Towards Refinement by Resolution Dependent Type Theory

František Farka

joint work with Kevin Hammond and Ekaterina Komendantskaya

November 9, 2017

Motivation

$$\textbf{data}\ \mathtt{maybe}_{\mathtt{A}}\ (a : \mathtt{A}) \quad : \mathtt{Bool} \to \mathtt{type}\ \textbf{where}$$
$$\quad \mathit{nothing} \quad : \mathtt{maybe}_{\mathtt{A}}\ \mathit{ff}$$
$$\quad \mathit{just} \quad : \mathtt{A} \to \mathtt{maybe}_{\mathtt{A}}\ \mathit{tt}$$

$$\mathtt{fromJust}\ :\ \mathtt{maybe}_{\mathtt{A}}\ \mathit{tt} \to \mathtt{A}$$
$$\mathtt{fromJust}\ (\mathit{just}\ x) = x$$

Motivation

$$\textbf{data}\ \texttt{maybe}_{\texttt{A}}\ (a : \texttt{A}) \quad : \texttt{Bool} \to \texttt{type}\ \textbf{where}$$
$$\quad \textit{nothing} \quad : \texttt{maybe}_{\texttt{A}}\ \textit{ff}$$
$$\quad \textit{just} \quad : \texttt{A} \to \texttt{maybe}_{\texttt{A}}\ \textit{tt}$$

$$\texttt{fromJust} : \texttt{maybe}_{\texttt{A}}\ \textit{tt} \to \texttt{A}$$
$$\texttt{fromJust}\ (\textit{just}\ x) = x$$

$$\downarrow$$

$$\begin{aligned} \texttt{t}_{\texttt{fromJust}} = \lambda(m : \texttt{maybe}_{\texttt{A}}\ \texttt{tt}).\texttt{elim}_{\texttt{maybe}_{\texttt{A}}}\ \texttt{tt}\ m \\ (\lambda(w : \texttt{tt} \equiv \texttt{ff}).\texttt{elim}_{\equiv}\ w) \\ (\lambda(w : \texttt{tt} \equiv \texttt{tt}).\lambda(x : A).x) \end{aligned}$$

Motivation

$$\begin{array}{ll} \textbf{data}\ \mathtt{maybe}_{\mathtt{A}}\ (a : \mathtt{A}) & : \mathtt{Bool} \to \mathtt{type}\ \textbf{where} \\ \quad \textit{nothing} & : \mathtt{maybe}_{\mathtt{A}}\ \textit{ff} \\ \quad \textit{just} & : \mathtt{A} \to \mathtt{maybe}_{\mathtt{A}}\ \textit{tt} \end{array}$$

$$\mathtt{fromJust}\ :\ \mathtt{maybe}_{\mathtt{A}}\ \textit{tt} \to \mathtt{A}$$
$$\mathtt{fromJust}\ (\textit{just}\ x) = x$$

$$\downarrow$$

$$\begin{aligned} \mathtt{t}_{\mathtt{fromJust}} = \lambda(m : \mathtt{maybe}_{\mathtt{A}}\ \mathtt{tt}).\mathtt{elim}_{\mathtt{maybe}_{\mathtt{A}}}\ ?_a\ m \\ (\lambda(w : \ ?_A\ ).\ ?_b\ ) \\ (\lambda(w : \ ?_B\ ).\lambda(x : A).x) \end{aligned}$$

Problem

$$\begin{aligned} \mathtt{t}_{\mathtt{fromJust}} = \lambda(m : \mathtt{maybe}_{\mathtt{A}}\ \mathtt{tt}).\mathtt{elim}_{\mathtt{maybe}_{\mathtt{A}}}\ ?_a\ m \\ (\lambda(w :\quad ?_A \quad).\quad ?_b \quad) \\ (\lambda(w :\quad ?_B \quad).\lambda(x : A).x) \end{aligned}$$

Problem

$$
\begin{aligned}
\mathtt{t}_{\mathtt{fromJust}} = \lambda(m : \mathtt{maybe}_{\mathtt{A}}\ \mathtt{tt}).\mathtt{elim}_{\mathtt{maybe}_{\mathtt{A}}}\ ?_a\ m \\
(\lambda(w :\ ?_A\ ).\ ?_b\ ) \\
(\lambda(w :\ ?_B\ ).\lambda(x : A).x)
\end{aligned}
$$

