This is the Project Title

Your Name

Master of Science
Computer Science
School of Informatics
University of Edinburgh
2020

Abstract

Formal development of Frank.

Chapter 1

Formalisation of Frank

(data types)	D
(value type variables)	X
(effect type variables)	E
(value types)	$A,B ::= D\,\overline{R}$
	$\mid\ \{C\} \mid X$
(computation types)	$C ::= \overline{T \to} \ G$
(argument types)	$T ::= \langle\Delta\rangle A$
(return types)	$G ::= [\Sigma]A$
(type binders)	$Z ::= X \mid [E]$
(type arguments)	$R ::= A \mid [\Sigma]$
(polytypes)	$P ::= \forall\overline{Z}.A$

(interfaces)	I
(term variables)	x,y,z,f
(instance variables)	s,a,b,c
(seeds)	$\sigma ::= \emptyset \mid E$
(abilities)	$\Sigma ::= \sigma \mid \Xi$
(extensions)	$\Xi ::= \iota \mid \Xi, I\,\overline{R}$
(adaptors)	$\Theta ::= \iota \mid \Theta, I(S \to S')$
(adjustments)	$\Delta ::= \Theta \mid \Xi$
(instance patterns)	$S ::= s \mid S\ a$
(kind environments)	$\Phi,\Psi ::= \cdot \mid \Phi, Z$
(type environments)	$\Gamma ::= \cdot \mid \Gamma, x:A \mid \Gamma, f:P$
(instance environments)	$\Omega ::= s:\Sigma \mid \Omega, a: I\,\overline{R}$

Figure 1.1: Types

(constructors)	k
(commands)	c
(uses)	$m ::= x \mid f\,\overline{R} \mid m\,\overline{n} \mid \uparrow(n:A)$
(constructions)	$n ::= \downarrow m \mid k\,\overline{n} \mid c\,\overline{R}\,\overline{n} \mid \{e\}$
	$\mid\ \textbf{let}\ f:P = n\ \textbf{in}\ n' \mid \textbf{letrec}\ \overline{f:P = e}\ \textbf{in}\ n$
	$\mid\ \langle\Theta\rangle\ n$
(computations)	$e ::= \overline{\overline{r} \mapsto n}$
(computation patterns)	$r ::= p \mid \langle c\,\overline{p} \to z\rangle \mid \langle x\rangle$
(value patterns)	$p ::= k\,\overline{p} \mid x$

Figure 1.2: Terms

$$\boxed{\Phi;\Gamma\,[\Sigma]\!\vdash m \Rightarrow A}$$

$$\textsc{T-Var}\quad \frac{x : A \in \Gamma}{\Phi;\Gamma\,[\Sigma]\!\vdash x \Rightarrow A} \qquad\qquad \textsc{T-PolyVar}\quad \frac{\Phi \vdash \overline{R} \qquad f : \forall \overline{Z}.A \in \Gamma}{\Phi;\Gamma\,[\Sigma]\!\vdash f\;\overline{R} \Rightarrow A[\overline{R}/\overline{Z}]}$$

$$\textsc{T-App}\quad \frac{\begin{array}{c}\Sigma' = \Sigma \qquad (\Sigma \vdash \Delta_i \dashv \Sigma'_i)_i \\ \Phi;\Gamma\,[\Sigma]\!\vdash m \Rightarrow \{\overline{\langle\Delta\rangle A \to}\; [\Sigma']B\} \qquad (\Phi;\Gamma\,[\Sigma'_i]\!\vdash n_i : A_i)_i\end{array}}{\Phi;\Gamma\,[\Sigma]\!\vdash m\;\overline{n} \Rightarrow B} \qquad\qquad \textsc{T-Ascribe}\quad \frac{\Phi;\Gamma\,[\Sigma]\!\vdash n : A}{\Phi;\Gamma\,[\Sigma]\!\vdash \uparrow(n : A) \Rightarrow A}$$

$$\boxed{\Phi;\Gamma\,[\Sigma]\!\vdash n : A}$$

$$\textsc{T-Switch}\quad \frac{\Phi;\Gamma\,[\Sigma]\!\vdash m \Rightarrow A \qquad A = B}{\Phi;\Gamma\,[\Sigma]\!\vdash \downarrow m : B} \qquad\qquad \textsc{T-Data}\quad \frac{k\;\overline{A} \in D\;\overline{R} \qquad (\Phi;\Gamma\,[\Sigma]\!\vdash n_j : A_j)_j}{\Phi;\Gamma\,[\Sigma]\!\vdash k\;\overline{n} : D\;\overline{R}}$$

$$\textsc{T-Command}\quad \frac{\Phi \vdash \overline{R} \qquad c : \forall \overline{Z}.\overline{A \to}\; B \in \Sigma \qquad (\Phi;\Gamma\,[\Sigma]\!\vdash n_j : A_j[\overline{R}/\overline{Z}])_j}{\Phi;\Gamma\,[\Sigma]\!\vdash c\;\overline{R}\;\overline{n} : B[\overline{R}/\overline{Z}]} \qquad\qquad \textsc{T-Thunk}\quad \frac{\Phi;\Gamma \vdash e : C}{\Phi;\Gamma\,[\Sigma]\!\vdash \{e\} : \{C\}}$$

$$\textsc{T-Let}\quad \frac{\begin{array}{c}P = \forall \overline{Z}.A \\ \Phi,\overline{Z};\Gamma\,[\emptyset]\!\vdash n : A \qquad \Phi;\Gamma, f : P\,[\Sigma]\!\vdash n' : B\end{array}}{\Phi;\Gamma\,[\Sigma]\!\vdash \mathbf{let}\; f : P = n\;\mathbf{in}\; n' : B}$$

$$\textsc{T-LetRec}\quad \frac{\begin{array}{c}(P_i = \forall \overline{Z}_i.\{C_i\})_i \\ (\Phi,\overline{Z}_i;\Gamma,\overline{f : P} \vdash e_i : C)_i \qquad \Phi;\Gamma,\overline{f : P}\,[\Sigma]\!\vdash n : B\end{array}}{\Phi;\Gamma\,[\Sigma]\!\vdash \mathbf{letrec}\;\overline{f : P = e}\;\mathbf{in}\; n : B} \qquad\qquad \textsc{T-Adapt}\quad \frac{\Sigma \vdash \Theta \dashv \Sigma' \qquad \Phi;\Gamma\,[\Sigma']\!\vdash n : A}{\Phi;\Gamma\,[\Sigma]\!\vdash \langle\Theta\rangle\; n : A}$$

$$\boxed{\Phi;\Gamma \vdash e : C}$$

$$\textsc{T-Comp}\quad \frac{\begin{array}{c}(\Phi \vdash r_{i,j} : T_j \dashv\!\![\Sigma]\; \exists \Psi_{i,j}.\Gamma'_{i,j})_{i,j} \\ (\Phi,(\Psi_{i,j})_j;\Gamma,(\Gamma'_{i,j})_j\,[\Sigma]\!\vdash n_i : B)_i \qquad ((r_{i,j})_i \text{ covers } T_j)_j\end{array}}{\Phi;\Gamma \vdash ((r_{i,j})_j \mapsto n_i)_i : (T_j \to)_j\; [\Sigma]B}$$

