) = \mathcal{C}_1$, $\tau_2(\mathcal{E}_2) = \mathcal{C}_2$ et il existe des preuves par résolution de $\forall\mathcal{E}_1$ et de $\forall\mathcal{E}_2$ à partir de $\forall T$.
De plus, comme l'on a affaire à un arbre de coupure, on sait qu'il existe une coupure de $\mathcal{C}_1$ et $\mathcal{C}_2$ amenant à $\mathcal{C}$.
On commence par séparer les variables libres : soit σ une permutation de l'ensemble des variables telle que $\mathcal{E}_1$ et $\mathcal{E}_3 := \sigma(\mathcal{E}_2)$ n'aient aucune variable en commun (ce qui est posible puisque ces clauses sont finies sur des termes finis et que l'ensemble des variables est infini).
On pose alors $\mu : \begin{array}{ccll} \mathbb{V} & \to & \mathcal{T} & \\ v_i & \mapsto & \tau_1(v_i) & \text{si } v_i \in VL(\mathcal{E}_1) \\ & & \tau_2 \circ \sigma^{-1}(v_i) & \text{si } v_i \in VL(\sigma(\mathcal{E}_2)) \\ & & v_i & \textit{sinon} \end{array}$
Par construction, $\mu(\mathcal{E}_1) = \mathcal{C}_1$ et $\mu(\mathcal{E}_3) = \mathcal{C}_2$
On note $\mathcal{E}_1 = A_1 \vee A_2 \vee \ldots \vee A_a \vee \neg B_1 \vee \neg B_2 \vee \ldots \vee \neg B_b$
et $\mathcal{E}_3 = C_1 \vee C_2 \vee \ldots \vee C_d \vee \neg D_1 \vee \neg D_2 \vee \ldots \vee \neg D_e$
On sait que l'on peut appliquer la règle de coupure à $\mathcal{C}_1$ et $\mathcal{C}_2$, c'est-à-dire à $\mu(\mathcal{E}_1)$ et à $\mu(\mathcal{E}_3)$. On peut supposer sans perdre de généralité qu'elle est appliquée en enlevant des littéraux négatifs $\{\mu(B_x), x \in X\}$ de $\mathcal{C}_1$ et des littéraux positifs $\{\mu(C_y), y \in Y\}$ de $\mathcal{C}_2$ (et donc qu'il y a égalité entre ces ensembles). De plus, on a :

$$\mathcal{C} = \left(\bigvee_{i=1}^{a} \mu(A_i)\right) \vee \left(\bigvee_{j \notin X} \mu(B_j)\right) \vee \left(\bigvee_{k \notin Y} \mu(C_k)\right) \vee \left(\bigvee_{l=1}^{e} \mu(D_l)\right)$$

Avec des hypothèses raisonnables de non-égalité entre les littéraux.
Cela veut dire que μ est unificateur principal de $\{B_x, x \in X\}$ et de $\{C_y, y \in Y\}$, et que de plus, en posant :

$$\mathcal{E} = \left(\bigvee_{i=1}^{a} A_i\right) \vee \left(\bigvee_{j \notin X} B_j\right) \vee \left(\bigvee_{k \notin Y} C_k\right) \vee \left(\bigvee_{l=1}^{e} D_l\right)$$

On obtient une résolution de $\mathcal{E}_1$ et $\mathcal{E}_3$ vers $\mathcal{E}$. Il ne reste plus qu'à introduire une reformulation de $\mathcal{E}_2$ vers $\mathcal{E}_3$ pour conclure la démonstration. □

[4] On fera attention sur la dernière égalité à ne pas inverser, car si certaines substitutions, comme les permutations de variables, sont inversibles, ce n'est pas le cas de toutes.

On peut alors démontrer le théorème :
Si $\forall T$ est un ensemble de clauses universelles non satisfiable,
Alors $\forall T$ n'est pas satisfiable dans la structure de Herbrand.
Donc pour tous les termes $t_1, \ldots, t_n$, il n'existe aucune valuation du 1er ordre saisfaisant $\{C(t_1, \ldots, t_n), C \in T\}$ dans cette même structure.
C'est à dire pour toute substitution σ, $\{\sigma(C), C \in T\}$ n'est pas satisfiable[5].
Il en découle que l'ensemble $\Gamma = \{\sigma(C), \sigma \text{ une substitution}, C \in T\}$ n'est pas satisfiable (au 1er ordre, dans la structure de Herbrand).
Donc cet ensemble Γ n'est pas satisfiable (propositionnelement) dans $\mathbb{P}$
Par complétude de la preuve par coupure, on en déduit qu'il existe une preuve par coupure de $\perp$ à partir de Γ
On peut alors appliquer le lemme pour trouver une preuve par résolution de $\perp$ à partir de T [6]

[5] Il y a un abus de langage ici : les formules concernées ne sont pas closes, donc la notion de satisfaisabilité n'est pas définie. Le mot "satisfiable" réfère donc à ce qui est décrit une ligne au dessus

[6] pour toute substitution σ, on a $\sigma(P) = \perp$ ssi $P = \perp$

Unification : trop long à ⇝ Écrire l'algo ds le plan.

↳ Regarder la preuve / la rédac° du Bracker.

↳ Expliquer l'intérêt de l'algo (présentat° du plan).

Questions : • Montrer que l'unificateur est bien principal.

• M que le mgu est UNIQUE.
(à renommage des variables près)

• Expliquer les complexités (cf. plan).

[DAG : graphe dirigé acyclique
cf. fig. 1]

Résolution – Ça veut dire quoi "unifier des $\boxed{\text{ensembles}}$" ?

Exercice : On suppose $\exists x\ A[x] \vee B[x]$

Montrer $(\exists x\ A[x]) \vee (\exists x\ B[x])$