Assignment 2

Vivek Pandya (vpandya)

Problem 2

Part 1:

Step 1:

$$\cfrac{\cfrac{\cfrac{\cfrac{}{\lambda\ \_\ .\ x\ \mathsf{val}}\ (\textsc{D-Lam})}{\{x \to D\} \vdash (\lambda\ x\ .\ \lambda\ \_\ .\ x)\ L \mapsto \lambda\ \_\ .\ x}\ (\textsc{D-App-Done})}{\{x \to D\} \vdash (\lambda\ x\ .\ \lambda\ \_\ .\ x)\ L\ * \mapsto (\lambda\ \_\ .\ x)\ *}\ (\textsc{D-App-Lam})}{\varnothing \vdash (\lambda\ x\ .\ (\lambda\ x\ .\ \lambda\ \_\ .\ x)\ L\ *)\ D \mapsto (\lambda\ x\ .\ (\lambda\ \_\ .\ x)\ *)\ D}\ (\textsc{D-App-Body})$$

Step 2:

$$\cfrac{\cfrac{\cfrac{(x \to D \in \Gamma)}{\{x \to D, \_ \to *\} \vdash x \mapsto D}\ (\textsc{D-Var})}{\{x \to D\} \vdash (\lambda\ \_\ .\ x)\ * \mapsto (\lambda\ \_\ .\ D)\ *}\ (\textsc{D-App-Body})}{\varnothing \vdash (\lambda\ x\ .\ (\lambda\ \_\ .\ x)\ *)\ D \mapsto (\lambda\ x\ .\ (\lambda\ \_\ .\ D)\ *)\ D}\ (\textsc{D-App-Body})$$

Step 3:

$$\cfrac{\cfrac{D\ \mathsf{val}}{\{x \to D\} \vdash (\lambda\ \_\ .\ D)\ * \mapsto D}\ (\textsc{D-App-Done})}{\varnothing \vdash (\lambda\ x\ .\ (\lambda\ \_\ .\ D)\ *)\ D \mapsto (\lambda\ x\ .\ D)\ D}\ (\textsc{D-App-Body})$$

Step 4:

$$\cfrac{D\ \mathsf{val}}{\varnothing \vdash (\lambda\ x\ .\ D)\ D \mapsto D}\ (\textsc{D-App-Done})$$

Part 2:

Following rules in addition to all rules described in problem 2 are required to support let syntax with dynamic scope:

$$\cfrac{\Gamma, x \to e_{\mathsf{var}} \vdash e_{\mathsf{body}} \mapsto e'_{\mathsf{body}}}{\Gamma \vdash let\, x\ =\ e_{\mathsf{var}}\, in\, e_{\mathsf{body}} \mapsto e'_{\mathsf{body}}}\ (\textsc{D-Let1})$$

$$\cfrac{e_{\mathsf{body}}\ \mathsf{val}}{\Gamma \vdash let\, x\ =\ e_{\mathsf{var}}\, in\, e_{\mathsf{body}} \mapsto e_{\mathsf{body}}}\ (\textsc{D-Let2})$$

Problem 3

Consider follwing counter example for let construct.

$let x : (\mathsf{number} \mapsto \mathsf{number}) = 2\, in\, (x\, 2))$

Now given that x : $(\mathsf{number} \mapsto \mathsf{number})$ by inversion of T-app rule we can say that $(x\, 2) : \mathsf{number}$

So premise for T-let holds here so we can say that

$let x : (\mathsf{number} \mapsto \mathsf{number}) = 2\, in\, (x\, 2)) : \mathsf{number}$

but now we try to step the above expression by applying D-let rule then we get

$e^{`}_{\mathsf{body}} = (2\, 2)$ which is a stuck state as we don't have any rule to make a further progress also (2 2) it self is not a val. Here preservation also don't hold because type of (2 2) is not number

Note: This can be fixed if we have restriction on the type of e_{var} , type of e_{var} should be same as τ_{var}.

For rec construct consider following proofs.

Proof. Preservation: if $\emptyset \vdash e\ : \tau$ and $e \mapsto e^{`}$ then $\emptyset \vdash e^{`} : \tau$.
Proof. By rule induction on the static semantics.

