APL: Sheet Four

Hopefully you all know the routine: new lang, new rules + drawing same derivations and transition sequences to acquaint ourselves with them.

(will only present in class if asked)

① i) let $\Gamma = x : Str + (Str \times Num)$
$\tau_1 = Str$
$\tau_2 = (Str \times Num)$
$\tau_3 = Str$
$\tau_4 = Num$

$$\dfrac{\dfrac{}{\Gamma \vdash x : \tau_1 + \tau_2}\,\text{VAR} \quad \dfrac{}{\Gamma, y : \tau_1 \vdash y : Str}\,\text{VAR} \quad \dfrac{\dfrac{}{\Gamma, z : \tau_2 \vdash z : \tau_3 \times \tau_4}\,\text{VAR}}{\Gamma, z : \tau_2 \vdash \pi_1(z) : Str}\,\text{PROJ}_1}{\Gamma \vdash (case\ (x; y.y; z.\pi_1(z))) : Str}\,\text{CASE}$$

ii) let $\Gamma = x : Str + Num$
$\tau_1 = Str$
$\tau_2 = Num$

$$\dfrac{\dfrac{}{\Gamma \vdash x : \tau_1 + \tau_2}\,\text{VAR} \quad \dfrac{\dfrac{}{\Gamma, y : \tau_1 \vdash y : Str}\,\text{VAR}}{\Gamma, y : \tau_1 \vdash inr(y) : Num + Str}\,\text{INR} \quad \dfrac{\dfrac{}{\Gamma, z : \tau_2 \vdash z : Num}\,\text{VAR}}{\Gamma, z : \tau_2 \vdash inL(z) : Num + Str}\,\text{INL}}{\Gamma \vdash case\ (x; y.inr(y); z.inL(z)) : Num + Str}\,\text{CASE}$$

$$\dfrac{\vdash \lambda x : Str + Num\ .\ case\ (x; y.inr(y); z.inL(z)) : Str + Num \to Num + Str}{}\,\text{LAM}$$

iii) let $\Gamma = f : Num \times Shr \to Num,\ x : Shr$
$\sigma = Num \times Shr \to Num$

$$\dfrac{\dfrac{}{\Gamma \vdash f : \sigma \to Num}\,VAR \qquad \dfrac{\dfrac{\dfrac{n \in \mathbb{N}}{\Gamma \vdash Num[n] : Num}\,NUM \qquad \dfrac{}{\Gamma \vdash x : Shr}\,VAR}{\Gamma \vdash (\langle Num[n], x \rangle) : \sigma}\,PROD}{\Gamma \vdash f(\langle Num[n], x \rangle) : Num}\,APP$$

② (i)

$case\ (inr(\langle Shr['hi'], Num[0] \rangle);\ y.y;\ z.\pi_1(z))$

$\to \pi_1(\langle Shr['hi'], Num[0] \rangle)$

$\to Shr['hi']$

ii)

$(\lambda x.\ Shr + Num.case(x;\ y.inr(y);\ z.inl(z)))\ (inl(Num[0]))$

$\to case(inl(Num[0]);\ y.inr(y);\ z.inl(z))$

$\to inr(Num[0])$

iii)

$(\lambda z : Num \times Shr.\ \pi_1(z))(\langle Num[0], Shr['hi'] \rangle)$

$\to \pi_1(\langle Num[0], Shr['hi'] \rangle)$

$\to Num[0]$

(3) (i)

In STLC we have the following types:

- Str
- Num
- +
- ×
- 1

Even in Haskell, we can redefine MaybeString using these:

```
> type MaybeString = Either () String
```

$$+ \quad 1 \quad Str$$

So in STLC, we can use:

$$1 + Str$$

ii) Recall that derivable means that a derivation of the conclusion ends in a derivation of the premise.

