HOT Compilation Notes

Rahul Manne
rmanne@andrew.cmu.edu

Disclaimer/README

These are only reference notes, and by no means fully capture what is taught in class.

Notes for 170131 (on substitution) are extremely incoherent so I did not include them by default.

There may be errors, feel free to report them to me.

1 Compiler Structure

SML
$\xrightarrow{\text{elaborate}}$ IL-Module
$\xrightarrow{\text{phase-splitting}}$ IL-Direct
$\xrightarrow{\text{cps conversion}}$ IL-CPS
$\xrightarrow{\text{closure conversion}}$ IL-Closure
$\xrightarrow{\text{hoisting}}$ IL-Hoist
$\xrightarrow{\text{allocation}}$ IL-Alloc
$\xrightarrow{\text{code-generation}}$ C

2 Introduction to the $F\omega$ type system

2.1 Grammar

$$
\begin{aligned}
k &::= \mathrm{T} \mid k \to k \\
c &::= \alpha \mid c \to c \mid \forall \alpha : k \,.\, c \mid c\ c \\
e &::= x \mid \lambda x : c \,.\, e \mid e\ e \mid \Lambda \alpha : k \,.\, e \mid e[c]
\end{aligned}
$$

Kind k, Type Constructor c, and Term e.

Note: T is often referred to with just "T" (by Crary), for simplicity.

2.2 Context for Judgements

$$\Gamma ::= \epsilon \mid \Gamma, x : \tau \mid \Gamma, \alpha : k \tag{1}$$

Note: For simplicity, whenever a new α appears in the context, we implicitly ensure that α is not already in Γ.

2.3 $\Gamma \vdash c : k$

$$\frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha : k} \qquad \frac{\Gamma \vdash \tau : T \qquad \Gamma \vdash \tau_2 : T}{\Gamma \vdash \tau_1 \to \tau_2 : T} \qquad \frac{\Gamma, \alpha : k \vdash \tau : T}{\Gamma \vdash \forall \alpha : k \,.\, \tau} \qquad \frac{\Gamma, \alpha : k \vdash c : k'}{\Gamma \vdash \lambda \alpha : k \,.\, c : k \to k'}$$

$$\frac{\Gamma \vdash c_1 : k \to k' \qquad \Gamma \vdash c_2 : k}{\Gamma \vdash c_1\ c_2 : k'}$$

2.4 $\Gamma \vdash e : \tau$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau} \qquad \frac{\Gamma, x : \tau \vdash e : \tau'}{\Gamma \vdash \lambda x : \tau \,.\, e : \tau \to \tau'} \qquad \frac{\Gamma \vdash e_1 : \tau \to \tau' \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1\ e_2 : \tau'} \qquad \frac{\Gamma, \alpha : k \vdash e : \tau}{\Gamma \vdash \Lambda \alpha : k \,.\, e : \forall \alpha : k \,.\, \tau}$$

$$\frac{\Gamma \vdash e : \forall \alpha : k \,.\, e \qquad \Gamma \vdash c : k}{\Gamma \vdash e[c] : [c/\alpha]e} \qquad \frac{\Gamma \vdash e : \tau \qquad \Gamma \vdash \tau \equiv \tau' : T}{\Gamma \vdash e : \tau'}$$

2.5 $\Gamma \vdash c \equiv c : k$

Definitional Equivalence.

$$\frac{\Gamma \vdash c : k}{\Gamma \vdash c \equiv c : k} \qquad \frac{\Gamma \vdash c \equiv c' : k}{\Gamma \vdash c' \equiv c : k} \qquad \frac{\Gamma \vdash c_1 \equiv c_2 : k \qquad \Gamma \vdash c_2 \equiv c_2 : k}{\Gamma \vdash c_1 \equiv c_3 : k}$$

The above are identity, reflexivity, and transitivity respectively.

The following are "compatibility" rules.

$$\frac{\Gamma \vdash c_1 \equiv c_1' : k \qquad \Gamma \vdash c_2 \equiv c_2' : k}{\Gamma \vdash c_1 c_2 \equiv c_1' c_2' : k} \qquad \frac{\Gamma, \alpha : k_1 \vdash c \equiv c' : k_2}{\Gamma \vdash \lambda\alpha : k_1 \,.\, c \equiv \lambda\alpha : k_1 \,.\, c' : k_1 \to k_2}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_1' : k \qquad \Gamma \vdash \tau_2 \equiv \tau_2' : k}{\Gamma \vdash \tau_1 \to \tau_2 \equiv \tau_1' \to \tau_2' : T} \qquad \frac{\Gamma, \alpha : k \vdash \tau \equiv \tau' : T}{\Gamma \vdash \forall\alpha : k \,.\, \tau \equiv \forall\alpha : k \,.\, \tau' : T}$$

congruence = compatible equivalence relation
The following are the rules for beta equivalence and extensionality:

$$\frac{\Gamma \vdash c_2 : k \qquad \Gamma, \alpha : k \vdash c_1 : k'}{\Gamma \vdash (\lambda\alpha : k \,.\, c_1)\ c_2 \equiv [c_2/\alpha]c_1 : k'} \qquad \frac{\Gamma, \alpha : k_1 \vdash c\ \alpha \equiv c'\ \alpha : k_2 \qquad \Gamma \vdash c : k_1 \to k_2 \qquad \Gamma \vdash c' : k_1 \to k_2}{\Gamma \vdash c \equiv c' : k_1 \to k_2}$$

2.6 Extending $F\omega$

Note: This helps in the understanding of sml's module system

Grammar:

$$k ::= \ldots \mid k \times k$$
$$c ::= \ldots \mid \langle c, c\rangle \mid \pi_1 c \mid \pi_2 c$$

New Judgements:

$$\frac{\Gamma \vdash c_1 : k_2 \qquad \Gamma \vdash c_2 : k_2}{\Gamma \vdash \langle c_1, c_2\rangle : k_1 \times k_2} \qquad \frac{\Gamma \vdash c : k_1 \times k_2}{\Gamma \vdash \pi_i\ c : k_1} \qquad \frac{\Gamma \vdash c_1 \equiv c_1' : k_1 \qquad \Gamma \vdash c_2 \equiv c_2' : k_2}{\Gamma \vdash \langle c_1, c_2\rangle \equiv \langle c_1', c_2'\rangle : k_1 \times k_2} \qquad \frac{\Gamma \vdash c \equiv c' : k_1 \times k_2}{\Gamma \vdash \pi_i\ c \equiv \pi_i\ c' : k_i}$$

$$\frac{\Gamma \vdash c_1 : k_1 \qquad \Gamma \vdash c_2 : k_2}{\Gamma \vdash \pi_i\langle c_1, c_2\rangle \equiv c_i : k_i} \qquad \frac{\Gamma \vdash \pi_1\ c \equiv \pi_1\ c' : k_1 \qquad \Gamma \vdash \pi_2\ c \equiv \pi_2\ c' : k_2}{\Gamma \vdash c \equiv c' : k_1 \times k_2}$$

3 Algorithmic Equivalence in the $F\omega$ Type System

3.1 Normalize-and-Compare

Note: We don't use this.

$\lambda\alpha : k \,.\, c_1 \; c_2 \xrightarrow{\beta} [c_2/\alpha]c_1$
$\pi_i\langle c_1, c_2\rangle \xrightarrow{\beta} c_i$
+ some η reduction rules

According to some equivalence theorem, they will have normal forms and those normal forms will be equal if they are equivalent.

3.2 Grammar and Properties

Paths:
$p ::= \alpha \mid p\ c \mid \pi_1\ p \mid \pi_2\ p$

Weak-Head Normal Form:
$n ::= p \mid c_1 \to c_2 \mid \forall\alpha : k \,.\, c.$

Regularity:
If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash c_1 \equiv c_2 : k$, then $\Gamma \vdash c_1 : k$ and $\Gamma \vdash c_2 : k$.
If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash c : k$, then $\Gamma \vdash k : \text{kind}$.

Soundness:
If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash c_1, c_2 : k$ and $\Gamma \vdash c_1 \Leftrightarrow c_2 : k$, then $\vdash c_1 \equiv c_2 : k$.

Completeness:
If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash c_1 \equiv c_2 : k$, then $\Gamma \vdash c_1 \Leftrightarrow c_2 : k$.

$$\frac{}{\vdash \epsilon\,\text{ok}} \qquad \frac{\vdash \Gamma\,\text{ok} \qquad \Gamma \vdash k : \text{kind}}{\vdash \Gamma, \alpha : k\,\text{ok}} \qquad \frac{\vdash \Gamma\,\text{ok} \qquad \Gamma \vdash \tau : \mathrm{T}}{\vdash \Gamma, x : \tau\,\text{ok}}$$

3.3 Algorithmic Constructor Equivalence

Form: $\overset{+}{\Gamma} \vdash \overset{+}{c_1} \Leftrightarrow \overset{+}{c_2} : \overset{+}{k}$

Note: $\overset{+}{x}$ indicates that x is an input.

$$\frac{\Gamma, \alpha : k_1 \vdash c\ \alpha \Leftrightarrow c'\ \alpha : k_2}{\Gamma \vdash c \Leftrightarrow c' : k_1 \to k_2} \qquad \frac{\Gamma \vdash \pi_1\ c \leftrightarrow \pi_1\ c' : k_1 \qquad \Gamma \vdash \pi_2\ c \leftrightarrow \pi_2\ c' : k_2}{\Gamma \vdash c \Leftrightarrow c' : k_1 \times k_2}$$

$$\frac{c_1 \Downarrow c_1' \qquad c_2 \Downarrow c_2' \qquad \Gamma \vdash c_1' \leftrightarrow c_2' : \mathrm{T}}{\Gamma \vdash c_1 \Leftrightarrow c_2 : \mathrm{T}}$$

3.4 Algorithmic Path Equivalence

Form: $\overset{+}{\Gamma} \vdash \overset{+}{c_1} \Leftrightarrow \overset{+}{c_2} : \overset{-}{k}$

Note: $\overset{-}{x}$ indicates that x is an output.

$$\frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha \leftrightarrow \alpha : k} \qquad \frac{\Gamma \vdash p \leftrightarrow p' : k_1 \to k_2 \qquad \Gamma \vdash c \Leftrightarrow c' : k_1}{\Gamma \vdash p\ c \leftrightarrow p'\ c' : k_1} \qquad \frac{\Gamma \vdash p \leftrightarrow p' : k_1 \times k_2}{\Gamma \vdash \pi_i\ p \leftrightarrow \pi_i\ p' : k_i}$$

$$\frac{\Gamma \vdash c_1 \Leftrightarrow c_1' : T \qquad \Gamma \vdash c_1 \Leftrightarrow c_2' : T}{\Gamma \vdash c_1 \to c_2 \leftrightarrow c_1' \to c_2' : T} \qquad \frac{\Gamma, \alpha : k \vdash c \Leftrightarrow c' : T}{\Gamma \vdash \forall \alpha : k\ .\ c \leftrightarrow \forall \alpha : k\ .\ c' : T}$$

3.5 Weak-Head Normalization

Form: $\overset{+}{c} \Downarrow \overset{-}{n}$

$$\frac{c \rightsquigarrow c' \qquad c' \Downarrow c''}{c \Downarrow c''} \qquad \frac{c \not\rightsquigarrow}{c \Downarrow c}$$

3.6 Weak-Head Reduction

Form: $\overset{+}{c} \rightsquigarrow \overset{-}{c'}$

$$\frac{}{(\lambda \alpha : k\ .\ c_1)\ c_2 \rightsquigarrow [c_2/\alpha]c_1} \qquad \frac{}{\pi_i \langle c_1, c_2 \rangle \rightsquigarrow c_i} \qquad \frac{c_1 \rightsquigarrow c_1'}{c_1\ c_2 \rightsquigarrow c_1'\ c_2} \qquad \frac{c \rightsquigarrow c'}{\pi_i c \rightsquigarrow \pi_i c'}$$

3.7 Kind Synthesis and Checking

Form: $\overset{+}{\Gamma} \vdash \overset{+}{c} \Rightarrow \overset{-}{k}$ and $\overset{+}{\Gamma} \vdash \overset{+}{c} \Leftarrow \overset{+}{k}$

$$\frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha \Rightarrow k} \qquad \frac{\Gamma, \alpha : k \vdash c \Rightarrow k'}{\Gamma \vdash \lambda\alpha : k \,.\, c \Rightarrow k \to k'} \qquad \frac{\Gamma \vdash c_1 \Rightarrow k \to k' \qquad \Gamma \vdash c_2 \Leftarrow k}{\Gamma \vdash c_1\ c_2 \Rightarrow k'} \qquad \frac{\Gamma \vdash c_1 \Rightarrow k_1 \qquad \Gamma \vdash c_2 \Rightarrow k_2}{\Gamma \vdash \langle c_1, c_2 \rangle \Rightarrow k_1 \times k_2}$$

$$\frac{\Gamma \vdash c \Rightarrow k_1 \times k_2}{\Gamma \vdash \pi_i\ c \Rightarrow k_1} \qquad \frac{\Gamma \vdash c_1 \Leftarrow T \qquad \Gamma \vdash c_2 \Leftarrow T}{\Gamma \vdash c_1 \to c_2 \Rightarrow T} \qquad \frac{\Gamma, \alpha : k \vdash c \Leftarrow T}{\Gamma \vdash \forall\alpha : k \,.\, c \Rightarrow T} \qquad \frac{\Gamma \vdash c \Rightarrow k}{\Gamma \vdash c \Leftarrow k}$$

3.8 Type Checking and Synthesis

Form: $\overset{+}{\Gamma} \vdash \overset{+}{e} \Rightarrow \overset{-}{c}$ and $\overset{+}{\Gamma} \vdash \overset{+}{e} \Leftarrow \overset{+}{c}$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x \Rightarrow \tau} \qquad \frac{\Gamma \vdash \tau \Leftarrow T \qquad \Gamma, x : \tau \vdash e \Rightarrow \tau'}{\Gamma \vdash \lambda x : \tau \,.\, e \Rightarrow \tau \to \tau'} \qquad \frac{\Gamma \vdash e_1 \Rightarrow \tau_1 \qquad \tau_1 \Downarrow \tau \to \tau' \qquad \Gamma \vdash e_2 \Leftarrow \tau}{\Gamma \vdash e_1\ e_2 \Rightarrow \tau'}$$

$$\frac{\Gamma, \alpha : k \vdash e \Rightarrow \tau}{\Gamma \vdash \Lambda\alpha :.\, e \Rightarrow \forall\alpha : k \,.\, \tau} \qquad \frac{\Gamma \vdash e \Rightarrow \tau \qquad \tau \Downarrow \forall\alpha : k \,.\, \tau' \qquad \Gamma \vdash c \Leftarrow k}{\Gamma \vdash e[c] \Rightarrow [c/\alpha]\tau'} \qquad \frac{\Gamma \vdash e \Rightarrow \tau' \qquad \Gamma \vdash \tau \Leftrightarrow \tau' : T}{\Gamma \vdash e \Leftarrow \tau}$$

4 Singleton Kinds

```
sig
  type t
  type 'a u
  type ('a,'b) v
  type w = int
  type w' = w
  .
  .
  .
end
```

To represent this in type our type system, $t : \mathrm{T}$
$u : \mathrm{T} \to \mathrm{T}$
$v : \mathrm{T} \to \mathrm{T} \to \mathrm{T}$
(or $v : \mathrm{T} \times \mathrm{T} \to \mathrm{T}$)
$w : S(\int)$
$w' : S(w)$

4.1 Grammar and Judgements (Attempt 1)

Grammar:

$k ::= \mathrm{T} \mid k \to k \mid k \times k \mid S(c)$
$c ::= \dots$

Judgements:

$$\frac{}{\Gamma \vdash c : S(c)} \qquad \frac{\Gamma \vdash c : S(c)}{\Gamma \vdash c \equiv c' : \mathrm{T}} \qquad \frac{\Gamma \vdash c : \mathrm{T}}{\Gamma \vdash S(c) : \mathrm{kind}}$$

Signature for `list`.

```
sig
  .
  .
  .
  type 'a s = 'a list
  type 'a t
end
```

So we have $t : \mathrm{T} \to \mathrm{T}$.
How do we represent 'a s? Is $s : \mathrm{T} \to S(\alpha)$? But then what's α.

4.2 Dependent Kinds (Grammar)

$k ::= \mathrm{T} \mid \Pi\alpha : k \mathrel{.} k \mid \Sigma\alpha : k \mathrel{.} k \mid S(c)$
$c ::= \ldots$

Note: Π is also known as "dependent product"

Σ is also known as "dependent sum" (but also sometimes as "dependent product").

To avoid confusion, we name Π "dependent function (spaces)".

Now, we have $s : \Pi\alpha : \mathrm{T} \mathrel{.} S(list\, \alpha)$.

