Lecture 12: Simply typed λ-calculus

Summary.
(1) Introducing types in the λ-calculus.

(2) The Curry-Howard correspondence (with intuitionistic propositional logic).

Luís Soares Barbosa,
UNIV. MINHO *(Informatics Dep.)* & INL *(Quantum Software Engineering Group)*

Types.

The simply-typed λ-calculus reintroduces the notions of domain and codomain in the definition of a function. If the former are sets (or any other type of semantic entities), types are *names* for them, i.e. purely syntactic entities.

Given a set Θ of basic types, the set of simple types is given by

$$A, B \ \ni\ \theta \mid A \longrightarrow B \mid A \times B \mid \mathbf{1}$$

where $\theta \in \Theta$.

By convention $\times$ binds stronger than $\longrightarrow$, and the latter associates to the right. Typed λ-terms are the inhabitants of these types, defined by

$$t, t' \ \ni\ x \mid t\, t' \mid \lambda x^A \,.\, t \mid \langle t, t' \rangle \mid \pi_1\, t \mid \pi_2\, t \mid *$$

where $x \in X$, for X a set of variables, as before.

The notions of free and bound variables, as well as of α-conversion and α-equivalence, remain as in the untyped case.

Terms are subjected to a typing discipline given by the following set of rules. Note that rules relate *typing judgements*

$$x_1 : A_1,\ x_2 : A_2,\ \cdots\ x_n : A_n\ \vdash\ t : A$$

reading as t *is a well-typed term of type* A *under the assumption that each* x_i *is also a well-typed term of type* A_i. Note that $\mathcal{FV}(t) \subseteq \{x_1, x_2, \cdots x_n\}$: to compute the type of M one needs to make assumptions about the types of its free variables.

The following set of typing rules avoid the construction of meanignless terms.

$$\frac{}{\Gamma, x : A\ \vdash\ x : A}\,(var) \qquad \frac{}{\Gamma\ \vdash\ * : \mathbf{1}}\,(one)$$

$$\frac{\Gamma, x : A\ \vdash\ t : B}{\Gamma\ \vdash\ \lambda x^A . t : A \longrightarrow B}\,(abs) \qquad \frac{\Gamma\ \vdash\ t : A \longrightarrow B \quad \Gamma\ \vdash\ u : A}{\Gamma\ \vdash\ t\,u : B}\,(app)$$

$$\frac{\Gamma\ \vdash\ u : A \quad \Gamma\ \vdash\ v : B}{\Gamma\ \vdash\ \langle u, v \rangle : A \times B}\,(split) \qquad \frac{\Gamma\ \vdash\ t : A \times B}{\Gamma\ \vdash\ \pi_1\, t : A}\,(p1) \qquad \frac{\Gamma\ \vdash\ t : A \times B}{\Gamma\ \vdash\ \pi_2\, t : B}\,(p2)$$

Note that there is a rule (exactly one) for each sort of term. Typing derivations are built in a bottom-up way in which the choice of the rule to apply is always unique. Indeed, the only choice to make is the one that fixes the types to assign to variables.

Exercise 1

Write a typing derivation for the following judgements

1. $\vdash\ \lambda x^{A \longrightarrow A}.\lambda y^{A}.x\,(x\,y)\ :\ (A \longrightarrow A) \longrightarrow A \longrightarrow A$
2. $\vdash\ \lambda x^{A \times B}.\langle \pi_2\,x, \pi_1\,x\rangle\ :\ (A \times B) \longrightarrow (B \times A)$

Exercise 2

Not all terms can be typed, i.e. assigned types to all free and bound variables such that the corresponding type judgement is derivable. Discuss why such is the case for terms $\pi_2(\lambda x^{A}.\,t)$ and $\lambda x^{A}.\,x\,x$.

Exercise 3

Match, if possible, inhabitants (i.e. closed λ-terms) for the following types:

1. $(A \times B) \longrightarrow A$
2. $A \longrightarrow (A \times B)$
3. $A \longrightarrow A \longrightarrow A$
4. $A \longrightarrow B \longrightarrow (A \times B)$
5. $(A \longrightarrow B) \longrightarrow (B \longrightarrow C) \longrightarrow (A \longrightarrow C)$
6. $((A \longrightarrow A) \longrightarrow B) \longrightarrow B$
7. $(A \longrightarrow C) \longrightarrow C$

from

1. $\lambda x^{A}.\,\lambda y^{A}.\,x$
2. $\lambda x^{(A \longrightarrow A) \longrightarrow B}.\,x\,(\lambda y^{A}.y)$
3. $\lambda x^{A}.\,\lambda x^{B}.\,\langle x, y\rangle$

4. $\lambda x^{A \longrightarrow B}. \lambda y^{B \longrightarrow C}. \lambda z^{A}. y\,(x\,z)$
5. $\lambda x^{A \times B}. \pi_1 x$
6. $\lambda x^{A}. \lambda y^{A}. y$

Hint. Rewrite $\times$ as conjunction and $\longrightarrow$ as implication and identify the propositional tautologies. Notice that types (2) and (7) do not correspond, under this conversion, to tautologies. Indeed, the types which have inhabitants are the ones that are valid once 'seen? as logic formulas.

