Elaborating dependent (co)pattern matching

Jesper Cockx Andreas Abel

Chalmers & Gothenburg University

23 March 2018

Type systems & proof assistants: science or faith?

surface language

core language

Type systems & proof assistants: science or faith?

$$\boxed{\textbf{surface language}} \xRightarrow{\textit{elaboration}} \boxed{\textbf{core language}}$$

Type systems & proof assistants: science or faith?

science

$\downarrow$

surface language $\xRightarrow{\textit{elaboration}}$ **core language**

Type systems & proof assistants: science or faith?

$$\begin{array}{ccc} & \text{faith} & \text{science} \\ & \downarrow & \downarrow \\ \boxed{\begin{array}{l}\textbf{surface}\\ \textbf{language}\end{array}} & \xRightarrow{\textit{elaboration}} & \boxed{\begin{array}{l}\textbf{core}\\ \textbf{language}\end{array}} \end{array}$$

Type systems & proof assistants: science or faith?

faith → *elaboration* (surface language ⟹ core language)

science → core language

$$\boxed{\textbf{surface language}} \xRightarrow{\textit{elaboration}} \boxed{\textbf{core language}}$$

Goal: turn piece of faith into science.

Presenting. . .

A core language with inductive data types, coinductive record types, an identity type, and typed case trees.

Presenting. . .

A core language with inductive data types, coinductive record types, an identity type, and typed case trees.

An elaboration algorithm from copattern matching to a well-typed case tree.

Presenting...

A core language with inductive data types, coinductive record types, an identity type, and typed case trees.

An elaboration algorithm from copattern matching to a well-typed case tree.

A proof that elaboration preserves the first-match semantics of the clauses.

Dependent copattern matching

Surface and core languages

From clauses to a case tree

Preservation of first-match semantics

Example: maximum

$$
\begin{aligned}
&\mathsf{max} : \mathbb{N} \to \mathbb{N} \to \mathbb{N} \\
&\mathsf{max}\ \mathsf{zero}\ \qquad\ \ \ y \qquad\qquad = y \\
&\mathsf{max}\ x \qquad\quad\ \ \mathsf{zero} \qquad = x \\
&\mathsf{max}\ (\mathsf{suc}\ x)\ (\mathsf{suc}\ y) = \mathsf{suc}\ (\mathsf{max}\ x\ y)
\end{aligned}
$$

Example: maximum

$$
\begin{aligned}
&\mathsf{max} : \mathbb{N} \to \mathbb{N} \to \mathbb{N} \\
&\mathsf{max}\ \mathsf{zero}\ \ \ \ \ \ \ \ \ y \qquad\quad = y \\
&\mathsf{max}\ x \qquad\ \ \ \ \mathsf{zero} \quad\ = x \\
&\mathsf{max}\ (\mathsf{suc}\ x)\ (\mathsf{suc}\ y) = \mathsf{suc}\ (\mathsf{max}\ x\ y)
\end{aligned}
$$

First-match semantics:
We don't have $\mathsf{max}\ x\ \mathsf{zero} = x$,
but only $\mathsf{max}\ (\mathsf{suc}\ x)\ \mathsf{zero} = \mathsf{suc}\ x$.

Example: conatural numbers

$$
\begin{aligned}
&\mathsf{record}\ \mathbb{N}^\infty : \mathsf{Set}\ \mathsf{where} \\
&\quad \mathsf{iszero} : \mathbb{B} \\
&\quad \mathsf{pred} \quad : \mathsf{iszero} \equiv_{\mathbb{B}} \mathsf{false} \to \mathbb{N}^\infty
\end{aligned}
$$

Example: conatural numbers

record $\mathbb{N}^\infty$: Set where

  iszero : $\mathbb{B}$

  pred : iszero $\equiv_{\mathbb{B}}$ false $\to \mathbb{N}^\infty$

zero : $\mathbb{N}^\infty$

zero .iszero = true

zero .pred $\emptyset$

suc : $\mathbb{N}^\infty \to \mathbb{N}^\infty$

suc n .iszero = false

suc n .pred = n

inf : $\mathbb{N}^\infty$

inf .iszero = false

inf .pred = inf

Example: C Streams

record $\mathbb{S}$: Set where

  head : $\mathbb{N}$

  tail : $(m : \mathbb{N}) \to \text{head} \equiv_{\mathbb{N}} \text{suc}\ m \to \mathbb{S}$

