Déduction automatique et certification de preuve pour la Méthode B

Pierre Halmagrand

10 décembre 2016

Cnam-Cedric / Inria-Deducteam / ENS Paris-Saclay-LSV

Paris et son métro

(Source : Wikipedia.org)

- Inauguré en 1900
- 16 lignes
- 220 km de lignes de métro
- 5,23 millions voyageurs/jour

Paris, ville pionnière des métros automatiques

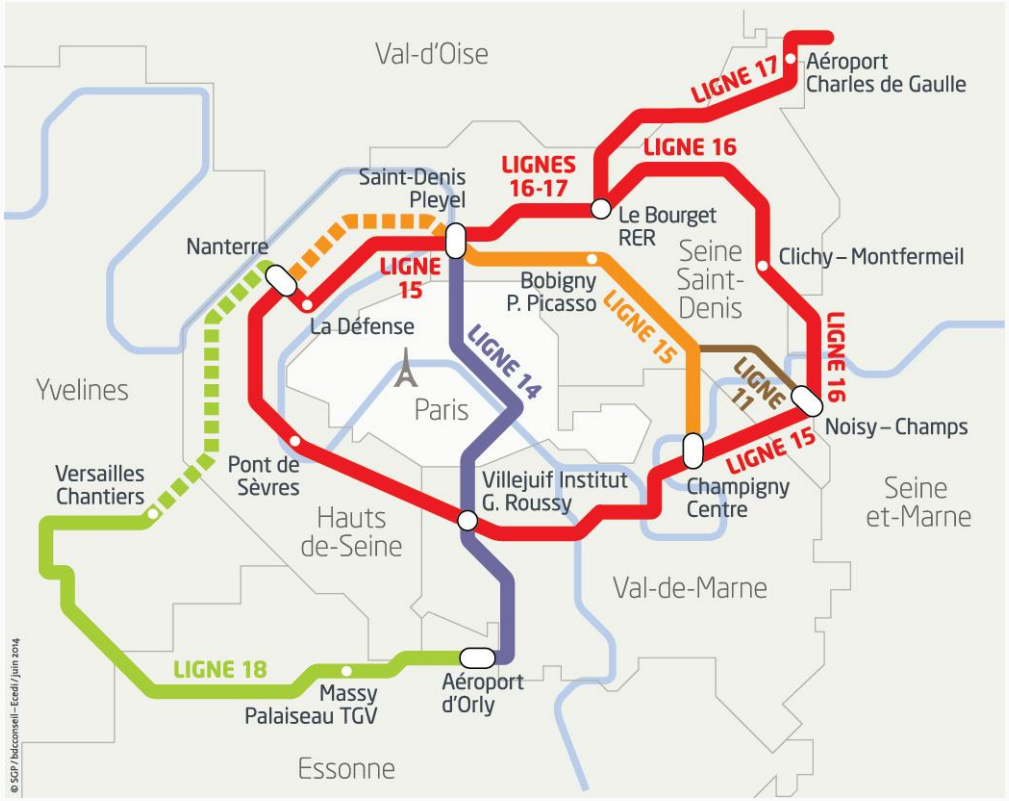
La ligne 14

- Ouverte en 1998
- Conduite intégralement automatisée
- Première ligne automatique en réseau ouvert

Depuis

- Ligne 1 automatisée en 2011
- Ligne 4 programmée pour 2020

Grand Paris Express



(Source : Yvelines.fr)

- Horizon 2030
- Intégralement automatique
- 4 nouvelles lignes
- 200 km de lignes de métro

Les métros automatiques

Avantages

- Absence d'erreurs humaines / de manipulation
- Augmente la fréquence des métros

Inconvénients

- Confiance dans la fiabilité du logiciel embarqué
- Risque de défaillances / de bugs ?

Les systèmes critiques

Un métro automatique est un système critique

Système Critique

Un système critique est un système pour lequel une défaillance peut avoir des conséquences graves, par exemple des morts, des blessés, une catastrophe environnementale ou des pertes financières importantes.

Les logiciels dans les systèmes critiques

Les logiciels : au cœur des systèmes critiques

- Transport : aide au freinage d'urgence des voitures
- Énergie : contrôle des centrales nucléaires
- Santé : logiciel embarqué des appareils médicaux
- Militaire : système de guidage des missiles

Les méthodes formelles

Développer des logiciels sûrs est un enjeu majeur

- Prévenir l'apparition de défaillances / de bugs
- Bon comportement par rapport à ce qui est attendu

Méthodes formelles

- Raisonner sur des programmes informatiques
- Prouver la correction d'un programme par rapport à sa spécification
- Haut niveau de confiance dans le logiciel

La Méthode B

- De la spécification au développement du logiciel
- Créée par J-R. Abrial dans les années 1980/1990
- Logiciels critiques dans l'industrie ferroviaire

Réussite industrielle : les métros automatiques

- 1998 : ligne 14 à Paris
- 2005 : ligne 1 à Paris
- 2006 : ligne L à New-York City
- 2009 : ligne 9 à Barcelone

La Méthode B

Fonctionnement

- Raffinements successifs d'une spécification abstraite à du code
- Prouver correcte chaque étape de raffinement
- Génère des milliers d'obligations de preuve (OP)

Logique

- Logique classique du premier ordre
- Théorie des ensembles
- Notion de typage des expressions

