Démonstration automatique de théorèmes

Nicolas Daire

Résumé

Dans ce TIPE on cherche à mettre au point un algorithme de démonstration automatique de théorèmes en logique du premier ordre.

On montre dans un premier des résultats théoriques permettant de comprendre les enjeux et les limites d'un tel algorithme.

On aborde ensuite l'implémentation de l'algorithme en utilisant la méthode de résolution.

1 Syntaxe

1.1 Langage

Définition 1.1. Un langage L est un ensemble de symboles constitué :
- des parenthèses ( ), des connecteurs $\neg \ \wedge \ \vee \ \Rightarrow \Leftrightarrow$, et des quantificateurs $\forall \ \exists$
- d'un ensemble $\mathfrak{V} = \{v_n\}$ infini dénombrable de variables
- d'un ensemble $\mathfrak{C}$ de symboles de constantes
- d'une réunion d'ensembles $\mathfrak{F}_n$ $(n \in \mathbb{N})$ de symboles de fonctions n-aires, où $\mathfrak{F}_0 = \mathfrak{C}$
- d'une réunion d'ensembles $\mathfrak{R}_n$ $(n \in \mathbb{N})$ de symboles de relations n-aires, où $\mathfrak{R}_0$ est l'ensemble des propositions atomiques qui contient parfois $\{\top, \bot\}$

1.2 Formules

Définition 1.2. On définit les termes, atomes et formules par induction structurelle :
- L'ensemble $\mathfrak{T}(L)$ des termes est le plus petit ensemble qui contient $\mathfrak{V}$ et qui est stable par $(t_1, \ldots, t_n) \mapsto f(t_1, \ldots, t_n)$ pour tout $f \in \mathfrak{F}_n$
- L'ensemble $\mathfrak{A}(L)$ des atomes est l'ensemble des $R(t_1, \ldots, t_n)$ où $R \in \mathfrak{R}_n$ et $(t_1, \ldots, t_n) \in \mathfrak{T}(L)^n$
- L'ensemble $\mathfrak{F}(L)$ des formules du premier ordre est le plus petit ensemble qui contient $\mathfrak{A}(L)$, et $\neg A, \ A \wedge B, \ A \vee B, \ A \Rightarrow B, \ A \Leftrightarrow B, \ \forall v_n \ A, \ \exists v_n \ A$ lorsqu'il contient A et B

Définition 1.3. Les occurences d'une variable v dans une formule F peuvent être liées ou libres :
- Si $F = \neg G, \ G \gamma H$ (γ *connecteur*), $\forall w \ G, \ \exists w \ G$ $(w \neq v)$ alors les occurences libres dans F sont les occurences libres dans G et H
- Si $F = \forall v \ G, \ \exists v \ G$ alors aucunes des occurences n'est liée

Les occurences non libres de v sont dites liées.

Une formule où aucune variable n'a d'occurence libre est dite close.

On écrit $F[v_1, \ldots, v_n]$ pour dire que les variables ayant une occurences libres sont parmi les v_i

La clôture universelle de $F[v_1, \ldots, v_n]$ est la formule close $\forall v_1 \ldots \forall v_n \ F$

Définition 1.4. On définit les substitutions dans les termes et les formules :
- Si t est un terme, on note $t_{u_1/v_1,\ldots,u_n/v_n}$ le terme où on a remplacé les occurences de v_i par le terme u_i
- Si F est une formule, on note $F_{u_1/v_1,\ldots,u_n/v_n}$ la formule où on a remplacé les occurences libres de v_i par le terme u_i.

On pourra noter noter $[v_1 \to u_1, \ldots, v_n \to u_n]$ voire $[u_1, \ldots, u_n]$ s'il n'y a pas ambiguïté.

1.3 Implémentation

On définit les types récursifs OCaml `term, atom, fol` pour représenter les termes, atomes et formules. On passe d'une chaîne de caractères représentant une formule (avec la notation polonaise ou infix) à la structure OCaml avec les fonctions du module `parse`. On peut ensuite afficher les formules sous forme de chaîne de caractères ou d'arbre avec les fonctions du module `disp`.

