Proving with Computer Assistance

Lecture

Simple Type theory and *Formulas-as-Types* for propositional logic

Herman Geuvers

Typed λ calculus as the basis for a Proof Assistant

Typed λ calculus forms the basis for a variety of proof Assistants, e.g. Coq (and Lean, Agda, Nuprl, Matita).

λ-term	type
program	specification
proof	formula

Integrated system for proving and programming

Types are not sets

Types are a bit like sets, but types give syntactic information, e.g.

$$3 + (7 \times 8)^5 \;:\; \mathsf{nat}$$

whereas sets give semantic information, e.g.

$$3 \;\in\; \{n \in \mathbb{N} \mid \forall x, y, z \in \mathbb{N}^+(x^n + y^n \neq z^n)\}.$$

- $3 + (7 \times 8)^5$ is of type nat because $3, 7, 8$ are natural numbers and $\times$, $+$ and power are operations on natural numbers.
- $3 \in \{n \in \mathbb{N} \mid \forall x, y, z \in \mathbb{N}^+(x^n + y^n \neq z^n)\}$ because there are no positive x, y, z such that $x^3 + y^3 = z^3$, which is an instance of Fermat's last Theorem, proved by Wiles.
- To establish that 3 is an element of the given set, we need a proof, we can't just read it off from the components of the statement.
- To establish $3 + (7 \times 8)^5$: nat we don't need a proof but a simple computation (the "reading the type of of the term").

Decidability of :, undecidability of $\in$

- Membership is undecidable in set theory, as it requires a proof to establish $a \in A$.
- Type checking is decidable: Verifying whether M is of type A requires purely syntactic methods, which can be cast into a typing algorithm.

$$3 + (7 \times 8)^5 : \mathsf{nat} \qquad \text{versus} \qquad \frac{1}{2}\sum_{n=0}^{\infty} 2^{-n} \in \mathbb{N}$$

Question: Can we turn (e.g.)

$$\{n \in \mathbb{N} \mid \forall x, y, z \in \mathbb{N}^{+}(x^n + y^n \neq z^n)\}$$

into a (syntactic) type, with decidable type checking?

Phrased differently: can we talk about this set as a "subtype of nat"?

Formulas are also types; proofs are terms

$$\{n \in \mathsf{nat} \mid \forall x, y, z \in \mathbb{N}^+(x^n + y^n \neq z^n)\}$$

is a type.
Its terms are pairs $\langle n, p \rangle$ where

- n : nat
- $p : \forall x, y, z \in \mathbb{N}^+(x^n + y^n \neq z^n)$

So p is a proof, and we view the formula $\forall x, y, z \in \mathbb{N}^+(x^n + y^n \neq z^n)$ as the type of its proofs.

If we have decidable proof checking, then it is decidable whether a given pair $\langle n, p \rangle$ is typable with the above type or not.

We summarize:

- proof checking = type checking,
- type checking is decidable (so proof checking is decidable),
- proof finding is not decidable (proof finding is required to check an $\in$-judgment).

Simple Type Theory

Simplest system: $\lambda\!\to$ or simple type theory, STT. Just arrow types

$$\mathsf{Typ} := \mathsf{TVar} \mid (\mathsf{Typ}\to\mathsf{Typ})$$

- Examples: $(\alpha\to\beta)\to\alpha$, $(\alpha\to\beta)\to((\beta\to\gamma)\to(\alpha\to\gamma))$
- Brackets associate to the right and outside brackets are omitted:
  $(\alpha\to\beta)\to(\beta\to\gamma)\to\alpha\to\gamma$
- Types are denoted by $\sigma, \tau, \ldots$.

Terms:

- typed variables $x_1^\sigma, x_2^\sigma, \ldots$, countably many for every σ.
- application: if $M : \sigma\to\tau$ and $N : \sigma$, then $(MN) : \tau$
- abstraction: if $P : \tau$, then $(\lambda x^\sigma.P) : \sigma\to\tau$

Examples of simply typed terms

$$\begin{array}{rcl}
\lambda x^{\sigma}.\lambda y^{\tau}.x & : & \sigma{\to}\tau{\to}\sigma \\
\lambda x^{\alpha\to\beta}.\lambda y^{\beta\to\gamma}.\lambda z^{\alpha}.y(xz) & : & (\alpha{\to}\beta){\to}(\beta{\to}\gamma){\to}\alpha{\to}\gamma \\
\lambda x^{\alpha}.\lambda y^{(\beta\to\alpha)\to\alpha}.y(\lambda z^{\beta}.x) & : & \alpha{\to}((\beta{\to}\alpha){\to}\alpha){\to}\alpha
\end{array}$$

For every type there is a term of that type:

$$x^{\sigma} : \sigma$$

Not for every type there is a closed term of that type:

$(\alpha{\to}\alpha){\to}\alpha$ is not inhabited

[That is: there is no closed term of type $(\alpha{\to}\alpha){\to}\alpha$.]

