A Type-Theoretic Alternative to LISP

Alex Kavvos

Department of Computer Science, University of Oxford

TYPES 2017, 29 May 2017

The Semantics of Intensionality

Consider *functional computation*.

- *The extensional paradigm: programs as first-class citizens.*
  - higher-order arguments
  - a program can *call* higher-order argument, at a *finite number of points* (continuity!)

  Examples: purely functional programming.
- *The intensional paradigm: programs as data.*
  - programs have access to the *source code*, or *intension* of other programs
  - they can manipulate them, optimize them, call them, evaluate them for some steps and peek inside to see what it is doing, etc.

  Examples: LISP, partial evaluation.

The Semantics of Intensionality

Consider *functional computation*.

- *The extensional paradigm: programs as first-class citizens.*
  - higher-order arguments
  - a program can *call* higher-order argument, at a *finite number of points* (continuity!)

  Examples: purely functional programming.
- *The intensional paradigm: programs as data.*
  - programs have access to the *source code*, or *intension* of other programs
  - they can manipulate them, optimize them, call them, evaluate them for some steps and peek inside to see what it is doing, etc.

  Examples: LISP, partial evaluation.

The semantics of the first is well-understood.
But the second is often reduced to symbolic evaluation.

Impossibility of Quoting

Intensionality is difficult: the waters are treacherous.
See theorems by Gödel, Tarski, Rice, Kleene, ...
An example from [Barendregt, 1991]:
Suppose $\mathbf{Q} \in \Lambda$ such that

$$\mathbf{Q}\ M =_\beta \ulcorner M \urcorner$$

Then

$$\ulcorner M \urcorner =_\beta \mathbf{Q}\ M =_\beta \mathbf{Q}\ (\mathbf{I}\, M) =_\beta \ulcorner \mathbf{I}\, M \urcorner$$

which makes two distinct nf's equal.
But the λ-calculus is confluent, hence consistent.

Impossibility of Quoting

Intensionality is difficult: the waters are treacherous.
See theorems by Gödel, Tarski, Rice, Kleene, ...
An example from [Barendregt, 1991]:
Suppose $\mathbf{Q} \in \Lambda$ such that

$$\mathbf{Q}\ M =_\beta \ulcorner M \urcorner$$

Then

$$\ulcorner M \urcorner =_\beta \mathbf{Q}\ M =_\beta \mathbf{Q}\ (\mathbf{I}\, M) =_\beta \ulcorner \mathbf{I}\, M \urcorner$$

which makes two distinct nf's equal.
But the λ-calculus is confluent, hence consistent.

Moral

Quoting is impossible.

But...

Once something is already quoted, everything is OK.
E.g. there are $\mathbf{gnum}, \mathbf{app} \in \Lambda$ such that

$$\mathbf{gnum} \ulcorner M \urcorner =_\beta \ulcorner \ulcorner M \urcorner \urcorner \quad \text{and} \quad \mathbf{app} \ulcorner M \urcorner \ulcorner N \urcorner =_\beta \ulcorner M\, N \urcorner$$

Also, Kleene proved:

$$\exists \mathbf{E} \in \Lambda^0.\ \forall M \in \Lambda^0.\ \mathbf{E} \ulcorner M \urcorner =_\beta M$$

But...

Once something is already quoted, everything is OK.
E.g. there are $\mathbf{gnum}, \mathbf{app} \in \Lambda$ such that

$$\mathbf{gnum}\ \ulcorner M \urcorner =_\beta \ulcorner \ulcorner M \urcorner \urcorner \quad \text{and} \quad \mathbf{app}\ \ulcorner M \urcorner\ \ulcorner N \urcorner =_\beta \ulcorner M\, N \urcorner$$

Also, Kleene proved:

$$\exists \mathbf{E} \in \Lambda^0.\ \forall M \in \Lambda^0.\ \mathbf{E}\ \ulcorner M \urcorner =_\beta M$$

So

- *intensional operations* are admissible
- can go from *intension* ($\ulcorner M \urcorner$) to *extension* (M)

But:

Moral

Extension must not flow into Intension.

To understand intensionality, use types!

To understand intensionality, use types!

Strangely, intensionality follows a typing discipline.

To understand intensionality, use types!

Strangely, intensionality follows a typing discipline.
If $M : A$, then let $\ulcorner M \urcorner : \Box A$.

To understand intensionality, use types!

Strangely, intensionality follows a typing discipline.
If $M : A$, then let $\ulcorner M \urcorner : \Box A$.

