Bidirectional Interpolation for the λ-Calculus

Revisiting and Formalising Craig-Čubrić Interpolation

ITP – July 28th 2026

Meven Lennon-Bertrand Alexis Saurin

INRIA – CNRS – IRIF, Université Paris Cité

Interpolation

A cute property from proof theory

If $F \vdash G$ then there exists M such that

- $F \vdash M \vdash G$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(F) \cap \mathrm{voc}(G)$

Interpolation

A cute property from proof theory

If $F \vdash G$ then there exists M such that

- $F \vdash M \vdash G$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(F) \cap \mathrm{voc}(G)$

$$A \wedge B \vdash B \vee (A \to C) \qquad \rightsquigarrow \qquad A \wedge B \vdash B \vdash B \vee (A \to C)$$

A cute property from proof theory

If $F \vdash G$ then there exists M such that

- $F \vdash M \vdash G$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(F) \cap \mathrm{voc}(G)$

$$A \wedge B \vdash B \vee (A \to C) \qquad \rightsquigarrow \qquad A \wedge B \vdash B \vdash B \vee (A \to C)$$

Alexis (a logician): “It would be cool to prove an interpolation theorem for dependent types!”
Me (a type theorist): “Sure, how does an interpolation proof look like?”

Instead of provability $A \vdash B$, consider terms $\Gamma \vdash t : B$

Instead of provability $A \vdash B$, consider terms $\Gamma \vdash t : B$

Simply-typed λ-calculus ($\times, \top, \to, +, \bot$) has the interpolation property (Čubrić, 1994)

If $x\colon A \vdash t : B$ then there exist M, r, l such that:

- $x\colon A \vdash l : M$
- $y\colon M \vdash r : B$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(A) \cap \mathrm{voc}(B)$ (base types appearing in M appear both in A and B)
- $r[l] \rightsquigarrow^* t$

Instead of provability $A \vdash B$, consider terms $\Gamma \vdash t : B$

Simply-typed λ-calculus ($\times, \top, \to, +, \bot$) has the interpolation property (Čubrić, 1994)

If $x \colon A \vdash t : B$ then there exist M, r, l such that:

- $x \colon A \vdash l : M$
- $y \colon M \vdash r : B$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(A) \cap \mathrm{voc}(B)$ (base types appearing in M appear both in A and B)
- $r[l] \rightsquigarrow^* t$

Nice consequences for bi-cartesian closed categories

Instead of provability $A \vdash B$, consider terms $\Gamma \vdash t : B$

Simply-typed λ-calculus $(\times, \top, \to, +, \bot)$ has the interpolation property (Čubrić, 1994)

If $x{:}\, A \vdash t : B$ then there exist M, r, l such that:

- $x{:}\, A \vdash l : M$
- $y{:}\, M \vdash r : B$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(A) \cap \mathrm{voc}(B)$ (base types appearing in M appear both in A and B)
- $r[l] \rightsquigarrow^* t$

Nice consequences for bi-cartesian closed categories

Great!

Instead of provability $A \vdash B$, consider terms $\Gamma \vdash t : B$

Simply-typed λ-calculus ($\times, \top, \to, +, \bot$) has the interpolation property (Čubrić, 1994)

If $x{:}\, A \vdash t : B$ then there exist M, r, l such that:

- $x{:}\, A \vdash l : M$
- $y{:}\, M \vdash r : B$
- $\mathrm{voc}(M) \subseteq \mathrm{voc}(A) \cap \mathrm{voc}(B)$ (base types appearing in M appear both in A and B)
- $r[l] \rightsquigarrow^{*} t$

Nice consequences for bi-cartesian closed categories

Great! Or?

Instead of provability $A \vdash B$ consider…

First we need the following lemma (cf. [Pra65, Cor. 3, p. 54]).

Lemma 3.5. *For a free $\lambda\delta$-calculus (no additional equations but with constants) the following holds: let t^C be a ρ-normal term which is one of: $\pi_i(s)$, $s`r$, $\delta x\cdot u, y\cdot v; s$, $\varepsilon^C(s)$ or an atomic term (i.e. a variable or a constant (or $*$)). Then there exists a chain $t\equiv t_0 \succ \ldots \succ t_n$ of successive subterms of t such that they are in the following relation: for every $0\leqq i<n$, t_i is one of $\pi_j(t_{i+1})$, $t_{i+1}`u$, $\delta x\cdot u, y\cdot v; t_{i+1}$ (for some u, v, x and y) or $t_i\equiv\varepsilon^B(t_{i+1})$, and moreover t_n is an atomic term (i.e. $t_n\equiv v^E$ where v is a variable or a constant (or $*$)).*

In particular if the term t is not $*$ then t_n is not $*$. So, if in addition the $\lambda\delta$-calculus doesn't have constants then t_n is a variable.