2 Sémantique

2.1 Structures et modèles

Définition 2.1. Une L-structure $\mathfrak{M}$ est constituée :
- d'un ensemble de base M
- d'éléments $\overline{c}^{\mathfrak{M}}$ interprétant les symboles de constantes $c \in \mathfrak{C}$
- de fonctions $\overline{f}^{\mathfrak{M}} : M^k \to M$ interprétant les symboles de fonctions $f \in \mathfrak{F}$
- de sous-ensembles $\overline{R}^{\mathfrak{M}} \subseteq M^k$ interprétant les symboles de relations $R \in \mathfrak{R}$

2.2 Satisfiabilité

Définition 2.2. Un environnement e est une fonction $\mathfrak{V} \to M$.

La valeur d'un terme t dans un environnement e est défini par induction :
- $V(c) = \overline{c}$
- $V(v) = e(v)$
- $V(f(v_1, \ldots, v_n) = f(V(v_1, \ldots, V(v_n))$

La valeur d'une formule F dans un environnement e est une fonction à valeurs dans $\{0, 1\}$ définie par induction avec le sens courant que l'on donne aux symboles.

Définition 2.3. On dit que $\mathfrak{M}$ satisfait F dans l'environnement e, noté $\mathfrak{M}, e \models F$, lorsque $V_{\mathfrak{M}}(F, e) = 1$.

On dit que $\mathfrak{M}$ satisfait F, ou $\mathfrak{M}$ est un modèle de F, noté $\mathfrak{M} \models F$, lorsque $\mathfrak{M}$ satisfait F dans tout environnement e, où de manière équivalente que $\mathfrak{M}$ satisfait la clôture universelle de F.

Définition 2.4.
- Une formule close F est universellement valide, noté $\models F$, si elle est satisfaite dans toute L-structure. Si $\models \neg F$, on dit que F est contradictoire.
- Deux formules F et G sont universellement équivalentes, noté $F \sim G$, si $\models F \Leftrightarrow G$.
- Une théorie est un ensemble de formules closes, souvent appelées axiomes.
- Une L-structure $\mathfrak{M}$ satisfait la théorie T, ou est un modèle de T, noté $\mathfrak{M} \models T$, lorsque $\mathfrak{M}$ satisfait toutes les formules de T.
- Une théorie T est consistente si elle admet un modèle, sinon elle est contradictoire.
- Une formule close F est conséquence sémantique de T si $\mathfrak{M} \models F$ pour tout modèle $\mathfrak{M}$ de T.
- Deux théories T_1 et T_2 sont équivalentes lorsque $\forall F \in T_1, T_2 \models F$ et inversement.

2.3 Compacité

On parle de logique propositionnelle lorsque L se réduit aux parenthèses, aux connecteurs et à $\mathfrak{P} = \mathfrak{R}_0$. On identifie alors une interprétation à une fonction δ de $\{0, 1\}^{\mathfrak{P}}$ appelée distribution de valeurs de vérité, et la valeur de l'interprétation à la fonction $\overline{\delta}$ de $\{0, 1\}^{\mathfrak{F}}$ qui coïncide avec δ sur $\mathfrak{P}$.

Théorème 2.1 (Tychonov)**.** *L'espace topologique produit d'une famille d'espaces compacts est compact.*

Théorème 2.2 (Compacité du calcul propositionnel)**.** *Un ensemble T de formules du calcul propositionnel est satisfiable si et seulement si T est finiment satisfiable.*

Démonstration. Montrons la contraposée du sens réciproque.
Si $F \in T$, note $\Delta(F) = \{\delta \in \{0, 1\}^{\mathfrak{P}} \mid \overline{\delta}(F) = 1\}$. T n'est pas satisfiable si et seulement si $\bigcap_{F \in T} \Delta(F) = \varnothing$. On munit $\{0, 1\}$ de la topologie discrète dans laquelle tous les sous-ensembles sont des ouverts-fermés. Les formules de $F \in T$ sont finies donc ne dépendent que d'un nombre fini de variables propositionnelles donc $\Delta(F) = \bigcup_{\epsilon \in E} \{\delta \in \{0, 1\}^{\mathfrak{P}} \mid \delta \restriction_I = \epsilon\}$ avec $I \subseteq \mathfrak{P}$ fini et $E \subseteq \{0, 1\}^I$, donc les $\Delta(F)$ sont des fermés de $\{0, 1\}^{\mathfrak{P}}$ comme unions finies d'ouverts-fermés. D'après le théorème de Tychonov l'espace produit $\{0, 1\}^{\mathfrak{P}}$ est compact donc il existe une partie finie $S \subseteq T$ telle que $\bigcap_{F \in S} \Delta(F) = \varnothing$, T n'est donc pas finiment satisfiable. □