Example: C Streams

$$
\begin{array}{l}
\mathsf{record}\ \mathbb{S} : \mathsf{Set}\ \mathsf{where} \\
\quad \mathsf{head} : \mathbb{N} \\
\quad \mathsf{tail} \quad : (m : \mathbb{N}) \to \mathsf{head} \equiv_{\mathbb{N}} \mathsf{suc}\ m \to \mathbb{S}
\end{array}
$$

$$
\begin{array}{llll}
\multicolumn{4}{l}{\mathsf{timer} : \mathbb{N} \to \mathbb{S}} \\
\mathsf{timer}\ n & .\mathsf{head} & & = n \\
\mathsf{timer}\ \mathsf{zero} & .\mathsf{tail} & m\ \emptyset & \\
\mathsf{timer}\ (\mathsf{suc}\ m) & .\mathsf{tail} & m\ \mathsf{refl} & = \mathsf{timer}\ m
\end{array}
$$

Example based on #2896

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$
$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$\mathsf{foo} : (m : \mathbb{N}) \to \mathsf{D}\ (\mathsf{suc}\ m) \to \mathbb{N}$$
$$\mathsf{foo}\ m\ (\mathsf{c}\ (\mathsf{suc}\ n)) = m + n$$

Example based on #2896

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$
$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$\mathsf{foo} : (m : \mathbb{N}) \to \mathsf{D}\ (\mathsf{suc}\ m) \to \mathbb{N}$$
$$\mathsf{foo}\ m\ (\mathsf{c}\ (\mathsf{suc}\ n)) = m + n$$

What does this even mean???

Dependent copattern matching

Surface and core languages

From clauses to a case tree

Preservation of first-match semantics

Term syntax (surface and core)

$$
\begin{aligned}
A, B, u, v \; &::= (x : A) \to B \mid \mathsf{Set}_\ell \\
&\mid \; \mathsf{D}\ \bar{u} \mid \mathsf{R}\ \bar{u} \mid u \equiv_A v \\
&\mid \; x\ \bar{e} \mid \mathsf{f}\ \bar{e} \mid \mathsf{c}\ \bar{u} \mid \mathsf{refl} \\
e \; &::= u \mid .\pi \\
\Delta \; &::= \epsilon \mid (x : A)\Delta
\end{aligned}
$$

Surface language

$$
\begin{aligned}
\mathit{decl} &::= \mathsf{data}\ \mathsf{D}\ \Delta : \mathsf{Set}_\ell\ \mathsf{where}\ \overline{\mathsf{c}\ \Delta} \\
&\ \ \mid\ \ \mathsf{record}\ \mathit{self} : \mathsf{R}\ \Delta : \mathsf{Set}_\ell\ \mathsf{where}\ \overline{\pi : A} \\
&\ \ \mid\ \ \mathsf{definition}\ \mathsf{f} : A\ \mathsf{where}\ \overline{\mathit{cls}} \\
\mathit{cls} &::= \bar{q} \hookrightarrow u \mid \bar{q} \hookrightarrow \mathsf{impossible} \\
q &::= p \mid .\pi \\
p &::= x \mid \mathsf{c}\ \bar{p} \mid \mathsf{refl} \mid \lfloor u \rfloor \mid \emptyset
\end{aligned}
$$

Core language: typing rules

$$\frac{\vdash \Gamma}{\Gamma \vdash \mathsf{Set}_\ell : \mathsf{Set}_{\ell+1}} \qquad \frac{\Gamma \vdash A : \mathsf{Set}_\ell \qquad \Gamma(x : A) \vdash B : \mathsf{Set}_{\ell'}}{\Gamma \vdash (x : A) \to B : \mathsf{Set}_{\max(\ell,\ell')}}$$

$$\frac{\mathsf{D} : \mathsf{Set}_\ell \in \Sigma}{\Gamma \vdash \mathsf{D} : \mathsf{Set}_\ell} \qquad \frac{\mathsf{R} : \mathsf{Set}_\ell \in \Sigma}{\Gamma \vdash \mathsf{R} : \mathsf{Set}_\ell} \qquad \frac{\Gamma \vdash A : \mathsf{Set}_\ell \qquad \Gamma \vdash u : A \qquad \Gamma \vdash v : A}{\Gamma \vdash u \equiv_A v : \mathsf{Set}_\ell}$$

$$\frac{x : A \in \Gamma \qquad \Gamma \mid x : A \vdash \bar{e} : C}{\Gamma \vdash x\ \bar{e} : C} \qquad \frac{\mathsf{f} : A \in \Sigma \qquad \Gamma \mid \mathsf{f} : A \vdash \bar{e} : C}{\Gamma \vdash \mathsf{f}\ \bar{e} : C}$$

$$\frac{\mathsf{c}\ \Delta_\mathsf{c} : \mathsf{D} \in \Sigma \qquad \Gamma \vdash \bar{v} : \Delta_\mathsf{c}}{\Gamma \vdash \mathsf{c}\ \bar{v} : \mathsf{D}} \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathsf{refl} : u \equiv_A u}$$

$$\frac{\Gamma \vdash v : A \qquad \Gamma \mid u\ v : B[v \,/\, x] \vdash \bar{e} : C}{\Gamma \mid u : (x : A) \to B \vdash v\ \bar{e} : C}$$

