(Co)ends and (Co)structure

HoTTEST Seminar – 01 December 2022
Jacob Neumann (jacob.neumann@nottingham.ac.uk)
Where to find these slides

jacobneu.github.io/research
Impredicative Encodings
In System F, we can obtain encodings of inductive types using the impredicative $\forall$ operator, e.g. $\mathbb{N}$ can be encoded as

$$\forall \alpha.(\alpha \to \alpha) \times \alpha \to \alpha.$$ 

Awodey, Frey, and Speight (2018) studies how to do something similar in HoTT.

**Example $\mathbb{N}$**

- Unrefined encoding

$$\mathbb{N}^* \equiv \prod_{C: \text{Set}} (C \to C) \times C \to C$$

- Can define 0 and succ, prove (judgmental) $\beta$ laws
- Can’t rule out nonstandard elements: no $\eta$ law
Analogous work in HoTT

- Unrefined encoding
  \[ \mathbb{N}^* \equiv \prod_{C: \text{Set}} (C \to C) \times C \to C \]

- Refined encoding
  \[ \mathbb{N}^+ \equiv \sum_{\phi: \prod_{C: \text{Set}} (C \to C) \times C \to C} \text{isNat } \phi \]

  where
  \[ \text{isNat } \phi \equiv \prod_{f: C \to D} (f \ c_0 = d_0) \to (f \circ \gamma = \delta \circ f) \to f(\phi_C(\gamma, c_0)) = \phi_D(\delta, d_0) \]

✓ Can define 0 and succ, prove (judgmental) \( \beta \) laws
✓ Can prove (propositional) \( \eta \) law, principle of induction
**Defn** If $T : \text{Set} \to \text{Set}$ is a functor, then define the category of $T$-algebras by

$$|\text{\textit{T-Alg}|} : \equiv \sum_{C:\text{Set}} T(C) \to C \quad (C, \gamma) \to (D, \delta) : \equiv \sum_{f:C \to D} f \circ \gamma = \delta \circ T(f)$$

**Thm** The underlying set of the initial $T$-algebra is given by

$$\mu_T : \equiv \sum_{\phi: \prod_{C:\text{Set}} (T(C) \to C) \to C} \text{isNat } \phi$$

where

$$\text{isNat } \phi : \equiv \prod_{(C, \gamma), (D, \delta): \text{\textit{T-Alg}}} \prod_{f:(C, \gamma) \to (D, \delta)} f(\phi_C \gamma) = \phi_D \delta$$
Ingredients for an encoding:
- Polymorphic operation
- Naturality condition
1 A Structure Calculus
**Defn** Given a category $\mathcal{C}$ and a profunctor $F : \mathcal{C}^{\text{op}} \times \mathcal{C} \to \text{Set}$, the end of $F$ is defined as

$$\int_{c : \mathcal{C}} F(c, c) : \equiv \Sigma_{(\phi : \prod_{c : \mathcal{C}} F(c, c))} \text{isNat} \phi$$

where $\text{isNat}(\phi) : \text{Prop}$ is defined as

$$\text{isNat} \phi : \equiv \prod_{c, d : \mathcal{C}} \prod_{f : \text{Hom}_\mathcal{C}(c, d)} F(c, f) \phi_c = F(f, d) \phi_d$$
If \( F, G \) are both covariant functors (or both contravariant functors), then
\[
\int_{C : \mathcal{C}} F(C) \rightarrow G(C)
\]
is the type of natural transformations from \( F \) to \( G \).

**Lemma (Yoneda)** For any \( K : \mathcal{C}^{\text{op}} \rightarrow \text{Set} \) and \( C_0 : \mathcal{C} \),
\[
K(C_0) \simeq \int_{C : \mathcal{C}} \mathbf{y} \ C_0 \ C \rightarrow K(C)
\]
Idea: Use ends for impredicative encodings

Does it work to define

\[ \mu_T \equiv \int_{\mathbf{C} : \mathbf{Set}} (T(C) \to C) \to C \, ? \]

✓ Polymorphic operation, \( \beta \) laws

✗ Naturality condition: not right

\( \text{isNat } \phi \) : for all \( f : C \to D \) and all \( \theta : T(D) \to C \),

\[ f(\phi_C(\theta \circ T(f))) = \phi_D(f \circ \theta) \]
**Defn** For a profunctor $F : \mathcal{C}^{\text{op}} \times \mathcal{C} \to \text{Set}$, define the category $F$-Struct as

$$|F\text{-Struct}| \equiv \sum_{C : \mathcal{C}} F(C, C)$$

$$(C, \gamma) \to (D, \delta) \equiv \sum_{f : \text{Hom}_\mathcal{C}(C, D)} F(C, f) \gamma = F(f, D) \delta$$

