Relating System F and $\lambda 2$: A Case Study in Coq, Abella and Beluga

Jonas Kaiser   Brigitte Pientka   Gert Smolka

FSCD 2017, Oxford

September 4, 2017

SAARLAND UNIVERSITY
COMPUTER SCIENCE

System F [Girard '72] / PTLC [Reynolds '74]

Some History

- Developed in the context of proof theory and polymorphism.
- Commonly phrased as a two-sorted system: *Types* & *Terms*
- We consider F as presented in [Harper '13].
  - Explicitly scopes type variables.

Meanwhile ...

- Study of CC led to single-sorted Pure Type Systems (PTS):
  - The λ-cube of [Barendregt '91].
- System F appears as the corner $\lambda 2$.

Goal: Transport of Results

$$\mathsf{F} \quad \leftrightsquigarrow \quad \lambda 2$$

bidirectional reduction of typing

Related Work

- The reduction result is partially discussed in [Geuvers '93].
  - Primarily argues the forward preservation of typing.
  - The syntactic correspondence is left implicit.
- Coq formalisation of the full reduction in [K/Tebbi/Smolka '17].
  - Pairs of translation functions establish the syntactic correspondence.
  - Requires involved cancellation laws.
  - Proofs based on an extension of context morphism lemmas [Goguen/McKinna '97, Adams '06].
- Goal of this work:

Correspondence Proof as *benchmark*
for reasoning about
syntax and *contextual information*.

Syntactic Variants F and $\lambda 2$

SAARLAND UNIVERSITY
COMPUTER SCIENCE

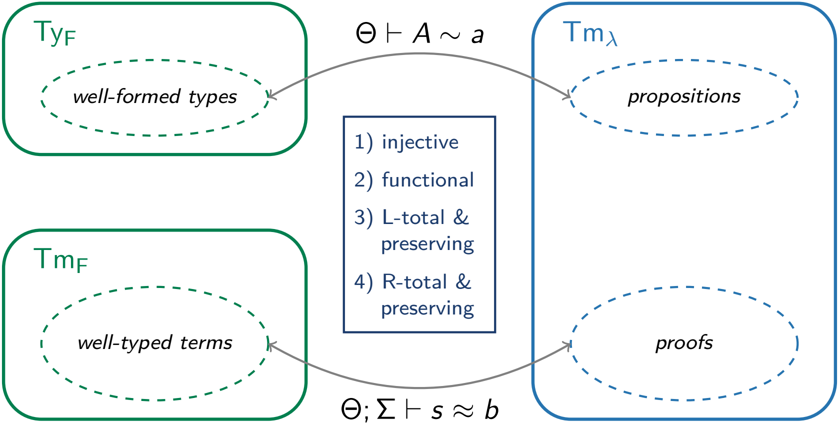
Two-sorted non-uniform syntax:

$\mathsf{Ty_F}$	$A, B ::= X \mid A \to B \mid \forall X.A$
$\mathsf{Tm_F}$	$s, t ::= x \mid s\ t \mid \lambda x : A.s \mid s\ A \mid \Lambda X.s$
Type Formation	$\Delta \vdash A\ \mathbf{ty}$
Typing	$\Delta; \Gamma \vdash s :_F A$

Single-sorted uniform PTS syntax:

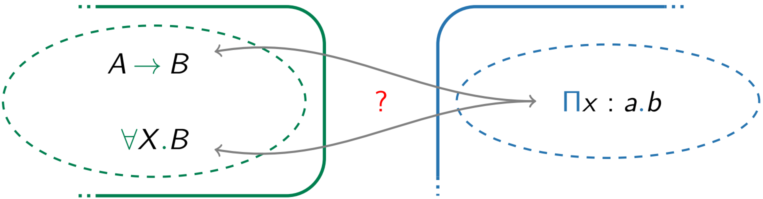
Tm_λ	$a, b ::= x \mid * \mid \square \mid a\ b \mid \lambda x : a.b \mid \Pi x : a.b$
Typing	$\Psi \vdash a :_2 b$

Syntactic Correspondence



Syntactic Correspondence – Two Complications

1. Non-uniform vs. uniform:



2. Open terms & contextual assumptions about
   - *well-formedness*: in $X \to X$, is X in scope?
   - *typing*: in $a\ b$, is b a proof or proposition?
   - *related variables*: in the variable case, does $\Theta \vdash X \sim x$ hold?

