Honey Bunches of OSum

Eric Bond

January 2024

1 Untyped Syntax

value types	A	$::=$	$Bool$	
		$\mid$	X	*type variables*
		$\mid$	$CaseA$	
		$\mid$	$OSum$	
		$\mid$	$A \times A$	
		$\mid$	$A * A$	
		$\mid$	$\exists X.A$	
		$\mid$	$U\underline{B}$	
computation types	B	$::=$	$A \to \underline{B}$	
		$\mid$	$A \mathrel{-\!\!*} \underline{B}$	
		$\mid$	$\forall X.\underline{B}$	
		$\mid$	$F\ A$	
values	V	$::=$	true	
		$\mid$	false	
		$\mid$	x	*variable*
		$\mid$	σ	*type tag*
		$\mid$	$\mathtt{inj}_V\, V$	
		$\mid$	(V, V)	
		$\mid$	$(V * V)$	
		$\mid$	$\mathtt{pack}(A, V)\ \mathtt{as}\ \exists X.A$	
		$\mid$	$\mathtt{thunk}\ \ M$	
computations	M	$::=$	$\mho$	
		$\mid$	$\mathtt{force}\ \ V$	
		$\mid$	$\mathtt{ret}\ \ V$	
		$\mid$	$x \leftarrow M;\ N$	
		$\mid$	$M\ N$	
		$\mid$	$M@N$	
		$\mid$	$\lambda x : A.M$	
		$\mid$	$\alpha x : A.M$	
		$\mid$	$\mathtt{newcase}_A\ x;\ M$	
		$\mid$	$\mathtt{match}\ \ V\ \ \mathtt{with}\ \ V\{\mathtt{inj}\, x.M \mid N\}$	
		$\mid$	$\mathtt{let}\ \ (x, y) = V;\ M$	
		$\mid$	$\mathtt{let}\ \ (x * y) = V;\ M$	
		$\mid$	$\Lambda X.M$	
		$\mid$	$M[A]$	
		$\mid$	$\mathtt{if}\ \ V\ \ \mathtt{then}\ \ M\ \ \mathtt{else}\ \ N$	
value typing context	Γ	$::=$	$x : A$	
		$\mid$	ϕ	*additive unit*
		$\mid$	$\Gamma; \Gamma$	*additive combination (product)*
		$\mid$	φ	*multiplicative unit*
		$\mid$	$\Gamma \fatsemi \Gamma$	*multiplicative combination (tensor)*
stoup	Δ	$::=$	$\cdot$	
		$\mid$	$\bullet$	

2 Typed Syntax

This mostly follows from Figure 10.13 with modified rules for the context. (Σ; Δ; $\Gamma, x : A$ becomes Σ; Δ; (Γ; $x : A$), replacing ',' with '; '). The additional rules for the new connectives are below.

Normally, separation logic would partition the value context. In this case, we have a type context and a case store. Here, it seems we don't care about partitioning the type variable context. The case store is a bit trickier.. The typing rules for the original CBPV OSum in figure 10.13 demonstrate that the case store is not updated in the typing. This is also true for the cast calculus, PolyC, which introduced the type tag store for its operational semantics (see figure 10.11). The CBPV OSum typing show that the newcase term updates the value context, and it should also update the case store in the operational semantics of CPBV Osum. Thus for typing purposes, we only require the value contexts to be distinct.