Church' simple type theory

Church formulation of simple type theory:terms with type information.
Inductive definition of the terms:

- typed variables $x_1^\sigma, x_2^\sigma, \ldots$, countably many for every σ.
- application: if $M : \sigma \to \tau$ and $N : \sigma$, then $(MN) : \tau$
- abstraction: if $P : \tau$, then $(\lambda x^\sigma.P) : \sigma \to \tau$

Alternative: Inductive definition of the terms in rule form:

$$\frac{}{x^\sigma : \sigma} \qquad \frac{M : \sigma \to \tau \quad N : \sigma}{MN : \tau} \qquad \frac{P : \tau}{\lambda x^\sigma.P : \sigma \to \tau}$$

Advantage: We also have a derivation tree, a proof of the fact that the term has that type.
We can reason over derivations.

Simple type theory à la Church with contexts

Formulation with contexts to declare the free variables:

$$x_1 : \sigma_1, x_2 : \sigma_2, \ldots, x_n : \sigma_n$$

is a context, usually denoted by Γ.
Derivation rules of $\lambda\!\to$ (à la Church):

$$\frac{x{:}\sigma \in \Gamma}{\Gamma \vdash x : \sigma} \qquad \frac{\Gamma \vdash M : \sigma\to\tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau} \qquad \frac{\Gamma, x{:}\sigma \vdash P : \tau}{\Gamma \vdash \lambda x{:}\sigma.P : \sigma\to\tau}$$

$\Gamma \vdash_{\lambda\to} M : \sigma$ if there is a derivation using these rules with conclusion $\Gamma \vdash M : \sigma$

Reading the typing rules top down

Inductive definition of the "derivable judgments"

$$\frac{x{:}\sigma \in \Gamma}{\Gamma \vdash x : \sigma} \qquad \frac{\Gamma \vdash M : \sigma\to\tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau} \qquad \frac{\Gamma, x{:}\sigma \vdash P : \tau}{\Gamma \vdash \lambda x{:}\sigma.P : \sigma\to\tau}$$

Deriving

$$\vdash \lambda x{:}\alpha.\lambda y{:}(\beta\to\alpha)\to\alpha.y(\lambda z{:}\beta.x) : \alpha\to((\beta\to\alpha)\to\alpha)\to\alpha$$

Reading the typing rules bottom up

Trying to solve a typing problem / an inhabitation problem

$$\frac{x{:}\sigma \in \Gamma}{\Gamma \vdash x : \sigma} \qquad \frac{\Gamma \vdash M : \sigma{\to}\tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau} \qquad \frac{\Gamma, x{:}\sigma \vdash P : \tau}{\Gamma \vdash \lambda x{:}\sigma.P : \sigma{\to}\tau}$$

Formulas-as-Types (Curry, Howard)

There are two readings of a judgement $M : \sigma$

1. term as algorithm/program, type as specification:
   M is a function of type σ
2. type as a proposition, term as its proof:
   M is a proof of the proposition σ

- There is a one-to-one correspondence:

  typable terms in $\lambda\!\rightarrow$ $\simeq$ derivations in minimal proposition logic

- $x_1 : \tau_1, x_2 : \tau_2, \ldots, x_n : \tau_n \vdash M : \sigma$ can be read as
  M is a proof of σ from the assumptions $\tau_1, \tau_2, \ldots, \tau_n$.

Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\alpha\to\beta\to\gamma]^3 \quad [\alpha]^1}{\beta\to\gamma} \qquad \dfrac{[\alpha\to\beta]^2 \quad [\alpha]^1}{\beta}}{\gamma}}{\alpha\to\gamma}\,1}{(\alpha\to\beta)\to\alpha\to\gamma}\,2}{(\alpha\to\beta\to\gamma)\to(\alpha\to\beta)\to\alpha\to\gamma}\,3 \quad \simeq \quad \begin{array}{c} \lambda x{:}\alpha\to\beta\to\gamma.\lambda y{:}\alpha\to\beta.\lambda z{:}\alpha.xz(yz) \\ : (\alpha\to\beta\to\gamma)\to(\alpha\to\beta)\to\alpha\to\gamma \end{array}$$

Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[x : \alpha\to\beta\to\gamma]^3\quad[z : \alpha]^1}{xz : \beta\to\gamma}\qquad\dfrac{[y : \alpha\to\beta]^2\quad[z : \alpha]^1}{yz : \beta}}{xz(yz) : \gamma}}{\lambda z{:}\alpha.xz(yz) : \alpha\to\gamma}\;1}{\lambda y{:}\alpha\to\beta.\lambda z{:}\alpha.xz(yz) : (\alpha\to\beta)\to\alpha\to\gamma}\;2}{\lambda x{:}\alpha\to\beta\to\gamma.\lambda y{:}\alpha\to\beta.\lambda z{:}\alpha.xz(yz) : (\alpha\to\beta\to\gamma)\to(\alpha\to\beta)\to\alpha\to\gamma}\;3$$

Exercise: Give the derivation that corresponds to

$$\lambda x{:}\gamma\to\varepsilon.\lambda y{:}(\gamma\to\varepsilon)\to\varepsilon.y(\lambda z.y\,x) : (\gamma\to\varepsilon)\to((\gamma\to\varepsilon)\to\varepsilon)\to\varepsilon$$

Flag style deductions

The Fitch style (also: flag style) presentation of $\lambda\!\to$.

$$\begin{array}{r|l|ll}
1 & & x : \sigma & \\ \cline{3-3}
2 & & \dots & \\
3 & & \dots & \\
4 & & M : \tau & \\
5 & \multicolumn{2}{l}{\lambda x{:}\sigma.M : \sigma \to \tau} & \text{abs, 1, 4}
\end{array}$$

abs-rule

$$\begin{array}{r|ll}
1 & \dots & \\
2 & \dots & \\
3 & M : \sigma \to \tau & \\
4 & \dots & \\
5 & \dots & \\
6 & N : \sigma & \\
7 & \dots & \\
8 & M\,N : \tau & \text{app, 3, 6}
\end{array}$$

app-rule

Example

$$\begin{array}{r|l}
1 & \;\Big|\; \underline{x : \alpha\to\beta\to\gamma} \\
2 & \;\Big|\;\Big|\; \underline{y : \alpha\to\beta} \\
3 & \;\Big|\;\Big|\;\Big|\; \underline{z : \alpha} \\
4 & \;\Big|\;\Big|\;\Big|\; x\,z : \beta\to\gamma \\
5 & \;\Big|\;\Big|\;\Big|\; y\,z : \beta \\
6 & \;\Big|\;\Big|\;\Big|\; x\,z(y\,z) : \gamma \\
7 & \;\Big|\;\Big|\; \lambda z{:}\alpha.x\,z(y\,z) : \alpha\to\gamma \\
8 & \;\Big|\; \lambda y{:}\alpha\to\beta.\lambda z{:}\alpha.x\,z(y\,z) : (\alpha\to\beta)\to\alpha\to\gamma \\
9 & \lambda x{:}\alpha\to\beta\to\gamma.\lambda y{:}\alpha\to\beta.\lambda z{:}\alpha.x\,z(y\,z) : (\alpha\to\beta\to\gamma)\to(\alpha\to\beta)\to\alpha\to\gamma
\end{array}$$

Computation

- β-reduction: $(\lambda x{:}\sigma.M)P \to_\beta M[x := P]$

Cut-elimination

Cut-eliminationin minimal logic $=$ β-reduction in $\lambda{\to}$.

$$\dfrac{\dfrac{\begin{matrix}[\sigma]^1\\ \mathcal{D}_1\\ \tau\end{matrix}}{\sigma{\to}\tau}\,1 \qquad \dfrac{\mathcal{D}_2}{\sigma}}{\tau} \qquad \longrightarrow \qquad \begin{matrix}\mathcal{D}_2\\ \sigma\\ \mathcal{D}_1\\ \tau\end{matrix}$$

$$\dfrac{\dfrac{\begin{matrix}[x:\sigma]^1\\ \mathcal{D}_1\\ M:\tau\end{matrix}}{\lambda x{:}\sigma.M:\sigma{\to}\tau}\,1 \qquad \dfrac{\mathcal{D}_2}{P:\sigma}}{(\lambda x{:}\sigma.M)P:\tau} \qquad \longrightarrow_\beta \qquad \begin{matrix}\mathcal{D}_2\\ P:\sigma\\ \mathcal{D}_1\\ M[x:=P]:\tau\end{matrix}$$