The Reduction Proof: F $\leftrightsquigarrow$ $\lambda 2$

Assume we are given syntactic relations $\sim$ and $\approx$ which are both:

1. functional
2. injective
3. left-total and judgement preserving on suitable fragment
4. right-total and judgement preserving on suitable fragment

Theorem (Reduction F $\rightsquigarrow$ $\lambda 2$)

$$\vdash A \; \mathbf{ty} \iff \exists a.\ \vdash A \sim a \ \wedge\ \vdash a :_2 *$$

$$\vdash s :_F A \iff \exists ba.\ \vdash s \approx b \ \wedge\ \vdash A \sim a \ \wedge\ \vdash b :_2 a \ \wedge\ \vdash a :_2 *$$

Theorem (Reduction $\lambda 2$ $\rightsquigarrow$ F)

$$\vdash a :_2 * \iff \exists A.\ \vdash A \sim a \ \wedge\ \vdash A \; \mathbf{ty}$$

$$\vdash b :_2 a \ \wedge\ \vdash a :_2 * \iff \exists sA.\ \vdash s \approx b \ \wedge\ \vdash A \sim a \ \wedge\ \vdash s :_F A$$

Formalising the Proof

We consider three approaches:

- Coq — first-order de Bruijn, par. substitutions, invariants
- Abella — HOAS, ∇-quantification, relational proof search
- Beluga — HOAS, 1^{st}-class contexts, context schemas

Topics of Interest

- Representation of syntax and judgements.
- Management of local variable binding.
- Tracking of contextual information.
- Technicalities: Usability / Libraries / Tool Support

SAARLAND UNIVERSITY
COMPUTER SCIENCE

– Coq –

first-order de Bruijn, parallel substitutions, invariants

Coq – Representation

- Syntax: first-order de Bruijn

$$A, B ::= n_{\mathsf{ty}} \mid A \to B \mid \forall.A \qquad n \in \mathbb{N}$$
$$s, t ::= n_{\mathsf{tm}} \mid s\ t \mid \lambda A.s \mid s\ A \mid \Lambda.s$$

- Typing contexts:

$\Delta : \mathbb{N}$ – *excl. upper bound for free type variables*

$\Gamma : \text{list } \mathsf{Ty}_\mathsf{F}$ – *dangling indices reference by position*

- Judgements as inductive predicates, e.g.:

$$\_;\_ \vdash \_ :_\mathsf{F} \_ \quad : \quad \mathbb{N} \to \text{list } \mathsf{Ty}_\mathsf{F} \to \mathsf{Tm}_\mathsf{F} \to \mathsf{Ty}_\mathsf{F} \to \mathbf{Prop}$$

- Parallel substitutions from *Autosubst* library [Schäfer/Tebbi/Smolka '15]:

$$\sigma : \mathbb{N} \to \mathcal{T} \qquad (\forall.A)[\sigma] = \forall.A[\Uparrow\sigma] \qquad \Uparrow\sigma := 0_{\mathsf{ty}} \cdot (\sigma \circ \uparrow)$$

