Logique propositionnelle

David Delahaye

Faculté des Sciences
David.Delahaye@lirmm.fr

Licence L3 2021-2022

La logique en informatique : preuve de programmes

Ligne de métro 14 (Meteor)

- Ouverte en 1998 ;
- Développée par Siemens (Matra) ;
- Utilisation de la méthode B ;
- Théorie des ensembles en première page des journaux ;
- Siemens continue à développer des métros sans conducteur.

La logique en informatique : preuve de programmes

JavaCard (Gemalto)

- Formalisation de l'architecture JavaCard ;
- Utilisation de Coq.

Le compilateur certifié CompCert (Inria, Gallium)

- Produit du code assembleur pour PowerPC, ARM, et x86 ;
- Développé en Coq et certifié correct.

Projet L4.verified (NICTA)

- Formalisation du micro-noyau seL4 ;
- Utilisation d'Isabelle/HOL.

Preuves formelles

Plusieurs pré-requis

- Avoir une bonne connaissance de la sémantique de son langage ;
- Être capable d'exprimer cette sémantique formellement ;
- Savoir spécifier précisément le comportement de son programme ;
- Faire en sorte que la spécification soit totale.

Plusieurs langages en jeu

- Le langage de programmation ;
- Le langage de spécification ;
- Le langage de preuve.

Si ces trois langages sont réunis au sein du même environnement, c'est beaucoup plus pratique pour le développeur !

Spécification

Qu'est-ce que c'est ?

- C'est le « quoi » du programme, ce qu'il doit faire ;
- Peut-être exprimé dans le langage naturel (mais ambigu) :
  - Exemple : « ce programme calcule la racine carrée ».
- Plus formellement : spécification = type d'un programme.

Plusieurs degrés de spécifications

- Spécifications partielles :
  - Exemple : `sqrt : float -> float` ;
  - Donne de l'information mais pas assez ;
  - Beaucoup de fonctions ont ce type (pas seulement racine carrée).
- Spécifications totales :
  - Exemple : $\forall x \in \mathbb{R}^+ . f(x) \geq 0 \land f(x) \times f(x) = x$;
  - Seule racine carrée vérifie cette proposition ;
  - Nécessite un langage basé sur la logique.

Preuves

Objectifs

- Mettre en adéquation un programme et sa spécification ;
- Apporter une garantie sur l'exécution du programme.

Remarques

- C'est plus simple de faire des preuves sur des programmes fonctionnels.
- Le fonctionnel sert aussi à encoder les preuves pour certains outils.
- Outils basé sur du fonctionnel : Coq, HOL, PVS, etc.
- Outils basé sur de l'impératif : Atelier B.

Peut-on automatiser ce processus ?

- Pas totalement (problème semi-décidable) ;
- Certains fragments sont décidables :
  - Logique propositionnelle, arithmétique linéaire, réels, géométrie, etc.

Une preuve triviale

Spécification

- On cherche à écrire une fonction f telle que :
  $\forall x \in \mathbb{N}. f(x) = x \times x$

Programme

- On considère le programme (fonction) suivant :
  $g(x) = x \times x$

Preuve d'adéquation

- On doit prouver que le programme g vérifie la spécification :
  $\forall x \in \mathbb{N}. g(x) = x \times x$
- On « déplie » la définition de g :
  $\forall x \in \mathbb{N}. x \times x = x \times x$
- Ce qui est trivial.

Une preuve plus difficile

Spécification

- La même que précédemment, c'est-à-dire :
  $\forall x \in \mathbb{N}. f(x) = x \times x$

Programme

- On considère le programme (fonction) suivant :
  $$h(x, i) = \begin{cases} x, & \text{si } i = 0, 1 \\ x + h(x, i - 1), & \text{sinon} \end{cases}$$
  $$g(x) = h(x, x)$$

Preuve ?

- Par récurrence.

Mécanisation des preuves

Outils d'aide à la preuve

- Beaucoup d'outils existants ;
- Développés par des équipes de recherche ;
- Mais pas que : Atelier B (Alstom, ClearSy) ;
- Mécanisation ne signifie pas automatisation :
  - Outils de preuve interactive (Coq, HOL, etc.) ;
  - Outils de preuve automatique (Vampire, Zenon, etc.).

Transfert industriel ?

- Difficile au début ;
- Investit les milieux R&D progressivement ;
- Plus facile si l'outil vient d'une initiative industrielle (Atelier B) ;
- Plusieurs succès académiques récents changent la donne.