Solution:

$$
\begin{aligned}
?_A &\to \mathtt{tt} \equiv \mathtt{ff} \\
?_B &\to \mathtt{tt} \equiv \mathtt{tt} \\
?_a &\to \mathtt{tt} \\
?_b &\to \mathtt{elim}_{\equiv}\ w
\end{aligned}
$$

Problem

$$
\begin{aligned}
\mathtt{t}_{\mathtt{fromJust}} = \lambda(m : \mathtt{maybe}_{\mathtt{A}}\ \mathtt{tt}).\mathtt{elim}_{\mathtt{maybe}_{\mathtt{A}}}\ ?_a\ m \\
(\lambda(w :\ \ ?_A\ \ ).\ \ ?_b\ \ ) \\
(\lambda(w :\ \ ?_B\ \ ).\lambda(x : A).x)
\end{aligned}
$$

Solution:

$$
\begin{aligned}
&?_A \to \mathtt{tt} \equiv \mathtt{ff} \\
&?_B \to \mathtt{tt} \equiv \mathtt{tt} \\
&?_a \to \mathtt{tt} \\
&?_b \to \mathtt{elim}_{\equiv}\ w
\end{aligned}
$$

Method:

- Refinement calculus to first-order Horn-clause logic
- Proof-relevant resolution
- Interpretation of answer substitutions as solutions

Problem

$$\begin{aligned} \mathtt{t}_{\mathtt{fromJust}} = \lambda(m : \mathtt{maybe}_{\mathtt{A}}\ \mathtt{tt}).\mathtt{elim}_{\mathtt{maybe}_{\mathtt{A}}}\ ?_a\ m \\ (\lambda(w : \quad ?_A \quad).\quad ?_b \quad) \\ (\lambda(w : \quad ?_B \quad).\lambda(x : A).x) \end{aligned}$$

First-order Horn clause logic (fohc) with resolution:

judgements	$\longrightarrow$	atoms / goals
type-level metavariables	$\longrightarrow$	logic variables
inference rules	$\longrightarrow$	program clauses
term-level metavariables	$\longrightarrow$	

Problem

$$\begin{aligned}\mathtt{t}_{\mathtt{fromJust}} = \lambda(m : \mathtt{maybe}_{\mathtt{A}}\ \mathtt{tt}).\mathtt{elim}_{\mathtt{maybe}_{\mathtt{A}}}\ ?_a\ m \\ (\lambda(w : \ ?_A\ ).\ ?_b\ ) \\ (\lambda(w : \ ?_B\ ).\lambda(x : A).x)\end{aligned}$$

First-order Horn clause logic (fohc) with resolution:

judgements	$\longrightarrow$	atoms / goals
type-level metavariables	$\longrightarrow$	logic variables
inference rules	$\longrightarrow$	program clauses
term-level metavariables	$\longrightarrow$	proof-terms

Proof-Relevant Resolution

$$
\begin{array}{rrl}
\text{Terms} & Ter ::= & Var \mid \mathcal{F}(Ter, \ldots, Ter) \\
\text{Atoms} & At ::= & \mathcal{P}(Ter, \ldots, Ter)
\end{array}
\qquad
\begin{array}{rrl}
\text{HC's} & HC ::= & At \leftarrow At, \ldots, At \\
\text{Progs} & P ::= & \cdot \mid P, HC
\end{array}
$$

$$
A \leftarrow B_1, \ldots, B_n \in P \; \frac{P \vdash \qquad \sigma B_1 \qquad \ldots \qquad P \vdash \qquad \sigma B_n}{P \vdash \qquad \sigma A}
$$

Proof-Relevant Resolution

$$
\begin{array}{llll}
\text{Terms} & Ter ::= & Var \mid \mathcal{F}(Ter, \ldots, Ter) \\
\text{Atoms} & At ::= & \mathcal{P}(Ter, \ldots, Ter)
\end{array}
\qquad
\begin{array}{llll}
\text{HC's} & HC ::= & \mathcal{K} : At \leftarrow At, \ldots, At \\
\text{Progs} & P ::= & \cdot \mid P, HC
\end{array}
$$

$$
c : A \leftarrow B_1, \ldots, B_n \in P \; \frac{P \vdash e_1 : \sigma B_1 \qquad \ldots \qquad P \vdash e_n : \sigma B_n}{P \vdash c \;\; e_1 \;\; \ldots \;\; e_n : \sigma A}
$$

