Preuves assistées par ordinateur

Hugo Herbelin, Théo Zimmermann

basé sur du matériel d'Alexandre Miquel et Pierre Letouzey

De l'importance de la logique, notamment en informatique

- une société très dépendante de l'informatique
- des bugs qui coûtent chers :
  - explosion d'Ariane 5 en raison d'une erreur de calcul de flottant
  - accidents automobiles
  - accidents médicaux, ...
  - vulnérabilité Heartbleed dans openssh, vulnérabilité de gnuPG, vulnérabilité Log4shell dans Log4j, ...
  - etc., etc.

$\hookrightarrow$ spécifier le comportement des programmes et prouver leur correction vis à vis de la spécification

De grands projets de certification de programmes

- le compilateur C certifié CompCert (Leroy *et al*) et ses extensions

- le micro-noyau certifié SeL4

- la logique de séparation IRIS, Fiat, Vellum, ...

- la chaîne de certification DeepSpec

- ...

Les outils de certification

- La limite des tests et de l'automatisation : comment tester un nombre infini de comportements possibles en un temps fini ?

- Les assistants à la preuve, des outils :

  - pour spécifier

  - pour combiner raisonnement par récurrence, raisonnement équationnel et preuve automatique

  - fournissant toute la puissance du raisonnement mathématique

  - pour aboutir à la certification de programmes, c'est-à-dire à la preuve que ces programmes sont conformes à leur spécification, c'est-à-dire à ce que l'on attend d'eux

  $\hookrightarrow$ Coq, Isabelle, HOL4, ...

- sans oublier la question informelle de déterminer ce qu'on veut spécifier, qui est forcément une approximation du réel (p.ex. voudra-t-on spécifier qu'on ne peut pas surveiller une communication cryptée en mesurant la dissipation de chaleur du processeur ?)

Au cœur de la certification : la logique

- Socrate et les syllogismes

- Boole, Frege et la logique formelle du 19e siècle

- Cantor, Russell et les fondations ensemblistes du début 20e

- Hilbert, Gödel, Gentzen et l'étude de la structure des preuves des années 1930

- Church, Curry, Kleene, Turing et les fondations calculatoires des années 1930

- Curry, Howard et la correspondance entre preuves et programmes

- Martin-Löf, Coquand, Voevodsky : la théorie des types comme fondation alternative

Le développement de la logique formelle (Boole 1854, Peirce, de Morgan)

- connecteurs : $\wedge$, $\vee$, $\neg$, $\top$, $\bot$, $\Rightarrow$, $\Leftrightarrow$

- quantificateurs : $\forall$, $\exists$

- domaine du discours : x, $f(t,u)$, ...

- propositions, prédicats : $P(t,u)$, $t = u$, ...

- propriétés algébriques : $\neg(A \vee B) \Leftrightarrow (\neg A \wedge \neg B)$, ...

- premier système formel de preuves (Frege 1879) : $\Gamma \vdash A$

Les fondations mathématiques

- l'arithmétique de Peano (1889) : $\mathbb{N}$, $0 \neq 1$, induction, ...

- la théorie des ensembles naïve de Cantor (1874), de Zermelo (Z, 1908), de Zermelo-Fraenkel (ZF, 1922) : $t \in u$, $\emptyset$, $\{x|P(x)\}$, $x \mapsto t$, ...

- Principia Mathematica (Russell-Whitehead, 1910)

Les fondements de la provabilité (Hilbert, Gödel 1932, Gentzen 1934, ...)

- cohérence (peut-on prouver l'absurde ?) : $\vdash \perp$?
  - Gödel met une fin au programme de Hilbert : on ne peut pas prouver la cohérence d'un système arithmétique sans faire appel à un système plus puissant que celui qu'on considère
  - dans tout système logique cohérent, il y a des propositions ni prouvables ni réfutables
  - vérifier une preuve est décidable mais, dès qu'on parle de nombres, l'existence d'une preuve n'est pas décidable
- relations entre langage et métalangage (complétude)
- systèmes axiomatiques (à la Hilbert) : $A \Rightarrow B \Rightarrow A$, Modus Ponens, ...
- déduction naturelle (Gentzen 1934) : $\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}$
- calcul « logistique » des séquents (Gentzen 1934) : $\Gamma \vdash \Delta$

Les fondements de la calculabilité (Curry 1930, Church 1932, Turing 1935, ...)