$$\frac{\mathit{self} : \mathsf{R} \vdash .\pi : A \in \Sigma \qquad \Gamma \mid u\ .\pi : A[u \,/\, \mathit{self}] \vdash \bar{e} : C}{\Gamma \mid u : \mathsf{R} \vdash .\pi\ \bar{e} : C}$$

$$\frac{\Gamma \vdash u : A \qquad \Gamma \vdash A = B}{\Gamma \vdash u : B} \qquad \frac{\Gamma \vdash A = A' \qquad \Gamma \mid u : A' \vdash \bar{e} : C}{\Gamma \mid u : A \vdash \bar{e} : C}$$

Core language: case trees

$$
\begin{aligned}
Q ::= \; & u \\
\mid \; & \lambda x.\ Q \\
\mid \; & \mathsf{record}\{\pi_1 \mapsto Q_1; \ldots; \pi_n \mapsto Q_n\} \\
\mid \; & \mathsf{case}_x\{\mathsf{c}_1\ \hat{\Delta}_1 \mapsto Q_1; \ldots; \mathsf{c}_n\ \hat{\Delta}_n \mapsto Q_n\} \\
\mid \; & \mathsf{case}_x\{\mathsf{refl} \mapsto^{\tau} Q\}
\end{aligned}
$$

Case tree typing

$$\Gamma \mid \mathsf{f}\ \bar{q} : A \vdash Q$$

"The case tree Q gives a well-typed implementation of f applied to copatterns $\bar{q}$"

Case tree typing: v

$$\frac{\Gamma \vdash v : C}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash v}$$

Side effect: $\Sigma := \Sigma, (\Gamma \vdash \mathsf{f}\ \bar{q} \hookrightarrow v : C)$

Case tree typing: $\lambda x.\ Q$

$$\frac{\Gamma(x : A) \mid \mathsf{f}\ \bar{q}\ x : B \vdash Q}{\Gamma \mid \mathsf{f}\ \bar{q} : (x : A) \to B \vdash \lambda x.\ Q}$$

Case tree typing: record$\{\ldots\}$

$$\frac{\mathsf{record}\ \mathit{self} : \mathsf{R} : \mathsf{Set}_\ell\ \mathsf{where}\ \overline{\pi_{\mathsf{i}} : A_i} \in \Sigma \qquad (\Gamma \mid \mathsf{f}\ \bar{q}\ .\pi_{\mathsf{i}} : A_i[\mathsf{f}\ \lceil \bar{q} \rceil \,/\, \mathit{self}] \vdash Q_i)_{i=1\ldots n}}{\Gamma \mid \mathsf{f}\ \bar{q} : \mathsf{R} \vdash \mathsf{record}\{\pi_1 \mapsto Q_1; \ldots ; \pi_{\mathsf{n}} \mapsto Q_n\}}$$

Case tree typing: $\mathsf{case}_x\{\ldots\}$

$$\dfrac{\mathsf{D} : \mathsf{Set}_\ell \ \mathsf{where}\ \overline{\mathsf{c_i}\ \Delta_i} \in \Sigma \qquad \left(\begin{array}{c} \rho_i = [\mathsf{c_i}\ \hat{\Delta}_i \,/\, x] \\ \Gamma_1 \Delta_i (\Gamma_2 \rho_i) \mid \mathsf{f}\ \bar{q}\rho_i : C\rho_i \vdash Q_i \end{array}\right)_{i=1\ldots n}}{\begin{array}{l} \Gamma_1 (x : \mathsf{D}) \Gamma_2 \mid \mathsf{f}\ \bar{q} : C \vdash \\ \qquad \mathsf{case}_x\{\mathsf{c_1}\ \hat{\Delta}_1 \mapsto Q_1; \ldots; \mathsf{c_n}\ \hat{\Delta}_n \mapsto Q_n\} \end{array}}$$

Case tree typing: $\mathsf{case}_x\{\mathsf{refl} \mapsto^\tau Q\}$

$$\frac{\Gamma_1 \vdash u =^? v : B \Rightarrow \textsc{yes}(\Gamma_1', \rho, \tau) \qquad \Gamma_1'(\Gamma_2\rho) \mid \mathsf{f}\ \bar{q}\rho : C\rho \vdash Q}{\Gamma_1(x : u \equiv_B v)\Gamma_2 \mid \mathsf{f}\ \bar{q} : C \vdash \mathsf{case}_x\{\mathsf{refl} \mapsto^\tau Q\}}$$

$$\begin{pmatrix} \Gamma_1' \vdash u\rho = v\rho : A\rho \\ \Gamma_1' \vdash \tau; \rho = \mathbb{1} : \Gamma_1' \end{pmatrix}$$

Case tree typing: $\mathsf{case}_x\{\}$

$$\frac{\Gamma_1 \vdash u =^{?} v : B \Rightarrow \mathrm{NO}}{\Gamma_1(x : u \equiv_B v)\Gamma_2 \mid \mathsf{f}\ \bar{q} : C \vdash \mathsf{case}_x\{\}}$$

Dependent copattern matching

Surface and core languages

From clauses to a case tree

Preservation of first-match semantics

From clauses to a case tree

The clauses guide us in the construction of a well-typed case tree:

From clauses to a case tree

The clauses guide us in the construction of a well-typed case tree: as we construct the case tree, we deconstruct the clauses.