T-Rec: if $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) : \tau and \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) \mapsto e^{`} then e^{`} : \mathsf{number}$
Frist by premises
$e_{\mathsf{arg}} : \mathsf{number}, e_{\mathsf{base}} : \tau,\ \ x_{\mathsf{num}} : \mathsf{number}, x_{\mathsf{acc}} : \tau \vdash e_{\mathsf{acc}} : \tau$

Induction Hypothesis : For $e_{\mathsf{arg}}, e_{\mathsf{base}}, x_{\mathsf{num}}, x_{\mathsf{acc}}, e_{\mathsf{acc}}$ preservation rule holds true.
Now we have 3 ways for $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) \mapsto e^{`}$

(D-Rec-Step) : Assume $e_{\mathsf{arg}} \mapsto e^{`}_{\mathsf{arg}}$ so $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) \mapsto \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e^{`}_{\mathsf{arg}})$

Now due to Induction Hypothesis we can have $e^{`}_{\mathsf{arg}} : \tau$ and given premises
by T-Rec $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e^{`}_{\mathsf{arg}}) : \tau$

(D-Rec-Base) : Assume $e_{\mathsf{arg}} = 0$ then by Inversion T-Num $e_{\mathsf{arg}} : \mathsf{number}$ *so* $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(0)\ :\ \tau$ and $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(0) \mapsto e_{\mathsf{base}}$, now based on premises $e_{\mathsf{base}} : \tau$

(D-Rec-Dec) : Assume e_{arg} =n and n $>$ 0

$\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(n) \mapsto [x_{\mathsf{num}} \to n, x_{\mathsf{acc}} \to \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(n-1)]e_{\mathsf{acc}}$

now if n : number then for (n - 1) by inversion of T-Binop , (n-1) : 0 , based on induction hypothesis we get $x_{\mathsf{num}} \mapsto n$ and $x_{\mathsf{acc}} \mapsto \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(n{-}1)$ are type preserving operations, and from premises $\Gamma, x_{\mathsf{num}} : \mathsf{number}, x_{\mathsf{acc}} : \tau \vdash e_{\mathsf{acc}} : \tau$

then by the substitution typing lemma (as reffered in lacture notes (foot note: 6))

$[x_{\mathsf{num}} \to n, x_{\mathsf{acc}} \to \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(n-1)]\ e_{\mathsf{acc}} : \tau$

Hence, preservation holds in either case.

□

Proof. Progress: if $\emptyset \vdash e\ : \tau$ then *either* $e\,\mathsf{val}$ *or* $e \mapsto e^{`}$.
Proof : By rule induction on static semantics.

T-Rec: if $e = \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) : \tau$ then either $e\,\mathsf{val}$ or
$\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) \mapsto e^{`}$

From premises we know that
$e_{\mathsf{arg}} : \mathsf{number}, e_{\mathsf{base}} : \tau,\ \ x_{\mathsf{num}} : \mathsf{number}, x_{\mathsf{acc}} : \tau \vdash e_{\mathsf{acc}} : \tau$

By the inductive hypothesis (IH), we got to assume that progress holds true for e_{arg} so either $e_{\mathsf{arg}} val$ or $e_{\mathsf{arg}} \mapsto e^{`}_{\mathsf{arg}}$

Now we case on different possible states of e_{arg} derived from IH:

A: $e_{\mathsf{arg}} \mapsto e^{`}_{\mathsf{arg}}$ then by D-Rec-Step $\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e_{\mathsf{arg}}) \mapsto \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(e^{`}_{\mathsf{arg}})$

B: $e_{\mathsf{arg}} val$ and by premise $e_{\mathsf{arg}} : \mathsf{number}$ then by inversion of D-Num we know that $e_{\mathsf{arg}} = n$ Now if n $=$ 0 then by D-Rec-Base

$\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(0) \mapsto e_{\mathsf{base}}$

if n $>$ 0 then by D-Rec-Dec

$\mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(n) \mapsto [x_{\mathsf{num}} \to n, x_{\mathsf{acc}} \to \mathsf{rec}(e_{\mathsf{base}};\ x_{\mathsf{num}}.x_{\mathsf{acc}}.e_{\mathsf{acc}})(n-1)]e_{\mathsf{acc}}$
In each case expression steps, so progress holds. □