BWare

Le projet BWare

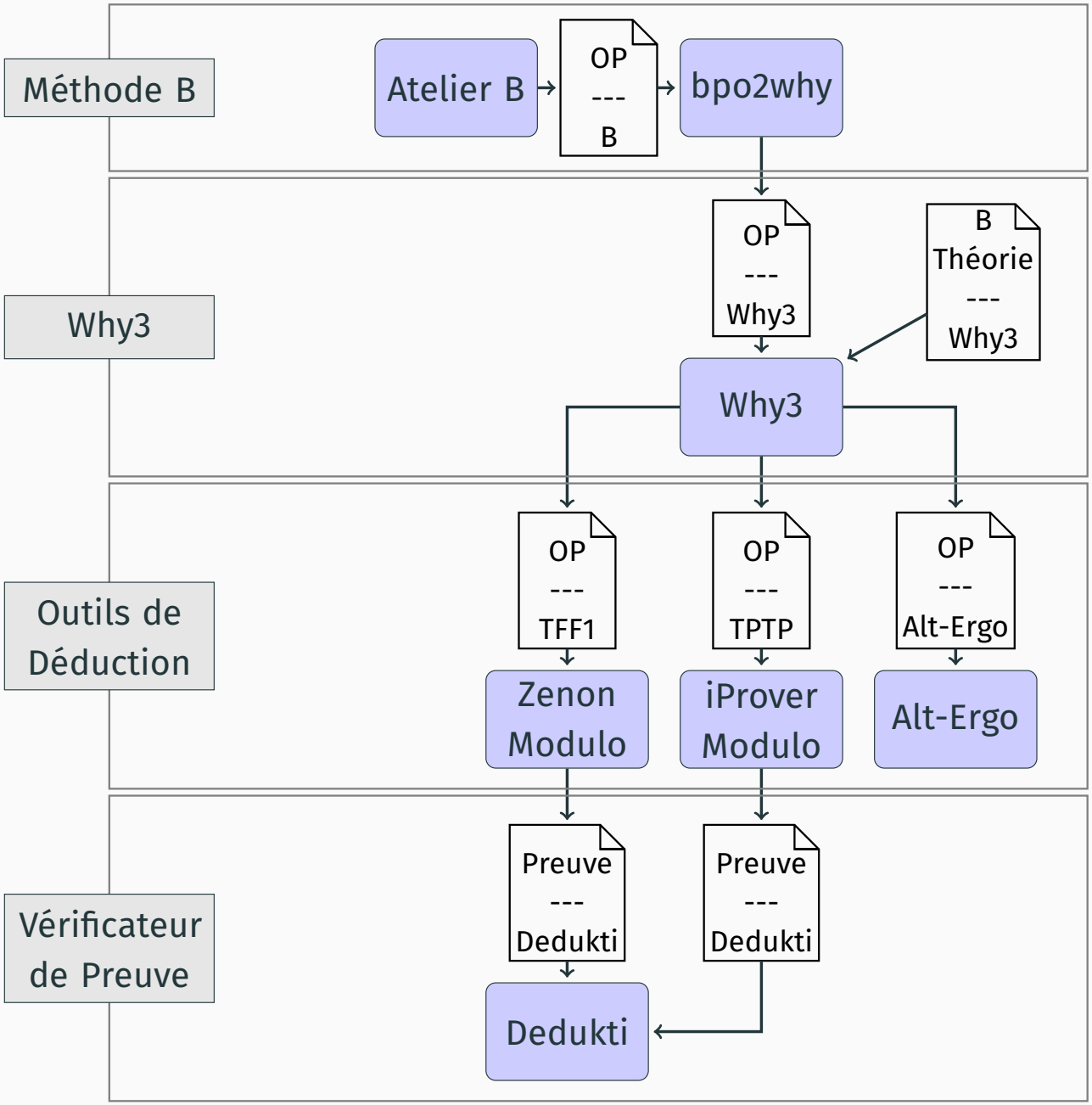
Objectifs

- Améliorer la preuve automatique des obligations de preuve (OP)
- Haut niveau de confiance dans les preuves produites

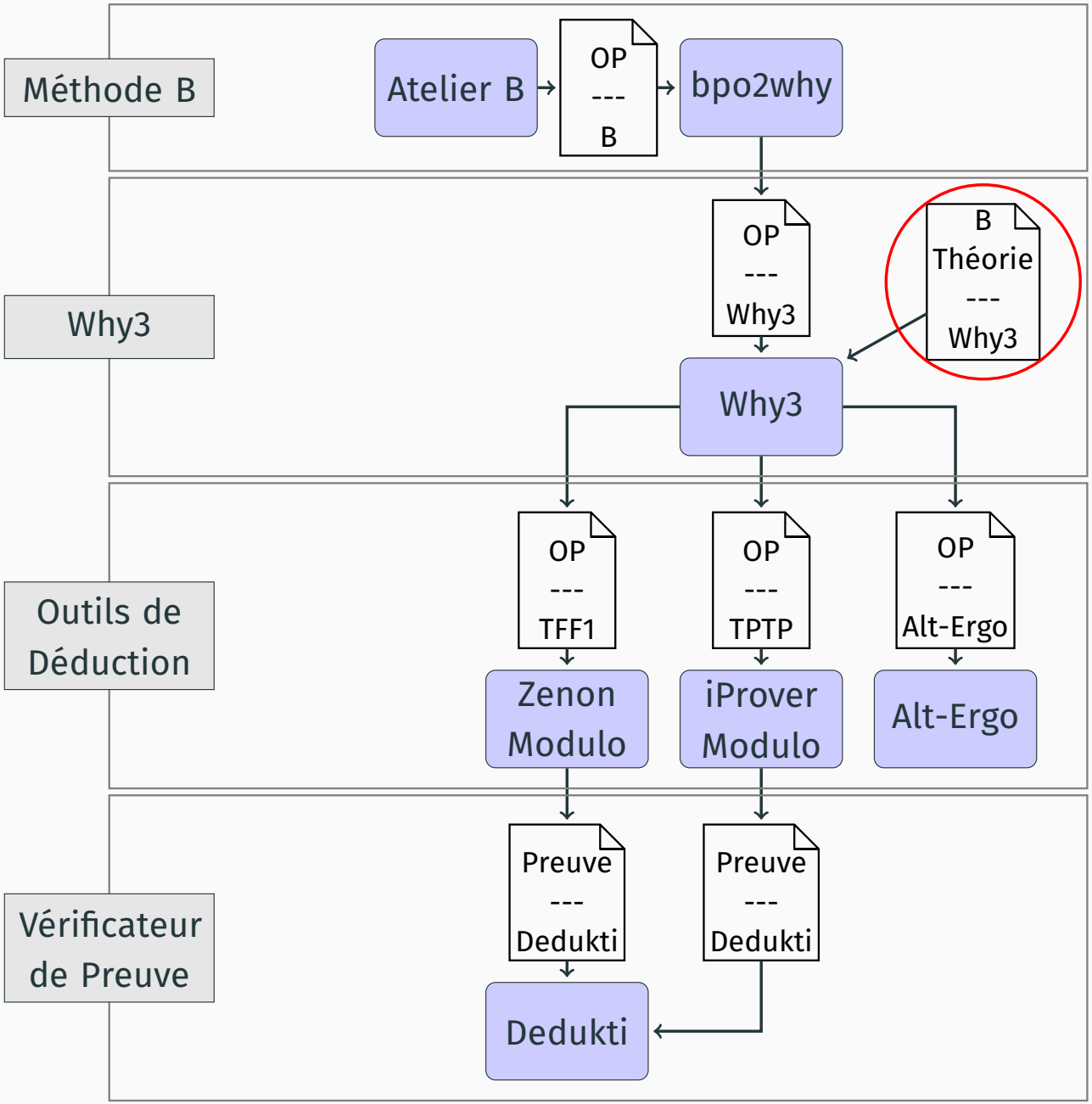
Méthodologie

- Utiliser et améliorer des outils de déduction automatique existants
- Générer des certificats de preuve

