Suggested Solutions for Exercises

Lau
lask@cs.au.dk

August 18, 2015

Lecture 1 exercises

1. Prove SN preserved by forward/backward reduction

The lemma we want to show:

Lemma. *If* $\bullet \vdash e : \tau$ *and* $e \mapsto e'$ *then*

1. *if* $\mathrm{SN}_\tau(e')$ *then* $\mathrm{SN}_\tau(e)$
2. *if* $\mathrm{SN}_\tau(e)$ *then* $\mathrm{SN}_\tau(e')$

Proof. We proof the implications one at a time.

1. Proof by induction on the structure of τ
   Case $\tau = \mathit{bool}$,
   Assume:

   $$\bullet \vdash e : \mathit{bool} \tag{1}$$
   $$e \mapsto e' \tag{2}$$
   $$\mathrm{SN}_{\mathit{bool}}(e') \tag{3}$$

   Show:

   $$\mathrm{SN}_{\mathit{bool}}(e) \equiv \bullet \vdash e : \mathit{bool} \ \wedge \ e \Downarrow$$

   $\bullet \vdash e : \mathit{bool}$ follows from assumption 1. By definition of assumption 3 we have $e' \Downarrow$. Further, from assumption 2 we have that we can step from e to e' which gives us $e \Downarrow$
   Case $\tau = \tau_1 \to \tau_2$,
   Assume:

   $$\bullet \vdash e : \tau_1 \to \tau_2 \tag{4}$$
   $$e \mapsto e' \tag{5}$$
   $$\mathrm{SN}_{\tau_1 \to \tau_2}(e') \tag{6}$$

   Show:

   $$\mathrm{SN}_{\tau_1 \to \tau_2}(e) \equiv \bullet \vdash e : \tau_1 \to \tau_2 \ \wedge \ e \Downarrow \ \wedge \forall e''.\ \mathrm{SN}_{\tau_1}(e'') \implies \mathrm{SN}_{\tau_2}(e\ e'')$$

We get $\bullet \vdash e : \tau_1 \to \tau_2$ and $e \Downarrow$ from an argument similar to the one in the previous case. So it suffices to show

$$\forall e''.\ \mathrm{SN}_{\tau_1}(e'') \implies \mathrm{SN}_{\tau_2}(e\ e'')$$

Let e'' be given and suppose $\mathrm{SN}_{\tau_2}(e'')$. From assumption 6 we have

$$\forall e''.\ \mathrm{SN}_{\tau_1}(e'') \implies \mathrm{SN}_{\tau_2}(e'\ e'')$$

which we instantiate to get $\mathrm{SN}_{\tau_2}(e'\ e'')$. Now consider one of the induction hypotheses:

$$\forall e_1 e_2.\ \bullet \vdash e_1 : \tau_2\ \wedge\ e_1 \mapsto e_2\ \wedge\ \mathrm{SN}_{\tau_2}(e_2) \implies \mathrm{SN}_{\tau_2}(e_1)$$

We want to instantiate this with $e_1 = e\ e''$ and $e_2 = e'\ e''$. We get $\bullet \vdash e\ e'' : \tau_2$ from assumption 4 and $\mathrm{SN}_{\tau_2}(e'')$ along with the typing rule for application. To get $e\ e'' \mapsto e'\ e''$ we use assumption 5 along with our evaluation rules (use the evaluation context $[]\ e''$). Finally, we got $\mathrm{SN}_{\tau_2}(e'\ e'')$ just before we considered the induction hypothesis above. We therefore got everything needed to instantiate the induction hypothesis which gives us $\mathrm{SN}_{\tau_2}(e'\ e'')$, which was what we needed to be done.

2. Proof by induction on the structure of τ
Case $\tau = bool$,
Assume

$$\bullet \vdash e : bool \tag{7}$$
$$e \mapsto e' \tag{8}$$
$$\mathrm{SN}_{bool}(e) \tag{9}$$

Show:

$$\mathrm{SN}_{bool}(e') \equiv \bullet \vdash e' : bool\ \wedge\ e' \Downarrow$$

We get $e' \Downarrow$ from the assumptions 8 and 9 and the fact that our language is deterministic. From 9 we specifically use $e \Downarrow$. If we know that e can take several steps to some value, and we know that the first step e takes is to e', then e' will step to the same value. To argue $\bullet \vdash e' : bool$ we bring out the big guns, namely the preservation lemma[1]:

$$\Gamma \vdash e : \tau\ \wedge\ e \mapsto e' \implies \Gamma \vdash e' : \tau$$

If we use this lemma with assumption 7 and 8 we get $\bullet \vdash e' : bool$.
Case $\tau = \tau_1 \to \tau_2$,
Assume:

$$\bullet \vdash e : \tau_1 \to \tau_2 \tag{10}$$
$$e \mapsto e' \tag{11}$$
$$\mathrm{SN}_{\tau_1 \to \tau_2}(e) \tag{12}$$

Show:

$$\mathrm{SN}_{\tau_1 \to \tau_2}(e') \equiv \bullet \vdash e' : \tau_1 \to \tau_2\ \wedge\ e' \Downarrow\ \wedge \forall e''.\ \mathrm{SN}_{\tau_1}(e'') \implies \mathrm{SN}_{\tau_2}(e'\ e'')$$

