Type Systems

Lecture 10: Classical Logic and Continuation-Passing Style

Neel Krishnaswami
University of Cambridge

Proof (and Refutation) Terms

$$
\begin{array}{llll}
\text{Propositions} & A & ::= & \top \mid A \wedge B \mid \bot \mid A \vee B \mid \neg A \\
\text{True contexts} & \Gamma & ::= & \cdot \mid \Gamma, x : A \\
\text{False contexts} & \Delta & ::= & \cdot \mid \Delta, u : A \\
\text{Values} & e & ::= & \langle\rangle \mid \langle e, e' \rangle \mid \mathsf{L}\, e \mid \mathsf{R}\, e \mid \mathsf{not}(k) \\
 & & \mid & \mu u : A.\, c \\
\text{Continuations} & k & ::= & [] \mid [k, k'] \mid \mathsf{fst}\, k \mid \mathsf{snd}\, k \mid \mathsf{not}(e) \\
 & & \mid & \mu x : A.\, c \\
\text{Contradictions} & c & ::= & \langle e \mid_A k \rangle
\end{array}
$$

Expressions — Proof Terms

(No rule for $\bot$)

$$\frac{}{\Gamma; \Delta \vdash \langle\rangle : \top \text{ true}}\ \top\mathrm{P}$$

$$\frac{\Gamma; \Delta \vdash e : A \text{ true} \qquad \Gamma; \Delta \vdash e' : B \text{ true}}{\Gamma; \Delta \vdash \langle e, e' \rangle : A \wedge B \text{ true}}\ \wedge\mathrm{P}$$

$$\frac{\Gamma; \Delta \vdash e : A \text{ true}}{\Gamma; \Delta \vdash \mathrm{L}\, e : A \vee B \text{ true}}\ \vee\mathrm{P}_1 \qquad \frac{\Gamma; \Delta \vdash e : B \text{ true}}{\Gamma; \Delta \vdash \mathrm{R}\, e : A \vee B \text{ true}}\ \vee\mathrm{P}_2$$

$$\frac{x : A \in \Gamma}{\Gamma; \Delta \vdash x : A \text{ true}}\ \textsc{Hyp} \qquad \frac{\Gamma; \Delta \vdash k : A \text{ false}}{\Gamma; \Delta \vdash \mathrm{not}(k) : \neg A \text{ true}}\ \neg\mathrm{P}$$

Continuations — Refutation Terms

(No rule for $\top$)

$$\frac{}{\Gamma; \Delta \vdash [] : \bot \text{ false}} \;\bot R$$

$$\frac{\Gamma; \Delta \vdash k : A \text{ false} \qquad \Gamma; \Delta \vdash k' : B \text{ false}}{\Gamma; \Delta \vdash [k, k'] : A \vee B \text{ false}} \;\vee R$$

$$\frac{\Gamma; \Delta \vdash k : A \text{ false}}{\Gamma; \Delta \vdash \text{fst}\, k : A \wedge B \text{ false}} \;\wedge R_1 \qquad \frac{\Gamma; \Delta \vdash k : B \text{ false}}{\Gamma; \Delta \vdash \text{snd}\, k : A \wedge B \text{ false}} \;\wedge R_2$$

$$\frac{x : A \in \Delta}{\Gamma; \Delta \vdash x : A \text{ false}} \;\text{Hyp} \qquad \frac{\Gamma; \Delta \vdash e : A \text{ true}}{\Gamma; \Delta \vdash \text{not}(e) : \neg A \text{ false}} \;\neg R$$

Contradictions

$$\frac{\Gamma; \Delta \vdash e : A \text{ true} \qquad \Gamma; \Delta \vdash k : A \text{ false}}{\Gamma; \Delta \vdash \langle e \mid_A k \rangle \text{ contr}} \textsc{Contr}$$

$$\frac{\Gamma; \Delta, u : A \vdash c \text{ contr}}{\Gamma; \Delta \vdash \mu u : A.\, c : A \text{ true}} \qquad \frac{\Gamma, x : A; \Delta \vdash c \text{ contr}}{\Gamma; \Delta \vdash \mu x : A.\, c : A \text{ false}}$$