Figure 1.3: Term Typing Rules

$$
\begin{array}{lrl}
\text{(uses)} & m & ::= \cdots \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \\
\text{(constructions)} & n & ::= \cdots \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \\
\text{(use values)} & u & ::= x \mid f\,\overline{R} \mid \uparrow(v : A) \\
\text{(non-use values)} & v & ::= k\,\overline{w} \mid \{e\} \\
\text{(construction values)} & w & ::= \downarrow u \mid v \\
\text{(normal forms)} & t & ::= w \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \\
\text{(evaluation frames)} & \mathcal{F} & ::= [\,]\,\overline{n} \mid u\ (\bar{t}, [\,], \overline{n}) \mid \uparrow([\,] : A) \\
 & & \mid \ \downarrow[\,] \mid k\ (\overline{w}, [\,], \overline{n}) \mid c\,\overline{R}\ (\overline{w}, [\,], \overline{n}) \\
 & & \mid \ \mathbf{let}\ f : P = [\,]\ \mathbf{in}\ n \mid \langle \Theta \rangle\ [\,] \\
\text{(evaluation contexts)} & \mathcal{E} & ::= [\,] \mid \mathcal{F}[\mathcal{E}]
\end{array}
$$

Figure 1.4: Runtime Syntax

$\boxed{\Phi; \Gamma\,[\Sigma]\!\vdash m \Rightarrow A}$ $\boxed{\Phi; \Gamma\,[\Sigma]\!\vdash n : A}$

$$
\textsc{T-Freeze-Use}\quad \frac{\neg(\mathcal{E}\ \mathsf{handles}\ c) \qquad \Phi; \Gamma\,[\Sigma]\!\vdash \mathcal{E}[c\,\overline{R}\,\overline{w}] \Rightarrow A}{\Phi; \Gamma\,[\Sigma]\!\vdash \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \Rightarrow A}
$$

$$
\textsc{T-Freeze-Cons}\quad \frac{\neg(\mathcal{E}\ \mathsf{handles}\ c) \qquad \Phi; \Gamma\,[\Sigma]\!\vdash \mathcal{E}[c\,\overline{R}\,\overline{w}] : A}{\Phi; \Gamma\,[\Sigma]\!\vdash \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil : A}
$$

Figure 1.5: Frozen Commands

$$\boxed{m \rightsquigarrow_{\mathrm{u}} m'} \quad \boxed{n \rightsquigarrow_{\mathrm{c}} n'} \quad \boxed{m \longrightarrow_{\mathrm{u}} m'} \quad \boxed{n \longrightarrow_{\mathrm{c}} n'}$$

$$\textsc{R-Handle}\quad \frac{k = \min_i \{i \mid \exists \overline{\theta}.(r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv\![\Sigma]\ \theta_j)_j\} \qquad (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv\![\Sigma]\ \theta_j)_j}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t} \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)}$$

$$\textsc{R-Ascribe-Use}\quad \frac{}{\uparrow(\downarrow u : A) \rightsquigarrow_{\mathrm{u}} u} \qquad \textsc{R-Ascribe-Cons}\quad \frac{}{\downarrow\uparrow(w : A) \rightsquigarrow_{\mathrm{c}} w} \qquad \textsc{R-Let}\quad \frac{}{\mathbf{let}\ f : P = w\ \mathbf{in}\ n \rightsquigarrow_{\mathrm{c}} n[\uparrow(w : P)/f]}$$

$$\textsc{R-LetRec}\quad \frac{\overline{e = \overline{\overline{r} \to n}}}{\mathbf{letrec}\ \overline{f : P = e}\ \mathbf{in}\ n' \rightsquigarrow_{\mathrm{c}} n'\overline{[\uparrow(\{\overline{\overline{r} \to \mathbf{letrec}\ \overline{f : P = e}\ \mathbf{in}\ n}\} : P)/f]}} \qquad \textsc{R-Adapt}\quad \frac{}{\langle \Theta \rangle\ w \rightsquigarrow_{\mathrm{c}} w}$$

$$\textsc{R-Freeze-Comm}\quad \frac{}{c\ \overline{R}\ \overline{w} \rightsquigarrow_{\mathrm{c}} \lceil c\ \overline{R}\ \overline{w} \rceil}$$

$$\textsc{R-Freeze-Frame-Use}\quad \frac{\neg(\mathcal{F}[\mathcal{E}]\ \text{handles}\ c)}{\mathcal{F}[\lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil] \rightsquigarrow_{\mathrm{u}} \lceil \mathcal{F}[\mathcal{E}[c\ \overline{R}\ \overline{w}]] \rceil} \qquad \textsc{R-Freeze-Frame-Cons}\quad \frac{\neg(\mathcal{F}[\mathcal{E}]\ \text{handles}\ c)}{\mathcal{F}[\lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil] \rightsquigarrow_{\mathrm{c}} \lceil \mathcal{F}[\mathcal{E}[c\ \overline{R}\ \overline{w}]] \rceil}$$

$$\textsc{R-Lift-UU}\quad \frac{m \rightsquigarrow_{\mathrm{u}} m'}{\mathcal{E}[m] \longrightarrow_{\mathrm{u}} \mathcal{E}[m']} \qquad \textsc{R-Lift-UC}\quad \frac{m \rightsquigarrow_{\mathrm{u}} m'}{\mathcal{E}[m] \longrightarrow_{\mathrm{c}} \mathcal{E}[m']} \qquad \textsc{R-Lift-CU}\quad \frac{n \rightsquigarrow_{\mathrm{c}} n'}{\mathcal{E}[n] \longrightarrow_{\mathrm{u}} \mathcal{E}[n']} \qquad \textsc{R-Lift-CC}\quad \frac{n \rightsquigarrow_{\mathrm{c}} n'}{\mathcal{E}[n] \longrightarrow_{\mathrm{c}} \mathcal{E}[n']}$$

Figure 1.6: Operational Semantics

$\boxed{r : T \leftarrow t \dashv[\Sigma]\ \theta}$

$$\textsc{B-Value}\quad \frac{\Sigma \vdash \Delta \dashv \Sigma' \qquad p : A \leftarrow w \dashv \theta}{p : \langle\Delta\rangle A \leftarrow w \dashv[\Sigma]\ \theta}$$

$$\textsc{B-Request}\quad \frac{\begin{array}{c}\Sigma \vdash \Delta \dashv \Sigma' \qquad \mathcal{E}\ \mathsf{poisedfor}\ c \\ \Delta = \Theta \mid \Xi \qquad c : \forall \overline{Z}.\overline{B \to}\ B' \in \Xi \qquad (p_i : B_i \leftarrow w_i \dashv \theta_i)_i\end{array}}{\langle c\ \overline{p} \to z\rangle : \langle\Delta\rangle A \leftarrow \lceil\mathcal{E}[c\ \overline{R}\ \overline{w}]\rceil \dashv[\Sigma]\ \overline{\theta}[\uparrow(\{x \mapsto \mathcal{E}[x]\} : \{B' \to [\Sigma']A\})/z]}$$

$$\textsc{B-CatchAll-Value}\quad \frac{\Sigma \vdash \Delta \dashv \Sigma'}{\langle x\rangle : \langle\Delta\rangle A \leftarrow w \dashv[\Sigma]\ [\uparrow(\{w\}{:}\{[\Sigma']A\})/x]}$$

$$\textsc{B-CatchAll-Request}\quad \frac{\begin{array}{c}\Sigma \vdash \Delta \dashv \Sigma' \qquad \mathcal{E}\ \mathsf{poisedfor}\ c \\ \Delta = \Theta \mid \Xi \qquad c : \forall \overline{Z}.\overline{B \to}\ B' \in \Xi\end{array}}{\langle x\rangle : \langle\Delta\rangle A \leftarrow \lceil\mathcal{E}[c\ \overline{R}\ \overline{w}]\rceil \dashv[\Sigma]\ [\uparrow(\{\lceil\mathcal{E}[c\ \overline{R}\ \overline{w}]\rceil\}{:}\{[\Sigma']A\})/x]}$$