[1] Here I will just assume the preservation lemma, as the full proof is rather extensive. It is however a well-known lemma for the simply typed lambda calculus, and a good account of the proof can be found in Benjamin Pierce's *Types and Programming Languages.*

The argument for $\vdash e' : \tau_1 \to \tau_2$ and $e' \Downarrow$ is similar to the one we did in the $\tau = \mathit{bool}$ case. So it suffices to show:

$$\mathrm{SN}_{\tau_1}(e'') \implies \mathrm{SN}_{\tau_2}(e'\ e'')$$

The reasoning here is similar to the same reasoning done in the same situation for the backwards part of the lemma. In short it was: suppose $\mathrm{SN}_{\tau_1}(e'')$ have $\bullet \vdash e\ e'' : \tau_2$ from assumption 10, $\mathrm{SN}_{\tau_1}(e'')$ and the application typing rule. Moreover, have $e\ e'' \mapsto e'\ e''$ from assumption 11 and the evaluation rules. Use these three things to instantiate the induction hypothesis:

$$\forall e_1, e_2.\ \bullet \vdash e_1 : \tau_2 \ \wedge\ e_1 \mapsto e_2 \ \wedge\ \mathrm{SN}_{\tau_1 \to \tau_2}(e_1) \implies \mathrm{SN}_{\tau_2}(e_2)$$

to get $\mathrm{SN}_{\tau_2}(e'\ e'')$.

□

2. Prove the substitution lemma

3. Prove the if-case of "ⓐ Generaliesd"

Prove the if case of

Theorem. *If* $\Gamma \vdash e : \tau$ *and* $\gamma \models \Gamma$*, then* $\mathrm{SN}_\tau(\gamma(e))$

Proof. The proof is by induction on the typing derivation.

Case $\dfrac{\Gamma \vdash e : \mathit{bool} \qquad \Gamma \vdash e_1 : \tau \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash \text{if } e \text{ then } e_1 \text{ else } e_2 : \tau}\ \text{T-If}$,
Assume:

$$\Gamma \vdash \text{if } e \text{ then } e_1 \text{ else } e_2 : \tau \tag{13}$$
$$\gamma \models \Gamma \tag{14}$$

Show:

$$\mathrm{SN}_\tau(\gamma(\text{if } e \text{ then } e_1 \text{ else } e_2)) \equiv \mathrm{SN}_\tau(\text{if } \gamma(e) \text{ then } \gamma(e_1) \text{ else } \gamma(e_2))$$

In this case we have the following induction hypotheses:

$$\Gamma \vdash e : \mathit{bool} \ \wedge\ \gamma \models \Gamma \implies \mathrm{SN}_{\mathit{bool}}(\gamma(e))$$
$$\Gamma \vdash e_1 : \tau \ \wedge\ \gamma \models \Gamma \implies \mathrm{SN}_\tau(\gamma(e_1))$$
$$\Gamma \vdash e_2 : \tau \ \wedge\ \gamma \models \Gamma \implies \mathrm{SN}_\tau(\gamma(e_2))$$

□

We can immediately apply the induction hypotheses as we have that e, e_1, and e_2 are well-typed with respect to *bool*, τ, and *tau*, repspectively aswell as $\gamma \models \Gamma$. We thus have $\mathrm{SN}_{\mathit{bool}}(\gamma(e))$, $\mathrm{SN}_\tau(\gamma(e_1))$, and $\mathrm{SN}_\tau(\gamma(e_2))$.

If we can show if $\gamma(e)$ then $\gamma(e_1)$ else $\gamma(e_2)$ is closed and well-typed and steps to $\gamma(e_1)$ or $\gamma(e_2)$. We can then case on what the if-expression steps to and in both cases apply the backwards part of the SN preservation lemma $\mathrm{SN}_\tau(\text{if } \gamma(e) \text{ then } \gamma(e_1) \text{ else } \gamma(e_2))$ which is the result we need.

So it suffices to show if $\gamma(e)$ then $\gamma(e_1)$ else $\gamma(e_2) \mapsto^* \gamma(e_1)$ or if $\gamma(e)$ then $\gamma(e_1)$ else $\gamma(e_2) \mapsto^* \gamma(e_1)$ and $\bullet \vdash \text{if } \gamma(e) \text{ then } \gamma(e_1) \text{ else } \gamma(e_2) : \tau$.

Let us first argue that the if-expression is well-typed and closed: from $\mathrm{SN}_{bool}(\gamma(e))$ we get $\bullet \vdash \gamma(e) : bool$, from $\mathrm{SN}_{\tau}(\gamma(e_1))$ we get $\bullet \vdash \gamma(e_1) : \tau$, and from $\mathrm{SN}_{\tau}(\gamma(e_2))$ we get $\bullet \vdash \gamma(e_2) : \tau$. This can be used with the typing rule for if-expressions to conclude $\bullet \vdash \text{if } \gamma(e) \text{ then } \gamma(e_1) \text{ else } \gamma(e_2) : \tau$.