Operational Semantics

$$\begin{array}{lcl}
\langle \langle e_1, e_2 \rangle \mid_{A \wedge B} \mathsf{fst}\, k \rangle & \mapsto & \langle e_1 \mid_A k \rangle \\
\langle \langle e_1, e_2 \rangle \mid_{A \wedge B} \mathsf{snd}\, k \rangle & \mapsto & \langle e_2 \mid_B k \rangle \\
\langle \mathsf{L}\, e \mid_{A \vee B} [k_1, k_2] \rangle & \mapsto & \langle e \mid_A k_1 \rangle \\
\langle \mathsf{R}\, e \mid_{A \vee B} [k_1, k_2] \rangle & \mapsto & \langle e \mid_B k_2 \rangle \\
\langle \mathsf{not}(k) \mid_{\neg A} \mathsf{not}(e) \rangle & \mapsto & \langle e \mid_A k \rangle
\end{array}$$

$$\begin{array}{lcl}
\langle \mu u : A.\, c \mid_A k \rangle & \mapsto & [k/u]c \\
\langle e \mid_A \mu x : A.\, c \rangle & \mapsto & [e/x]c
\end{array}$$

Type Safety?

Preservation If $\cdot; \cdot \vdash c$ contr and $c \leadsto c'$ then $\cdot; \cdot \vdash c'$ contr.

Proof By *case analysis* on evaluation derivations!

(We don't even need induction!)

Type Preservation

$\langle\langle e_1, e_2\rangle \mid_{A\wedge B} \mathsf{fst}\, k\rangle \rightsquigarrow \langle e_1 \mid_A k\rangle$ — Assumption

$$\dfrac{\overbrace{\Gamma; \Delta \vdash \langle e_1, e_2\rangle : A \wedge B \text{ true}}^{(1)} \qquad \overbrace{\Gamma; \Delta \vdash \mathsf{fst}\, k : A \wedge B \text{ false}}^{(2)}}{\Gamma; \Delta \vdash \langle\langle e_1, e_2\rangle \mid_{A\wedge B} \mathsf{fst}\, k\rangle \text{ contr}}$$

Assumption

$$\dfrac{\overbrace{\Gamma; \Delta \vdash e_1 : A \text{ true}}^{(3)} \qquad \Gamma; \Delta \vdash e_2 : B \text{ true}}{\Gamma; \Delta \vdash \langle e_1, e_2\rangle : A \wedge B \text{ true}} \wedge\mathrm{P}$$

Analysis of (1)

$$\dfrac{\overbrace{\Gamma; \Delta \vdash k : A \text{ false}}^{(4)}}{\Gamma; \Delta \vdash \mathsf{fst}\, k : A \wedge B \text{ false}} \wedge\mathrm{R}_1$$

Analysis of (2)

$\cdot; \cdot \vdash \langle e_1 \mid_A k\rangle \text{ contr}$ — By rule on (3), (4)

Embedding Classical Logic into Intuitionistic Logic

- Intuitionistic logic has a clean computational reading
- Classical logic *almost* has a clean computational reading
- Q: Is there any way to equip classical logic with computational meaning?
- A: Embed classical logic *into* intuitionistic logic

But Isn't Classical Logic an Extension of Constructive Logic?

- Normally, we think of classical logic as "constructive logic plus double-negation elimination"
- Surprisingly, classical logic is **also** a subset of intuitionistic logic!
- How can this work?

Double Negation Elimination

- The definition of negation is $\neg A \triangleq A \to \bot$
- We cannot prove there are any functions $((A \to 0) \to 0) \to A$

Proof sketch

- We can model each type as a set.
- Model 0 as the empty set.
- Model $A \to B$ as the set of functions from A to B.
- A mathematical function from A to B is a functional relation.
- If A has any elements at all, then $A \to 0$ has *no* elements.
- Then $(A \to 0) \to 0$ has no elements if A has any elements (and one otherwise).