Théorème 2.3 (Compacité du calcul des prédicats)**.** *Une théorie T dans un langage du premier ordre est consistente si et seulement si T est finiment consistente.*

Lemme 2.4. *Soient $A_1, \ldots, A_n$ des variables propositionnelles, $J[A_1, \ldots, A_n]$ une formule propositionnelle, et $F_1, \ldots, F_n$ des formules du premier ordre. Si J est une tautologie, alors $\models J[F_1, \ldots, F_n]$.*

Lemme 2.5. *Si G est une sous-formule de F, et $G \sim G'$, alors $F \sim F'$ obtenue à partir de F en remplaçant une occurence de G par G'.*

Théorème 2.6. *Toute formule du premier ordre est universellement équivalente à une formule n'utilisant que les symboles de connecteur d'un système complet ainsi que l'un des symboles de quantification $\forall$ et $\exists$.*

Une formule est sous forme prénexe si elle s'écrit $Q_1 v_1 \dots Q_n v_n\, F$ où $Q_1, \dots, Q_n$ sont des quantificateurs, $v_1, \dots, v_n$ sont des variables et F est une formule sans quantificateur. On appelle alors $Q_1 v_1 \dots Q_n v_n$ son préfixe.
Une formule sous forme prénexe est polie si les variables quantifiées sont deux à deux distinctes.
Une formule est universelle (resp. existentielle) si elle ne contient pas de quantificateur existentiel (resp. universel).

Théorème 2.7. *Toute formule du premier ordre est universellement équivalente à une formule sous forme prénexe polie.*

Démonstration. On en donne une preuve constructive par induction.
- Si $F = \neg F'$, par récurrence $F' \sim Q_1 v_1 \dots Q_n v_n\ F''$ où F'' est sans quantificateur, il suffit de prendre $\overline{Q_1} v_1 \dots \overline{Q_n} v_n\ \neg F''$.
- Si $F = F' \vee G'$, par récurrence $F' \sim Q_1 v_1 \dots Q_m v_m\ F''$ et $G' \sim Q_{m+1} v_{m+1} \dots Q_{m+n} v_{m+n}\ G''$ où F'' et G'' sont sans quantificateurs, soient $w_1, \dots, w_{m+n}$ des variables n'ayant aucune occurence dans F'' et G'', il suffit de prendre $Q_1 w_1 \dots Q_{m+n} w_{m+n}\ F''_{w_1/v_1,\dots,w_m/v_m} \vee G''_{w_{m+1}/v_{m+1},\dots,w_{m+n}/v_{m+n}}$.
- Si $F = \exists v\ F'$, par récurrence $F' \sim Q_1 v_1 \dots Q_n v_n\ F''$ où F'' est sans quantificateur. Si $v \notin \{v_1, \dots, v_n$, il suffit de prendre $\exists v Q_1 v_1 \dots Q_n v_n\ F''$, sinon il suffit de prendre $Q_1 v_1 \dots Q_n v_n\ F''$.

□

Remarque. En utilisant les équivalences $\forall v F \wedge \forall v G \sim \forall v (F \wedge G)$ et $\exists v F \vee \exists v G \sim \exists v (F \vee G)$ et en renommant les variables, on peut obtenir une forme prénexe polie utilisant moins de variables.

Soit F une formule du premier ordre, $Q_1 v_1 \dots Q_n v_n\ G$ où G est sans quantificateur une forme prénexe polie équivalente.

Pour $i \in [\![1, n]\!]$ tel que $Q_i = \exists$, soient $j_1, \dots, j_k$ les indices inférieurs à i des variables quantifiées universellement, on introduit un nouveau symbole de fonction k-aire f^i_S appelée fonction de Skolem et un terme $t^i_S = f^i_S(v_{j_1}, \dots, v_{j_k})$. On note L_S l'extension de L enrichie des symboles de fonction de Skolem.