The Curry-Howard correspondence.

The last exercise illustrates a deep connection between types (of λ-terms) and (intuitionistic propositional) formulas. Recall propositional intuitionistic logic previously studied:

$$\frac{}{\Gamma, \phi \vdash \phi}\,(Ax) \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash \phi}\,(\bot out) \qquad \frac{}{\Gamma \vdash \top}\,(\top in)$$

$$\frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \Rightarrow \psi}\,(\Rightarrow in) \qquad \frac{\Gamma \vdash \phi \Rightarrow \psi \quad \Gamma \vdash \phi}{\Gamma \vdash \psi}\,(\Rightarrow out)$$

$$\frac{\Gamma \vdash \phi \quad \Gamma \vdash \psi}{\Gamma \vdash \phi \wedge \psi}\,(\wedge in) \qquad \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \phi}\,(\wedge_1 out) \qquad \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \psi}\,(\wedge_2 out)$$

$$\frac{\Gamma \vdash \phi}{\Gamma \vdash \phi \vee \psi}\,(\vee in_1) \qquad \frac{\Gamma \vdash \psi}{\Gamma \vdash \phi \vee \psi}\,(\vee in_2) \qquad \frac{\Gamma, \phi \vdash \rho \quad \Gamma, \psi \vdash \rho \quad \Gamma \vdash \phi \vee \psi}{\Gamma \vdash \rho}\,(\vee out)$$

Notice that there are no introduction rule for $\bot$ (which somehow mirrors the absence of an elimination rule for $\top$).

$\boxed{\text{Formulas-as-Types and Proofs-as-Programs}}$

As illustrated in the exercise above, there is a bijective correspondence between *types* (of λ-terms) and *formulas* (of intuitionistic proposicional logic), once the set of basic types is identified with the set of atomic formulas. If this correspondence is straightforward, one can also characterise a bijection between *proofs*, i.e. derivations in the logic, and *terms* in the simply-typed λ-calculus.

Example:
To prove $A \wedge B \Rightarrow A$, one assumes hypothesis $A \wedge B$ and concludes A precisely by the first part of the hypothesis. Terms can be constructed to witness the proof:

$$\underbrace{\text{Assume } A \wedge B}_{\lambda x^{A \times B}}\ \underbrace{\text{then by the first part of the assumption,}}_{\pi_1 x}\ \underbrace{A \text{ holds.}}_{\lambda x^{A \times B}. \pi_1 x}$$

Thus, term $\lambda x^{A\times B}.\,\pi_1 x$ corresponds to the proof of proposition $A \wedge B \Rightarrow A$. In general, the correspondence is given by the following rules, in which we restricted ourselves to the $\wedge, \Rightarrow$-fragment of the logic:

1. If the derivation is by (Ax), the term is $t = x$, because $\Gamma, x : A \vdash x : A$ is a valid typing judgement by rule (var).
2. If the derivation is by $(\top in)$, the term is $t = *$, because $\vdash * : \mathbf{1}$ is a valid typing judgement by rule (one).
3. If the derivation is by $(\Rightarrow in)$, the term is $t = \lambda x^A.u$, where u is the term associated to the sub-derivations. By induction hypothesis $\Gamma, x : A \vdash u : B$, which entails, by rule (abs), $\Gamma \vdash \lambda x^A.u : A \longrightarrow B$.
4. If the derivation is by $(\Rightarrow out)$, the term is $t = u\,v$, where u and v are the terms associated to the sub-derivations. By induction hypothesis $\Gamma \vdash u : A \longrightarrow B$ and $\Gamma \vdash v$, which entails, by rule (app), $\Gamma \vdash u\,v : B$,
5. If the derivation is by $(\wedge in)$, the term is $t = \langle u, v\rangle$, where u and v are the terms associated to the sub-derivations. By induction hypothesis $\Gamma \vdash u : A$, $\Gamma \vdash v : B$, which entails, by rule $(split)$, $\Gamma \vdash \langle u, v\rangle : A \times B$.
6. If the derivation is by $(\wedge_1 out)$, the term is $t = \pi_1 u$, where u is the term associated to the sub-derivation. By induction hypothesis $\Gamma \vdash u : A \times B$, which entails, by rule (p_1), $\Gamma \vdash \pi_1 u : A$.