New judgements we need to be able to make:
$\Gamma \vdash k : \text{kind}$
$\Gamma \vdash k \equiv k' : kind$
$\Gamma \vdash k \leq k'$
$\Gamma \vdash c : k$
$\Gamma \vdash c \equiv c' : k$
$\Gamma \vdash e : \tau$

Note: $S(\int) \leq \mathrm{T}$

4.3 $\Gamma \vdash k : \text{kind}$

$$\frac{}{\Gamma \vdash \mathrm{T} : \text{kind}} \qquad \frac{\Gamma \vdash c : \mathrm{T}}{\Gamma \vdash S(c) : \text{kind}} \qquad \frac{\Gamma \vdash k_1 : \text{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 : \text{kind}}{\Gamma \vdash \Pi\alpha : k_1 \mathrel{.} k_2 : \text{kind}}$$

$$\frac{\Gamma \vdash k_1 : \text{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 : \text{kind}}{\Gamma \vdash \Sigma\alpha : k_1 \mathrel{.} k_2 : \text{kind}}$$

4.4 $\Gamma \vdash k \equiv k' : \text{kind}$

$$\frac{\Gamma \vdash k : \text{kind}}{\Gamma \vdash k \equiv k : \text{kind}} \qquad \frac{\Gamma \vdash k_1 \equiv k_2 : \text{kind}}{\Gamma \vdash k_2 \equiv k_1 : \text{kind}} \qquad \frac{\Gamma \vdash k_1 \equiv k_2 : \text{kind} \qquad \Gamma \vdash k_2 \equiv k_2 : \text{kind}}{\Gamma \vdash k_1 \equiv k_2 : \text{kind}} \qquad \frac{\Gamma \vdash c \equiv c' : \mathrm{T}}{\Gamma \vdash S(c) \equiv S(c') : \text{kind}}$$

$$\frac{\Gamma \vdash k_1 \equiv k_1' : \text{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 \equiv k_2' : \text{kind}}{\Gamma \vdash \Pi\alpha : k_1 \mathrel{.} k_2 \equiv \Pi\alpha : k_1' \mathrel{.} k_2' : \text{kind}} \qquad \frac{\Gamma \vdash k_1 \equiv k_1' : \text{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 \equiv k_2' : \text{kind}}{\Gamma \vdash \Sigma\alpha : k_1 \mathrel{.} k_2 \equiv \Sigma\alpha : k_1' \mathrel{.} k_2' : \text{kind}}$$

Note: for the latter two, keep $\Pi\alpha : k_1 \mathrel{.} k_2 \overset{?}{\equiv} \Pi\alpha' : k_1' \mathrel{.} k_2'$ in mind

4.5 $\Gamma \vdash \alpha : k$

$$\frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha : k} \qquad \frac{\Gamma \vdash c_1 : \mathrm{T} \qquad \Gamma \vdash c_2 : \mathrm{T}}{\Gamma \vdash c_1 \to c_2 : \mathrm{T}} \qquad \frac{\Gamma \vdash k : kid \qquad \Gamma, \alpha : k \vdash c : \mathrm{T}}{\Gamma \vdash \forall\alpha : k \mathrel{.} c : \mathrm{T}} \qquad \frac{\Gamma \vdash k_1 : \text{kind} \qquad \Gamma, \alpha : k_1 \vdash c : k_2}{\Gamma \vdash \lambda\alpha : k_1 \mathrel{.} c : \Pi\alpha : k_1 \mathrel{.} k_2}$$

$$\frac{\Gamma \vdash c_1 : \Pi\alpha : k \mathrel{.} k' \qquad \Gamma \vdash c_2 : k}{\Gamma \vdash c_1\ c_2 : [c_1/\alpha]k'} \qquad \frac{\Gamma \vdash c_1 : k_2 \qquad \Gamma \vdash c_2 : [c_1/\alpha]k_2 \qquad \Gamma, \alpha : k_1 \vdash k_2 : \text{kind}}{\Gamma \vdash \langle c_1, c_2 \rangle : \Sigma\alpha : k_2 \mathrel{.} k_2}$$

$$\frac{\Gamma \vdash c : \Sigma\alpha : k_1 \mathrel{.} k_2}{\Gamma \vdash \pi_1 c : k_1} \qquad \frac{\Gamma \vdash c : \Sigma\alpha : k_1 \mathrel{.} k_2}{\Gamma \vdash \pi_2 c : [\pi_1 c/\alpha]k_2} \qquad \frac{\Gamma \vdash c : k \qquad \Gamma \vdash k \leq k'}{\Gamma \vdash c : k'}$$

Additional Judgements If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash c : k$, then $\Gamma \vdash k : \text{kind}$.
If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash k_1 \equiv k_2$, then $\Gamma \vdash k_1, k_2\,\text{kind}$.
If $\vdash \Gamma\,\text{ok}$ and $\Gamma \vdash k_1 \leq k_2$, then $\Gamma \vdash k_1, k_2\,\text{kind}$.

$$\frac{\Gamma \vdash c : \mathrm{T}}{\Gamma \vdash c : S(c)}$$

Sub-kinding:

$$\frac{\Gamma \vdash k \equiv k' : \text{kind}}{\Gamma \vdash k \leq k'} \qquad \frac{\Gamma \vdash k_1 \leq k_2 \qquad \Gamma \vdash k_2 \leq k_3}{\Gamma \vdash k_1 \leq k_3} \qquad \frac{\Gamma \vdash c : \mathrm{T}}{\Gamma \vdash S(c) \leq \mathrm{T}} \qquad \frac{\Gamma \vdash c \equiv c' : \mathrm{T}}{\Gamma \vdash S(c) \leq S(c')}$$

$$\frac{\Gamma \vdash k_1' \leq k_1 \qquad \Gamma, \alpha : k_1' \vdash k_2 \leq k_2' \qquad \Gamma, \alpha : k_1 \vdash k_2 : \text{kind}}{\Gamma \vdash \Pi\alpha : k_1 \,.\, k_2 \leq \Pi\alpha : k_1' \,.\, k_2'}$$

$$\frac{\Gamma \vdash k_1 \leq k_1' \qquad \Gamma, \alpha : k_1 \vdash k_2 \leq k_2' \qquad \Gamma, \alpha : k_1' \vdash k_2' : \text{kind}}{\Gamma \vdash \Sigma\alpha : k_1 \,.\, k_2 \leq \Sigma\alpha : k_1' \,.\, k_2'}$$

Note: Something about contravariance for 1st condition. Π contravariant the same way arrow is contravariant. Covariance for 2nd condition. This is for Π.

4.6 $\Gamma \vdash c \equiv c : k$

$$\frac{\Gamma \vdash c : k}{\Gamma \vdash c \equiv c : k} \qquad \frac{\Gamma \vdash c_1 \equiv c_2 : k}{\Gamma \vdash c_2 \equiv c_1 : k} \qquad \frac{\Gamma \vdash c_1 \equiv c_2 : k \qquad \Gamma \vdash c_2 \equiv c_3 : k}{\Gamma \vdash c_1 \equiv c_3 : k}$$

$$\frac{\Gamma \vdash c_2 : k \qquad \Gamma, \alpha : k \vdash c_1 : k'}{\Gamma \vdash (\lambda\alpha : k \,.\, c_1)\ c_2 \equiv [c_2/\alpha]c_1 : [c_2/\alpha]k'} \qquad \frac{\Gamma \vdash c_1 : k_1 \qquad \Gamma \vdash c_2 : k_2}{\Gamma \vdash \pi_i\langle c_1, c_2\rangle \equiv c_i : k_i} \qquad \frac{\Gamma \vdash c : S(c')}{\Gamma \vdash c \equiv c' : S(c')}$$

$$\frac{\Gamma \vdash c_1 \equiv c_2 : k \qquad \Gamma \vdash k \leq k'}{\Gamma \vdash c_1 \equiv c_2 : k'}\,\star \qquad \frac{\Gamma \vdash c \equiv c' : \mathrm{T}}{\Gamma \vdash c \equiv c' : S(c)}\,\star \qquad \frac{\Gamma \vdash k_1 \equiv k_1' : \text{kind} \qquad \Gamma, \alpha : k_1 \vdash c \equiv c' : k_2}{\Gamma \vdash \lambda\alpha : k_1 \,.\, c \equiv \lambda\alpha : k_1' \,.\, c' : \Pi\alpha : k_1 \,.\, k_2}$$

$$\frac{\Gamma \vdash c_1 \equiv c_1' : \Pi\alpha : k \,.\, k' \qquad \Gamma \vdash c_2 \equiv c_2' : k}{\Gamma \vdash c_1\ c_2 \equiv c_1'\ c_2' : [c_2/\alpha]k'}$$

5 Sub-Typing

$\tau \leq \tau'$ means you can use a τ whrever a τ' is expected.

$$\frac{\Gamma \vdash e : \tau \qquad \tau \leq \tau'}{\Gamma \vdash e : \tau'} \qquad\qquad \frac{\tau_1 \leq \tau_1' \qquad \tau_2 \leq \tau_2'}{\Gamma \vdash \tau \times \tau_2 \leq \tau_1' \times \tau_2'} \qquad\qquad \frac{\tau_1' \leq \tau_1 \qquad \tau_2 \leq \tau_2'}{\tau_1 \to \tau_2 \leq \tau_1' \to \tau_2'}$$

Note 1: This is a case of covariance on both sides.
Note 2: This is contravariant on the left and covariant on the right.

$\mathcal{N} \leq \mathcal{R}$.
Assume we have $f : \mathcal{N} \to \mathcal{N}$.
$f \not: \mathcal{R} \to \mathcal{R}$.

Assume we have $f : \mathcal{R} \to \mathcal{R}$.
Contravariance: $f : \mathcal{N} \to \mathcal{R}$.

$$\frac{\tau \equiv \tau' : \mathrm{T}}{\mathrm{ref}(\tau) \leq \mathrm{ref}(\tau')}$$

ref(τ) is neither covariant nor contravariant. Called “invariant”. (Poorly named, but it’s what’s used in literature.)

$$\frac{\Gamma \vdash k_1' \leq k_1 \qquad \Gamma, \alpha : k_1' \vdash k_2 \leq k_2' \qquad \Gamma, \alpha : k_1 \vdash k_2 : kind}{\Gamma \vdash \Pi\alpha : k_1 \,.\, k_2 \leq \Pi\alpha : k_1' \,.\, k_2'}$$

$$\frac{\Gamma \vdash k_1' \leq k_1 \qquad \Gamma, \alpha : k_1' \vdash k_2 \leq k_2' \qquad \Gamma, \alpha : k_1 \vdash k_2 : \mathrm{kind}}{\Gamma \vdash \Sigma\alpha : k_1 \,.\, k_2 \leq \Sigma\alpha : k_1' \,.\, k_2'} \qquad \frac{\Gamma \vdash c : S(c')}{\Gamma \vdash c \equiv c' : S(c')} \qquad \frac{\Gamma \vdash c : S(c')}{\Gamma \vdash c \equiv c' : \mathrm{T}}$$

$$\frac{\Gamma \vdash c \equiv c' : \mathrm{T}}{\Gamma \vdash c : c' : S(c)}$$

Note 1: Sound, but not what we want.

More compatiblity rules.

$$\frac{\Gamma \vdash c_1 \equiv c_1' : k_1 \qquad \Gamma \vdash c_2 \equiv c_2' : [c_1/\alpha]k_2 \qquad \Gamma, \alpha : k_1 \vdash k_2 : \mathrm{kind}}{\Gamma \vdash \langle c_1, c_2 \rangle \equiv \langle c_1', c_2' \rangle : \Sigma\alpha : k_1 \,.\, k_2} \qquad \frac{\Gamma \vdash c \equiv c' : \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_1 c \equiv \pi_1 c' : k_1}$$

$$\frac{\Gamma \vdash c \equiv c' : \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_2 c \equiv \pi_2 c' : [\pi_1 c/\alpha]k_2} \qquad \frac{\Gamma \vdash c_1 \equiv c_1' : \mathrm{T} \qquad \Gamma \vdash c_2 \equiv c_2' : \mathrm{T}}{\Gamma \vdash c_1 \to c_2 \equiv c_1' \to c_2' : \mathrm{T}} \qquad \frac{\Gamma \vdash k \equiv k' : \mathrm{kind} \qquad \Gamma, \alpha : k \vdash c \equiv c' : \mathrm{T}}{\Gamma \vdash \forall\alpha : k \,.\, c \equiv \forall\alpha : k' \,.\, c' : \mathrm{T}}$$

Rules for extentionality.

$$\frac{\Gamma, \alpha : k_1 \vdash c\ \alpha \equiv c'\ \alpha : k_2 \qquad \Gamma \vdash c : \Pi alpha : k_1 \,.\, k_2' \qquad \Gamma \vdash c' : \Pi alpha : k_1 \,.\, k_2''}{\Gamma \vdash c \equiv c' : \Pi alpha : k_1 \,.\, k_2}$$

$$\frac{\Gamma, \alpha : k_1 \vdash c\ \alpha \equiv c'\ \alpha : k_2 \qquad \Gamma \vdash c \equiv c' : \Pi alpha : k_1 \,.\, k_2'}{\Gamma \vdash c \equiv c' : \Pi\alpha : k_1 \,.\, k_2}$$

$$\frac{\Gamma \vdash \pi_1 c \equiv \pi_1 c' : k_1 \qquad \Gamma \vdash \pi_2 c \equiv \pi_2 c' : [\pi_1 c/\alpha]k_2 \qquad \Gamma, \alpha : k_1 \vdash k_2 : \mathrm{kind}}{\Gamma \vdash c \equiv c' : \Sigma alpha : k_1 \,.\, k_2}$$

Note 1: We only need this for proofs (regularity). We can safely ignore this.

We have no way of dealing with $S(c : k)$. So instead of redefining everything, treat it as a macro following the following rules:
$S(c : \mathrm{T}) = S(c)$
$S(c : \Pi\alpha : k_1 \,.\, k_2) = \Pi\alpha : k_1 \,.\, S(c\ \alpha : k_2)$
$S(c : S(c')) = S(c)$ (note here, $c \equiv c'$, so we can use either, but it's easier for us to use c)
$S(c : \Pi\alpha : k_1 \,.\, k_2) = \Sigma\alpha : S(\pi_1 c : k_1) \,.\, S(\pi_2 c : k_2)$
OR $S(\pi_1 c : k_1) \times S(\pi_2 c : [\pi_1 c/\alpha]k_2)$
We use the 2nd because it's nicer when not working without theory. The first is more theoretic, the second is syntactic.

1. If $\Gamma \vdash c : k$, then $\Gamma \vdash c : S(c : k)$
2. If $\Gamma \vdash c : S(c' : k)$, then $\Gamma \vdash c \equiv c' : k$

But the first doesn't hold. So let's make it hold. Add "declarative" rules:

$$\frac{\Gamma \vdash k_1 : \mathrm{kind} \qquad \Gamma, \alpha : k_1 \vdash c\ \alpha : k_2}{\Gamma \vdash c : \Pi\alpha : k_1 \,.\, k_2} \qquad \frac{\Gamma \vdash \pi_1 c : k_2 \qquad \Gamma \vdash \pi_1 c : [\pi_1 c/\alpha]k_2 \qquad \Gamma, \alpha : k_1 \vdash k_2 : \mathrm{kind}}{\Gamma \vdash c : \Sigma\alpha : k_1 \,.\, k_2}$$

Notes on definitional equivalence:
$\alpha : \mathrm{T} \vdash \alpha \not\equiv \mathrm{int} : \mathrm{T}$
$\alpha : S(\mathrm{int}) \vdash \alpha \equiv \mathrm{int} : \mathrm{T}$
$\vdash \lambda\alpha : \mathrm{T} \,.\, \alpha \not\equiv \lambda\alpha : \mathrm{T} \,.\, \mathrm{int} : \mathrm{T} \to \mathrm{T}$
$\vdash \lambda\alpha : \mathrm{T} \,.\, \alpha \not\equiv \lambda\alpha : \mathrm{T} \,.\, \mathrm{int} : S(\mathrm{int}) \to \mathrm{T}$
$\beta : (\mathrm{T} \to \mathrm{T}) \to \mathrm{T} \vdash \beta(\lambda\alpha : \mathrm{T} \,.\, \alpha \not\equiv \beta(\lambda\alpha : \mathrm{T} \,.\, \mathrm{int} : \mathrm{T}$
$\beta : (S(\mathrm{int}) \to \mathrm{T}) \to \mathrm{T} \vdash \beta(\lambda\alpha : \mathrm{T} \,.\, \alpha \equiv \beta(\lambda\alpha : \mathrm{T} \,.\, \mathrm{int} : \mathrm{T}$
$\mathrm{T} \to \mathrm{T} \leq S(\mathrm{int}) \to \mathrm{T}$

5.1 Algorithm for Equivalence Checking

$$\frac{\Gamma, \alpha : k_1 \vdash c\ \alpha \Leftrightarrow c'\ \alpha : k_2}{\Gamma \vdash c \Leftrightarrow c' : \Pi\alpha : k_1 \,.\, k_2} \qquad \frac{\Gamma \vdash \pi_1 c \Leftrightarrow \pi_2 c' : k_1 \Gamma \vdash \pi_2 c \Leftrightarrow \pi_2 c' : [\pi_1 c/\alpha]k_2}{\Gamma \vdash c \Leftrightarrow c' : \Sigma\alpha : k_1 \,.\, k_2}$$

$$\frac{\Gamma \vdash c_1 \Downarrow c_1' \qquad \Gamma \vdash c_2 \Downarrow c_2' \qquad \Gamma \vdash c_1' \leftrightarrow c_2' : \mathrm{T}}{\Gamma \vdash c_1 \Leftrightarrow c_2 : \mathrm{T}} \qquad \frac{\Gamma \vdash c \rightsquigarrow c' \qquad \Gamma \vdash c' \Downarrow c''}{\Gamma \vdash c \Downarrow c''} \qquad \frac{\Gamma \vdash c \not\rightsquigarrow}{\Gamma \vdash c \Downarrow c}$$

$$\frac{}{\Gamma \vdash (\lambda\alpha : k \,.\, c_1)\ c_2 \rightsquigarrow [c_2/\alpha]c_1} \qquad \frac{\Gamma \vdash c_1 \rightsquigarrow c_1'}{\Gamma \vdash c_1\ c_2 \rightsquigarrow c_1'\ c_2} \qquad \frac{}{\Gamma \vdash \pi_i\langle c_1, c_2\rangle \rightsquigarrow c_i} \qquad \frac{\Gamma \vdash c \rightsquigarrow c'}{\Gamma \vdash pi_i c \rightsquigarrow pi_i c'}$$

$$\frac{\Gamma \vdash p \uparrow S(c)}{\Gamma \vdash p} \qquad \frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha \uparrow k} \qquad \frac{\Gamma \vdash p \uparrow \Pi\alpha : k_1 \,.\, k_2}{\Gamma \vdash p\ c \uparrow [c/\alpha]k_2} \qquad \frac{\Gamma \vdash p \uparrow \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_1 p \uparrow k_1} \qquad \frac{\Gamma \vdash p \uparrow \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_2 p \uparrow [\pi_1 p/\alpha]k_2}$$

$$\frac{\Gamma \vdash p \uparrow S(c)}{\Gamma \vdash p \rightsquigarrow c}$$

Example:

$$\frac{\ldots \vdash (\lambda\alpha : \mathrm{T}\alpha\alpha \rightsquigarrow \alpha \qquad \dfrac{\dfrac{\alpha : S(\mathrm{int}) \vdash \alpha \uparrow S(\mathrm{int})}{\alpha : S(\mathrm{int}) \vdash \alpha \rightsquigarrow \mathrm{int}} \qquad \dfrac{\ldots \vdash \mathrm{int} \not\rightsquigarrow}{\ldots \vdash \mathrm{int} \Downarrow \mathrm{int}}}{\ldots \vdash \alpha \Downarrow \mathrm{int}}}{\alpha : S(\mathrm{int}) \vdash (\lambda\alpha : \mathrm{T} \,.\, \alpha)\alpha \Downarrow} \qquad \frac{}{\alpha : S(\mathrm{int}) \vdash (\lambda\alpha : \mathrm{Tint})\alpha \Downarrow \mathrm{int}}$$

$$\frac{\dfrac{\dfrac{}{\alpha : S(\mathrm{int}) \vdash \mathrm{int} \leftrightarrow \mathrm{int} : \mathrm{T}}}{\alpha : S(\mathrm{int}) \vdash (\lambda\alpha : \mathrm{T} \,.\, \alpha)\alpha \Leftrightarrow (\lambda\alpha : \mathrm{T} \,.\, \mathrm{int})\alpha \Leftrightarrow}}{\vdash \lambda\alpha : \mathrm{T} \,.\, \alpha \Leftrightarrow \lambda\alpha : \mathrm{T} \,.\, \mathrm{int} : S(\mathrm{int}) \to \mathrm{T}}$$

One final rule:

$$\frac{}{\Gamma \vdash c_1 \Leftrightarrow c_2 : S(c)}$$

The precondition is that both c_1 and c_2 belong to $S(c)$, meaning they are equivalent to c and by transitivity, equivalent to each other.