Coq – Relating Indices

- Relating open terms requires *explicit* tracking of related indices:

$$R, S : \text{list } (\mathbb{N} \times \mathbb{N})$$

- Traversal of binders requires context adjustments:

$$\frac{R \vdash A \sim a \qquad R^{\Uparrow} \vdash B \sim b}{R \vdash A \to B \sim \Pi a.b} \qquad \frac{R^{\text{ext}} \vdash A \sim a}{R \vdash \forall.A \sim \Pi *.a}$$

$$\begin{aligned} R^{\text{ext}} &:= (0,0) :: \text{map } (\uparrow \times \uparrow)\ R \\ R^{\Uparrow} &:= \text{map } (\text{id} \times \uparrow)\ R \end{aligned}$$

$$\frac{R \vdash A \sim a \qquad R^{\Uparrow}; S^{\text{ext}} \vdash s \approx b}{R; S \vdash \lambda A.s \approx \lambda a.b}$$

Coq – Custom Invariants

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Left-Totality and Preservation of Type Formation of $\sim$

1. Define Invariant:

$$\Delta \xrightarrow{R} \Psi \; := \; \forall x < \Delta.\ \exists y.\ (x, y) \in R \;\wedge\; (y :_2 *) \in_\lambda \Psi$$

2. Prove Extension Laws:

$$\Delta \xrightarrow{R} \Psi \;\Rightarrow\; \Delta \xrightarrow{R^{\Uparrow}} \Psi, a \qquad \text{– *ext. with new term variable*}$$

$$\Delta \xrightarrow{R} \Psi \;\Rightarrow\; \Delta + 1 \xrightarrow{R^{\text{ext}}} \Psi, * \qquad \text{– *ext. with new type variable*}$$

3. Prove by induction on $\Delta \vdash A$ **ty**:

$$\Delta \vdash A \ \textbf{ty} \;\Rightarrow\; \forall R, \Psi.\ \Delta \xrightarrow{R} \Psi \;\Rightarrow\; \exists a.\ R \vdash A \sim a \;\wedge\; \Psi \vdash a :_2 *$$

4. Repeat for remaining three preservation results.

– *Abella* –

HOAS, ∇-quantification, relational proof search

Abella [Miller, Chaudhuri et al. '14]

Two-level logic:

- *Specification Level:* λProlog, HOAS, logic predicates, proof search

$$\lambda\_.\_ \;:\; \mathsf{Ty_F} \to (\mathsf{Tm_F} \to \mathsf{Tm_F}) \to \mathsf{Tm_F}$$
$$\Pi\_.\_ \;:\; \mathsf{Tm}_\lambda \to (\mathsf{Tm}_\lambda \to \mathsf{Tm}_\lambda) \to \mathsf{Tm}_\lambda$$

$$\_ :_{\mathsf{F}} \_ \;:\; \mathsf{Tm_F} \to \mathsf{Ty_F} \to \mathbf{o} \qquad + \lambda\text{Prolog rules}$$
$$\_ \approx \_ \;:\; \mathsf{Tm_F} \to \mathsf{Tm}_\lambda \to \mathbf{o} \qquad + \lambda\text{Prolog rules}$$

- *Reasoning Level:* $\mathcal{G}$ – intuitionistic, predicative, STT, ∇-quantification

$n_1, n_2, \ldots$	– *nominals represent free variables*
$\nabla x.\ \nabla y.\ x \neq y$	– *theorem of $\mathcal{G}$*
$\{L \vdash J\}$	– *logical embedding*

Abella – Logical Embedding

$$\{\_ \vdash \_\} \;:\; [\mathbf{o}] \to \mathbf{o} \to \mathbf{Prop}$$

- $\{L \vdash J\}$ holds in $\mathcal{G}$ iff J has a λProlog-derivation from hypotheses L.
- Mobility of binders, consider:

$$\frac{\mathbf{\Pi} x\, y.\; x \sim y \;\Rightarrow\; s\langle x\rangle \approx b\langle y\rangle}{\Lambda.s \approx \lambda{*}.b}$$

