On the Simply-Typed Functional Machine Calculus

Categorical Semantics and Strong Normalisation

Chris Barrett
University of Bath

Overview

- The Functional Machine Calculus by Heijltjes (2022)
- Part I: **Categorical Semantics**
  - Typed terms modulo machine equivalence form a CCC
  - Develop finer equational theory giving the free CCC
  - Assumption of uniform treatment of locations makes this a semantics of the *calculus*, and *not* of effects
- Part II: **Strong Normalisation**
  - Slight variant of Gandy's proof gives SN for typed terms
  - Operational intuition: counting machine steps
- Associated **publications** at MFPS 2022 and CSL 2023 [1]

Also published 'A Subatomic Proof System for Decision Trees' in ToCL 2022

Review of the Functional Machine Calculus

$$M, N \quad ::= \quad \star \;\mid\; c.\,M \;\mid\; [V]a.\,M \;\mid\; a\langle x\rangle.\,M \;\mid\; ?V.\,M$$

$$V, W \quad ::= \quad x \;\mid\; v \;\mid\; !M \qquad c \in \Sigma_c,\; v \in \Sigma_v$$

$$S \;::= \epsilon \;\mid\; S \cdot V \qquad S_A \;\triangleq\; \{\, S_a \mid a \in A\}$$

$$\frac{(\; S_A \;;\; S_a \quad ,\; [V]a.\,M \;)}{(\; S_A \;;\; S_a{\cdot}V \;,\quad M \;)} \qquad \frac{(\; S_A \;;\; S_a{\cdot}V \;,\; a\langle x\rangle.\,M \;)}{(\; S_A \;;\; S_a \quad ,\; \{V/x\}M \;)} \qquad \frac{(\; S_A \;,\; ?!N.\,M \;)}{(\; S_A \;,\; N\,;M \;)}$$

Successful run: $\dfrac{(\; S_A \;,\; M \;)}{(\; T_A \;,\; \star \;)}$

Sequential lambda-calculus (SLC) on *one* location
Capture-avoiding $N\,;M$ and $\{V/x\}M$

Review of the Functional Machine Calculus

$$
\begin{array}{rcl}
M, N & ::= & \star \ \mid\ c.\,M \ \mid\ [V]a.\,M \ \mid\ a\langle x\rangle.\,M \ \mid\ ?V.\,M \\
V, W & ::= & x \ \mid\ v \ \mid\ !M \qquad c \in \Sigma_c,\ v \in \Sigma_v
\end{array}
$$

$$
S \ ::= \epsilon \ \mid\ S \cdot V \qquad S_A \triangleq \{ S_a \mid a \in A \}
$$

$$
\frac{(\ S_A\ ;\ S_a\ ,\ [V]a.\,M\ )}{(\ S_A\ ;\ S_a{\cdot}V\ ,\ M\ )}
\qquad
\frac{(\ S_A\ ;\ S_a{\cdot}V\ ,\ a\langle x\rangle.\,M\ )}{(\ S_A\ ;\ S_a\ ,\ \{V/x\}M\ )}
\qquad
\frac{(\ S_A\ ,\ ?!N.\,M\ )}{(\ S_A\ ,\ N\,;M\ )}
$$

$$
\begin{array}{rclcrcl}
a\langle x\rangle.\,[x]a & =_{\mathsf{id}} & \star & & & & \\
[V]a.\,a\langle x\rangle.\,M & \rightarrow_\beta & \{V/x\}M & & ?!M & \rightarrow_\tau & M \\
[V]a.\,b\langle x\rangle.\,M & \rightarrow_\pi & b\langle x\rangle.\,[V]a.\,M & & !?V & \rightarrow_\phi & V
\end{array}
$$

