Lecturer: Dexter Kozen

1 Soundness from the Operational Perspective

We will now look at the soundness of the $\lambda^{\rightarrow}$ typing rules from the operational perspective. This means:

The typing rules are sound $\iff$ *no well-formed program gets stuck.*

For this language, *well-formed* and *well-typed* are synonymous. To be more precise, let us call e *irreducible* and write Irred(e) if there is no reduction possible on e. All values of $\lambda^{\rightarrow}$ are irreducible. If e is irreducible but is not a value, then e is said to be *stuck*. We wish to show

Theorem 1 (Operational Soundness) $\vdash e:\tau \ \wedge\ e \xrightarrow{*} e' \ \wedge\ \mathrm{Irred}(e') \ \Rightarrow\ e' \in \mathbf{Val} \wedge \vdash e':\tau.$

We will prove this in two steps using the following two lemmas:

Lemma 2 (Type Preservation) $\Gamma \vdash e:\tau \ \wedge\ e \rightarrow e' \ \Rightarrow\ \Gamma \vdash e':\tau.$

Lemma 3 (Progress) $\vdash e:\tau \ \wedge\ \mathrm{Irred}(e) \ \Rightarrow\ e \in \mathbf{Val}.$

The type preservation lemma says that as we evaluate a program, its type is preserved at each step. The progress lemma says that every program is either a value or can be stepped to another program (and by the preservation lemma, this will be of the same type).

Operational soundness follows easily from these two lemmas. Type preservation says every step preserves the type, so we use induction on the number of steps taken in $e \xrightarrow{*} e'$ to show that e' must have the same type as e. Then progress can be applied to e' to show that the evaluation isn't stuck there. We will now set out to prove these two lemmas.

2 Proof of the Type Preservation Lemma

Assuming that $\Gamma \vdash e:\tau$ and $e \rightarrow e'$, we wish to show that $\Gamma \vdash e':\tau$. We will do this by induction on the well-founded subterm relation.

If $e \rightarrow e'$, there are three cases corresponding to the three evaluation rules:

$$\frac{e_0 \rightarrow e_0'}{e_0\ e_1 \rightarrow e_0'\ e_1}\ (\mathrm{L}) \qquad \frac{e \rightarrow e'}{v\ e \rightarrow v\ e'}\ (\mathrm{R}) \qquad \frac{}{(\lambda x\!:\!\sigma.\, e)\ v \rightarrow e\{v/x\}}\ (\beta)$$

- Case (L): $e_0\ e_1 \rightarrow e_0'\ e_1$.

  Because we have a typing derivation for $e_0\ e_1$, we know that there are typing derivations for e_0 and e_1 too. We must have $\Gamma \vdash e_0:\sigma \rightarrow \tau$ and $\Gamma \vdash e_1:\sigma$ for some type σ. By the induction hypothesis, the reduction $e_0 \rightarrow e_0'$ also preserves type, so $\Gamma \vdash e_0':\sigma \rightarrow \tau$. Applying the typing rule for applications, we have that $\Gamma \vdash e_0'\ e_1:\tau$.

- Case (R): $v\ e \rightarrow v\ e'$.

  This case is symmetrical to case (L). In this case it is the right-hand subexpression making the step.

- Case (β): $(\lambda x\!:\!\sigma.\, e)\ v \rightarrow e\{v/x\}$.

  The typing derivation of $\Gamma \vdash (\lambda x\!:\!\sigma.\, e)\ v:\tau$ must look like this:

$$\frac{\dfrac{\Gamma,\, x\!:\!\sigma \vdash e:\tau}{\Gamma \vdash (\lambda x\!:\!\sigma.\, e):\sigma \rightarrow \tau} \quad \Gamma \vdash v:\sigma}{\Gamma \vdash (\lambda x\!:\!\sigma.\, e)\ v:\tau}$$

  We want to show that $\Gamma \vdash e\{v/x\}:\tau$ using the facts $\Gamma,\, x\!:\!\sigma \vdash e:\tau$ and $\vdash v:\sigma$. Our induction hypothesis doesn't help us here; we need to prove this separately. It follows as a special case of the substitution lemma below, which captures the type preservation of β-reduction.

3 The Substitution Lemma

Lemma 4 (Substitution Lemma) $\vdash v : \sigma \quad \Rightarrow \quad (\Gamma,\, x : \sigma \vdash e : \tau \quad \Leftrightarrow \quad \Gamma \vdash e\{v/x\} : \tau)$.

We will prove this by structural induction on e.

- Case 1: $x \notin FV(e)$.

  This case covers the base cases $e \in \{n, \mathsf{true}, \mathsf{false}, \mathsf{null}\}$ and $e = y \neq x$ and λ-abstractions $\lambda x : \rho.\, e$ that bind x. In this case the substitution has no effect and any binding of x in the type environment Γ is irrelevant, thus the lemma reduces to the trivial statement

  $$\vdash v : \sigma \quad \Rightarrow \quad (\Gamma \vdash e : \tau \quad \Leftrightarrow \quad \Gamma \vdash e : \tau).$$

- Case 2: $e = x$.

  In this case the lemma reduces to

  $$\vdash v : \sigma \quad \Rightarrow \quad (\Gamma,\, x : \sigma \vdash x : \tau \quad \Leftrightarrow \quad \Gamma \vdash v : \tau),$$

  since $x\{v/x\} = v$. Since v is closed, the type environment Γ is irrelevant, so the statement further reduces to

  $$\vdash v : \sigma \quad \Rightarrow \quad (x : \sigma \vdash x : \tau \quad \Leftrightarrow \; \vdash v : \tau).$$

  Since types are unique, both sides of the double implication say that $\sigma = \tau$, so again the lemma reduces to a tautology.