From $\mathrm{SN}_{bool}(\gamma(e))$ we know that $\gamma(e) \Downarrow b$, for some value b. We further know $\mathrm{SN}_{bool}(b)$ from the forward part of the SN preservation lemma[2]. This gives us $\bullet \vdash b : bool$ which we can use with the canonical forms lemma[3] to get $b = \text{true}$ or $b = \text{false}$. If $b = \text{true}$, then by the evaluation rule for if-expressions we have if true then $\gamma(e_1)$ else $\gamma(e_2) \mapsto \gamma(e_1)$, so we have demonstrated that if $\gamma(e)$ then $\gamma(e_1)$ else $\gamma(e_2) \mapsto^* \gamma(e_1)$. Similar for $b = \text{false}$ we get if $\gamma(e)$ then $\gamma(e_1)$ else $\gamma(e_2) \mapsto^* \gamma(e_2)$.

We have now shown all it sufficed to show, so we are done.

4. Extend the language with pairs and do the proofs

First we extend the language with pairs as follows:

$$
\begin{aligned}
\tau &::= \ldots \mid \tau * \tau \\
e &::= \ldots \mid \langle e, e \rangle \mid \text{snd } e \mid \text{fst } e \\
v &::= \ldots \mid \langle v, v \rangle \\
E &::= \ldots \mid \text{fst } E \mid \text{snd } E \mid \langle E, e \rangle \mid \langle v, E \rangle
\end{aligned}
$$

$$
\begin{aligned}
\text{fst } \langle v_1, v_2 \rangle &\mapsto v_1 \\
\text{snd } \langle v_1, v_2 \rangle &\mapsto v_2
\end{aligned}
$$

$$
\frac{\Gamma \vdash e : \tau_1 * \tau_2}{\Gamma \vdash \text{fst } e : \tau_1} \qquad \frac{\Gamma \vdash e : \tau_1 * \tau_2}{\Gamma \vdash \text{snd } e : \tau_2} \qquad \frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 * \tau_2}
$$

Finally we extend the logical predicate:

$$
\mathrm{SN}_{\tau_1 * \tau_2}(e) \iff \bullet \vdash e : \tau_1 * \tau_2 \ \wedge \ e \Downarrow \ \wedge \ \mathrm{SN}_{\tau_1}(\text{fst } e) \ \wedge \ \mathrm{SN}_{\tau_2}(\text{snd } e)
$$

Now we just need to adjust all our proofs. We need to show safety which we did in two parts, (a) and (b). Furthermore, we had a substitution lemma and a SN preserved by backward and forward reduction lemma.

Type safety, (b)

The (b) part of type safety states that if an expression e is in the interpretation of τ, then it is strongly normalising. This is indeed the case after extended SN, as it is one of the conditions of the predicate. So this result follows trivially.

Substitution lemma

SN preserved by forward reduction

Lemma. *If* $\bullet \vdash e : \tau$, $e \mapsto e'$, *and* $\mathrm{SN}_{\tau}(e)$, *then* $\mathrm{SN}_{\tau}(e')$

[2] The lemma can be applied successively until b is reached.

[3] Not proven here. See Benjamin Pierce's *Types and Programming Languages* for an account of how to prove it.

Proof. The proof is by induction on the structure of τ. The cases we had in the proof before we added products does not change, so it only remains to show the product case:
Case $\tau = \tau_1 * \tau_2$,
Assume:

$$\bullet \vdash e : \tau_1 * \tau_2$$
$$e \mapsto e'$$
$$\mathrm{SN}_{\tau_1 * \tau_2}(e) \equiv \bullet \vdash e : \tau_1 * \tau_2 \ \wedge \ e \Downarrow \ \wedge \ \mathrm{SN}_{\tau_1}(\mathrm{fst}\ e) \ \wedge \ \mathrm{SN}_{\tau_2}(\mathrm{snd}\ e)$$

Show:

$$\mathrm{SN}_{\tau_1 * \tau_2}(e') \equiv$$
$$\bullet \vdash e' : \tau_1 * \tau_2 \wedge \tag{15}$$
$$e' \Downarrow \ \wedge \tag{16}$$
$$\mathrm{SN}_{\tau_1}(\mathrm{fst}\ e') \wedge \tag{17}$$
$$\mathrm{SN}_{\tau_2}(\mathrm{snd}\ e') \tag{18}$$

15 follows from the preservation lemma[4] used with $\bullet \vdash e : \tau_1 * \tau_2$ and $e \mapsto e'$. 16 follows from $e \mapsto e'$ and $e \Downarrow$. This time the argument is that our evaluation rules are deterministic, so if e steps to e' and e is strongly normalising, then e' will also be strongly normalising (and it will evaluate to the same value as e). Finally, we show 17 and 18 in similar fashion, so we will only show 17. To do this we apply one of the induction hypotheses, namely:

$$\bullet \vdash \mathrm{fst}\, e : \tau_1 \ \wedge \ \mathrm{fst}\ e \mapsto \mathrm{fst}\ e' \ \wedge \ \mathrm{SN}_{\tau_1}(\mathrm{fst}\ e) \implies \mathrm{SN}_{\tau_1}(\mathrm{fst}\, e')$$

