Lecture Notes on Representation Theorems

15-814: Types and Programming Languages
Frank Pfenning

Lecture 5
September 14, 2021

The subject reduction theorem from the last lecture established that if we start computation by reduction with an expression of type τ, then each expression in the chain of reductions will continue to have the same type. So if computation terminates in a normal form, we are still at type τ. In particular, if we start with a closed expression of type bool, then the normal form will also have type $\mathsf{bool} = \alpha \to (\alpha \to \alpha)$. But does this normal form really then represent either *true* $= \lambda x.\, \lambda y.\, x$ or *false* $= \lambda x.\, \lambda y.\, y$? Establishing this is the goal of this lecture.

Since we are in the setting of the typed λ-calculus, we can add another piece of information: reduction of every expression terminates. So every closed expression of type bool eventually reduces to either *true* or *false*. Furthermore, due to the Church-Rosser theorem (also knows as confluence), it must reduce to either one or the other, but cannot reduce to both guaranteeing the uniqueness of the meaning of the expression.

Unfortunately, when we got to a more complete language where we can freely use recursion, the termination property will fail so we don't investigate it in detail or try to prove it in this first part of the course. Also, we will contrive to make reduction deterministic so that the confluence property is rather immediate. So, for the moment, our focus is on the representation theorem for Booleans, as a stand-in for other representation theorems we could prove like Church numerals.

Before we proving the representation theorem, we need to make sure that our judgment e *normal* is suitably related to reduction.

1 Normal Forms and Reduction

The characterization of normal forms via inference rules is compact, but is it really the same as saying that an expression does not reduce? We would like to work as much as possible with positive characterizations, so we break this down into the following two properties

1. For all expressions e, either e reduces or e is normal.

2. For all expressions e, it is not that case that e reduces and e is normal.

The second property just states that the "either/or" in part 1 is an exclusive or. In mathematical language, saying "either/or" doesn't automatically imply that the properties are exclusive, so both should be stated. We will prove the first, and leave the second as Exercise 2.

To make the proof just a bit easier to write, we introduce a new judgment $e \longrightarrow$ expressing that e reduces, but we do not care what to. We obtain it by erasing the right-hand sides of all the reduction rules. It is then immediate (although formally done by induction) that $e \longrightarrow e'$ for some e' iff $e \longrightarrow$.

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

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

Theorem 1 (Reduction and normal forms, Part (i))
For every expression e, either $e \longrightarrow$ or e normal.

Proof: We are only given an expression e, so the proof is likely by induction on the structure of e. Such a proof has the following parts:

(i) We have to establish the property outright for $e = x$.

(ii) We have to establish the property for $e = \lambda x.\, e_1$, where the induction hypothesis is the property for e_1.

(iii) We have the establish the property for $e = e_1\, e_2$ where the induction hypotheses are the properties for e_1 and e_2.

If we can cover all three cases we know that the property must hold for all expressions. Let's try!

Case: $e = x$. Then

x *neutral*	By rule neut/var
x *normal*	By rule norm/neut

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

Either $e_1 \longrightarrow$ or e_1 *normal*	By ind.hyp. on e_1
$e_1 \longrightarrow$	First subcase
$e = \lambda x.\, e_1 \longrightarrow$	By rule rbl/lam
e_1 *normal*	Second subcase
$e = \lambda x.\, e_1$ *normal*	By rule norm/lam

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

Either $e_1 \longrightarrow$ or e_1 *normal*	By ind.hyp. on e_1
$e_1 \longrightarrow$	First subcase
$e_1\ e_2 \longrightarrow$	By rule rbl/app$_1$
e_1 *normal*	Second subcase
Either $e_1 = \lambda x.\, e_1'$ and e_1' *normal*	
or e_1 *neutral*	By inversion on e_1 *normal*
$e_1 = \lambda x.\, e_1'$	First sub^{2}case
$e = e_1\, e_2 = (\lambda x.\, e_1')\, e_2 \longrightarrow$	By rule rbl/beta
e_1 *neutral*	Second sub^{2}case
Either $e_2 \longrightarrow$ or e_2 *normal*	By ind.hyp. on e_2
$e_2 \longrightarrow$	First sub^{3}case
$e = e_1\, e_2 \longrightarrow$	By rule rbl/app$_2$
e_2 *normal*	Second sub^{3}case
$e = e_1\, e_2$ *neutral*	By rule neut/app

$\square$

This step in a proof is called *inversion* because we infer, at the metalevel at which we reason about our judgments, that the premise of a rule must hold if the conclusion does. This is only valid if we consider all the possible cases, of which there are two in this particular situation. Often, there is only one, and sometimes there is none (which means that the case were are in is actually impossible).

