Lecture Notes on
Subject Reduction and Normal Forms

15-814: Types and Programming Languages
Frank Pfenning

Lecture 4
September 13, 2018

1 Introduction

In the last lecture we proved some key aspect of a *representation theorem* for Booleans, namely that the closed normal forms type $\alpha \to (\alpha \to \alpha)$ are either *true* $= \lambda x.\, \lambda y.\, x$ or *false* $= \lambda x.\, \lambda y.\, y$. But is our characterization of normal forms correct? We would like to prove that any expression e is either a normal form (that is, satisfies e *nf* or can be reduced. We prove this theorem first. Then we show that typing is preserved under reduction, which means that if we start with an expression e of type τ and we reduce it all the way to a normal form e', the 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.

2 Reduction and Normal Form

Recall our characterization of reduction from Lecture 2, written here as a collection of inference rules.

$$\frac{e \longrightarrow e'}{\lambda x.\, e \longrightarrow \lambda x.\, e'}\ \mathsf{lm} \qquad \frac{e_1 \longrightarrow e_1'}{e_1\, e_2 \longrightarrow e_1'\, e_2}\ \mathsf{ap}_1 \qquad \frac{e_1 \longrightarrow e_2'}{e_1\, e_2 \longrightarrow e_1\, e_2'}\ \mathsf{ap}_2$$

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

And our characterization of normal forms:

$$\frac{e\ \mathit{nf}}{\lambda x.\, e\ \mathit{nf}}\ \mathsf{nf/lam} \qquad \frac{e\ \mathit{neutral}}{e\ \mathit{nf}}\ \mathsf{nf/ne}$$

$$\frac{e_1\ \mathit{neutral} \quad e_2\ \mathit{nf}}{e_1\ e_2\ \mathit{neutral}}\ \mathsf{ne/app} \qquad \frac{}{x\ \mathit{neutral}}\ \mathsf{ne/var}$$

The correctness of this characterization consists of two parts, of which we will prove one: (i) every term either reduces or is a normal form, and (ii) normal forms don't reduce.

Theorem 1 (Reduction and normal forms)
For every expression e, either $e \longrightarrow e'$ for some e', or e nf.

Proof: We are only given an expression e, so the proof is likely by induction on the structure of e. Let's try! We write $e \longrightarrow$ if there is some e' such that $e \longrightarrow e'$.

Case: $e = x$. Then

x *neutral*	By rule ne/var
x *nf*	By rule nf/ne

Case: $e = \lambda x.\, e_1$. Then

Either $e_1 \longrightarrow$ or e_1 *nf*	By ind.hyp. on e'
$e_1 \longrightarrow$	First subcase
$e = \lambda x.\, e_1 \longrightarrow$	By rule lm
e_1 *nf*	Second subcase
$e = \lambda x.\, e_1$ *nf*	By rule nf/lam

Case: $e = e_1\, e_2$. Then

Either $e_1 \longrightarrow$ or e_1 *nf*	By ind.hyp. on e_1
$e_1 \longrightarrow$	First subcase
$e_1\ e_2 \longrightarrow$	By rule ap_1
e_1 *nf*	Second subcase

Either $e_1 = \lambda x.\, e_1'$ or e_1 *neutral*	By inversion on e_1 *nf*
$e_1 = \lambda x.\, e_1'$	First sub^{2}case
$e = e_1\, e_2 = (\lambda x.\, e_1')\, e_2 \longrightarrow$	By rule β
e_1 *neutral*	Second sub^{2}case
Either $e_2 \longrightarrow$ or e_2 *nf*	By ind.hyp. on e_2
$e_2 \longrightarrow$	First sub^{3}case
$e = e_1\, e_2 \longrightarrow$	By rule ap_2
e_2 *nf*	Second sub^{3}case
$e = e_1\, e_2$ *neutral*	By rule $\mathsf{ne/app}$

□

The next in our quest for a representation theorem will be to show that types are preserved under reduction. We start with $e : \tau$ and want to know that if $e \longrightarrow^* e'$ where e' is a normal form, then $e' : \tau$. This is almost universally proven by showing that a single step of reduction preserves types, from which the above follows by a simple induction over the reduction sequence.

We begin by conjecturing a version of the theorem for closed expression of arbitrary type, because these are the expressions we are ultimately interested in.

Conjecture 2 (Subject reduction, v1)
If $\cdot \vdash e : \tau$ *and* $e \longrightarrow e'$ *then* $\cdot \vdash e' : \tau$.

Proof attempt: In this conjecture, we are given both an expression e and a reduction $e \longrightarrow e'$, so a priori there are three possible inductions: rule induction on $\cdot \vdash e : \tau$, rule induction on $e \longrightarrow e'$, and induction on the structure of e. Having done this kind of proof about a gazillion times, I know it should go by rule induction on $e \longrightarrow e'$.

Case:

$$\frac{e_1 \longrightarrow e_1'}{\lambda x.\, e_1 \longrightarrow \lambda x.\, e_1'}\ \mathsf{lm}$$

where $e = \lambda x.\, e_1'$. We gather knowledge, and then apply inversion because we cannot yet apply the induction hypothesis.

$\cdot \vdash \lambda x.\, e_1 : \tau$	Assumption
$x : \tau_2 \vdash e_1 : \tau_1$ and $\tau = \tau_2 \to \tau_1$ for some τ_1 and τ_2	By inversion

It looks like we are ready for an appeal to the induction hypothesis, but we are stuck because the context in the typing of e_1 the context is not empty! We realize have to generalize the theorem to allow arbitrary contexts Γ.

□

Theorem 3 (Subject reduction, v2)
If $\Gamma \vdash e : \tau$ *and* $e \longrightarrow e'$ *then* $\Gamma' \vdash e' : \tau$.

Proof: By rule induction on the deduction of $e \longrightarrow e'$.

Case:

$$\frac{e_1 \longrightarrow e_1'}{\lambda x.\, e_1 \longrightarrow \lambda x.\, e_1'}\ \mathsf{lm}$$

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:

$$\frac{e_1 \longrightarrow e_1'}{e_1\, e_2 \longrightarrow e_1'\, e_2}\ \mathsf{ap}_1$$

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 they e_1' has the type of e_1.

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

Case:

$$\frac{e_2 \longrightarrow e_2'}{e_1\, e_2 \longrightarrow e_1'\, e_2}\ \mathsf{ap}_2$$

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* (4)

□

Theorem 4 (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 $y : \tau''$ 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 $y : \tau''$ 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.

3 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
For any e, either $e \longrightarrow$ or e *nf* (Theorem L4.1)

Representation of Booleans in normal form
For any e with $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ and e *nf*, either $e = \mathit{true} = \lambda x.\, \lambda y.\, x$ or $e = \mathit{false} = \lambda x.\, \lambda y.\, y$. (Theorem L3.10(i))

Subject reduction
For any e with $\Gamma \vdash e : \tau$ and $e \longrightarrow e'$ we have $\Gamma \vdash e' : \tau$. (Theorem L4.3)

Subject reduction to normal form
For any e with $\Gamma \vdash e : \tau$ and $e \longrightarrow^* e'$ with e' *nf* we have $\Gamma \vdash e' : \tau$. (Corollary of subject reduction)

Missing at this point are the following theorems

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

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

In this context, *normalization* (sometimes called *termination*) shows that any closed expression of type $\alpha \to (\alpha \to \alpha)$ denotes a Boolean. *Confluence* (also known as the Church-Rosser property) show that this Boolean is unique.

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

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.