From clauses to a case tree

The clauses guide us in the construction of a well-typed case tree: as we construct the case tree, we deconstruct the clauses.

$$\Gamma \mid \mathsf{f}\ \bar{q} : A \vdash P \rightsquigarrow Q$$

From clauses to a case tree

The clauses guide us in the construction of a well-typed case tree: as we construct the case tree, we deconstruct the clauses.

$$\Gamma \mid \mathsf{f}\ \bar{q} : A \vdash P \leadsto Q$$

$$\text{entails } \Gamma \mid \mathsf{f}\ \bar{q} : A \vdash Q$$

From clauses to a case tree

The clauses guide us in the construction of a well-typed case tree: as we construct the case tree, we deconstruct the clauses.

$$\Gamma \mid \mathsf{f}\ \bar{q} : A \vdash P \rightsquigarrow Q$$

$$\text{entails } \Gamma \mid \mathsf{f}\ \bar{q} : A \vdash Q$$

$$P = \left\{ [w_{ik} \ /^{?}\ p_{ik}]\ \bar{q}_i \hookrightarrow rhs_i \right\}_{i=1\ldots n}$$

$$\mathsf{max} : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$$

$$\begin{array}{lll}
\mathsf{zero} & j & \hookrightarrow j \\
i & \mathsf{zero} & \hookrightarrow i \\
(\mathsf{suc}\ k) & (\mathsf{suc}\ l) & \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)
\end{array}$$

$$(m : \mathbb{N}) \mid \mathsf{max}\ m : \mathbb{N} \to \mathbb{N}$$

$$\begin{array}{lll}
[m \mathrel{/^?} \mathsf{zero}] & j & \hookrightarrow j \\
[m \mathrel{/^?} i] & \mathsf{zero} & \hookrightarrow i \\
[m \mathrel{/^?} \mathsf{suc}\ k] & (\mathsf{suc}\ l) & \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)
\end{array}$$

$$\mathsf{max}\ \mathsf{zero} : \mathbb{N} \to \mathbb{N}$$

$$\begin{array}{lll}
[\mathsf{zero} \mathrel{/^?} \mathsf{zero}] & j & \hookrightarrow j \\
[\mathsf{zero} \mathrel{/^?} i] & \mathsf{zero} & \hookrightarrow i \\
[\mathsf{zero} \mathrel{/^?} \mathsf{suc}\ k] & (\mathsf{suc}\ l) & \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)
\end{array}$$

$$(p : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p) : \mathbb{N} \to \mathbb{N}$$

$$\begin{array}{lll}
[\mathsf{suc}\ p \mathrel{/^?} \mathsf{zero}] & j & \hookrightarrow j \\
[\mathsf{suc}\ p \mathrel{/^?} i] & \mathsf{zero} & \hookrightarrow i \\
[\mathsf{suc}\ p \mathrel{/^?} \mathsf{suc}\ k] & (\mathsf{suc}\ l) & \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)
\end{array}$$

$\mathsf{max}\ \mathsf{zero} : \mathbb{N} \to \mathbb{N}$

$$
\begin{aligned}
j \qquad &\hookrightarrow j \\
[\mathsf{zero} \ /^{?}\ i]\ \mathsf{zero} &\hookrightarrow i
\end{aligned}
$$

$(p : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p) : \mathbb{N} \to \mathbb{N}$

$$
\begin{aligned}
[\mathsf{suc}\ p \ /^{?}\ i]\ \mathsf{zero} \qquad &\hookrightarrow i \\
[p \ /^{?}\ k] \qquad (\mathsf{suc}\ l) &\hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)
\end{aligned}
$$

$$(n : \mathbb{N}) \mid \mathsf{max}\ \mathsf{zero}\ n : \mathbb{N}$$

$$\begin{aligned} &[n \mathrel{/^?} j] && \hookrightarrow j \\ &[\mathsf{zero} \mathrel{/^?} i, n \mathrel{/^?} \mathsf{zero}] && \hookrightarrow i \end{aligned}$$

$$(p : \mathbb{N})(n : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ n : \mathbb{N}$$

