Lecture 15: The Curry-Howard Correspondance

Polyvios Pratikakis

Computer Science Department, University of Crete

Type Systems and Programming Languages
Curry-Howard Correspondance

- Another use of $\lambda$-calculus
- Roughly:
  - Types correspond to theorems
  - Programs correspond to proofs
  - Typed languages correspond to logics
  - A typechecker is a proof verifier
Classical propositional logic

- Formulas of the form

\[ \phi ::= p \mid \bot \mid \phi \lor \phi \mid \phi \land \phi \mid \phi \rightarrow \phi \]

- Where \( p \in \mathcal{P} \) is an atomic proposition, e.g. “Socrates is a man”

- Convenient abbreviations:
  - \( \neg \phi \) means \( \phi \rightarrow \bot \)
  - \( \phi \leftrightarrow \phi' \) means \((\phi \rightarrow \phi') \land (\phi' \rightarrow \phi)\)
Semantics of classical logic

- Interpretation $m : \mathcal{P} \rightarrow \{\text{true}, \text{false}\}$

\[ [p]^m = m(p) \]
\[ [\bot]^m = \text{false} \]
\[ [\phi \land \phi']^m = [\phi]^m \land [\phi']^m \]
\[ [\phi \lor \phi']^m = [\phi]^m \lor [\phi']^m \]
\[ [\phi \rightarrow \phi']^m = \neg [\phi]^m \lor [\phi']^m \]

- Where $\land, \lor, \neg$ are the standard boolean operations on $\{\text{true}, \text{false}\}$
Terminology

- A formula $\phi$ is **valid** if $\lbrack \phi \rbrack^m = \text{true}$ for all $m$
- A formula $\phi$ is **unsatisfiable** if $\lbrack \phi \rbrack^m = \text{false}$ for all $m$

**Law of excluded middle:**
- Formula $\phi \lor \neg \phi$ is valid for any $\phi$

- A **proof system** attempts to determine the validity of a formula
Proof theory for classical logic

- Proves judgements of the form $\Gamma \vdash \phi$:
  - For any interpretation, under assumption $\Gamma$, $\phi$ is true
- Syntactic deduction rules that produce “proof trees” of $\Gamma \vdash \phi$:
  - Natural deduction
- Problem: classical proofs only address truth value, not constructive
- Example: “There are two irrational numbers $x$ and $y$, such that $x^y$ is rational”
  - Proof does not include much information
Intuitionistic logic

- Get rid of the law of excluded middle
- Notion of “truth” is not the same
  - A proposition is true, if we can construct a proof
  - Cannot assume predefined truth values without constructed proofs (no “either true or false”)
- Judgements are not expression of “truth”, they are constructions
  - \( \vdash \phi \) means “there is a proof for \( \phi \)”
  - \( \vdash \phi \rightarrow \bot \) means “there is a refutation for \( \phi \)”, not “there is no proof”
  - \( \vdash (\phi \rightarrow \bot) \rightarrow \bot \) only means the absence of a refutation for \( \phi \), does not imply \( \phi \) as in classical logic
Proofs in intuitionistic logic

\[
\frac{\Gamma, \phi \vdash \phi}{\Gamma \vdash \phi} \\
\frac{\Gamma \vdash \phi \quad \Gamma \vdash \psi}{\Gamma \vdash \phi \land \psi} \\
\frac{\Gamma \vdash \phi}{\Gamma \vdash \phi \lor \psi} \\
\frac{\Gamma \vdash \psi}{\Gamma \vdash \phi \lor \psi} \\
\frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \rightarrow \psi} \quad \frac{\Gamma \vdash \perp}{\Gamma \vdash \phi} \\
\frac{\Gamma \vdash \phi \land \psi}{\Gamma \vdash \phi} \\
\frac{\Gamma \vdash \phi \land \psi}{\Gamma \vdash \psi} \\
\frac{\Gamma \vdash \phi}{\Gamma \vdash \phi} \\
\frac{\Gamma, \psi \vdash \rho}{\Gamma \vdash \phi \lor \psi} \\
\frac{\Gamma, \phi \vdash \rho}{\Gamma \vdash \rho} \\
\frac{\Gamma \vdash \phi \rightarrow \psi \quad \Gamma \vdash \phi}{\Gamma \vdash \psi}
\]

Does that resemble anything?
Curry-Howard correspondence

- We can mechanically translate formulas $\phi$ into type $\tau$ for every $\phi$ and the reverse
  - E.g. replace $\land$ with $\times$, $\lor$ with $+$, ...
- If $\Gamma \vdash e : \tau$ in simply-typed lambda calculus, and $\tau$ translates to $\phi$, then $\text{range}(\Gamma) \vdash \phi$ in intuitionistic logic
- If $\Gamma \vdash \phi$ in intuitionistic logic, and $\phi$ translates to $\tau$, then there exists $e$ and $\Gamma'$ such that $\text{range}(\Gamma') = \Gamma$ and $\Gamma' \vdash e : \tau$
- Proof by induction on the derivation $\Gamma \vdash \phi$
  - Can be simplified by fixing the logic and type languages to match
Consequences

- Lambda terms encode proof trees
- Evaluation of lambda terms is proof simplification
- Automated proving by trying to construct a lambda term with the wanted type
- Verifying a proof is typechecking
  - Increased trust in complicated proofs when machine-verifiable
- Proof-carrying code
- Certifying compilers