**Note:** If $F(C^{-}, C^{+})$ is $T(C^{-}) \to C^{+}$, then $F$-Struct $\equiv T$-Alg

**Defn** Given $F, G : \mathcal{C}^{\text{op}} \times \mathcal{C} \to \text{Set}$, define

$$\int \limits_{C : \mathcal{C}} F(C, C) \, dG(C, C) \equiv \sum_{\phi : \prod_{(C, \gamma) : F\text{-Struct}} G(C, C)} \text{isNat } \phi$$
Structure Integral is the type of strong dinatural transforms

\[
\phi : \prod_{(C, \gamma) : F\text{-Struct}} G(C, C) \prod_{C : \mathcal{C}} F(C, C) \to G(C, C)
\]

isNat \( \phi \) \( \equiv \)

\[
\prod_{(C, \gamma), (D, \delta) : F\text{-Struct}} \prod_{f : (C, \gamma) \to (D, \delta)} G(C, f) (\phi_{C} \gamma) = G(f, D) (\phi_{D} \delta)
\]
For any functor $T : \text{Set} \to \text{Set}$, the set

$$\mu_T \equiv \int_{C: \text{Set}} T(C) \to C \ dC$$

equipped with

$$\text{in}_T \equiv \lambda x. (\lambda (C, \gamma). \gamma (T(\phi_C \gamma) x), \ldots) : T(\mu_T) \to \mu_T$$

is an initial $T$-algebra.

We also get the more general Yoneda-style lemma (due to Uustalu):

For any $K : \text{Set} \to \text{Set}$, and for any $T$ with initial algebra $(\mu_T, \text{in}_T)$,

$$K(\mu_T) \simeq \int_{C: \text{Set}} T(C) \to C \ dK(C)$$
With this framework, we can also obtain the curried encoding of $\mathbb{N}$: if

$$
\phi : \int_{C:\text{Set}} (C \to C) \text{d}(C \to C)
$$

then this means

$$
\phi : \prod_{C:\text{Set}} (C \to C) \to (C \to C)
$$

such that

$$
\prod_{\gamma:C\to C} \prod_{\delta:D\to D} \prod_{f:C\to D} (f \circ \gamma = \delta \circ f) \to f \circ (\phi_C \gamma) = (\phi_D \delta) \circ f
$$
We can also use it to calculate free theorems. For instance, the type

\[ \forall \alpha. (\alpha \to \alpha \to \text{bool}) \to \text{List}(\alpha) \to \text{List}(\alpha) \]

If we take the structure integral

\[ \int_{C : \text{Set}} (C \to C \to \text{bool}) \, d(\text{List}C \to \text{List}C) \]

then the naturality condition comes out as:

If \( f : (C, \preceq_C) \to (D, \preceq_D) \) is monotone (in the sense that \( (c \preceq_C c') = (f \circ c \preceq_D f \circ c') \) for all \( c, c' : C \)), then

\[ (\text{map } f) \circ (\phi_C \preceq_C) = (\phi_D \preceq_D) \circ (\text{map } f) \]
2 A Costructure Calculus
Defn For $F, G : \mathcal{C}^{\text{op}} \times \mathcal{C} \to \text{Set}$, the costructure integral is defined as

$$\int \mathcal{C}^{\mathcal{C}} F(C, C) p G(C, C) \equiv \left( \sum_{(C, \gamma) : \text{F-Struct}} G(C, C) \right) \big/ \text{Sim}_{F, G}$$

where $\text{Sim}_{F, G}$ is the least equivalence relation such that

$$\prod_{(C, \gamma), (D, \delta) : \text{F-Struct}} \prod_{f : (C, \gamma) \to (D, \delta)} \prod_{\psi : G(D, C)} \text{Sim}_{F, G} (C, \gamma, G(f, C) \psi) (D, \delta, G(D, f) \psi)$$

Jacob Neumann

(Co)ends and (Co)structure

01 December 2022 13 / 17
Costructural Yoneda Lemma

**Lemma** For any $T$ with terminal coalgebra $(\nu_T, \text{out}_T)$ and any $K : \text{Set} \to \text{Set}$,

$$K(\nu_T) \cong \int^{C : \text{Set}} C \to T(C) \cdot pK(C)$$

This allows us to give an impredicative encoding of **coinductive** types, e.g. the type $\text{Stream}(A)$ can be encoded as the costructure integral

$$\text{Stream}(A) : \equiv \int^{C : \text{Set}} C \to A \times C \cdot pC$$
Cut for time:

Bisimulations and coinduction
Consider
\[ Q(X^-, X^+) \equiv (A \times X^- \rightarrow X^+) \times (X^- \rightarrow \text{Maybe}(A \times X^+)) \times X^+ \]

Then a Q-Struct is an implementation of queues.

Then the costructure integral
\[ \int_{C: \text{Set}} Q(C, C) \]
"glues together" those implementations of queues which are bisimilar (representation independence).
3 Future Work
Avenues to explore

- Improvements to Impredicative Encodings
  - Higher Inductive Types?
  - Eliminate into arbitrary types (á la Shulman)
- Parametricity and Strong Dinaturality
- Develop the calculus more
- Semantics
- In Directed Type Theory
Thank you!