Some rules that we will never use:

$$\frac{}{\Gamma \vdash c_1 \to c_2 \uparrow \mathrm{T}} \qquad \frac{}{\Gamma \vdash \forall\alpha : k \,.\, c \uparrow \mathrm{T}}$$

Structural rules:

$$\frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha \leftrightarrow \alpha : k} \qquad \frac{\Gamma \vdash p \leftrightarrow p' : \Pi\alpha : k_1 \,.\, k_2 \qquad \Gamma \vdash c \Leftrightarrow c' : k_1}{\Gamma \vdash p\ c \leftrightarrow p'\ c' : [c/\alpha]k_2} \qquad \frac{\Gamma \vdash p \leftrightarrow p' : \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_1 p \leftrightarrow \pi_1 p' : k_1}$$

$$\frac{\Gamma \vdash p \leftrightarrow p' : \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_1 p \leftrightarrow \pi_1 p' : [\pi_1 p/\alpha]k_2} \qquad \frac{\Gamma \vdash c_1 \Leftrightarrow c_1' : \mathrm{T} \qquad \Gamma \vdash c_2 \Leftrightarrow c_2' : \mathrm{T}}{\Gamma \vdash c_1 \to c_2 \leftrightarrow c_1' \to c_2' : \mathrm{T}}$$

$$\frac{\Gamma \vdash k \Leftrightarrow k' : \mathrm{kind} \qquad \Gamma, \alpha : k \vdash c \Leftrightarrow c' : \mathrm{T}}{\Gamma \vdash \forall\alpha : k \,.\, c \leftrightarrow \forall\alpha : k' \,.\, c' : \mathrm{T}}$$

If $\Gamma \vdash c \leftrightarrow c' : k$ then $\Gamma \vdash c \uparrow k$ also $\exists k' \,.\, \Gamma \vdash c' \uparrow k'$ and $\Gamma \vdash k \equiv k' : \mathrm{kind}$

Structural comparison:

$$\frac{}{\Gamma \vdash \mathrm{T} \Leftrightarrow \mathrm{T} : \mathrm{kind}} \quad \frac{\Gamma \vdash c \Leftrightarrow c' : \mathrm{T}}{\Gamma \vdash S(c) \Leftrightarrow S(c') : \mathrm{kind}} \qquad \frac{\Gamma \vdash k_1 \Leftrightarrow k_1' : \mathrm{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 \Leftrightarrow k_2' : \mathrm{kind}}{\Gamma \vdash \Pi\alpha : k_1 \,.\, k_2 \Leftrightarrow \Pi\alpha : k_1' \,.\, k_2'}$$

$$\frac{\Gamma \vdash k_1 \Leftrightarrow k_1' : \mathrm{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 \Leftrightarrow k_2' : \mathrm{kind}}{\Gamma \vdash \Sigma\alpha : k_1 \,.\, k_2 \Leftrightarrow \Sigma\alpha : k_1' \,.\, k_2'}$$

$\Gamma \vdash k \trianglelefteq k'$

$$\frac{}{\Gamma \vdash \mathrm{T} \trianglelefteq \mathrm{T}} \qquad \frac{}{\Gamma \vdash S(c) \trianglelefteq \mathrm{T}} \qquad \frac{\Gamma \vdash c \Leftrightarrow c' : \mathrm{T}}{\Gamma \vdash S(c) \trianglelefteq S(c')} \qquad \frac{\Gamma \vdash k_1' \trianglelefteq k_1 \qquad \Gamma, \alpha : k_1' \vdash k_2 \trianglelefteq k_2'}{\Gamma \vdash \Pi\alpha : k_1 \,.\, k_2 \trianglelefteq \Pi\alpha : k_1' \,.\, k_2'}$$

$$\frac{\Gamma \vdash k_1 \trianglelefteq k_1' \qquad \Gamma, \alpha : k_1 \vdash k_2 \trianglelefteq k_2'}{\Gamma \vdash \Sigma\alpha : k_1 \,.\, k_2 \trianglelefteq \Sigma\alpha : k_1' \,.\, k_2'}$$

$\Gamma \vdash k \Leftarrow \text{kind}$

$$\frac{}{\Gamma \vdash \mathrm{T} \Leftarrow \text{kind}} \qquad \frac{\Gamma \vdash c \Leftarrow \mathrm{T}}{\Gamma \vdash S(c) \Leftarrow \text{kind}} \qquad \frac{\Gamma \vdash k_1 \Leftarrow \text{kind} \qquad \Gamma, \alpha : k_1 \vdash k_2 \Leftarrow \text{kind}}{\Gamma \vdash \Pi\alpha : k_1 \,.\, k_2 \Leftarrow \text{kind}}$$

Suppose $\vdash \Gamma\,\text{ok}$. Then:

Soundness

- If $\Gamma \vdash c_1, c_2 : k$ and $\Gamma \vdash c_1 \Leftrightarrow c_2 : k$ then $\Gamma \vdash c_1 \equiv c_2 : k$
- If $\Gamma \vdash k_1, k_2 : \text{kind}$ and $\Gamma \vdash k_1 \Leftrightarrow k_2 : \text{kind}$ then $\Gamma \vdash k_1 \equiv k_2 : \text{kind}$
- If $\Gamma \vdash k_1, k_2 : \text{kind}$ and $\Gamma \vdash k_1 \trianglelefteq k_2$ then $\Gamma \vdash k_1 \leq k_2$
- If $\Gamma \vdash k \Leftarrow \text{kind}$ then $\Gamma \vdash k : \text{kind}$
- If $\Gamma \vdash c \Rightarrow k$ then $\Gamma \vdash c : k$

Completeness

- If $\Gamma \vdash c_1 \equiv c_2 : k$ then $\Gamma \vdash c_1 \Leftrightarrow c_2 : k$
- If $\Gamma \vdash k_1 \equiv k_2 : \text{kind}$ then $\Gamma \vdash k_1 \Leftrightarrow k_2 : \text{kind}$
- If $\Gamma \vdash k_1 \leq k_2$ then $\Gamma \vdash k_1 \trianglelefteq k_2$
- If $\Gamma \vdash k : \text{kind}$ then $\Gamma \vdash k \Leftarrow \text{kind}$
- If $\Gamma \vdash c : k$ then $\Gamma \vdash c \Rightarrow k'$ and $\Gamma \vdash k' \leq S(c : k)$

TODO: principle type
TODO: principle kind is a subkind of every other kind

Checking principle...
$\Gamma \vdash c \Rightarrow k$

$$\frac{\Gamma \vdash c \Rightarrow k' \qquad \Gamma \vdash k' \trianglelefteq k}{\Gamma \vdash c \Leftarrow k}$$

$$\frac{\Gamma(\alpha) = k}{\Gamma \vdash \alpha \Rightarrow S(\alpha : k)} \qquad \frac{\Gamma \vdash k \Leftarrow \text{kind} \qquad \Gamma, \alpha : k \vdash c \Rightarrow k'}{\Gamma \vdash \lambda\alpha : k \,.\, c \Rightarrow \Pi\alpha : k \,.\, k'} \qquad \frac{\Gamma \vdash c_1 \Rightarrow \Pi\alpha : k \,.\, k' \qquad \Gamma \vdash c_2 \Leftarrow k}{\Gamma \vdash c_1\ c_2 \Rightarrow [c_2/\alpha]k'}$$

$$\frac{\Gamma \vdash c_1 \Rightarrow k_1 \qquad \Gamma \vdash c_2 \Rightarrow k_2}{\Gamma \vdash \langle c_1, c_2 \rangle \Rightarrow k_1 \times k_2} \qquad \frac{\Gamma \vdash c \Rightarrow \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_1 c \Rightarrow k_1} \qquad \frac{\Gamma \vdash c \Rightarrow \Sigma\alpha : k_1 \,.\, k_2}{\Gamma \vdash \pi_2 c \Rightarrow [\pi_1 c/\alpha]k_2}$$

$$\frac{\Gamma \vdash c_1 \Leftarrow \mathrm{T} \qquad \Gamma \vdash c_2 \Leftarrow \mathrm{T}}{\Gamma \vdash c_1 \to c_2 \Rightarrow S(c_1 \to c_2)} \qquad \frac{\Gamma \vdash k \Leftarrow \text{kind} \qquad \Gamma, \alpha : k \vdash c \Leftarrow \mathrm{T}}{\Gamma \vdash \forall\alpha : k \,.\, c \Rightarrow S(\forall\alpha : k \,.\, c)}$$

6 Checking Expressions

$\Gamma \vdash e \Rightarrow \tau$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x \Rightarrow \tau} \qquad \frac{\Gamma \vdash e_1 \Rightarrow \tau_1 \qquad \Gamma \vdash \tau_1 \Downarrow \tau \to \tau' \qquad \Gamma \vdash e_2 \Leftarrow \tau}{\Gamma \vdash e_1\ e_2 \Rightarrow \tau'}$$

7 Type-Directed Translation / Syntax-Directed Translation

A more accurate name: "Typing-derivation-directed translation". We proceed by the analysis of the typing derivation of the rules.
Let's represent the source and target languages in different colors, to indicate that they are different.

Property:
$\Gamma \vdash e : \tau$ if and only if $\exists e\ .\ \Gamma \vdash e : \tau \rightsquigarrow e$.

We also want:
If $\Gamma \vdash e : \tau \rightsquigarrow e$, something like $\Gamma \vdash e : \tau$.
But we have no concept of Γ or τ or its derivations.
Instead:
Property:
If $\Gamma \vdash e : \tau \rightsquigarrow e$ and $\tau \rightsquigarrow \tau$ and $\Gamma \rightsquigarrow \Gamma$, then $\Gamma \vdash e : \tau$.
Why not $\Gamma \vdash \tau : \mathrm{T} \rightsquigarrow \tau$.

Simply, we'll use " If $\Gamma \vdash e : \tau \rightsquigarrow e$ then $\Gamma \vdash e : \tau$

7.1 Coherence

For Terms:
Suppose $\Gamma \vdash e : \tau \rightsquigarrow e$ and $\Gamma \vdash e : \tau \rightsquigarrow e'$.
$\Gamma \vdash e \cong e' : \tau$.
This is too hard to even define, this is left to graduate courses. We aspire to it but it's too much of a pain to actually do.

For Types:
Suppose $\Gamma \vdash c : k \rightsquigarrow c$ and $\Gamma \vdash c : k \rightsquigarrow c'$.
Then,
$\Gamma \vdash c \equiv c' : k$.
This is not an aspiration, we cannot live without this.
The 2nd property above can't even be made without this, but it doesn't have to be kind directed. And instead, we'll just make it syntax directed, which will trivially prove that the two are equivalent.

7.2 Definition of e

$$\begin{aligned}
\alpha &= \alpha \\
\tau_1 \times \tau_2 &= \tau_1 \times \tau_2 \\
\tau_1 \to \tau_2 &= \text{unit} \to \tau_1 \to \tau_2 \\
&\dots \\
\epsilon &= \epsilon \\
\Gamma, x : \tau &= \Gamma, x : \tau \\
\Gamma, \alpha : k &= \Gamma, \alpha : k
\end{aligned}$$

Convoluted example:

$$\frac{\Gamma \vdash \tau : \mathrm{T} \qquad \Gamma, x : \tau \vdash e : \tau \rightsquigarrow e}{\Gamma \vdash \lambda x : \tau \,.\, e : \tau \to \tau' \rightsquigarrow \lambda z : \mathrm{unit} \,.\, \lambda x : \tau \,.\, e}$$

NOTE: the right part (after $\rightsquigarrow$) indicates the need to shift in terms of debruijn indices.

More:

$$\frac{\Gamma \vdash e_1 : \tau \to \tau' \rightsquigarrow {e_1}^{:\tau_1 \to \tau_2 = \mathrm{unit} \to \tau \to \tau'} \qquad \Gamma \vdash e_2 : \tau \rightsquigarrow {e_2}^{:\tau}}{\Gamma \vdash e_1\ e_2 : \tau' \rightsquigarrow e_1 <> e_2}$$

More (only well typed things translate):

$$\frac{\Gamma \vdash e_1 \Rightarrow \tau_1 \rightsquigarrow e_1 \qquad \Gamma \vdash e_1 \Downarrow \tau \to \tau' \qquad \Gamma \vdash e_2 \Rightarrow \tau_2 \rightsquigarrow e_2 \qquad \Gamma \vdash \tau_2 \Leftrightarrow \tau : \mathrm{T}}{\Gamma \vdash e_1\ e_2 \Rightarrow \tau' \rightsquigarrow e_1 <> e_2}$$

7.3 $\Gamma \vdash e : \tau \rightsquigarrow e$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau \rightsquigarrow x} \qquad \frac{\Gamma \vdash e_1 : \tau_1 \rightsquigarrow e_1 \qquad \Gamma \vdash e_2 : \tau_2 \rightsquigarrow e_2}{\Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 \times \tau_2 \rightsquigarrow \langle e_1, e_2 \rangle} \qquad \frac{\Gamma \vdash e : \tau_1 \times \tau_2 \rightsquigarrow e}{\Gamma \vdash \pi_1 e : \tau_1 \rightsquigarrow \pi_1 e}$$

8 More Things

$$\frac{\Gamma \vdash c : 1 \qquad \Gamma \vdash c' : 1}{\Gamma \vdash c \equiv c' : 1} \qquad\qquad \frac{}{\Gamma \vdash * : 1}$$

$$\frac{\Gamma \vdash c : k \qquad \Gamma \vdash e : [c/\alpha]\tau \qquad \Gamma, \alpha : k \vdash \tau : \mathrm{T}}{\Gamma \vdash (\mathtt{pack}[c, e]\, as\, \exists \alpha : k \,.\, \tau) : \exists \alpha : k \,.\, \tau} \qquad \frac{\Gamma \vdash e_1 : \exists \alpha : k \,.\, \tau \qquad \Gamma, \alpha : k, x : \tau \vdash e_2 : \tau' \qquad \Gamma \vdash \tau' : \mathrm{T}}{\Gamma \vdash \mathtt{unpack}[\alpha, x] = e_1 ine_2 : \tau'}$$

$$\frac{\Gamma \vdash \tau : \mathrm{T}}{\Gamma \vdash \mathtt{newtag}[\tau] : \mathtt{tag}t} \qquad\qquad \frac{\Gamma \vdash e_1 : \mathtt{tag} \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash \mathtt{tag}(e_1, e_2) : \mathtt{exn}}$$

$$\frac{\Gamma \vdash e_1 : \mathtt{tag} \qquad \Gamma \vdash e_2 : \mathtt{exn} \qquad \Gamma, x : e \vdash e_3 : \tau' \qquad \Gamma \vdash e_4 : \tau'}{\Gamma \vdash \mathtt{iftag}(e_1, e_2, x \,.\, e_3, e_4) : \tau'}$$

$$\frac{\Gamma \vdash e_1 : \exists \alpha : k \,.\, \tau \qquad \Gamma, \alpha : k, x : \tau \vdash e_2 : \tau' \qquad \Gamma \vdash \tau : \mathrm{T}}{\Gamma \vdash \mathtt{unpack}[\alpha, x] = e_1 ine_2 : \tau'}$$

$$\frac{\Gamma \vdash e_1 \Rightarrow \tau_1 \qquad \Gamma \vdash c : k \qquad \Gamma \vdash e_1 \Downarrow \exists \alpha : k \,.\, \tau \qquad \Gamma, \alpha : k, x : \tau \vdash e_2 \Rightarrow \tau' \qquad \Gamma, \alpha : k \vdash \tau \Leftrightarrow [c/\alpha]\tau' : \mathrm{T}}{\Gamma \vdash \mathtt{unpack}[\alpha, x] = e_1 ine_2 \Rightarrow [c/\alpha]\tau'}$$

9 Continuation-Passing Style (CPS)

- control-flow is explicit
- name all intermediate results
- reify control-flow (continuations) as data

The first two are often called “monadic form” or in literature, “A-normal form” (or by Harper, 2/3 CPS).

9.1 Target Language

Still have k, c, and now we have expressions e (which do not return) and values v.

$k ::= \dots$
$c ::= \cdots \mid \cancel{\tau \to \tau} \mid \cancel{\forall\alpha : k \,.\, \tau} \mid \neg\tau$
$e ::= v\ v \mid \texttt{unpack}[\alpha, x] = vine \mid \texttt{let}\ x = \pi_i v\ \texttt{in}\ e\ \texttt{end} \mid \texttt{let}\ x = v\ \texttt{in}\ e\ \texttt{end} \mid \texttt{halt} \mid \dots v \quad ::= x \mid \lambda x : \tau \,.\, e \mid \texttt{pack}[c, v] as \exists\alpha$

Judgements:
$\Gamma \vdash v : \tau$
$\Gamma \vdash e : 0$
(e does not return, so we use ‘0’ for ‘OK’)

$$\frac{\Gamma \vdash \tau : \mathrm{T} \qquad \Gamma, x : \tau \vdash e : 0}{\Gamma \vdash \lambda x : \tau \,.\, e : \neg\tau} \qquad \frac{\Gamma \vdash v_1 : \neg\tau \qquad \Gamma \vdash v_2 : \tau}{\Gamma \vdash v_1\ v_2 : 0} \qquad \frac{\Gamma \vdash c : k \qquad \Gamma \vdash v : [c/\alpha]\tau \qquad \Gamma, \alpha : k \vdash \tau : \mathrm{T}}{\Gamma \vdash \texttt{pack}[c, v] as \exists\alpha : k \,.\, \tau : \exists\alpha : k \,.\, \tau}$$

$$\frac{\Gamma \vdash v : \exists\alpha : k \,.\, \tau\Gamma, \alpha : k, x : \tau \vdash e : 0}{\Gamma \vdash \texttt{unpack}[\alpha, x] = v \in e : 0} \qquad \frac{\Gamma \vdash v_i : \tau_i \qquad (\forall i \in [n])}{\Gamma \vdash \langle v_1, \dots v_2 \rangle : x[\tau, \dots \tau_n]} \qquad \frac{\Gamma \vdash v : x[\tau_0, \dots \tau_{n-1} \qquad \Gamma, x : \tau_i \vdash e : 0}{\Gamma v \texttt{let}\ x = \pi_i v\ \texttt{in}\ e\ \texttt{end} : 0}$$

$$\frac{\Gamma \vdash v : \tau \qquad \Gamma, x : \tau \vdash e : 0}{\Gamma \vdash \texttt{let}\ x = v\ \texttt{in}\ e\ \texttt{end} : 0} \qquad \frac{}{\Gamma \vdash \texttt{halt} : 0}$$

9.2 Translation

(NOTE: this is syntax-directed)

$$\begin{aligned}
T &= T \\
\Pi\alpha : k_1 \,.\, k_2 &= \Pi\alpha : k_1 \,.\, k_2 \\
S(c) &= S(c) \\
1 &= 1 \\
\alpha &= \alpha \\
\lambda\alpha : k \,.\, c &= \lambda\alpha : k \,.\, c \\
c_1\ c_2 &= c_1\ c_2 \\
\langle c_1, c_2\rangle &= \langle c_1, c_2\rangle \\
\pi_1 c &= \pi_1 c \\
\pi_2 c &= \pi_2 c
\end{aligned}$$

$$\begin{aligned}
\tau_1 \to \tau_2 &= \neg(\tau_1 \times \neg\tau_2) \\
x[\tau_1, \ldots, \tau_n] &= x[\tau_1, \ldots, \tau_n] \\
\forall\alpha : k \,.\, \tau &= \neg(\exists\alpha : k \,.\, \neg\tau) \\
\exists\alpha : k \,.\, \tau &= \exists\alpha : k \,.\, \tau
\end{aligned}$$

$$\begin{aligned}
\epsilon &= \epsilon \\
\Gamma, \alpha : k &= \Gamma, \alpha : k \\
\Gamma, x : \tau &= \Gamma, x : \tau \\
[c_1/\alpha]c_2 &= [c_1/\alpha]c_2 \\
\alpha &= \alpha
\end{aligned}$$

Type directed translation:
Judgement:
$\Gamma \vdash e : \tau \rightsquigarrow x \,.\, e$
Note here that this an expression where we compute the value of e and send it to the continuation x.

