Type Theory

Lecture 1: Natural Deduction and Curry-Howard

Andreas Abel

Department of Computer Science and Engineering
Chalmers and Gothenburg University

ESSLLI 2016
28th European Summer School in Logic, Language, and Information
unibz, Bozen/Bolzano, Italy
15-19 August 2016

Contents

Constructivism

- Brouwer's intuitionism in opposition to Hilbert's formalism
- Constructive logic vs. classical logic
- Disjunction property

  If the disjunction $A \vee B$ is provable, then either A is provable or B is provable.

- Drop principle of excluded middle $A \vee \neg A$
- Propositions A with $A \vee \neg A$ are called *decidable*
- Existence property

  A proof of the existential statement $\exists x.\, A(x)$ includes an algorithm to compute a witness t with $A(t)$.

Brouwer-Heyting-Kolmogorov Interpretation

Characterizing canonical proofs.

- A proof of $A \wedge B$ is a pair of a proof of A and a proof of B.
- A proof of $A \vee B$ is a proof of A or a proof of B, plus a bit indicating which of the two.
- A proof of $A \Rightarrow B$ is an algorithm computing a proof of B given a proof of A.
- No canonical proof of $\bot$ exists (consistency!).
- A proof of $\neg A$ is a proof of $A \Rightarrow \bot$.
- A proof of $\forall x.A(x)$ is an algorithm computing a proof of $A(t)$ given any object t.
- A proof of $\exists x.A(x)$ is a pair of a witness t and a proof of $A(t)$.

Propositional logic

- Formulæ

$$
\begin{array}{lll}
P, Q & & \text{atomic proposition} \\
A, B, C ::= & P & \\
 & \mid A \Rightarrow B & \text{implication} \\
 & \mid A \wedge B \mid \top & \text{conjunction, truth} \\
 & \mid A \vee B \mid \bot & \text{disjunction, absurdity}
\end{array}
$$

- Formula = (binary) abstract syntax tree
- Subformula = subtree
- Principal connective = root label

Well-formedness vs. truth

- Let

$$\begin{array}{lcl} \mathsf{SH} & := & \text{“Socrates is a human”} \\ \mathsf{FL} & := & \text{“Socrates has four legs”} \end{array}$$

- Implication $\mathsf{SH} \Rightarrow \mathsf{FL}$ is well-formed.
- Implication $\mathsf{SH} \Rightarrow \mathsf{FL}$ is not necessarily true ;-).

$$\mathsf{SH} \Rightarrow \mathsf{FL}\ \mathit{true}$$

is a *judgement* which requires *proof*

Judgements and derivations

- Propositional logic has a single judgement form $A\ \mathit{true}$.
- J refers to a judgement.
- Inference rules have form

$$\frac{J_1 \ldots J_n}{J}\ r$$

- Derivation (trees):

$$\dfrac{\dfrac{}{J_1}\ r_1 \qquad \dfrac{\dfrac{}{J_3}\ r_3 \qquad J_4 \qquad J_5}{J_2}\ r_2}{J_0}\ r_0$$

- $D_0 :: J_0$ with $\mathcal{D}_0 = r_0^{J_0}(r_1^{J_1}, r_2^{J_2}(r_3^{J_3}, \mathcal{D}_4, \mathcal{D}_5))$

Introduction and elimination

- Introduction rules: composing information

$$\frac{A\ \mathit{true} \qquad B\ \mathit{true}}{A \wedge B\ \mathit{true}}\ \wedge\mathrm{I}$$

- Elimination rules: retrieving/using information

$$\frac{A \wedge B\ \mathit{true}}{A\ \mathit{true}}\ \wedge\mathrm{E}_1 \qquad \frac{A \wedge B\ \mathit{true}}{B\ \mathit{true}}\ \wedge\mathrm{E}_2$$

- Orthogonality: define meaning of logical connective (e.g. $\wedge$) independently of other connectives (e.g. $\Rightarrow$).