$\boxed{p : A \leftarrow w \dashv \theta}$

$$\textsc{B-Var}\quad \frac{}{x : A \leftarrow w \dashv [\uparrow(w : A)/x]} \qquad\qquad \textsc{B-Data}\quad \frac{k\ \overline{A} \in D\ \overline{R} \qquad (p_i : A_i \leftarrow w_i \dashv \theta_i)_i}{k\ \overline{p} : D\ \overline{R} \leftarrow k\ \overline{w} \dashv \overline{\theta}}$$

Figure 1.7: Pattern Binding

Chapter 2

Arbitrary Thread Interruption

2.1 Relaxing Catches

$$\frac{\Sigma \vdash \Delta \dashv \Sigma'}{\langle x \rangle : \langle \Delta \rangle A \leftarrow \lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil \dashv_{[\Sigma]} [\uparrow(\{\lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil\}{:}\{[\Sigma']A\})/x]}\ \textsc{B-CatchAll-Request-Loose}$$

Figure 2.1: Updated B-CATCHALL-REQUEST

2.2 Interrupting Arbitrary Terms

$$
\begin{array}{lrl}
\text{(uses)} & m & ::= \cdots \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \\
\text{(constructions)} & n & ::= \cdots \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \\
\text{(use values)} & u & ::= x \mid f\,\overline{R} \mid \uparrow(v : A) \\
\text{(non-use values)} & v & ::= k\,\overline{w} \mid \{e\} \\
\text{(construction values)} & w & ::= \downarrow u \mid v \\
\text{(normal forms)} & t & ::= w \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \mid !(m) \\
\text{(evaluation frames)} & \mathcal{F} & ::= [\,]\,\overline{n} \mid u\,(\overline{t},[\,],\overline{n}) \mid \uparrow([\,] : A) \\
 & & \mid\ \ \downarrow[\,] \mid k\,(\overline{w},[\,],\overline{n}) \mid c\,\overline{R}\,(\overline{w},[\,],\overline{n}) \\
 & & \mid\ \ \mathbf{let}\ f : P = [\,]\ \mathbf{in}\ n \mid \langle\Theta\rangle\,[\,] \\
\text{(evaluation contexts)} & \mathcal{E} & ::= [\,] \mid \mathcal{F}[\mathcal{E}]
\end{array}
$$

Figure 2.2: Runtime Syntax, Updated with Suspension of Uses

$$
\textsc{B-CatchAll-Interrupt}\quad
\frac{\Sigma \vdash \Delta \dashv \Sigma'}{\langle x \rangle : \langle\Delta\rangle A \leftarrow !(m) \dashv\!\![\Sigma]\ [\uparrow(\{m\}:\{[\Sigma']A\})/x]}
$$

Figure 2.3: Catching Interrupts rule.

2.3 Freezing

Another way of doing it is to just let any use become frozen, in the same way as commands become frozen once invoked.

TODO: Do we need the 'hoisting' rules for general suspended terms?

2.4 Yielding

Note that R-YIELD-EF relies on the predicate $\mathcal{F}[\mathcal{E}]$ allows c. This states that the Yield interface is in the ability of the term in the evaluation context.

For any frame apart from argument frames, $\mathcal{F}[\mathcal{E}]$ allows c = false. In this case, it is defined as follows;

$$
\uparrow(v : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\overline{t},[\,],\overline{n})\ \text{allows}\ c = \Sigma'\ \text{allows}\ c \quad \text{where}\ \Sigma \vdash \Delta_{|\overline{t}|} \dashv \Sigma'
$$

TODO: What happens for 0-ary constructors? I don't know if it's exactly how I think it is.

$$\frac{\textsc{R-Interrupt}}{m \rightsquigarrow_{\mathrm{u}} !(m)}$$

Figure 2.4: Use interruption rule

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$ $\boxed{n \rightsquigarrow_{\mathrm{c}} n'}$ $\boxed{m \longrightarrow_{\mathrm{u}} m'}$ $\boxed{n \longrightarrow_{\mathrm{c}} n'}$

$$\textsc{R-Freeze-Use}\quad \frac{}{m \rightsquigarrow_{\mathrm{u}} \lceil m \rceil} \qquad \textsc{R-Freeze-Comm}\quad \frac{}{c\,\overline{R}\,\overline{w} \rightsquigarrow_{\mathrm{c}} \lceil c\,\overline{R}\,\overline{w} \rceil}$$

$$\textsc{R-Freeze-Frame-Use}\quad \frac{}{\mathcal{F}[\lceil m \rceil] \rightsquigarrow_{\mathrm{u}} \lceil \mathcal{F}[m] \rceil} \qquad \textsc{R-Freeze-Frame-Cons}\quad \frac{}{\mathcal{F}[\lceil m \rceil] \rightsquigarrow_{\mathrm{c}} \lceil \mathcal{F}[m] \rceil}$$

$$\textsc{R-Freeze-Frame-Use}\quad \frac{\neg(\mathcal{F}[\mathcal{E}] \text{ handles } c)}{\mathcal{F}[\lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil] \rightsquigarrow_{\mathrm{u}} \lceil \mathcal{F}[\mathcal{E}[c\,\overline{R}\,\overline{w}]] \rceil} \qquad \textsc{R-Freeze-Frame-Cons}\quad \frac{\neg(\mathcal{F}[\mathcal{E}] \text{ handles } c)}{\mathcal{F}[\lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil] \rightsquigarrow_{\mathrm{c}} \lceil \mathcal{F}[\mathcal{E}[c\,\overline{R}\,\overline{w}]] \rceil}$$

Figure 2.5: Updated Freezing

For an ability $\Sigma = \sigma \mid \Xi$, Σ allows c is true if $c \in I$ for some $I \in \Xi$.

Informally, $\mathcal{E}$ allows c is true when $\mathcal{E}$ is a handler where the command c is a member of an interface in its ability when modified by the adaptor at the corresponding position.

TODO: Clean this up

We also make use of an auxilary combinator $\_;\_$. This is the traditional sequential composition operator, where both arguments are evaluated and hte result of the second one is returned. In the context of R-YIELD-EF this means we will perform the yield effect and then the use m, but discard the result from yield.

2.5 Counting

In practise we count up through the amount of R-HANDLE rules we apply and only insert the yield when this count exceeds a threshold value t_y.