$$\frac{\Sigma, \Delta, \Gamma_1 \vdash V_1 : A_1 \qquad \Sigma, \Delta, \Gamma_2 \vdash V_2 : A_2}{\Sigma, \Delta, (\Gamma_1 \fatsemi \Gamma_2) \vdash V_1 * V_2 : A_1 * A_2} *Intro$$

$$\frac{\Sigma, \Delta, \Gamma_1(x : A \fatsemi y : B) \mid \cdot \vdash N : C \qquad \Sigma, \Delta, \Gamma_2 \vdash M : A * B}{\Sigma, \Delta, \Gamma_1(\Gamma_2) \mid \cdot \vdash \mathbf{let}(x, y) = M \ \mathbf{in} \ N : C} *Elim$$

$$\frac{\Delta \vdash A \qquad \Sigma, \Delta, (\Gamma \fatsemi (x : A)) \mid \cdot \vdash M : \underline{B}}{\Sigma, \Delta, \Gamma \mid \cdot \vdash (\alpha x : A.\ M) : A \twoheadrightarrow \underline{B}} \twoheadrightarrow Intro$$

$$\frac{\Sigma, \Delta, \Gamma_1 \mid \Theta \vdash M : A \twoheadrightarrow \underline{B} \qquad \Sigma, \Delta, \Gamma_2 \vdash V : A}{\Sigma, \Delta, (\Gamma_1 \fatsemi \Gamma_2) \mid \Theta \vdash M @ V : \underline{B}} \twoheadrightarrow Elim$$

reconsider the rules with $\_ : Case\_$

3 Type and Environment Translation

Here lies the motivation for this exercise. But to make sense of this, we need to check that the cast calculus can be faithfully translated.

$$\begin{aligned}
[\![\Sigma, \Gamma \vdash \exists^v X.A]\!] &= \exists X. U(Case X \twoheadrightarrow F[\![\Sigma, \Gamma, X \vdash A]\!]) \\
[\![\Sigma, \Gamma \vdash \forall^v X.A]\!] &= U(\forall X. Case X \twoheadrightarrow F[\![\Sigma, \Gamma, X \vdash A]\!])
\end{aligned}$$

Figure 1: Type Translation Fragment

First, consider the environment translation. For the following equations, assume $[\![\Sigma \vdash \Gamma]\!] = \Delta'; \Gamma'$.

$$[\![\Sigma \vdash \cdot]\!] = \cdot; \cdot$$

$$[\![\Sigma \vdash \Gamma, x : A]\!] = \Delta'; (\Gamma'; x : [\![\Sigma; \Gamma \vdash A]\!])$$
$$[\![\Sigma \vdash \Gamma, X]\!] = \Delta', X; (\Gamma' \fatsemi c_x : CaseX)$$
$$[\![\Sigma \vdash \Gamma, X \cong A]\!] = \Delta'; (\Gamma' \fatsemi c_x : Case[\![\Sigma; \Gamma \vdash A]\!])$$

4 Semantics

Preliminary definitions and constructions

4.1 Presheaves

Redo using the presentation of Presheaves as ”Predicators”

Let C be a small category. A **presheaf** on C is a functor $F : C^{op} \to Set$. $Psh(C)$ denotes the functor category where objects are presheaves on C. This category will be used to denote value types. We will want a doubly cartesian closed structure on this category.

4.1.1 Terminal Object

The terminal object $\top$ is a functor $\top : C^{op} \to Set$. Have $\top_0(X) := \{*\}$ where $\top$ maps all objects of C^{op} to a singleton set. Have $\top_1(f : X \to Y) : * \to *$ so $\top$ maps morphisms in C^{op} to the identity function on the singleton set $*$. Say $F : C^{op} \to Set$ is an object in $Psh(C)$ and consider the natural transformation $F \Rightarrow \top$.

$$\begin{tikzcd} F(X) \arrow[r, "F(f)"] \arrow[d, "\alpha_X"] & F(Y) \arrow[d, "\alpha_Y"] \\ \perp(X) \arrow[r, "id_{\{*\}}"] & \perp(Y) \end{tikzcd} .$$

Any natural transformation from F to $\perp$ is determined by the components, of which there is no choice but the terminal map in Set. Thus, $Psh(C)$ has a terminal object.

4.1.2 Products

Given F, G presheaves on C, construct their product object in $[C^{op}, Set]$.