Local soundness

- Introductions followed immediately by eliminations are a removable detour.

$$\dfrac{\dfrac{\begin{array}{c}\mathcal{D}_1\\ A\ \mathit{true}\end{array} \quad \begin{array}{c}\mathcal{D}_2\\ B\ \mathit{true}\end{array}}{A \wedge B\ \mathit{true}}\ \wedge\mathsf{I}}{A\ \mathit{true}}\ \wedge\mathsf{E}_1 \quad \longrightarrow_\beta \quad \begin{array}{c}\mathcal{D}_1\\ A\ \mathit{true}\end{array}$$

$$\dfrac{\dfrac{\begin{array}{c}\mathcal{D}_1\\ A\ \mathit{true}\end{array} \quad \begin{array}{c}\mathcal{D}_2\\ B\ \mathit{true}\end{array}}{A \wedge B\ \mathit{true}}\ \wedge\mathsf{I}}{B\ \mathit{true}}\ \wedge\mathsf{E}_2 \quad \longrightarrow_\beta \quad \begin{array}{c}\mathcal{D}_2\\ B\ \mathit{true}\end{array}$$

- Otherwise, an elimination rule is too strong (unsound).
- *Exercise: Give a unsound, too strong $\wedge$E-rule.*

Local completeness

- Reconstruct a judgement by introduction from parts obtained by elimination.

$$\begin{array}{c}\mathcal{D}\\ A \wedge B \; \mathit{true}\end{array} \quad \longrightarrow_{\eta^-} \quad \dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ A \wedge B \; \mathit{true}\end{array}}{A \; \mathit{true}} \wedge\mathrm{E}_1 \qquad \dfrac{\begin{array}{c}\mathcal{D}\\ A \wedge B \; \mathit{true}\end{array}}{B \; \mathit{true}} \wedge\mathrm{E}_2}{A \wedge B \; \mathit{true}} \wedge\mathrm{I}$$

- Otherwise, elimination rules are too weak (incomplete).
- *Exercise: Give a set of* $\wedge$E*-rules which is incomplete.*

Truth

- Introduction of trivial proposition $\top$:

$$\frac{}{\top\ \mathit{true}}\ \top I$$

- No information to obtain by elimination!
- No β-reduction.
- η-expansion:

$$\begin{array}{c}\mathcal{D}\\ \top\ \mathit{true}\end{array} \quad \longrightarrow_{\eta^-} \quad \frac{}{\top\ \mathit{true}}\ \top I$$

Proving an implication

- How to prove $(A \wedge B) \Rightarrow (B \wedge A)\ \mathit{true}$?
- First, construct an open derivation:

$$\dfrac{\dfrac{A \wedge B\ \mathit{true}}{B\ \mathit{true}} \qquad \dfrac{A \wedge B\ \mathit{true}}{A\ \mathit{true}}}{B \wedge A\ \mathit{true}}$$

- Then, close by discharging the hypothesis $x :: A \wedge B\ \mathit{true}$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \wedge B\ \mathit{true}}\,x}{B\ \mathit{true}} \qquad \dfrac{\dfrac{}{A \wedge B\ \mathit{true}}\,x}{A\ \mathit{true}}}{B \wedge A\ \mathit{true}}}{(A \wedge B) \Rightarrow (B \wedge A)\ \mathit{true}}\Rightarrow\!\mathrm{I}_x$$

Rules for implication

- Elimination = modus ponens

$$\frac{A \Rightarrow B\ \mathit{true} \qquad A\ \mathit{true}}{B\ \mathit{true}} \Rightarrow\mathrm{E}$$

- Introduction = internalizing a meta-implication (hypothetical judgement)

$$\dfrac{\begin{array}{c}\overline{A\ \mathit{true}}\ x \\ \vdots \\ B\ \mathit{true}\end{array}}{A \Rightarrow B\ \mathit{true}} \Rightarrow\mathrm{I}_x$$

- *Exercise: How many different derivations of* $A \Rightarrow (A \Rightarrow A)\ \mathit{true}$*?*