Soient $i_1, \dots, i_k$ les indices des variables quantifiées existentiellement, $j_1, \dots, j_{k'}$ les indices des variables quantifiées universellement, on appelle forme de Skolem la formule $F_S = Q_{j_1} v_{j_1} \dots Q_{j_{k'}} v_{j_{k'}}\ G_{t^{i_1}_S/v_{i_1},\dots,t^{i_k}_S/v_{i_k}}$ du langage L_s.

Théorème 2.8. *Soit F une formule du premier ordre. F et sa forme de Skolem F_S sont équisatisfiables.*

Démonstration. Si F admet un modèle $\mathfrak{M}$, on peut construire une extension $\mathfrak{N}$ sur le langage L_S qui soit un modèle de F_S en interprétant les fonctions de Skolem f_s par des fonctions de choix obtenues avec l'axiome du choix par récurrence sur le nombre de quantificateurs existentiels.

Si F_S admet un modèle $\mathfrak{N}$, F est satisfaite dans la restriction $\mathfrak{M}$ de $\mathfrak{N}$ privée des symboles de fonction de Skolem. □

3 Démonstration

3.1 Séquents

Pour formaliser les démonstrations en calcul des prédicats on se munit d'un ensemble de règles de démonstration :

Deux règles de déduction :
- Le modus ponens : $\dfrac{F,\ F \Rightarrow G}{G}$
- La règle de généralisation : $\dfrac{F}{\forall v\ F}$

Des axiomes logiques :
- Les tautologies
- Les axiomes de quantificateurs :
  - $\exists v\ F \Leftrightarrow \neg \forall v\ \neg F$
  - $\forall v\ (F \Rightarrow G) \Rightarrow (F \Rightarrow \forall v\ G)$ si v n'a pas d'occurence dans F
  - $\forall v\ F \Rightarrow F_{t/v}$ si les occurences libres de v ne sont pas dans le champ d'un quantificateur liant une variable de t

Définition 3.1. Soit T une théorie et F une formule close. On appelle démonstration de F dans T une suite $\mathfrak{D} = (F_0, \dots, F_n)$ avec $F_n = F$ vérifiant pour tout i une des conditions suivantes :
- $F_i \in T$
- F_i est un axiome logique
- F_i se déduit à partir de formules parmi $F_0, \dots, F_{i-1}$ et d'une règle de déduction

S'il existe un démonstration de F dans T on dit que F est conséquence syntaxique de T, ou que T démontre F, noté $T \vdash F$.

Définition 3.2.
- T est cohérente si on n'a jamais $T \vdash F$ et $T \vdash \neg F$, ou de manière équivalente $T \nvdash \bot$, ou encore il existe une formule F telle que $T \nvdash F$.
- T est complète si T est cohérente et pour toute formule close F, $T \vdash F$ ou $T \vdash \neg F$.

Théorème 3.1. *Soit F une formule close. Si $T, F \vdash G$, alors $T \vdash F \Rightarrow G$.*

Démonstration. Si $(G_0, \ldots, G_n)$ est une démonstration de G dans $T \cup \{F\}$, on peut déduire une démonstration de $F \Rightarrow G$ dans T en ajoutant des étapes dans la suite $(F \Rightarrow G_0, \ldots, F \Rightarrow G_n)$. □

Corollaire 3.1.1. *$T \vdash F$ si et seulement si $T \cup \{\neg F\}$ est incohérente.*

Démonstration. Si $T \cup \{\neg F\}$ est incohérente, en particulier $T, \neg F \vdash F$ donc $T \vdash \neg F \Rightarrow F$, et $(\neg F \Rightarrow F) \Rightarrow F$ est une tautologie, donc par modus ponens $T \vdash F$. □

Définition 3.3. Une preuve par réfutation de F dans T est la démonstration $T, F \vdash \bot$.

