Lecture Notes on Subject Reduction

15-814: Types and Programming Languages
Frank Pfenning

Lecture 5
September 15, 2020

1 Introduction

In the last lecture we laid the groundwork for a *representation theorem* on Booleans, which will prove in this lecture. This provides a clear relationship between normal forms and one particular type and is an exemplar of many similar theorems characterizing the normal forms of given types.

In the second part of the this lecture we establish a relationship between *computation* and types, complementing the relation between *normal forms* and types. The essence of this is that if $\Gamma \vdash e : \tau$ and $e \longrightarrow e'$ then $\Gamma \vdash e' : \tau$.

The third part (which we already proved) is that every well-typed expression either reduces or is a normal form so there is no "loophole" in the type system.

Together, these three parts form the basic pillars for interpreting the meaning of types in programming languages, studied here in the setting of the simply-typed λ-calculus which we can think of as a proto-programming-language, and which we will find embedded in richer and more practical languages.

2 A Representation Theorem for Booleans

Theorem 1 (Representation of Booleans, v4) *If* $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ *and* e *normal then* $e =$ *true* $= \lambda x.\, \lambda y.\, x$ *or* $e =$ *false* $= \lambda x.\, \lambda y.\, y$.

We postpone the proof to first show an important lemma about *neutral terms* which will be used in the proof.

Lemma 2 (Neutrality) *If $x_1 : \alpha_1, \ldots, x_n : \alpha_n \vdash e : \tau$ and e neutral then $e = x_i$ and $\tau = \alpha_i$ for some $1 \leq i \leq n$.*

Proof: The intuition behind this theorem is that a neutral term e has the form $((x\, e_1) \ldots e_k)$ but there is no variable x that has a function type so $k = 0$ and $e = x$. But the only variables x in the context are $x_i : \alpha_i$.

There are essentially three different forms of induction we could apply here (abbreviating $\Gamma_0 = x_1 : \alpha_1, \ldots, x_n : \alpha_n$)

1. Over the structure of the expression e
2. Over the derivation of $\Gamma_0 \vdash e : \tau$
3. Over the derivation of e *neutral*

Generally, when we have additional information about an expression such as e, we rarely perform an induction over the structure of e, but we prefer to directly exploit the knowledge about e. Secondly (and also a heuristic), we can easily apply inversion to syntax-directed judgments such as typing, and less directly so for others. Therefore, we prefer rule induction over judgments *other* than typing.

More formally, we proceed by rule induction on e *neutral*. There are just two cases.

Case:

$$\frac{}{x \textit{ neutral}}\ \mathsf{neut/var}$$

where $e = x$. Then we reason

$x_1 : \alpha_1, \ldots, x_n : \alpha_n \vdash x : \tau$	Assumption
$x = x_i$ and $\tau = \alpha_i$ for some $1 \leq i \leq n$	By inversion

"Inversion" here refers to the fact that there is only one typing rule for variables, tp/var, and this rule requires x to be one of the variables in the context and τ to be the corresponding type.

Case:

$$\frac{e_1 \textit{ neutral} \quad e_2 \textit{ normal}}{e_1\, e_2 \textit{ neutral}}\ \mathsf{neut/app}$$

where $e = e_1\, e_2$. Then we reason

$\Gamma_0 \vdash e_1\, e_2 : \tau$	Assumption
$\Gamma_0 \vdash e_1 : \tau_2 \to \tau$ and $\Gamma_0 \vdash e_2 : \tau_2$ for some τ_2	By inversion
$e_1 = x_i$ and $\tau_2 \to \tau = \alpha_i$ for some $1 \leq i \leq n$	By ind. hyp.
Contradiction	Since $\tau_2 \to \tau = \alpha_i$ is impossible

Therefore, the second case is impossible, as we already noted informally at the outset. The appeal to the induction hypothesis relies on the derivations of e_1 *neutral* and $\Gamma_0 \vdash e_1 : \tau_2 \to \tau$ and is correct because e_1 *neutral* is a subderivation (in fact, the immediate premise) of the given derivation for $e = e_1\, e_2$.

□

Now we are ready to tackle the proof of the representation theorem for normal forms.

Proof: (of Theorem 1) Let's remind ourselves:

> *If* $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ *and* e *normal then* $e =$ *true* $= \lambda x.\, \lambda y.\, x$ *or* $e =$ *false* $= \lambda x.\, \lambda y.\, y$.

Again we have a choice: we could try induction over the structure of e (not a good idea), rule induction over the derivation of $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ (okay), or rule induction over e *normal* (even better). As it turns out, we can do a *proof by cases*, since the induction hypothesis is never needed! This is, of course, a special case of induction but we would like to be precise if a simpler proof principle suffices.