- des modèles universels de calcul :
  - machine de Turing (1935)
  - logique combinatoire (Schönfinkel 1924, Haskell Curry 1930) : K, S et application
  - λ-calcul (Church 1932) : x, $\lambda x.t$, $t\,u$
  - fonctions partielles récursives (Gödel 1934, Kleene) : $T(n, p, q)$, théorème smn, ...
- la théorie des types simples (STT, Church 1940) aussi connue sous le nom de logique d'ordre supérieur (HOL), qui sont des variantes du Système F_ω de Girard

Les prémices d'un lien entre preuves et programmes

- logique intuitionniste (Brouwer ∼1920, Kolmogorov 1925, Heyting 1930) : $A \vee \neg A$ rejeté

- réalisabilité (Kleene 1945, Gödel 1958, Kreisel 1959) : si on a $\vdash \forall x \, \exists y \, P(x, y)$ alors il existe un programme f tel que pour toute valeur t, on a $\vdash P(t, f(t))$

La correspondance directe entre preuves et programmes

- système axiomatique Automath basé sur le λ-calcul (de Bruijn 1960's)

- système axiomatique = logique combinatoire (Curry 1934)

- déduction naturelle = λ-calcul (Howard 1968)

- théorie des types de Martin-Löf d'abord naïve (**Type** : **Type** en 1970), puis prédicative, un formalisme qui est à la fois une logique et un langage de programmation (MLTT)

autres exemples

- calcul des séquents = machine abstraite

- logique propositionnelle du second ordre = Système F de Girard et Reynolds

- $\perp \Rightarrow A$ = opérateurs d'interruption du flux de contrôle (`return`, `break`, `exit`, ...)

- $A \vee \neg A$ = opérateurs de contrôle (`goto`, `callcc`, ...)

- forcing $\simeq$ mémoire modifiable (`x <- t`)

- modèles syntaxiques $\simeq$ Lisp's `quote`

L'implémentation de logiciels de développement et vérification de preuves (assistants à la preuve, aussi connus comme prouveur de théorèmes interactifs - ITP)

- Mizar (Białistok, 1973), basé sur la théorie des ensembles

- Isabelle/ZF, Isabelle/HOL (Cambridge, Munich 1990-)

- Coq, basé sur le calcul des constructions inductives (CC 1984, puis CIC 1990, étendant MLTT), puis Lego, Matita, Lean

- NuPrl (université de Cornell), Agda (université de Göteborg), basés sur d'autres variantes de théorie des types

La question des fondements à la fin du 20e et début du 21e siècle

- la logique linéaire au cœur d'une décomposition des connecteurs (Girard, 1987)

- la théorie des types homotopiques (à partir de 2005) établit de nouveaux liens entre logique et géométrie : $t =_A u$ représente l'ensemble des chemins entre deux points t et u d'un espace A

- la logique polarisée et les adjonctions catégoriques au cœur de la description des effets de bord

- une convergence entre logique, géométrie, informatique, algèbre :
  - théorie des types = topos
  - propositions $\subset$ espaces = types = catégories s
  - types inductifs = colimites = types positifs : $\mathbb{N}$, listes, arbres
  - types coinductifs = limites = types négatifs : streams, ...

Un autre pan des assistants à la preuve : la formalisation des mathématiques

- une preuve formelle exhaustive du théorème des 4 couleurs (Gonthier *et al*, 2005)

- le théorème de l'ordre impair, dit Feit-Thompson (Gonthier *et al*, 2012)

- une preuve formelle de la conjecture de Kepler sur l'empilement optimal des oranges sur l'étal du primeur (Hales *et al*, 2014)

- un intérêt croissant pour la formalisation des mathématiques par les mathématiciens

- la bibliothèque Mathlib

Les objectifs du cours

- notions de base de logique (formules, calcul des propositions, calcul des prédicats, cohérence, décidabilité)

- les logiques standard (arithmétique, théorie des ensembles, Système F, HOL, théorie des types)

- la structure des preuves et leur lien avec les programmes (Système T, récurrence et récursion, raisonnement égalitaire)

- utilisation d'un assistant à la preuve pour programmer, spécifier, prouver