$$\begin{aligned} &[\mathsf{suc}\ p \mathrel{/^?} i, n \mathrel{/^?} \mathsf{zero}] && \hookrightarrow i \\ &[p \mathrel{/^?} k, n \mathrel{/^?} \mathsf{suc}\ l] && \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l) \end{aligned}$$

$$(n : \mathbb{N}) \mid \mathsf{max}\ \mathsf{zero}\ n \hookrightarrow n : \mathbb{N}$$

$$(p : \mathbb{N})(n : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ n : \mathbb{N}$$

$$\begin{aligned} &[\mathsf{suc}\ p \mathbin{/^?} i, n \mathbin{/^?} \mathsf{zero}] &&\hookrightarrow i \\ &[p \mathbin{/^?} k, n \mathbin{/^?} \mathsf{suc}\ l] &&\hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l) \end{aligned}$$

$$(n : \mathbb{N}) \mid \mathsf{max}\ \mathsf{zero}\ n \hookrightarrow n : \mathbb{N}$$

$$(p : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ \mathsf{zero} : \mathbb{N}$$

$$[\mathsf{suc}\ p \ /^{?}\ i] \hookrightarrow i$$

$$(p : \mathbb{N})(q : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ (\mathsf{suc}\ q) : \mathbb{N}$$

$$[p \ /^{?}\ k, q \ /^{?}\ l] \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)$$

$$(n : \mathbb{N}) \mid \mathsf{max}\ \mathsf{zero}\ n \hookrightarrow n : \mathbb{N}$$

$$(p : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ \mathsf{zero} \hookrightarrow \mathsf{suc}\ p : \mathbb{N}$$

$$(p : \mathbb{N})(q : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ (\mathsf{suc}\ q) : \mathbb{N}$$

$$[p \mathbin{/^?} k, q \mathbin{/^?} l] \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ k\ l)$$

$$(n : \mathbb{N}) \mid \mathsf{max}\ \mathsf{zero}\ n \hookrightarrow n : \mathbb{N}$$

$$(p : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ \mathsf{zero} \hookrightarrow \mathsf{suc}\ p : \mathbb{N}$$

$$\begin{aligned}(p : \mathbb{N})(q : \mathbb{N}) \mid \mathsf{max}\ (\mathsf{suc}\ p)\ (\mathsf{suc}\ q) \\ \hookrightarrow \mathsf{suc}\ (\mathsf{max}\ p\ q) : \mathbb{N}\end{aligned}$$

Case tree for max

$$\lambda m.\ \mathsf{case}_m \left\{ \begin{array}{l} \mathsf{zero} \mapsto \lambda n.\ n \\ \mathsf{suc}\ p \mapsto \\ \qquad \lambda n.\ \mathsf{case}_n \left\{ \begin{array}{l} \mathsf{zero} \mapsto \mathsf{suc}\ p \\ \mathsf{suc}\ q \mapsto \\ \qquad \mathsf{suc}\ (\mathsf{max}\ p\ q) \end{array} \right\} \end{array} \right\}$$

$\text{zero} : \mathbb{N}^\infty$

$$
\begin{aligned}
&.\text{iszero} && \hookrightarrow \text{true} \\
&.\text{pred} \quad \emptyset && \hookrightarrow \text{impossible}
\end{aligned}
$$

$\mathsf{zero}\ .\mathsf{iszero} : \mathbb{B}$

$\hookrightarrow \mathsf{true}$

$\mathsf{zero}\ .\mathsf{pred} : \mathsf{zero}\ .\mathsf{iszero} \equiv_{\mathbb{B}} \mathsf{false} \to \mathbb{N}^{\infty}$

$\emptyset \hookrightarrow \mathsf{impossible}$

$\mathsf{zero}\ .\mathsf{iszero} \hookrightarrow \mathsf{true} : \mathbb{B}$

$\mathsf{zero}\ .\mathsf{pred} : \mathsf{zero}\ .\mathsf{iszero} \equiv_{\mathbb{B}} \mathsf{false} \to \mathbb{N}^{\infty}$

$\emptyset \hookrightarrow \mathsf{impossible}$

$\mathsf{zero}\ .\mathsf{iszero} \hookrightarrow \mathsf{true} : \mathbb{B}$

$x : \mathsf{zero}\ .\mathsf{iszero} \equiv_{\mathbb{B}} \mathsf{false} \mid \mathsf{zero}\ .\mathsf{pred}\ x : \mathbb{N}^{\infty}$

$[x \ /^{?}\ \emptyset] \hookrightarrow \mathsf{impossible}$

$\mathsf{zero}\ \mathsf{.iszero} \hookrightarrow \mathsf{true} : \mathbb{B}$

Case tree for zero

$$\mathsf{record}\left\{\begin{array}{lll}\mathsf{iszero} & \mapsto & \mathsf{true} \\ \mathsf{pred} & \mapsto & \lambda x.\ \mathsf{case}_x\{\}\end{array}\right\}$$