For $\rightarrow_\pi$, $x \notin \mathsf{fv}(V), a \neq b$

Simply Typed Functional Machine Calculus

$$\tau \;::=\; \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \mid \alpha \in \Sigma_0 \qquad \overrightarrow{\tau} \;::=\; \tau_n \dots \tau_1 \qquad \overrightarrow{\tau}_A \;::=\; \{\overrightarrow{\tau}_a \mid a \in A\}$$

$$\frac{}{\Gamma \vdash_{\mathsf{c}} \star : \overleftarrow{\tau}_A \Rightarrow \overrightarrow{\tau}_A}\;\mathsf{id} \qquad \frac{}{\Gamma, x : \tau, \Delta \vdash_{\mathsf{v}} x : \tau}\;\mathsf{var}$$

$$\frac{\Gamma \vdash_{\mathsf{c}} M : \overleftarrow{\sigma}_A \overleftarrow{\tau}_A \Rightarrow \overrightarrow{\upsilon}_A \qquad c : \overleftarrow{\rho}_A \Rightarrow \overrightarrow{\sigma}_A \in \Sigma_c}{\Gamma \vdash_{\mathsf{c}} c.\,M : \overleftarrow{\rho}_A \overleftarrow{\tau}_A \Rightarrow \overrightarrow{\upsilon}_A}\;\mathsf{cconst} \qquad \frac{v : \tau \in \Sigma_v}{\Gamma \vdash_{\mathsf{v}} v : \tau}\;\mathsf{vconst}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : \rho \qquad \Gamma \vdash_{\mathsf{c}} M : a(\rho)\overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A}{\Gamma \vdash_{\mathsf{c}} [V]a.\,M : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A}\;\mathsf{app} \qquad \frac{x : \rho,\, \Gamma \vdash_{\mathsf{c}} M : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A}{\Gamma \vdash_{\mathsf{c}} a\langle x\rangle.\,M : a(\rho)\,\overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A}\;\mathsf{abs}$$

$$\frac{\Gamma \vdash_{\mathsf{c}} M : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A}{\Gamma \vdash_{\mathsf{v}} !M : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A}\;\mathsf{thunk} \qquad \frac{\Gamma \vdash_{\mathsf{v}} V : \overleftarrow{\rho}_A \Rightarrow \overrightarrow{\sigma}_A \qquad \Gamma \vdash_{\mathsf{c}} M : \overleftarrow{\sigma}_A \overleftarrow{\tau}_A \Rightarrow \overrightarrow{\upsilon}_A}{\Gamma \vdash_{\mathsf{c}} ?V.\,M : \overleftarrow{\rho}_A \overleftarrow{\tau}_A \Rightarrow \overrightarrow{\upsilon}_A}\;\mathsf{force}$$

Theorem : Termination (Heijltjes, 2022)

$$\vdash M : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \text{ then } \forall S_A : \overleftarrow{\sigma}_A\,.\;\exists T_A : \overrightarrow{\tau}_A\,.\; \frac{(\,S_A\;,\;M\,)}{(\,T_A\;,\;\star\,)}\,{}_2$$

Termination result for signature with only value constants of base type

Part I: Categorical Semantics

Definition : FMC$(\Sigma)/$??

- **Objects**: memory types $\vec{\tau}_A$ over Σ_0,
- **Morphisms**: closed terms $M : \overleftarrow{\sigma}_A \Rightarrow \vec{\tau}_A$ over Σ_v, Σ_c, mod ??
- **Composition**: $M\,;N$ with identity $\star$

A natural notion of equivalence?

Machine Equivalence

Definition : Machine Equivalence

$$M \sim M' : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \ \triangleq\ \forall S_A \sim S'_A : \overrightarrow{\sigma}_A.\ (S_A, M)\Downarrow \sim (S'_A, M')\Downarrow : \overrightarrow{\tau}_A$$

$$V \sim V' : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \ \triangleq\ ?V \sim ?V' : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A$$

$$v \sim v' : \alpha \ \triangleq\ v \equiv v' : \alpha$$

Extended to stacks pointwise, open terms by substitution. [3]