So we supplement the operational semantics with a *counter* c_y, so that our transi-

$$
\begin{array}{lrl}
\text{(uses)} & m & ::= \ldots \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \mid \boxed{\lceil m \rceil} \\
\text{(constructions)} & n & ::= \ldots \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \mid \boxed{\lceil m \rceil} \\
\text{(use values)} & u & ::= x \mid f\,\overline{R} \mid \uparrow(v : A) \\
\text{(non-use values)} & v & ::= k\,\overline{w} \mid \{e\} \\
\text{(construction values)} & w & ::= \downarrow u \mid v \\
\text{(normal forms)} & t & ::= w \mid \lceil \mathcal{E}[c\,\overline{R}\,\overline{w}] \rceil \mid \boxed{\lceil m \rceil} \\
\text{(evaluation frames)} & \mathcal{F} & ::= [\,]\,\overline{n} \mid u\,(\overline{t},[\,],\overline{n}) \mid \uparrow([\,] : A) \\
 & & \mid\ \ \downarrow[\,] \mid k\,(\overline{w},[\,],\overline{n}) \mid c\,\overline{R}\,(\overline{w},[\,],\overline{n}) \\
 & & \mid\ \ \textbf{let}\ f : P = [\,]\ \textbf{in}\ n \mid \langle \Theta \rangle\,[\,] \\
\text{(evaluation contexts)} & \mathcal{E} & ::= [\,] \mid \mathcal{F}[\mathcal{E}]
\end{array}
$$

Figure 2.6: Runtime Syntax, Updated with Freezing of Uses

$$
\frac{\Sigma \vdash \Delta \dashv \Sigma'}{\langle x \rangle : \langle \Delta \rangle A \leftarrow \lceil m \rceil \dashv_{\Sigma} [\uparrow(\{m\}:\{[\Sigma']A\})/x]}\ \textsc{B-CatchAll-Interrupt}
$$

Figure 2.7: Catching Frozen Terms rule.

tions are e.g. $m; c_y \rightsquigarrow_{\mathrm{u}} m'; c_y{}'$. We adopt the convention that, when the counter is not mentioned in a transition[1], the counter stays the same. Hence e.g. $m \rightsquigarrow_{\mathrm{u}} m'$ desugars to $m; c_y \rightsquigarrow_{\mathrm{u}} m'; c_y$.

So to get our counting semantics we just need to supplement Figure [**?**] with the updated rule in

2.6 Counting — Take Two

In the final semantics we count up through the number of R-HANDLE uses. This lets us track when to next insert a yield.

We use a counter for this, labelled c_y in the semantics. This counter essentially has two states; it is either simply counting, i.e. is $c(n)$ for some n or is a message to yield as soon as possible, i.e. yield.

To increment this counter, we use a slightly modified version of addition, denoted $\oplus$. This is simply defined as

[1]That is, the transition is of the form $m \rightsquigarrow_{\mathrm{u}} m'$.

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$

R-YIELD

$$\frac{\Sigma = \sigma \mid \Xi \qquad \mathsf{Yield} \in \Xi}{\begin{aligned}\uparrow(\{n\} : \{[\Sigma]A\}) \rightsquigarrow_{\mathrm{u}} \uparrow(\;& \mathbf{let}\ f_n : \{\mathsf{Unit} \to [\Sigma]A\} = \{\_ \mapsto n\}\ \mathbf{in} \\ & \mathbf{let}\ y : \{[\Sigma]\mathsf{Unit}\} = \{\mathsf{yield!}\}\ \mathbf{in}\ \{f_n\ (\downarrow(y!))\} : \{[\Sigma]A\})\end{aligned}}$$

R-YIELD-EF

$$\frac{\mathcal{F}[\mathcal{E}]\ \text{allows Yield}}{\mathcal{F}[\mathcal{E}[m]] \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[\mathsf{yield!}; m]]}$$

Figure 2.8: Inserting Yields

TODO: In R-Yield-EF we have to let terms inside eval CTXs be also uses. Does this not mean that we can then put any term in either one? Is that a problem?

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$ $\boxed{n \rightsquigarrow_{\mathrm{c}} n'}$ $\boxed{m \longrightarrow_{\mathrm{u}} m'}$ $\boxed{n \longrightarrow_{\mathrm{c}} n'}$

R-HANDLE

$$\frac{k = \min_i \{i \mid \exists \overline{\theta}.(r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv [\Sigma]\ \theta_j)_j\} \qquad (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv [\Sigma]\ \theta_j)_j}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to} [\Sigma]B\})\ \bar{t}; c_y \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)); c_y + 1}$$

Figure 2.9: R-Handle with counting.

$$x \oplus y = \begin{cases} c(x+y) & \text{if } x + y \leq t_y \\ \mathsf{yield} & \text{otherwise} \end{cases}$$

where t_y is the threshold at which we force a yield. Thus the updated semantics for R-HANDLE follows.

TODO: Talk about how everything else implicitly just passes by; only if the counter is in the form $c(n)$.

R-HANDLE-COUNT in Figure **??** expresses that when handling, if our counter is of the form $c(n)$ — i.e. we do not have to yield — we perform the handling step as usual and increment the counter, potentiall yielding if we need to. R-YIELD-CAN in Figure 2.13 expresses that if the counter is in the form yield and the evaluation context allows us to yield then we *must* yield, and reset the counter to 0. R-YIELD-CAN'T

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$

$$\textsc{R-Yield}\quad \frac{c_y \geq t_y}{m; c_y \rightsquigarrow_{\mathrm{u}} \textbf{let } \mathsf{yield} : [\mathsf{Yield}]\mathsf{Unit} = y \textbf{in } m; 0}$$

Figure 2.10: Inserting Yields, when over counter

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$ $\boxed{n \rightsquigarrow_{\mathrm{c}} n'}$ $\boxed{m \longrightarrow_{\mathrm{u}} m'}$ $\boxed{n \longrightarrow_{\mathrm{c}} n'}$

$$\textsc{R-Handle-Count}\quad \frac{k = \min_i \{i \mid \exists \overline{\theta}.(r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv\!\![\Sigma]\, \theta_j)_j\} \qquad (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv\!\![\Sigma]\, \theta_j)_j}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to [\Sigma] B}\})\, \bar{t};\, c(n) \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B));\, n \oplus 1}$$

Figure 2.11: R-Handle with better counting.

says that if the evaluation context does not permit yielding, but m would otherwise reduce to some term m', then we perform that reduction inside the context. This is important; we do not block the rest of the code from reducing if we need to yield. The programmer may of course never want to yield; they may want their system to remain fully synchronous. As such they can simply never write the Yield interface in their program anywhere and the program will not yield.

The rest of the operational semantics remains the same. We adopt the syntactic sugar that any non-labelled transition $m \rightsquigarrow_{\mathrm{u}} m'$ (resp. $\rightsquigarrow_{\mathrm{c}}$) is shorthand for $m; c(n) \rightsquigarrow_{\mathrm{u}} m'; c(n)$; that is, it only applies when the counter is plain and non-blocking, and it leaves the counter unmodified.

2.7 Soundness

We now state the soundness property for our extended system, as well as the subject reduction theorem needed for this proof. Our system is nothing more than the system of [?] with extra rules; as such we omit most of the details.

Theorem 1 (Subject Reduction) • *If* $\Phi; \Gamma\, [\Sigma]\!\!\vdash m \Rightarrow A$ *and* $m; c_y \rightsquigarrow_{\mathrm{u}} m'; c_y'$ *then* $\Phi; \Gamma\, [\Sigma]\!\!\vdash m' \Rightarrow A$.

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$

R-YIELD-CAN

$$\frac{\mathcal{F}[\mathcal{E}] \text{ allows Yield}}{\mathcal{F}[\mathcal{E}[m]];\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[\mathsf{yield!};m]];c(0)}$$

R-YIELD-CAN'T

$$\frac{\neg(\mathcal{F}[\mathcal{E}] \text{ allows Yield}) \qquad m;c(n) \rightsquigarrow_{\mathrm{u}} m';c'}{\mathcal{F}[\mathcal{E}[m]];\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[m]];\mathsf{yield}}$$

Figure 2.12: Inserting Yields when forced to.