Logique propositionnelle

Il s'agit d'un rappel

- Cours de HAI304I (Logique propositionnelle) en L2 ;
- Sensiblement la même chose ;
- Quelques différences essentiellement dans les notations.

Définition préliminaire

- $\mathcal{V} \equiv$ ensemble de variables de propositions A, B, etc.

Formules

- Plus petit ensemble $\mathcal{F}$ t.q. :
  - Si $A \in \mathcal{V}$ alors $A \in \mathcal{F}$;
  - $\perp, \top \in \mathcal{F}$;
  - Si $\Phi \in \mathcal{F}$ alors $\neg\Phi \in \mathcal{F}$;
  - Si $\Phi, \Phi' \in \mathcal{F}$ alors $\Phi \wedge \Phi', \Phi \vee \Phi', \Phi \Rightarrow \Phi', \Phi \Leftrightarrow \Phi' \in \mathcal{F}$.

Logique propositionnelle

Associativité des connecteurs

- $\wedge$, $\vee$, et $\Leftrightarrow$ associent à gauche :
  - $A \wedge B \wedge C \equiv (A \wedge B) \wedge C$.
- $\Rightarrow$ associe à droite :
  - $A \Rightarrow B \Rightarrow C \equiv A \Rightarrow (B \Rightarrow C)$.

Précédence des connecteurs

- On a la précédence suivante : $\neg \succ \wedge \succ \vee \succ \Rightarrow \succ \Leftrightarrow$;
- Exemples :
  - $A \wedge B \Rightarrow C \equiv (A \wedge B) \Rightarrow C$;
  - $A \wedge \neg B \vee C \Rightarrow D \equiv ((A \wedge \neg B) \vee C) \Rightarrow D$;
  - $A \Rightarrow B \Leftrightarrow C \wedge D \equiv (A \Rightarrow B) \Leftrightarrow (C \wedge D)$.

Sémantique

Logique propositionnelle classique

- Chaque formule est censée être soit vraie, soit fausse ;
- Ensemble des valeurs de vérité : $\mathcal{B} = \{T, F\}$ (booléens), où $T \neq F$;
- Tables de vérité :

A	B	$\neg_{\mathcal{B}} A$	$A \wedge_{\mathcal{B}} B$	$A \vee_{\mathcal{B}} B$	$A \Rightarrow_{\mathcal{B}} B$	$A \Leftrightarrow_{\mathcal{B}} B$
F	F	T	F	F	T	T
F	T	T	F	T	T	F
T	F	F	F	T	F	F
T	T	F	T	T	T	T

- $\wedge_{\mathcal{B}}$, $\vee_{\mathcal{B}}$, $\Rightarrow_{\mathcal{B}}$, et $\Leftrightarrow_{\mathcal{B}}$: fonctions de $\mathcal{B} \times \mathcal{B}$ vers $\mathcal{B}$;
- $\neg_{\mathcal{B}}$: fonction de $\mathcal{B}$ vers $\mathcal{B}$.

Sémantique

Définition

- Affectation (ou interprétation) ρ : application de l'ensemble $\mathcal{V}$ des variables de propositions vers $\mathcal{B}$;
- La sémantique $[\![\Phi]\!]_\rho$ d'une formule Φ dans l'affectation ρ est définie par récurrence structurelle sur Φ par :
  - Si $A \in \mathcal{V}$ alors $[\![A]\!]_\rho = \rho(A)$;
  - $[\![\top]\!]_\rho = T$, $[\![\bot]\!]_\rho = F$;
  - Si $\Phi \in \mathcal{F}$ alors $[\![\neg\Phi]\!]_\rho = \neg_\mathcal{B}[\![\Phi]\!]_\rho$;
  - Si $\Phi, \Phi' \in \mathcal{F}$ alors :
    - $[\![\Phi \wedge \Phi']\!]_\rho = [\![\Phi]\!]_\rho \wedge_\mathcal{B} [\![\Phi']\!]_\rho$;
    - $[\![\Phi \vee \Phi']\!]_\rho = [\![\Phi]\!]_\rho \vee_\mathcal{B} [\![\Phi']\!]_\rho$;
    - $[\![\Phi \Rightarrow \Phi']\!]_\rho = [\![\Phi]\!]_\rho \Rightarrow_\mathcal{B} [\![\Phi']\!]_\rho$;
    - $[\![\Phi \Leftrightarrow \Phi']\!]_\rho = [\![\Phi]\!]_\rho \Leftrightarrow_\mathcal{B} [\![\Phi']\!]_\rho$.