On objects:

$$(F \times_{Psh(C)} G)(X) := F_0(X) \times_{Set} G_0(X)$$

On morphisms: given $(f : X \to Y)$[1]

$$
\begin{aligned}
&(F \times_{Psh(C)} G)(f) : (F_0(X) \times_{Set} G_0(X)) \to (F_0(Y) \times_{Set} G_0(Y)) \\
&(F \times_{Psh(C)} G)(f)(Fx, Gy) := (F_1(f)(Fx), G_1(f)(Gy))
\end{aligned}
$$

Preserves identity (holds pairwise)[2]

$$
\begin{aligned}
(F \times_{Psh(C)} G)(id_X) &= F_1(id_X), G_1(id_X) \\
&= id_{F_0(X)}, id_{G_0(X)} \\
&= id_{(F \times_{Psh(C)} G)(X)}
\end{aligned}
$$

Preserves composition (holds pairwise)

$$
\begin{aligned}
(F \times_{Psh(C)} G)(g \circ f) &= F_1(g \circ f), G_1(g \circ f) \\
&= F_1(g) \circ F_1(f), G_1(g) \circ G_1(f)
\end{aligned}
$$

[1] $(f^{op} : Y \to X)$. Alternatively, we could write something like $\langle F_1(f) \circ \pi_1, G_1(f) \circ \pi_2 \rangle$. Being in set, we are abusing notation in the function definition by implicitly abstracting over $(F_0(X) \times_{Set} G_0(X))$ and unpacking/repacking products

[2] Again I am abusing notation by performing implicit ”computation” in Set

This describes an object $F \times_{Psh(C)} G \in Psh(C)$. The projection maps π_1, π_2 are natural transformations. Consider $\pi_1 : (F \times G) \Rightarrow F$.

$$\begin{tikzcd}
F(X) \times G(X) \arrow[r, "F(f)\times G(f)"] \arrow[d, "\pi_{1-Set}"] & F(Y) \times G(Y) \arrow[d, "\pi_{1-Set}"] \\
F(X) \arrow[r, "F(f)"] & F(Y)
\end{tikzcd}$$

The commuting diagram lives in Set and $(F \times G)(X) := (F(X), G(X))$. So the components of π_1 are the projections maps of products in Set. From what we know about Set, we know this diagram commutes. What remains is to demonstrate the universal properties for products.

$$\begin{tikzcd}
& C \arrow[dl, Rightarrow] \arrow[d, dotted] \arrow[dr, Rightarrow] & \\
F & F \times G \arrow[l, Rightarrow, "\pi_1"'] \arrow[r, Rightarrow, "\pi_2"] & G
\end{tikzcd}$$

This looks like the usual definition of product except that all the morphism are natural transformations. We can work with functions and sets if consider the underlying components and we fix an arbitrary object $X \in Ob\ C$. We know this commutes because Set has products.

$$\begin{tikzcd}
& C(X) \arrow[dl, "i"] \arrow[d, dotted, "h"] \arrow[dr, "j"] & \\
F(X) & F(X) \times G(X) \arrow[l, "\pi_{1-Set}"'] \arrow[r, "\pi_{2-Set}"] & G
\end{tikzcd}$$

4.1.3 Exponentials

To construct the exponential object for two presheaves F, G, we first need to describe the Yoneda embedding.

4.1.4 Yonedda Embedding

The Yonedda Embedding is a functor $\text{よ} : C \to Psh(C)$ which maps an object in C to its contravariant hom functor.

$$\text{よ}_0(X) := Hom(\_, X)$$

Where Any Y is mapped to the set of maps from Y to X.

$$Hom(\_, X)_1(f : B \to A) : Hom(A, X) \to Hom(B, X)$$
$$Hom(\_, X)_1(f) := \_ \circ f$$

And the action on morphisms $f^{op} : A \to B$ is precomposition.

$$
\begin{aligned}
Hom(\_, X)(id_A)(g : A \to X) &= g \circ id_A \\
&= g
\end{aligned}
$$