$\bullet \vdash \mathrm{fst}\, e : \tau_1$ follows from $\bullet \vdash e : \tau_1 * \tau_2$ and the appropriate typing rule. $\mathrm{fst}\ e \mapsto \mathrm{fst}\ e'$ follows from the evaluation rule about evaluation contexts, $e \mapsto e'$ and the context $E = \mathrm{fst}\ []$. Finally, we have $\mathrm{SN}_{\tau_1}(\mathrm{fst}\ e)$ from one of our initial assumptions. We now have all the premises of the induction hypothesis, so we apply it to get our desired result, namely $\mathrm{SN}_{\tau_1}(\mathrm{fst}\, e')$.u $\square$

SN preserved by backward reduction

Lemma. *If* $\bullet \vdash e : \tau$, $e \mapsto e'$, *and* $\mathrm{SN}_\tau(e')$, *then* $\mathrm{SN}_\tau(e)$

Proof. The proof is by induction on the structure of τ. The cases we had in the proof before we added products does not change, so it only remains to show the product case:
Case $\tau = \tau_1 * \tau_2$,
Assume:

$$\bullet \vdash e : \tau_1 * \tau_2$$
$$e \mapsto e'$$
$$\mathrm{SN}_{\tau_1 * \tau_2}(e') \equiv \bullet \vdash e' : \tau_1 * \tau_2 \ \wedge \ e' \Downarrow \ \wedge \ \mathrm{SN}_{\tau_1}(\mathrm{fst}\ e') \ \wedge \ \mathrm{SN}_{\tau_2}(\mathrm{snd}\ e')$$

Show:

$$\mathrm{SN}_{\tau_1 * \tau_2}(e) \equiv$$
$$\bullet \vdash e : \tau_1 * \tau_2 \wedge \tag{19}$$
$$e \Downarrow \ \wedge \tag{20}$$
$$\mathrm{SN}_{\tau_1}(\mathrm{fst}\ e) \wedge \tag{21}$$
$$\mathrm{SN}_{\tau_2}(\mathrm{snd}\ e) \tag{22}$$

[4] We do not show the preservation lemma here.

We need to show the four condition above. 19 follows directly from the assumptions. 20 follows from $e \mapsto e'$ and $e' \Downarrow$. We can first take a step from e to e' and e' is strongly normalising, thus so is e. 21 and 22 are shown in similarly, so we just show 21 here. To show this we use the induction hypothesis that says:

$$\bullet \vdash \text{fst}\, e : \tau_1 \ \wedge\ \text{fst}\ e \mapsto \text{fst}\ e' \ \wedge\ \text{SN}_{\tau_1}(\text{fst}\ e') \implies \text{SN}_{\tau_1}(\text{fst}\, e)$$

We get $\bullet \vdash e : \tau_1$ from the approriate typing rule and $\bullet \vdash e : \tau_1 * \tau_2$. $\text{fst}\ e \mapsto \text{fst}\ e'$ follows from $e \mapsto e'$ and the evaluation rules if we use the context $E = \text{fst}\ []$. Finally, $\text{SN}_{\tau_1}(\text{fst}\ e')$ was assumed intially. We can now apply the induction hypothesis which gives us the desired result, $\text{SN}_{\tau_1}(\text{fst}\ e)$. □

Type safety, (a)

To show (a) we show the generalised version:

Theorem. *If* $\Gamma \vdash e : \tau$ *and* $\gamma \models \Gamma$*, then* $\text{SN}_\tau(\gamma(e))$

Proof. The proof is by induction on the typing derivation. The cases, we had before adding products, does not change, so it only remains to show the new cases.

Case $\dfrac{\Gamma \vdash e : \tau_1 * \tau_2}{\Gamma \vdash \text{fst}\ e : \tau_1}$,
Assume:

$$\gamma \models \Gamma$$
$$\Gamma \vdash e : \tau_1 * \tau_2$$

Show:

$$\text{SN}_{\tau_1}(\gamma(\text{fst}\ e)) \equiv \text{SN}_{\tau_1}(\text{fst}\ \gamma(e))$$

To show this we first apply the induction hypothesis:

$$\gamma \vdash e : \tau_1 * \tau_2 \ \wedge\ \gamma \models \Gamma \implies \text{SN}_{\tau_1 * \tau_2}(\gamma(e))$$