Case:

$$\frac{e \textit{ neutral}}{e \textit{ normal}}\ \mathsf{norm/neut}$$

We conclude that this case is impossible as follows:

$\cdot \vdash e : \alpha \to (\alpha \to \alpha)$	Assumption
e *neutral*	Premise in this case
Contradiction	By Lemma 2

Case:

$$\frac{e_1 \textit{ normal}}{\lambda x.\, e_1 \textit{ normal}}\ \mathsf{norm/lam}$$

where $e = \lambda x.\, e_1$. We continue:

$\cdot \vdash \lambda x.\, e_1 : \alpha \to (\alpha \to \alpha)$ Assumption
$x : \alpha \vdash e_1 : \alpha \to \alpha$ By inversion
Either e_1 *neutral* or $e_1 = \lambda x.\, e_2$ for some e_2 and e_2 *normal*
By inversion on e_1 *normal*

Here the appeal to inversion yields two cases, because the conclusion e_1 *normal* could be derived by two different rules (norm/neut or norm/lam).

Subcase: e_1 *neutral*. Again, this case is impossible by neutrality.

$x : \alpha \vdash e_1 : \alpha \to \alpha$ From above
e_1 *neutral* This case
Contradiction By Lemma 2

Subcase: $e_1 = \lambda y.\, e_2$ for some e_2 and e_2 *normal*. Then

$x : \alpha \vdash \lambda y.\, e_2 : \alpha \to \alpha$ From above with $e_1 = \lambda y.\, e_2$
$x : \alpha, y : \alpha \vdash e_2 : \alpha$ By inversion
e_2 *normal* This subcase
$e_2 = \lambda z.\, e_3$ for some e_3 *normal*
or e_2 *neutral* By inversion on e_2 *normal*

We now distinguish the reasoning in these two subcases.

Sub^{2}case: $e_2 = \lambda z.\, e_3$ for some e_3 with e_3 *normal*. Now it is this case that is impossible:

$x : \alpha, y : \alpha \vdash \lambda z.\, e_3 : \alpha$ From above with $e_2 = \lambda x\, e_3$
Contradiction By inversion
(no typing rule matches this conclusion)

Sub^{2}case: e_2 *neutral*. Then

$x : \alpha, y : \alpha \vdash e_2 : \alpha$ From above
e_2 *neutral* This case
$e_2 = x$ or $e_2 = y$ By neutrality (Lemma 2)
$e = \lambda x.\, e_1 = \lambda x.\, \lambda y.\, e_2 = \lambda x.\, \lambda y.\, x$
or $e = \lambda x.\, e_1 = \lambda x.\, \lambda y.\, e_2 = \lambda x.\, \lambda y.\, y$
By form of e, e_1, and e_2 in this case

□

3 Subject Reduction

Let's put the representation theorem into the bigger picture. We had previously conjectured:

Conjecture 3 (L3.4, Representation of Booleans, v2)
If $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ *then* $e =_\beta$ *true or* $e =_\beta$ *false.*

But we want to relate this to computation. Fortunately, by the Church-Rosser Theorem, $e =_\beta e'$ for a normal form e' if and only if $e \longrightarrow^* e'$ (where $\longrightarrow^*$ is the reflexive and transitive closure of single-step reduction we have been mostly working with). So we recast this one more time, relating typing to computation and representation.

Conjecture 4 (Computation of Booleans)
If $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ *then* $e \longrightarrow^*$ *true or* $e \longrightarrow^*$ *false.*

Since every well-typed expression has a normal form (which we did not prove), the missing link in our reasoning chain is that typing is preserved under reduction: if we start with an expression e of type τ and we reduce it all the way to a normal form e', then e' will still have type τ. For the special case where $\tau = \alpha \to (\alpha \to \alpha)$ which means that any expression e of type τ that has a normal form represents a Boolean.