$\boxed{m \rightsquigarrow_{\mathrm{u}} m'}$

R-YIELD-CAN

$$\frac{\mathcal{E} \text{ allows Yield}}{\mathcal{E}[n];\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\mathsf{yield!};n];c(0)}$$

R-YIELD-CAN'T

$$\frac{\neg(\mathcal{E} \text{ allows Yield}) \qquad n;c(k) \rightsquigarrow_{\mathrm{u}} n';c'}{\mathcal{E}[n];\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n'];\mathsf{yield}}$$

Figure 2.13: Inserting Yields when forced to; with newer version

- *If* $\Phi;\Gamma\,[\Sigma]\!\vdash n : A$ *and* $n;c_y \rightsquigarrow_{\mathrm{c}} n';c_y'$ *then* $\Phi;\Gamma\,[\Sigma]\!\vdash n' : A$.

By induction on the transitions $\rightsquigarrow_{\mathrm{u}}, \rightsquigarrow_{\mathrm{c}}$.

We first consider the two possible states for c_y. If it is in the form $c(n)$, then the reduction rules are simply the same as in [**?**], as we do not change the counter. The only exception to this is the updated R-HANDLE rule, which is essentially the same except for modifications to the counter; regardless of the counter, the resulting term m' still remains the same type.

Thus the only new cases are R-YIELD-CAN and R-YIELD-CAN'T.

Case R-YIELD-CAN By the assumption we have that $\mathcal{E}$ allows yield. This only holds if the context is of the form

$$\mathcal{E}[\,] = \uparrow(v : \{\overline{\langle\Delta\rangle A \to [\Sigma]B}\})\ (\bar{t},[\,],\overline{n'})$$

Assume that

$$\Phi;\Gamma\,[\Sigma]\!\vdash \uparrow\!(v : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{\iota}, \mathcal{E}'[n], \overline{n'}) \Rightarrow B$$

Then by inversion on T-APP we have $\Phi;\Gamma\,[\Sigma'_{|\bar{\iota}|}]\!\vdash \mathcal{E}'[n] : A_{|\bar{\iota}|}$. We now require that $\Phi;\Gamma\,[\Sigma'_{|\bar{\iota}|}]\!\vdash \mathcal{E}'[\mathsf{yield}; n] : A_{|\bar{\iota}|}$. This follows from the assumption $\mathcal{E}$ allows yield, which entails that yield $\in \Sigma'_{|\bar{\iota}|}$. Thus $\Phi;\Gamma\,[\Sigma]\!\vdash \mathcal{E}[\mathsf{yield}; n] \Rightarrow B$.

Case R-YIELD-CAN'T This case is more straightforward. By the assumption we have that the evaluation frame $\mathcal{F}$ does not permit yielding, but the term inside the frame could otherwise reduce.

Assume $\Phi;\Gamma\,[\Sigma]\!\vdash \mathcal{F}[n] : A$, and therefore $\Phi;\Gamma\,[\Sigma]\!\vdash n : A'$. By the assumption and subject reduction, $\Phi;\Gamma\,[\Sigma]\!\vdash n' : A'$. Then clearly $\Phi;\Gamma\,[\Sigma]\!\vdash \mathcal{F}[n'] : A$.

Theorem 2 (Type Soundness)

- *If* $\cdot;\cdot\,[\Sigma]\!\vdash m \Rightarrow A$ *then either m is a normal form such that m respects* Σ *or there exists a unique* $\cdot;\cdot\,[\Sigma]\!\vdash m' \Rightarrow A$ *such that* $m \longrightarrow_{\mathrm{u}} m'$.
- *If* $\cdot;\cdot\,[\Sigma]\!\vdash n : A$ *then either n is a normal form such that n respects* Σ *or there exists a unique* $\cdot;\cdot\,[\Sigma]\!\vdash n' : A$ *such that* $n \longrightarrow_{\mathrm{c}} n'$.

The proof proceeds by simultaneous induction on $\cdot;\cdot\,[\Sigma]\!\vdash m \Rightarrow A$ and $\cdot;\cdot\,[\Sigma]\!\vdash n : A$, with use of Theorem 1.

2.8 "Tree" Yielding

TODO: c_{n+1} needs to be fresh in Add-Counter

TODO: Also need to emphasise that it's not necessarily sequential counters.

$$\textsc{Add-Counter}\quad \frac{\mathcal{E}[n]; c_1, \ldots, c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n']; c_1, \ldots, c_n \qquad \mathcal{F} \text{ is handler}}{\mathcal{F}[\mathcal{E}[n]]; c_1, \ldots, c_n, c_{n+1} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[n']]; c_1, \ldots, c_n, c_{n+1}}$$

$$\textsc{Handle-In}\quad \frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \qquad \forall\, j \leq\ n\ .\ c_j \neq \text{yield} \end{array}}{\mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma] B\})\ \bar{t}]; c_1, \ldots, c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c_n \oplus 1}$$

$$\textsc{Yield-Can}\quad \frac{\mathcal{E} \text{ allows Yield} \qquad \forall\, j \leq\ (n-1)\ .\ c_j \neq \text{yield}}{\mathcal{E}[m]; c_1, \ldots, c_{n-1}, \text{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\text{yield!}; m]; c_1, \ldots, c_{n-1}, c(0)}$$

$$\textsc{Yield-Can't}\quad \frac{\begin{array}{c} \neg(\mathcal{E} \text{ allows Yield}) \\ \forall\, j \leq\ (n-1)\ .\ c_j \neq \text{yield} \qquad \mathcal{E}[m]; c_1, \ldots, c_{n-1}, c(k)\ \rightsquigarrow_{\mathrm{u}}\ \mathcal{E}[m']; c_1, \ldots, c_{n-1}, c'_n \end{array}}{\mathcal{E}[m]; c_1, \ldots, c_{n-1}, \text{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[m']; c_1, \ldots, c_{n-1}, \text{yield}}$$

Figure 2.14: New counting

$$\textsc{Add-Counter}\quad \frac{\mathcal{E}[n]; c_n, \ldots, c_1 \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n']; c_n, \ldots, c_1 \qquad \mathcal{F} \text{ is handler}}{\mathcal{F}[\mathcal{E}[n]]; c_{n+1}, c_n, \ldots, c_1 \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[n']]; c_{n+1}, c_n, \ldots, c_1}$$

$$\textsc{Handle-In}\quad \frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \qquad \forall\, j \leq\ n\ .\ c_j \neq \text{yield} \end{array}}{\mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma] B\})\ \overline{t}]; c_n, \ldots, c_1 \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\uparrow((\overline{\theta}(n_k) : B))]; c_n \oplus 1, \ldots, c_1 \oplus 1}$$

$$\textsc{Yield-Can}\quad \frac{\mathcal{E} \text{ allows Yield} \qquad \forall\, j \leq\ (n-1)\ .\ c_j \neq \text{yield}}{\mathcal{E}[m]; c_n, \ldots, c_2, \text{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\text{yield!}; m]; c_n, \ldots, c_2, c(0)}$$

$$\textsc{Yield-Can't}\quad \frac{\begin{array}{c} \neg(\mathcal{E} \text{ allows Yield}) \\ \forall\, j \leq\ (n-1)\ .\ c_j \neq \text{yield} \qquad \mathcal{E}[m]; c_n, \ldots, c_2, c(k)\ \rightsquigarrow_{\mathrm{u}}\ \mathcal{E}[m']; c_n, \ldots, c_2, c'_1 \end{array}}{\mathcal{E}[m]; c_n, \ldots, c_2, \text{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[m']; c_n, \ldots, c_2, \text{yield}}$$