which we can do immediately using our assumptions. From $\text{SN}_{\tau_1*\tau_2}(\gamma(e))$ we get that $\gamma(e)$ is strongly normalising, so there exists a value v that $\gamma(e)$ evaluates to, i.e., $\gamma(e) \mapsto^* v$. Using this along with $\text{SN}_{\tau_1*\tau_2}(\gamma(e))$ and the forward part of the SN preservation lemma[5] we can now conclude $\text{SN}_{\tau_1*\tau_2}(v)$. By definition of SN this further gives us $\text{SN}_{\tau_1}(\text{fst}\ v)$. We can further use $\gamma(e) \mapsto^* v$ to conclude $\text{fst}\ \gamma(e) \mapsto^* \text{fst}\ v$ which is done using the evaluation rule repeatedly in the evaluation context $E = \text{fst}\ []$. Assume for now that we have the well-typedness conditions we need to use the backwards part of the SN preservation lemma with $\text{fst}\ \gamma(e) \mapsto^* \text{fst}\ v$ and $\text{SN}_{\tau_1}(\text{fst}\ v)$. If we have that then we get the result we want, $\text{SN}_{\tau_1}(\text{fst}\ \gamma(e))$. So it suffices to argue the well-typedness condition is okay all the way from $\text{fst}\ v$ up to $\text{fst}\ \gamma(e)$. We get this by first using the substitution lemma with $\gamma \vdash e : \tau_1 * \tau_2$ and $\gamma \models \Gamma$ to get $\bullet \vdash \gamma(e) : \tau_1 * \tau_2$. This we can use with the typing rule for fst to get $\bullet \vdash \text{fst}\ \gamma(e) : \tau_1$. We can then use preservation to argue that all the intermediate expressions when evaluating from $\text{fst}\, \gamma(e)$ to $\text{fst}\ v$ are all well typed.

Case $\dfrac{\Gamma \vdash e : \tau_1 * \tau_2}{\Gamma \vdash \text{snd}\ e : \tau_2}$,

[5] We may need to apply it repeatedly.

This case is symmetric to the one for fst, so it is omitted here.

Case $\dfrac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 * \tau_2}$,

Assume:

$$\gamma \models \Gamma$$
$$\Gamma \vdash e_1 : \tau_1$$
$$\Gamma \vdash e_2 : \tau_2$$

Show:

$$\begin{aligned} \mathrm{SN}_{\tau_1 * \tau_2}(\gamma(\langle e_1, e_2 \rangle)) \equiv \mathrm{SN}_{\tau_1 * \tau_2}(\gamma(\langle \gamma(e_1), \gamma(e_2) \rangle)) \equiv \\ \bullet \vdash \langle \gamma(e_1), \gamma(e_2) \rangle : \tau_1 * \tau_2 \wedge & \quad (23) \\ \langle \gamma(e_1), \gamma(e_2) \rangle \Downarrow \ \wedge & \quad (24) \\ \mathrm{SN}_{\tau_1}(\mathrm{fst}\ \langle \gamma(e_1), \gamma(e_2) \rangle) \wedge & \quad (25) \\ \mathrm{SN}_{\tau_2}(\mathrm{snd}\ \langle \gamma(e_1), \gamma(e_2) \rangle) & \quad (26) \end{aligned}$$

We can show 23 by first applying the substitution lemma twice with $\gamma \models \Gamma$ as the substitution in both cases and with $\Gamma \vdash e_1 : \tau_1$ and $\Gamma \vdash e_2 : \tau_2$ as the two typing judgements. This gives us $\Gamma \vdash \gamma(e_1) : \tau_1$ and $\Gamma \vdash \gamma(e_2) : \tau_2$. It is then a simple matter of applying the typing rule to get $\Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 * \tau_2$.

To show the rest we need to use the induction hypotheses:

$$\Gamma \models \gamma \ \wedge \ \Gamma \vdash e_1 : \tau_1 \implies \mathrm{SN}_{\tau_1}(\gamma(e_1))$$
$$\Gamma \models \gamma \ \wedge \ \Gamma \vdash e_2 : \tau_2 \implies \mathrm{SN}_{\tau_2}(\gamma(e_2))$$

We can apply the induction hypotheses immidiately to get $\mathrm{SN}_{\tau_1}(\gamma(e_1))$ and $\mathrm{SN}_{\tau_2}(\gamma(e_2))$. No matter what type τ_1 and τ_2 are we have the following:

$$\mathrm{SN}_{\tau_1}(\gamma(e_1)) \implies \gamma(e_1) \Downarrow$$
$$\mathrm{SN}_{\tau_2}(\gamma(e_2)) \implies \gamma(e_2) \Downarrow$$

Assume that $\gamma(e_1)$ and $\gamma(e_2)$ respectively evaluate to some values v_1 and v_2. If we inspect the evaluation rules we see that this gives us

$$\langle \gamma(e_1), \gamma(e_2) \rangle \mapsto^* \langle v_1, \gamma(e_2) \rangle \mapsto^* \langle v_1, v_2 \rangle$$

So we have shown 24. It remains to show 25 and 26. The two turns out to be symmetric, so we will just show 25 here. To do so we first observe that the above evaluation used with the evaluation rules in the evaluation context $E = \mathrm{fst}\ []$ gives us

$$\mathrm{fst}\ \langle \gamma(e_1), \gamma(e_2) \rangle \mapsto^* v_1$$

We further had $\mathrm{SN}_{\tau_1}(\gamma(e_1))$ from our induction hypothesis which we use with $\gamma(e_1) \mapsto^* v_1$ (follows from the strong normalisation property of $\mathrm{SN}_{\tau_1}(\gamma(e_1))$) and $\bullet \vdash \gamma(e_1) : \tau_1$ (which we had previously as an intermediate derivation) with the SN forward preservation lemma to conclude $\mathrm{SN}_{\tau_1}(v_1)$. If we use $\bullet \vdash \langle \gamma(e_1), \gamma(e_2) \rangle : \tau_1 * \tau_2$ with the fst typing rule, we get $\bullet \vdash \mathrm{fst}\ \langle \gamma(e_1), \gamma(e_2) \rangle : \tau_1 * \tau_2$ which we can use with the preservation lemma and the evaluation to v_1 to conclude that every intermediate step is closed and well-typed. This sets the scene for the backward part of the SN preservation lemma which we use repeatedly to conclude $\mathrm{SN}_{\tau_1}(\mathrm{fst}\ \langle \gamma(e_1), \gamma(e_2) \rangle)$ which is the result we wanted. □