$$\begin{array}{ll} & \{L \vdash \mathbf{\Pi} x\, y.\; x \sim y \;\Rightarrow\; s\langle x\rangle \approx b\langle y\rangle\} \\ \rightsquigarrow & \nabla x, y.\{L, x \sim y \vdash s\langle x\rangle \approx b\langle y\rangle\} \\ \rightsquigarrow & \{L, n_1 \sim n_2 \vdash s\langle n_1\rangle \approx b\langle n_2\rangle\} \end{array}$$

$$\frac{\{L \vdash A \sim a\} \qquad \{L, n_1 \sim n_2 \vdash s\langle n_1\rangle \approx b\langle n_2\rangle\}}{\{L \vdash s\langle A\rangle \approx b\langle a\rangle\}}\ \textsc{inst}\ \&\ \textsc{cut}$$

Abella – Context Management

- Contexts $L : [\mathbf{o}]$ are lists of arbitrary logical predicate instances.
- The embedding has a backchaining rule:

$$J \in L \;\Rightarrow\; \{L \vdash J\}$$

- We want typing/relational contexts that only contain information about variables, i.e. *nominals*. $\Rightarrow$ inductive $\mathcal{G}$-predicates:

$$\begin{array}{l} \text{Define } C_{\approx} : [\mathbf{o}] \to \textbf{Prop} \text{ by} \\ \quad C_{\approx}(\bullet); \\ \quad \nabla x\, y,\;\; C_{\approx}(L, x \sim y) \;:=\; C_{\approx}(L); \\ \quad \nabla x\, y,\;\; C_{\approx}(L, x \approx y) \;:=\; C_{\approx}(L). \end{array}$$

1. Avoid spurious instances of backchaining.
2. Constrains L to exactly track related variables.
3. Forces L to be injective, functional & range-disjoint.

Abella – Relating Contexts

Left-Totality and Preservation of Type Formation of $\sim$

1. Define a compound inductive predicate C_R:

$$\frac{}{C_R(\bullet \mid \bullet \mid \bullet)} \qquad \frac{C_R(L_F \mid L_\approx \mid L_2) \qquad x, y \text{ fresh for } L_F, L_\approx, L_2}{C_R(L_F, x \ \mathbf{ty} \mid L_\approx, x \sim y \mid L_2, y :_2 *)}$$

$$\frac{\{L_F \vdash A \ \mathbf{ty}\} \qquad \{L_\approx \vdash A \sim a\} \qquad \{L_2 \vdash a :_2 *\} \qquad C_R(L_F \mid L_\approx \mid L_2) \qquad x, y \text{ fresh for } L_F, L_\approx, L_2, A, a}{C_R(L_F, x :_F A \mid L_\approx, x \approx y \mid L_2, y :_2 a)}$$

2. Prove extraction laws that yield connected assumptions:

$$x \ \mathbf{ty} \in L_F \ \Rightarrow \ C_R(L_F \mid L_\approx \mid L_2) \ \Rightarrow \ \ldots$$

3. Prove by induction on $\{L_F \vdash A \ \mathbf{ty}\}$:

$$\{L_F \vdash A \ \mathbf{ty}\} \ \Rightarrow \ \forall L_\approx \, L_2. \ C_R(L_F \mid L_\approx \mid L_2) \ \Rightarrow \ \exists a. \ \{L_\approx \vdash A \sim a\} \ \wedge \ \{L_2 \vdash a :_2 *\}$$

– *Beluga* –

HOAS, 1^{st}-class contexts, context schemas

Beluga – Contextual Objects

SAARLAND UNIVERSITY
COMPUTER SCIENCE

- Objects K (types, terms, derivations) paired with 1^{st}-class context Γ:

$$[\Gamma \vdash K]$$

- No concept of *free variable*:
  - In Coq: $0 \vdash 0_{\text{ty}} \to 0_{\text{ty}}\ \ \textbf{ty} \quad \Rightarrow \quad \bot$ provable.
  - In Abella: $\{\bullet \vdash n_0 \to n_0\ \ \textbf{ty}\} \quad \Rightarrow \quad \bot$ provable.
  - In Beluga $[\bullet \vdash x \to x\ \ \textbf{ty}]$ syntactically ill-formed since $x \notin \bullet$.