Schéma du projet BWare

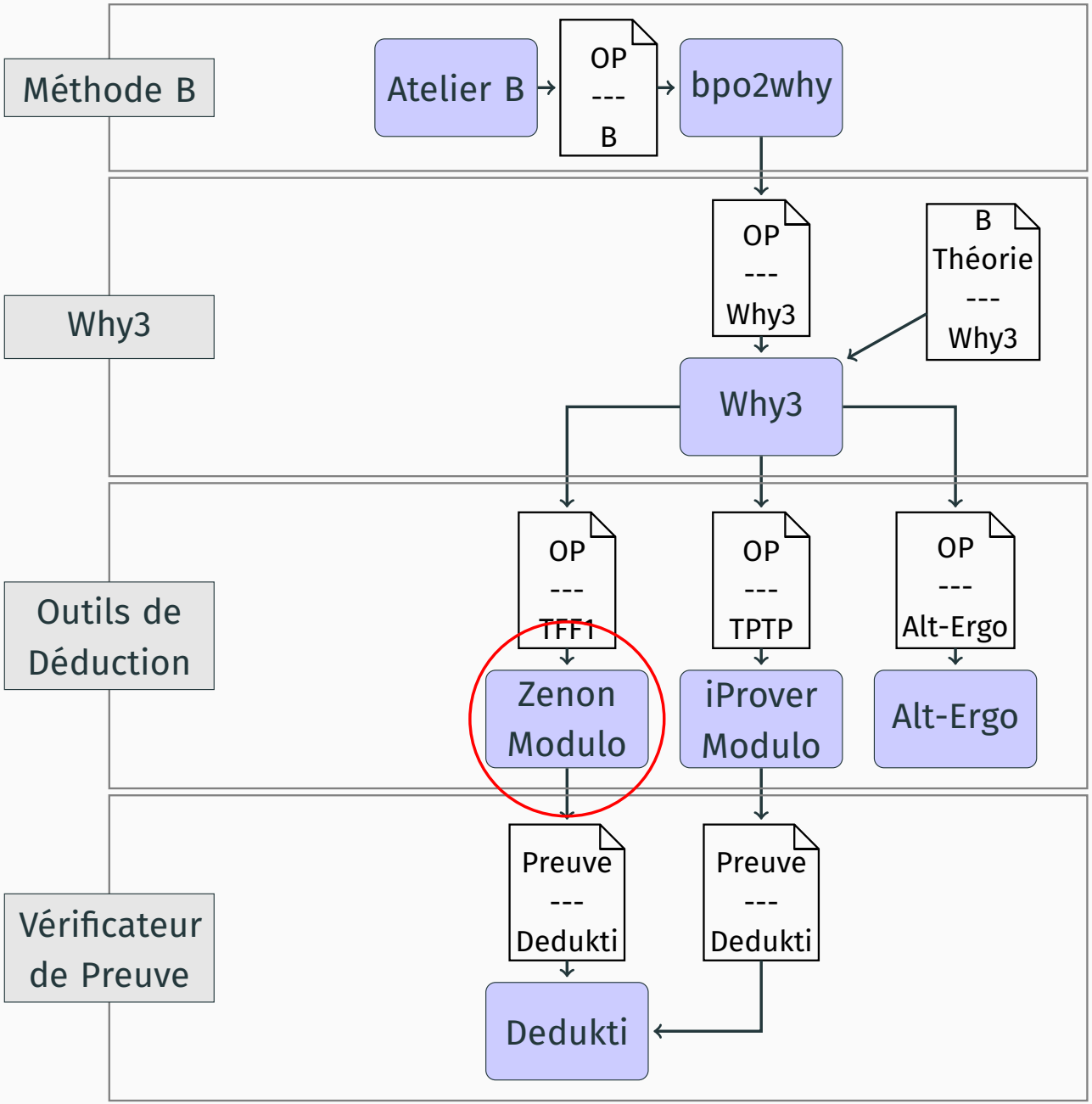


Plan



1. Théorie B en Why3

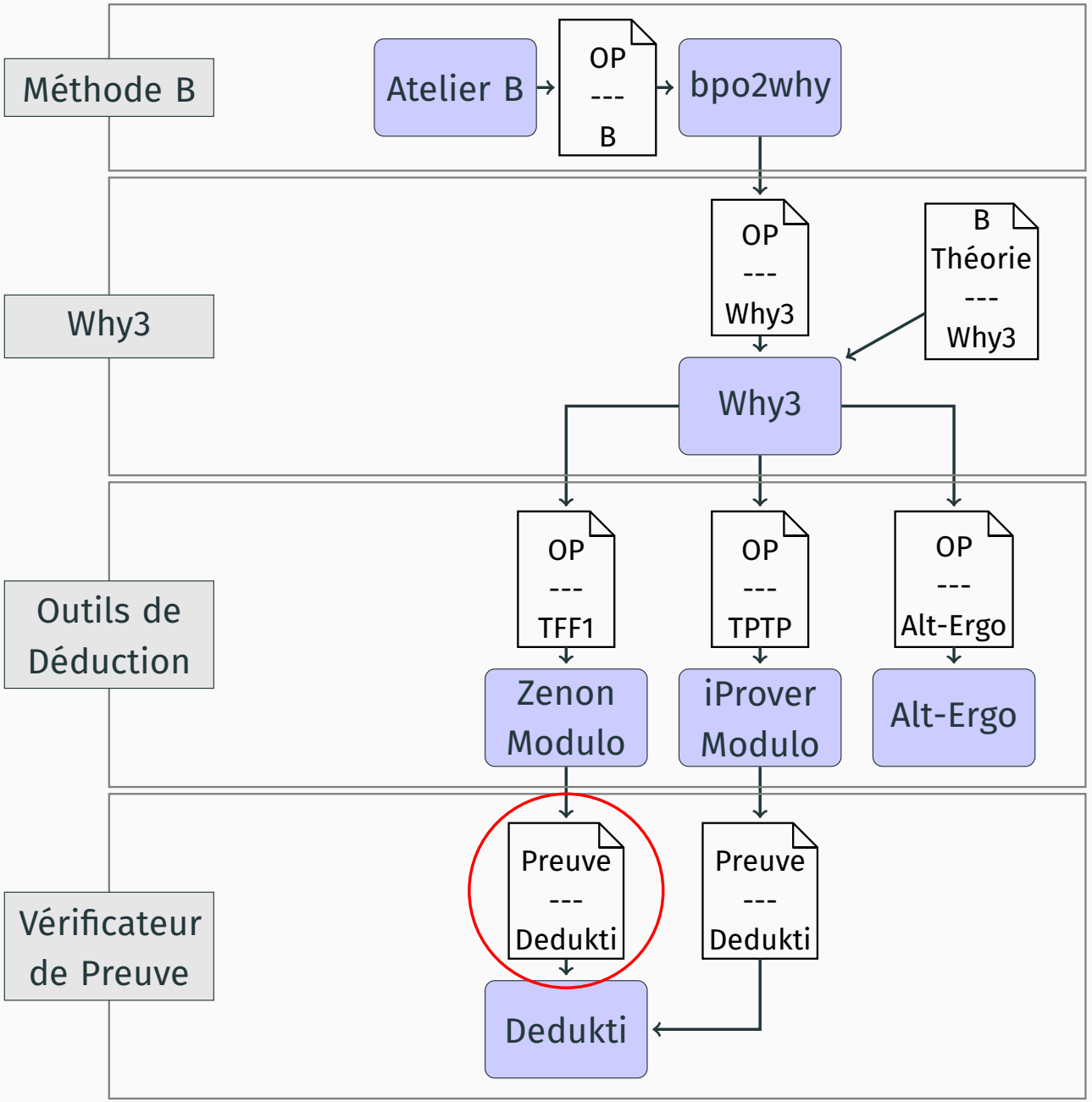
Plan



1. Théorie B en Why3

2. Extension de Zenon au typage et à la réécriture

Plan

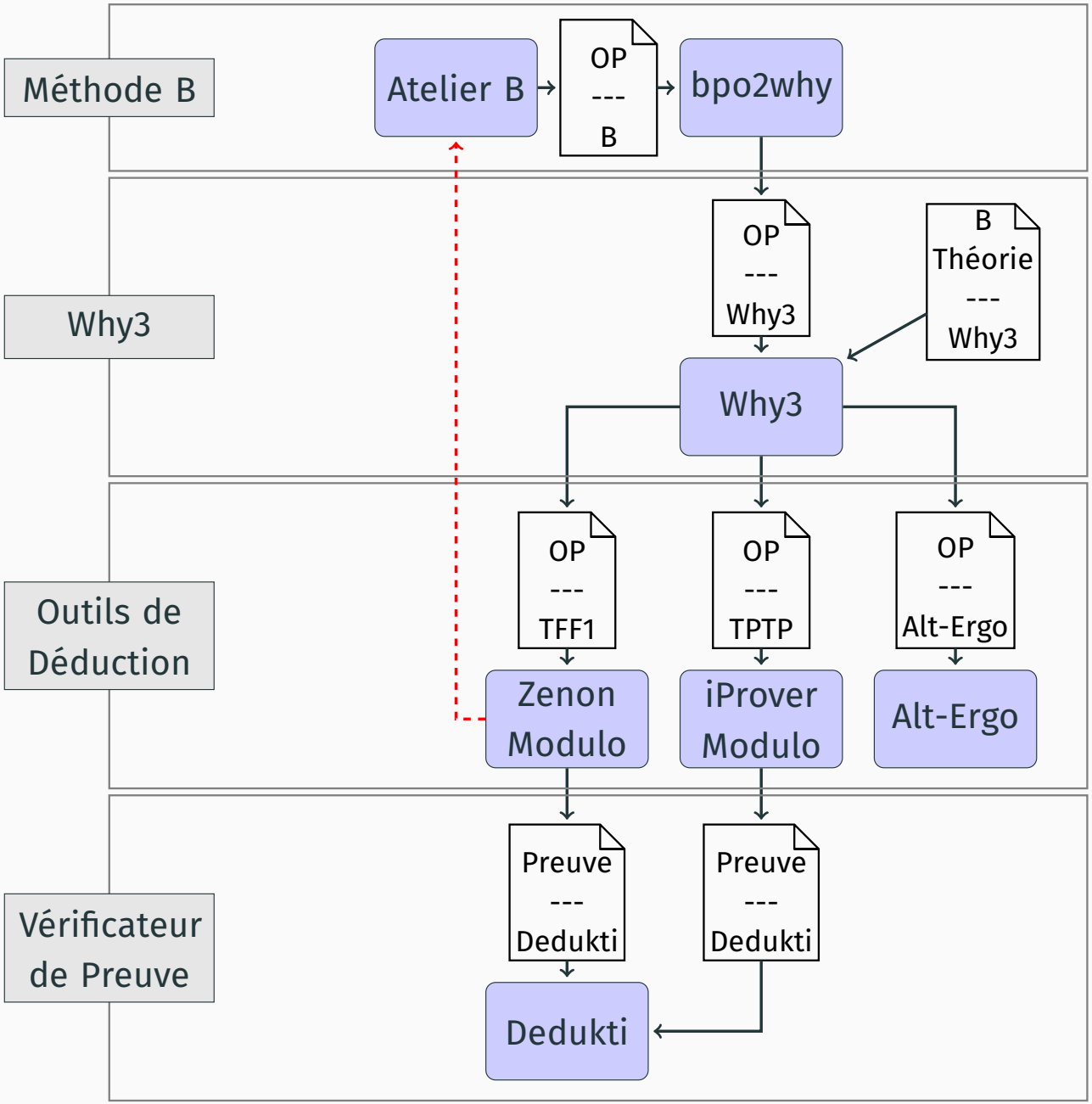


1. Théorie B en Why3

2. Extension de Zenon au typage et à la réécriture

3. Certificats de preuve en Dedukti

Plan



1. Théorie B en Why3

2. Extension de Zenon au typage et à la réécriture

3. Certificats de preuve en Dedukti

4. Traduction des preuves vers B (non implémenté)

La Théorie B en Why3

Why3

Plateforme

- Pour la vérification de programme
- Logique du premier ordre
- Système de types polymorphes à la ML

Pour BWare

- Traduit OP + théorie vers chaque prouveur automatique

Polymorphisme

Généralités

- Définir des axiomes génériques
- Inférence de type décidable
- Noté Poly-FOL dans la suite

Preuve automatique

- Alt-Ergo : SMT solver (Bobot et al. - 2008)
- Zipperposition : prouveur basé sur la superposition (Cruanes - 2014)