Substitution

- β-reduction replaces hypothesis x by derivation $\mathcal{D}$:

$$\dfrac{\dfrac{\begin{array}{c}\dfrac{}{A\ \mathit{true}}\,x \\ \vdots\ \mathcal{E} \\ B\ \mathit{true}\end{array}}{A \Rightarrow B\ \mathit{true}}\Rightarrow\!\mathsf{I}_x \qquad \begin{array}{c}\mathcal{D}\\ A\ \mathit{true}\end{array}}{B\ \mathit{true}}\Rightarrow\!\mathsf{E} \qquad \longrightarrow_\beta \qquad \begin{array}{c}\mathcal{D}\\ A\ \mathit{true}\\ \vdots\ \mathcal{E}\\ B\ \mathit{true}\end{array}$$

- More precise notation:

$$\begin{array}{c}\vdots\ \mathcal{E}[\mathcal{D}/x]\\ B\ \mathit{true}\end{array}$$

Local completeness for implication

- η-expansion

$$\begin{array}{c}\mathcal{D}\\ A \Rightarrow B\ \mathit{true}\end{array} \quad \longrightarrow_{\eta^-} \quad \dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ A \Rightarrow B\ \mathit{true}\end{array} \qquad \dfrac{}{A\ \mathit{true}}\,x}{B\ \mathit{true}}\,{\Rightarrow}\mathrm{E}}{A \Rightarrow B\ \mathit{true}}\,{\Rightarrow}\mathrm{I}_x$$

Disjunction

- Introduction: choosing an alternative

$$\frac{A\ \mathit{true}}{A \vee B\ \mathit{true}}\ \vee\mathrm{I}_1 \qquad \frac{B\ \mathit{true}}{A \vee B\ \mathit{true}}\ \vee\mathrm{I}_2$$

- Elimination: case distinction

$$\frac{A \vee B\ \mathit{true} \qquad \begin{array}{c}\overline{A\ \mathit{true}}^{\,x} \\ \vdots \\ C\ \mathit{true}\end{array} \qquad \begin{array}{c}\overline{B\ \mathit{true}}^{\,y} \\ \vdots \\ C\ \mathit{true}\end{array}}{C\ \mathit{true}}\ \vee\mathrm{E}_{x,y}$$

Disjunction: local soundness

$$\dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ A\ \mathit{true}\end{array}}{A \vee B\ \mathit{true}}\vee\mathrm{I}_1 \qquad \begin{array}{c}\overline{A\ \mathit{true}}^{\,x}\\ \vdots\ \mathcal{E}_1\\ C\ \mathit{true}\end{array} \qquad \begin{array}{c}\overline{B\ \mathit{true}}^{\,y}\\ \vdots\ \mathcal{E}_2\\ C\ \mathit{true}\end{array}}{C\ \mathit{true}}\vee\mathrm{E}_{x,y} \quad \longrightarrow_\beta \quad \begin{array}{c}\vdots\ \mathcal{E}_1[\mathcal{D}/x]\\ C\ \mathit{true}\end{array}$$

$$\dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ B\ \mathit{true}\end{array}}{A \vee B\ \mathit{true}}\vee\mathrm{I}_2 \qquad \begin{array}{c}\overline{A\ \mathit{true}}^{\,x}\\ \vdots\ \mathcal{E}_1\\ C\ \mathit{true}\end{array} \qquad \begin{array}{c}\overline{B\ \mathit{true}}^{\,y}\\ \vdots\ \mathcal{E}_2\\ C\ \mathit{true}\end{array}}{C\ \mathit{true}}\vee\mathrm{E}_{x,y} \quad \longrightarrow_\beta \quad \begin{array}{c}\vdots\ \mathcal{E}_2[\mathcal{D}/y]\\ C\ \mathit{true}\end{array}$$

Disjunction: local completeness

Introduction happens in branches of elimination:

$$\begin{array}{c}\mathcal{D}\\ A \vee B\ \mathit{true}\end{array} \longrightarrow_{\eta^-} \dfrac{\begin{array}{c}\mathcal{D}\\ A \vee B\ \mathit{true}\end{array} \quad \dfrac{\overline{A\ \mathit{true}}\ x}{A \vee B\ \mathit{true}}\ \vee\mathsf{I}_1 \quad \dfrac{\overline{B\ \mathit{true}}\ y}{A \vee B\ \mathit{true}}\ \vee\mathsf{I}_2}{A \vee B\ \mathit{true}}\ \vee\mathsf{E}_{x,y}$$

Absurdity and negation

- No introduction (phew!), strongest elimination:

$$\frac{\bot\ \mathit{true}}{C\ \mathit{true}}\ \bot\mathrm{E}$$

- Only global soundness (consistency).
- Negation is definable:

$$\neg A = A \Rightarrow \bot$$

- So is logical equivalence:

$$A \iff B = (A \Rightarrow B) \land (B \Rightarrow A)$$

Careful with discharging!