Case tree for timer

$$\lambda n.\ \mathsf{record}\left\{\begin{array}{l} \mathsf{head} \mapsto n \\ \mathsf{tail} \mapsto \lambda m, p. \\ \qquad \mathsf{case}_n\left\{\begin{array}{l} \mathsf{zero} \mapsto \mathsf{case}_p\{\} \\ \mathsf{suc}\ n' \mapsto \\ \qquad \mathsf{case}_p\left\{\begin{array}{l} \mathsf{refl} \mapsto^{\mathbb{1}} \\ \qquad \mathsf{timer}\ m \end{array}\right\} \end{array}\right\} \end{array}\right\}$$

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$
$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$\mathsf{foo} : (m : \mathbb{N}) \to \mathsf{D}\ (\mathsf{suc}\ m) \to \mathbb{N}$$

$$\quad m\ (\mathsf{c}\ (\mathsf{suc}\ n)) \hookrightarrow m + n$$

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$
$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$(m : \mathbb{N})(x : \mathsf{D}\ (\mathsf{suc}\ m)) \mid \mathsf{foo}\ m\ x : \mathbb{N}$$

$$[m \mathbin{/^?} m, x \mathbin{/^?} \mathsf{c}\ (\mathsf{suc}\ n)] \hookrightarrow m + n$$

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$
$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$(m : \mathbb{N}) \mid \mathsf{foo}\ m\ (\mathsf{c}\ (\mathsf{suc}\ m)) : \mathbb{N}$$

$$[m \ /^? \ m, \mathsf{c}\ (\mathsf{suc}\ m) \ /^? \ \mathsf{c}\ (\mathsf{suc}\ n)] \hookrightarrow m + n$$

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$

$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$(m : \mathbb{N}) \mid \mathsf{foo}\ m\ (\mathsf{c}\ (\mathsf{suc}\ m)) : \mathbb{N}$$

$$[m \mathbin{/^?} m, m \mathbin{/^?} n] \hookrightarrow m + n$$

$$\mathsf{data}\ \mathsf{D} : \mathbb{N} \to \mathsf{Set}\ \mathsf{where}$$
$$\quad \mathsf{c} : (n : \mathbb{N}) \to \mathsf{D}\ n$$

$$(m : \mathbb{N}) \mid \mathsf{foo}\ m\ (\mathsf{c}\ (\mathsf{suc}\ m)) \hookrightarrow m + m : \mathbb{N}$$

Case tree for foo

$$\lambda m, x.\ \mathsf{case}_x \left\{ \begin{array}{l} \mathsf{c}\ n\ p \mapsto \\ \qquad \mathsf{case}_p \left\{ \mathsf{refl} \mapsto^{\mathbb{1}_m} (m + m) \right\} \end{array} \right\}$$

Constructing a case tree: nothing to split

$$\frac{\bar{q}_1 = \epsilon \qquad \Gamma \vdash E_1 \Rightarrow \textsc{solved}(\sigma) \qquad rhs_1 = v \qquad \Gamma \vdash v\sigma : C}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \leadsto v\sigma}$$

Side effect: $\Sigma = \Sigma, \Gamma \vdash \mathsf{f}\ \bar{q} \hookrightarrow v\sigma : C$

Constructing a case tree: introduce new variable

$$\frac{\bar{q}_1 = p\ \bar{q}_1' \qquad C \searrow (x : A) \to B \qquad \Gamma(x : A) \mid \mathsf{f}\ \bar{q}\ x : B \vdash P\ (x : A) \leadsto Q}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \leadsto \lambda x.\ Q}$$

Constructing a case tree: split on result

$$\frac{\bar{q}_1 = .\pi_{\mathsf{i}}\ \bar{q}'_1 \qquad C \searrow \mathsf{R} \qquad \mathsf{record}\ \mathit{self} : \mathsf{R} : \mathsf{Set}_\ell\ \mathsf{where}\ \overline{\pi_{\mathsf{i}} : A_i} \in \Sigma \qquad (\Gamma \mid \mathsf{f}\ \bar{q}\ .\pi_{\mathsf{i}} : A_i[\mathsf{f}\ \lceil \bar{q} \rceil\ /\ \mathit{self}] \vdash P\ .\pi_{\mathsf{i}} \rightsquigarrow Q_i)_{i=1\ldots n}}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \rightsquigarrow \mathsf{record}\{\pi_1 \mapsto Q_1; \ldots; \pi_{\mathsf{n}} \mapsto Q_n\}}$$