Furthermore, by C-normality we have: if $t_{n-1}\equiv d\,x\cdot u, y\cdot v; t_n$ then the above chain $t\equiv t_0 \succ \ldots \succ t_n$ is actually just $t\equiv\delta x\cdot u, y\cdot v; t_1$ and t_1 is either $x^{E_1+E_2}$ or a constant (for some ρ-normal terms u, v).

Similarly, by E-normality, if $t_{n-1}\equiv\varepsilon^B(t_n)$ then $t\equiv\varepsilon^B(t_1)$ and t_1 is a variable x^0 or a constant of type 0.

Simply-… 994)

If $x: A \vdash$ …

- $x: A$ …
- $y: M$ …
- voc(…
- $r[l] \rightsquigarrow$ …

Nice conse…

Great! Or?

Instead of prova…

Simply-… 994)

If $x{:}\,A \vdash$ …

- $x{:}\,A$ …
- $y{:}\,M$ …
- voc(…
- $r[l] \rightsquigarrow$ …

Case 6: (In this case only, we use Lemma 3.5.) $t\equiv\pi_i(s)$ or $t\equiv s`r$ or $t\equiv\delta x\cdot u, y\cdot v; s$ or $t\equiv\varepsilon^C(s)$ where r, s, u and v are some ρ-normal terms. Then by Lemma 3.5 we have that there exists a chain of immediate subterms: $t\equiv t_0\succ t_1\succ\ldots\succ t_i\succ\ldots\succ t_n$ such that the following subcases can take place: $t^{E_i}_{n-1}\equiv\pi_i(x^{E_1\times E_2})$ or $t^{E_1}_{n-1}\equiv x^{E_1^{E_2}}`u^{E_2}$ or $t_{n-1}\equiv\delta x\cdot u, y\cdot v; x^{E_1+E_2}$ or $t_{n-1}\equiv\varepsilon^B(x^0)$ for some ρ-normal terms u and v since in the calculus we don't have additional constants and $t\not\equiv *$.

Subcase 6.1.1: $t^{E_i}_{n-1}\equiv\pi_i(x^{E_1\times E_2})$ and $x^{E_1\times E_2}\in\Gamma_1$.

Let t' be like the term t expect that it has x^{E_i} instead of t_{n-1} (notice however, that t' can contain $x^{E_1\times E_2}$) so $t\equiv_\Gamma t'(\pi_i(x^{E_1\times E_2})/x^{E_i})$. The complexity of t' is lower then the complexity of t, so we can apply the induction hypothesis on $(\Gamma\cup\{x^{E_i}\}\rhd t')$ and the partition $\Gamma, x^{E_i}=(\Gamma_1, x^{E_i})\cup\Gamma_2$. Then by the induction hypothesis then exist $(\Gamma_1, x^{E_i}\rhd R^A)$ and $(\Gamma_2, y^A\rhd S^C)$ such that

$$t'=_{\Gamma,x^{E_i}}S(R/y),\qquad A^+\subseteq(\Gamma_1\cup E_i)^+\cap(\Gamma_2^-\cup C^+),\qquad A^-\subseteq(\Gamma_1\cup E_i)^-\cap(\Gamma_2^+\cup C^-).$$

Recall that $x^{E_1\times E_2}$ is in Γ_1. So we define $r=R(\pi_i(x^{E_1\times E_2}))$, $s=S$ and A stays the same and we can check that the lemma is satisfied. First:

$$t=_\Gamma t'(\pi_i(x^{E_1\times E_2})/x^{E_i})=_\Gamma S((R(\pi_i(x^{E_1\times E_2})/x^{E_i}))/y)\equiv s(r/y).$$

The second and third part of the conclusion are satisfied since $E_i^s\subset(E_1\times E_2)^s$ where $s=+,-$. It is obvious that the stronger hypothesis gives the reduction instead of the equality.

Nice conse…

Great! Or?