Definition for signature with only value constants of base type

Cartesian Closure

Theorem : Cartesian Closure of FMC$(\Sigma)/\sim$

$$
\begin{array}{rcr}
\vec{\tau}_A \times M & = & M : \overleftarrow{\rho}_A \overleftarrow{\tau}_A \Rightarrow \vec{\tau}_A \vec{\sigma}_A \\
M \times \vec{\tau}_A & = & \langle \overleftarrow{x}_A \rangle . (M \,;[\vec{x}_A]) : \overleftarrow{\tau}_A \overleftarrow{\rho}_A \Rightarrow \vec{\sigma}_A \vec{\tau}_A \\
\mathsf{sym} & = & \langle \overleftarrow{x}_A \rangle \langle \overleftarrow{y}_A \rangle . [\vec{x}_A] . [\vec{y}_A] : \overleftarrow{\tau}_A \overleftarrow{\sigma}_A \Rightarrow \vec{\tau}_A \vec{\sigma}_A \\
! & = & \langle \overleftarrow{x}_A \rangle : \overleftarrow{\tau}_A \Rightarrow \\
\Delta & = & \langle \overleftarrow{x}_A \rangle . [\vec{x}_A] . [\vec{x}_A] : \overleftarrow{\tau}_A \Rightarrow \vec{\tau}_A \vec{\tau}_A \\
\mathsf{eval}^a & = & a\langle f \rangle . ?f : a(\overleftarrow{\sigma}_A \Rightarrow \vec{\tau}_A) \overleftarrow{\sigma}_A \Rightarrow \vec{\tau}_A \\
\mathsf{curry}^a(M) & = & \langle \overleftarrow{x}_A \rangle . [![\vec{x}_A] . M]a : \overleftarrow{\tau}_A \Rightarrow a(\overleftarrow{\sigma}_A \Rightarrow \vec{\tau}_A)
\end{array}
$$

But too few typed contexts to distinguish some expected terms!

Equational Theory

Definition : Equational Theory $=_{\mathsf{eqn}}$

First-order: id +

$$S\,;\langle\overleftarrow{x}_A\rangle.\,[\overrightarrow{x}_A].\,[\overrightarrow{x}_A] =_\Delta S\,;S \qquad \Rightarrow\overrightarrow{\sigma}_A\overrightarrow{\sigma}_A$$

$$S\,;\langle\overleftarrow{x}_A\rangle =_! \star \qquad \Rightarrow$$

$$S\,;\langle\overleftarrow{x}_A\rangle.\,(P\,;[\overrightarrow{x}_A]) =_\iota P\,;S \qquad \overleftarrow{\pi}_A\Rightarrow\overrightarrow{\rho}_A\overrightarrow{\sigma}_A$$

Locational: π

Higher-order: β, ϕ, τ +

$$S\,;\langle\overleftarrow{x}_A\rangle.\,[![x_A].\,N]a =_\eta [!S\,;N]a \qquad \Rightarrow a(\overleftarrow{\tau}_A\Rightarrow\overrightarrow{\upsilon}_A)$$

Equational Theory

Definition : Equational Theory $=_{\text{eqn}}$

First-order: id +

$$S\,;\langle\overleftarrow{x}_A\rangle.\,[\overrightarrow{x}_A].\,[\overrightarrow{x}_A] =_{\Delta} S\,;S \qquad \Rightarrow\overrightarrow{\sigma}_A\overrightarrow{\sigma}_A$$

$$S\,;\langle\overleftarrow{x}_A\rangle =_{!} \star \qquad \Rightarrow$$

$$S\,;\langle\overleftarrow{x}_A\rangle.\,(P\,;[\overrightarrow{x}_A]) =_{\iota} P\,;S \qquad \overleftarrow{\pi}_A\Rightarrow\overrightarrow{\rho}_A\overrightarrow{\sigma}_A$$

Locational: π +

$$[V]a.\,a\langle y\rangle.\,[y]b =_{\rho} [V]b \qquad \Rightarrow b(\tau)$$

Higher-order: ϕ, τ +

$$[V]a.\,a\langle f\rangle.\,?f =_{\beta'} ?V \qquad \overleftarrow{\sigma}_A\Rightarrow\overrightarrow{\tau}_A$$

$$S\,;\langle\overleftarrow{x}_A\rangle.\,[![x_A].\,N]a =_{\eta} [!S\,;N]a \qquad \Rightarrow a(\overleftarrow{\tau}_A\Rightarrow\overrightarrow{\upsilon}_A)$$