Example

Proof of $A{\to}A{\to}B, (A{\to}B){\to}A \vdash B$

$$\dfrac{\dfrac{\dfrac{[A]^1 \quad \dfrac{[A]^1 \quad A{\to}A{\to}B}{A{\to}B}}{B}}{A{\to}B} \qquad \dfrac{(A{\to}B){\to}A \quad \dfrac{\dfrac{[A]^1 \quad \dfrac{[A]^1 \quad A{\to}A{\to}B}{A{\to}B}}{B}}{A{\to}B}}{A}}{B}$$

It contains a cut: a $\to$-i directly followed by an $\to$-e.

Example ctd

Proof of $A\to A\to B, (A\to B)\to A \vdash B$ after reduction

$$\dfrac{\dfrac{(A\to B)\to A \qquad \dfrac{\dfrac{[A]^1 \qquad \dfrac{[A]^1 \quad A\to A\to B}{A\to B}}{B}}{A\to B}}{A} \qquad \dfrac{\dfrac{(A\to B)\to A \qquad \dfrac{\dfrac{[A]^1 \qquad \dfrac{[A]^1 \quad A\to A\to B}{A\to B}}{B}}{A\to B}}{A} \qquad A\to A\to}{A\to B}}{B}$$

Example ctd

Proof of $A{\to}A{\to}B, (A{\to}B){\to}A \vdash B$ with term information.

$$
\dfrac{
  \dfrac{
    \dfrac{
      [y : A]^1 \quad \dfrac{[y : A]^1 \quad p : A{\to}A{\to}B}{p\,y : A{\to}B}
    }{p\,y\,y : B}
  }{\lambda y{:}A.p\,y\,y : A{\to}B}
  \qquad
  \dfrac{
    q : (A{\to}B){\to}A \quad
    \dfrac{
      \dfrac{
        [x : A]^1 \quad \dfrac{[x : A]^1 \quad p : A{\to}A{\to}B}{p\,x : A{\to}B}
      }{p\,x\,x : B}
    }{\lambda x{:}A.p\,x\,x : A{\to}B}
  }{q(\lambda x{:}A.p\,x\,x) : A}
}{(\lambda y{:}A.p\,y\,y)(q(\lambda x{:}A.p\,x\,x)) : B}
$$

Term contains a β-redex: $(\lambda x{:}A.p\,x\,x)\,(q(\lambda x{:}A.p\,x\,x))$

Example ctd

Reduced proof of $A \to A \to B, (A \to B) \to A \vdash B$ with term info.

$$
\dfrac{
\dfrac{
q : (A \to B) \to A
\qquad
\dfrac{
\dfrac{
[x : A]^1
\qquad
\dfrac{[x : A]^1 \quad p : A \to A \to B}{p\,x : A \to B}
}{p\,x\,x : B}
}{\lambda x{:}A.\,p\,x\,x : A \to B}
}{q(\lambda x{:}A.\,p\,x\,x) : A}
\qquad
\dfrac{
\dfrac{
q : (A \to B) \to A
\qquad
\dfrac{
\dfrac{
[x : A]^1
\qquad
\dfrac{[x : A]^1 \quad p : A \to A \to B}{p\,x : A \to B}
}{p\,x\,x : B}
}{\lambda x{:}A.\,p\,x\,x : A \to B}
}{q(\lambda x{:}A.\,p\,x\,x) : A}
\qquad
p : A \to A \to B
}{p(q(\lambda x{:}A.\,p\,x\,x)) : A \to B}
}{p(q(\lambda x{:}A.\,p\,x\,x))(q(\lambda x{:}A.\,p\,x\,x)) : B}
$$

Extension with other connectives: STT with product types $\times$ (proposition logic with conjunction $\wedge$)

$$\frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_1 M : \sigma} \qquad \frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_2 M : \tau} \qquad \frac{\Gamma \vdash P : \sigma \quad \Gamma \vdash Q : \tau}{\Gamma \vdash \langle P, Q \rangle : \sigma \times \tau}$$

With reduction rules

$$\begin{array}{lcl} \pi_1 \langle P, Q \rangle & \to & P \\ \pi_2 \langle P, Q \rangle & \to & Q \end{array}$$