Figure 2.15: New counting, backwards ordered though

$$\textsc{Add-Counter}\quad \frac{\mathcal{E}[n]; c_1, \ldots, c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n']; c_1, \ldots, c_n \qquad \mathcal{F} \text{ is handler}}{\mathcal{F}[\mathcal{E}[n]]; c_1, \ldots, c_n, c_{n+1} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[n']]; c_1, \ldots, c_n, c_{n+1}}$$

$$\textsc{Handle-In}\quad \frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \qquad \forall\, j \leq\ n\ .\ c_j \neq \text{yield} \end{array}}{\mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma] B\})\ \overline{t}]; c_1, \ldots, c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c_n \oplus 1}$$

$$\textsc{Handle-In-Yield}\quad \frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \qquad \forall\, j \leq\ n\ .\ c_j \neq \text{yield} \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma] B\})\ \overline{t}]; c_1, \ldots, c_{n-1}, \text{yield} \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\text{yield!}; \uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c_{n-1} \oplus 1 \end{array}}$$

Figure 2.16: New new counting

Add-Counter

$$\frac{\mathcal{E}[n]; c_1, \ldots, c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n']; c_1, \ldots, c_n \qquad \mathcal{F} \text{ is 1st argument to scheduler}}{\mathcal{F}[\mathcal{E}[n]]; c_1, \ldots, c_n, c_{n+1} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[n']]; c_1, \ldots, c_n, c_{n+1}}$$

Arg-Handle

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \qquad \neg(\exists i. \text{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i) \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t}]; c_1, \ldots, c(k) \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c(k) \oplus 1 \end{array}}$$

Arg-Yield-Can't

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \qquad (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \\ \neg(\exists i. \text{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i) \qquad \neg(\mathcal{E} \text{ allows Yield}) \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t}]; c_1, \ldots, \text{yield} \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, \text{yield} \end{array}}$$

Arg-Yield

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \qquad \neg(\exists i. \text{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i) \qquad \mathcal{E} \text{ allows Yield} \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t}]; c_1, \ldots, c_{n-1}, \text{yield} \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\text{yield!}; \uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c_{n-1} \oplus 1, c(0) \end{array}}$$

Scheduler-Exit

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \qquad \exists i. \text{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t}]; c_1, \ldots, c(k), c_n \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c(k) \oplus 1 \end{array}}$$

Scheduler-Yield

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \qquad \exists i. \text{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i \qquad \mathcal{E} \text{ allows Yield} \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t}]; c_1, \ldots, \text{yield}, c_n \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\text{yield!}; \uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c_{n-2} \oplus 1, c(0) \end{array}}$$

Scheduler-Yield-Can't

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \} \\ (r_{k,j} : \langle \Delta_j \rangle A_j \leftarrow t_j \mathrel{-\!\![\Sigma]}\ \theta_j)_j \qquad \exists i. \text{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i \qquad \neg(\mathcal{E} \text{ allows Yield}) \end{array}}{\begin{array}{c} \mathcal{E}[\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle \Delta \rangle A \to}\ [\Sigma]B\})\ \overline{t}]; c_1, \ldots, \text{yield}, c_n \rightsquigarrow_{\mathrm{u}} \\ \mathcal{E}[\text{yield!}; \uparrow((\overline{\theta}(n_k) : B))]; c_1 \oplus 1, \ldots, c_{n-2} \oplus 1, \text{yield} \end{array}}$$

Figure 2.17: New new new counting

$$\textsc{Add-Counter}\quad \frac{\mathcal{E}[n]; c_1,\ldots,c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n']; c_1,\ldots,c_n \qquad \mathcal{F} \text{ is 1st argument to scheduler}}{\mathcal{F}[\mathcal{E}[n]]; c_1,\ldots,c_n,c_{n+1} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[n']]; c_1,\ldots,c_n,c_{n+1}}$$

$$\textsc{Handle-Exit}\quad \frac{\begin{array}{c} k = \min_i \{i \mid \exists\overline{\theta}.(r_{i,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv\![\Sigma]\ \theta_j)_j\} \\ (r_{k,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv\![\Sigma]\ \theta_j)_j \qquad \boxed{\neg(\exists i.\mathsf{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i)} \end{array}}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ \overline{t}; c_1,\ldots,c_n \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)); c_1 \oplus 1,\ldots,c_n \oplus 1}$$

$$\textsc{Scheduler-Exit}\quad \frac{\begin{array}{c} k = \min_i \{i \mid \exists\overline{\theta}.(r_{i,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv\![\Sigma]\ \theta_j)_j\} \\ (r_{k,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv\![\Sigma]\ \theta_j)_j \qquad \boxed{\exists i.\mathsf{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i} \end{array}}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ \overline{t}; c_1,\ldots,c_{n-1},c_n \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)); c_1 \oplus 1,\ldots,c_{n-1} \oplus 1}$$

$$\textsc{Yield-Can}\quad \frac{\mathcal{E} \text{ allows Yield}}{\mathcal{E}[m]; c_1,\ldots,\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\mathsf{yield!}; m]; c_1,\ldots,c(0)}$$

$$\textsc{Yield-Can't}\quad \frac{\neg(\mathcal{E} \text{ allows Yield}) \qquad \mathcal{E}[m]; c_n,\ldots,c_2,c(k) \rightsquigarrow_{\mathrm{u}} \mathcal{E}[m']; c_n,\ldots,c_2,c'_1}{\mathcal{E}[m]; c_n,\ldots,c_2,\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[m']; c_n,\ldots,c_2,\mathsf{yield}}$$

Figure 2.18: Concise counting.

ADD-COUNTER

$$\frac{\mathcal{E}[n]; c_1,\ldots,c_n \rightsquigarrow_{\mathrm{u}} \mathcal{E}[n']; c_1,\ldots,c_n \qquad \mathcal{F} \text{ is 1st argument to scheduler}}{\mathcal{F}[\mathcal{E}[n]]; c_1,\ldots,c_n,c_{n+1} \rightsquigarrow_{\mathrm{u}} \mathcal{F}[\mathcal{E}[n']]; c_1,\ldots,c_n,c_{n+1}}$$

HANDLE-EXIT

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}.(r_{i,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \} \\ (r_{k,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \qquad \neg(\exists i.\mathsf{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i) \end{array}}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ \bar{t}; c_1,\ldots,c_n \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)); c_1 \oplus 1,\ldots,c_n \oplus 1}$$

SCHEDULER-EXIT

$$\frac{\begin{array}{c} k = \min_i \{ i \mid \exists \overline{\theta}.(r_{i,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \} \\ (r_{k,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \qquad \exists i.\mathsf{Yield} \in \Xi_i \text{ for } \Delta_i = \Theta_i \mid \Xi_i \end{array}}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ \bar{t}; c_1,\ldots,c_{n-1},c_n \rightsquigarrow_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)); c_1 \oplus 1,\ldots,c_{n-1} \oplus 1}$$

YIELD-CAN

$$\frac{\mathcal{E} \text{ allows Yield}}{\mathcal{E}[m]; c_1,\ldots,\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[\mathsf{yield!}; m]; c_1,\ldots,c(0)}$$