Theorem : Beta is derivable

$$[V]a.\,a\langle x\rangle.\,M =_{\text{eqn}} \{V/x\}M$$

Free Cartesian Closed Category

Theorem : $\mathbf{FMC}(\Sigma)/{=_{\mathbf{eqn}}} \cong \mathbf{STLC}(\Sigma)$

- $\mathsf{FMC}(\Sigma) \cong \mathsf{SLC}(\Sigma)$: [4]
  - Embed $\{-\}_\lambda : \mathsf{SLC}(\Sigma) \to \mathsf{FMC}(\Sigma)$ on a location λ
  - Collapse memory to stack via order $<$ on locations,
  - Extend to functor on terms $[\![-]\!]^< : \mathsf{FMC}(\Sigma) \to \mathsf{SLC}(\Sigma)$
  - Prove inverse: one direction easy, the other up-to nat. iso:

$$\begin{tikzcd}
\vec{\sigma}_A \arrow[r, "M"] \arrow[d, "\kappa"'] & \vec{\tau}_A \arrow[d, "\kappa"] \\
\{[\![\vec{\sigma}_A]\!]\} \arrow[r, "\{[\![M]\!]\}"'] & \{[\![\vec{\tau}_A]\!]\}
\end{tikzcd}$$

- $\mathsf{SLC}(\Sigma) \cong \mathsf{STLC}(\Sigma)$: $\vdash \mapsto \Rightarrow$, so input stack maps to context

[4] Stated here for a signature on *one* location λ

Translation: SLC to STLC

Definition : $[\![-]\!] : \textbf{SLC}(\Sigma) \to \textbf{STLC}(\Sigma)$

$$[\![\Gamma \vdash_{\mathsf{v}} V : \tau]\!] : [\![\Gamma]\!] \vdash [\![\tau]\!]$$

$$[\![\Gamma \vdash_{\mathsf{c}} M : \overleftarrow{\sigma} \Rightarrow \overrightarrow{\tau}]\!] : [\![\overrightarrow{\sigma}]\!] \mid [\![\Gamma]\!] \vdash [\![\overrightarrow{\tau}]\!]$$

$$\begin{aligned}
[\![\Gamma, t, \Delta \vdash_{\mathsf{v}} t]\!](v, t, w) &= t \\
[\![\Gamma \vdash_{\mathsf{v}} v]\!](w) &= v \\
[\![\Gamma \vdash_{\mathsf{v}} !M]\!](v) &= \lambda s. [\![\Gamma \vdash_{\mathsf{c}} M]\!](s \mid v)
\end{aligned}$$

$$\begin{aligned}
[\![\Gamma \vdash_{\mathsf{c}} \star]\!](s \mid v) &= s \\
[\![\Gamma \vdash_{\mathsf{c}} c.\, M]\!](s, r \mid v) &= [\![\Gamma \vdash_{\mathsf{c}} M]\!](s, c @ r \mid v) \\
[\![\Gamma \vdash_{\mathsf{c}} \langle x \rangle.\, M]\!](s, r \mid v) &= [\![x, \Gamma \vdash_{\mathsf{c}} M]\!](s \mid r, v) \\
[\![\Gamma \vdash_{\mathsf{c}} [V].\, M]\!](s \mid v) &= [\![\Gamma \vdash_{\mathsf{c}} M]\!](s, [\![\Gamma \vdash_{\mathsf{v}} V]\!](v) \mid v) \\
[\![\Gamma \vdash_{\mathsf{c}} ?V.\, M]\!](t, r \mid v) &= [\![\Gamma \vdash_{\mathsf{c}} M]\!](t, [\![\Gamma \vdash_{\mathsf{v}} V]\!](v) @ r \mid v)
\end{aligned}$$

Part II: Strong Normalisation

Definition : Interpretation of types $(\llbracket \tau \rrbracket, \leq_{\llbracket \tau \rrbracket})$

Interpret implication as the set of *non-strict* monotone functions

$$\llbracket \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \rrbracket = \llbracket \overrightarrow{\sigma}_A \rrbracket \to \mathbb{N} \times \llbracket \overrightarrow{\tau}_A \rrbracket$$

given extensional ordering, and stack/memory types

$$\llbracket \tau_1 \ldots \tau_n \rrbracket = \llbracket \tau_1 \rrbracket \times \ldots \times \llbracket \tau_n \rrbracket \qquad \llbracket \overrightarrow{\tau}_A \rrbracket = \Pi_{a \in A} \llbracket \overrightarrow{\tau}_a \rrbracket$$

given pointwise ordering.