Théorème 3.2. *Si $T \vdash F$, alors $T \models F'$ où F' est une clôture universelle de F.*

Corollaire 3.2.1. *Si T est consistante, alors T est cohérente.*

Théorème 3.3. *Si T est cohérente, alors T est consistante.*

Démonstration. On construit une théorie $Th \supseteq T$ sur $L' \supseteq L$ qui vérifie :
- Th est complète
- Si $F[v]$ est une formule de L' ayant v comme seule variable libre, alors il existe $c_F \in \mathfrak{C}(L')$ tel que $Th \vdash \exists y\ F[y] \Rightarrow F[c_F]$

Puis on construit un modèle de Th qui donnera un modèle de T. □

Théorème 3.4 (Complétude du calcul des prédicats)**.** *Soient T une théorie et F une formule close, $T \vdash F$ si et seulement si $T \models F$.*

Démonstration.
$T \vdash F$
ssi $T \cup \{\neg F\}$ est incohérente
ssi $T \cup \{\neg F\}$ est inconsistante
ssi aucun modèle ne satisfait $\neg F$ ssi tout modèle satisfait F ssi $T \models F$ □

Démonstration. (Théorème de compacité du calcul des prédicats) Si T est inconsistante, alors T est incohérente, $T \vdash \bot$ donc il existe un sous-ensemble fini de formules $T' \subseteq T$ tel que $T' \vdash \bot$. Par le théorème de complétude $T' \models \bot$ donc T' est inconsistante. □

3.2 Conversion

Définition 3.4.

- Un littéral L est un atome A ou sa négation $\neg A$.
- Une clause est une disjonction de littéraux $L_1 \vee \cdots \vee L_n$. On peut aussi écrire $A_1 \wedge \cdots \wedge A_m \Rightarrow B_1 \vee \cdots \vee B_n$ où $A1, \ldots, A_m, B_1, \ldots, B_n$ sont des atomes, auquel cas on appelle $A_1 \wedge \cdots \wedge A_m$ la prémisse et $B_1 \vee \cdots \vee B_n$ la conclusion de la clause.
- Une formule du premier ordre sous forme normale clausale est une conjonction de clauses $\bigwedge_i \bigvee_j L_{ij}$.

Théorème 3.5. *Toute formule F admet une forme normale clausale équisatisfiable.*

Démonstration. Il suffit pour cela de calculer la forme de Skolem F_S de F, puis d'éliminer les quantificateurs universels de F_s, enfin de mettre F_s sous forme normale conjonctive et de rassembler les sous-formules en clauses. □

On introduit alors les types OCaml `literal, clause, cnf` pour représenter les formules sous forme clausale comme `list list literal`.

3.3 Démonstration par coupure

On introduit une procédure de décision du calcul propositionnel par réfutation. On cherche une démonstration de la clause vide à partir d'un ensemble de formules sous forme clausale au moyen de deux règles.

— La règle de simplification : $\dfrac{A \vee A \vee B_1 \vee \cdots \vee B_n}{B_1 \vee \cdots \vee B_n}$ où $A, B_1, \ldots, B_n$ sont des littéraux.

— La règle de coupure qui généralise le modus ponens : $\dfrac{A \vee B_1 \vee \cdots \vee B_m,\ \neg A \vee C_1 \vee \cdots \vee C_n}{B_1 \vee \ldots B_m \vee C_1 \vee \cdots \vee C_n}$ où $A, B_1, \ldots, B_m, C_1, \ldots, C_n$ sont des littéraux.

On peut combiner les deux règles : $\dfrac{A \vee \cdots \vee A \vee B_1 \vee \cdots \vee B_m,\ \neg A \vee \cdots \vee \neg A \vee C_1 \vee \cdots \vee C_n}{B_1 \vee \ldots B_m \vee C_1 \vee \cdots \vee C_n}$

Théorème 3.6 (Complétude de la démonstration par coupure)**.** *Tout ensemble de clauses insatisfiable est réfutable par coupure.*

Démonstration. Soit Γ un ensemble de clauses insatisfiable.

Par compacité du calcul propositionnel on peut supposer Γ fini. On peut de plus supposer que Γ ne contient pas de tautologies ni la clause vide, et que toutes ses clauses sont simplifiées. On peut également supposer que toute variable propositionnelle de Γ apparaît à la fois positivement et négativement.

Si C est une clause, on note C^- sa prémisse et C^+ sa conclusion.