In general, $\neg\neg X \to X$ is not derivable constructively. However, the following *is* derivable:

Lemma For all X, there is a function tne : $(\neg\neg\neg X) \to \neg X$

$$\dfrac{\dots \vdash k : \neg\neg\neg X \qquad \dfrac{\dfrac{\dots \vdash q : X \to p \qquad \dots \vdash x : X}{k : \neg\neg\neg X, x : X, q : \neg X \vdash q\,x : p}}{k : \neg\neg\neg X, x : X \vdash \lambda q.\, q\,x : \neg\neg X}}{\dfrac{k : \neg\neg\neg X, x : X \vdash k\,(\lambda q.\, q\,x) : p}{\dfrac{k : \neg\neg\neg X \vdash \lambda x.\, k\,(\lambda q.\, q\,a) : \neg X}{\cdot \vdash \underbrace{\lambda k.\, \lambda a.\, k\,(\lambda q.\, q\,a)}_{\text{tne}} : (\neg\neg\neg X) \to \neg X}}}$$

Quasi-negation

- As a mathematical function space, $A \to 0$ has at most one element.
- From a programming perspective, this is terrible!
- Define "quasi-negation" $\sim X \triangleq X \to p$, where p is a fixed *arbitrary* type.

Triple-Negation Elimination for Quasi-Negation

Triple-negation elimination still holds for quasi-negations:

Lemma For all X, there is a function tne : $(\sim\sim\sim X) \to \sim X$

$$
\dfrac{\dots \vdash k : \sim\sim\sim X \qquad \dfrac{\dfrac{\dots \vdash q : X \to p \qquad \dots \vdash x : X}{k : \sim\sim\sim X, x : X, q : \sim X \vdash q\,x : p}}{k : \sim\sim\sim X, x : X \vdash \lambda q.\, q\,x : \sim\sim X}}{\dfrac{k : \sim\sim\sim X, x : X \vdash k\,(\lambda q.\, q\,x) : p}{\dfrac{k : \sim\sim\sim X \vdash \lambda x.\, k\,(\lambda q.\, q\,a) : \sim X}{\cdot \vdash \underbrace{\lambda k.\, \lambda a.\, k\,(\lambda q.\, q\,a)}_{\text{tne}} : (\sim\sim\sim X) \to \sim X}}}
$$

The Recipe for Embedding Classical Logic into Intuitionistic Logic

1. We define a translation function A°
2. It takes a classical type A and produces an intuitionistic type X.
3. A and X should be equivalent classically, but not necessarily constructively.
4. Then, we define a translation function from classical terms into intuitionistic terms.

Many Different Embeddings

- Many different translations of classical logic were discovered many times
  - Gerhard Gentzen and Kurt Gödel
  - Andrey Kolmogorov
  - Valery Glivenko
  - Sigekatu Kuroda
- The key property is to show that $\sim\sim A^\circ \to A^\circ$ holds.

The Kolmogorov Translation

Now, we can define another translation on types as follows:

$$
\begin{array}{lcl}
\neg A^\bullet & = & \sim\sim\sim A^\bullet \\
A \supset B^\bullet & = & \sim\sim(A^\bullet \to B^\bullet) \\
\top^\bullet & = & \sim\sim 1 \\
(A \wedge B)^\bullet & = & \sim\sim(A^\bullet \times B^\bullet) \\
\bot^\bullet & = & \sim\sim\bot \\
(A \vee B)^\bullet & = & \sim\sim(A^\bullet + B^\bullet)
\end{array}
$$

- Uniformly stick a double-negation in front of each connective.
- Deriving $\sim\sim A^\bullet \to A^\bullet$ is particularly easy:
  - The tne term will always work!

The Embedding Theorem

Theorem Classical terms embed into intutionistic terms:

1. If $\Gamma; \Delta \vdash e : A$ true then $\Gamma^\bullet, \sim\Delta^\bullet \vdash e^\bullet : A^\bullet$.
2. If $\Gamma; \Delta \vdash k : A$ false then $\Gamma^\bullet, \sim\Delta^\bullet \vdash k^\bullet : \sim A^\bullet$.
3. If $\Gamma; \Delta \vdash c$ contr then $\Gamma^\bullet, \sim\Delta \vdash c^\bullet : p$.

Proof By induction on derivations – for a suitable choice of translation function.