Type Principle:
If $\Gamma \vdash e : \tau \rightsquigarrow x \,.\, e$ (and $\vdash \Gamma$ ok) then $\Gamma, k : \neg\tau \vdash e : 0$.

Note: k is not the metavariable for kind in this case.

9.3 $\Gamma \vdash e : \tau \leadsto k\ .\ e$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau \leadsto k : \neg\tau\ .\ kx} \qquad \frac{\Gamma \vdash e : x[\tau_0, \ldots, \tau_{n-1} \leadsto k' : \neg x[\tau_0, \ldots, \tau_{n-1}]\ .\ e}{\Gamma \vdash \pi_i e : \tau_i \leadsto k : \not\tau_i\ .\ \texttt{let}\ k' = \lambda x : *[\tau_0, \ldots, \tau_{n-1}]\ .\ \texttt{let}\ y = \pi_i x\ \texttt{in}\ ky\ \texttt{end}\ \texttt{in}\ e\ \texttt{end}}$$

$$\frac{\Gamma \vdash e_i : \tau_i \leadsto k_i : \neg\tau_i\ .\ e_i \qquad (i = 1 \ldots n)}{\begin{array}{c}\Gamma \vdash \langle e_1, \ldots e_n\rangle : *[\tau_i, \ldots, \tau_n] \leadsto \\ k : \neg x[\tau_1, \ldots, \tau_n\ .\ \texttt{let}\ k_1 = \lambda x_1 : \tau_1\ .\ \texttt{let}\ k_2 = \lambda x_2 : \tau_2\ .\ \ldots \texttt{let}\ k_n = \lambda x_n\ .\ \tau_n k\langle x_1, \ldots, x_n\ \texttt{in}\ e_n\ \texttt{end}\ \texttt{in}\ e_2\ \texttt{end}\ \texttt{in}\ e_1\ \texttt{end}\end{array}}$$

$$\frac{\Gamma \vdash \tau_1 : \mathrm{T} \qquad \Gamma, x : \tau_1 \vdash e : \tau_2 \leadsto k' : \not\tau_2\ .\ e}{\Gamma \vdash \lambda x : \tau_1\ .\ e : \tau_1 \to \tau_2 \leadsto k^{:\not\tau_1 \to \tau_2 = (\not\tau_1 \times \not\tau_2)}\ .\ k(\lambda y : \tau_1 \times \tau_2\ .\ \texttt{let}\ x = \pi_0 y\ \texttt{in}\ \texttt{let}\ k = \pi_1 y\ \texttt{in}\ e\ \texttt{end}\ \texttt{end})}$$

$$\frac{\Gamma \vdash e_1 : \tau \to \tau' \leadsto k^{:\neg\tau \to \tau' = \neg\neg(\tau_1 \times \neg\tau_2)}\ .\ e \qquad \Gamma \vdash e_2 : \tau \leadsto k_2^{:\neg\tau}\ .\ e_2}{\Gamma \vdash e_1\ e_2 : \tau' \leadsto k^{\neg\tau'}\ .\ \texttt{let}\ k_1 = \lambda f : \neg(\tau \times \neg\tau'\ .\ \texttt{let}\ k_2 = \lambda x : \tau\ .\ f\langle x, k\rangle\ \texttt{in}\ e_2\ \texttt{end}\ \texttt{in}\ e_1\ \texttt{end}}$$

$$\frac{\Gamma \vdash c : k \qquad \Gamma \vdash e : [c/\alpha]\tau \leadsto k' : \neg[c/\alpha]\tau\ .\ e \qquad \Gamma, \alpha : k \vdash \tau : \mathrm{T}}{\begin{array}{c}\Gamma \vdash \texttt{pack}[c, e]\,\mathrm{as}\,\exists\alpha : k\ .\ \tau : \exists\alpha : k\ .\ e \leadsto \\ k^{:\neg\exists\alpha:k.\tau = \neg\exists\alpha:k.\tau}\ .\ \texttt{let}\ k' : \lambda x : [c/\alpha]\tau\ .\ k(\texttt{pack}[c, x]\,\mathrm{as}\,\exists\alpha : k\ .\ \tau = \ \texttt{in}\ e\ \texttt{end}\end{array}}$$

$$\frac{\Gamma \vdash e_1 : \exists\alpha : k\ .\ \tau \leadsto \neg\exists\alpha : k\ .\ \tau \qquad \Gamma, \alpha : k, x : \tau \vdash e_2 : \tau' \leadsto k_2^{:\neg\tau'}\ .\ e_2 \qquad \Gamma \vdash \tau : \mathrm{T}}{\Gamma \vdash \texttt{unpack}[\alpha, x] = e_1 \texttt{in} e_2 \leadsto \qquad k^{:\neg\tau'}\ .\ \texttt{let}\ k_1 = \lambda x_1 : \exists\alpha : k\tau\ .\ \texttt{unpack}[\alpha, x] = x_1 \texttt{in}[k/k_2]e_2\ \texttt{in}\ e_1\ \texttt{end}}$$

$$\frac{\Gamma \vdash k : \mathrm{kind} \qquad \Gamma, \alpha : k \vdash e : \tau \leadsto k'^{:\neg\tau}\ .\ e}{\Gamma \vdash \Lambda\alpha : k\ .\ e : \forall\alpha : k\ .\ \tau \leadsto k^{:\neg\forall\alpha:k.\tau = \neg\neg(\exists\alpha:k.\tau)}\ .\ k(\lambda x : \exists\alpha : k\ .\ \neg\tau\ .\ \texttt{unpack}[\alpha, k'] = x \texttt{in} e)}$$

$$\frac{\Gamma \vdash e : \forall\alpha : k\ .\ \tau \rightsquigarrow k'^{:\neg\forall\alpha:k.\tau = \neg\neg(\exists\alpha.k\neg\tau)}\ .\ e \qquad \Gamma \vdash c : k}{\Gamma \vdash e[c] : [c/\alpha]\tau \rightsquigarrow k^{:\neg[c/\alpha]\tau}\ .\ \texttt{let}\ k' = \lambda f : \neg(\exists\alpha : k\ .\ \neg\tau)\ .\ f(\texttt{pack}[c, k]\ \mathrm{as}\ \exists\alpha : k\ .\ \neg\tau)\ \texttt{in}\ e\ \texttt{end}}$$

Note $\neg[c/\alpha]\tau = \neg[c/\alpha]\tau$.

9.4 Exceptions

$\tau_1 \to \tau_2 = \neg(\times[\tau_1, \neg\tau_2, \neg\texttt{exn}]$
$\forall\alpha : k\ .\ \tau = \neg(\exists\alpha : k\ .\ \neg\tau \times \neg\texttt{exn})$

Judgement: $\Gamma \vdash e : \tau \rightsquigarrow k^{:\neg\tau} k_{ex}^{:\neg\texttt{exn}}\ .\ e$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau \rightsquigarrow k k_{ex}\ .\ kx}$$

$$\frac{\Gamma \vdash e_i : \tau_i \rightsquigarrow k_i^{:\neg\tau_1} k_{ex_i}^{:\neg\texttt{exn}}\ .\ e_i \qquad (i = 1 \ldots n)}{\Gamma \vdash \langle e_1, \ldots, e_n\rangle : x[\tau_1, \ldots, \tau_n] \rightsquigarrow k k_{ex}\ .\ \texttt{let}\ k_1 = \lambda x_1 : \tau_1\ .\ \ldots \texttt{let}\ k_n = \lambda x_n : \tau_n\ .\ k\langle x_i, \ldots x_n\rangle\ \texttt{in}\ e_n\ \texttt{end} \ldots\ \texttt{in}\ e_1\ \texttt{end}}$$

$$\frac{\Gamma \vdash e : \mathrm{T} \qquad \Gamma \vdash e : \texttt{exn} \rightsquigarrow k'^{:\neg\texttt{exn}} k_{ex}'^{:\neg\texttt{exn}}\ .\ e}{\Gamma \vdash \texttt{raise}_\tau e : \tau \rightsquigarrow k^{:\neg\tau} k_{ex}^{:\neg\texttt{exn}}\ .\ \texttt{let}\ k' = k_{ex}\ \texttt{in}\ e\ \texttt{end}}$$

$$\frac{\Gamma \vdash e : \tau \rightsquigarrow k^{:\neg\tau} k_{ex1}\ .\ e_1 \qquad \Gamma, x : \texttt{exn} \vdash e_2 : \tau \rightsquigarrow k^{:\neg\tau} k_{ex}\ .\ e_2}{\Gamma \vdash \texttt{handle}(e_1, x\ .\ e_2 : \tau \rightsquigarrow k^{:\neg\tau} k_{ex}^{:\neg\texttt{exn}}\ .\ \texttt{let}\ k_{em} = \lambda x : \texttt{exn}\ .\ e_2\ \texttt{in}\ e_1\ \texttt{end}}$$

$$\frac{\Gamma \vdash \tau_1 : \mathrm{T} \qquad \Gamma, x : \tau_1 \vdash e : \tau_2 \rightsquigarrow k'^{\neg\tau_2} k'_{ex}\ .\ e_2}{\begin{array}{c}\Gamma \vdash \lambda x : \tau_1\ .\ e : \tau_1 \to \tau_2 \rightsquigarrow \\ k^{:\neg\tau_1 \to \tau_2 = \neg\neg(\times[\tau_1, \neg\tau_2, \neg\texttt{exn}])} k_{ex}\ .\ k(\lambda y : x[\tau_1, \neg\tau_2, \neg\texttt{exn}]\ .\ \texttt{let}\ x = \pi_0 y\ \texttt{in}\ \texttt{let}\ k' = \pi_1 y\ \texttt{in}\ \texttt{let}\ k'_{ex} = \pi_2 y\ \texttt{in}\ e\ \texttt{end}\ \texttt{end}\ \texttt{end})\end{array}}$$

$$\frac{\Gamma \vdash e_1 : \tau \to \tau' \rightsquigarrow k_1^{\neg\tau \to \tau' = \neg\neg(\times[\tau, \neg\tau', \neg\texttt{exn}])} k_{ex}\ .\ e_1 \qquad \Gamma \vdash e_2 : \tau \rightsquigarrow k_2^{\neg\tau} k_{ex2}\ .\ e_2}{\Gamma \vdash e_1\ e_2 : \tau' \rightsquigarrow k^{\neg\tau'} k_{ex}\ .\ \texttt{let}\ k_1 = (\lambda f : \neg(x[\tau, \neg\tau', \neg\texttt{exn}])\ .\ \texttt{let}\ k_2 = \lambda x : \tau\ .\ f\langle x, k, k_{ex}\rangle\ \texttt{in}\ e_2\ \texttt{end})\ \texttt{in}\ e_1\ \texttt{end}}$$

9.5 Continuations

```
callcc : ('a cont → 'a) → 'a
throw : ('a cont × 'a) → 'b
```

Typing Rules

$$\frac{\Gamma \vdash \tau : \mathrm{T} \qquad \Gamma, x : \texttt{cont}\ \tau \vdash e : \tau}{\Gamma \vdash \texttt{callcc}_\tau x\ .\ e : \tau} \qquad\qquad \frac{\Gamma \vdash \tau' : \mathrm{T} \qquad \Gamma \vdash e_1 : \tau \qquad \Gamma \vdash e_2 : \texttt{cont}\ \tau}{\Gamma \vdash \texttt{throw}_{\tau'} e_1\ \texttt{to}\ e_2 : \tau'}$$

Translation Rules

$$\frac{\Gamma \vdash \tau : \mathrm{T} \qquad \Gamma, x : \texttt{cont}\ \tau \vdash e : \tau \rightsquigarrow k'^{\neg\tau}\ .\ e}{\Gamma \vdash \texttt{callcc}_\tau x\ .\ e : \tau \rightsquigarrow k^{:\neg\tau}\ .\ \texttt{let}\ k' = k\ \texttt{in}\ \texttt{let}\ x = k\ \texttt{in}\ e\ \texttt{end}\ \texttt{end}}$$

$$\frac{\Gamma \vdash \tau' : \mathrm{T} \qquad \Gamma \vdash e_1 : \tau \rightsquigarrow k_1^{:\neg\tau}\ .\ e_1 \qquad \Gamma \vdash e_2 : \texttt{cont}\ \tau \rightsquigarrow k_2^{:\neg\texttt{cont}\ \tau = \neg\neg\tau}\ .\ e_2}{\Gamma \vdash \texttt{throw}_{\tau'} e_1\ \texttt{to}\ e_2 : \tau' \rightsquigarrow k^{:\neg\tau'}\ .\ \texttt{let}\ k_1 = \lambda x : \tau\ .\ \texttt{let}\ x_2 = \lambda y : \neg\tau\ .\ y\ x\ \texttt{in}\ e_2\ \texttt{end}\ \texttt{in}\ e_1\ \texttt{end}}$$

10 Closure Conversion

A closure is a tuple containing code and the environment that will be passed in as an additional argument to the code. The code is closed, with no open terms.

Concrete example:
$\lambda x : \text{int} . x + y + z$
$\rightsquigarrow \langle(\lambda x : \text{int} . \lambda\texttt{env} : \text{int} \times \text{int} . \texttt{let } y = \pi_0\texttt{env in let } z = \pi_1\texttt{env in } x + y + z \texttt{ end end}), \langle y, z\rangle\rangle$
$e\ 5$
$\rightsquigarrow \texttt{let } f = \pi_0 e \texttt{ in let env} = \pi_1 e \texttt{ in } f\ 5\ \texttt{env end end}$

This can get really messy and really inefficient, so what we really want to do is convert some curried and some tupled functions and turn them into some special internal representation for multi-argument functions.

Types may be different if the environment is different, so when converting, we can try something like, where we don't care about the existential, since we won't be manipulating it:
$\tau_1 \to \tau_2 = \exists\alpha_{\texttt{env}} : \mathrm{T} . (\tau_1 \to \alpha_{\texttt{env}} \to \tau_2) \times \alpha_{\texttt{env}}$

10.1 Target Language (IL-Closure)

$\Delta; \Gamma \vdash e : 0$
$\Delta; \Gamma \vdash v : \tau$
$\Gamma ::= \epsilon \mid \Gamma, x : \tau$
$\Delta ::= \epsilon \mid \Delta, \alpha : k$

Only rule additional we need:

$$\frac{\Delta \vdash \tau : \mathrm{T} \qquad \Gamma; (\epsilon, x : \tau) \vdash e : 0}{\Delta; \Gamma \vdash \lambda x : \tau . e : \neg\tau}$$

10.2 Type Translation

$$\begin{aligned} \alpha &= \alpha \\ &\vdots \\ \neg\tau &= \exists\alpha_{\texttt{env}} : \mathrm{T} . \neg(\tau \times \alpha_{\texttt{env}}) \times \alpha_{\texttt{env}} \\ x[\tau_1, \ldots, \tau_2] &= x[\tau_1, \ldots, \tau_n] \end{aligned}$$

How would we deal with $\forall\alpha : k . \tau$? Turns out it's really hard. But because of how we got rid of them back during CPS conversion, we don't even have to deal with it anymore!

10.3 Type Principle

If $\Delta; \Gamma \vdash e : 0 \rightsquigarrow e$ then $\Delta; \Gamma \vdash e : 0$.
If $\Delta; \Gamma \vdash v : \tau \rightsquigarrow v$ then $\Delta; \Gamma \vdash v : \tau$.

10.4 $\Delta;\Gamma \vdash e : 0 \rightsquigarrow e$, $\Delta;\Gamma \vdash v : \tau \rightsquigarrow v$

$$\frac{\Gamma(x) = \tau}{\Delta;\Gamma \vdash x : \tau \rightsquigarrow x} \qquad \frac{\Delta;\Gamma \vdash v_i : \tau_i \rightsquigarrow v_i}{\Delta;\Gamma \vdash \langle v_1, \dots v_n\rangle : x[\tau_1, \dots, \tau_n] \rightsquigarrow \langle v_1, \dots, v_n\rangle}$$

$$\frac{\Delta \vdash \tau : \mathrm{T} \qquad \Delta;\Gamma, x : \tau \vdash e : 0 \rightsquigarrow e \qquad \Gamma = x_1 : \tau_1, \dots, x_n : \tau_n}{\begin{array}{c}\Delta;\Gamma \vdash \lambda x : \tau \,.\, e : \neg\tau \rightsquigarrow \texttt{pack}[x[\tau_1, \dots, \tau_n], \langle \\ (\lambda y : \tau \times x[\tau_1, \dots, \tau_n] \,.\, \texttt{let}\ x = \pi_0 y\ \texttt{in}\ \texttt{let}\ \texttt{env} = \pi_1 y\ \texttt{in}\ \texttt{let}\ x_1 = \pi_0\texttt{env}\ \texttt{in}\ \dots \texttt{let}\ x_n = \pi_{n-1}\texttt{env}\ \texttt{in}\ e\ \texttt{end end end en} \\ \langle x_1, \dots, x_n\rangle\rangle]\ \text{as}\ \exists\alpha_{\texttt{env}} : \mathrm{T} \,.\, \neg(\tau \times \alpha_{\texttt{env}}) \times \alpha_{\texttt{env}}\end{array}}$$

$$\frac{\Delta;\Gamma \vdash v_1 : \neg\tau \rightsquigarrow v_1{}^{:\neg\tau = \exists\alpha.\neg(\tau\times\alpha)\times\alpha} \qquad \Delta;\Gamma \vdash v_2 : \tau \rightsquigarrow v_2{}^{:\tau}}{\Delta;\Gamma \vdash v_1\ v_2 : 0 \rightsquigarrow \texttt{unpack}[\alpha_{\texttt{env}}, x] = v_1\texttt{inlet}\ f = \pi_0 x\ \texttt{in let env} = \pi_1 x\ \texttt{in}\ f\langle v_2, \texttt{env}\rangle\ \texttt{end end}}$$

To solve some inefficiency, instead of passing around the environment, instead, pass around the closure.
In environment passing, we had $\tau_1 \to \tau_2 = \exists\alpha_{\texttt{env}} : \mathrm{T} \,.\, (\tau_1 \times \alpha_{\texttt{env}} \to \tau_2) \times \alpha_{\texttt{env}}$.
In closure passing, we have $\mu\beta \,.\, \exists\alpha_{\texttt{env}} \,.\, \tau_1 \times \beta \to \tau_2 \times \alpha_{\texttt{env}}$.

We can apparently use this to understand objected oriented programming better and some of the research might not even be all that wrong.

There's more stuff one can do ....

11 Hoisting

Type Translation:
$\tau = \tau$

11.1 Target Language: IL-Hoist

$$\begin{aligned}
c &::= \cdots \mid \forall\alpha : k \,.\, c \\
e &::= \dots \\
v &::= \cdots \mid \lambda x : \cancel{\tau} \,.\, e \mid v[c] \mid \cancel{\Lambda\alpha : k \,.\, v} \\
f &::= \lambda x : \tau \,.\, e \mid \Lambda\alpha : k \,.\, f \\
b &::= x = f \\
P &::= \texttt{let}\ b\ \texttt{in}\ P \mid e
\end{aligned}$$

To hoist the type: ‘type-erasure semantics‘.