Constructing a case tree: absurd split on result

$$\frac{\bar{q}_1 = \emptyset \qquad m = 1 \qquad C \searrow \mathsf{R}\ \bar{v} \qquad \mathsf{record}\ \_ : \mathsf{R} : \mathsf{Set}_\ell\ \mathsf{where}\ \epsilon \in \Sigma \qquad \mathit{rhs}_1 = \mathsf{impossible}}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \rightsquigarrow \mathsf{record}\{\}}$$

Constructing a case tree: split on variable

$$
\frac{
\begin{array}{c}
(x \mathbin{/^?} \mathsf{c_j}\ \bar{p} : A) \in E_1 \qquad A \searrow \mathsf{D} \qquad \Gamma = \Gamma_1 (x : A) \Gamma_2 \\
\mathsf{data}\ \mathsf{D} : \mathsf{Set}_\ell\ \mathsf{where}\ \overline{\mathsf{c_i}\ \Delta_i} \in \Sigma \\
\left(\begin{array}{c} \rho_i = [\mathsf{c_i}\ \hat{\Delta}_i \mathbin{/} x] \qquad P\rho_i \Rightarrow P_i \\ (\Gamma_1 \Delta_i (\Gamma_2 \rho_i) \mid \mathsf{f}\ \bar{q}\rho_i : C\rho_i \vdash P_i \rightsquigarrow Q_i \end{array} \right)_{i=1\ldots n}
\end{array}
}{
\begin{array}{l}
\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \\
\qquad \rightsquigarrow \mathsf{case}_x \{ \mathsf{c_1}\ \hat{\Delta}_1 \mapsto Q_1 ; \ldots ; \mathsf{c_n}\ \hat{\Delta}_n \mapsto Q_n \}
\end{array}
}
$$

Constructing a case tree: split on equation

$$\frac{\begin{array}{cc} (x \,/^{?}\, \mathsf{refl} : A) \in E_1 & A \searrow u \equiv_B v \\ \Gamma = \Gamma_1(x : A)\Gamma_2 & \Gamma_1 \vdash u =^{?} v : B \Rightarrow \mathrm{YES}(\Gamma_1', \rho, \tau) \\ \Sigma \vdash P\rho \Rightarrow P' & \Gamma_1'(\Gamma_2\rho) \mid \mathsf{f}\ \bar{q}\rho : C\rho \vdash P' \rightsquigarrow Q \end{array}}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \rightsquigarrow \mathsf{case}_x\{\mathsf{refl} \mapsto^{\tau'} Q\}}$$

Constructing a case tree: split on empty type

$$\frac{(x \;/^{?}\; \emptyset : A) \in E_1 \qquad \Gamma \vdash \emptyset : A \qquad rhs_1 = \textsf{impossible}}{\Gamma \mid \mathsf{f}\ \bar{q} : C \vdash P \rightsquigarrow \textsf{case}_x\{\}}$$

Dependent copattern matching

Surface and core languages

From clauses to a case tree

Preservation of first-match semantics

Evaluation of case trees

$$(\lambda x.\ Q)\sigma\ u\ \bar{e} \longrightarrow Q(\sigma \uplus [u\,/\,x])\ \bar{e}$$

Evaluation of case trees

$$(\lambda x.\ Q)\sigma\ u\ \bar{e} \longrightarrow Q(\sigma \uplus [u \,/\, x])\ \bar{e}$$

$$(\mathsf{record}\{\pi_1 \mapsto Q_1; \ldots; \pi_\mathsf{n} \mapsto Q_n\})\sigma\ .\pi_\mathsf{i}\ \bar{e} \longrightarrow Q_i\sigma\ \bar{e}$$

Evaluation of case trees

$$(\lambda x.\ Q)\sigma\ u\ \bar{e} \longrightarrow Q(\sigma \uplus [u \,/\, x])\ \bar{e}$$

$$(\mathsf{record}\{\pi_1 \mapsto Q_1; \ldots; \pi_\mathsf{n} \mapsto Q_n\})\sigma\ .\pi_\mathsf{i}\ \bar{e} \longrightarrow Q_i\sigma\ \bar{e}$$

$$\begin{aligned}&(\mathsf{case}_x\{\mathsf{c}_1\ \hat{\Delta}_1 \mapsto Q_1; \ldots; \mathsf{c}_\mathsf{n}\ \hat{\Delta}_n \mapsto Q_n\})\sigma\ \bar{e}\\ &\qquad \longrightarrow Q_i(\sigma\backslash x \uplus [\bar{u} \,/\, \hat{\Delta}_i])\ \bar{e}\end{aligned}$$

$$(\text{if } x\sigma \searrow \mathsf{c}\ \bar{u})$$

Evaluation of case trees

$$(\lambda x.\ Q)\sigma\ u\ \bar{e} \longrightarrow Q(\sigma \uplus [u \,/\, x])\ \bar{e}$$