- Consider this derivation:

$$\dfrac{\dfrac{\dfrac{}{(A \Rightarrow A) \Rightarrow (A \Rightarrow A)\ \mathit{true}}\,f \qquad \dfrac{\dfrac{}{A\ \mathit{true}}\,x}{A \Rightarrow A\ \mathit{true}}\,{\Rightarrow}\mathrm{I}_x}{A \Rightarrow A\ \mathit{true}}\,{\Rightarrow}\mathrm{E} \qquad \dfrac{}{A\ \mathit{true}}\,x}{\dfrac{A\ \mathit{true}}{((A \Rightarrow A) \Rightarrow (A \Rightarrow A)) \Rightarrow A\ \mathit{true}}\,{\Rightarrow}\mathrm{I}_f}\,{\Rightarrow}\mathrm{E}$$

- Does it prove $((A \Rightarrow A) \Rightarrow (A \Rightarrow A)) \Rightarrow A\ \mathit{true}$?

Explicit hypotheses

- Explicitly hypothetical judgement:

$$A_1\ \mathit{true}, \ldots, A_n\ \mathit{true} \vdash C\ \mathit{true}$$

- New rule (with Γ: list of hypotheses)

$$\frac{A\ \mathit{true} \in \Gamma}{\Gamma \vdash A\ \mathit{true}}\ \mathrm{hyp}$$

- Implication rules

$$\frac{\Gamma, A\ \mathit{true} \vdash B\ \mathit{true}}{\Gamma \vdash A \Rightarrow B\ \mathit{true}}\ {\Rightarrow}\mathrm{I} \qquad \frac{\Gamma \vdash A \Rightarrow B\ \mathit{true} \qquad \Gamma \vdash A\ \mathit{true}}{\Gamma \vdash B\ \mathit{true}}\ {\Rightarrow}\mathrm{E}$$

- *Exercise: adapt the remaining rules to explicit hypotheses!*

Origins of lambda calculus

- Haskell Curry: untyped lambda-calculus as logical foundation (inconsistent)
- Alonzo Church: *Simple Theory of Types* (1936)
- Today: basis of functional programming languages

Untyped lambda-calculus

- Lambda-calculus with tuples and variants:

x, y, z		variables
r, s, t	$::= x \mid \lambda x.t \mid r\,s$	pure lambda-calculus
	$\mid \langle s,\ t\rangle \mid \mathsf{fst}\ r \mid \mathsf{snd}\ r$	pairs and projections
	$\mid \mathsf{inl}\ t \mid \mathsf{inr}\ t$	injections
	$\mid \mathsf{case}\ r\ \mathsf{of}\ \mathsf{inl}\,x \Rightarrow s \mid \mathsf{inr}\,y \Rightarrow t$	case distinction
	$\mid \langle\rangle$	empty tuple
	$\mid \mathsf{abort}\ r$	exception

- Free variables:

$$\begin{array}{lcl} \mathsf{FV}(x) & = & \{x\} \\ \mathsf{FV}(\lambda x.t) & = & \mathsf{FV}(t) \setminus \{x\} \\ \mathsf{FV}(r\,s) & = & \mathsf{FV}(r) \cup \mathsf{FV}(s) \\ \dots \end{array}$$