Sémantique

Vocabulaire

- Soit Φ une formule et ρ une affectation ;
- ρ est un modèle de Φ ou ρ satisfait Φ, noté $\rho \models \Phi$, ssi $[\![\Phi]\!]_\rho = T$;
- Un ensemble G de formules entraîne Φ, noté $G \models \Phi$, ssi toutes les affectations satisfaisant toutes les formules de G en même temps (les modèles de G) sont aussi des modèles de Φ, c'est-à-dire quand $\rho \models \Phi'$ pour tout $\Phi' \in G$ implique $\rho \models \Phi$;
- Φ est valide ssi Φ est vraie dans toute affectation ($[\![\Phi]\!]_\rho = T$ pour tout ρ, noté $\models \Phi$), et est invalide sinon ;
- Une formule valide est aussi appelée une tautologie ;
- Φ est satisfiable ssi elle est vraie dans au moins une affectation ($[\![\Phi]\!]_\rho = T$ pour un certain ρ, c'est-à-dire elle a un modèle), et est insatisfiable sinon.

Sémantique

Vocabulaire

- Toutes les formules valides sont satisfiables, et toutes les formules insatisfiables sont invalides ;
- Ceci divise l'espace des formules en trois catégories :
  - Les valides (toujours vraies) ;
  - Les insatisfiables (toujours fausses) ;
  - Les formules contingentes (parfois vraies, parfois fausses).
- La validité et l'insatisfiabilité se correspondent via négation : Φ est valide ssi $\neg\Phi$ est insatisfiable, Φ est insatisfiable ssi $\neg\Phi$ est valide.

Sémantique

Exemples

- $A \wedge B \Rightarrow A$ est valide, c'est-à-dire $\models A \wedge B \Rightarrow A$;
- On a : $A \wedge B \models A$;
- $A \wedge B \Rightarrow C$ est contingent ;
- $A \wedge \neg A$ est insatisfiable.

Exercice

- Le démontrer (à faire à la maison).

Preuves

Séquents

- Un séquent de Gentzen est un couple Γ, Δ d'ensembles finis de formules, noté $\Gamma \vdash \Delta$.

Système de preuve

- Calcul des séquents de Gentzen ;
- Version propositionnelle : LK_0.

Calcul des séquents propositionnel (LK_0)

Règles

$$\frac{}{\Gamma, A \vdash \Delta, A}\,\text{ax} \qquad \frac{\Gamma \vdash A \qquad \Gamma, A \vdash B}{\Gamma \vdash B}\,\text{cut}$$

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\,\text{cont}_{\text{left}} \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\,\text{cont}_{\text{right}}$$

Calcul des séquents propositionnel (LK_0)

Règles

$$\frac{\Gamma \vdash \Delta, A \qquad \Gamma, B \vdash \Delta}{\Gamma, A \Rightarrow B \vdash \Delta} \Rightarrow_{\text{left}} \qquad \frac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta, A \Rightarrow B} \Rightarrow_{\text{right}}$$

$$\frac{\Gamma \vdash \Delta, A, B, C \qquad \Gamma, A, B \vdash \Delta, C}{\Gamma, A \Leftrightarrow B \vdash \Delta} \Leftrightarrow_{\text{left}}$$

$$\frac{\Gamma, A \vdash \Delta, B \qquad \Gamma, B \vdash \Delta, A}{\Gamma \vdash \Delta, A \Leftrightarrow B} \Leftrightarrow_{\text{right}}$$

Calcul des séquents propositionnel (LK_0)

Règles

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge_{\text{left}} \qquad \frac{\Gamma \vdash \Delta, A \qquad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B}\ \wedge_{\text{right}}$$

$$\frac{\Gamma, A \vdash \Delta \qquad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta}\ \vee_{\text{left}} \qquad \frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \vee B}\ \vee_{\text{right}}$$

$$\frac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\ \neg_{\text{left}} \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\ \neg_{\text{right}}$$

$$\frac{}{\Gamma, \bot \vdash \Delta}\ \bot_{\text{left}} \qquad \frac{}{\Gamma \vdash \Delta, \top}\ \top_{\text{right}}$$

Exemple de preuve

Une preuve simple

$$A, B \vdash A$$
$$A \wedge B \vdash A$$
$$\vdash A \wedge B \Rightarrow A$$

Exemple de preuve

Une preuve simple

$$\dfrac{\dfrac{A, B \vdash A}{A \wedge B \vdash A}}{\vdash A \wedge B \Rightarrow A}\Rightarrow_{\text{right}}$$