Example

$$\frac{}{\Gamma \vdash \mathsf{type} : \mathsf{kind}} \textsc{TypeAx}$$

Example

$$P = \ldots,$$
$$c_{typeAx} : \quad kind(type, X) \quad \leftarrow$$

$$\frac{}{\Gamma \vdash \text{type} : \text{kind}}\ \textsc{TypeAx}$$

Example

$P = \ldots,$

$c_{typeAx} : \quad kind(type, X) \quad \leftarrow$

$$\frac{}{\Gamma \vdash \text{type} : \text{kind}}\ \textsc{TypeAx}$$

Goal: `kind(Y, nil)`

$$\frac{}{P \vdash c_{typeAx} : kind(type, nil)}$$

Refinement

$\mathcal{S} \vdash P$

$\mathcal{S}; \Gamma; A \vdash (G \mid K)$
$\mathcal{S}; \Gamma; M \vdash (G \mid A)$

Refinement

$\mathcal{S} \vdash P$

$\mathcal{S}; \Gamma; A \vdash (G \mid K)$

$\mathcal{S}; \Gamma; M \vdash (G \mid A)$

LF:

$$\frac{\mathcal{S}; \Gamma \vdash: \Pi x : A.B \qquad \mathcal{S}; \Gamma \vdash N : A}{\mathcal{S}; \Gamma \vdash MN : B[M/x]} \ \Pi\text{-}\textsc{elim}$$

Refinement calculus:

$$\frac{\mathcal{S}; \Gamma; M \vdash (G_M \mid C) \qquad \mathcal{S}; \Gamma; N \vdash (G_N \mid A)}{\mathcal{S}; \Gamma; MN \vdash (G_M \wedge G_N \wedge C \equiv \Pi A.?_B \mid ?_B[M/x])} \ \textsc{ref}\text{-}\Pi\text{-}\textsc{elim}$$

Refinement

$\mathcal{S} \vdash P$

$\mathcal{S}; \Gamma; A \vdash (G \mid K)$
$\mathcal{S}; \Gamma; M \vdash (G \mid A)$

LF:

$$\frac{\mathcal{S}; \Gamma \vdash : \Pi x : A.B \qquad \mathcal{S}; \Gamma \vdash N : A}{\mathcal{S}; \Gamma \vdash MN : B[M/x]} \ \Pi\text{-ELIM}$$

Refinement calculus:

$$\frac{\mathcal{S}; \Gamma; M \vdash (G_M \mid C) \qquad \mathcal{S}; \Gamma; N \vdash (G_N \mid A)}{\mathcal{S}; \Gamma; MN \vdash (G_M \wedge G_N \wedge C \equiv \Pi A.?_B \mid ?_B[M/x])} \ \text{REF-}\Pi\text{-ELIM}$$

Lemma

Let t be a refinement problem in a well-formed signaure $\mathcal{S}$ such that a solution (ρ, R) exists. Then there is a goal G and an extended type A such that $\mathcal{S}; \cdot \vdash (G \mid A)$.

Nameless Representation

builds on *Locally nameless representation* (Urban *et al.*; 2011)

$$\lambda B.x0z \underset{close}{\overset{open}{\leftrightarrows}} \quad \lambda \quad A.\lambda \quad B.1\ 0\ z$$

Nameless Representation

builds on *Locally nameless representation* (Urban *et al.*; 2011)

$$\lambda B.x0z \underset{close}{\overset{open}{\leftrightarrows}} \qquad \lambda \quad A.\lambda \quad B.1\ 0\ z$$

$$\Gamma \vdash \lambda x : A.\lambda y : B.xy\ 0 \underset{shift}{\overset{unshift}{\leftrightarrows}} \Gamma, C \vdash \lambda x : A.\lambda y : B.xy1$$