⇒ we will work on a derivation of the conclusion, hoping to hit the premise.

```
Nothing : MaybeString
≡ Def. 3
inL  :  1 + Str
```

1. Re-written as defined type
2. Showed derivability

$$\frac{\dfrac{}{\Gamma \vdash () : 1}\ \text{UNIT}}{\Gamma \vdash \text{inL}\ () : 1 + Str}\ \text{INL}$$

Just(e) : Maybe Str
≡ {def}
inr(e) : 1 + Str

$$\frac{\vdots \text{ premise of rule}}{\dfrac{\Gamma \vdash e : \mathrm{Str}}{\Gamma \vdash \mathrm{inr}(e) : \mathrm{Str}}\ \mathrm{INR}}$$

We can derive the premise of the rule from its conclusion => this rule is derivable.

Match(e; en; x.es) : τ
≡ {def}
case(e; inl(y).en; inr(x).es)

let $\tau_1 = 1$, $\tau_2 = \mathrm{Str}$

$$\frac{\dfrac{\vdots}{\Gamma \vdash e : \tau_1 + \tau_2} \qquad \dfrac{\dfrac{\vdots}{\Gamma \vdash e_n : \tau}}{\Gamma, x : \tau_1 \vdash e_n : \tau}\ \mathrm{WK} \qquad \dfrac{\vdots}{\Gamma, y : \tau_2 \vdash e_s : \tau}}{\Gamma \vdash \mathrm{case}(e; \mathrm{inl}(y).e_n; \mathrm{inr}(x).e_s) : \tau}\ \mathrm{CASE}$$

WK = "weakening"
This is the following admissible rule:

$$\frac{\Gamma \vdash e : \tau}{\Gamma, x : \tau_1 \vdash e : \tau}\ \mathrm{WK}$$

Admissibility requires an inductive proof, but basically we have a derivation for $\Gamma \vdash e : \tau$ and in each case where weakening we have to the context does nothing.

We can derive the premise of the rule from its conclusion => this rule is derivable.

(4) Big Q. So best make sure we unpack it right to ensure we answer it

constants-and-functions fragment

excellent this will save us time as we only need to deal with

Statics | Dynamics

LAM | D-BETA

APP | D-APP-1

(VAL-LAM) but not when reducing an ↦

(as the hint says, last week's proofs did the rest of the fragment)

The Q even suggests us steps!

Plan:

1. extend
   - inversion — CLAIM 1
   - subst — CLAIM 2
   - canonical forms to function type — CLAIM 3
2. Assume weakening — ASS-WK
3. Prove progress — CLAIM 4
4. Prove preservation — CLAIM 5

CLAIM 1. (Inversion):

Suppose $\Gamma \vdash e : \tau$

(LAM INV) • If $e = \lambda x:\sigma.e'$ then $\tau = \sigma \to \tau'$ for some type τ', and $\Gamma, x:\sigma \vdash e_2 : \tau'$

(APP INV) • If $e = e_1(e_2)$ then there exists a τ_2 s.t. $\Gamma \vdash e_1 : \tau_2 \to \tau$ and $\Gamma \vdash e_2 : \tau_2$

this is precisely what the rules say.

Proof by inspection

$$\text{LAM}\ \frac{\Gamma, x:\sigma \vdash e:\tau}{\Gamma \vdash \lambda x:\sigma.e : \sigma \to \tau}$$

To have the judgement $\Gamma \vdash \lambda x:\sigma.e : \sigma \to \tau$, we must have a derivation for $\Gamma, x:\sigma \vdash e:\tau$.

In other words LAM-INV

$$\text{APP}\ \frac{\Gamma \vdash e_1 : \sigma \to \tau \quad \Gamma \vdash e_2 : \sigma}{\Gamma \vdash e_1(e_2) : \tau}$$

To have a judgement $\Gamma \vdash e_1(e_2) : \tau$ we must have derivations for $\Gamma \vdash e_1 : \sigma \to \tau$ and $\Gamma \vdash e_2 : \sigma$

In other words APP-INV.

