Quantitative Types for the Functional Machine Calculus

Willem Heijltjes
University of Bath

FSCD 2025, Birmingham

Lambda-calculus and computational effects

$$M, N ::= x \mid \lambda x.\, M \mid M\, N$$

$$\sigma, \tau ::= o \mid \sigma \to \tau$$

Good features allow reasoning about terms (functional programs):

- Confluent reduction
- Equational theory
- Types give strong normalization
- Minimal syntax
- Intuitive semantics (functions)

How to extend this to computational effects
such as store, input/output, probabilistic choice?

Reasoning for effects

- Continuations [Landin 1965] [Reynolds 1993]
- Hoare logic [Hoare 1969]
- Idealized Algol [Reynolds 1978]
- Monads [Moggi 1991]
- Call–by–push–value [Levy 1999]
- Intersection types [Davies & Pfenning 2000]
- Algebraic effects [Plotkin & Power 2002]
- Effect handlers [Plotkin & Pretnar 2009]
- Functional Machine Calculus [H 2022]

Intersection types

Types: $\sigma, \tau ::= o \mid \mu \to \tau$

Collection types: $\mu, \nu ::= [\tau_1, \ldots, \tau_n]$

- Idempotent intersection types: collections are sets[1]
- Non-idempotent or quantitative types: collections are multisets[2]

For global higher-order store[3]: given mutable variables $A = \{a, b, c, \ldots\}$

$$\mu_A \to (\nu_A \times \tau) \qquad \text{where} \qquad \mu_A ::= \{\mu_a \mid a \in A\}$$

For example, where $M : \mu$ and $N : \tau$ and $A = \{a\}$,

$$a := M ; N : [\,]_a \to (\mu_a \times \tau)$$

[1] [Coppo & Dezani 1980] [2] [Bernadet & Lengrand 2011]
[3] [Davies & Pfenning 2000] [De'Liguoro & Treglia 2021] [Alves, Kesner & Ramos 2023]

The Functional Machine Calculus (FMC)

Idea: λ-calculus as a stack language, via the Krivine Machine[1]

Extend with effects and control flow while preserving:

- Confluence
- Typed termination
- Minimal syntax

Progress:

- Effects (store, input/output, probabilistic choice)[2]
- Sequencing (embedding CBV, Monads, CBPV)[2]
- Control flow (conditionals, exceptions, loops)[3]
- Relational computation (automata, interaction nets, Prolog)[4]
- This work: quantitative types for effects and sequencing

[1] [Krivine 2007] [2] [H 2022] [Barrett, H & McCusker 2023]
[3] [H, MFPS 2025] [4] [Barrett, Castle & H 2024]

The λ-calculus as a stack language

$$M, N ::= x \mid MN \mid \lambda x.\, M$$

$$M, N ::= x \mid [N].\, M \mid \langle x \rangle.\, M$$

Stacks:

$$S, T ::= \varepsilon \mid S\,M$$

Evaluate M with stack S:

$$S \overset{M}{\longmapsto\!\!\!\!\!\rightarrow}$$

Application $[N].\, M$ is push:

$$S \xmapsto{[N]} S\,N \overset{M}{\longmapsto\!\!\!\!\!\rightarrow}$$

Abstraction $\langle x \rangle.\, M$ is pop:

$$S\,N \xmapsto{\langle x \rangle} S \overset{\{N/x\}M}{\longmapsto\!\!\!\!\!\rightarrow}$$

Beta-reduction $[N].\, \langle x \rangle.\, M \longrightarrow \{N/x\}M$ shortens evaluation:

$$S \xmapsto{[N]} S\,N \xmapsto{\langle x \rangle} S \overset{\{N/x\}M}{\longmapsto\!\!\!\!\!\rightarrow} \quad \longrightarrow \quad S \overset{\{N/x\}M}{\longmapsto\!\!\!\!\!\rightarrow}$$

[Krivine 2007]

Sequencing

$$M, N ::= \overbrace{x \mid [N].M \mid \langle x \rangle.M}^{\lambda\text{-calculus}} \mid \overbrace{\star \mid M\,;N}^{\text{sequencing}}$$

Skip/identity $\star$

$$S \overset{\star}{\longmapsto\!\!\!\!\!\rightarrow} S$$

Sequence/composition $M\,;N$

$$R \overset{M}{\longmapsto\!\!\!\!\!\rightarrow} S \overset{N}{\longmapsto\!\!\!\!\!\rightarrow} T$$