- For **gnum** $\ulcorner M \urcorner =_\beta \ulcorner\ulcorner M \urcorner\urcorner$, we need

$$\textbf{gnum} : \Box A \to \Box\Box A$$

- For **app** $\ulcorner M \urcorner \ulcorner N \urcorner =_\beta \ulcorner M\, N \urcorner$, we need

$$\textbf{app} : \Box(A \to B) \to \Box A \to \Box B$$

- For **E** $\ulcorner M \urcorner =_\beta M$, we need

$$\textbf{E} : \Box A \to A$$

This was first noticed by Neil Jones.
Rowan Davies and Frank Pfenning: S4 + homogeneous metaprogramming.

Intensional Recursion

A strange phenomenon that is not well-understood.

Theorem (First Recursion Theorem)

$\forall f \in \Lambda.\ \exists u \in \Lambda.\ u = f\ u$

Theorem (Second Recursion Theorem)

$\forall f \in \Lambda.\ \exists u \in \Lambda.\ u = f\ \ulcorner u \urcorner$

In the second, u has access to its own code.
The second implies the first (use the interpreter).

Question

What is the programming power of this strong type of recursion?

LISP is untyped, unstructured, unhelpful for understanding this.

And the type of intensional recursion is...?

Let $u : A$.

And the type of intensional recursion is...?

Let $u : A$. Then certainly $\ulcorner u \urcorner : \Box A$.

And the type of intensional recursion is...?

Let $u : A$. Then certainly $\ulcorner u \urcorner : \Box A$. Then

$$u = f \ulcorner u \urcorner \quad \Longrightarrow \quad f : \Box A \to A$$

And the type of intensional recursion is...?

Let $u : A$. Then certainly $\ulcorner u \urcorner : \Box A$. Then

$$u = f \ulcorner u \urcorner \quad \Longrightarrow \quad f : \Box A \to A$$

Hence the logical meaning of the SRT:

Logical interpretation of the Second Recursion Theorem

From $f : \Box A \to A$, we obtain $u : A$ such that

$$u = f \ulcorner u \urcorner$$

And the type of intensional recursion is...?

Let $u : A$. Then certainly $\ulcorner u \urcorner : \Box A$. Then

$$u = f \ulcorner u \urcorner \quad \Longrightarrow \quad f : \Box A \to A$$

Hence the logical meaning of the SRT:

Logical interpretation of the Second Recursion Theorem

From $f : \Box A \to A$, we obtain $u : A$ such that

$$u = f \ulcorner u \urcorner$$

It's Löb's rule from provability logic!

$$\frac{\Box A \to A}{A}$$

Davies-Pfenning S4 and Intensional PCF

- Judgments: $\Delta\,;\Gamma \vdash M : A$ where $\begin{cases} \Delta = \text{modal/code variables} \\ \Gamma = \text{intuitionistic/value variables} \end{cases}$

Davies-Pfenning S4 and Intensional PCF

- Judgments: $\Delta\,;\Gamma \vdash M : A$ where $\begin{cases} \Delta = \text{modal/code variables} \\ \Gamma = \text{intuitionistic/value variables} \end{cases}$
- If one forgets about Δ, the system is merely simply-typed λ-calculus.

Davies-Pfenning S4 and Intensional PCF

- Judgments: $\Delta\,;\Gamma \vdash M : A$ where $\begin{cases} \Delta = \text{modal/code variables} \\ \Gamma = \text{intuitionistic/value variables} \end{cases}$
- If one forgets about Δ, the system is merely simply-typed λ-calculus.
- Using modal variables (internalises $\Box A \to A$):

$$\frac{}{\Delta, u{:}A, \Delta'\,;\Gamma \vdash u : A}$$

Davies-Pfenning S4 and Intensional PCF

- Judgments: $\Delta\,;\Gamma \vdash M : A$ where $\begin{cases}\Delta = \text{modal/code variables}\\ \Gamma = \text{intuitionistic/value variables}\end{cases}$
- If one forgets about Δ, the system is merely simply-typed λ-calculus.
- Using modal variables (internalises $\Box A \to A$):

$$\frac{}{\Delta, u{:}A, \Delta'\,;\Gamma \vdash u : A}$$

- $\Box$ is introduced if and only if *all variables are modal*:

$$\frac{\Delta\,;\cdot \vdash M : A}{\Delta\,;\Gamma \vdash \mathsf{box}\ M : \Box A}$$

Davies-Pfenning S4 and Intensional PCF

- Judgments: $\Delta\,;\Gamma \vdash M : A$ where $\begin{cases}\Delta = \text{modal/code variables}\\ \Gamma = \text{intuitionistic/value variables}\end{cases}$
- If one forgets about Δ, the system is merely simply-typed λ-calculus.
- Using modal variables (internalises $\Box A \to A$):

$$\frac{}{\Delta, u{:}A, \Delta'\,;\Gamma \vdash u : A}$$

- $\Box$ is introduced if and only if *all variables are modal*:

$$\frac{\Delta\,;\cdot \vdash M : A}{\Delta\,;\Gamma \vdash \mathsf{box}\ M : \Box A}$$

- But where are the intensional operations?
  What about intensional recursion?