- Prototype basé sur SPASS (Wand - 2014)

Exemple de théorème B

Schéma d'axiome de l'ensemble des parties

$$s \in \mathbb{P}(t) \Leftrightarrow \forall x \cdot (x \in s \Rightarrow x \in t)$$

Obligation de preuve : étant donné un ensemble u, on veut prouver

$$u \in \mathbb{P}(u)$$

Exemple de théorème B en Poly-FOL

Signatures

$$\begin{aligned} \mathbf{set} &:: 1 \\ \in &: \Pi\alpha.\ \alpha \times \mathbf{set}(\alpha) \to o \\ \mathbb{P} &: \Pi\alpha.\ \mathbf{set}(\alpha) \to \mathbf{set}(\mathbf{set}(\alpha)) \end{aligned}$$

Axiome

$$\forall\alpha.\ \forall s, t : \mathbf{set}(\alpha).\ \ \in(\mathbf{set}(\alpha); s, \mathbb{P}(\alpha; t)) \Leftrightarrow \forall x : \alpha.\ \ \in(\alpha; x, s) \Rightarrow \in(\alpha; x, t)$$

Exemple de théorème B en Poly-FOL

Signatures

$$\begin{aligned} \mathsf{set} &:: 1 \\ \in &: \Pi\alpha.\ \alpha \times \mathsf{set}(\alpha) \to o \\ \mathbb{P} &: \Pi\alpha.\ \mathsf{set}(\alpha) \to \mathsf{set}(\mathsf{set}(\alpha)) \end{aligned}$$