Exemple de preuve

Une preuve simple

$$\dfrac{\dfrac{A, B \vdash A}{A \wedge B \vdash A}\wedge_{\text{left}}}{\vdash A \wedge B \Rightarrow A}\Rightarrow_{\text{right}}$$

Exemple de preuve

Une preuve simple

$$\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,\text{ax}}{A \wedge B \vdash A}\,\wedge_{\text{left}}}{\vdash A \wedge B \Rightarrow A}\,\Rightarrow_{\text{right}}$$

Exemple de preuve

Une autre preuve

$$\dfrac{\dfrac{\dfrac{A, B \vdash A \qquad A, B \vdash B}{A, B \vdash A \wedge B}}{A \vdash B \Rightarrow A \wedge B}}{\vdash A \Rightarrow B \Rightarrow A \wedge B}$$

Exemple de preuve

Une autre preuve

$$\dfrac{\dfrac{A, B \vdash A \qquad A, B \vdash B}{A, B \vdash A \wedge B}}{\dfrac{A \vdash B \Rightarrow A \wedge B}{\vdash A \Rightarrow B \Rightarrow A \wedge B} \Rightarrow_{\text{right}}}$$

Exemple de preuve

Une autre preuve

$$\dfrac{\dfrac{\dfrac{A, B \vdash A \qquad A, B \vdash B}{A, B \vdash A \wedge B}}{A \vdash B \Rightarrow A \wedge B} \Rightarrow_{\text{right}}}{\vdash A \Rightarrow B \Rightarrow A \wedge B} \Rightarrow_{\text{right}}$$

Exemple de preuve

Une autre preuve

$$\dfrac{\dfrac{\dfrac{A, B \vdash A \qquad A, B \vdash B}{A, B \vdash A \wedge B}\,\wedge_{\text{right}}}{A \vdash B \Rightarrow A \wedge B}\,\Rightarrow_{\text{right}}}{\vdash A \Rightarrow B \Rightarrow A \wedge B}\,\Rightarrow_{\text{right}}$$

Exemple de preuve

Une autre preuve

$$
\dfrac{\dfrac{}{A, B \vdash A}\,\text{ax} \qquad A, B \vdash B}{\dfrac{A, B \vdash A \wedge B}{\dfrac{A \vdash B \Rightarrow A \wedge B}{\vdash A \Rightarrow B \Rightarrow A \wedge B}\Rightarrow_{\text{right}}}\Rightarrow_{\text{right}}}\wedge_{\text{right}}
$$

Exemple de preuve

Une autre preuve

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,\text{ax} \qquad \dfrac{}{A, B \vdash B}\,\text{ax}}{A, B \vdash A \wedge B}\,\wedge_{\text{right}}}{A \vdash B \Rightarrow A \wedge B}\,\Rightarrow_{\text{right}}}{\vdash A \Rightarrow B \Rightarrow A \wedge B}\,\Rightarrow_{\text{right}}
$$

Propriétés

Prouvabilité

- $\Gamma \vdash \Delta$ est prouvable dans LK_0, noté $\Gamma \vdash_{LK_0} \Delta$, ssi il existe une dérivation dans LK_0 se terminant sur $\Gamma \vdash \Delta$.

Correction

- Notation : $\Gamma \models \Delta \equiv \Gamma \models \bigvee_{\Phi \in \Delta} \Phi$;
- Si $\Gamma \vdash_{LK_0} \Delta$ alors $\Gamma \models \Delta$.

Complétude

- Si $\Gamma \models \Delta$ alors $\Gamma \vdash_{LK_0} \Delta$.

Élimination des coupures

- Il existe un algorithme qui prend une preuve dans LK_0 et la transforme en une preuve sans coupure du même séquent.

Déduction automatique

Méthode naïve : tester toutes les assignations

- On donne toutes les valeurs possibles aux variables propositionnelles ;
- Les valeurs d'une variable propositionnelle sont $\top$ et $\bot$;
- La proposition est valide si elle donne $\top$ dans tous les cas ;
- La proposition est insatisfiable si elle donne $\bot$ dans tous les cas ;
- La proposition est non valide si elle donne $\bot$ dans certains cas ;
- La proposition est satisfiable si elle donne $\top$ dans certains cas.

Remarques

- Méthode naïve car exponentielle (donc inefficace) ;
- Pour n variables, on a 2^n cas à tester.

Exemple

Tester la validité d'une formule

- $A \wedge B \Rightarrow A$;
- On fait une table de vérité :

A	B	$A \wedge B$	$A \wedge B \Rightarrow A$
$\top$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\top$
$\bot$	$\top$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$	$\top$

- On peut faire un arbre aussi, puis annoter les noeuds/feuilles par les valeurs de vérité, c'est plus visuel.