- *Exercise: Complete the definition of* FV*!*

Substitution and renaming

- $t[s/x]$ substitutes s for x in t:

$$
\begin{array}{lcll}
x[s/x] & = & s & \\
y[s/x] & = & y & \text{if } x \neq y \\
(t\,t')[s/x] & = & (t[s/x])\,(t[s/x]') & \\
(\lambda x.t)[s/x] & = & \lambda x.t & \\
(\lambda y.t)[s/x] & = & \lambda y.t[s/x] & \text{if } x \neq y \text{ and } y \notin \mathrm{FV}(s) \\
(\lambda y.t)[s/x] & = & \lambda y'.t[y'/y][s/x] & \text{if } x \neq y \text{ and } y' \notin \mathrm{FV}(x, y, s, t) \\
\dots & & &
\end{array}
$$

- Bound variables can be renamed (α-equivalence).

$$\lambda x.t \quad =_\alpha \quad \lambda x'.t[x'/x] \qquad \text{if } x' \notin \mathrm{FV}(t)$$

Simple types

- Types rule out meaningless/stuck terms like $\mathsf{fst}\,(\lambda x.x)$ and $(\lambda y.\,\mathsf{fst}\,y)\,(\lambda x.x)$.
- Simple types:

$$
\begin{array}{rcll}
R, S, T, U & ::= & S \to T & \text{function type} \\
& \mid & S \times T & \text{product type} \\
& \mid & S + T & \text{disjoint sum type} \\
& \mid & 1 & \text{unit type} \\
& \mid & 0 & \text{empty type}
\end{array}
$$

- Context Γ be a finite map from variables x to types T.

Type assignment

- Judgement $\Gamma \vdash t : T$ "in context Γ, term t has type T".
- Rules for functions:

$$\frac{\Gamma(x) = T}{\Gamma \vdash x : T}$$

$$\frac{\Gamma, x{:}S \vdash t : T}{\Gamma \vdash \lambda x.t : S \to T} \qquad \frac{\Gamma \vdash r : S \to T \qquad \Gamma \vdash s : S}{\Gamma \vdash r\, s : T}$$

- Rules for pairs:

$$\frac{\Gamma \vdash s : S \qquad \Gamma \vdash t : T}{\Gamma \vdash \langle s, t\rangle : S \times T} \qquad \frac{\Gamma \vdash r : S \times T}{\Gamma \vdash \mathsf{fst}\, r : S} \qquad \frac{\Gamma \vdash r : S \times T}{\Gamma \vdash \mathsf{snd}\, r : T}$$

Type assignment (ctd.)

- Rules for variants:

$$\frac{\Gamma \vdash s : S}{\Gamma \vdash \mathsf{inl}\, s : S + T} \qquad \frac{\Gamma \vdash t : T}{\Gamma \vdash \mathsf{inr}\, t : S + T}$$

$$\frac{\Gamma \vdash r : S + T \qquad \Gamma, x{:}S \vdash s : U \qquad \Gamma, y{:}T \vdash t : U}{\Gamma \vdash \mathsf{case}\, r\, \mathsf{of}\, \mathsf{inl}\, x \Rightarrow s \mid \mathsf{inr}\, y \Rightarrow t : U}$$

- Rules for unit and empty type:

$$\frac{}{\Gamma \vdash \langle\rangle : 1} \qquad \frac{\Gamma \vdash r : 0}{\Gamma \vdash \mathsf{abort}\, r : U}$$

Properties of typing

- Scoping: If $\Gamma \vdash t : T$, then $\mathsf{FV}(t) \subseteq \mathsf{dom}(\Gamma)$.
- Inversion:
  - If $\Gamma \vdash \lambda x.t : U$ then $U = S \to T$ for some types S, T and $\Gamma, x{:}S \vdash t : T$.
  - If $\Gamma \vdash r\,s : T$ then there exists some type S such that $\Gamma \vdash r : S \to T$ and $\Gamma \vdash s : S$.
  - *Exercise: complete this list!*
  - *Exercise: prove impossibility of* $\Gamma \vdash \lambda x.(x\,x) : T$*!*
- Substitution: If $\Gamma, x{:}S \vdash t : T$ and $\Gamma \vdash s : S$ then $\Gamma \vdash t[s/x] : T$.