11.2 Judgements

$\Delta; \Gamma \vdash e : 0 \leadsto let\vec{b}in e$
$\Delta; \Gamma \vdash v : \tau \leadsto let\vec{b}in v$

11.3 $\Delta; \Gamma \vdash f : \tau$

$$\frac{\Delta, \alpha : k; \Gamma \vdash f : \tau}{\Delta; \Gamma \vdash \Lambda\alpha : k \,.\, f : \forall\alpha : k \,.\, \tau} \qquad \frac{\Delta \vdash \tau : \mathrm{T} \qquad \Delta; \Gamma, x : \tau \vdash e : 0}{\Delta; \Gamma \vdash \lambda x : \tau \,.\, e : \neg\tau}$$

11.4 $\Gamma \vdash P : 0$

$$\frac{\cdot; \Gamma \vdash f : \tau \qquad \Gamma, x : \tau \vdash P : 0}{\Gamma \vdash \texttt{let } x = f \texttt{ in } P \texttt{ end} : 0} \qquad \frac{\cdot; \Gamma \vdash e : 0}{\Gamma \vdash e : 0}$$

11.5 $\Delta; \Gamma \vdash v : \tau \leadsto \texttt{let } \vec{b} \texttt{ in } v \texttt{ end}$

$$\frac{\Gamma(x) = \tau}{\Delta; \Gamma \vdash x : \tau \leadsto \texttt{let \ in } x \texttt{ end}} \qquad \frac{\Delta; \Gamma \vdash v_i : \tau_i \leadsto \texttt{let } \vec{b_i} \texttt{ in } v_i \texttt{ end} \qquad (\mathrm{for} i = 1 \ldots n)}{\Delta; \Gamma \vdash \langle v_1 \ldots v_n \rangle : \times[\tau_1 \ldots \tau_n]\texttt{let } \vec{b_1} \ldots \vec{b_n} \texttt{ in } \langle\langle v_1 \ldots v_n \rangle \texttt{ end}}$$

$$\frac{\Delta; \Gamma \vdash v_1 : \neg\tau \leadsto \texttt{let } \vec{b_1} \texttt{ in } v_1 \texttt{ end} \qquad \Delta; \Gamma \vdash v_2 : \tau \leadsto \texttt{let } \vec{b_2} \texttt{ in } v_2 \texttt{ end}}{\Delta; \Gamma \vdash v_1 v_2 : 0 \leadsto \texttt{let } \vec{b_1}, \vec{b_2} \texttt{ in } v_1 v_2 \texttt{ end}}$$

$$\frac{\Delta \vdash \tau : \mathrm{T} \qquad \Delta; x : \tau \vdash e : 0 \leadsto \texttt{let } \vec{b} \texttt{ in } e \texttt{ end} \qquad y \notin FV(\Gamma), y \neq x, y \notin BV(\vec{b}) \qquad \Delta = \alpha_1 : k_1, \ldots, \alpha_n : k_n}{\Delta; \Gamma \vdash \lambda x : \tau \,.\, e : \neg\tau \leadsto \texttt{let } \vec{b}, y = \Lambda\alpha_1 : k_1 \,.\, \ldots \Lambda\alpha_n : k_n \,.\, \lambda x : \tau \,.\, e \texttt{ in } y[\alpha_1] \ldots [\alpha_n] \texttt{ end}}$$

11.6 $\Delta; \Gamma \vdash e : 0 \leadsto \texttt{let } \vec{b} \texttt{ in } v \texttt{ end}$

$$\frac{\cdot; \cdot \vdash e : 0 \leadsto \texttt{let } \vec{b} \texttt{ in } e \texttt{ end} \qquad (\vec{b} = b_1, \ldots, b_n)}{\vdash_{\texttt{top}} e : 0 \leadsto \texttt{let } b_1 \texttt{ in } \ldots \texttt{let } b_n \texttt{ in } e \texttt{ end end}}$$

TODO: ASIDE PREVIOUS: CLOSURE CONVERSION

$$\frac{\cdot; \cdot \vdash e : 0 \leadsto e}{\vdash_{\texttt{top}} e : 0 \leadsto e}$$

TODO: ASIDE PREVIOUS: CPS CONVERSION

$$\frac{\cdot; \cdot \vdash e : \tau \leadsto k k_{ex} \,.\, e}{\vdash_{\texttt{top}} e : \tau \leadsto \texttt{let } k = \lambda x : \tau \,.\, \texttt{halt in let } k_{ex} = \lambda x : \texttt{exn} \,.\, \texttt{let } \_ = \texttt{print "uncaught exception" in halt end in } e \texttt{ end end}}$$

12 Alloc

12.1 Target Language: IL-Alloc

$$\begin{aligned}
a &::= x = \texttt{alloc}[n] \mid x = \pi_i y \mid \pi_i y ::- v \mid x = v \\
e &::= \texttt{let}\ a\ \texttt{in}\ e\ \texttt{end} \mid x\ x \mid \texttt{halt} \\
v &::= x \mid \cdots \mid \cancel{\langle v_1 \ldots v_n \rangle} \\
f &::= \lambda x : \tau\ .\ e \\
b &::= x = f \\
P &::= \texttt{let}\ b\ \texttt{in}\ P \mid e
\end{aligned}$$

12.2 Judgements and Translations

$\Delta; \Gamma \vdash e : 0 \rightsquigarrow e$
$\Delta; \Gamma \vdash v : \tau \rightsquigarrow \texttt{let}\ \vec{a}\ \texttt{in}\ v\ \texttt{end}$

12.3 $\Delta; \Gamma \vdash v : \tau \rightsquigarrow \texttt{let}\ \vec{a}\ \texttt{in}\ v\ \texttt{end}$

$$\frac{\Delta; \Gamma \vdash v_i : \tau_i \rightsquigarrow \texttt{let}\ \vec{a_i}\ \texttt{in}\ v_i\ \texttt{end} \qquad (\text{for } i = 1 \ldots n) \qquad \texttt{fresh}[x]}{\Delta; \Gamma \vdash \langle v_1 \ldots v_n \rangle : \times[\tau_1 \ldots \tau_n] \rightsquigarrow \texttt{let}\ \vec{a_1} \ldots \vec{a_n}, x = \texttt{alloc}[n], \pi_0 x = v_1 \ldots, \pi_{n-1} x = v_n\ \texttt{in}\ x\ \texttt{end}}$$

12.4 $\Delta; \Gamma \vdash e : 0 \rightsquigarrow e$

$$\frac{\Delta; \Gamma \vdash v : \tau \rightsquigarrow \texttt{let}\ \vec{a}\ \texttt{in}\ i\ \texttt{end} nv \qquad \Delta; \Gamma, x : \tau \vdash e : 0 \rightsquigarrow e \qquad \vec{a} = a_1, \ldots, a_n}{\Delta; \Gamma \vdash \texttt{let}\ x = v\ \texttt{in}\ e\ \texttt{end} : 0 \rightsquigarrow \texttt{let}\ a_1\ \texttt{in}\ \ldots \texttt{let}\ a_n\ \texttt{in}\ \texttt{let}\ x = v\ \texttt{in}\ e\ \texttt{end}\ \texttt{end}\ \texttt{end}}$$

$$\frac{\Delta; \Gamma \vdash v_1 : \neg\tau \rightsquigarrow \texttt{let}\ \vec{a_1}\ \texttt{in}\ v_1\ \texttt{end} \qquad \Delta; \Gamma \vdash v_2 : \tau \rightsquigarrow \texttt{let}\ \vec{a_2}\ \texttt{in}\ v_2\ \texttt{end}}{\Delta; \Gamma \vdash v_1 v_2 : 0 \rightsquigarrow \texttt{let}\ \vec{a_1}\ \texttt{in}\ \texttt{let}\ \vec{a_2}\ \texttt{in}\ v_1 v_2\ \texttt{end}\ \texttt{end}}$$

13 Module

13.1 Target Language: IL-Module

$$\begin{aligned}
k &::= \mathrm{T} \mid S(c) \mid \Pi\alpha : k_1\ .\ k_2 \mid \Sigma\alpha : k_1\ .\ k_2 \mid 1 \\
c &::= \cdots \mid c_1 \to c_2 \mid \forall\alpha : k\ .\ c \mid \ldots \\
e &::= \cdots \mid \texttt{ext}\ M \\
\sigma &::= 1 \mid (\!| k |\!) \mid \langle\!| \tau |\!\rangle \mid \Pi^{\texttt{gen}}\alpha : \sigma_1\ .\ \sigma_2 \mid \Pi^{\texttt{app}}\alpha : \sigma_1\ .\ \sigma_2 \mid \Sigma\alpha : \sigma_1\ .\ \sigma_2 \\
M &::= * \mid (\!| k |\!) \mid \langle\!| e |\!\rangle
\end{aligned}$$

$$
\begin{aligned}
k &::= \mathrm{T} \mid S(c) \mid \Pi\alpha : k_1 \,.\, k_2 \mid \Sigma\alpha : k_1 \,.\, k_2 \mid 1 \\
\tau, c &::= \alpha \mid \lambda\alpha : k \,.\, c \mid c\ c \mid \langle c, c\rangle \mid \pi_1 c \mid \pi_2 c \mid * \mid \tau \to \tau \mid \langle \tau_1 \ldots \tau_n\rangle \mid \cancel{\texttt{ext}\ M} \\
e &::= \cdots \mid \texttt{ext}\ M \\
\sigma &::= 1 \mid \llparenthesis k \rrparenthesis \mid \langle\!| \tau |\!\rangle \mid \Pi^{\texttt{gen}}\alpha/\cancel{s} : \sigma_1 \,.\, \sigma_2 \mid \Pi^{\texttt{app}}\alpha/\cancel{s} : \sigma_1 \,.\, \sigma_2 \mid \Sigma\alpha/\cancel{s} : \sigma_1 \,.\, \sigma_2 \\
M &::= s \mid * \mid \llparenthesis c \rrparenthesis \mid \langle\!| e |\!\rangle \mid \lambda^{\texttt{gen}}\alpha/s : \sigma \,.\, M \mid \lambda^{\texttt{app}}\alpha/s : \sigma \,.\, M \mid M \cdot M \mid M\ M \mid \langle M, M\rangle \mid \Pi_1 M \mid \Pi_2 M \mid M :> \sigma \\
\Gamma &::= \epsilon \mid \Gamma, \alpha : k \mid \Gamma, x : \tau \mid \Gamma, \alpha/s : \sigma
\end{aligned}
$$

σ for signatures
M for modules
α/s is called twinning. α stands for the static portion of s.
in σ, we don't need twinning because no kind ever contains s, so we can't actually even make use of it even on accident.

So how does this map to SML modules?

```
sig
  type t
  val f : t
end

struct
  type t = int
  val f = 12
end
```

The signature translates to $\Sigma\alpha/\cancel{s} : \llparenthesis\, \mathrm{T} \,\rrparenthesis \,.\, \langle\!| \texttt{ext}\ s |\!\rangle$
The structure translates to $\langle \llparenthesis\, \mathrm{int} \,\rrparenthesis, \langle\!| 12 |\!\rangle\rangle$

1. Phase distinction: static vs. dynamic
   Type Checking relies only on static components
   so: static can never depend on dynamic
   Having a 2nd class module system solves this problem.
   If we had a 1st class module system, we need to try very hard to preserve phase distinction.

2. Static Projection

$$
\Gamma \vdash \texttt{Fst}(M) \gg c \qquad \Gamma \vdash \texttt{Fst}(*) \gg * \qquad \Gamma \vdash \texttt{Fst}(\llparenthesis c \rrparenthesis) \gg c \qquad \Gamma \vdash \texttt{Fst}(\langle\!| c |\!\rangle) \gg *
$$

$$
\frac{\Gamma \vdash \texttt{Fst}(M_1) \gg c_1 \qquad \Gamma \vdash \texttt{Fst}(M_2) \gg c_2}{\Gamma \vdash \texttt{Fst}(M_1 \cdot M_2) \gg c_1\ c_2}
\qquad
\frac{\Gamma \vdash \texttt{Fst}(M_1) \gg c_1 \qquad \Gamma \vdash \texttt{Fst}(M_2) \gg c_2}{\Gamma \vdash \texttt{Fst}(\langle M_1, M_2\rangle) \gg \langle c_1, c_2\rangle}
$$

$$
\frac{\Gamma \vdash \texttt{Fst}(M) \gg c}{\Gamma \vdash \texttt{Fst}(\pi_i M) \gg \pi_i c}
\qquad
\frac{\alpha/s : \sigma \in \Gamma}{\Gamma \vdash \texttt{Fst}(s) \gg \alpha}
\qquad
\frac{\Gamma, \alpha/s : \sigma \vdash \texttt{Fst}(M) \gg c}{\Gamma \vdash \texttt{Fst}(\lambda^{\texttt{ap}} s : \sigma \,.\, M) \gg \lambda\alpha : \texttt{Fst}(\sigma) \,.\, c}
$$

3. Twinning

4. Sealing is an effect

$$\begin{aligned}
\mathtt{Fst}(1) &= 1\\
\mathtt{Fst}((\!|k|\!)) &= k\\
\mathtt{Fst}(\langle\!|k|\!\rangle) &= 1\\
\mathtt{Fst}(\Pi^{\mathsf{ap}}\alpha/s:\sigma_1\,.\,\sigma_2) &= \Pi\alpha:\mathtt{Fst}(\sigma_1)\,.\,\mathtt{Fst}(\sigma_2)\\
\mathtt{Fst}(\Sigma\alpha/s:\sigma_1\,.\,\sigma_2) &= \Sigma\alpha:\mathtt{Fst}(\sigma_1)\,.\,\mathtt{Fst}(\sigma_2)
\end{aligned}$$

13.2 $\Gamma \vdash s : \sigma$

$$\cancel{\frac{\Gamma(\alpha)=k}{\Gamma\vdash\alpha:k}} \qquad \frac{\alpha:k\in\Gamma}{\Gamma\vdash\alpha:k} \qquad \frac{\alpha/s:\sigma\in\Gamma}{\Gamma\vdash\alpha:\mathtt{Fst}(\sigma)}$$

$$\frac{\alpha/s:\sigma\in\Gamma}{\Gamma\vdash s:\sigma} \qquad \frac{}{\Gamma\vdash *:1} \qquad \frac{\Gamma\vdash c:k}{\Gamma\vdash(\!|c|\!):(\!|k|\!)} \qquad \frac{\Gamma\vdash e:\tau}{\Gamma\vdash\langle\!|e|\!\rangle:\langle\!|\tau|\!\rangle} \qquad \frac{\Gamma\vdash\sigma_1:\mathtt{sig}\qquad\Gamma,\alpha/s:\sigma_1\vdash M:\sigma_2}{\Gamma\vdash\lambda^{\mathsf{ap}}\alpha/s:\sigma_1\,.\,M:\Pi^{\mathsf{ap}}\alpha:\sigma_1\,.\,\sigma_2}$$

$$\frac{\Gamma\vdash M_1:\Pi^{\mathsf{ap}}\alpha:\sigma_1\,.\,\sigma_2\qquad\Gamma\vdash M_2:\sigma_1\qquad\Gamma\vdash\mathtt{Fst}(M_2)\gg c_2}{\Gamma\vdash M_1\cdot M_2:[c_2/\alpha]\sigma_2} \qquad \frac{\Gamma\vdash M_1:\sigma_2\qquad\Gamma\vdash M_2:\sigma_2}{\Gamma\vdash\langle M_1,M_2\rangle:\sigma_1\times\sigma_2}$$

$$\frac{\Gamma\vdash M:\Sigma\alpha:\sigma_1\,.\,\sigma_2}{\Gamma\vdash\pi_1M:\sigma_1} \qquad \frac{\Gamma\vdash M:\Sigma\alpha:\sigma_1\,.\,\sigma_2\qquad\Gamma\vdash\mathtt{Fst}(M)\gg c}{\Gamma\vdash\pi_2M:[\pi_1c/\alpha]\sigma_2} \qquad \frac{\Gamma\vdash M:\sigma\qquad\Gamma\vdash\sigma\le\sigma'}{\Gamma\vdash M:\sigma'}$$

13.3 $\Gamma \vdash s : \mathtt{sig}$, $\Gamma \vdash \sigma \le \sigma$

$$\frac{\Gamma\vdash\sigma:\mathtt{sig}}{\Gamma\vdash\sigma\le\sigma} \qquad \frac{\Gamma\vdash\sigma_1\le\sigma_2\qquad\Gamma\vdash\sigma_2\le\sigma_3}{\Gamma\vdash\sigma_1\le\sigma_3} \qquad \frac{\Gamma\vdash k\le k'}{\Gamma\vdash(\!|k|\!)\le(\!|k'|\!)} \qquad \cancel{\frac{\Gamma\vdash\tau\equiv\tau':\mathrm{T}}{\Gamma\vdash\langle\!|\tau|\!\rangle\le\langle\!|\tau'|\!\rangle}}$$

$$\frac{\Gamma\vdash\sigma_1'\le\sigma_1\qquad\Gamma,\alpha:\mathtt{Fst}(\sigma_1')\vdash\sigma_2\le\sigma_2'\qquad\Gamma,\alpha:\mathtt{Fst}(\sigma_1)\vdash\sigma_2:\mathtt{sig}}{\Gamma\vdash\Pi^{\mathsf{ap}}\alpha:\sigma_1\,.\,\sigma_2\le\Pi^{\mathsf{ap}}\alpha:\sigma_1'\,.\,\sigma_2'}$$

$$\frac{\Gamma\vdash\sigma_1'\le\sigma_1\qquad\Gamma,\alpha:\mathtt{Fst}(\sigma_1)\vdash\sigma_2\le\sigma_2'\qquad\Gamma,\alpha:\mathtt{Fst}(\sigma_1')\vdash\sigma_2':\mathtt{sig}}{\Gamma\vdash\Sigma\alpha:\sigma_1\,.\,\sigma_2\le\Sigma\alpha:\sigma_1'\,.\,\sigma_2'}$$

$$\frac{}{\Gamma\vdash 1:\mathtt{sig}} \qquad \frac{\Gamma\vdash k:\mathrm{kind}}{\Gamma\vdash(\!|k|\!):\mathtt{sig}} \qquad \frac{\Gamma\vdash\tau:\mathrm{T}}{\Gamma\vdash\langle\!|\tau|\!\rangle:\mathtt{sig}} \qquad \frac{\Gamma\vdash\sigma:\mathtt{sig}\qquad\Gamma,\alpha:\mathtt{Fst}(\sigma_1)\vdash\sigma_2:\mathtt{sig}}{\Gamma\vdash\Pi^{\mathsf{ap}}\alpha:\sigma_1\,.\,\sigma_2:\mathtt{sig}}$$

(Last rule also works for Π^{gen} and Σ

$$\frac{\Gamma\vdash\sigma:\mathtt{sig}}{\Gamma\vdash\sigma\equiv\sigma:\mathtt{sig}} \qquad \frac{\Gamma\vdash\sigma_1\equiv\sigma_2:\mathtt{sig}}{\Gamma\vdash\sigma_2\equiv\sigma_1:\mathtt{sig}} \qquad \frac{\Gamma\vdash\sigma_1\equiv\sigma_2:\mathtt{sig}\qquad\Gamma\vdash\sigma_2\equiv\sigma_3:\mathtt{sig}}{\Gamma\vdash\sigma_1\equiv\sigma_3:\mathtt{sig}} \qquad \frac{\Gamma\vdash k\equiv k':\mathrm{kind}}{\Gamma\vdash(\!|k|\!)\equiv(\!|k'|\!):\mathtt{sig}}$$

$$\frac{\Gamma\vdash\tau\equiv\tau':\mathrm{T}}{\Gamma\vdash\langle\!|\tau|\!\rangle\equiv\langle\!|\tau'|\!\rangle:\mathtt{sig}} \qquad \frac{\Gamma\vdash\sigma_1\equiv\sigma_1':\mathtt{sig}\qquad\Gamma,\alpha:\mathtt{Fst}(\sigma_1)\vdash\sigma_2\equiv\sigma_2':\mathtt{sig}}{\Gamma\vdash\Pi^{\mathsf{ap}}\alpha:\sigma_1\,.\,\sigma_2\equiv\Pi^{\mathsf{ap}}\alpha:\sigma_1'\,.\,\sigma_2':\mathtt{sig}}$$

(Last rule also works for Π^{gen} and Σ

If we have $\Gamma\vdash M:\sigma\leadsto[c,e]$, we can split it up into its static portion and the rest of it, "phase separation". More on this next time.