Definition : Collapse

Given $f \in \llbracket \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \rrbracket$,

$$\lfloor f \rfloor_{\overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A} = \pi_1(f(0_{\overrightarrow{\rho}_A})) \in \mathbb{N}$$

where $0_{\overrightarrow{\tau}_A} \in \llbracket \overrightarrow{\tau}_A \rrbracket$ is least element.

Part II: Strong Normalisation

Definition : Interpretation of types $(\llbracket\tau\rrbracket, \leq_{\llbracket\tau\rrbracket})$

Interpret implication as the set of *non-strict* monotone functions

$$\llbracket \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A \rrbracket = \llbracket \overrightarrow{\sigma}_A \rrbracket \to \mathbb{N} \times \llbracket \overrightarrow{\tau}_A \rrbracket$$

given extensional ordering, and stack/memory types

$$\llbracket \tau_1 \dots \tau_n \rrbracket = \llbracket \tau_1 \rrbracket \times \dots \times \llbracket \tau_n \rrbracket \qquad \llbracket \overrightarrow{\tau}_A \rrbracket = \Pi_{a \in A} \llbracket \overrightarrow{\tau}_a \rrbracket$$

given pointwise ordering.

Lemma : Reduction is monotonic

$$\Gamma \vdash M \to_\beta M' : \tau \quad \text{implies} \quad \llbracket M \rrbracket_v \geq_{\llbracket\tau\rrbracket} \llbracket M' \rrbracket_v$$

Theorem : Strong Normalisation

$$M \to_\beta M' : \tau \quad \text{implies} \quad \lfloor \llbracket M \rrbracket \rfloor >_{\mathbb{N}} \lfloor \llbracket M' \rrbracket \rfloor$$

Weak Interpretation of Terms

Definition : $[\![\Gamma \vdash M : \tau]\!]^W_v \in [\![\tau]\!]$

$$[\![\star]\!]^W_v(t) = (0, t)$$

$$[\![a\langle x\rangle.\, M]\!]^W_v(s, a(r)) = (1 + n, t) \text{ where } (n, t) = [\![M]\!]^W_{v\{x\leftarrow r\}}(s)$$

$$[\![[N]a.\, M]\!]^W_v(s) = (1 + n, t) \text{ where } (n, t) = [\![M]\!]^W_v(s, a([\![N]\!]^W_v))$$

$$[\![x.\, M]\!]^W_v(s, r) = (n + m, t)$$

$$\text{where } (m, t) = [\![M]\!]^W_v(s, u) \text{ and } (n, u) = v(x)(r)$$

Theorem : Interpretation counts machine steps

For closed $M : \overleftarrow{\sigma}_A \Rightarrow \overrightarrow{\tau}_A$, $S_A : \overrightarrow{\sigma}_A$ and $T_A : \overrightarrow{\tau}_A$,

$$[\![M]\!]^W([\![S]\!]^W) = (n, [\![T]\!]^W) \quad \text{implies} \quad \frac{(\ S_A\ \ ,\ M\ )}{(\ T_A\ \ ,\ \ \star\ )}\ (n \text{ steps})$$

Strong Interpretation of Terms

Definition : $[\![\Gamma \vdash M : \tau]\!]_v \in [\![\tau]\!]$

$$\ldots$$

$$[\![[N]a.\, M]\!]_v(s) = (1 + n + \lfloor [\![N]\!]_v \rfloor, t) \text{ where } (n, t) = [\![M]\!]_v(s, a([\![N]\!]_v))$$

Theorem : Strong Normalisation

$$M \rightarrow_\beta M' : \tau \quad \text{implies} \quad \lfloor [\![M]\!] \rfloor >_{\mathbb{N}} \lfloor [\![M']\!] \rfloor$$

Conclusion

- An **operational refinement** of λ-calculus that can **encode effects** but preserves many **good properties** (confluence, denotational semantics, strong normalisation)
- But the semantics presented is *not* for effects – an operational, equational and denotational account of this comes next...