Exercise 4

Make the proof in the reverse direction: given a well-typed λ-term t associated to a typing judgement $\Gamma \vdash t : A$, construct a derivation of A from assumptions Γ.

Simply-typed λ dynamics.

As expected, β and η reductions have to be extended to the new syntax[1]. Thus,

$$\begin{array}{rclrl}
(\lambda x^A.\,u)\,v & \longrightarrow & u[x := v] & & (\beta_{\longrightarrow}) \\
\pi_1\langle u, v\rangle & \longrightarrow & u & & (\beta_{\pi_1}) \\
\pi_2\langle u, v\rangle & \longrightarrow & v & & (\beta_{\pi_2}) \\
\langle \pi_1 u, \pi_2 u\rangle & \longrightarrow & u & & (\eta_{\times}) \\
\lambda x^A.\,u\,x & \longrightarrow & u & \Leftarrow x \notin \mathcal{FV}(u) & (\eta_{\longrightarrow}) \\
u & \longrightarrow & * & \Leftarrow u : \mathbf{1} & (\eta_{\mathbf{1}})
\end{array}$$

[1]To be precise, reduction need to be defined between typing judgements rather than terms, as some rules, namely $(\eta_{\mathbf{1}})$, depend on the terms involved.

Subject reduction theorem. Well-typed terms reduce to well-typed terms of the same type. Formally, if $\Gamma \vdash u : A$ and $u \longrightarrow_{\beta,\eta} v$ then $\Gamma \vdash v : A$.

Unicity of normal forms. The Church-Rosser theorem does not hold for η-reduction. Actually, for a variable $x : A \times \mathbf{1}$, the term $\langle \pi_1 x, \pi_2 x\rangle$ reduces to x by $(\eta_\times)$ and to $\langle \pi_1 x, *\rangle$ by $(\eta_\mathbf{1})$, and both are normal forms.

This can be addressed omitting type $\mathbf{1}$ (and the term $*$) from the language, or, alternatively, keeping the language but forgetting about η-reduction. Actually, the computation dynamics is entirely located in the β-reduction, η-reduction being essentially used to simplify the result. In particular,

- η-reduction always reduces the size of a term;
- and it does not interfere with β-reduction, i.e. if $u \longrightarrow_\eta v$ and v has a β-redex, the original term u also has a corresponding redex.

Exercise 5

A main consequence of the correspondence *proofs-as-programs*, i.e. *proofs-as-λ-terms*, is that β,η-*reduction* corresponds to *proof simplification*. For example, reduction

$$\pi_1 \langle u, v\rangle \quad\longrightarrow\quad u$$

corresponds to the following proof simplification:

$$\dfrac{\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}(\wedge in)}{\Gamma \vdash A}(\wedge_1 out) \qquad\longrightarrow\qquad \Gamma \vdash A$$

Once you have checked that the given terms correspond to the given proofs, explain the proof simplifications corresponding to the following reductions:

1. $\lambda x^A . u\, x \quad\longrightarrow\quad u \quad$ where $x \notin \mathcal{FV}(u)$
2. $\langle \pi_1 u, \pi_2 u\rangle \quad\longrightarrow\quad u$
3. $(\lambda x^A . u)\, v \quad\longrightarrow\quad u[x := v]$

Exercise 6

This exercise proposes to extend the correspondence between logic and computation discussed above to the whole propositional intuitionistic logic. On the 'computation' side, one needs to extend

- the set of simple types with *sum* (or *disjoint*) and *empty*,

$$A, B \ni \cdots \mid A + B \mid \emptyset$$

- the set of typed λ-terms

$$t, t', u, u' \ \ni \ \cdots \mid \mathsf{either}\ t\ (x^A \Rightarrow u \mid y^B \Rightarrow u') \mid \iota_1\, t \mid \iota_2\, t \mid ?_A\, t$$

- and the typing rules

$$\frac{\Gamma \vdash t : \emptyset}{\Gamma \vdash ?_A\, t : A}\,(zero)$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \iota_1\, t : A + B}\,(i1) \qquad \frac{\Gamma \vdash t : B}{\Gamma \vdash \iota_2\, t : A + B}\,(i2)$$

$$\frac{\Gamma \vdash t : A + B \quad \Gamma, x : A \vdash u : C \quad \Gamma, y : B \vdash v : C}{\Gamma \vdash \mathsf{either}\ t\ (x^A \Rightarrow u \mid y^B \Rightarrow v) : C}\,(either)$$

Explain the *rationale* underlying these extensions and build the correspondence between the new terms and proofs in the extended logic fragment.