Consider

```
σ =
  sig
    type t
    val x : t
    val f : t → t
    val g : t → bool
  end

M₁ =
  struct
    type t = bool
    val x = true
    val f = not
    val g = λx. x
  end

M₂ =
  struct
    type t = int
    val x = 0
    val f = λx. x + 1
    val g = even?
  end
```

$M_1 : \sigma$, $M_2 : \sigma$
note that the two ts are different unless they are sealed, eg:
$M_1 :> \sigma$, $M_2 :> \sigma$

We don't want to even be able to ask questions about the equivalence of the internal types (t) after being sealed. We will call $M_1 :> \sigma$ "indeterminate".

Going back to one of the old judgements, $\Gamma \vdash \texttt{Fst}(M) \gg c$ only applies when M is "determinate".

Now, consider
$F : \sigma_1 \rightarrow \sigma_2$
$M : \sigma_1$
$F\ M : \sigma_2$

Typesystem has to track whether or not a module is pure. (Pure in this sense meaning determinate meaning unsealed.) We treat sealing as an effect.

Thus, we use judgement assigning with the purity class κ: $\Gamma \vdash_\kappa M : \sigma$, with $\kappa ::= P \mid I$

13.4 $\Gamma \vdash e : \tau$

Only new rule we need:

$$\frac{\Gamma \vdash_I M : \langle\!|\tau|\!\rangle}{\Gamma \vdash \texttt{Ext}\ M : \tau}$$

13.5 $\Gamma \vdash_\kappa M : \sigma$

$$\frac{}{\Gamma \vdash_P * : 1} \qquad \frac{\alpha/s : \sigma \in \Gamma}{\Gamma \vdash_P s : \sigma} \qquad \frac{\Gamma \vdash c : k}{\Gamma \vdash_P \llparenthesis c \rrparenthesis : \llparenthesis k \rrparenthesis} \qquad \frac{\Gamma \vdash e : \tau}{\Gamma \vdash_P \langle\!| e |\!\rangle : \langle\!| \tau |\!\rangle} \qquad \frac{\Gamma \vdash_P M : \sigma}{\Gamma \vdash_I M : \sigma}$$

$$\frac{\Gamma \vdash_\kappa M : \sigma \qquad \Gamma \vdash \sigma \leq \sigma'}{\Gamma \vdash_\kappa M : \sigma'} \qquad \frac{\Gamma \vdash_I M : \sigma}{\Gamma \vdash_I M :> \sigma : \sigma} \qquad \frac{\Gamma \vdash \sigma_1 : \mathtt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \vdash_I M : \sigma_2}{\Gamma \vdash_P \lambda^{\mathsf{gen}}\alpha/s : \sigma_1 \,.\, M : \Pi^{\mathsf{gen}}\alpha : \sigma_1 \,.\, \sigma_2}$$

$$\frac{\Gamma \vdash_I M_1 : \Pi^{\mathsf{gen}}\alpha : \sigma \,.\, \sigma' \qquad \Gamma \vdash_P M_2 : \sigma \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash_I M_1\ M_2 : [c_2/\alpha]\sigma'} \qquad \frac{\Gamma \vdash \sigma_1 : \mathtt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \vdash_P M : \sigma_2}{\Gamma \vdash_P \lambda^{\mathsf{app}}\alpha/s : \sigma_1 \,.\, M : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2}$$

$$\frac{\Gamma \vdash_\kappa M_1 : \Pi^{\mathsf{app}}\alpha : \sigma \,.\, \sigma' \qquad \Gamma \vdash_P M_2 : \sigma \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash_\kappa M_1 \cdot M_2 : [c_2/\alpha]\sigma'} \qquad \frac{\Gamma \vdash_\kappa M_1 : \sigma_1 \qquad \Gamma \vdash_\kappa M_2 : \sigma_2}{\Gamma \vdash_\kappa \langle M_1, M_2 \rangle : \sigma_1 \times \sigma_2}$$

$$\frac{\Gamma \vdash_P M : \Sigma\alpha : \sigma_1 \,.\, \sigma_2}{\Gamma \vdash_P \pi_1 M : \sigma_1} \qquad \frac{\Gamma \vdash_P M : \Sigma\alpha : \sigma_1 \,.\, \sigma_2 \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c}{\Gamma \vdash_P \pi_2 M : [c/\alpha]\sigma_1}$$

13.6 $\Gamma \vdash \mathtt{Fst}(M) \gg k$

$$\begin{aligned}
\mathtt{Fst}(1) &= 1 \\
\mathtt{Fst}(\llparenthesis k \rrparenthesis) &= k \\
\mathtt{Fst}(\langle\!| \tau |\!\rangle) &= 1 \\
\mathtt{Fst}(\Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2) &= \Pi\alpha : \mathtt{Fst}(\sigma_1) \,.\, \mathtt{Fst}(\sigma_2) \\
\mathtt{Fst}(\Pi^{\mathsf{gen}}\alpha : \sigma_1 \,.\, \sigma_2) &= 1 \\
\mathtt{Fst}(\Sigma\alpha : \sigma_1 \,.\, \sigma_2) &= \Sigma\alpha : \mathtt{Fst}(\sigma_1) \,.\, \mathtt{Fst}(\sigma_2)
\end{aligned}$$

13.7 $\Gamma \vdash \mathtt{Fst}(M) \gg c$

$$\frac{\alpha/s : \sigma \in \Gamma}{\Gamma \vdash \mathtt{Fst}(s) \gg \alpha} \qquad \frac{}{\Gamma \vdash \mathtt{Fst}(*) \gg *} \qquad \frac{}{\Gamma \vdash \mathtt{Fst}(\llparenthesis c \rrparenthesis) \gg c} \qquad \frac{}{\Gamma \vdash \mathtt{Fst}(\langle\!| e |\!\rangle) \gg *}$$

$$\frac{\Gamma \vdash \alpha/s : \sigma_1 \vdash \mathtt{Fst}(M) \gg c}{\Gamma \vdash \mathtt{Fst}(\lambda^{\mathsf{app}}\alpha/s : \sigma_1\ .)\lambda\alpha : \mathtt{Fst}(\sigma_1) \,.\, c} \qquad \frac{\Gamma \vdash \mathtt{Fst}(M_1) \gg c_1 \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash \mathtt{Fst}(M_1 \cdot M_2) \gg c_1\ c_2}$$

$$\frac{}{\Gamma \vdash \mathtt{Fst}(\lambda^{\mathsf{gen}}\alpha/s : \sigma_1 \,.\, M) \gg *} \qquad \cancel{\frac{}{\Gamma \vdash \mathtt{Fst}(M_1\ M_2) \gg}} \qquad \frac{\Gamma \vdash \mathtt{Fst}(M_1) \gg c_1 \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash \mathtt{Fst}(\langle M_1, M_2 \rangle) \gg \langle c_1, c_2 \rangle}$$

$$\frac{\Gamma \vdash \mathtt{Fst}(M) \gg c}{\Gamma \vdash \mathtt{Fst}(\pi_1 M) \gg \pi_i c}$$

Need to support some sort of let binding in the language in order for us to do anything with sealed modules. So let's change the language as such:
$M ::= \cdots \mid \texttt{let}\ \alpha/s = M\ \texttt{in}\ M\ \texttt{end}$

with typing rule

$$\frac{\Gamma \vdash_I M_1 : \sigma_1 \qquad \Gamma, \alpha/s : \sigma_1 \vdash_I M_2 : \sigma_2 \qquad \Gamma \vdash \sigma_2 : \texttt{sig}}{\Gamma \vdash_I \texttt{let}\ \alpha/s = M_1\ \texttt{in}\ M_2\ \texttt{end}} \qquad \frac{\Gamma \vdash M_1 \Rightarrow \sigma_1 \qquad \Gamma, \alpha/s : \sigma_1 \vdash M_2 \Rightarrow \sigma_2}{\Gamma \vdash \texttt{let}\ \alpha/s = M_1\ \texttt{in}\ M_2\ \texttt{end} \Rightarrow}$$

Big idea #5: "Avoidance Problem".
There is no "best" signature for this last rule above.
Problem:
Given $\Gamma, \alpha : k \vdash \sigma : \texttt{sig}$
obtain σ' s.t.
1. $\Gamma \vdash \sigma' : \texttt{sig}$
2. $\Gamma, \alpha : k \vdash \sigma \leq \sigma'$
3. forall σ'' if $\Gamma \vdash \sigma'' : \texttt{sig}$ and $\Gamma, \alpha : k \vdash \sigma \leq \sigma''$ then $\Gamma \vdash \sigma' \leq \sigma''$
But this problem has no solution.

Example: $\alpha : \mathrm{T} \vdash (\mathrm{T} \to S(\alpha)) \times S(\alpha) : \mathrm{kind}$
This is a sub-kind of $\sigma\beta : \mathrm{T} \to \mathrm{T}\ .\ S(\beta\,\mathrm{int})$
And also a sub-kind of $\sigma\beta : \mathrm{T} \to \mathrm{T}\ .\ S(\beta\,\mathrm{string})$
But these two that we just generated are not equivalent.

so instead, we ask the programmer for the σ_2
$M ::= \cdots \mid \texttt{let}\ \alpha/s = M\ \texttt{in}\ M : \sigma\ \texttt{end}$

$$\frac{\Gamma \vdash M_1 \Rightarrow \sigma_1 \qquad \Gamma \vdash \sigma \Leftarrow \texttt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \vdash M_2 \Leftarrow \sigma_2}{\Gamma \vdash \texttt{let}\ \alpha/s = M_1\ \texttt{in}\ M_2 : \sigma_2\ \texttt{end} \Rightarrow \sigma_2}$$

But there is an issue we cannot deal with in this language. When we parse a structure into a tuple, we would have a bunch of $: \sigma$ in the nested lets (multiple structures one after another in a structure). While this is not incorrect, it's grossly inefficient.
So instead, we add one more feature:
$M ::= \cdots \mid \texttt{let}\ \alpha/s = M\ \texttt{in}\ M : \sigma\ \texttt{end} \mid \langle \alpha/s = M, M \rangle$

$$\frac{\Gamma \vdash_\kappa M_1 : \sigma_1 \qquad \Gamma, \alpha/s : \sigma_1 \vdash_\kappa M_2 : \sigma_2}{\Gamma \vdash_\kappa \langle \alpha/s = M_1, M_2 \rangle : \Sigma\alpha : \sigma_1\ .\ \sigma_2}$$

This is somewhat motivated by the Avoidance Problem.
Add one more feature:
$M ::= \cdots \mid \texttt{letp}\ \alpha/s = M\ \texttt{in}\ M\ \texttt{end}$

$$\frac{\Gamma \vdash_P M_1 : \sigma_1 \qquad \Gamma, \alpha/s : \sigma_1 \vdash_\kappa M_2 : \sigma_2 \qquad \Gamma \vdash \texttt{Fst}(M_1) \gg c_1}{\Gamma \vdash_\kappa \texttt{letp}\ \alpha/s = M_1\ \texttt{in}\ M_2\ \texttt{end} : [c_1/\alpha]\sigma_2}$$

Note, in the first rule in the next section below, we find that we can't just use σ because it's not the "best" signature. So we need to introduce a new $S(\sigma)$
Kind level:

$$\begin{aligned}
S(c : \mathrm{T}) &= S(c) \\
S(c : \Pi\alpha : k_1\ .\ k_2) &= \Pi\alpha : k_1\ .\ S(c\ \alpha : k_2) \\
S(c : \Sigma\alpha : k_1\ .\ k_2) &= S(\pi_1 c : k_1) \times S(\pi_2 c : [\pi_1 c/\alpha]k_2) \\
S(c : S(c')) &= S(c)
\end{aligned}$$

What we think the property should look like:
"$c' : S(c : k)$ iff $c' : k$ and $c \equiv c' : k$" "$\vdash_P M : S(c : \sigma)$ iff $M : \sigma$ and $\mathtt{Fst}(M) \equiv c : \mathtt{Fst}(\sigma)$" Module level:

$$\begin{aligned}
S(c : 1) &= 1 \\
S(c : (\!| k |\!)) &= (\!| S(c : k) |\!) \\
S(c : \langle\!| \tau |\!\rangle) &= \langle\!| \tau |\!\rangle \\
S(c : \Pi^{\mathsf{gen}} \alpha : \sigma_1 \,.\, \sigma_2) &= \Pi^{\mathsf{gen}} \alpha : \sigma_1 \,.\, \sigma_2 \\
S(c : \Pi^{\mathsf{app}} \alpha : \sigma_1 \,.\, \sigma_2) &= \Pi^{\mathsf{app}} \alpha : \sigma_1 \,.\, S(c\ \alpha : \sigma_2) \\
S(c : \Sigma \alpha : \sigma_1 \,.\, \sigma_2) &= S(\pi_1 c : \sigma_1) \times S(\pi_2 c : [\pi_1 c/\alpha]\sigma_2)
\end{aligned}$$

NOTE selfification: when you take the signature of a module and write it back into the module.

So now, we want the property to be:
If $\Gamma \vdash_P M : \sigma$ and $\Gamma \vdash \mathtt{Fst}(M) \gg c$ then $\Gamma \vdash_P M : S(c : \sigma)$

But as it is right now, it's not actually true. So like back in the singleton calculus, we need to add in the extentionality rules:

13.8 $\overset{+}{\Gamma} \vdash_{\bar{\kappa}} \overset{+}{M} \Rightarrow \bar{\sigma}$

$$\frac{\alpha/s : \sigma \in \Gamma}{\Gamma \vdash_P s \Rightarrow S(\alpha : \sigma)}$$

13.9 Extentionality Rules

Suppose we had some
$F : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2$ where
$\mathtt{Fst}(F) \gg \varphi$

Then, $F : S(\varphi : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2)$ should be true, but we have no way of proving it right now.

$$\frac{\Gamma \vdash_P M : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2' \qquad \Gamma, \alpha : \sigma_1 \vdash_P M \cdot \alpha : \sigma_2}{\Gamma \vdash_P M : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2}$$

$$\frac{\Gamma \vdash \pi_1 M : \sigma_1 \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c \qquad \Gamma \vdash \pi_2 M : [\pi_1 c/\alpha]\sigma_2 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1) \vdash \sigma_2 : \mathtt{sig}}{\Gamma \vdash M : \sigma\alpha : \sigma_1 \,.\, \sigma_2}$$

$$\frac{f : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2 \qquad \alpha/s : \sigma_1 \vdash f\ s : S(\varphi\ \alpha : \sigma_2)}{\Gamma \vdash f : S(\varphi : \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2)}$$

(Note in the last one, $\Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2$ is equivalent to $\Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, S(\varphi\ \alpha : \sigma_2)$.)

Also:

$$\frac{\Gamma \vdash_P M : (\!|k'|\!) \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c \qquad \Gamma \vdash c : k}{\Gamma \vdash_P M : (\!|k|\!)}$$

Selfification property:
If $\vdash \Gamma\,\mathrm{ok}$, $\Gamma \vdash M : \sigma$, $\Gamma \vdash \mathtt{Fst}(M) \gg c$
then $\Gamma \vdash M : S(c : \sigma)$.

13.10 $\Gamma \vdash M \Rightarrow \sigma$

$$\frac{\alpha/s : \sigma \in \Gamma}{\Gamma \vdash_P s \Rightarrow S(\alpha : \sigma)} \qquad \frac{\Gamma \vdash c \Rightarrow k}{\Gamma \vdash_P (\!|c|\!) \Rightarrow (\!|k|\!)} \qquad \frac{\Gamma \vdash e \Rightarrow \tau}{\Gamma \vdash_P \langle\!| e |\!\rangle \Rightarrow \langle\!| \tau |\!\rangle} \qquad \frac{\Gamma \vdash \sigma_1 \Leftarrow \mathtt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \vdash_\kappa M \Rightarrow \sigma_2}{\Gamma \vdash_P \lambda^{\mathsf{gen}}\alpha/s : \sigma_1 \,.\, M \Rightarrow \Pi^{\mathsf{gen}}\alpha : \sigma_1 \,.\, \sigma_2}$$

$$\frac{\Gamma \vdash_\kappa M_1 \Rightarrow \Pi^{\mathsf{gen}}\alpha : \sigma \,.\, \sigma' \qquad \Gamma \vdash_P M_2 \Leftarrow \sigma \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash_I M_1\ M_2 \Rightarrow [c_2/\alpha]\sigma'}$$

$$\frac{\Gamma \vdash \sigma_1 \Leftarrow \mathtt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \vdash_P M \Rightarrow \sigma_2}{\Gamma \vdash_P \lambda^{\mathsf{app}}\alpha/s : \sigma_1 \,.\, M \Rightarrow \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2}$$

$$\frac{\Gamma \vdash_\kappa M_1 \Rightarrow \Pi^{\mathsf{app}}\alpha : \sigma \,.\, \sigma' \qquad \Gamma \vdash_P M_2 \Leftarrow \sigma \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash M_1 \cdot M_2 \Rightarrow [c_2/\alpha]\sigma'}$$

$$\frac{\Gamma \vdash_{\kappa_1} M_1 \Rightarrow \sigma_1 \qquad \Gamma, \alpha/s : \sigma_1 \vdash_{\kappa_2} M_2 \Rightarrow \sigma_2}{\Gamma \vdash_{\kappa_1 \cup \kappa_2} \langle \alpha/s = M_1, M_2 \rangle \Rightarrow \Sigma\alpha : \sigma_1 \,.\, \sigma_2} \qquad \frac{\Gamma \vdash_P M \Rightarrow \Sigma\alpha : \sigma_1 \,.\, \sigma_2}{\Gamma \vdash_P \pi_1 M \Rightarrow \sigma_1}$$

$$\frac{\Gamma \vdash_P M \Rightarrow \Sigma\alpha : \sigma_1 \,.\, \sigma_2 \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c}{\Gamma \vdash_P \pi_2 M \Rightarrow [\pi_1 c/\alpha]\sigma_2} \qquad \frac{\Gamma \vdash \sigma \Leftarrow \mathtt{sig} \qquad \Gamma \vdash_\kappa M \Leftarrow \sigma}{\Gamma \vdash_I M :> \sigma \Rightarrow \sigma}$$

$$\frac{\Gamma \vdash_{\kappa_1} M_1 \Rightarrow \sigma_1 \qquad \Gamma \vdash \sigma_2 \Leftarrow \mathtt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \vdash_{\kappa_2} M_2 \Leftarrow \sigma_2}{\Gamma \vdash_I \mathtt{let}\ \alpha/s = M_1\ \mathtt{in}\ M_2\ \mathtt{end} : \sigma_2 \Rightarrow \sigma_2}$$

$$\frac{\Gamma \vdash_P M_1 \Rightarrow \sigma_1 \qquad \Gamma \vdash \mathtt{Fst}(M_1) \gg c_1 \qquad \Gamma, \alpha/s : \sigma_1 \vdash_\kappa M_2 \Rightarrow \sigma_2}{\Gamma \vdash \mathtt{letp}\ \alpha/s = M_1\ \mathtt{in}\ M_2\ \mathtt{end} \Rightarrow [c_1/\alpha]\sigma_2}$$

13.11 $\Gamma \vdash M \Leftarrow \sigma$

$$\frac{\Gamma \vdash_\kappa M \Rightarrow \sigma' \qquad \Gamma \vdash \sigma' \trianglelefteq \sigma}{\Gamma \vdash_\kappa M \Leftarrow \sigma}$$

13.12 $\Gamma \vdash e : \tau$

$$\frac{\Gamma_\kappa M \Rightarrow \langle\!| \tau |\!\rangle}{\Gamma \vdash \mathtt{Ext}\ M \Rightarrow \tau}$$

13.13 Closing

We skipped $\Gamma \vdash \sigma_1 \trianglelefteq \sigma_2$ and $\Gamma \vdash \sigma \Leftarrow \mathtt{sig}$ because they are fairly straightforward.