Axiome

$$\forall\alpha.\ \forall s, t : \mathsf{set}(\alpha).\ \in(\mathsf{set}(\alpha); s, \mathbb{P}(\alpha; t)) \Leftrightarrow \forall x : \alpha.\ \in(\alpha; x, s) \Rightarrow \in(\alpha; x, t)$$

$$\forall\alpha.\ \forall s, t : \mathsf{set}(\alpha).\ s \in_{\mathsf{set}(\alpha)} \mathbb{P}_\alpha(t) \Leftrightarrow \forall x : \alpha.\ x \in_\alpha s \Rightarrow x \in_\alpha t$$

Exemple de théorème B en Poly-FOL

Signatures

$$\begin{aligned} \mathsf{set} &:: 1 \\ \in &: \Pi\alpha.\ \alpha \times \mathsf{set}(\alpha) \to o \\ \mathbb{P} &: \Pi\alpha.\ \mathsf{set}(\alpha) \to \mathsf{set}(\mathsf{set}(\alpha)) \end{aligned}$$

Axiome

$$\forall\alpha.\ \forall s,t : \mathsf{set}(\alpha).\ \in(\mathsf{set}(\alpha); s, \mathbb{P}(\alpha; t)) \Leftrightarrow \forall x : \alpha.\ \in(\alpha; x, s) \Rightarrow \in(\alpha; x, t)$$

$$\forall\alpha.\ \forall s,t : \mathsf{set}(\alpha).\ s \in_{\mathsf{set}(\alpha)} \mathbb{P}_\alpha(t) \Leftrightarrow \forall x : \alpha.\ x \in_\alpha s \Rightarrow x \in_\alpha t$$

Obligation de preuve

$$\begin{aligned} \tau &:: 0 \\ u &: \mathsf{set}(\tau) \end{aligned}$$

$$u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u)$$

Les constructeurs dérivés

Une cinquantaine de constructeurs dérivés définis en B

- Constructions ensemblistes usuelles : union, ensemble vide, ...
- Très présents dans les OP
- Utilisent les ensembles en compréhension

Étant donné u, s et t tels que $s \subseteq u$ et $t \subseteq u$

$$s \cup t := \{a \mid a \in u \land (a \in s \lor a \in t)\}$$

Les constructeurs dérivés

Une cinquantaine de constructeurs dérivés définis en B

- Constructions ensemblistes usuelles : union, ensemble vide, ...
- Très présents dans les OP
- Utilisent les ensembles en compréhension

Étant donné u, s et t tels que $s \subseteq u$ et $t \subseteq u$

$$s \cup t := \{a \mid a \in u \wedge (a \in s \vee a \in t)\}$$

Traduit en Poly-FOL

$$\cup \ : \ \Pi\alpha.\ \mathbf{set}(\alpha) \times \mathbf{set}(\alpha) \to \mathbf{set}(\alpha)$$

$$\forall\alpha.\ \forall s, t : \mathbf{set}(\alpha).\ \forall x : \alpha. \quad x \in_\alpha s \cup_\alpha t \Leftrightarrow x \in_\alpha s \vee x \in_\alpha t$$

Preuve automatique : Zenon Modulo

Zenon

Zenon

- Premier ordre avec égalité
- Méthode des Tableaux
- Preuves dans le calcul des séquents
- Certificats de preuve vérifiables dans Coq

Problème : résultats expérimentaux insuffisants sur les OP de B

- Pas typé : utilise un encodage
- Espace de recherche trop grand (théorie B)

Zenon Modulo

Solution proposée

- Typage polymorphe
- Déduction modulo théorie

Résultat : Zenon Modulo

- Problèmes avec typage polymorphe
- Règles de réécriture en entrée (en plus des axiomes)
- Certificats de preuve vérifiables dans Dedukti

Implémentation du polymorphisme

Objectif

- Minimiser les modifications du code

Impact sur le code

- Type-checking après le parsing
- Expressions bien typées pendant la recherche
- Instanciation des variables de type
- Faible impact sur les problèmes non-typés

Déduction modulo théorie

Principe de Poincaré

Séparer le *calcul* et le *raisonnement déductif*

Déduction modulo théorie

Généralités

- Proposé par Gilles Dowek, Thérèse Hardin et Claude Kirchner
- Simplifie les preuves dans les théories axiomatiques
- Raisonner modulo relation de congruence $\equiv$ sur les propositions
- Réécriture sur les termes et les propositions
- Systèmes de réécriture confluents et terminants

Règle de conversion (calcul des séquents de Zenon Modulo)

$$\frac{\Gamma, Q \vdash \bot}{\Gamma, P \vdash \bot}\ \text{conv}\ P \equiv Q$$

Système de réécriture

Transformer les axiomes en règle de réécriture :

$$\forall \vec{x}.\ P \Leftrightarrow S \vee Q \qquad \text{devient} \quad P \longrightarrow S \vee Q$$
$$\forall \vec{x}.\ s = t \qquad \text{devient} \quad s \longrightarrow t$$

Système de réécriture

Transformer les axiomes en règle de réécriture :

$$\begin{array}{lll} \forall \vec{x}.\, P \Leftrightarrow S \vee Q & \text{devient} & P \longrightarrow S \vee Q \\ \forall \vec{x}.\, s = t & \text{devient} & s \longrightarrow t \end{array}$$

Par exemple

$$\begin{array}{rcl} s \in_{\mathsf{set}(\alpha)} \mathbb{P}_\alpha(t) & \longrightarrow & \forall x : \alpha.\, x \in_\alpha s \Rightarrow x \in_\alpha t \\ x \in_\alpha s \cup_\alpha t & \longrightarrow & x \in_\alpha s \vee x \in_\alpha t \end{array}$$

Système de réécriture

Transformer les axiomes en règle de réécriture :

$$\begin{array}{lll} \forall \vec{x}.\ P \Leftrightarrow S \vee Q & \text{devient} & P \longrightarrow S \vee Q \\ \forall \vec{x}.\ s = t & \text{devient} & s \longrightarrow t \end{array}$$

Par exemple

$$\begin{array}{rcl} s \in_{\mathsf{set}(\alpha)} \mathbb{P}_\alpha(t) & \longrightarrow & \forall x : \alpha.\ x \in_\alpha s \Rightarrow x \in_\alpha t \\ x \in_\alpha s \cup_\alpha t & \longrightarrow & x \in_\alpha s \vee x \in_\alpha t \end{array}$$

Théorie B

- Système de réécriture orthogonal : confluent
- Ordre sur les symboles : terminant