Lecture 2 exercies

Prove the TApp case of the Fundamental Property theorem

Theorem (Fundamental Property)**.** *If* $\Gamma \vdash e : \tau$ *then* $\Gamma \models e : \tau$

Proof. proof by induction over the typing derivation. Here we will only show the case asked for in the exercises.

Case $\dfrac{\Gamma \vdash e_1 : \tau_2 \to \tau \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\ e_2 : \tau}$ T-App,

In this case we assume $\Gamma \vdash e_1\ e_2 : \tau$ and we want to show $\Gamma \models e_1\ e_2 : \tau$. Suppose we have a $\gamma \in \mathcal{G}[\![\Gamma]\!]$, then we need to show $\gamma(e_1\ e_2) \in \mathcal{E}[\![\Gamma]\!]$ which is equivalent to $\gamma(e_1)\ \gamma(e_2) \in \mathcal{E}[\![(]\!]\Gamma)$. Looking at the definition of $\mathcal{E}$ we may now suppose $\gamma(e_1)\ \gamma(e_2) \mapsto^* e'$ and irred(e') for some e'. We then need to show $e' \in \mathcal{V}[\![\tau]\!]$.

We now turn our attention to the induction hypotheses:

$$\Gamma \models e_1 : \tau_2 \to \tau$$
$$\Gamma \models e_2 : \tau_2$$

From the first induction hypothesis instantiated with $\gamma \in \mathcal{G}[\![\gamma]\!]$ we get $\gamma(e_1) \in \mathcal{E}[\![\tau_2 \to \tau]\!]$. Using the operational semantics we know that $\gamma(e_1)$ evaluates to some expression e_1' where irred(e_1'). With this we instantiate $\gamma(e_1) \in \mathcal{E}[\![\tau_2 \to \tau]\!]$ to get $e_1' \in \mathcal{V}[\![\tau_2 \to \tau]\!]$ which further means that e_1' must be of the form $\lambda x : \tau.\ e''$ for some e''.

From the second induction hypothesis instantiated with $\gamma \in \mathcal{G}[\![\Gamma]\!]$ we get $gamma(e_2) \in \mathcal{E}[\![\tau_2]\!]$. Again from the operational semantics we have $\gamma(e_2)$ evaluates to some e_2' in a number of steps and irred e_2'. With this we instantiate $gamma(e_2) \in \mathcal{E}[\![\tau_2]\!]$ to get $e_2' \in \mathcal{V}[\![\tau_2]\!]$.

We can now instantiate $e_1' \in \mathcal{V}[\![\tau_2 \to \tau]\!]$ with $e_2' \in \mathcal{V}[\![\tau_2]\!]$. to get $e''[^{e_2'}/_x] \in \mathcal{E}[\![\tau]\!]$. Now let us take a step back and consider what we have. We have argued the parts necessary for the following steps to take place:

$$\begin{aligned}\gamma(e_1)\ \gamma(e_2) &\mapsto^* e_1'\ \gamma(e_2)\\ &\equiv (\lambda x : \tau.\ e'')\ \gamma(e_2)\\ &\mapsto^* (\lambda x : \tau.\ e'')\ e_2'\\ &\mapsto e''[^{e_2'}/_x] \qquad (27)\end{aligned}$$

We can do the β-reduction in the last step (27) because e_2' is in the value interpretation of τ_2 which means that it is a value.

Previously in the proof, we assumed $\gamma(e_1)\ \gamma(e_2) \mapsto^* e'$ which we can use with the fact that our language is deterministic to argue that $e''[^{e_2'}/_x] \mapsto^* e'$. This allows us to instantiate $e''[^{e_2'}/_x] \in \mathcal{E}[\![\tau]\!]$ and get the result we needed to show, namely $e' \in vpred\tau$. □

Lecture 3 exercises

2. Attempt to prove monotonicity with adjusted value interpretation

Exercise:

> Try to prove the monotonicity lemma where the definition of the value interpretation has been adjusted with:
>
> $$\mathcal{V}_k[\![\tau_1 \to \tau_2]\!] = \{\lambda x : \tau_1.\ e \mid\ \forall v \in \mathcal{V}_k[\![\tau_1]\!].\ e[^v/_x] \in \mathcal{E}_k[\![\tau_2]\!]\}$$

This will fail, but it is instructive to see how it fails.