Intensional functions

Let $f : \mathcal{T}(A) \to \mathcal{T}(B)$ be *any function* from closed terms of type A to closed terms of type B. Include it as a constant $\tilde{f} : \Box A \to \Box B$ such that

$$\Delta\,;\Gamma \vdash \tilde{f}(\text{box}\ M) = \text{box}\ f(M) : \Box B$$

Intensional functions

Let $f : \mathcal{T}(A) \to \mathcal{T}(B)$ be *any function* from closed terms of type A to closed terms of type B. Include it as a constant $\tilde{f} : \Box A \to \Box B$ such that

$$\Delta\,;\Gamma \vdash \tilde{f}(\text{box } M) = \text{box } f(M) : \Box B$$

And for intensional recursion, include Löb's rule:

$$\frac{\Delta\,; z : \Box A \vdash M : A}{\Delta\,;\Gamma \vdash \text{fix } z \text{ in box } M : \Box A}$$

with

$$\text{fix } z \text{ in box } M \longrightarrow \text{box } M[\text{fix } z \text{ in box } M/z]$$

Intensional functions

Let $f : \mathcal{T}(A) \to \mathcal{T}(B)$ be *any function* from closed terms of type A to closed terms of type B. Include it as a constant $\tilde{f} : \Box A \to \Box B$ such that

$$\Delta\,;\Gamma \vdash \tilde{f}(\text{box}\ M) = \text{box}\ f(M) : \Box B$$

And for intensional recursion, include Löb's rule:

$$\frac{\Delta\,; z : \Box A \vdash M : A}{\Delta\,;\Gamma \vdash \text{fix}\ z\ \text{in box}\ M : \Box A}$$

with

$$\text{fix}\ z\ \text{in box}\ M \longrightarrow \text{box}\ M[\text{fix}\ z\ \text{in box}\ M/z]$$

As long as the congruence rule $\dfrac{\Delta\,;\cdot \vdash M = N : A}{\Delta\,;\Gamma \vdash \text{box}\ M = \text{box}\ N : \Box A}$ is excluded,

Theorem (K, IMLA 2017)

The above system (Intensional PCF) is confluent.

Intensional functions

Let $f : \mathcal{T}(A) \to \mathcal{T}(B)$ be *any function* from closed terms of type A to closed terms of type B. Include it as a constant $\tilde{f} : \Box A \to \Box B$ such that

$$\Delta\,;\Gamma \vdash \tilde{f}(\text{box}\ M) = \text{box}\ f(M) : \Box B$$

And for intensional recursion, include Löb's rule:

$$\frac{\Delta\,; z : \Box A \vdash M : A}{\Delta\,;\Gamma \vdash \text{fix}\ z\ \text{in box}\ M : \Box A}$$

with

$$\text{fix}\ z\ \text{in box}\ M \longrightarrow \text{box}\ M[\text{fix}\ z\ \text{in box}\ M/z]$$

As long as the congruence rule $\dfrac{\Delta\,;\cdot \vdash M = N : A}{\Delta\,;\Gamma \vdash \text{box}\ M = \text{box}\ N : \Box A}$ is excluded,

Theorem (K, IMLA 2017)

The above system (Intensional PCF) is confluent.

Categorical semantics?

We no longer have $\vdash M = N : A$ implying $\vdash \mathrm{box}\, M = \mathrm{box}\, N : \Box A$: categorically, we need to stop $f = g \Rightarrow \Box f = \Box g$.
SOLUTION: $\Box$ is an *exposure*.

1. Replace categories with *P-categories*: axioms hold up to a PER $\sim_{A,B}$ on each hom-set $\mathcal{C}(A, B)$.
2. Replace cartesian comonads by *exposures*, $Q : \mathcal{C} \nrightarrow \mathcal{C}$. Maps on objects and morphisms that *do not respect the PERs, but reflect them instead*. Functorial otherwise!

Theorem (K, FoSSaCS 2017)

There are natural examples of exposures, both from classical logic—corresponding to Gödel numberings—and from realizability theory—where they expose the 'implementation.'

Ideas for possible applications

- Foundation for *typed intensional programming*.
  - Essentially a typed LISP with full recursion.
  - Non-functional operations are now possible, in a controlled and orderly way.
  - Of course, we have included 'too many intensional functions.' The question of which of those are computable, and which are the right primitives to express them, are both still open.
  - Resulting language can be used to write, amongst other strange programs, a **computer virus** (!)
- Relationship with metaprogramming still unclear, but certainly very interesting.
- Possible application: recursion in type theory.
- Possible application: higher-order computability.