- Case 3: $e = e_0\ e_1$.

  Suppose $\vdash v : \sigma$.

  $$\begin{array}{rcll} \Gamma,\, x : \sigma \vdash e_0\ e_1 : \tau & \Leftrightarrow & \exists \sigma \;\; \Gamma,\, x : \sigma \vdash e_0 : \sigma \to \tau \;\wedge\; \Gamma,\, x : \sigma \vdash e_1 : \sigma & \text{typing rule for applications} \\ & \Leftrightarrow & \exists \sigma \;\; \Gamma \vdash e_0\{v/x\} : \sigma \to \tau \;\wedge\; \Gamma \vdash e_1\{v/x\} : \sigma & \text{induction hypothesis} \\ & \Leftrightarrow & \Gamma \vdash e_0\{v/x\}\ e_1\{v/x\} : \tau & \text{typing rule for applications} \\ & \Leftrightarrow & \Gamma \vdash (e_0\ e_1)\{v/x\} : \tau & \text{definition of substitution.} \end{array}$$

- Case 4: $e = \lambda y : \rho.\, e'$, where $y \neq x$ (the case $y = x$ was covered in Case 1).

  Suppose $\vdash v : \sigma$. The type of $\lambda y : \rho.\, e'$, if it exists, must be $\rho \to \tau$ for some τ. Similarly, the type of $(\lambda y : \rho.\, e')\{v/x\} = \lambda y : \rho.\, (e'\{v/x\})$, if it exists, must be $\rho \to \tau'$ for some τ'.

  $$\begin{array}{rcll} \Gamma,\, x : \sigma \vdash (\lambda y : \rho.\, e') : \rho \to \tau & \Leftrightarrow & \Gamma,\, x : \sigma,\, y : \rho \vdash e' : \tau & \text{typing rule for abstractions} \\ & \Leftrightarrow & \Gamma,\, y : \rho,\, x : \sigma \vdash e' : \tau & \text{exchange} \\ & \Leftrightarrow & \Gamma,\, y : \rho \vdash e'\{v/x\} : \tau & \text{induction hypothesis} \\ & \Leftrightarrow & \Gamma \vdash \lambda y : \rho.\, (e'\{v/x\}) : \rho \to \tau & \text{typing rule for abstractions} \\ & \Leftrightarrow & \Gamma \vdash (\lambda y : \rho.\, e')\{v/x\} : \rho \to \tau & \text{definition of substitution.} \end{array}$$

4 Proof of the Progress Lemma

To finish the proof of soundness, it remains to prove the progress lemma. Recall that this lemma states

$$\vdash e : \tau \;\wedge\; \mathrm{Irred}(e) \quad \Rightarrow \quad e \in \mathbf{Val},$$

or equivalently,

$$\vdash e : \tau \;\wedge\; e \notin \mathbf{Val} \quad \Rightarrow \quad \exists e'\ e \to e'.$$

In other words, we cannot get stuck when evaluating a well-typed expression.

We prove the progress lemma using structural induction on e. Recall the definition of a term in $\lambda^{\rightarrow}$:

$$e \quad ::= \quad b \;\mid\; x \;\mid\; \lambda x\!:\!\tau.\, e \;\mid\; e_0\, e_1,$$

where b denotes a constant. This gives four cases:

- Case $e = b$.

  Since $b \in \mathbf{Val}$, we are done.

- Case $e = x$.

  This case is impossible, because we cannot assign a type to x if the type environment is empty.

- Case $e = \lambda x\!:\!\sigma.\, e'$.

  Since $e \in \mathbf{Val}$, we are done.

- Case $e = e_0\, e_1$.

  We know that there is a type derivation of $\Gamma \vdash e_0\, e_1 : \tau$, and the last step of this derivation must have the form

  $$\frac{\Gamma \vdash e_0 : \sigma \rightarrow \tau \quad \Gamma \vdash e_1 : \sigma}{\Gamma \vdash e_0\, e_1 : \tau}$$

  for some type σ. By the induction hypothesis, either $e_0 \in \mathbf{Val}$ or $\exists e_0'\ e_0 \rightarrow e_0'$, and either $e_1 \in \mathbf{Val}$ or $\exists e_1'\ e_1 \rightarrow e_1'$. This gives three possibilities:

  – Both e_0 and e_1 are values. Since e_0 is a value with an arrow type $\sigma \rightarrow \tau$, it has to be an abstraction, say $e_0 = \lambda x\!:\!\sigma.\, e''$, and e_1 is some value v of type σ. Then

  $$e \quad = \quad (\lambda x\!:\!\sigma.\, e'')\ v \quad \rightarrow \quad e''\{v/x\},$$

  so e can be further reduced.

  – e_0 is not a value. Then $\exists e_0'\ e_0 \rightarrow e_0'$, and we have

  $$\frac{e_0 \rightarrow e_0'}{e_0\, e_1 \rightarrow e_0'\, e_1},$$

  so $e = e_0\, e_1$ can be further reduced.

  – e_0 is some value v, but e_1 is not a value. Then $\exists e_1'\ e_1 \rightarrow e_1'$, and we have

  $$\frac{e_1 \rightarrow e_1'}{v\, e_1 \rightarrow v\, e_1'},$$

  so $e = v\, e_1$ can be further reduced.

This completes the proof.