Type Systems

Lecture 2: The Curry-Howard Correspondence

Neel Krishnaswami
University of Cambridge

Type Systems for Programming Languages

- Type systems lead a double life
- They are a fundamental concept from logic and proof theory
- They are an essential part of modern programming languages

Natural Deduction

- In the early part of the 20th century, mathematics grew very abstract
- As a result, simple numerical and geometric intuitions no longer seemed to be sufficient to justify mathematical proofs (eg, Cantor's proofs about infinite sets)
- Big idea of Frege, Russell, Hilbert: what if we treated theorems and proofs as ordinary mathematical objects?
- Dramatic successes and failures, but the formal systems they introduced were unnatural – proofs didn't look like human proofs
- In 1933 (at age 23!) Gerhard Gentzen invented natural deduction
- "Natural" because the proof style is natural (with a little squinting)

Natural Deduction: Propositional Logic

What are propositions?

- $\top$ is a proposition
- $P \wedge Q$ is a proposition, if P and Q are propositions
- $\bot$ is a proposition
- $P \vee Q$ is a proposition, if P and Q are propositions
- $P \supset Q$ is a proposition, if P and Q are propositions

These are the formulas of propositional logic (i.e., no quantifiers of the form "for all x, $P(x)$" or "there exists x, $P(x)$").

Judgements

- Some claims follow (e.g. $P \land Q \supset Q \land P$).
- Some claims don't. (e.g., $\top \supset \bot$)
- We judge which propositions hold, and which don't with judgements
- In particular, "P true" means we judge P to be true.
- How do we justify judgements? With inference rules!

Truth and Conjunction

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

$$\frac{P \text{ true} \qquad Q \text{ true}}{P \land Q \text{ true}} \land I$$

$$\frac{P \land Q \text{ true}}{P \text{ true}} \land E_1 \qquad \frac{P \land Q \text{ true}}{Q \text{ true}} \land E_2$$

Implication

- To prove $P \supset Q$ in math, we assume P and prove Q
- Therefore, our notion of judgement needs to keep track of assumptions as well!
- So we introduce $\Psi \vdash P$ true, where Ψ is a list of assumptions
- Read: "Under assumptions Ψ, we judge P true"

$$\frac{P \in \Psi}{\Psi \vdash P \text{ true}}\ \textsc{Hyp} \qquad \frac{\Psi, P \vdash Q \text{ true}}{\Psi \vdash P \supset Q \text{ true}}\ {\supset}\mathrm{I}$$

$$\frac{\Psi \vdash P \supset Q \text{ true} \qquad \Psi \vdash P \text{ true}}{\Psi \vdash Q \text{ true}}\ {\supset}\mathrm{E}$$

$$\frac{\Psi \vdash P \text{ true}}{\Psi \vdash P \vee Q \text{ true}} \vee I_1 \qquad \frac{\Psi \vdash Q \text{ true}}{\Psi \vdash P \vee Q \text{ true}} \vee I_2$$

$$\frac{\Psi \vdash P \vee Q \text{ true} \qquad \Psi, P \vdash R \text{ true} \qquad \Psi, Q \vdash R \text{ true}}{\Psi \vdash R \text{ true}} \vee E$$

(no intro for $\bot$)

$$\frac{\Psi \vdash \bot \text{ true}}{\Psi \vdash R \text{ true}} \bot E$$

Example

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(P \vee Q) \supset R, P \vdash (P \vee Q) \supset R \text{ true}} \quad \dfrac{\dfrac{}{(P \vee Q) \supset R, P \vdash P \text{ true}}}{(P \vee Q) \supset R, P \vdash P \vee Q \text{ true}}}{(P \vee Q) \supset R, P \vdash R \text{ true}}}{(P \vee Q) \supset R \vdash P \supset R \text{ true}} \quad \dots}{(P \vee Q) \supset R \vdash (P \supset R) \wedge (Q \supset R) \text{ true}}}{\cdot \vdash ((P \vee Q) \supset R) \supset ((P \supset R) \wedge (Q \supset R)) \text{ true}}
$$

The Typed Lambda Calculus

$$
\begin{array}{llcl}
\text{Types} & X & ::= & 1 \mid X \times Y \mid 0 \mid X + Y \mid X \to Y \\
\text{Terms} & e & ::= & x \mid \langle\rangle \mid \langle e, e\rangle \mid \mathsf{fst}\, e \mid \mathsf{snd}\, e \\
 & & \mid & \mathsf{abort} \mid \mathsf{L}\, e \mid \mathsf{R}\, e \mid \mathsf{case}(e, \mathsf{L}\, x \to e', \mathsf{R}\, y \to e'') \\
 & & \mid & \lambda x : X.\, e \mid e\, e' \\
\text{Contexts} & \Gamma & ::= & \cdot \mid \Gamma, x : X
\end{array}
$$

A typing judgement is of the form $\Gamma \vdash e : X$.