Beluga – Representation

- Syntax: standard HOAS.
- Judgements:
  - $\sim$, $\approx$, $\_ :_2 \_$ identical to Abella.
  - $\_$ **ty** does not exist as contextual objects are always well-scoped.
  - $\_ :_F \_$ Abella version with all $\_$ **ty** premises removed.
- *Context Schemas* type dependent lists of dependent records:

$$S_{\lambda W} \;:=\; [x : \mathrm{Tm}_\lambda, x :_2 *] + [x : \mathrm{Tm}_\lambda, x :_2 a, a :_2 *]$$

Beluga – Working with Schemas

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Functionality of $\sim$

1. Define schema:

$$S_\sim \;:=\; [x : \mathsf{Ty}_\mathsf{F}, y : \mathsf{Tm}_\lambda, x \sim y] + [y : \mathsf{Tm}_\lambda]$$

2. Implement, using *pattern matching* and *higher-order unification*:

$$f_{\mathsf{ty}} \;:\; \forall \Gamma : S_\sim.\; [\Gamma \vdash A \sim a] \;\Rightarrow\; [\Gamma \vdash A \sim a'] \;\Rightarrow\; [\Gamma \vdash a = a']$$

Variable case:

- From pattern matching: $x \sim y$ obtained from some $r \in \Gamma$.
- Unification: $x \sim y'$ from some $r' \in \Gamma$.
- Unification: x is local to r, hence $r = r'$, hence $y =_\lambda y'$.

Beluga – Complex Schemas

SAARLAND UNIVERSITY COMPUTER SCIENCE

Left-Totality and Preservation of Type Formation of $\sim$

1. Define schema $S^{\rightarrow}_{\sim W}$ with specific typing information:

$$S^{\rightarrow}_{\sim W} \;:=\; [x : \mathsf{Ty_F}, y : \mathsf{Tm}_\lambda, x \sim y, y :_2 *] + [y : \mathsf{Tm}_\lambda, y :_2 a]$$

2. Implement recursive function $p^{\rightarrow}_{\sim}$ by recursion on $A : [\Gamma \vdash \mathsf{Ty_F}]$, s.t.:

$$p^{\rightarrow}_{\sim} \;:\; \forall \Gamma : S^{\rightarrow}_{\sim W}.\ \forall A : [\Gamma \vdash \mathsf{Ty_F}].\ [\Gamma \vdash \exists a.A \sim a \land a :_2 *]$$

REMARK:

Schemas like $S^{\rightarrow}_{\sim W}$ are probably not automatically inferrable from the involved inductive families, contrary to common belief.

Conclusion

- Summary:
  - Result: reduction of typing for two variants of System F.
  - Formalised using three different approaches: first-order de Bruijn, HOAS with nominals, HOAS with 1^{st}-class contexts

- Formalisation effort (**approximate LOC**):

	mode	**Infrastructure**	**Properties**	**Main Thm.**
Coq	tactics	1200	130	40
Abella	tactics	580	220	30
Beluga	proof terms	100	250	20

- Future Work:
  - STLC, F_ω.
  - Correspondence of reduction?
  - Other techniques: LN [Aydemir et al. '08], HYBRID [Capretta/Felty '06] (both Isabelle and Coq), Twelf, ...

The Take-Home Lesson

- *There is no silver bullet!*
- However, certain techniques go well together:
  - De Bruijn/parallel substitutions/CML-style invariants.
  - HOAS with context constraints/schemas and corresponding inversions.
  - Relations capture correspondences which hold on language fragments.
- Formalising the proof three times was quite instructive.
  - Separate technicalities from inherent complications.

Thank you for your attention.

http://www.ps.uni-saarland.de/extras/fscd17/