Computation

- Values of programs are computed by iterated application of these reductions:

$$
\begin{array}{lcl}
(\lambda x.t)s & \longrightarrow & t[s/x] \\
\\
\mathsf{fst}\,\langle s,\ t\rangle & \longrightarrow & s \\
\mathsf{snd}\,\langle s,\ t\rangle & \longrightarrow & t \\
\\
\mathsf{case}\,(\mathsf{inl}\ r)\ \mathsf{of}\ \mathsf{inl}\,x \Rightarrow s \mid \mathsf{inr}\,y \Rightarrow t & \longrightarrow & s[r/x] \\
\mathsf{case}\,(\mathsf{inr}\ r)\ \mathsf{of}\ \mathsf{inl}\,x \Rightarrow s \mid \mathsf{inr}\,y \Rightarrow t & \longrightarrow & t[r/y]
\end{array}
$$

- Reductions can be applied deep inside a term.
- Type preservation under reduction ("subject reduction"):

  If $\Gamma \vdash t : T$ *and* $t \longrightarrow t'$ *then* $\Gamma \vdash t' : T$.

Computation example

$$
\begin{array}{ll}
 & (\lambda p.\,\mathsf{fst}\,p)\,(\mathsf{case}\,\mathsf{inl}\,\langle\rangle\ \mathsf{of}\ \mathsf{inl}\,x \Rightarrow \langle x,\,x\rangle \mid \mathsf{inr}\,y \Rightarrow y) \\
\longrightarrow & (\lambda p.\,\mathsf{fst}\,p)\,(\langle x,\,x\rangle[\langle\rangle/x]) \\
= & (\lambda p.\,\mathsf{fst}\,p)\,\langle\langle\rangle,\,\langle\rangle\rangle \\
\longrightarrow & \mathsf{fst}\,\langle\langle\rangle,\,\langle\rangle\rangle \\
\longrightarrow & \langle\rangle
\end{array}
$$

Normal forms

- A term which does not reduce is in *normal form*.
- Grammar that rules out redexes and meaningless terms:

$$\begin{array}{lll} \mathsf{Nf} \ni v, w & ::= u \mid \lambda x.v \mid \langle\rangle \mid \langle v,\ w\rangle \mid \mathsf{inl}\ v \mid \mathsf{inr}\ v & \text{normal form} \\ \mathsf{Ne} \ni u & ::= x \mid u\ v \mid \mathsf{fst}\ u \mid \mathsf{snd}\ u \mid \mathsf{abort}\ u & \text{neutral normal form} \\ & \quad \mid \mathsf{case}\ u\ \mathsf{of}\ \mathsf{inl}\ x \Rightarrow v \mid \mathsf{inr}\ y \Rightarrow w & \end{array}$$

- Progress: If $\Gamma \vdash t : T$ then either $t \longrightarrow t'$ or $t \in \mathsf{Nf}$.
- Type soundness:

  If $\Gamma \vdash t : T$ then either t reduces infinitely or there is some $v \in \mathsf{Nf}$ such that $t \longrightarrow^* v$ and $\Gamma \vdash v : T$.

Normalization

- Our calculus has no recursion and is terminating.
- Weak normalization:

  If $\Gamma \vdash t : T$ *then there is some* $v \in \mathsf{Nf}$ *such that* $t \longrightarrow^* v$.

- Strong normalization:

  If $\Gamma \vdash t : T$ *then any reduction sequence*
  $t \longrightarrow t_1 \longrightarrow t_2 \longrightarrow \dots$ *starting with* t *is finite.*

- Proof of normalization is non-trivial!

The Curry-Howard Isomorphism

- H. Curry & W. A. Howard and N. de Bruijn
- Propositional formulæ correspond to simple types.

Proposition	Type
$A \Rightarrow B$	$S \to T$
$A \wedge B$	$S \times T$
$A \vee B$	$S + T$
$\top$	1
$\bot$	0

The Curry-Howard Isomorphism (ctd.)

- Inference rules correspond to terms.