Exemple en calcul des séquents typés

Preuve de $u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u)$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{c \in_\tau u, c \notin_\tau u \vdash \bot}\,\mathrm{Ax}}{\neg(c \in_\tau u \Rightarrow c \in_\tau u) \vdash \bot}\,\neg\Rightarrow}{\neg(\forall x : \tau.\, x \in_\tau u \Rightarrow x \in_\tau u) \vdash \bot}\,\neg\forall \quad \dfrac{}{u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u) \vdash \bot}\,\mathrm{Ax}}{(\forall x : \tau.\, x \in_\tau u \Rightarrow x \in_\tau u) \Rightarrow u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u) \vdash \bot}\,\Rightarrow}{u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u) \Leftrightarrow (\forall x : \tau.\, x \in_\tau u \Rightarrow x \in_\tau u) \vdash \bot}\,\wedge}{\forall t : \mathsf{set}(\tau).\, u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(t) \Leftrightarrow (\forall x : \tau.\, x \in_\tau u \Rightarrow x \in_\tau t) \vdash \bot}\,\forall}{\forall s : \mathsf{set}(\tau), t : \mathsf{set}(\tau).\, s \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(t) \Leftrightarrow (\forall x : \tau.\, x \in_\tau s \Rightarrow x \in_\tau t) \vdash \bot}\,\forall}{\forall \alpha. \forall s : \mathsf{set}(\alpha), t : \mathsf{set}(\alpha).\, s \in_{\mathsf{set}(\alpha)} \mathbb{P}_\alpha(t) \Leftrightarrow (\forall x : \alpha.\, x \in_\alpha s \Rightarrow x \in_\alpha t), \neg(u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u)) \vdash \bot}\,\forall_{\mathsf{Type}}$$

Exemple en deduction modulo théorie

On a

$$u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u) \quad \equiv \quad \forall x : \tau.\, x \in_\tau u \Rightarrow x \in_\tau u$$

La preuve est plus simple

$$\dfrac{\dfrac{\dfrac{\dfrac{}{c \in_\tau u, c \notin_\tau u \vdash \bot}\,\mathrm{Ax}}{\neg(c \in_\tau u \Rightarrow c \in_\tau u) \vdash \bot}\,\neg\Rightarrow}{\neg(\forall x : \tau.\, x \in_\tau u \Rightarrow x \in_\tau u) \vdash \bot}\,\neg\forall}{\neg(u \in_{\mathsf{set}(\tau)} \mathbb{P}_\tau(u)) \vdash \bot}\,\mathrm{conv}$$

Implémentation dans Zenon Modulo

Système de réécriture

- Automatique : heuristique dans Zenon Modulo
- À la main : annotation des axiomes

Recherche de preuve

- Normalisation pendant la recherche de preuve
- Réécriture paresseuse : seulement les atomes
- Pas de mémoire de la réécriture

MLproof : recherche de preuve de Zenon Modulo

Méthode des tableaux

- Preuve par contradiction
- Négation du but insatisfiable
- Présentation de haut en bas

Déduction modulo théorie

- Pas de règle de conversion
- [P] classe d'équivalence de P modulo réécriture

Règles séparées en 5 catégories :

- Ordre de priorité pour l'application des règles

$$\odot \prec \alpha \prec \delta \prec \beta \prec \gamma$$

Syntaxe pour la recherche de preuve

Types

$$
\begin{array}{lcll}
\tau & ::= & \alpha & \textit{(variable de type)} \\
 & | & \mathsf{T}(\tau_1, \ldots, \tau_m) & \textit{(constructeur de type)} \\
 & | & A_{\mathsf{Type}} & \textit{(métavariable de type)}
\end{array}
$$

Schémas de Type

$$
\begin{array}{lcll}
\sigma & ::= & \Pi\alpha_1 \cdots \alpha_m.\ \tau_1 \times \cdots \times \tau_n \to \tau & \textit{(fonctions)} \\
 & | & \Pi\alpha_1 \cdots \alpha_m.\ \tau_1 \times \cdots \times \tau_n \to o & \textit{(prédicats)}
\end{array}
$$

Syntaxe pour la recherche de preuve

Termes

$$
\begin{array}{lll}
t & ::= & x \qquad \textit{(variable)} \\
 & | & f(\tau_1, \ldots, \tau_m; t_1, \ldots, t_n) \qquad \textit{(application de fonction)} \\
 & | & X_\tau \qquad \textit{(métavariable)} \\
 & | & \varepsilon(x : \tau).\ \varphi \qquad \textit{(ε-terme de Hilbert)}
\end{array}
$$

Formules

$$
\begin{array}{lll}
\varphi & ::= & \top \mid \bot \qquad \textit{(vrai et faux)} \\
 & | & t_1 =_\tau t_2 \qquad \textit{(égalité)} \\
 & | & P(\tau_1, \ldots, \tau_m; t_1, \ldots, t_n) \qquad \textit{(application de prédicat)} \\
 & | & \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \\
 & | & \varphi_1 \Rightarrow \varphi_2 \mid \varphi_1 \Leftrightarrow \varphi_2 \qquad \textit{(connecteurs logiques)} \\
 & | & \forall x : \tau.\ \varphi \mid \exists x : \tau.\ \varphi \qquad \textit{(quantification)}
\end{array}
$$

Formules avec quantification de type

$$
\begin{array}{lll}
\varphi_\tau & ::= & \varphi \qquad \textit{(Formule)} \\
 & | & \forall \alpha.\ \varphi_\tau \qquad \textit{(quantification de type)}
\end{array}
$$

Système de preuve MLproof

$$\frac{[P], [\neg P]}{\odot}\ \odot \qquad \frac{[P \wedge Q]}{P,\ Q}\ \alpha_{\wedge} \qquad \frac{[P \vee Q]}{P \mid Q}\ \beta_{\vee}$$

- $\odot$ règles de clôture
- α connecteurs logiques, génèrent une branche
- β connecteurs logiques, génèrent deux branches

Système de preuve MLproof

$$\frac{[\exists x : \tau.\ P(x)]}{P(\varepsilon(x : \tau).\ P(x))}\ \delta_{\exists} \qquad \frac{[\forall x : \tau.\ P(x)]}{P(X_\tau)}\ \gamma_{\forall M} \qquad \frac{[\forall x : \tau.\ P(x)]}{P(t)}\ \gamma_{\forall \mathbf{inst}}\ t : \tau$$

$$\frac{[\forall \alpha.\ P(\alpha)]}{P(A_{\mathsf{Type}})}\ \gamma_{\forall M_{\mathsf{Type}}} \qquad \frac{[\forall \alpha.\ P(\alpha)]}{P(\tau)}\ \gamma_{\forall \mathbf{inst}_{\mathsf{Type}}}\ \tau : \mathsf{Type}$$

- δ quantification existentielle
- γ quantification universelle

Système de preuve MLproof

$$\frac{[P(\tau_1, \ldots, \tau_m; a_1, \ldots, a_n)],\ [\neg P(\tau_1, \ldots, \tau_m; b_1, \ldots, b_n)]}{a_1 \neq b_1 \quad | \quad \ldots \quad | \quad a_n \neq b_n}\ \text{pred}$$

$$\frac{[f(\tau_1, \ldots, \tau_m; a_1, \ldots, a_n) \neq f(\tau_1, \ldots, \tau_m; b_1, \ldots, b_n)]}{a_1 \neq b_1 \quad | \quad \ldots \quad | \quad a_n \neq b_n}\ \text{fun}$$