On raisonne par récurrence sur le nombre de variables propositionnelles. Soit A une variable propositionnelle apparaissant dans Γ. On introduit les ensembles :

— $\Gamma_0 = \{C \mid A \text{ n'a pas d'occurrence dans } C\}$

— $\Gamma^- = \{C \mid A \text{ n'a pas d'occurrence dans } C^-\} = \{C_i\}$

— $\Gamma^+ = \{C \mid A \text{ n'a pas d'occurrence dans } C^+\} = \{D_j\}$

On applique exhaustivement la coupure sur les clauses C_i et D_j pour obtenir les clauses E_{ij}. On pose $\Gamma' = \Gamma \cup \{E_{ij}\}$, dans lequel A n'a pas d'ocurrence. Il suffit alors de montrer que l'ensemble Γ' est insatisfiable pour conclure par récurrence. □

Pour déterminer si un ensemble Γ est satisfiable, on élimine d'abord les tautologies et on simplifie les clauses, puis on applique exhaustivement la règle de coupure sur tous les couples de clauses. Le nombre de clauses simplifiées est fini donc la procédure termine, il suffit alors de regarder si on a pu dériver la clause vide.

3.4 Démonstration par résolution

On va adapter la méthode de démonstration par coupure au calcul des prédicats pour obtenir une procédure de réfutation.

Définition 3.5. Soit $S = \{(t_1, u_1), \ldots, (t_n, u_n)\}$ un ensemble de couples de termes. On appelle unificateur de S toute substitution σ telle que $\sigma(t_i) = \sigma(u_i)$ pour tout i.

On appelle unificateur principal de S tout unificateur π tel que si σ est un unificateur de S il existe une substitution τ telle que $\sigma = \tau \circ \pi$.

Théorème 3.7. *Si S admet un unificateur, alors S admet un unificateur principal.*

On va donner une preuve constructive sous la forme d'un algorithme qui détermine si un système admet un unificateur, et renvoie un unificateur principal le cas échéant.

On introduit pour cela deux procédures :

— Réduction : si $(f(t_1, \ldots, t_n), f(u_1, \ldots, u_n)) \in S$, on remplace le couple dans S par les couples $(t_1, u_1), \ldots, (t_n, u_n)$.

— Elimination : si $(v, t) \in S$ où v est une variable, on applique la substitution $(v \Rightarrow t)$ à chaque membre des autres couples de S.

On dit que le S est résolu si $S = \{(v_1, t_1), \ldots, (v_n, t_n)\}$ où les v_i sont des variables n'apparaîssant qu'une seule fois dans S. La substitution $(v_1 \to t_1, \ldots, v_n \to t_n)$ est alors un unificateur principal.

On exécute les étapes suivantes tant que c'est possible, sinon on termine avec succès.

— Si $(t, v) \in S$ où v est une variable mais pas t, on remplace le couple par (v, t).

— Si $(v, v) \in S$ où v est une variable, on retire le couple de S.

— Si $(t, u) \in S$ où t et u ne sont pas des variables, si t et u ne commencent pas par les mêmes symboles de fonction on termine avec échec, sinon on applique la réduction.

— Si $(v, t) \in S$ où v est une variable qui a une autre occurrence dans S et $v \neq t$, si v a une occurrence dans t on termine avec échec, sinon on applique l'élimination.

Théorème 3.8. *L'algorithme décrit termine toujours quelque soit l'ordre des opérations, et renvoie un unificateur principal si et seulement si le système admet un unificateur.*

On a ainsi un algorithme pour le problème d'unification, mais il a une complexité temporelle exponentielle.

Pour remédier on représente le problème par des graphes acycliques : on associe un sommet à chaque sous-terme, en réservant un unique sommet pour chaque variable, et (t, u) est une arête si et seulement si t est une fonction qui possède u comme argument.

On maintient une structure Union-Find qui renseigne les classes d'équivalence des variables et des sous-termes substitués par les mêmes termes. Ainsi pendant l'exécution on ne cherche à unifier que les parents, les termes unifiés deviennent donc inaccessibles pendant la recherche. Il suffit ensuite de vérifier que le graphe obtenu est acyclique. Le cas échéant on peut facilement reconstruire l'unificateur principal.