Implementing Classical Logic Axiomatically

- The proof theory of classical logic is elegant
- It is also very awkward to use:
  - Binding only arises from proof by contradiction
  - Difficult to write nested computations
  - Continuations/stacks are always explicit
- Functional languages make the stack implicit
- Can we make the continuations implicit?

The Typed Lambda Calculus with Continuations

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

Units and Pairs

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

$$\frac{\Gamma \vdash e : X \qquad \Gamma \vdash e' : Y}{\Gamma \vdash \left\langle e, e' \right\rangle : X \times Y}\ \times\mathrm{I}$$

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

Functions and Variables

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

Sums and the Empty Type

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

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

(no intro for 0)

$$\frac{\Gamma \vdash e : 0}{\Gamma \vdash \mathsf{abort}\, e : Z}\; 0\mathsf{E}$$

Continuation Typing

$$\frac{\Gamma, u : \neg X \vdash e : X}{\Gamma \vdash \text{letcont}\, u : \neg X.\ e : X}\ \textsc{Cont} \qquad \frac{\Gamma \vdash e : \neg X \qquad \Gamma \vdash e' : X}{\Gamma \vdash \text{throw}_Y(e, e') : Y}\ \textsc{Throw}$$

Examples

Double-negation elimination:

$$\begin{array}{rcl} \mathsf{dne}_X & : & \neg\neg X \to X \\ \mathsf{dne}_X & \triangleq & \lambda k : \neg\neg X.\ \mathsf{letcont}\, u : \neg X.\ \mathsf{throw}(k, u) \end{array}$$

The Excluded Middle:

$$\begin{array}{l} t : X \vee \neg X \\ t \triangleq \mathsf{letcont}\, u : \neg(X \vee \neg X). \\ \qquad \mathsf{throw}(u, \mathsf{R}\,(\mathsf{letcont}\, q : \neg\neg X. \\ \qquad\qquad\qquad \mathsf{throw}(u, \mathsf{L}\,(\mathsf{dne}_X\, q))) \end{array}$$

Continuation-Passing Style (CPS) Translation

Type translation (almost the Kolmogorov translation):

$$
\begin{array}{lcl}
\neg X^\circ & = & \sim X^\circ \\
X \to Y^\circ & = & (X^\bullet \to Y^\bullet) \\
1^\circ & = & 1 \\
(X \times Y)^\circ & = & (X^\bullet \times Y^\bullet) \\
0^\circ & = & 0 \\
(X + Y)^\circ & = & (X^\bullet + Y^\bullet) \\
\\
X^\bullet & = & \sim\sim X^\circ
\end{array}
$$

Translating contexts:

$$
\begin{array}{lcl}
(\cdot)^\bullet & = & \cdot \\
(\Gamma, x : A)^\bullet & = & \Gamma^\bullet, x : A^\bullet
\end{array}
$$

The CPS Translation Theorem

Theorem If $\Gamma \vdash e : X$ then $\Gamma^\bullet \vdash e^\bullet : X^\bullet$.

Proof: By induction on derivations – we “just” need to define $e^\bullet$.