Now that we have characterized normal forms, we will be able to prove a representation theorem for Booleans in the next lecture.

2 A Representation Theorem for Booleans

Theorem 2 (Representation of Booleans) *If* $\cdot \vdash e : \alpha \to (\alpha \to \alpha)$ *and* e *normal then* $e = \mathit{true} = \lambda x.\, \lambda y.\, x$ *or* $e = \mathit{false} = \lambda x.\, \lambda y.\, y$.

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

Lemma 3 (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:

$$\frac{}{x \; \mathit{neutral}} \; \mathsf{neut/var}$$

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:

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

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 2) Let's remind ourselves:

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

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:

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

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 3

Case:

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

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 3

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 3)
$e = \lambda x.\, e_1 = \lambda x.\, \lambda y.\, e_2 = \lambda x.\, \lambda y.\, x$ 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 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

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 (Theorem L4.3)
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 Fill in the blanks in the following typing judgments so the resulting judgment holds, or indicate there is no way to do so. You do not need to justify your answer or supply a typing derivation, and the types do not need to be "most general" in any sense. Remember that the function type constructor associates to the right, so that $\tau \to \sigma \to \rho = \tau \to (\sigma \to \rho)$.

(i) $\boxed{} \vdash y\, x : \alpha$

(ii) $\boxed{} \vdash x\, x : \boxed{}$

(iii) $\cdot \vdash \boxed{} : (\alpha \to \alpha) \to \alpha$

(iv) $\cdot \vdash (\lambda z.\, z)\, (\lambda x.\, \lambda y.\, \lambda p.\, p\, x\, y) : \boxed{}$

(v)

$$\begin{array}{l} \cdot \vdash \lambda f.\, \lambda g.\, \lambda x.\, (f\, x)\, (g\, x) \\ \quad : (\alpha \to \boxed{}) \to (\alpha \to \boxed{}) \to (\alpha \to \boxed{}) \end{array}$$

Since this is the first time we (that is, you) are proving theorems about judgments defined by rules, we ask you to be very explicit, as we were in the lectures and lecture notes. In particular:

- Explicitly state the overall structure of your proof: whether it proceeds by rule induction, and, if so, on the derivation of which judgment, or by structural induction, or by inversion, or just directly. If you need to split out a lemma for your proof, state it clearly and prove it separately. If you need to generalize your induction hypothesis, clearly state the generalized form.
- Explicitly list all cases in an induction proof. If a case is impossible, prove that is is impossible. Often, that's just inversion, but sometimes it is more subtle.
- Explicitly note any appeals to the induction hypothesis.
- Any appeals to inversion should be noted as such, as well as the rules that could have inferred the judgment we already know. This could lead to zero cases (a contradiction—the judgment could not have been derived), one case (there is exactly one rule whose conclusion matches our knowledge), or multiple cases, in which case your proof now splits into multiple cases.
- We recommend that you follow the line-by-line style of presentation where each line is justified by a short phrase. This will help you to check your proof and us to read and verify it.

Exercise 2 Prove that there does not exist an expression e such that $e \longrightarrow$ and e *normal*. In other words, the alternatives stated in Theorem 1 are exclusive.

As a reminder, the way we prove that a proposition A is false is to assume A is true and derive a contradiction. For those who care about such things, this is a perfectly valid intuitionistic (constructive) reasoning principle, as opposed to an *indirect proof*. The rule of indirect proof (which we should avoid at all cost, since all proofs in this course should be constructive) says that we can prove A is true by assuming that A is false and then deriving a contradiction from that.