Now we return to the main topic of this lecture, namely subject reduction. Recall our characterization of reduction:

$$\frac{e \longrightarrow e'}{\lambda x.\, e \longrightarrow \lambda x.\, e'}\ \mathsf{red/lam} \qquad \frac{e_1 \longrightarrow e_1'}{e_1\, e_2 \longrightarrow e_1'\, e_2}\ \mathsf{red/app_1} \qquad \frac{e_2 \longrightarrow e_2'}{e_1\, e_2 \longrightarrow e_1\, e_2'}\ \mathsf{red/app_2}$$

$$\frac{}{(\lambda x.\, e_1)\, e_2 \longrightarrow [e_2/x]e_1}\ \mathsf{beta}$$

And, for reference, here are the typing rules.

$$\frac{\Gamma, x_1 : \tau_1 \vdash e_2 : \tau_2}{\Gamma \vdash \lambda x_1.\, e_2 : \tau_1 \to \tau_2}\ \mathsf{lam} \qquad \frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau}\ \mathsf{var}$$

$$\frac{\Gamma \vdash e_1 : \tau_2 \to \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\, e_2 : \tau_1}\ \mathsf{app}$$

Theorem 5 (Subject Reduction)
If $\Gamma \vdash e : \tau$ *and* $e \longrightarrow e'$ *then* $\Gamma \vdash e' : \tau$.

Proof: In this theorem statement we are given derivations for two judgments: $\Gamma \vdash e : \tau$ and $e \longrightarrow e'$. Most likely, the proof will proceed by rule

induction on one of these and by inversion on the other. The typing judgment is syntax-directed and therefore amenable to reasoning by inversion, so we try rule induction over the reduction judgment.

By rule induction on the derivation of $e \longrightarrow e'$.

Case:

$$\frac{e_1 \longrightarrow e_1'}{\lambda x.\, e_1 \longrightarrow \lambda x.\, e_1'}\ \mathsf{red/lam}$$

where $e = \lambda x.\, e_1'$.

$\Gamma \vdash \lambda x.\, e_1 : \tau$	Assumption
$\Gamma, x : \tau_2 \vdash e_1 : \tau_1$ and $\tau = \tau_2 \to \tau_1$ for some τ_1 and τ_2	By inversion
$\Gamma, x : \tau_2 \vdash e_1' : \tau_1$	By induction hypothesis
$\Gamma \vdash \lambda x.\, e_1' : \tau_2 \to \tau_1$	By rule lam

Case:

$$\frac{e_1 \longrightarrow e_1'}{e_1\, e_2 \longrightarrow e_1'\, e_2}\ \mathsf{red/app_1}$$

where $e = e_1\, e_2$. We start again by restating what we know in this case and then apply inversion.

$\Gamma \vdash e_1\, e_2 : \tau$	Assumption
$\Gamma \vdash e_1 : \tau_2 \to \tau$ and $\Gamma \vdash e_2 : \tau_2$ for some τ_2	By inversion

At this point we have a type for e_1 and a reduction for e_1, so we can apply the induction hypothesis.

$\Gamma \vdash e_1' : \tau_2 \to \tau$	By ind.hyp.

Now we can just apply the typing rule for application. Intuitively, in the typing for $e_1\, e_2$ we have replaced e_1 by e_1', which is okay since e_1' has the type of e_1.

$\Gamma \vdash e_1'\, e_2 : \tau$	By rule lam

Case:

$$\frac{e_2 \longrightarrow e_2'}{e_1\, e_2 \longrightarrow e_1'\, e_2}\ \mathsf{red/app}_2$$

where $e = e_1\, e_2$. This proceeds completely analogous to the previous case.

Case:

$$\frac{}{(\lambda x.\, e_1)\, e_2 \longrightarrow [e_2/x]e_1}\ \beta$$

where $e = (\lambda x.\, e_1)\, e_2$. In this case we apply inversion twice, since the structure of e is two levels deep.

$\Gamma \vdash (\lambda x.\, e_1)\, e_2 : \tau$	Assumption
$\Gamma \vdash \lambda x.\, e_1 : \tau_2 \to \tau$	
and $\Gamma \vdash e_2 : \tau_2$ for some τ_2	By inversion
$\Gamma, x : \tau_2 \vdash e_1 : \tau$	By inversion

At this point we are truly stuck, because there is no obvious way to complete the proof.

To Show: $\Gamma \vdash [e_2/x]e_1 : \tau$

Fortunately, the gap that presents itself is exactly the content of the *substitution property*, stated below. The forward reference here is acceptable, since the proof of the substitution property does not depend on subject reduction.

$\Gamma \vdash [e_2/x]e_1 : \tau$	By the *substitution property* (Theorem 6)

□

Theorem 6 (Substitution Property)
If $\Gamma \vdash e : \tau$ *and* $\Gamma, x : \tau \vdash e' : \tau'$ *then* $\Gamma \vdash [e/x]e' : \tau'$

Proof sketch: By rule induction on the deduction of $\Gamma, x : \tau \vdash e' : \tau'$. Intuitively, in this deduction we can use $x : \tau$ only at the leaves, and there to conclude $x : \tau$. Now we replace this leaf with the given derivation of

$\Gamma \vdash e : \tau$ which concludes $e : \tau$. Luckily, $[e/x]x = e$, so this is the correct judgment.