Théorème 3.9. *Le nouvel algorithme est correct et a une complexité temporelle en $O(mG(n))$ où m et n sont les nombres d'arêtes et de sommets dans les graphes donnés en entrée, et G est la fonction inverse d'Ackermann, qui est inférieure à 5 en pratique. L'algorithme obtenu est ainsi presque linéaire.*

Pour la réfutation par résolution on introduit la règle de résolution, l'analogue de la règle de coupure pour la logique du premier ordre, mais au lieu de résoudre sur des atomes identiques on résoud sur des atomes unifiables.

Soient C et D deux clauses. On construit la clause D' en renommant les variables de D de sorte que C et D' n'aient aucune variable en commun. Soient $X \subseteq C^+$ et $Y \subseteq D'^-$ non vides tels que $X \cup Y$ soit unifiable. Alors la règle de résolution s'écrit $\dfrac{C,\ D'}{\pi_{X\cup Y}(C\backslash X,\ D'\backslash Y)}$ où π est un unificateur principal.

Théorème 3.10. *Si E se déduit de C, D, tout modèle de C, D est aussi un modèle de E.*

Définition 3.6. Soi Γ un ensemble de clause. On appelle réfutation de Γ toute suite de clause $(C_1, \dots, C_n)$ se terminant par la clause vide telle que pour tout i, $C_i \in \Gamma$ ou C_i se déduit de deux clauses antérieures de la suite. On dit que Γ est réfutable s'il existe une réfutation de Γ.

Théorème 3.11 (Complétude de la méthode de résolution)**.** *Un ensemble de clauses Γ est réfutable si et seulement si Γ est inconsistant.*

On introduit pour cela la méthode Herbrand qui permet de ramener la logique du premier ordre à la logique propositionnelle.

On cherche un modèle dont l'ensemble de base est $\mathfrak{T}(L)$, et les symboles fonctions ont leur interprétation naturelle. Pour tout $\delta \in \{0,1\}^{\mathfrak{A}(L)}$ on définit une interprétation $\mathfrak{I}_\delta$ dans laquelle si R est un symbole de relation et $t_1, \dots, t_n$ des termes, $(t_1, \dots, t_n) \in R$ si et seulement si $\delta(R(t_1, \dots, t_n)) = 1$.

Ainsi pour savoir si $\Gamma \models F$, on essaie de réfuter $\Gamma, \neg F$. On maintient deux listes correspondant aux clauses utilisées et non utilisée, et on défile la liste non utilisées en résolvant chaque clause avec toutes les clauses de la liste utilisées, jusqu'à déduire la clause vide ou vider la liste non utilisées.

Mais cette recherche peut ne jamais terminer, le calcul des prédicats est semi-décidable.

Il existe néanmoins diverses optimisations de la méthode de résolution.

Définition 3.7. Une clause C subsume une clause D, noté $C \leqslant D$, s'il existe une substitution σ telle que $\sigma(C) \subseteq D$.

On peut déterminer si $C \leqslant D$ à l'aide d'un algorithme de backtracking en construisant la substitution au fur et à mesure.

Théorème 3.12. *Si $C \leqslant C'$, alors tout résolvant de C', D est subsumé par C ou un résolvant de C, D.*

Corollaire 3.12.1. *Si $\Gamma \vdash C'$, alors il existe une démonstration $\Gamma \vdash C \leqslant C'$ où aucune clause n'est subsumée par une clause antérieure.*

On peut donc ignorer les résolvants subsumés par des clauses antérieures, et remplacer les clauses subsumées par les résolvants dans la liste utilisées.

Théorème 3.13. *Soit $\mathfrak{M}$ une L-structure. Si Γ est réfutable, il existe une réfutation qui ne résoud deux clauses que si l'une au moins n'est pas satisfaite dans $\mathfrak{M}$.*

Si Γ satisfait les deux clauses la contradiction doit venir des autres clauses, donc on peut trouver une telle réfutation.

La résolution positive est un cas particulier où l'on choisit une interprétation dans laquelle toutes les formules atomiques sont fausses, les résolutions font donc toujours intervenir au moins une clause constituée uniquement de littéraux positifs.