When written pointfree, $Hom(\_, X)(id) = id_{Hom(A,X)}$.
Finally, given $f : B \to A$, $g : C \to B$

$$
\begin{aligned}
Hom(\_, X)(g \circ^{op} f) &: Hom(A, X) \to Hom(C, X) \\
Hom(\_, X)(g \circ^{op} f)(h) &= h \circ (f \circ g) \\
&= (h \circ f) \circ g \\
&= (Hom(\_, X)(g) \circ Hom(\_, X)(f))(h)
\end{aligned}
$$

$よ_1$ maps morphisms in C to a natural transformation between hom functors.

$$よ_1(f : X \to Y) : Hom(\_, X) \Rightarrow Hom(\_, Y)$$

With components

$$
\begin{aligned}
&\eta(Z) : Hom(Z, X) \to Hom(Z, Y) \\
&\eta(Z) = f \circ \_
\end{aligned}
$$

such that for any morphism $g : W \to V$ in C,

$$
\begin{tikzcd}
Hom(V, X) \arrow[r, "Hom(g)=\_\circ g"] \arrow[d, "\eta=f\circ\_"] & Hom(W, X) \arrow[d, "\eta=f\circ\_"] \\
Hom(V, Y) \arrow[r, "Hom(g)=\_\circ g"] & Hom(W, Y)
\end{tikzcd}
$$

Pointwise, the naturality condition is $(f \circ h) \circ g = f \circ (h \circ g)$ which holds by associativity of functions in Set. What remains is the preservation of identity and composition.

$$よ_1(id_X) : Hom(\_, X) \Rightarrow Hom(\_, X)$$

Where, for a morphism $g : W \to V$,

$$
\begin{tikzcd}
Hom(V, X) \arrow[r, "Hom(g)=\_\circ g"] \arrow[d, "\eta=id_X\circ\_"'] & Hom(W, X) \arrow[d, "\eta=id_X\circ\_"] \\
Hom(V, X) \arrow[r, "Hom(g)=\_\circ g"] & Hom(W, X)
\end{tikzcd}
$$

The identity natural transformation where components are the identity function. Both commuting squares boil down to the equation $\_ \circ g = \_ \circ g$.

$$\begin{array}{ccc} Hom(V,X) & \xrightarrow{Hom(g)=\_\circ g} & Hom(W,X) \\ {\scriptstyle id}\downarrow & & \downarrow{\scriptstyle id} \\ Hom(V,X) & \xrightarrow{Hom(g)=\_\circ g} & Hom(W,X) \end{array}$$

For morphisms $f : A \to B$, $g : B \to C$,

$$よ_1(g \circ f) = よ_1(g) \circ よ_1(f)$$

$$W \xrightarrow{\quad h \quad} V$$

$$\begin{array}{ccc} Hom(V,A) & \xrightarrow{Hom(h)=\_\circ h} & Hom(W,A) \\ \downarrow{\scriptstyle (f\circ g)\circ\_} & & \downarrow{\scriptstyle (f\circ g)\circ\_} \\ Hom(V,C) & \xrightarrow{Hom(h)=\_\circ h} & Hom(W,C) \end{array}$$

$$\begin{array}{ccc} Hom(V,A) & \xrightarrow{\quad \_\circ h \quad} & Hom(W,A) \\ \downarrow{\scriptstyle f\circ\_} & & \downarrow{\scriptstyle f\circ\_} \\ Hom(V,B) & \xrightarrow{\quad \_\circ h \quad} & Hom(W,B) \\ \downarrow{\scriptstyle g\circ\_} & & \downarrow{\scriptstyle g\circ\_} \\ Hom(V,C) & \xrightarrow{\quad \_\circ h \quad} & Hom(W,C) \end{array}$$

The components of each natural transformations are equal.