Evaluation now returns a stack. Example: duplicate $\langle x \rangle.[x].[x].\star$

$$S\,N \overset{\langle x \rangle}{\longmapsto} S \overset{[N]}{\longmapsto} S\,N \overset{[N]}{\longmapsto} S\,N\,N \overset{\star}{\longmapsto\!\!\!\!\!\rightarrow} S\,N\,N$$

Example: swap $\langle x \rangle.\langle y \rangle.[x].[y].\star$

$$S\,N\,M \overset{\langle x \rangle}{\longmapsto} S\,N \overset{\langle y \rangle}{\longmapsto} S \overset{[M]}{\longmapsto} S\,M \overset{[N]}{\longmapsto} S\,M\,N \overset{\star}{\longmapsto\!\!\!\!\!\rightarrow} S\,M\,N$$

Cf. κ-calculus [Hasegawa 1995] [Power & Thielecke 1999], concatenative programming

Sequencing: types

$$M, N ::= \overbrace{x \mid [N].M \mid \langle x \rangle.M}^{\lambda\text{-calculus}} \mid \overbrace{\star \mid M\,;N}^{\text{sequencing}}$$

Types indicate the input stack and return stack:

$$\sigma, \tau ::= \sigma_1 \ldots \sigma_n \Rightarrow \tau_1 \ldots \tau_m$$

With vector notation:

$$\begin{array}{lrcl} \text{Types} & \sigma, \tau & ::= & \overline{\sigma} \Rightarrow \overline{\tau} \\ \text{Stack types} & \overline{\tau} & ::= & \tau_1 \ldots \tau_n \end{array}$$

Interpretation:

$$M : \overline{\sigma} \Rightarrow \overline{\tau} \quad \Longrightarrow \quad \forall S : \overline{\sigma}.\ \exists T : \overline{\tau}.\ S \overset{M}{\longmapsto\!\!\!\!\!\rightarrow} T$$

Embedded calculi

CBN λ-calculus

$$\sigma_1 \to \ldots \to \sigma_n \to o \;=\; \sigma_1 \ldots \sigma_n \Rightarrow \varepsilon$$

CBV λ-calculus

$$\begin{aligned} x_v &= x & o_v &= \varepsilon \Rightarrow \varepsilon \\ (\lambda x.M)_v &= \langle x \rangle . M_c & (\sigma \to \tau)_v &= \sigma_v \Rightarrow \tau_v \\ V_c &= [V_v] . \star & \tau_c &= \varepsilon \Rightarrow \tau_v \\ (M\,N)_c &= N_c \;;\; M_c \;;\; \langle x \rangle . x \end{aligned}$$

Computational metalanguage/CBPV

$$\begin{aligned} \text{return } M &= [M] . \star & \sigma_1 \to \cdots \to \sigma_n \to T\tau &= \sigma_1 \ldots \sigma_n \Rightarrow \tau \\ \text{let } x = M \text{ in } N &= M \;;\; \langle x \rangle . N \end{aligned}$$

[Douence & Fradet 1998] [Power & Thielecke 1999]

Locations

$$M, N ::= \overbrace{x \mid [N].M \mid \langle x \rangle.M}^{\lambda\text{-calculus}} \mid \underbrace{x \mid [N]a.M \mid a\langle x \rangle.M}_{\text{locations}} \mid \underbrace{\star \mid M;N}_{\text{sequencing}}$$

Multiple stacks (and streams) named by locations $A = \{\lambda, a, b, c, \ldots\}$

Evaluation is on a memory $S_A = S_\lambda \cdot S_a \cdot S_b \cdot S_c \ldots$ (finitely many non-empty)

$$S_A \overset{M}{\longmapsto\!\!\!\!\rightarrow} T_A$$

Push $[N]a.M$, pop $a\langle x \rangle.M$ operate on a chosen stack a, with default stack λ

$$[N].M = [N]\lambda.M \qquad \langle x \rangle.M = \lambda\langle x \rangle.M$$

Effects

$$M, N ::= \overbrace{x \mid [N]a.\,M \mid a\langle x\rangle.\,M}^{\text{locations}} \mid \overbrace{\star \mid M\,;N}^{\text{sequencing}}$$

Input/output, probabilistic choice as dedicated locations in, out, rnd

read: $\mathsf{in}\langle x\rangle.\,[x].\,\star$ print: $\langle x\rangle.\,[x]\mathsf{out}.\,\star$ random: $\mathsf{rnd}\langle x\rangle.\,[x].\,\star$