Mise en forme clausale

Règles de transformation

$$\neg\neg F \to F \quad \neg\top \to \bot \quad \neg\bot \to \top$$
$$\neg(F_1 \wedge F_2) \to \neg F_1 \vee \neg F_2 \quad \neg(F_1 \vee F_2) \to \neg F_1 \wedge \neg F_2$$
$$F_1 \Rightarrow F_2 \to \neg F_1 \vee F_2$$
$$F_1 \wedge \top \to F_1 \quad \top \wedge F_1 \to F_1 \quad F_1 \wedge \bot \to \bot \quad \bot \wedge F_1 \to \bot$$
$$F_1 \vee \top \to \top \quad \top \vee F_1 \to \top \quad F_1 \vee \bot \to F_1 \quad \bot \vee F_1 \to F_1$$
$$(F_1 \wedge F_2) \vee F_3 \to (F_1 \vee F_3) \wedge (F_2 \vee F_3)$$
$$F_3 \vee (F_1 \wedge F_2) \to (F_3 \vee F_1) \wedge (F_3 \vee F_2)$$

Mise en forme clausale

Exemple

- Proposition : $A \wedge B \Rightarrow A$.
- Étapes de clausification :
  $A \wedge B \Rightarrow A \to \neg(A \wedge B) \vee A \to \neg A \vee \neg B \vee A$
- L'ensemble de clauses est : $\{\neg A \vee \neg B \vee A\}$.

Exercice

Nier et mettre en forme clausale les propositions suivantes

1. $A \Rightarrow B \Rightarrow A$
2. $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$
3. $A \wedge B \Rightarrow B$
4. $B \Rightarrow A \vee B$
5. $(A \vee B) \Rightarrow (A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow C$
6. $A \Rightarrow \perp \Rightarrow \neg A$
7. $\perp \Rightarrow A$

Résolution

Principe de la méthode

- Méthode clausale par réfutation :
  - On nie la proposition initiale ;
  - On la met ensuite en forme clausale.
- Règle de résolution entre deux clauses :

$$\frac{C \vee A \qquad \neg A \vee C'}{C \vee C'}$$

- Les clauses au-dessus de la barre sont les prémisses ;
- La clause en dessous est le résolvant entre les clauses prémisses.

Procédure de résolution

Algorithme

$Sat := \emptyset$;
tant que $S \neq \emptyset$ faire
    choisir $C \in S$;
    $S := S \backslash \{C\}$;
    si $C = \square$ alors retourner « insatisfiable » ;
    si C est une tautologie alors ; (* passer à la clause suivante *)
    sinon, si $C \in Sat$ alors ; (* idem *)
    sinon pour tout résolvant C_1 entre C
    et une clause de $Sat \cup \{C\}$ faire
        $S := S \cup \{C_1\}$;
    $Sat := Sat \cup \{C\}$;
retourner « satisfiable ».

Exécution

Exemple

- Démontrer la validité de la formule : $A \wedge B \Rightarrow A$;
- $S = \{A, B, \neg A\}$;
- On applique la résolution :
  - $Sat = \emptyset$, $S = \{A, B, \neg A\}$;
  - On choisit la clause A : $Sat = \{A\}$, $S = \{B, \neg A\}$;
  - On choisit la clause B : $Sat = \{A, B\}$, $S = \{\neg A\}$;
  - On choisit la clause $\neg A$:
    - Résolution entre $\neg A$ et A : résolvant $\Box$;
    - $Sat = \{A, B, \neg A\}$, $S = \{\Box\}$;
  - On choisit la clause $\Box$, on retourne « insatisfiable ».

Exercice

Appliquer la résolution sur les propositions suivantes

1. $A \Rightarrow B \Rightarrow A$
2. $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$
3. $A \wedge B \Rightarrow B$
4. $B \Rightarrow A \vee B$
5. $(A \vee B) \Rightarrow (A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow C$
6. $A \Rightarrow \perp \Rightarrow \neg A$
7. $\perp \Rightarrow A$

UE HAI504I : Logique du premier ordre

Page Moodle du cours

- https://moodle.umontpellier.fr/course/view.php?id=23410 ;
- Clé d'inscription : hai504i;2021

Supports

- Cours et TD disponibles sous Moodle uniquement ;
- Aide-mémoire (sous Moodle) autorisé au CC et à l'examen.