Theorem. *If* $v \in \mathcal{V}_k[\![\tau]\!]$ *and* $j \leq k$ *then* $v \in \mathcal{V}_j[\![\tau]\!]$.

Proof. We attempt a proof by induction over the structure of τ.
Case $\tau = \tau_1 \to \tau_2$,
We assume

$$v \in \mathcal{V}_k[\![\tau]\!] \tag{28}$$
$$j \leq k$$

We need to show

$$v \in \mathcal{V}_j[\![\tau]\!] \tag{29}$$

To show 29 we suppose $v \in \mathcal{V}_j[\![\tau_1]\!]$, and then it suffices to show $e[v/x] \in \mathcal{E}_j[\![\tau_2]\!]$. By definition we know from 28

$$\forall v \in \mathcal{V}_k[\![\tau_1]\!].\ e[v/x] \in \mathcal{E}_k[\![\tau_2]\!] \tag{30}$$

Now let us take a step back and consider what we got, and what we want to show. We want to show $e[v/x] \in \mathcal{E}_j[\![\tau_2]\!]$, but the only thing we got that mentions the expression intepretation is 30, but it only talks about the expression interpretation at step k, and we need it at step j. Now let us consider the induction hypotheses:

$$v \in \mathcal{V}_k[\![\tau_n]\!] \wedge j \leq k \implies v \in \mathcal{V}_j[\![\tau_n]\!] \quad , n = 1, 2$$

We do have (28) that talks about the value interpretation for τ, but we already unfolded that and it did not give us anything that talks about the value interpretation for τ_1 or τ_2 at k steps. It would seem like we are stuck. ×

Lecture 4 exercises

Prove the following free theorem

Theorem (Free Theorem (II))**.** *If* $\bullet;\bullet \vdash e : \forall\alpha.\ ((\tau \to \alpha) \to \alpha)$ *and* $\bullet;\bullet \vdash k : \tau \to \tau_k$ *then*

$$\bullet;\bullet \vdash e[\tau_k]\ k \approx k\ (e[\tau]\ id) : \tau_k$$

where id *is* $\lambda x : \tau.\ x$

Proof. To show that the two expressions are equivalent we need to show three things:

1. $\bullet;\bullet \vdash e[\tau_k]\ k : \tau_k$
2. $\bullet;\bullet \vdash k\ (e[\tau]\ id)$
3. $\exists v_1, v_2. e[\tau_k]\ k \mapsto^* v_1\ \wedge\ k\ (e[\tau]\ id) \mapsto^* v_2\ \wedge\ (v_1, v_2) \in \mathcal{V}[\![\tau_k]\!]_\emptyset$

The two expressions are clearly well-typed (the difficult part of this proof is not to show this, so accept the handwaving or verify it yourself), so it remains to show the third item. Keep this proof goal in mind as we proceed.

Now, let us turn our attention to our two assumptions $\bullet;\bullet \vdash e : \forall\alpha.\ ((\tau \to \alpha) \to \alpha)$ and $\bullet;\bullet \vdash k : \tau \to \tau_k$. From the fundamental property these give us that the two are related to themself:

$$\bullet;\bullet \vdash e \approx e : \forall\alpha.(\tau \to \alpha) \to \alpha$$
$$\bullet;\bullet \vdash k \approx k : \tau \to \tau_k$$

This gives us that e evaluates to some value, say F, where $(F, F) \in \mathcal{V}[\![\tau \to \tau_k]\!]_\emptyset$. Which we can unfold to get:

$$\forall\tau_1, \tau_2, R \in \mathrm{Rel}[\tau_1, \tau_2].(F[\tau_1], F[\tau_2]) \in \mathcal{E}[\![(\tau \to \alpha) \to \alpha]\!]_{\emptyset[\alpha \mapsto (\tau_1, \tau_2, R)]}$$

Let us for convenience write $\rho = \emptyset[\alpha \mapsto (\tau_1, \tau_2, R)]$. We get something similar for k, but let us for now just take with us that k evaluates to some value k_λ. To proceed we need to instantiate the above with two types and a relation. To determine what types and relation to use we look at what we want to prove. In the free theorem, e is instantiated with τ_k on the left side and τ on the right side, so we probably need to use those two types. When it comes to the relation we again let us inspire by the free theorem[6]: On the left side we just apply e, and it is the last thing we do. On the right side we first apply e, and then we give the result to k. So the value on the left side should be related to the value on the right side, when it has been applied to k. We end up with the following relation:

$$R = \{(v_k, v) \mid \exists v'_k.\ k\ v \mapsto^* v'_k\ \wedge\ (v_k, v'_k) \in \mathcal{V}[\![\tau_k]\!]_\emptyset\}$$

The instantiation gives us that $(F[\tau_k], F[\tau]) \in \mathcal{E}[\![(\tau \to \alpha) \to \alpha]\!]_\rho$ which entails that $F[\tau_k]$ and $F[\tau]$ evaluate to some values:

$$F[\tau_k] \mapsto^* f_L$$
$$F[\tau] \mapsto^* f_R$$

where $(f_L, f_R) \in \mathcal{V}[\![(\tau \to \alpha) \to \alpha]\!]_\rho$. By definition of the value interpretation of the function type we then have

$$\forall(v_1, v_2) \in \mathcal{V}[\![\tau \to \alpha]\!]_\rho.\ (f_L\ v_1, f_R\ v_2) \in \mathcal{E}[\![\alpha]\!]_\rho$$