Store (mutable variables) as chosen locations a, b, c, ...

update: $a := M \;=\; M \;;\; \langle x\rangle.\,a\langle \_\rangle.\,[x]a.\,\star$

lookup: $!a \;=\; a\langle x\rangle.\,[x]a.\,[x].\,\star$ ([1])

Confluence: beta-eta equivalence gives the algebraic theory for store[2]

Typed termination: Landin's Knot[3] cannot be typed

[1] Cf. Haskell MVars [Peyton Jones, Gordon & Finne 1996] [2] [Plotkin & Power 2002] [3] [Landin 1964]

Types

$$M, N ::= \overbrace{x \mid [N]a.\,M \mid a\langle x\rangle.\,M}^{\text{locations}} \mid \overbrace{\star \mid M\,;N}^{\text{sequencing}}$$

Types: $\rho, \sigma, \tau ::= \overline{\sigma}_A \Rightarrow \overline{\tau}_A$

Stack types: $\overline{\tau} ::= \tau_1 \ldots \tau_n$

Memory types: $\overline{\tau}_A ::= \{\overline{\tau}_a \mid a \in A\}$ written $a_1(\overline{\tau}_1) \ldots a_n(\overline{\tau}_n)$

Interpretation:

$$M : \overline{\sigma}_A \Rightarrow \overline{\tau}_A \quad \Longrightarrow \quad \forall S_A : \overline{\sigma}_A.\ \exists T_A : \overline{\tau}_A.\ S_A \overset{M}{\longmapsto\!\!\!\!\rightarrow} T_A$$

Example

$$f = (a := \mathsf{rand}\,;\, !a)\,;\, f\,;\, f\,;\, +\,;\, \mathsf{print} \;:\; \mathsf{rnd}(\mathbb{Z}\,\mathbb{Z})\; a(\mathbb{Z}) \Rightarrow a(\mathbb{Z})\; \mathsf{out}(\mathbb{Z})$$

$$\overbrace{\mathsf{rnd}\langle x\rangle.\,[x].\,\star}^{\mathsf{rand}}\;;\; \overbrace{\langle y\rangle.\,a\langle\_\rangle.\,[y]a.\,\star}^{a := \ldots}\;;\; \overbrace{a\langle z\rangle.\,[z]a.\,[z].\,\star}^{!a} \;:\; \mathsf{rnd}(\mathbb{Z})\; a(\mathbb{Z}) \Rightarrow a(\mathbb{Z})\; \mathbb{Z}$$

$$\mathsf{rnd}(\mathbb{Z}) \Rightarrow \mathbb{Z} \qquad \mathbb{Z}\; a(\mathbb{Z}) \Rightarrow a(\mathbb{Z}) \qquad a(\mathbb{Z}) \Rightarrow a(\mathbb{Z})\; \mathbb{Z}$$

$$f \;;\; f \;;\; + \;;\; \mathsf{print}$$

$$\mathsf{rnd}(\mathbb{Z})\; a(\mathbb{Z}) \Rightarrow a(\mathbb{Z})\; \mathbb{Z} \qquad \mathsf{rnd}(\mathbb{Z})\; a(\mathbb{Z}) \Rightarrow a(\mathbb{Z})\; \mathbb{Z} \qquad \mathbb{Z}\,\mathbb{Z} \Rightarrow \mathbb{Z} \qquad \mathbb{Z} \Rightarrow \mathsf{out}(\mathbb{Z})$$

The machine

$$M, N ::= \overbrace{x \mid [N]a.M \mid a\langle x\rangle.M}^{\text{locations}} \mid \overbrace{\star \mid M;N}^{\text{sequencing}}$$

Stacks: $S ::= \varepsilon \mid S\,M$ Memories: $S_A ::= S_a \cdot S_b \cdot S_c \ldots$

States: (S_A, M, K) Continuation stacks: $K ::= \varepsilon \mid M\,K$

Transitions:

$$\frac{(\ S_A \cdot S_a\ ,\ [N]a.M\ ,\ K\ )}{(\ S_A \cdot (S\,N)_a\ ,\ M\ ,\ K\ )} \qquad \frac{(\ S_A\ ,\ M;N\ ,\ K\ )}{(\ S_A\ ,\ M\ ,\ N\,K\ )}$$

$$\frac{(\ S_A \cdot (S\,N)_a\ ,\ a\langle x\rangle.M\ ,\ K\ )}{(\ S_A \cdot S_a\ ,\ \{N/x\}M\ ,\ K\ )} \qquad \frac{(\ S_A\ ,\ \star\ ,\ N\,K\ )}{(\ S_A\ ,\ N\ ,\ K\ )}$$

$$S_A \overset{M}{\longmapsto\!\!\!\!\rightarrow} T_A \iff \frac{(\ S_A\ ,\ M\ ,\ \varepsilon\ )}{(\ T_A\ ,\ \star\ ,\ \varepsilon\ )}$$