Nameless Representation

builds on *Locally nameless representation* (Urban *et al.*; 2011)

$$\lambda B.x0z \underset{close}{\overset{open}{\leftrightarrows}} \quad \lambda \quad A.\lambda \quad B.1\ 0\ z$$

$$\Gamma \vdash \lambda x : A.\lambda y : B.xy\ 0 \underset{shift}{\overset{unshift}{\leftrightarrows}} \Gamma, C \vdash \lambda x : A.\lambda y : B.xy1$$

$$\Gamma \vdash \lambda \quad A.\lambda \quad B.1_T 0_T 0_\Gamma$$

Nameless Representation

builds on *Locally nameless representation* (Urban *et al.*; 2011)

$$\lambda B.x0z \overset{open}{\underset{close}{\leftrightarrows}} \quad \lambda \quad A.\lambda \quad B.1\ 0\ z$$

$$\Gamma \vdash \lambda x : A.\lambda y : B.xy\ 0 \overset{unshift}{\underset{shift}{\leftrightarrows}} \Gamma, C \vdash \lambda x : A.\lambda y : B.xy1$$

$$\Gamma \vdash \lambda \quad A.\lambda \quad B.1_T 0_T 0_\Gamma$$

Combined operations:

- simultaneous open/shift: $\overleftarrow{M}[0_\Gamma/0_T]$
- simultaneous close/unshift: $\overrightarrow{M}[0_T/0_\Gamma]$
- simultaneous open/substitution: $M[0_T/N]$

Nameless Representation

builds on *Locally nameless representation* (Urban *et al.*; 2011)

$$\lambda B.x0z \underset{close}{\overset{open}{\leftrightarrows}} \quad \lambda \quad A.\lambda \quad B.1\ 0\ z$$

$$\Gamma \vdash \lambda x : A.\lambda y : B.xy\ 0 \underset{shift}{\overset{unshift}{\leftrightarrows}} \Gamma, C \vdash \lambda x : A.\lambda y : B.xy1$$

$$\Gamma \vdash \lambda \quad A.\lambda \quad B.1_T 0_T 0_\Gamma$$

Combined operations:

- simultaneous open/shift: $\overleftarrow{M}[0_\Gamma/0_T]$
- simultaneous close/unshift: $\overrightarrow{M}[0_T/0_\Gamma]$
- simultaneous open/substitution: $M[0_T/N]$

Lemma (Equality of LF and namelesss LF)

$\mathcal{S}; \Gamma \vdash M : A$ *if and only if* $\ulcorner\mathcal{S}\urcorner; \ulcorner\Gamma\urcorner \vdash \ulcorner M\urcorner : \ulcorner A\urcorner$

Example

$$\frac{}{\Gamma, C \vdash 0_\Gamma : C}\ \textsc{CtxZero}$$

$$\frac{\overrightarrow{\Gamma} \vdash i_\Gamma : C}{\Gamma, D \vdash \sigma i_\Gamma : \overleftarrow{C}}\ \textsc{CtxSucc}$$

Example

$$P = \ldots,$$

$$c_{ctxZero}: \quad type(C, [C \mid \Gamma]) \leftarrow$$

$$c_{ctxSucc}: \quad type(C', [D \mid \Gamma]) \leftarrow$$

$$\Gamma \equiv \overleftarrow{\Gamma'}, type(\overrightarrow{C'}, \Gamma'), C' \equiv \overleftarrow{C}$$

$$\frac{}{\Gamma, C \vdash 0_\Gamma : C}\ \textsc{CtxZero}$$

$$\frac{\overrightarrow{\Gamma} \vdash i_\Gamma : C}{\Gamma, D \vdash \sigma i_\Gamma : \overleftarrow{C}}\ \textsc{CtxSucc}$$

Example

$$P = \ldots,$$

$$c_{ctxZero} : \quad type(C, [C \mid \Gamma]) \leftarrow$$

$$c_{ctxSucc} : \quad type(C', [D \mid \Gamma]) \leftarrow$$

$$\Gamma \equiv \overleftarrow{\Gamma'}, type(\overrightarrow{C'}, \Gamma'), C' \equiv \overleftarrow{C}$$

$$\frac{}{\Gamma, C \vdash 0_\Gamma : C}\ \textsc{CtxZero}$$

$$\frac{\overrightarrow{\Gamma} \vdash i_\Gamma : C}{\Gamma, D \vdash \sigma i_\Gamma : \overleftarrow{C}}\ \textsc{CtxSucc}$$