- S'appliquent lorsque les types sont égaux
- Aident à trouver des instances pour les variables de type

Exemple de recherche de preuve

Signature :

$$\begin{cases} P : \Pi\alpha.\ \alpha \times \alpha \to o \\ \tau :: 0 \\ a, b : \tau \end{cases}$$

Hypothèse :

$$\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y)$$

On veut prouver :

$$P(\tau; a, b)$$

Exemple de recherche de preuve

$$\underline{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}$$

Exemple de recherche de preuve

$$\frac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\ \gamma_{\forall M_{\mathsf{Type}}}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\ \gamma_{\forall M_{\mathsf{Type}}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\ \gamma_{\forall M}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\ \gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\ \gamma\forall M}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\ \gamma\forall M$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\ \gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\ \gamma\forall M}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\ \gamma\forall M}{\forall x, y : \tau.\ P(\tau; x, y)}\ \gamma\forall \mathbf{inst}_{\mathsf{Type}}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\ \gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\ \gamma\forall M}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\ \gamma\forall \mathbf{inst}_{\mathsf{Type}}}{\forall x, y : \tau.\ P(\tau; x, y)}\ \gamma\forall M}{\forall y : \tau.\ P(\tau; X_\tau, y)}\ \gamma\forall M$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\ \gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\ \gamma\forall M}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\ \gamma\forall \mathbf{inst}_{\mathsf{Type}}}{\forall x, y : \tau.\ P(\tau; x, y)}\ \gamma\forall M}{\forall y : \tau.\ P(\tau; X_\tau, y)}\ \gamma\forall M}{P(\tau; X_\tau, Y_\tau)}\ \gamma\forall M$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\,\gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\,\gamma\forall M}}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\,\gamma\forall M}}{\forall x, y : \tau.\ P(\tau; x, y)}\,\gamma\forall \mathrm{inst}_{\mathsf{Type}}}{\forall y : \tau.\ P(\tau; X_\tau, y)}\,\gamma\forall M}}{P(\tau; X_\tau, Y_\tau)}\,\gamma\forall M}}{X_\tau \neq_\tau a \qquad\qquad Y_\tau \neq_\tau b}\,\mathrm{pred}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\,\gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\,\gamma\forall M}}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\,\gamma\forall \mathbf{inst}_{\mathsf{Type}}}{\forall x, y : \tau.\ P(\tau; x, y)}\,\gamma\forall M}}{\forall y : \tau.\ P(\tau; X_\tau, y)}\,\gamma\forall M}}{P(\tau; X_\tau, Y_\tau)}}{\dfrac{X_\tau \neq_\tau a}{\forall y : \tau.\ P(\tau; a, y)}\,\gamma\forall \mathbf{inst}} \qquad Y_\tau \neq_\tau b}\,\mathbf{pred}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\,\gamma\forall M_{\mathsf{Type}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\,\gamma\forall M}}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\,\gamma\forall \mathbf{inst}_{\mathsf{Type}}}{\forall x, y : \tau.\ P(\tau; x, y)}\,\gamma\forall M}}{\forall y : \tau.\ P(\tau; X_\tau, y)}\,\gamma\forall M}}{P(\tau; X_\tau, Y_\tau)}}{X_\tau \neq_\tau a}\,\gamma\forall \mathbf{inst}}{\forall y : \tau.\ P(\tau; a, y)}\,\gamma\forall M}}{P(\tau; a, Y'_\tau)}\ \mathbf{pred}\qquad Y_\tau \neq_\tau b$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\,\gamma_{\forall M_{\mathsf{Type}}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\,\gamma_{\forall M}}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\,\gamma_{\forall M}}{\forall x, y : \tau.\ P(\tau; x, y)}\,\gamma_{\forall \mathbf{inst}_{\mathsf{Type}}}}{\forall y : \tau.\ P(\tau; X_\tau, y)}\,\gamma_{\forall M}}{P(\tau; X_\tau, Y_\tau)}\,\gamma_{\forall M}}{X_\tau \neq_\tau a \qquad Y_\tau \neq_\tau b}\,\mathbf{pred}}{\forall y : \tau.\ P(\tau; a, y)}\,\gamma_{\forall \mathbf{inst}}}{P(\tau; a, Y'_\tau)}\,\gamma_{\forall M}}{a \neq_\tau a \qquad Y'_\tau \neq_\tau b}\,\mathbf{pred}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\,\gamma_{\forall M_{\mathsf{Type}}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\,\gamma_{\forall M}}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\,\gamma_{\forall M}}{\forall x, y : \tau.\ P(\tau; x, y)}\,\gamma_{\forall \mathbf{inst}_{\mathsf{Type}}}}{\forall y : \tau.\ P(\tau; X_\tau, y)}\,\gamma_{\forall M}}{P(\tau; X_\tau, Y_\tau)}\,\gamma_{\forall M}}{X_\tau \neq_\tau a \qquad\qquad Y_\tau \neq_\tau b}\,\mathbf{pred}}{\forall y : \tau.\ P(\tau; a, y)}\,\gamma_{\forall \mathbf{inst}}}{P(\tau; a, Y'_\tau)}\,\gamma_{\forall M}}{a \neq_\tau a \qquad Y'_\tau \neq_\tau b}\,\mathbf{pred}}{\dfrac{a \neq_\tau a}{\odot}\,\odot_r \qquad P(\tau; a, b)}\,\gamma_{\forall \mathbf{inst}}$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall\alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; x, y)}\,\gamma_{\forall M_{\mathsf{Type}}}}{\forall y : A_{\mathsf{Type}}.\ P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, y)}\,\gamma_{\forall M}}{P(A_{\mathsf{Type}}; X_{A_{\mathsf{Type}}}, Y_{A_{\mathsf{Type}}})}\,\gamma_{\forall M}}{\forall x, y : \tau.\ P(\tau; x, y)}\,\gamma_{\forall \mathbf{inst}_{\mathsf{Type}}}}{\forall y : \tau.\ P(\tau; X_\tau, y)}\,\gamma_{\forall M}}{P(\tau; X_\tau, Y_\tau)}\,\gamma_{\forall M}}{X_\tau \neq_\tau a \qquad Y_\tau \neq_\tau b}\,\mathbf{pred}}{\forall y : \tau.\ P(\tau; a, y)}\,\gamma_{\forall \mathbf{inst}}}{P(\tau; a, Y'_\tau)}\,\gamma_{\forall M}}{a \neq_\tau a \qquad Y'_\tau \neq_\tau b}\,\mathbf{pred}}{\odot}\,\odot_r \qquad \dfrac{\dfrac{Y'_\tau \neq_\tau b}{P(\tau; a, b)}\,\gamma_{\forall \mathbf{inst}}}{\odot}\,\odot$$