Units and Pairs

$$\frac{}{\Gamma \vdash \langle\rangle : 1} \; 1\mathsf{I}$$

$$\frac{\Gamma \vdash e : X \qquad \Gamma \vdash e' : Y}{\Gamma \vdash \langle e, e' \rangle : X \times Y} \; \times\mathsf{I}$$

$$\frac{\Gamma \vdash e : X \times Y}{\Gamma \vdash \mathsf{fst}\, e : X} \; \times\mathsf{E}_1 \qquad\qquad \frac{\Gamma \vdash e : X \times Y}{\Gamma \vdash \mathsf{snd}\, e : Y} \; \times\mathsf{E}_2$$

Functions and Variables

$$\frac{x : X \in \Gamma}{\Gamma \vdash x : X}\ \textsc{Hyp} \qquad \frac{\Gamma, x : X \vdash e : Y}{\Gamma \vdash \lambda x : X.\, e : X \to Y}\ {\to}\mathrm{I}$$

$$\frac{\Gamma \vdash e : X \to Y \qquad \Gamma \vdash e' : X}{\Gamma \vdash e\, e' : Y}\ {\to}\mathrm{E}$$

Sums and the Empty Type

$$\frac{\Gamma \vdash e : X}{\Gamma \vdash \mathsf{L}\, e : X + Y}\ {+}\mathsf{I}_1 \qquad \frac{\Gamma \vdash e : Y}{\Gamma \vdash \mathsf{R}\, e : X + Y}\ {+}\mathsf{I}_2$$

$$\frac{\Gamma \vdash e : X + Y \qquad \Gamma, x : X \vdash e' : Z \qquad \Gamma, y : Y \vdash e'' : Z}{\Gamma \vdash \mathsf{case}(e, \mathsf{L}\, x \to e', \mathsf{R}\, y \to e'') : Z}\ {+}\mathsf{E}$$

$$\text{(no intro for 0)} \qquad \frac{\Gamma \vdash e : 0}{\Gamma \vdash \mathsf{abort}\, e : Z}\ 0\mathsf{E}$$

Example

$$\lambda f : (X + Y) \to Z.\, \langle \lambda x : X.\, f\,(\mathsf{L}\, x), \lambda y : Y.\, f\,(\mathsf{R}\, y) \rangle$$
$$:$$
$$((X + Y) \to Z) \to (X \to Z) \times (Y \to Z)$$

You may notice a similarity here…!

The Curry-Howard Correspondence, Part 1

Logic	Programming
Formulas	Types
Proofs	Programs
Truth	Unit
Falsehood	Empty type
Conjunction	Pairing/Records
Disjunction	Tagged Union
Implication	Functions

Something missing: language semantics?

Operational Semantics of the Typed Lambda Calculus

$$\text{Values} \quad v \quad ::= \quad \langle\rangle \mid \langle v, v' \rangle \mid \lambda x : A.\, e \mid \mathsf{L}\, v \mid \mathsf{R}\, v$$

The transition relation is $e \leadsto e'$, pronounced "e steps to e'".

Operational Semantics: Units and Pairs

(no rules for unit)

$$\frac{e_1 \leadsto e_1'}{\langle e_1, e_2 \rangle \leadsto \langle e_1', e_2 \rangle} \qquad \frac{e_2 \leadsto e_2'}{\langle v_1, e_2 \rangle \leadsto \langle v_1, e_2' \rangle}$$

$$\frac{}{\mathsf{fst}\,\langle v_1, v_2 \rangle \leadsto v_1} \qquad \frac{}{\mathsf{snd}\,\langle v_1, v_2 \rangle \leadsto v_2}$$

$$\frac{e \leadsto e'}{\mathsf{fst}\, e \leadsto \mathsf{fst}\, e'} \qquad \frac{e \leadsto e'}{\mathsf{snd}\, e \leadsto \mathsf{snd}\, e'}$$

Operational Semantics: Void and Sums

$$\frac{e \leadsto e'}{\mathsf{abort}\, e \leadsto \mathsf{abort}\, e'}$$

$$\frac{e \leadsto e'}{\mathsf{L}\, e \leadsto \mathsf{L}\, e'} \qquad \frac{e \leadsto e'}{\mathsf{R}\, e \leadsto \mathsf{R}\, e'}$$

$$\frac{e \leadsto e'}{\mathsf{case}(e, \mathsf{L}\, x \to e_1, \mathsf{R}\, y \to e_2) \leadsto \mathsf{case}(e', \mathsf{L}\, x \to e_1, \mathsf{R}\, y \to e_2)}$$

$$\frac{}{\mathsf{case}(\mathsf{L}\, v, \mathsf{L}\, x \to e_1, \mathsf{R}\, y \to e_2) \leadsto [v/x]e_1}$$

$$\frac{}{\mathsf{case}(\mathsf{R}\, v, \mathsf{L}\, x \to e_1, \mathsf{R}\, y \to e_2) \leadsto [v/y]e_2}$$