YIELD-CAN'T

$$\frac{\neg(\mathcal{E} \text{ allows Yield}) \qquad \mathcal{E}[m]; c_n,\ldots,c_2,c(k) \rightsquigarrow_{\mathrm{u}} \mathcal{E}[m']; c_n,\ldots,c_2,c'_1}{\mathcal{E}[m]; c_n,\ldots,c_2,\mathsf{yield} \rightsquigarrow_{\mathrm{u}} \mathcal{E}[m']; c_n,\ldots,c_2,\mathsf{yield}}$$

Figure 2.19: Concise counting, updated

ARGUMENT-INCREMENT

$$\frac{n \rightsquigarrow_{\mathrm{c}} n'}{\begin{array}{c} \uparrow(\{((r_{i,j})_j \to n_i)_i; c\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{t}, n, \bar{n}) \rightsquigarrow_{\mathrm{u}} \\ \uparrow(\{((r_{i,j})_j \to n_i)_i; \mathsf{tryReset}(c \oplus 1, \Delta_{|\bar{t}|})\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{t}, \mathsf{tryYield}(n', c \oplus 1, \Delta_{|\bar{t}|}), \bar{n}) \end{array}}$$

Figure 2.20: Concise counting, updated

ARGUMENT-INCREMENT

$$\frac{n \rightsquigarrow_{\mathrm{c}} n'}{\begin{array}{c}\uparrow(\{((r_{i,j})_j \to n_i)_i;\ \boxed{c}\ \} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{t}, n, \overline{n}) \rightsquigarrow_{\mathrm{u}} \\ \uparrow(\{((r_{i,j})_j \to n_i)_i;\ \boxed{\mathsf{tryReset}(c \oplus 1, \Delta_{|\bar{t}|})}\ \} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{t},\ \boxed{\mathsf{tryYield}(n', c \oplus 1, \Delta_{|\bar{t}|})}\ , \overline{n})\end{array}}$$

Figure 2.21: Concise counting v2

ARGUMENT-INCREMENT

$$\frac{n \rightsquigarrow_{\mathrm{c}} n'}{\begin{array}{c}\uparrow(\{((r_{i,j})_j \to n_i)_i\ \boxed{@c}\ \} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{t}, n, \overline{n}) \rightsquigarrow_{\mathrm{u}} \\ \uparrow(\{((r_{i,j})_j \to n_i)_i\ \boxed{@(\mathsf{tryReset}(c \oplus 1, \Delta_{|\bar{t}|}))}\ \} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ (\bar{t},\ \boxed{\mathsf{tryYield}(n', c \oplus 1, \Delta_{|\bar{t}|})}\ , \overline{n})\end{array}}$$

Figure 2.22: Concise counting v3

Chapter 3

Use-Only Reductions

Here we talk about how to change the operational semantics so that only uses may reduce.

$$
\begin{array}{lrl}
\text{(uses)} & m & ::= \cdots \mid \lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil \\
\text{(constructions)} & n & ::= \ldots \\
\text{(use values)} & u & ::= x \mid f\ \overline{R} \mid \uparrow\!(v : A) \\
\text{(non-use values)} & v & ::= k\ \overline{w} \mid \{e\} \\
\text{(construction values)} & w & ::= \downarrow\! u \mid v \\
\text{(normal forms)} & t & ::= w \mid \lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil \\
\text{(evaluation frames)} & \mathcal{F} & ::= [\,]\ \overline{n} \mid u\ (\overline{t}, [\,], \overline{n}) \mid \uparrow\!([\,] : A) \\
 & & \mid\ \ \downarrow\![\,] \mid k\ (\overline{w}, [\,], \overline{n}) \mid c\ \overline{R}\ (\overline{w}, [\,], \overline{n}) \\
 & & \mid\ \ \mathbf{let}\ f : P = [\,]\ \mathbf{in}\ n \mid \langle \Theta \rangle\ [\,] \\
\text{(evaluation contexts)} & \mathcal{E} & ::= [\,] \mid \mathcal{F}[\mathcal{E}]
\end{array}
$$

Figure 3.1: Runtime Syntax for Use-Only Reductions

TODO: Remove command uses from constructiosn?

TODO: Change eval ctxs (i can't remember what to change to)?

$\boxed{m \leadsto_{\mathrm{u}} m'}$ $\boxed{m \longrightarrow_{\mathrm{u}} m'}$

R-HANDLE
$$\frac{k = \min_i \{ i \mid \exists \overline{\theta}. (r_{i,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j \} \qquad (r_{k,j} : \langle\Delta_j\rangle A_j \leftarrow t_j \dashv[\Sigma]\ \theta_j)_j}{\uparrow(\{((r_{i,j})_j \to n_i)_i\} : \{\overline{\langle\Delta\rangle A \to}\ [\Sigma]B\})\ \overline{t} \leadsto_{\mathrm{u}} \uparrow((\overline{\theta}(n_k) : B)}$$

R-ASCRIBE-USE
$$\frac{}{\uparrow(\downarrow u : A) \leadsto_{\mathrm{u}} u}$$

R-LET
$$\frac{}{\uparrow(\mathbf{let}\ f : P = w\ \mathbf{in}\ n : A) \leadsto_{\mathrm{u}} \uparrow(n[\uparrow(w : P)/f] : A)}$$

R-LETREC
$$\frac{\overline{e = \overline{r} \to n}}{\uparrow(\mathbf{letrec}\ \overline{f : P = e}\ \mathbf{in}\ n' : A) \leadsto_{\mathrm{u}} \uparrow(n'[\overline{\uparrow(\{\overline{\overline{r} \to \mathbf{letrec}\ \overline{f : P = e}\ \mathbf{in}\ n}\} : P)/f}] : A)}$$

R-ADAPT
$$\frac{}{\uparrow(\langle\Theta\rangle\ w : A) \leadsto_{\mathrm{u}} \uparrow(w : A)}$$

R-FREEZE-COMM
$$\frac{}{c\ \overline{R}\ \overline{w} \leadsto_{\mathrm{u}} \lceil c\ \overline{R}\ \overline{w} \rceil}$$

R-FREEZE-FRAME-USE
$$\frac{\neg(\mathcal{F}[\mathcal{E}]\ \text{handles}\ c)}{\mathcal{F}[\lceil \mathcal{E}[c\ \overline{R}\ \overline{w}] \rceil] \leadsto_{\mathrm{u}} \lceil \mathcal{F}[\mathcal{E}[c\ \overline{R}\ \overline{w}]] \rceil}$$

R-LIFT-UU
$$\frac{m \leadsto_{\mathrm{u}} m'}{\mathcal{E}[m] \longrightarrow_{\mathrm{u}} \mathcal{E}[m']}$$

Figure 3.2: Operational Semantics

TODO: Can R-Ascribe-Cons just get removed?

TODO: Can we just move commands to be in uses?