$$
\begin{aligned}
x^{\bullet} &= \lambda k : {\sim}X^{\circ}.\, x\, k \\
\langle\rangle^{\bullet} &= \lambda k : {\sim}1.\, k\ \langle\rangle \\
\langle e_1, e_2\rangle^{\bullet} &= \lambda k : {\sim}(X \times Y)^{\circ}.\, e_1^{\bullet}\, (\lambda x : X^{\circ}.\, e_2{}^{\bullet}\, (\lambda y : Y^{\circ}.\, k\, (x, y))) \\
(\mathsf{fst}\, e)^{\bullet} &= \lambda k : {\sim}X^{\circ}.\, e^{\bullet}\, (\lambda p : (X \times Y)^{\circ}.\, k\, (\mathsf{fst}\, p)) \\
(\mathsf{snd}\, e)^{\bullet} &= \lambda k : {\sim}Y^{\circ}.\, e^{\bullet}\, (\lambda p : (X \times Y)^{\circ}.\, k\, (\mathsf{snd}\, p)) \\
(\mathsf{L}\, e)^{\bullet} &= \lambda k : {\sim}(X + Y)^{\circ}.\, e^{\bullet}\, (\lambda x : X^{\circ}.\, k\, (\mathsf{L}\, x)) \\
(\mathsf{R}\, e)^{\bullet} &= \lambda k : {\sim}(X + Y)^{\circ}.\, e^{\bullet}\, (\lambda y : Y^{\circ}.\, k\, (\mathsf{R}\, y)) \\
\mathsf{case}(e, \mathsf{L}\, x \to e_1, \mathsf{R}\, y \to e_2)^{\bullet} &= \lambda k : {\sim}Z^{\circ}.\, e^{\bullet}\, (\lambda v : (X + Y)^{\circ}.\, \mathsf{case}(v, \\
&\qquad\qquad \mathsf{L}\, (x : X^{\bullet}) \to e_1^{\bullet}\, k \\
&\qquad\qquad \mathsf{R}\, (y : Y^{\bullet}) \to e_2^{\bullet}\, k) \\
(\lambda x : X.\, e)^{\bullet} &= \lambda k : {\sim}(A \to B)^{\circ}.\, k\, (\lambda x : X^{\bullet}.\, e^{\bullet}) \\
(e_1\, e_2)^{\bullet} &= \lambda k : {\sim}Y^{\circ}.\, e_1^{\bullet}\, (\lambda f : (X \to Y)^{\circ}.\, f\, e_2^{\bullet}\, k))
\end{aligned}
$$

The CPS Translation for Continuations

$$
\begin{aligned}
(\mathsf{letcont}\, u : \neg X.\ e)^{\bullet} &= \lambda k : {\sim}X^{\circ}.\ [\mathsf{dni}(k)/u](e^{\bullet}) \\
\mathsf{throw}(e_1, e_2)^{\bullet} &= \lambda k : {\sim}Y^{\circ}.\ e_1^{\bullet}\ e_2^{\bullet} \\
\\
\mathsf{dni} : X \to {\sim}{\sim}X &= \lambda x : X.\ \lambda k : {\sim}X.\ k\,x
\end{aligned}
$$

- The rest of the CPS translation is bookkeeping to enable these two clauses to work!

Case $\Gamma \vdash \text{throw}(e_1, e_2) : Y$

$\Gamma \vdash \text{throw}(e_1, e_2) : X$	Assumption
$\Gamma \vdash e_1 : \neg X$	Subderivation
$\Gamma \vdash e_2 : X$	Subderivation
$\Gamma^\bullet \vdash e_1^\bullet : \neg X^\bullet$	Induction
$\Gamma^\bullet \vdash e_2^\bullet : X^\bullet$	Induction
$\neg X^\bullet = \sim X^\bullet$	Definition
$Y^\bullet = \sim\sim Y^\circ$	Definition
$\Gamma^\bullet \vdash e_1^\bullet\, e_2^\bullet : p$	By app rule
$\Gamma^\bullet, k : \sim Y \vdash e_1^\bullet\, e_2^\bullet : p$	By weakening
$\Gamma^\bullet \vdash \lambda k : \sim Y^\circ .\, e_1^\bullet\, e_2^\bullet : \sim\sim Y^\circ$	By lambda
$\Gamma^\bullet \vdash \lambda k : \sim Y^\circ .\, e_1^\bullet\, e_2^\bullet : Y^\bullet$	By above

Questions

1. Give the embedding (ie, the e° and k° translations) of classical into intuitionistic logic for the Gödel-Gentzen translation (see next slide).
2. Using the intuitionistic calculus extended with continuations, give a typed term proving *Peirce's law*:

$$((X \to Y) \to X) \to X$$

The Gödel-Gentzen Translation

Now, we can define a translation on types as follows:

$$
\begin{array}{lcl}
\neg A^{\circ} & = & {\sim}A^{\circ} \\
\top^{\circ} & = & 1 \\
(A \wedge B)^{\circ} & = & A^{\circ} \times B^{\circ} \\
\bot^{\circ} & = & p \\
(A \vee B)^{\circ} & = & {\sim}({\sim}A^{\circ} \times {\sim}B^{\circ})
\end{array}
$$

- This uses a different de Morgan duality for disjunction