Assume for now that $(k, id) \in \mathcal{V}[\![\tau \to \alpha]\!]_\rho$, then it must be the case that

$$f_L\ k \mapsto^* v_1$$
$$f_R\ id \mapsto^* v_2$$

where $(v_1, v_2) \in \mathcal{V}[\![\alpha]\!]_\rho$ which is equivalent to $(v_1, v_2) \in R$. Now let us take a step back and see what we have shown. We have shown that the following evaluation takes place:

$$\begin{aligned} e[\tau_k]\ k &\mapsto^* F[\tau_k]\ k \\ &\mapsto^* f_L\ k \\ &\mapsto^* f_L\ k_\lambda \\ &\mapsto^* v_1 \end{aligned}$$

[6]It may not be easy to find the correct relation in the first try, but it is a good idea to either try something that seems plausible or leave it open and continue unfolding definitions until you get to something where you need R and can see what constraints it needs to have.

Which shows the left equation of the equivalence evaluates to some vales. Further we have the evaluation

$$\begin{aligned} e[\tau]\ id &\mapsto^* F[\tau]\ k \\ &\mapsto^* f_R\ k \\ &\mapsto^* f_R\ k_\lambda \\ &\mapsto^* v_2 \end{aligned}$$

This shows that part of the right equation of the equivalence evaluates to a value, but it is not quite what we wanted. Luckily we defined our relation R such that it can take us the rest of the way. We know that $(v_1, v_2) \in R$ which gives us that $k\ v_2 \mapsto^* v_2'$ for some v_2' and $(v_1, v_2') \in \mathcal{V}[\![\tau_k]\!]_\emptyset$, so we end up with the evaluation

$$\begin{aligned} k\ (e[\tau]\ id) &\mapsto^* k_\lambda\ (e[\tau]\ id) \\ &\mapsto^* k_\lambda\ v_2 \\ &\mapsto^* v_2' \end{aligned}$$

where $(v_1, v_2') \in \mathcal{V}[\![\tau_k]\!]_\emptyset$ which is exactly the result we wanted.

We did, however, assume something under the way, so now it suffices to show $(k, id) \in \mathcal{V}[\![\tau \to \alpha]\!]_\rho$. To do this suppose $(v_L, v_R) \in \mathcal{V}[\![\tau]\!]_\rho$ and then show $(k\ v_L, id\ v_R) \in \mathcal{E}[\![\alpha]\!]_\rho$. To show this we need to show that $k\ v_L$ evaluates to some value, say v_L', and that $id\ v_R$ evaluates to some value, say v_R', where $(v_L', v_R') \in \mathcal{V}[\![\alpha]\!]_\rho \equiv R$. As id is the identity function we know that $id\ v_R$ evaluates to v_R in one step. To move on we need to consider what the fundamental property gave us about k. We already had that k evaluates to k_λ, but we also get that $(k, k) \in \mathcal{E}[\![\tau \to \tau_k]\!]_\emptyset$ which means that $(k_\lambda, k_\lambda) \in \mathcal{V}[\![\tau \to \tau_k]\!]_\emptyset$. If we unfold this we get:

$$\forall (v_1, v_2) \in \mathcal{V}[\![\tau]\!]_\emptyset.\ (k_\lambda\ v_1, k_\lambda\ v_2) \in \mathcal{E}[\![\tau_k]\!]_\emptyset$$

To instantiate this we need something in $\mathcal{V}[\![\tau]\!]_\emptyset$, but all we got is $(v_L, v_R) \in \mathcal{V}[\![\tau]\!]_\rho$. We will prove as a lemma that because τ is closed $(v_L, v_R) \in \mathcal{V}[\![\tau]\!]_\rho$ implies $(v_L, v_R) \in \mathcal{V}[\![\tau]\!]_\emptyset$. So for now let us assume that this is true so we have $(v_L, v_R) \in \mathcal{V}[\![\tau]\!]_\emptyset$ which we instantiate the above with to get $(k_\lambda\ v_L, k_\lambda\ v_R) \in \mathcal{E}[\![\tau_k]\!]_\emptyset$. This gives us that $k_\lambda\ v_L$ evaluates to some value, say v_L', and $k_\lambda\ v_R$ evaluates to some value, say v_R', and that $(v_L', v_R') \in \mathcal{V}[\![\tau]\!]_\rho$. This gives us one more of our goals namely that $k\ v_L$ evaluates to v_L' so it remains to show $(v_L', v_R) \in R$. To do this we need to show two things, first we need to show $k\ v_r$ evaluates to some value which it does, it evaluates to v_R'. Second, we need to show that $v_L', v_R') \in \mathcal{V}[\![\tau_k]\!]_\emptyset$ which is the other result we just got, so we have the result we needed. There is now nothing left to show[7], so we have proven the free theorem. □

[7] Apart from the mentioned lemma.