Appendix A

Remaining Formalisms

$\boxed{\Sigma \vdash \Delta \dashv \Sigma'}$

$$\textsc{A-Adj}\ \frac{\Sigma \vdash \Theta \dashv \Sigma' \qquad \Sigma' \vdash \Xi \dashv \Sigma''}{\Sigma \vdash \Theta \mid \Xi \dashv \Sigma''}$$

$\boxed{\Sigma \vdash \Xi \dashv \Sigma'}$

$$\textsc{A-Ext-Id}\ \frac{}{\Sigma \vdash \iota \dashv \Sigma} \qquad \textsc{A-Ext-Snoc}\ \frac{\Sigma \vdash \Xi \dashv \Sigma'}{\Sigma \vdash \Xi, I\,\overline{R} \dashv \Sigma', I\,\overline{R}}$$

$\boxed{\Sigma \vdash \Theta \dashv \Sigma'}$

$$\textsc{A-Adapt-Id}\ \frac{}{\Sigma \vdash \iota \dashv \Sigma} \qquad \textsc{A-Adapt-Snoc}\ \frac{\Sigma \vdash \Theta \dashv \Sigma' \qquad \Sigma' \vdash I(S \to S') \dashv \Sigma''}{\Sigma \vdash \Theta, I(S \to S') \dashv \Sigma''}$$

$\boxed{\Sigma \vdash I(S \to S') \dashv \Sigma'}$

$$\textsc{A-Adapt-Com}\ \frac{\Sigma \vdash S : I \dashv \Sigma'; \Omega \qquad \Omega \vdash S' : I \dashv \Xi \qquad \Sigma' \vdash \Xi \dashv \Sigma''}{\Sigma \vdash I(S \to S') \dashv \Sigma''}$$

$\boxed{\Sigma \vdash S : I \dashv \Sigma'; \Omega}$

$$\textsc{I-Pat-Id}\ \frac{}{\Sigma \vdash s : I \dashv \Sigma; s : \Sigma} \qquad \textsc{I-Pat-Bind}\ \frac{\Sigma \vdash S : I \dashv \Sigma'; \Omega}{\Sigma, I\,\overline{R} \vdash S\,a : I \dashv \Sigma'; \Omega, a : I\,\overline{R}} \qquad \textsc{I-Pat-Skip}\ \frac{\Sigma \vdash S\,a : I \dashv \Sigma'; \Omega \qquad I \neq I'}{\Sigma, I'\,\overline{R} \vdash S\,a : I \dashv \Sigma', I'\,\overline{R}; \Omega}$$

$\boxed{\Omega \vdash S : I \dashv \Xi}$

$$\textsc{I-Inst-Id}\ \frac{s \in \mathsf{dom}(\Omega)}{\Omega \vdash s : I \dashv \iota} \qquad \textsc{I-Inst-Lkp}\ \frac{a \in \mathsf{dom}(\Omega) \qquad \Omega \vdash S : I \dashv \Xi \qquad \Omega(a) = I\,\overline{R}}{\Omega \vdash S\,a : I \dashv \Xi, I\,\overline{R}}$$

Figure A.1: Action of an Adjustment on an Ability and Auxiliary Judgements

$$\mathcal{X} ::= A \mid C \mid T \mid G \mid Z \mid R \mid P \mid \sigma \mid \Sigma \mid \Xi \mid \Theta \mid \Delta \mid \Gamma \mid \exists\Psi.\Gamma \mid \Omega$$

$\boxed{\Phi \vdash \mathcal{X}}$

$$\textsc{WF-Val}\ \frac{}{\Phi, X \vdash X} \qquad \textsc{WF-Eff}\ \frac{}{\Phi, [E] \vdash E} \qquad \textsc{WF-Poly}\ \frac{\Phi, \overline{Z} \vdash A}{\Phi \vdash \forall \overline{Z}.A}$$

$$\textsc{WF-Data}\ \frac{(\Phi \vdash R)_i}{\Phi \vdash D\,\overline{R}} \qquad \textsc{WF-Thunk}\ \frac{\Phi \vdash C}{\Phi \vdash \{C\}} \qquad \textsc{WF-Comp}\ \frac{(\Phi \vdash T)_i \qquad \Phi \vdash G}{\Phi \vdash \overline{T \to}\, G} \qquad \textsc{WF-Arg}\ \frac{\Phi \vdash \Delta \qquad \Phi \vdash A}{\Phi \vdash \langle\Delta\rangle A}$$

$$\textsc{WF-Ret}\ \frac{\Phi \vdash \Sigma \qquad \Phi \vdash A}{\Phi \vdash [\Sigma]A} \qquad \textsc{WF-Ability}\ \frac{\Phi \vdash \Sigma}{\Phi \vdash [\Sigma]} \qquad \textsc{WF-Pure}\ \frac{}{\Phi \vdash \emptyset} \qquad \textsc{WF-Id}\ \frac{}{\Phi \vdash \iota} \qquad \textsc{WF-Ext}\ \frac{\Phi \vdash \Xi \qquad (\Phi \vdash R)_i}{\Phi \vdash \Xi, I\,\overline{R}}$$

$$\textsc{WF-Adapt}\ \frac{\Phi \vdash \Theta}{\Phi \vdash \Theta, I\ (S \to S')}$$

$$\textsc{WF-Empty}\ \frac{}{\Phi \vdash \cdot} \qquad \textsc{WF-Mono}\ \frac{\Phi \vdash \Gamma \qquad \Phi \vdash A}{\Phi \vdash \Gamma, x : A} \qquad \textsc{WF-Poly}\ \frac{\Phi \vdash \Gamma \qquad \Phi \vdash P}{\Phi \vdash \Gamma, f : P}$$

$$\textsc{WF-Existential}\ \frac{\Phi, \Psi \vdash \Gamma}{\Phi \vdash \exists\Psi.\Gamma} \qquad \textsc{WF-Interface}\ \frac{\Phi \vdash \Omega \qquad (\Phi \vdash R)_i}{\Phi \vdash \Omega, x : I\,\overline{R}}$$

Figure A.2: Well-Formedness Rules

$\boxed{\Phi \vdash p : A \dashv \Gamma}$

$$\textsc{P-Var}\ \frac{}{\Phi \vdash x : A \dashv x : A} \qquad \textsc{P-Data}\ \frac{k\,\overline{A} \in D\,\overline{R} \qquad (\Phi \vdash p_i : A_i \dashv \Gamma)_i}{\Phi \vdash k\,\overline{p} : D\,\overline{R} \dashv \overline{\Gamma}}$$

$\boxed{\Phi \vdash r : T \dashv\!\![\Sigma]\ \exists\Psi.\Gamma}$

$$\textsc{P-Value}\ \frac{\Sigma \vdash \Delta \dashv \Sigma' \qquad \Phi \vdash p : A \dashv \Gamma}{\Phi \vdash p : \langle\Delta\rangle A \dashv\!\![\Sigma]\ \Gamma} \qquad \textsc{P-CatchAll}\ \frac{\Sigma \vdash \Delta \dashv \Sigma'}{\Phi \vdash \langle x\rangle : \langle\Delta\rangle A \dashv\!\![\Sigma]\ x : \{[\Sigma']A\}}$$

$$\textsc{P-Command}\ \frac{\Sigma \vdash \Delta \dashv \Sigma' \qquad \Delta = \Theta \mid \Xi \qquad c : \forall\overline{Z}.\overline{A \rightarrow} B \in \Xi \qquad (\Phi, \overline{Z} \vdash p_i : A_i \dashv \Gamma_i)_i}{\Phi \vdash \langle c\ \overline{p} \rightarrow z\rangle : \langle\Delta\rangle B' \dashv\!\![\Sigma]\ \exists\overline{Z}.\overline{\Gamma}, z : \{\langle\iota \mid \iota\rangle B \rightarrow [\Sigma']B'\}}$$

Figure A.3: Pattern Matching Typing Rules