Example: $\Gamma = \mathtt{n} : \mathtt{A}, \mathtt{m} : \mathtt{maybe}_\mathtt{A}\ (\mathtt{tt})$ and a goal for $\mathtt{type}(?_\mathtt{b}, \mathtt{A}, \Gamma)$:

$$\frac{}{P \vdash c_{ctxZero} : type(A, [A, maybe_A(tt)])}$$

Soundness

Theorem

Let M be a term in $\mathcal{S}$. Let P_M and G_M be a program and a goal s.t. $\mathcal{S} \vdash P$ and $\mathcal{S}; \cdot; M \vdash (G_M \mid A)$. Let ρ, R be a substitution and a proof-term assignment such that $P \vdash^{\rho}_{R} G_M$. Then for any solution (ρ', R') s.t. $(\rho', R')M$ is a well-formed there is (ρ'', R'') such that

$$(\rho'', R'')((\rho, R)M) = (\rho', R')M$$

Conclusion and Future work

Conclusion

- type and term refinement in first-order type theory as first-order resolution
- nameless representation removes α-equivalence and name freshness issues
- proof-relevant resolution captures term-level metavariables

Future work

- higher-order DTT
- coinductive interpretation of the program
  - extend our work on coinductive soundess of Haskell typeclasses (Farka *et al.*; 2017)
- `https://github.com/frantisekfarka/resviaref`

Thank you

First-order Dependent Type Theory

Language we are working in

Language of FoDTT:

$$
\begin{aligned}
\text{Kinds} \quad K &::= \quad \text{type} \mid \Pi\mathcal{V} : T.K \\
\text{Types} \quad T &::= \quad \mathcal{B} \mid \Pi\mathcal{V} : T.T \mid Tt \\
\text{Terms} \quad t &::= \quad \mathcal{C} \mid \mathcal{V} \mid \lambda\mathcal{V} : T.t \mid tt
\end{aligned}
$$

Well-formedness rules:

$$
\begin{aligned}
\Sigma; \Gamma &\vdash K : \text{kind} \\
\Sigma; \Gamma &\vdash A : K \\
\Sigma; \Gamma &\vdash M : A
\end{aligned}
$$

Refinement

We extend the language of FoDTT to FoDTT*:

$$\begin{array}{rrl} \text{Types*} & T ::= & \cdots \,|\, ?\mathcal{B} \\ \text{Terms*} & t ::= & \cdots \,|\, ?\mathcal{V} \end{array}$$

and we look for assignements $?\mathcal{B} \rightarrow \texttt{Types}$ and $?\mathcal{V} \rightarrow \texttt{Terms}$.

Refinement

We extend the language of FoDTT to FoDTT*:

$$\begin{array}{rrl} \text{Types*} & T ::= & \cdots \mid ?\mathcal{B} \\ \text{Terms*} & t ::= & \cdots \mid ?\mathcal{V} \end{array}$$

and we look for assignements $?\mathcal{B} \rightarrow \texttt{Types}$ and $?\mathcal{V} \rightarrow \texttt{Terms}$.

Example:

$\lambda n :?_A.\lambda v : VecA\ (c_{succ}\ n).c_{elVec}?_{irr}(\lambda m :?_B.\lambda a :?_C.\lambda as :?_D.a)?_n?_v$

Refinement (cont'd)

By generation of goals that constrain type-level meta-variables and by binging their proof-terms to term-level meta-variables:

$$\Sigma; \Gamma; t \vdash_{TeGoal} (G \mid t' : A)$$
$$\Sigma; \Gamma; A \vdash_{TyGoal} (G \mid A' : K)$$
$$\Sigma; \Gamma; K \vdash_{KiGoal} (G \mid K' : \mathsf{kind})$$

Refinement (cont'd)

By generation of goals that constrain type-level meta-variables and by binging their proof-terms to term-level meta-variables:

$$\Sigma; \Gamma; t \vdash_{TeGoal} (G \mid t' : A)$$
$$\Sigma; \Gamma; A \vdash_{TyGoal} (G \mid A' : K)$$
$$\Sigma; \Gamma; K \vdash_{KiGoal} (G \mid K' : \text{kind})$$

Signature gives a program that solves the generated goals:

$$\Sigma \vdash_{Prog} P$$