□ CLAIM 1 (Inversion)

CLAIM 2 (substitution):

If $\Gamma \vdash e : \tau$ and $\Gamma, x:\tau \vdash u : \sigma$
then $\Gamma \vdash u\{e/x\} : \sigma$

We have already shown this for the previous parts of the lab (lecture 4 notes sheet 2) so we know this is the proof by induction on our second premise.

If we didn't already know we could look and see we have two premises of the same form and pick the one we're working on.

Proof by induction on $\Gamma, x:\tau \vdash u : \sigma$.

(remember only doing induction steps of LAM and APP)

[Case: LAM]

$\Rightarrow \Gamma, x:\tau \vdash u : \sigma$ has shape $\Gamma, x:\tau \vdash \lambda y:\sigma_1.u' : \sigma_1 \to \sigma_2$

and derivation

$$\dfrac{\dfrac{\vdots\ \text{SHAPE}}{\Gamma, x:\tau, y:\sigma_1 \vdash u' : \sigma_2}}{\Gamma, x:\tau \vdash \lambda y:\sigma_1.u' : \sigma_1 \to \sigma_2}\ \text{LAM}$$

smaller same shape $\Rightarrow$ IH

u': specialized with carefully chosen var names to avoid conflict

IH = if then $\Gamma, y:\sigma \vdash e:\tau$ and $\Gamma, x:\tau, y:\sigma \vdash u' : \sigma_2$ then $\Gamma, y:\sigma \vdash u'[e/x] : \sigma_2$

ASS(the premise) $= \Gamma \vdash e : \tau$

GOAL $= \Gamma \vdash u[e/x] : \sigma$

$= \Gamma \vdash (\lambda y.u')[e/x] : \sigma_1 \to \sigma_2$

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots\ \text{ASS}}{\Gamma \vdash e:\tau}\ \text{WK}}{\Gamma, y:\sigma_1 \vdash e:\tau} \qquad \dfrac{\vdots\ \text{SHAPE}}{\Gamma, x:\tau, y:\sigma \vdash u' : \sigma_2}}{\Gamma, y:\sigma_1 \vdash u'[e/x] : \sigma_2}\ \text{IH}}{\Gamma \vdash (\lambda y:\sigma_1.u'[e/x]) : \sigma_1 \to \sigma_2}\ \text{LAM}}{\Gamma \vdash (\lambda y:\sigma_1.u')[e/x] : \sigma_1 \to \sigma_2}\ \text{def subst}$$

$\square_{\text{LAM}}$

[Case: APP]

$\Rightarrow \Gamma \vdash u : \sigma$ has shape $\Gamma \vdash u_1(u_2) : \sigma$ and derivation

$$\dfrac{\vdots \text{SHAPE1} \qquad \vdots \text{SHAPE2}}{\dfrac{\Gamma, x:\tau \vdash u_1 : \tau_1 \to \sigma \qquad \Gamma, x:\tau \vdash u_2 : \tau_1}{\Gamma, x:\tau \vdash u_1(u_2) : \sigma}\ \text{APP}}$$

IH1 = if $\Gamma \vdash e : \tau$ and $\Gamma, x:\tau \vdash u_1 : \tau_1 \to \sigma$ then $\Gamma \vdash u_1[e/x] : \tau_1 \to \sigma$

IH2 = if $\Gamma \vdash e : \tau$ and $\Gamma, x:\tau \vdash u_2 : \tau_1$ then $\Gamma \vdash u_2[e/x] : \tau_1$