Intersection types

$$M, N ::= \overbrace{x \mid [N]a.\,M \mid a\langle x\rangle.\,M}^{\text{locations}} \mid \overbrace{\star \mid M\,;N}^{\text{sequencing}}$$

Types:	σ, τ	$::=$	$\overline{\mu}_A \Rightarrow \overline{\nu}_A$	
Collection types:	ι, κ, μ, ν	$::=$	$[\tau_1, \ldots, \tau_n]$	
Stack types:	$\overline{\mu}$	$::=$	$\mu_1 \ldots \mu_n$	
Memory types:	$\overline{\mu}_A$	$::=$	$\{\overline{\mu}_a \mid a \in A\}$	written $a_1(\overline{\mu}_1) \ldots a_n(\overline{\mu}_n)$

Weak quantitative typing

$$\frac{}{x : [\tau] \vdash x : \tau}\ \text{var} \qquad \frac{}{\vdash \star : \bar{\iota}_A \Rightarrow \bar{\iota}_A}\ \text{unit}$$

$$\frac{\Gamma, x : \iota \vdash M : \overline{\kappa}_A \Rightarrow \overline{\mu}_A}{\Gamma \vdash a\langle x\rangle . M : a(\iota)\,\overline{\kappa}_A \Rightarrow \overline{\mu}_A}\ \text{abs} \qquad \frac{\Gamma \vdash N : \bar{\iota}_A \Rightarrow \overline{\kappa}_A \qquad \Delta \vdash M : \overline{\kappa}_A \Rightarrow \overline{\mu}_A}{\Gamma + \Delta \vdash N ; M : \bar{\iota}_A \Rightarrow \overline{\mu}_A}\ \text{seq}$$

$$\frac{\Gamma \vdash N : \iota \qquad \Delta \vdash M : a(\iota)\,\overline{\kappa}_A \Rightarrow \overline{\mu}_A}{\Gamma + \Delta \vdash [N]a . M : \overline{\kappa}_A \Rightarrow \overline{\mu}_A}\ \text{app} \qquad \frac{(\Gamma_i \vdash M : \tau_i)_{1 \le i \le n}}{\Gamma_1 + \ldots + \Gamma_n \vdash M : [\tau_1, \ldots, \tau_n]}\ \text{col}$$

$a(\iota)\,\overline{\kappa}_A$: the memory type $\overline{\kappa}_A$ with ι added to the vector a

Theorem The following are equivalent:

- Evaluation: $S_A \overset{M}{\longmapsto} T_A$ for some S_A, T_A
- Weak typeability: $\vdash M : \tau$
- Spine normalization: $M \twoheadrightarrow_{\mathsf{w}} N$

These are moreover quantified: the size of the typing derivation for $\vdash M : \tau$ exactly measures evaluation steps and bounds normalization steps.

Applies to embedded calculi (CBV λ-calculus, CBPV)
and effects (store, input/output, probabilistic choice)

Agrees with previous intersection types for store[1]

$$\mu_A \to (\nu_A \times \tau) \sim \mu_A \Rightarrow \nu_A\ \tau$$

[1] [De'Liguoro & Treglia 2021] [Alves, Kesner & Ramos 2023]

Reduction

$$M, N ::= \overbrace{x \mid [N]a.\,M \mid a\langle x\rangle.\,M}^{\text{locations}} \mid \overbrace{\star \mid M\,;N}^{\text{sequencing}}$$

Beta	$[N]a.\,a\langle x\rangle.\,M$	$\rightarrowtriangle$ $\{N/x\}M$	
Passage	$[N]b.\,a\langle x\rangle.\,M$	$\rightarrowtriangle$ $a\langle x\rangle.\,[N]b.\,M$	$(a \neq b, x \notin \mathsf{fv}(N))$
Next	$\star\,;M$	$\rightarrowtriangle$ M	
Associate	$(M\,;N)\,;P$	$\rightarrowtriangle$ $M\,;(N\,;P)$	
Prefix (push)	$([N]a.\,M)\,;P$	$\rightarrowtriangle$ $[N]a.\,(M\,;P)$	
Prefix (pop)	$(a\langle x\rangle.\,M)\,;P$	$\rightarrowtriangle$ $a\langle x\rangle.\,(M\,;P)$	$(x \notin \mathsf{fv}(P))$