4.1.5 Exponential Object

Given presheaves F, G, the exponential object G^F is also a presheaf. For an object X of C, we get a set of natural transformations.

$$G_0^F(X) := Hom_{[C^{op},Set]}(よ_0(X) \times_{psh} F, G)$$

For a morphism $f : X \to Y$ of C, we have a function between sets of natural transformations

$$G_1^F(f) : Hom_{[C^{op},Set]}(よ_0(Y) \times_{psh} F, G) \to Hom_{[C^{op},Set]}(よ_0(X) \times_{psh} F, G)$$

Say we are given $nt : Hom_{[C^{op},Set]}(よ_0(Y) \times_{psh} F, G)$ which has components

$$\alpha : (Z : ObC) \to (Hom(Z,Y), F(Z)) \to G(Z)$$

We map nt to a natural transformation in $Hom_{[C^{op},Set]}(よ_0(X)\times_{psh}F, G)$, where the components are defined to be:

$$\eta : (Z : ObC) \to (Hom(Z,X), F(Z)) \to G(Z)$$
$$\eta(Z)(g : Z \to X, fz : F(Z)) := \alpha(Z)(f \circ g, fz)$$

Given $g : W \to V$, a morphism in C, the naturality square is of the form:[3]

$$F_1(g) : F(V) \to F(W) \qquad G_1(g) : G(V) \to G(W)$$

$$\begin{tikzcd}
Hom(V,X)\times F(W) \arrow[r, "(_\circ g)\times F_1(g)"] \arrow[d, "\eta_V=\alpha_V(f\circ_\times id_{F(X)})"'] & Hom(W,X)\times F(W) \arrow[d, "\eta_W=\alpha_W(f\circ_\times id_{F(X)})"] \\
G(V) \arrow[r, "G(g)"] & G(W)
\end{tikzcd}$$

From our original natural transformation nt, we have the naturality square:

4.2 Day Convolution

The Day convolution is used to model our separating connectives $\_*\_$ and $\_ \mathrel{-\!\!*} \_$. First, we define the category of **Worlds** and a partial monoidal structure on **Worlds**. Let N be a finite set. Take the objects of **Worlds** to be subsets of N and the morphisms to be set inclusion. Define separation on the subsets of N as:

$$X * Y := \begin{cases} X \cup Y & X \cap Y = \varnothing \\ undefined & otherwise \end{cases}$$

This ensures that $X * Y$ is only defined when X and Y are disjoint subsets of N. $(N, \varnothing, *)$ is a partial commutative monoid. When both sides of the equations are defined, we get the usual commutative monoid structure on $(N, \varnothing, \cup)$.

To "lift" this to a bifunctor, $\_*\_$ is the action on objects.(but the operation is only partial... make N a pointed set?). Given $(X*Y)$, $f : X \subseteq X'$, and $g : Y \subseteq Y'$, produce $(X' * Y')$ not target may not exist.. something seems to be missing?

The Day convolution take the monoidal structure on $World$ to a monoidal structure on $\widehat{World}$.

$$(A*B)X = \int^{Y,Z} A(Y)\times B(Z)\times World^{op}[X, Y*Z]$$

The concrete interpretation for this model is

$$(A*B)X = \{[(Y,Z,a \in A(Y), b \in B(Z))] \mid Y*Z \text{ is defined and } Y*Z \subseteq X\}$$

[3] note the type of F_1 and G_1 since the source category is opposite

where $[\cdot]$ denotes an equivalence class. $(Y, Z, a \in A(Y), b \in B(Z))$ and $(Y', Z', a' \in A(Y'), b' \in B(Z'))$ are equivalent if they have the same parent in the order determined by

$$(Y, Z, a \in A(Y), b \in B(Z)) \leq (Y', Z', a' \in A(Y'), b' \in B(Z'))$$
$$\text{if}$$
$$f : Y \subseteq Y', g : Z \subseteq Z', a' = A(f)a, b' = B(g)b$$

And for the "magic wand"

$$(A \mathrel{-\!\!*} B)X = \int_Z Set^{World^{op}}[A(Z), B(X * Z)] \cong Set^{World^{op}}[A, B(X * \_)]$$