ASS (the premise) = $\Gamma \vdash e : \tau$

GOAL = $\Gamma \vdash u[e/x] : \sigma$
$\Gamma \vdash u_1(u_2)[e/x] : \sigma$

$$\dfrac{\dfrac{\dfrac{\vdots \text{ASS} \qquad \vdots \text{SHAPE1}}{\Gamma \vdash e : \tau \qquad \Gamma, x:\tau \vdash u_1 : \tau_1 \to \sigma}\ \text{IH2} \qquad \dfrac{\vdots \text{ASS} \qquad \vdots \text{SHAPE2}}{\Gamma \vdash e : \tau \qquad \Gamma, x:\tau \vdash u_2 : \tau_1}\ \text{IH2}}{\Gamma \vdash u_1[e/x](u_2[e/x]) : \sigma}}{\Gamma \vdash u_1(u_2)[e/x] : \sigma}\ \text{subst}$$

$\square$ APP

$\square$ CLAIM 1 (Subst)

CLAIM 3 (canonical forms):

If $\vdash e : \sigma \to \tau$ and e val

then $e = \lambda x:\sigma.u$ (1) with $x:\sigma \vdash u : \tau$ (2)

This is another just where the name says

Proof by inspection.

$$\text{VAL-LAM}\ \frac{}{\lambda x:\tau.e\ \text{val}}$$

A function is a value. This gives us the first part of our conclusion (1)

$$\text{LAM}\ \frac{\Gamma, x:\sigma \vdash e : \tau}{\Gamma \vdash \lambda x:\sigma.e : \sigma \to \tau}$$

In order to conclude that $\Gamma \vdash \lambda x:\sigma.e : \sigma \to \tau$ we must have had $\Gamma, x:\sigma \vdash e : \tau$. This gives us (2).

CLAIM 4 (progress):

If $\vdash e : \tau$ then either e val or $e \mapsto e'$ (for some e')

Having done this before we again know this to be a proof by induction on our antecedent $\vdash e : \tau$

Proof by induction on $\vdash e : \tau$

[Case: LAM]

$\Rightarrow \vdash e : \tau$ has shape $\vdash \lambda x:\sigma.e : \sigma \to \tau$ and derivation

$$\text{LAM}\ \frac{\vdots\ \text{SHAPE}\quad x:\sigma \vdash e : \tau}{\vdash \lambda x:\sigma.e : \sigma \to \tau}$$

IH = if $x:\tau \vdash e:\tau$ then either e val or $e \to e'$ for some e'

GOAL = either $\lambda x:\tau.e$ val or $\exists e'$ s.t. $\lambda x:\tau.e \to e'$

The goal can actually just be proved using VAL-LAM, which has no premises. Still good to write out the IH and shape inversion needed then

$$\frac{}{\lambda x:\tau.e \text{ val}}\ \text{VAL-LAM}$$

$\square$ LAM

[case: APP]

$\Rightarrow \vdash e : \tau$ has shape $\vdash e_1(e_2):\tau$

and derivation

$$\frac{\overset{:\text{SHAPE 1}}{\vdash e_1 : \tau \to \tau} \qquad \overset{:\text{SHAPE 2}}{\vdash e_2 : \tau}}{\vdash e_1(e_2):\tau}\ \text{APP}$$

IH 1 = if $\vdash e_1 : \tau \to \tau$ then either e_1 val or $\exists e_1'$ s.t. $e_1 \mapsto e_1'$

IH 2 = if $\vdash e_2 : \tau$ then either e_2 val or $\exists e_2'$ s.t. $e_2 \mapsto e_2'$

GOAL = either $e_1(e_2)$ val
or $\exists e'$ s.t. $(e_1)e_2 \to e'$

This time there is no rule to eval this immediately, so we must use our IHs. To use these we must case split on the different possibilities of their results, proving our goal in each case.

[Subcase: e_1 val]

$e_1 : T \to U$, so by canonical forms, it must be $e_1 = \lambda x{:}T.u$ for some u.

So we have $(\lambda x{:}T.u)\, e_2$ which we either need to show to be a val, or produce a reduct for.

Applications are not values as there is still work to do, so we can show the latter.