Operational Semantics: Functions

$$\frac{e_1 \rightsquigarrow e_1'}{e_1\, e_2 \rightsquigarrow e_1'\, e_2} \qquad \frac{e_2 \rightsquigarrow e_2'}{v_1\, e_2 \rightsquigarrow v_1\, e_2'}$$

$$\frac{}{(\lambda x : X.\, e)\, v \rightsquigarrow [v/x]e}$$

Five Easy Lemmas

1. (Weakening) If $\Gamma, \Gamma' \vdash e : X$ then $\Gamma, z : Z, \Gamma' \vdash e : X$.
2. (Exchange) If $\Gamma, y : Y, z : Z, \Gamma' \vdash e : X$ then $\Gamma, z : Z, y : Y, \Gamma' \vdash e : X$.
3. (Substitution) If $\Gamma \vdash e : X$ and $\Gamma, x : X \vdash e' : Y$ then $\Gamma \vdash [e/x]e' : Y$.
4. (Progress) If $\cdot \vdash e : X$ then e is a value, or $e \leadsto e'$.
5. (Preservation) If $\cdot \vdash e : X$ and $e \leadsto e'$, then $\cdot \vdash e' : X$.

Proof technique similar to previous lecture. But what does it mean, logically?

Two Kinds of Reduction Step

Congruence Rules	Reduction Rules
$\dfrac{e_1 \leadsto e_1'}{\langle e_1, e_2 \rangle \leadsto \langle e_1', e_2 \rangle}$	$\dfrac{}{\text{fst}\,\langle v_1, v_2 \rangle \leadsto v_1}$
$\dfrac{e_2 \leadsto e_2'}{v_1\, e_2 \leadsto v_1\, e_2'}$	$\dfrac{}{(\lambda x : X.\, e)\, v \leadsto [v/x]e}$

- Congruence rules recursively act on a subterm
    - Controls evaluation order
- Reduction rules actually transform a term
    - Actually evaluates!

A Closer Look at Reduction

Let's look at the function reduction case:

$$(\lambda x : X.\, e)\, v \rightsquigarrow [v/x]e$$

$$\dfrac{\dfrac{\boxed{x : X \vdash e : Y}}{\cdot \vdash \lambda x : X.\, e : X \to Y}\ \to\!\mathrm{I} \qquad \boxed{\cdot \vdash v : X}}{\cdot \vdash (\lambda x : X.\, e)\, v : Y}\ \to\!\mathrm{E}$$

- Reducible term = intro immediately followed by an elim
- Evaluation = removal of this detour

All Reductions Remove Detours

$$\frac{}{\mathsf{fst}\,\langle v_1, v_2\rangle \leadsto v_1} \qquad \frac{}{\mathsf{snd}\,\langle v_1, v_2\rangle \leadsto v_2}$$

$$\frac{}{\mathsf{case}(\mathsf{L}\,v, \mathsf{L}\,x \to e_1, \mathsf{R}\,y \to e_2) \leadsto [v/x]e_1}$$

$$\frac{}{\mathsf{case}(\mathsf{R}\,v, \mathsf{L}\,x \to e_1, \mathsf{R}\,y \to e_2) \leadsto [v/y]e_2}$$

$$\frac{}{(\lambda x : X.\, e)\, v \leadsto [v/x]e}$$

Every reduction is of an introduction followed by an eliminator!

Values as Normal Forms

$$\text{Values} \quad v \quad ::= \quad \langle\rangle \;\mid\; \langle v, v' \rangle \;\mid\; \lambda x : A.\, e \;\mid\; \mathsf{L}\, v \;\mid\; \mathsf{R}\, v$$

- Note that values are introduction forms
- Note that values are not reducible expressions
- So programs evaluate towards a normal form
- Choice of which normal form to look at it determined by evaluation order

The Curry-Howard Correspondence, Continued

Logic	Programming
Formulas	Types
Proofs	Programs
Truth	Unit
Falsehood	Empty type
Conjunction	Pairing/Records
Disjunction	Tagged Union
Implication	Functions
Normal form	**Value**
Proof normalization	**Evaluation**
Normalization strategy	**Evaluation order**

The Curry-Howard Correspondence is Not an Isomorphism

The logical derivation:

$$\dfrac{\dfrac{}{P, P \vdash P \text{ true}} \qquad \dfrac{}{P, P \vdash P \text{ true}}}{P, P \vdash P \wedge P \text{ true}}$$

has 4 type-theoretic versions:

$$\dfrac{\vdots}{x : X, y : X \vdash \langle x, x \rangle : X \times X} \qquad \dfrac{\vdots}{x : X, y : X \vdash \langle y, y \rangle : X \times X}$$

$$\dfrac{\vdots}{x : X, y : X \vdash \langle x, y \rangle : X \times X} \qquad \dfrac{\vdots}{x : X, y : X \vdash \langle y, x \rangle : X \times X}$$

Exercises

For the 1, $\to$ fragment of the typed lambda calculus, prove type safety.

1. Prove weakening.
2. Prove exchange.
3. Prove substitution.
4. Prove progress.
5. Prove type preservation.