14 Phase Splitting

With phase distinction, we separate out the static (Fst) and dynamic components.
Target language is IL-Direct, which we already discussed in great detail.
Judgements:
$\sigma \rightsquigarrow [\alpha : k \,.\, \tau]$
$\Gamma \vdash_P M : \sigma \rightsquigarrow [c, e]$
$\Gamma \vdash_{\mp} M : \sigma \rightsquigarrow e$

If $\Gamma \vdash \sigma : \texttt{sig}$, $\sigma \rightsquigarrow [\alpha : k \,.\, \tau]$
then $\bar{\Gamma} \vdash k : \text{kind}$, $\bar{\Gamma}, \alpha : k \vdash \tau : \text{T}$.
If $\sigma \rightsquigarrow [\alpha : k \,.\, \tau]$ then $\texttt{Fst}(\sigma) = k$.

If $\sigma \rightsquigarrow [\alpha : k \,.\, \tau]$, $\Gamma \vdash_P M : \sigma \rightsquigarrow [c, e]$
then $\bar{\Gamma} \vdash c : k$, $\bar{\Gamma} \vdash e : [c/\alpha]\tau$.

If $\sigma \rightsquigarrow [\alpha : k \,.\, \tau]$, $\Gamma \vdash_I M : \sigma \rightsquigarrow e$
then $\bar{\Gamma} \vdash e : \exists \alpha : k \,.\, \tau$.

14.1 $\sigma \rightsquigarrow [\alpha : k \,.\, \tau]$

$$\frac{}{1 \rightsquigarrow [\alpha : 1 \,,\, \text{unit}]} \qquad \frac{}{(\!| k |\!) \rightsquigarrow [\alpha : k \,,\, \text{unit}]} \qquad \frac{}{\langle\!| \tau |\!\rangle \rightsquigarrow [\alpha : 1 \,,\, \tau]}$$

$$\frac{\sigma_1 \rightsquigarrow [\alpha_1 : k_1 \,,\, \tau_1] \qquad \sigma_2 \rightsquigarrow [\alpha_2 : k_2 \,,\, \tau_2]}{\Pi^{\mathsf{app}} \alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [\beta : \Pi\alpha : k_1 \,.\, k_2 \,,\, \forall \alpha : k_1 \,.\, [\alpha/\alpha_1]\tau_1 \to [\beta\ \alpha/\alpha_2]\tau_2]}$$

$$\frac{\sigma_1 \rightsquigarrow [\alpha_1 : k_1 \,,\, \tau_1] \qquad \sigma_2 \rightsquigarrow [\alpha_2 : k_2 \,,\, \tau_2]}{\Pi^{\mathsf{gen}} \alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [\beta : 1 \,,\, \forall \alpha : k_1 \,.\, [\alpha/\alpha_1]\tau_1 \to \exists \alpha : k_2 \,.\, \tau_2]}$$

$$\frac{\sigma_1 \rightsquigarrow [\alpha_1 : k_1 \,,\, \tau_1] \qquad \sigma_2 \rightsquigarrow [\alpha_2 : k_2 \,,\, \tau_2]}{\Sigma \alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [\beta : \Sigma\alpha : k_1 \,.\, k_2 \,,\, [\pi_1\beta/\alpha_1]\tau_1 \times [\pi_1\beta, \pi_2\beta/\alpha, \alpha_2]\tau_2]}$$

14.2 $\sigma \rightsquigarrow [k \,,\, \tau]$

Let's revisit all of the above rules in the Debruijn world.

$$\frac{\Gamma \vdash s : \Gamma' \qquad \Gamma \vdash m : A[s]}{\Gamma \vdash m \,.\, s : \Gamma', A} \qquad\qquad \frac{|\Gamma'| = k}{\Gamma, \Gamma' \vdash \uparrow k : \Gamma}$$

If $\Gamma \vdash \sigma : \texttt{sig}$ and $\sigma \rightsquigarrow [\alpha : k \,,\, \tau]$ then $\Gamma \vdash k : \text{kind}, \Gamma, \alpha : k \vdash \tau : \mathrm{T}$.

Now the Debruijn form:
If $\Gamma \vdash \sigma : \texttt{sig}$ and $\sigma \rightsquigarrow [k \,,\, \tau]$ then $\Gamma \vdash k : \text{kind}$ and $\Gamma, k \vdash \tau : \mathrm{T}$.

$$\frac{}{1 \rightsquigarrow [1 \,,\, \text{unit}\,]} \qquad\qquad \frac{}{(\!| k |\!) \rightsquigarrow [k \,,\, \text{unit}\,]} \qquad\qquad \frac{}{\langle\!| \tau |\!\rangle \rightsquigarrow [1 \,,\, \tau[\uparrow]]}$$

$$\frac{\sigma_1 \rightsquigarrow [k_1 \,,\, \tau_1] \qquad \sigma_2 \rightsquigarrow [k_2 \,,\, \tau_2]}{\Pi^{\texttt{app}}\sigma_1 \,.\, \sigma_2 \rightsquigarrow [\Pi k_1 \,.\, k_2 \,,\, \forall k_1[\uparrow] \,.\, \tau_1[0 \,.{\uparrow}^2] \to \tau_2[1\ 0 \,.\, 0 \,.{\uparrow}^2]]}$$

$$\frac{\sigma_1 \rightsquigarrow [k_1 \,,\, \tau_1] \qquad \sigma_2 \rightsquigarrow [k_2 \,,\, \tau_2]}{\Pi^{\texttt{gen}} : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [1 \,,\, \forall k_1[\uparrow] \,.\, \tau_1[0.\uparrow^2] \to \exists k_2[0.\uparrow^2] \,.\, \tau_2[0.1.\uparrow^3]]}$$

$$\frac{\sigma_1 \rightsquigarrow [k_1 \,,\, \tau_1] \qquad \sigma_2 \rightsquigarrow [k_2 \,,\, \tau_2]}{\Sigma\sigma_1 \,.\, \sigma_2 \rightsquigarrow [\Sigma k_1 \,.\, k_2 \,,\, \tau_1[\pi_1 0.\uparrow] \times \tau_2[\pi_2 0.\pi_1 0.\uparrow]]}$$

14.3 $\Gamma \rightsquigarrow \Gamma$

$$\begin{aligned} \epsilon &= \epsilon \\ \Gamma, \alpha : k &= \Gamma, \alpha : k \\ \Gamma, x : \tau &= \Gamma, x : \tau \\ \Gamma, \alpha/s : \sigma &= \Gamma, \alpha : k, s : \tau \end{aligned}$$

*NOTE: in the last one, $\sigma \rightsquigarrow [\alpha : k \,,\, \tau]$

14.4 $\Gamma \vdash_P M : \sigma \rightsquigarrow [c \,,\, e]$

$$\frac{\alpha/s : \sigma \in \Gamma}{\Gamma \models_{\mathrm{P}} s : \sigma \rightsquigarrow [\alpha \,,\, s]} \qquad \frac{\Gamma \models_{\mathrm{P}} * : 1 \rightsquigarrow [* \,,\, \langle\rangle]}{} \qquad \frac{\Gamma \vdash c : k}{\Gamma \models_{\mathrm{P}} (\!| c |\!) : (\!| k |\!) \rightsquigarrow [c \,,\, \langle\rangle]} \qquad \frac{\Gamma \vdash e : \tau}{\Gamma \vdash \langle\!| e |\!\rangle : \langle\!| \tau |\!\rangle \rightsquigarrow [* \,,\, e]}$$

$$\frac{\Gamma \vdash \sigma_1 : \texttt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \models_{\mathrm{P}} M : \sigma_2 \rightsquigarrow [c \,,\, e] \qquad \sigma_1 \rightsquigarrow [\alpha_1 : k_1 \,,\, \tau_1]}{\Gamma \vdash \lambda^{\texttt{app}}\alpha/s : \sigma_1 \,.\, M : \Pi\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [\lambda\alpha : k_1 \,.\, c \,,\, \Lambda\alpha : k_1 \,.\, \lambda s : [\alpha/\alpha_1]\tau_1 \,.\, e]}$$

$$\frac{\Gamma \Vdash_{\mathrm{P}} M_1 : \Pi^{\mathtt{app}}\alpha : \sigma \,.\, \sigma_2 \rightsquigarrow [c_1^{:\Pi\alpha:k_1.k_2} \,,\, e_1^{:\forall\alpha:k_1.[\alpha/\alpha_1]\tau_1 \to [c_1\ \alpha/\alpha_2]\tau_2}] \qquad \Gamma \Vdash_{\mathrm{P}} M_2 : \sigma_1 \rightsquigarrow [c_1^{:k_1} \,,\, e_1^{:[c_2/\alpha_1]\tau_1}] \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \Vdash_{\mathrm{P}} M_1 \cdot M_2 : [c_2/\alpha]\sigma_2 \rightsquigarrow [c_1\ c_2 \,,\, e_1[c_2]\ e_2]}$$

$$\frac{\Gamma \Vdash_{\mathrm{P}} M_1 : \sigma_1 \rightsquigarrow [c_1 \,,\, e_1] \qquad \Gamma, \alpha/s : \sigma_1 \Vdash_{\mathrm{P}} M_2 : \sigma_2 \rightsquigarrow [c_2 \,,\, e_2]}{\Gamma \Vdash_{\mathrm{P}} \langle \alpha/s = M_1, M_2 \rangle : \Sigma\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [\langle c_1, [c_1/\alpha]c_2 \rangle \,,\, \mathtt{let}\ s = e_1\ \mathtt{in}\ \langle s, [c_1/\alpha]e_2 \rangle\ \mathtt{end}]}$$

$$\frac{\Gamma \Vdash_{\mathrm{P}} M : \Sigma\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [c \,,\, e]}{\Gamma \Vdash_{\mathrm{P}} \pi_1 M : \sigma_1 \rightsquigarrow [\pi_1 c \,,\, \pi_1 e]} \qquad \frac{\Gamma \Vdash_{\mathrm{P}} M : \Sigma\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [c \,,\, e] \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c}{\Gamma \Vdash_{\mathrm{P}} \pi_2 M : [\pi_1 c/\alpha]\sigma_2 \rightsquigarrow [\pi_2 c \,,\, \pi_2 e]}$$

$$\frac{\Gamma \Vdash_{\mathrm{P}} M_1 : \sigma \rightsquigarrow [c_1 \,,\, e_1] \qquad \Gamma \Vdash_{\mathrm{P}} \mathtt{Fst}(M_1) \gg c_1 \qquad \Gamma, \alpha/s : \sigma_1 \Vdash_{\mathrm{P}} M_2 : \sigma_2 \rightsquigarrow [c_2 \,,\, e_2]}{\Gamma \Vdash_{\mathrm{P}} \mathtt{letp}\ \alpha/s = M_1\ \mathtt{in}\ M_2\ \mathtt{end} : [c_1/\alpha]\sigma_2 \rightsquigarrow [[c_1/\alpha]c_2 \,,\, \mathtt{let}\ s = e_1\ \mathtt{in}\ [c_1/\alpha]e_2\ \mathtt{end}]}$$

$$\frac{\Gamma \vdash \sigma_1 : \mathtt{sig} \qquad \Gamma, \alpha/s : \sigma_1 \Vdash_{\mathrm{I}} M : \sigma_2 \rightsquigarrow e \qquad \sigma_1 \rightsquigarrow [\alpha_1 : k_1 \,,\, \tau_1]\sigma_2 \rightsquigarrow [\alpha_2 : k_2 \,,\, \tau_2]}{\Gamma \Vdash_{\mathrm{P}} \lambda^{\mathtt{gen}}\alpha/s : \sigma_1 \,.\, M : \Pi^{\mathtt{gen}}\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [* \,,\, \Lambda\alpha : k_1 \,.\, \lambda s : [\alpha/\alpha_1]\tau_1 \,.\, e]}$$

Lemma:
$\mathtt{Fst}([c/\alpha]\sigma) = [c/\alpha]\mathtt{Fst}(\sigma)$
Lemma:
Suppose $\sigma \rightsquigarrow [\alpha : k \,,\, \tau]$
then $[c/\beta]\sigma \rightsquigarrow [\alpha : [c/\beta]k \,,\, [c/\beta]\tau]$

$$\frac{\Gamma \vdash \mathtt{Fst}(M_1) \gg c_1 \qquad \Gamma, \alpha/s \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash \mathtt{Fst}(\langle \alpha/s = M_1, M_2 \rangle) \gg \langle c_1, [c_1/\alpha]c_2 \rangle} \qquad \frac{\Gamma \vdash \mathtt{Fst}(M_1) \gg c_1 \qquad \Gamma, \alpha/s \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \vdash \mathtt{Fst}(\mathtt{letp}\ \alpha/s = M_1\ \mathtt{in}\ M_2\ \mathtt{end}) \gg [c_1/\alpha]c_2}$$

14.5 $\Gamma \Vdash_{\mathrm{I}} M : \sigma \rightsquigarrow e$

$$\frac{\Gamma \Vdash_{\mathrm{P}} M : \sigma \rightsquigarrow [c \,,\, e] \qquad \sigma \rightsquigarrow [\alpha : k \,,\, e]}{\Gamma \Vdash_{\mathrm{I}} M : \sigma \rightsquigarrow \mathtt{pack}\ [c \,,\, e]\ \mathtt{as}\ \exists\alpha : k \,.\, \tau}$$

$$\frac{\Gamma \Vdash_{\mathrm{P}} M_1 : \Pi^{\mathtt{gen}}\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow [c_1 \,,\, e_1] \qquad \Gamma \Vdash_{\mathrm{P}} M_2 : \sigma_1 \rightsquigarrow [c_2 \,,\, e_2] \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \Vdash_{\mathrm{I}} M_1\ M_2 : [c_2/\alpha]\sigma_2 \rightsquigarrow e_1[c_2]\ e_2}$$

$$\frac{\Gamma \Vdash_{\mathrm{I}} M_1 : \Pi^{\mathtt{gen}}\alpha : \sigma_1 \,.\, \sigma_2 \rightsquigarrow e \qquad \Gamma \Vdash_{\mathrm{P}} M_2 : \sigma_1 \rightsquigarrow [c_2 \,,\, e_2] \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2}{\Gamma \Vdash_{\mathrm{I}} M_1\ M_2 : [c_2/\alpha]\sigma_2 \rightsquigarrow \mathtt{unpack}\ [\beta, f] = e\ \mathtt{in}\ f[c_2]\ e_2\ \mathtt{end}}$$

$$\frac{\Gamma \vdash e : \tau \leadsto e \qquad \Gamma, x : \tau \vdash_{\kappa} M : \sigma \leadsto [c\ ,\ e']}{\Gamma \models_{\mathrm{P}} \mathtt{let}\ x = e\ \mathtt{in}\ M : \sigma\ \mathtt{end} \leadsto [c\ ,\ \mathtt{let}\ x = e\ \mathtt{in}\ e'\ \mathtt{end}]}$$

$$\frac{\Gamma \models_{\mathrm{I}} M_1 : \Pi^{\mathsf{app}}\alpha : \sigma_1\ .\ \sigma_2 \leadsto e_1 \qquad \Gamma \models_{\mathrm{P}} M_2 : \sigma_2 \leadsto [c_2\ ,\ e_2] \qquad \Gamma \vdash \mathtt{Fst}(M_2) \gg c_2 \qquad \sigma_2 \leadsto [\alpha_2 : k_2\ ,\ \tau_2]}{\Gamma \models_{\mathrm{I}} M_1 \cdot M_2 : [c_2/\alpha]\sigma_2 \leadsto \mathtt{unpack}\ [\beta, f] = e_1\ \mathtt{in}\ \mathtt{pack}\ [\beta\ c_2\ ,\ f[c_2]\ e_2]\ \mathtt{as}\ \exists\alpha_2 : [c_2/\alpha]k_2\ .\ [c_2/\alpha]\tau_2\ \mathtt{end}}$$

$$\frac{\Gamma \models_{\mathrm{I}} M_1 : \sigma_1 \leadsto e_1 \qquad \Gamma, \alpha/s : \sigma_1 \models_{\mathrm{I}} M_2 : \sigma_2 \leadsto e_2 \qquad \sigma_2 \leadsto [\alpha_2 : k_2\ ,\ \tau_2]}{\begin{array}{c}\Gamma \models_{\mathrm{I}} \langle \alpha/s = M_1, M_2 \rangle : \Sigma\alpha : \sigma_1\ .\ \sigma_2 \leadsto \\ \mathtt{unpack}\ [\alpha, s] = e_1\ \mathtt{in}\ \mathtt{unpack}\ [\alpha_2, s_2] = e_2\ \mathtt{in}\ \mathtt{pack}\ [\langle \alpha, \alpha_2 \rangle\ ,\ \langle s, s_2 \rangle]\ \mathtt{as}\ \exists\beta : \Sigma\alpha : k_1\ .\ k_2\ .\ [\pi_1\beta/\alpha_1]\tau_1 \times [\pi_1\beta, \pi_2\beta/\alpha, \alpha_2]\tau_2\end{array}}$$

$$\frac{\Gamma \models_{\mathrm{I}} M : \sigma \leadsto e}{\Gamma \models_{\mathrm{I}} M :> \sigma : \sigma \leadsto e} \qquad \frac{\Gamma \models_{\mathrm{I}} M_1 : \sigma_1 \leadsto e_1 \qquad \Gamma, \alpha/s : \models_{\mathrm{I}} M_2 : \sigma_2 \leadsto e_2}{\Gamma \models_{\mathrm{I}} \mathtt{let}\ \alpha/s = M_1\ \mathtt{in}\ M_2 : \sigma_2\ \mathtt{end} : \sigma_2 \leadsto \mathtt{unpack}\ [\alpha, s] = e_1\ \mathtt{in}\ e_2\ \mathtt{end}}$$

$$\frac{\Gamma \models_{\mathrm{P}} M_1 : \sigma_2 \leadsto [c_1\ ,\ e_1] \qquad \Gamma, \alpha/s : \sigma_1 \models_{\mathrm{I}} M_2 : \sigma_2 \leadsto e_2 \qquad \Gamma \vdash \mathtt{Fst}(M_1) \gg c_1}{\Gamma \models_{\mathrm{I}} \mathtt{letp}\ \alpha/s = M_1\ \mathtt{in}\ M_2\ \mathtt{end} : \sigma \leadsto \mathtt{let}\ s = e_1\ \mathtt{in}\ [c_1/\alpha]e_2\ \mathtt{end}}$$

$$\frac{\Gamma \vdash e : \tau \leadsto e' \qquad \Gamma, x : \tau \models_{\mathrm{I}} M : \sigma \leadsto e'}{\Gamma \models_{\mathrm{I}} \mathtt{let}\ x = e\ \mathtt{in}\ M\ \mathtt{end} : \sigma \leadsto \mathtt{let}\ x = e\ \mathtt{in}\ e'\ \mathtt{end}}$$

$$\frac{\Gamma \vdash M : (\!| k' |\!) \leadsto [c^{:k'}\ ,\ e^{:\mathrm{unit}}] \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c \qquad \Gamma \vdash c : k}{\Gamma \vdash M : (\!| k |\!) \leadsto [c\ ,\ e]}$$

$$\frac{\Gamma \models_{\mathrm{P}} M : \Pi^{\mathsf{app}}\alpha : \sigma_1\ .\ \sigma_2' \qquad \Gamma, \alpha/s : \sigma_1 \vdash M \cdot s : \sigma_2 \leadsto [c\ ,\ e]}{\Gamma \models_{\mathrm{P}} M : \Pi^{\mathsf{app}}\alpha : \sigma_1\ .\ \sigma_2 \leadsto [\lambda\alpha : k_1\ .\ c\ ,\ \Lambda\alpha : k_1\ .\ \lambda s : [\alpha/\alpha_1]\tau_1\ .\ e]}$$