$$(\mathsf{record}\{\pi_1 \mapsto Q_1; \ldots; \pi_n \mapsto Q_n\})\sigma\ .\pi_i\ \bar{e} \longrightarrow Q_i\sigma\ \bar{e}$$

$$(\mathsf{case}_x\{\mathsf{c}_1\ \hat{\Delta}_1 \mapsto Q_1; \ldots; \mathsf{c}_n\ \hat{\Delta}_n \mapsto Q_n\})\sigma\ \bar{e} \longrightarrow Q_i(\sigma\backslash x \uplus [\bar{u} \,/\, \hat{\Delta}_i])\ \bar{e} \qquad (\text{if } x\sigma \searrow \mathsf{c}\ \bar{u})$$

$$(\mathsf{case}_x\{\mathsf{refl} \mapsto^{\tau} Q\})\sigma\ \bar{e} \longrightarrow Q(\tau; \sigma)\ \bar{e} \qquad (\text{if } x\sigma \searrow \mathsf{refl})$$

Matching algorithm

$$\frac{}{[v \,/\, x] \searrow [v \,/\, x]} \qquad \frac{}{[v \,/\, \lfloor u \rfloor] \searrow []} \qquad \frac{v \searrow \mathsf{refl}}{[v \,/\, \mathsf{refl}] \searrow []}$$

$$\frac{v \searrow \mathsf{c}\ \bar{u} \qquad [\bar{u} \,/\, \bar{p}] \searrow \sigma_\perp}{[v \,/\, \mathsf{c}\ \bar{p}] \searrow \sigma_\perp} \qquad \frac{v \searrow \mathsf{c}_2\ \bar{u} \qquad \mathsf{c}_1 \neq \mathsf{c}_2}{[v \,/\, \mathsf{c}_1\ \bar{p}] \searrow \perp}$$

$$\frac{}{[.\pi \,/\, .\pi] \searrow []} \qquad \frac{\pi_1 \neq \pi_2}{[.\pi_2 \,/\, .\pi_1] \searrow \perp}$$

$$\frac{}{[\epsilon \,/\, \epsilon] \searrow []} \qquad \frac{[e \,/\, q] \searrow \sigma_\perp \qquad [\bar{e} \,/\, \bar{q}] \searrow \tau_\perp}{[e\ \bar{e} \,/\, q\ \bar{q}] \searrow \sigma_\perp \uplus \tau_\perp}$$

First-match semantics

If f is given by $\{\bar{q}_i \hookrightarrow \mathit{rhs}_i \mid i = 1 \ldots n\}$ and

- $[\bar{e} \,/\, \bar{q}_j] \searrow \perp$ for $j = 1 \ldots i-1$
- $[\bar{e} \,/\, \bar{q}_i] \searrow \sigma$

then f $\bar{e} \longrightarrow u_i\sigma$.

Preservation of first-match semantics

Let f be given by $P = \{\bar{q}_i \hookrightarrow rhs_i\}_{i=1\ldots n}$. If

- $\epsilon \mid \mathsf{f} : A \vdash P \rightsquigarrow Q$
- $\Gamma \mid \mathsf{f} : A \vdash \bar{\mathsf{e}} : B$
- $\mathsf{f}\ \bar{e} \longrightarrow u$ (according to first-match)

then also $Q\ \bar{e} \longrightarrow u$.

The shortcut rule for matching

Can we allow the following rule?

$$\frac{[e \mathbin{/} q] \searrow \bot}{[e\ \bar{e} \mathbin{/} q\ \bar{q}] \searrow \bot}$$

The shortcut rule for matching

Can we allow the following rule?

$$\frac{[e \,/\, q] \searrow \bot}{[e \; \bar{e} \,/\, q \; \bar{q}] \searrow \bot}$$

No! Otherwise this function has no case tree:

$$\begin{array}{l} \mathsf{f} : (A : \mathsf{Set}) \to A \to \mathbb{B} \to (A \equiv_{\mathsf{Set}} \mathbb{B}) \to \mathbb{B} \\ \mathsf{f} \ \lfloor \mathbb{B} \rfloor \ \mathsf{true} \ \mathsf{true} \ \mathsf{refl} = \mathsf{true} \\ \mathsf{f} \ \_ \quad \_ \quad \_ \quad \_ \quad = \mathsf{false} \end{array}$$

Conclusion

To use a typechecker or proof assistant,
we need to trust not only the core language
but also the elaboration algorithm.

Conclusion

To use a typechecker or proof assistant,
we need to trust not only the core language
but also the elaboration algorithm.

It pays off to formalize elaboration:
I found a bug in Agda by writing the proof.

Conclusion

To use a typechecker or proof assistant,
we need to trust not only the core language
but also the elaboration algorithm.

It pays off to formalize elaboration:
I found a bug in Agda by writing the proof.

Let's work together on a formally verified
typechecker for dependent types!