There is only a small hiccup: when we introduce a different variable $x_1 : \tau_1$ into the context in the lam rule, the contexts of the two assumptions no longer match. But we can apply *weakening*, that is, adjoin the unused hypothesis $x_1 : \tau_1$ to every judgment in the deduction of $\Gamma \vdash e : \tau$. After that, we can apply the induction hypothesis. □

We recommend you write out the cases of the substitution property in the style of our other proofs, just to make sure you understand the details.

The substitution property is so critical that we may elevate it to an intrinsic property of the turnstile ($\vdash$). Whenever we write $\Gamma \vdash J$ for any judgment J we imply that a substitution property for the judgments in Γ must hold. This is an example of a *hypothetical* and *generic* judgment [ML83]. We may return to this point in a future lecture, especially if the property appears to be in jeopardy at some point. It is worth remembering that, while we may not want to prove an explicit substitution property, we still need to make sure that the judgments we define *are* hypothetical/generic judgments.

4 Taking Stock

Where do we stand at this point in our quest for a representation theorems for Booleans? We have the following:

Reduction and Normal Forms

(i) For all e, either $e \longrightarrow$ or e *normal*.

(ii) There is no e such that $e \longrightarrow$ and e *normal*

Representation of Booleans in Normal Form (L5.1)
If $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ and e *normal* then either $e = \mathit{true} = \lambda x.\, \lambda y.\, x$ or $e = \mathit{false} = \lambda x.\, \lambda y.\, y$.

Subject Reduction (L5.5)
If $\Gamma \vdash e : \tau$ and $e \longrightarrow e'$ we have $\Gamma \vdash e' : \tau$.

We did not prove normalization (also called *termination*) or confluence (also called the Church-Rosser property).

Normalization
If $\Gamma \vdash e : \tau$ then $e \longrightarrow^* e'$ for some e' with e' *normal*.

Confluence
If $e \longrightarrow^* e_1$ and $e \longrightarrow^* e_2$ then there exists an e' such that $e_1 \longrightarrow^* e'$ and $e_2 \longrightarrow e'$.

We could replay the whole development for the representation of natural numbers instead of Booleans, with some additional complications, but we will forego this in favor of tackling more realistic programming languages.

Exercises

Exercise 1 Define multi-step reduction $e \longrightarrow^* e'$ by the following rules:

$$\frac{}{e \longrightarrow^* e}\ \mathsf{red}^*/\mathsf{refl} \qquad \frac{e \longrightarrow e' \quad e' \longrightarrow^* e''}{e \longrightarrow^* e''}\ \mathsf{red}^*/\mathsf{step}$$

Prove by rule induction that if $\Gamma \vdash e : \tau$ and $e \longrightarrow^* e'$ then $\Gamma \vdash e' : \tau$

Exercise 2 Define a new single-step relation $e \mapsto e'$ which means that e reduces to e' by *leftmost-outermost reduction*, using a collection of inference rules. Recall that I claimed this strategy is *sound* (it only performs β-reductions) and *complete for normalization* (if e has a normal form, we can reach it by performing only leftmost-outermost reductions). Prove the following statements about your reduction judgment:

(i) If $e \mapsto e'$ then $e \longrightarrow e'$.

(ii) $\mapsto$ is *small-step deterministic*, that is, if $e \mapsto e_1$ and $e \mapsto e_2$ then $e_1 = e_2$.

You should interpret $=$ as α-equality, that is, the two terms differ only in the names of their bound variables (which we always take for granted). For each of the following statements, either indicate that they are true (without proof) or provide a counterexample.

(iii) For all e, either $e \mapsto e'$ for some e' or e *normal*.

(iv) There does not exist an e such that $e \mapsto e'$ for some e' and e *normal*.

(v) If $e \longrightarrow e'$ then $e \mapsto e'$.

(vi) $\longrightarrow$ is *small-step deterministic*.

(vii) $\longrightarrow$ is *big-step deterministic*, that is, if $e \longrightarrow^* e_1$ and $e \longrightarrow^* e_2$ where e_1 *normal* and e_2 *normal*, then $e_1 = e_2$.

(viii) For arbitrary e and *normal* e', $e \longrightarrow^* e'$ iff $e \mapsto^* e'$.

References

[ML83] Per Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. Notes for three lectures given in Siena, Italy. Published in *Nordic Journal of Philosophical Logic*, 1(1):11-60, 1996, April 1983.