Note that the last rule in the previous section is wrong. We instead add the appropriate extensionality rules to replace it.

$$\frac{\Gamma \vDash_{\mathrm{P}} M : \sigma \rightsquigarrow [c\ ,\ e] \qquad \Gamma \vdash \mathtt{Fst}(M) \gg c}{\Gamma \vDash_{\mathrm{P}} M : S(c : \sigma) \rightsquigarrow [c\ ,\ e]}$$

To prove type correctness for the above, we want to show:
$c : k'$
$e : [c/\alpha]\tau'$

The first is trivially true, since $k' = S(c : k)$, since singleton is commutative.

For the second, lemma:
If $\Gamma \vdash \sigma : \mathtt{sig}$, $\Gamma \vdash c : k$, $\sigma \rightsquigarrow \alpha : k \,.\, \tau$,
then, $\Gamma, \alpha : k' \vdash \tau \equiv \tau' : \mathrm{T}$, $\Gamma \vdash [c/\alpha]\tau \equiv [c/\alpha]\tau : \mathrm{T}$

$$\frac{\alpha/s : \sigma \in \Gamma}{\Gamma \vDash_{\mathrm{P}} s \implies S(\alpha : \sigma) \rightsquigarrow [\alpha\ ,\ s]}$$

$$\frac{\Gamma \vDash_{\mathrm{P}} M : \sigma \rightsquigarrow [c\ ,\ e] \qquad \Gamma \vdash \sigma \le \sigma' a \rightsquigarrow f}{\Gamma \vDash_{\mathrm{P}} M : \sigma' \rightsquigarrow [c\ ,\ f[c]\ e]}$$

New judgement to construct the above:
If $\Gamma \vdash \sigma \le \sigma' \rightsquigarrow f$, $\sigma \rightsquigarrow \alpha : k \,.\, \tau$, $\sigma' \rightsquigarrow \alpha : k' \,.\, \tau'$, then

$$\frac{\Gamma \vdash \sigma \equiv \sigma' : \mathtt{sig} \qquad \sigma \rightsquigarrow \alpha : k \,.\, \tau}{\Gamma \vdash \sigma \le \sigma' \rightsquigarrow \Lambda\alpha : k \,.\, \lambda x : \tau \,.\, x} \qquad \frac{\Gamma \vdash \sigma_1 \le \sigma_2 \rightsquigarrow f_1 \qquad \Gamma \vdash \sigma_2 \le \sigma_3 \rightsquigarrow f_1 \qquad \sigma_1 \rightsquigarrow \alpha : k_1 \,.\, \tau_1}{\Gamma \vdash \sigma_1 \le \sigma_3 \rightsquigarrow \Lambda\alpha : k_1 \,.\, \lambda x : \tau_1 \,.\, f_2[\alpha]\ f_1[\alpha]\ x}$$

$$\frac{\Gamma \vdash k \le k'}{\Gamma \vdash (\!| k |\!) \le (\!| k' |\!) \rightsquigarrow \Lambda\alpha : k \,.\, \lambda x : \mathrm{unit} \,.\, x}$$

$$\frac{\Gamma \vdash \sigma_1' \le \sigma_1 \rightsquigarrow f_1 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1') \vdash \sigma_2 \le \sigma_2' \rightsquigarrow f_2 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1) \vdash \sigma_2 : \mathtt{sig}}{\begin{array}{c}\Gamma \vdash \Pi^{\mathsf{gen}}\alpha : \sigma_1 \,.\, \sigma_2 \le \Pi^{\mathsf{gen}}\alpha : \sigma_1' \,.\, \sigma_2' \rightsquigarrow \\ \Lambda_{\_} : 1 \,.\, \lambda f : \forall\alpha : k_1 \,.\, [\alpha/\alpha_1]\tau_1 \to \exists\alpha_2 : k_2 \,.\, \tau_2 \,.\, \Lambda\alpha : k_1' \,.\, \lambda x : [\alpha/\alpha_1]\tau_1' \,.\, \mathtt{unpack}\ [\alpha_2, y] = f[\alpha]\ f_1[\alpha]\ x\ \mathtt{in\ pack}\ [\alpha_2\ ,\ f_2[\alpha_2]\ y\end{array}}$$

$$\frac{\Gamma \vdash \sigma_1' \le \sigma_1 \rightsquigarrow f_1 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1') \vdash \sigma_2 \le \sigma_2' \rightsquigarrow f_2 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1) \vdash \sigma_2 : \mathtt{sig}}{\begin{array}{c}\Gamma \vdash \Pi^{\mathsf{app}}\alpha : \sigma_1 \,.\, \sigma_2 \le \Pi^{\mathsf{app}}\alpha : \sigma_1' \,.\, \sigma_2' \rightsquigarrow \\ \Lambda\beta : \Pi\alpha : k_1 \,.\, k_2 \,.\, \lambda f : \forall\alpha : k_1 \,.\, [\alpha/\alpha_1]\tau_1 \to [\beta\ \alpha/\alpha_2]\tau_2 \,.\, \Lambda\alpha : k_1' \,.\, \lambda x : [\alpha/\alpha_1]\tau_1' \,.\, f_2[\beta\ \alpha]\ f[\alpha]\ f_1[\alpha]\ x\end{array}}$$

$$\frac{\Gamma \vdash \sigma_1 \le \sigma_1' \rightsquigarrow f_1 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1) \vdash \sigma_2 \le \sigma_2' \rightsquigarrow f_2 \qquad \Gamma, \alpha : \mathtt{Fst}(\sigma_1) \vdash \sigma_2 : \mathtt{sig}}{\begin{array}{c}\Gamma \vdash \Sigma\alpha : \sigma_1 \,.\, \sigma_2 \le \Sigma\alpha : \sigma_1' \,.\, \sigma_2' \rightsquigarrow \\ \Lambda\beta : \sigma\alpha : k_1 \,.\, k_2 \,.\, \lambda x : [\pi_1\beta/\alpha_1]\tau_1 \times [\pi_1\beta, \pi_2\beta/\alpha_1, \alpha_2]\tau_2 \,.\, \langle f_1[\pi_1\beta]\ \pi_0^* x, [\pi_1\beta/\alpha]f_2[\pi_2\beta]\ \pi_1^* x\rangle\end{array}}$$

$$\frac{\Gamma \vDash_{\mathrm{P}} M : \sigma \rightsquigarrow [c\ ,\ e] \qquad \Gamma \vdash \sigma \le \sigma' \qquad \sigma \rightsquigarrow \alpha : k \,.\, \tau \qquad \sigma' \rightsquigarrow \alpha : k' \,.\, \tau' \qquad \Gamma, \alpha : k \vdash \tau \equiv \tau'}{\Gamma \vDash_{\mathrm{P}} M : \sigma' \rightsquigarrow [c\ ,\ e]}$$

14.6 TODO

Now let's look at sealing

$$\frac{\Gamma \vDash_{\mathrm{I}} M \Leftarrow \sigma}{\Gamma \vDash_{\mathrm{I}} M :> \sigma \Rightarrow \sigma \rightsquigarrow e} \qquad \frac{\Gamma \vDash_{\mathrm{I}} M \Rightarrow \sigma' \rightsquigarrow e \qquad \Gamma \vdash \sigma' \triangleq \sigma}{\Gamma \vDash_{\mathrm{I}} M \Leftarrow \sigma \rightsquigarrow e}$$

Coding Note (the SIMPLE way):

```
translatePure   : context -> module -> con * con * term * sg
translateImpure : context -> module -> term * sg
```

The better way:

```
datatype result = con * con * term * sg | term * sg
translate : context -> module -> result
```

14.7 Named Signatures

14.7.1 Target Language

$$\texttt{name} ::= \ldots$$
$$\sigma ::= \cdots \mid \texttt{name} : \sigma$$

$$\texttt{Fst}(\texttt{name} : \sigma) = \texttt{Fst}(\sigma)$$
$$\mathcal{S}(c : \texttt{name} : \sigma) = \texttt{name} : \mathcal{S}(c : \sigma)$$

14.7.2 Typing Judgements

$$\frac{\Gamma \vdash \sigma : \texttt{sig}}{\Gamma \vdash \texttt{name} : \sigma : \texttt{sig}} \qquad \frac{\Gamma \models_\kappa M : \sigma}{\Gamma \models_\kappa \texttt{in}_{\texttt{name}}\ M : (\texttt{name} : \sigma)} \qquad \frac{\Gamma \models_\kappa M : (\texttt{name} : \sigma)}{\Gamma \models_\kappa \texttt{out}\ M : \sigma}$$

$$\frac{\Gamma \vdash \texttt{Fst}(M) \gg c}{\Gamma \vdash \texttt{Fst}(\texttt{in}_{\texttt{name}}\ M) \gg c \qquad \Gamma \vdash \texttt{Fst}(\texttt{out}\ M) \gg c} \qquad \frac{\Gamma \models_{\mathrm{P}} M : \texttt{name} : \sigma' \qquad \Gamma \models_{\mathrm{P}} \texttt{out}\ M : \sigma}{\Gamma \models_{\mathrm{P}} M : \texttt{name} : \sigma}$$

$$\frac{\Gamma \models_\kappa M \Rightarrow \sigma}{\Gamma \models_\kappa \texttt{in}_M\ : \texttt{name} : \sigma} \qquad \frac{\Gamma \models_\kappa M \Rightarrow \texttt{name} : \sigma}{\Gamma \models_\kappa \texttt{out}\ M \Rightarrow \sigma} \qquad \frac{\sigma \rightsquigarrow [\alpha : k\ ,\ \tau]}{\texttt{name} : \sigma \rightsquigarrow [\alpha : k\ ,\ \tau]}$$

$$\frac{\Gamma \models_{\mathrm{P}} M \Rightarrow \sigma' \rightsquigarrow [c\ ,\ e] \qquad \cancel{\Gamma \vdash \sigma' \trianglelefteq \sigma} \qquad \cancel{\sigma \rightsquigarrow \alpha : k\ .\ \tau} \qquad \cancel{\sigma' \rightsquigarrow \alpha : k'\ .\ \tau'} \qquad \cancel{\Gamma, \alpha : k' \vdash \tau \equiv \tau' : \mathrm{T}} \qquad \Gamma \vdash \sigma' \leqq \sigma}{\Gamma \models_{\mathrm{P}} M \Rightarrow \sigma \rightsquigarrow [c\ ,\ e]}$$

And now we need to add in the appropriate rules

14.7.3 $\Gamma \vdash \sigma \leqq \sigma'$

$$\frac{}{\Gamma \vdash 1 \leqq 1} \qquad \frac{}{\Gamma \vdash (\!| k |\!) \leqq (\!| k' |\!)} \qquad \frac{}{\Gamma \vdash \langle\!| \tau |\!\rangle \leqq \langle\!| \tau |\!\rangle} \qquad \frac{\Gamma \vdash \sigma_1 \leqq \sigma_1' \qquad \Gamma, \alpha : \texttt{Fst}(\sigma_1) \vdash \sigma_2 \leqq \sigma_2'}{\Gamma \vdash \Sigma\alpha : \sigma_1\ .\ \sigma_2 \leqq \Sigma\alpha : \sigma_1'\ .\ \sigma_2'}$$

$$\frac{\Gamma \vdash \Pi^{\texttt{gen}}\alpha : \sigma_1\ .\ \sigma_2 \equiv \Pi^{\texttt{gen}}\alpha : \sigma_1'\ .\ \sigma_2'}{\Gamma \vdash \Pi^{\texttt{gen}}\alpha : \sigma_1\ .\ \sigma_2 \leqq \Pi^{\texttt{gen}}\alpha : \sigma_1'\ .\ \sigma_2'}$$

15 GC

1. Mark and Sweep
2. Copying
3. Reference Counting

For the first two, the idea is that we have pointers to objects in the heap called the "roots". These objects have pointers to other objects. Accessible objects are objects we can encounter by following some set of pointers from the roots. Goal is to keep accessible objects and get rid of the rest.

Reference counting cannot deal with circular pointers. So we don't even want to talk about it really.

Copying has the best asymptotic time (only traverse accessible objects).
Mark is just as good, but the sweep phase is too expensive (the whole heap).

Spacial locality is a huge benefit of copying (because defragmentation).
Copying sacrifices about half of the heap space.
Copying also has trouble with “mutation” (we will talk about this later).
Mark-sweep supports pinning (eg: C#) (for external devices that cannot deal with copying).

15.1 Copying

Stop-the-World/Incremental/Concurrent
Stop-the-World is easiest, and other than its undesirably behavior, it has no potential memory-leak problems.

Semispace/Generational
EG: 2-generation generational copying collector
Make 3 segments, ”Minor”, ”Major”, and ”2-Space”
Algo: we allocate in minor until we run out; collect into the major space
The reason this works is because objects in the major heap cannot point to objects in the minor heap in the absence of mutation, which we have in our functional languages.
Once we run out of memory in the major space, we collect it into the 2-space, which then becomes the major space, while the old major space becomes the 2-space.
Note that the Major-collection is certainly more expensive.

This has a major limitation when we DO have mutation. We lose most of the performance if we have to traverse everything in the end.
Thus, to fix this, we maintain a reference table where we maintain pointers to everything we assign. Everytime we make the assignment, we will need to update this table. Each generation other than the first gets its own reference table.

Another problem: Pointer identification
“Conservative”, assume it’s a pointer and trace, unless it’s definitely not, but this does not work at all with copying (because value must change).
“Tag bit”, tag each object with a value that decides whether or not the value is a pointer. Downside: size of int now becomes 31 instead of 32.
“Tag word”, header word has bitmask that tells what values are pointers.
“Tag free”, use the types to check whether or not something is traceable.

with “tag bit”, But then how do we get Int32? “boxing” (pass-by-reference)

15.2 Cheney-Scan

Stop-the-world semispace copy-and-collect garbage collector that we will be implementing.

15.2.1 Heap Traversal

white = unvisited
black = visited
gray = partially visited

Initially, gray the root. Then, color all of the nodes reachable from the root also gray, and once we do, color the root black.
Gray nodes are ones that are copied to the to space.
Black nodes are ones whose members are also copied to the to space.
Be careful when you need to gray a node that is already grayed. (eg: just update the pointer etc.)
Done with collection when have nothing but black and white nodes.
Note: Cheney-Scan happens to be BFS, but DFS may actually provide better locality.

Tri-color Invariant:

- no edges from black to white
- any reachable white object is reachable from a gray object

15.2.2 Implementation

AP "Allocation Pointer"
LP "Limit Pointer"

When mutator asks for n words, increment AP by (n + 1), first one being the H "Header", and then the n allocated words after it.
Before the incrementing, check if ap + (n + 1) > lp, and if so, gc.

On gc, set AP and LP to the ends of the to-space.
Then, copy over roots.
Then, copy over nodes connected to those roots.
Then, copy over nodes connected to those nodes, etc.
Define gray nodes to be all nodes between TP and AP, where TP "Trace Pointer" is maintained appropriately, and is incremented as we black away a node.
Define black nodes to be all nodes between start of to space and TP.

16 Elaboration

1. ~~Syntatic~~ Semantic Sugar
2. Nonsense
    - open
    - include
3. no $\underline{\text{good}}$ type theory yet
    - avoidance problem
    - pattern matching

16.1 Target Language

$$\begin{aligned}
exp &::= \\
dec &::= \\
ty &::= \\
pat &::= \\
match &::= \\
mod &::= \\
sig &::= \\
spec &::= \\
id &::=
\end{aligned}$$

Identifier Resolution is an important topic (where did the 'x' come from).

16.2 Judgement

If $\Gamma \vdash M_{ec} : \sigma_{ec} \triangleright exp \rightsquigarrow e : \tau$ and $\vdash \Gamma\,\mathrm{ok}$, $\Gamma \vdash M_{ec} : \sigma_{ec}$, then $\Gamma \vdash e : \tau$
$\Gamma \vdash M_{ec} : \sigma_{ec} \triangleright id \rightsquigarrow e : M : \sigma$

Also, extend IL-Module with names (and certain "namespaces").

$$\begin{aligned}
\sigma &::= \cdots \mid \texttt{name} : \sigma \\
\texttt{name} &::= \texttt{VAL}\ \ id \mid \texttt{CON}\ \ id \mid \texttt{MOD}\ \ id \mid \texttt{FUN}\ \ id \mid \ldots
\end{aligned}$$

eg: $M : \Sigma\alpha : \sigma_1 \,.\, \texttt{VAL}\ foo : \sigma_2 \triangleright \texttt{VAL}\ foo \rightsquigarrow \texttt{out}\ \pi_2 M$