Derivation	Term
$\Rightarrow\mathsf{I}_x(\mathcal{D})$	$\lambda x.t$
$\Rightarrow\mathsf{E}(\mathcal{D}_1, \mathcal{D}_2)$	$t_1\ t_2$
$\wedge\mathsf{I}(\mathcal{D}_1, \mathcal{D}_2)$	$\langle t_1,\ t_2\rangle$
$\wedge\mathsf{E}_1(\mathcal{D})$	$\mathsf{fst}\ t$
$\wedge\mathsf{E}_2(\mathcal{D})$	$\mathsf{snd}\ t$
$\vee\mathsf{I}_1(\mathcal{D})$	$\mathsf{inl}\ t$
$\vee\mathsf{I}_2(\mathcal{D})$	$\mathsf{inr}\ t$
$\vee\mathsf{E}_{x,y}(\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3)$	$\mathsf{case}\ t_1\ \mathsf{of}\ \mathsf{inl}\, x \Rightarrow t_2 \mid \mathsf{inr}\, y \Rightarrow t_3$
$\top\mathsf{I}$	$\langle\rangle$
$\bot\mathsf{E}(\mathcal{D})$	$\mathsf{abort}\ t$

- Proof reduction corresponds to computation.

Proof terms

- Judgement $\Gamma \vdash M : A$ "in context Γ, term M proves A".
- Rules for hypotheses and implication:

$$\frac{\Gamma(x) = A}{\Gamma \vdash x : A}\ \text{hyp}$$

$$\frac{\Gamma, x{:}A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ {\Rightarrow}\text{I} \qquad \frac{\Gamma \vdash M : A \Rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\,N : B}\ {\Rightarrow}\text{E}$$

- Rules for conjuction:

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N\rangle : A \wedge B}\ \wedge\text{I} \quad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \text{fst}\ M : A}\ \wedge\text{E}_1 \quad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \text{snd}\ M : B}\ \wedge\text{E}_2$$

Proof terms (ctd.)

- Rules for disjunction:

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \mathsf{inl}\, M : A \vee B} \vee\mathrm{I}_1 \qquad \frac{\Gamma \vdash M : B}{\Gamma \vdash \mathsf{inr}\, M : A \vee B} \vee\mathrm{I}_2$$

$$\frac{\Gamma \vdash M : A \vee B \qquad \Gamma, x{:}A \vdash N : C \qquad \Gamma, y{:}B \vdash O : C}{\Gamma \vdash \mathsf{case}\, M \,\mathsf{of}\, \mathsf{inl}\, x \Rightarrow N \mid \mathsf{inr}\, y \Rightarrow O : C} \vee\mathrm{E}$$

- Rules for truth and absurdity:

$$\frac{}{\Gamma \vdash \langle\rangle : \top} \top\mathrm{I} \qquad \frac{\Gamma \vdash M : \bot}{\Gamma \vdash \mathsf{abort}\, M : C} \bot\mathrm{E}$$

Normalization implies consistency

Theorem (Consistency of propositional logic)

There is no derivation of $\vdash \bot$ true.

Beweis.

Suppose $\mathcal{D} :: \vdash \bot\ \mathit{true}$. By Curry-Howard, there exists a closed term $\vdash t : 0$ of the empty type. By Normalization, there exists a closed normal form $v \in \mathsf{Nf}$ of the empty type $\vdash v : 0$. By Inversion, this can only be a neutral term $v \in \mathsf{Ne}$. Every neutral term has at least one free variable. This is a contradiction to the closedness of v. □

Normalization implies the disjunction property

Theorem (Disjunction property)

If $\vdash A \vee B$ *true* *then* $\vdash A$ *true* *or* $\vdash B$ *true*.

Beweis.

Again, by Curry-Howard, Normalization, and Inversion. □

Conclusion

- The Curry-Howard Isomorphism unifies programming and proving into one language (λ-calculus).
- Inspired Martin-Löf Type Theory and its implementations, e.g. Coq and Agda.
- Provides cross-fertilization between Logic and Programming Language Theory.

References

Alonzo Church.
A formulation of the simple theory of types.
JSL, 5(2):56–68, 1940.

Gerhard Gentzen.
Untersuchungen über das logische Schließen.
Mathematische Zeitschrift, 39:176–210, 405–431, 1935.

William A. Howard.
Ordinal analysis of terms of finite type.
JSL, 45(3):493–504, 1980.

Frank Pfenning.
Lecture notes on natural deduction.
Course CMU 15317: Constructive Logic, 2009.