Happily the beta rule with no premises matches our case.

$$\text{D-BETA}\ \frac{}{(\lambda x{:}T.u)\, e_2 \longmapsto \underbrace{u[e_2/x]}_{e'}}$$

$\square\ e_1$ val

[Subcase: $\exists e_1'$ s.t. $e_1 \mapsto e_1'$]

Again this can't be a value so we can refer to a appropriate $\mapsto$ rule to find a reduct of $e_1 e_2$.

$$\text{D-APP-1}\ \frac{e_1 \longmapsto e_1'}{e_1(e_2) \longmapsto e_1'(e_2)}$$

$\square$ APP

Using just IH 1 in either case we get our goal $\Rightarrow \square$

$\square\ \exists e_1'$ s.t. $e \to e_1'$

CLAIM 5 (Preservation):

If $\vdash e : \tau$ and $e \mapsto e'$
then $\vdash e' : \tau$

Hopefully you know the drill by now. We induct on $e \mapsto e'$ (that is chosen as it mentions e' making it the most relevant premise)

Proof by induction on $e \mapsto e'$

This will be 3 cases: D-APP-1, D-APP-2 and D-BETA. (VAL-LAM is not a $\mapsto$ judgement)

[Case: D-APP-1]

$\Rightarrow$ $e \mapsto e'$ has shape $e_1(e_2) \mapsto e_1'(e_2)$

and derivation

$$\dfrac{\dfrac{\vdots\ \text{SHAPE}}{e_1 \mapsto e_1'}}{e_1(e_2) \mapsto e_1'(e_2)}\ \text{D-APP-1}$$

IH = if $\vdash e_1 : \sigma$ and $e_1 \mapsto e_1'$
then $\vdash e_1' : \sigma$

GOAL = $e_1'(e_2) : \tau$

ASS = $e_1(e_2) : \tau$

By inversion on ASS, we know
$e_1 : \tau_2 \to \tau$ (INV1)
$e_2 : \tau_2$ (INV2)

$$\dfrac{\dfrac{\dfrac{\vdots\ \text{INV1}}{e_1 : \tau_2 \to \tau} \quad \dfrac{\vdots\ \text{SHAPE}}{e_1 \mapsto e_1'}}{e_1' : \tau_2 \to \tau}\ \text{IH} \qquad \dfrac{\vdots\ \text{INV2}}{e_2 : \tau}}{e_1'(e_2) : \tau}\ \text{APP}$$

$\square_{\text{D-APP-1}}$

Alternatively you can start deriving goal, see what is needed and try find your derivation ingredients. I just like depth first and the routine nature of
- case
- unto shape
- ASSs/IHs
- Goal.

I really am in the habit of just unrolling/recentering all helpful things before trying to derive (goal).

and having done that the derivation of the goal just falls out.

[Case: D-BETA]

$\Rightarrow e \to e'$ has shape $(\lambda x{:}T.\, e_1)\, e_2 \mapsto e_1[e_2/x]$

and derivation

$$\frac{}{(\lambda x{:}T.\, e_1)\, e_2 \mapsto e_1[e_2/x]}\ \text{D-BETA}$$

ASS(antecedent) $= \vdash (\lambda x{:}T.\, e_1)\, e_2 : G$

GOAL $= \vdash e_1[e_2/x] : G$

This looks very like the substitution lemma.

By the substitution lemma to conclude

$$\vdash e_1[e_2/x] : G$$

(need):

- $\vdash e_2 : T$ (OBLIGATION 1)
- $x{:}T \vdash e_1 : G$ (OBLIGATION 2)

These can both be discharged by inversion on ASS.

$$\text{LAM}\ \frac{x{:}T \vdash e_1 : G}{\vdash (\lambda x{:}T.\, e_1) : T \to G} \qquad \vdash e_2 : T \ \text{APP}$$

$$\vdash (\lambda x{:}T.\, e_1)\, e_2 : G$$

$\square$ D-BETA

$\square$ CLAIMS (preservation)