Reduction ($\rightarrowtriangle$): in any context

Spine reduction ($\rightarrowtriangle_{\mathsf{w}}$): in any context except argument position (N in $[N]a.\,M$)

Strong quantitative typing

$$\frac{\Gamma \vdash_s N : \iota \qquad \Delta \vdash_s M : a(\iota)\,\overline{\kappa}_A \Rightarrow \overline{\mu}_A \qquad \Lambda \vdash_s N : \tau}{\Gamma + \Delta + \Lambda \vdash_s [N]a.\,M : \overline{\kappa}_A \Rightarrow \overline{\mu}_A}\ \text{APP} \qquad \frac{\Gamma \vdash_s M : \tau}{\Gamma + \Delta \vdash_s M : \tau}\ \text{WEAK}$$

Theorem The following are equivalent:

- Strong typeability: $\Gamma \vdash_s M : \tau$
- Strong normalization: $M \not\twoheadrightarrow^{\infty}$
- Perpetual evaluation: $M \blacktriangleright \mathcal{M}$

Quantified: the size of the typing derivation for $\Gamma \vdash_s M : \tau$ bounds normalization and evaluation steps.

Perpetual evaluation

Strong normalization as an inductive relation, via iterated head reduction

Cf. perpetual reduction [Bergstra & Klop 1982] [Van Raamsdonk et al 1999]

Beta
$$\frac{[S_A].\ \{N/x\}M \triangleright \mathcal{M} \qquad N \triangleright \mathcal{N}}{[S_A].\ [N]a.\ a\langle x\rangle.\ M \triangleright \mathcal{M}}$$

Passage
$$\frac{[S_A].\ a\langle x\rangle.\ [N]b.\ M \triangleright \mathcal{M}}{[S_A].\ [N]b.\ a\langle x\rangle.\ M \triangleright \mathcal{M}}$$

Next
$$\frac{[S_A].\ M \triangleright \mathcal{M}}{[S_A].\ (\star\,;M) \triangleright \mathcal{M}}$$

Prefix (pop)
$$\frac{[S_A].\ a\langle x\rangle.\ (N\,;M) \triangleright \mathcal{M}}{[S_A].\ (a\langle x\rangle.\ N\,;M) \triangleright \mathcal{M}}$$

Associate
$$\frac{[S_A].\ (P\,;(N\,;M)) \triangleright \mathcal{M}}{[S_A].\ ((P\,;N)\,;M) \triangleright \mathcal{M}}$$

Prefix (push)
$$\frac{[S_A].\ [P]a.\ (N\,;M) \triangleright \mathcal{M}}{[S_A].\ ([P]a.\ N\,;M) \triangleright \mathcal{M}}$$

Normal (abstraction)
$$\frac{M \triangleright \mathcal{M}}{a\langle x\rangle.\ M \triangleright a\langle x\rangle.\ \mathcal{M}}$$

Normal (unit)
$$\frac{(N_i \triangleright \mathcal{N}_i)_{i\leq n}}{[N_1]a_1 \ldots [N_n]a_n.\ \star \triangleright [\mathcal{N}_1]a_1 \ldots [\mathcal{N}_n]a_n.\ \star}$$

Normal (variable)
$$\frac{(N_i \triangleright \mathcal{N}_i)_{i\leq n}}{[N_1]a_1 \ldots [N_n]a_n.\ x \triangleright [\mathcal{N}_1]a_1 \ldots [\mathcal{N}_n]a_n.\ x}$$

Normal (sequence)
$$\frac{(N_i \triangleright \mathcal{N}_i)_{i\leq n} \qquad M \triangleright \mathcal{M}}{[N_1]a_1 \ldots [N_n]a_n.\ (x\,;M) \triangleright [\mathcal{N}_1]a_1 \ldots [\mathcal{N}_n]a_n.\ (x\,;\mathcal{M})}$$

Conclusions

The quantitatively-typed Functional Machine Calculus:

- characterizes termination and strong normalization
- for higher-order computation with effects (store, I/O, probabilities)
- by standard intersection type techniques
- agreeing with known intersection types for effects.

Current work:

- probabilistic types
- algebraic effects
- concurrency

Thank you