Exemple de recherche de preuve

$$\dfrac{\dfrac{\dfrac{\dfrac{\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y), \neg P(\tau; a, b)}{\forall x, y : \tau.\ P(\tau; x, y)}\ \gamma\forall\mathbf{inst}_{\mathsf{Type}}}{\forall y : \tau.\ P(\tau; a, y)}\ \gamma\forall\mathbf{inst}}{P(\tau; a, b)}\ \gamma\forall\mathbf{inst}}{\odot}\ \odot$$

Exemple en déduction modulo théorie

On transforme l'axiome :

$$\forall \alpha.\ \forall x, y : \alpha.\ P(\alpha; x, y)$$

En règle de réécriture :

$$P(\alpha; x, y) \longrightarrow \top$$

On obtient alors l'arbre de preuve :

$$\dfrac{\neg P(\tau; a, b)}{\odot}\ \odot_{\neg \top}$$

Résultats expérimentaux

Benchmark d'obligations de preuve

Ensemble d'obligations de preuve

- Fourni par les partenaires industriels
- Constitué de 12.876 OP
- Toutes prouvées dans l'Atelier B (automatique ou interactif)
- Couvre un large spectre de problèmes B
- Taille importante des fichiers

Résultats Zenon

	Tous les outils (98,9%)					
12.876	mp	Zenon	Zenon Typé	Zenon Arith	Zenon Modulo	Zenon Mod+Ari
%	85%	2%	48%	57%	80%	95%
Temps (s)	-	6,9	2,3	2,5	3,0	2,6
Unique	329	0	0	0	34	946

Protocole

- Processeur Intel Xeon E5-2660 v2
- Temps 120 s
- Mémoire 1 GiB

Zenon Arith : Extension à l'arithmétique linéaire par Guillaume Bury

Résultats globaux

	Tous les outils (99,4%)							
	BWare (99,2%)				Autres			
12.876	mp	Zen M+A	iProv Mod	Alt Ergo	Vamp	E	CVC4	Z3
%	85%	95%	28%	98%	78%	61%	94%	84%
Temps	-	2,6	5,5	0,56	12	4,7	0,69	0,31
Uniq.1	109	4	0	65				
Uniq.2	84	0	0	13	0	0	1	12

Certification de preuve

Problématique

Prouveurs automatiques : des logiciels complexes

- Dizaines de milliers de lignes de code
- Heuristiques sophistiquées
- Risque important de bug

Trois types de résultats

- Binaire oui/non : aucune garantie
- Trace de preuve : oblige à lire la preuve + reconstruction partielle
- Certificat de preuve : vérification automatique

Certificats de preuve

Vérificabilité sceptique a posteriori

- Prouveurs : générateurs de certificats de preuve
- Réponse finale : vérificateur de preuve
- Critère de De Bruijn : vérificateur simple et auditable

Dedukti

Zenon Modulo utilise Dedukti

- $\lambda\Pi$-calcul modulo théorie (types dépendants et réécriture)
- Backend de vérification de preuve universel
- Preuves utilisant de la réécriture
- Code source léger ($\approx$ 2000 ligne de code)

Vérification des preuves de Zenon Modulo

Certificats reflètent fidèlement les preuves

- Logique classique : ajout de l'axiome du tiers exclu
- Pas de transformation de la preuve

Principe

- Encodage de la logique du premier ordre avec types polymorphes
- Encodage du calcul des séquents de Zenon Modulo

100% des preuves de Zenon Modulo vérifiées par Dedukti

Traduction des preuves de Zenon Modulo vers B

Problématique

bpo2why : traduction de B vers Why3

- Logiciel propriétaire : code source non accessible
- Algorithmes sophistiqués : inférence de type, ...
- Difficile à certifier

Why3 versus B

- Logiques différentes
- En particulier les systèmes de types

Une solution (pas encore implémentée)

Reconstruire des preuves B à partir de celles de Zenon Modulo

- Encodage des formules B dans Poly-FOL

$$\forall x \cdot (x \in \mathbb{P}(x)) \quad \hookrightarrow \quad \forall x : \alpha.\ (x \in_\alpha \mathbb{P}_\alpha(x))$$

- Traduction des preuves de Zenon Modulo vers B

$$\frac{\Pi_{ZM}}{\Gamma, \neg \forall x : \alpha.\ (x \in_\alpha \mathbb{P}_\alpha(x)) \vdash_{ZM} \bot} \quad \hookrightarrow \quad \frac{\Pi_B}{\Gamma \vdash_B \forall x \cdot (x \in \mathbb{P}(x))}$$

Traduction des preuves de Zenon Modulo

Théorème

Pour Γ un ensemble de formules B et P une OP, si il existe une preuve Zenon Modulo du sequent $\langle \Gamma^{\star s} \rangle, \neg \langle P^{\star s} \rangle \vdash_{ZM} \bot$, alors il existe une preuve B du séquent $\Gamma \vdash_B P$

Démonstration

1. Soit Π_{ZM} une preuve de $\langle \Gamma^{\star s} \rangle, \neg \langle P^{\star s} \rangle \vdash_{ZM} \bot$
2. On construit une preuve sans réécriture Π_Z
2. Π_{Kleene} : instanciation des variables de type à la racine
3. Π_{mono} : sous-preuve sans formule polymorphe
4. Traduction syntaxique des noeuds de preuve résultant en B :

$$\dfrac{\dfrac{\langle \Pi_{mono} \rangle^{-1}}{\Gamma, \neg P \vdash_B \bot} \quad \dfrac{}{\Gamma, \neg P \vdash_B \neg \bot}\,\text{BR6}}{\Gamma \vdash_B P}\,\text{R5}$$

Conclusion

Conclusion

Résultats pratiques

- Théorie B en Why3
- Extension de Zenon au typage et à la réécriture : Zenon Modulo
- Résultats expérimentaux dans BWare
- 100% des preuves certifiées par Dedukti

Résultats théoriques

- Encodage des formules B en Poly-FOL
- Traduction des preuves Zenon Modulo vers B

Publications

- 3 conférences internationales
- 4 workshops

Perspective : certifier les traces d'autres prouveurs

TSTP

- format de traces de preuve utilisé par la communauté
- Liste d'axiomes nécessaires et lemmes intermédiaires

Utiliser Zenon Modulo pour générer des certificats

- Sélection d'axiomes : réduit l'espace de recherche
- lemmes intermédiaires : guide la preuve

Perspectives : implémenter la traduction vers B

Atelier B

- Noyau de preuve certifié
- Utilisé comme vérificateur de preuve B

Traduction des preuves de Zenon Modulo

- Génére des certificats de preuve B
- Sans l'OP et la théorie
- Pas de risque de bug dans la traduction de l'énoncé