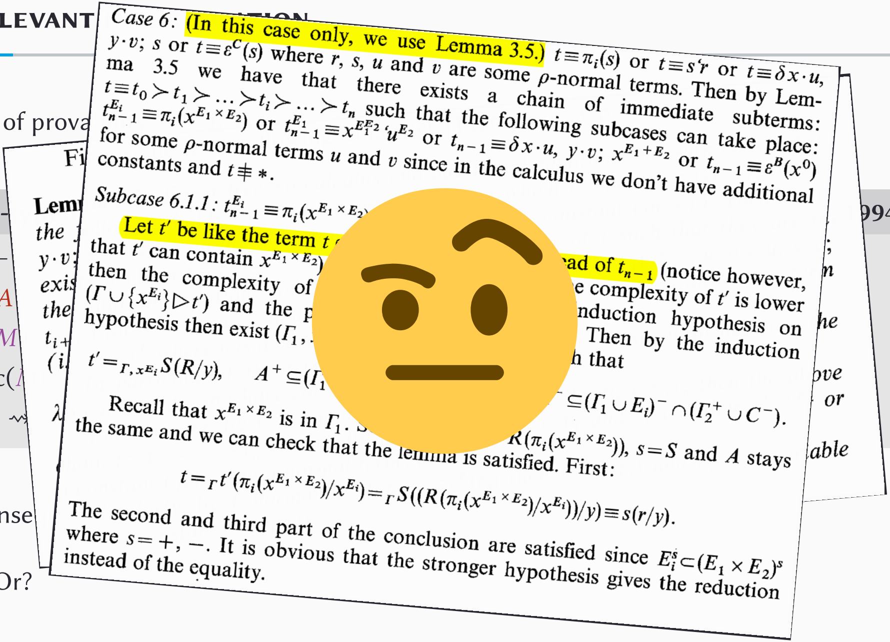
Proof-relevant ...

Instead of prova...

Simply-... 994)

If $x: A \vdash$

- $x: A$
- $y: M$
- voc(
- $r[l] \rightsquigarrow$



Case 6: (In this case only, we use Lemma 3.5.) $t\equiv\pi_i(s)$ or $t\equiv s`r$ or $t\equiv\delta x\cdot u, y\cdot v; s$ or $t\equiv\varepsilon^C(s)$ where r, s, u and v are some ρ-normal terms. Then by Lemma 3.5 we have that there exists a chain of immediate subterms: $t\equiv t_0\succ t_1\succ\ldots\succ t_i\succ\ldots\succ t_n$ such that the following subcases can take place: $t_{n-1}^{E_i}\equiv\pi_i(x^{E_1\times E_2})$ or $t_{n-1}^{E_1}\equiv x^{E_1^{E_2}}`u^{E_2}$ or $t_{n-1}\equiv\delta x\cdot u, y\cdot v; x^{E_1+E_2}$ or $t_{n-1}\equiv\varepsilon^B(x^0)$ for some ρ-normal terms u and v since in the calculus we don't have additional constants and $t\not\equiv *$.

Subcase 6.1.1: $t_{n-1}^{E_i}\equiv\pi_i(x^{E_1\times E_2})$ ...

Let t' be like the term t ... ead of t_{n-1} (notice however, that t' can contain $x^{E_1\times E_2}$) ... e complexity of t' is lower then the complexity of ... nduction hypothesis on $(\Gamma\cup\{x^{E_i}\}\rhd t')$ and the p... Then by the induction hypothesis then exist $(\Gamma_1,$ ... h that

$t'=_{\Gamma,x^{E_i}} S(R/y)$, $A^+\subseteq(\Gamma_1$ ... $\subseteq(\Gamma_1\cup E_i)^-\cap(\Gamma_2^+\cup C^-)$.

Recall that $x^{E_1\times E_2}$ is in Γ_1. ... $R(\pi_i(x^{E_1\times E_2}))$, $s=S$ and A stays the same and we can check that the lemma is satisfied. First:

$$t=_\Gamma t'(\pi_i(x^{E_1\times E_2})/x^{E_i})=_\Gamma S((R(\pi_i(x^{E_1\times E_2})/x^{E_i}))/y)\equiv s(r/y).$$

The second and third part of the conclusion are satisfied since $E_i^s\subset(E_1\times E_2)^s$ where $s=+,-$. It is obvious that the stronger hypothesis gives the reduction instead of the equality.

Nice conse...

Great! Or?

A new interpolation proof

That's just neutral terms!

Remember Lemma 3.5?

That's just neutral terms!

Remember Lemma 3.5?

$$\frac{}{\mathsf{nf}\ *} \qquad \frac{\mathsf{nf}\ t \qquad \mathsf{nf}\ u}{\mathsf{nf}\ (t, u)} \qquad \frac{\mathsf{nf}\ t}{\mathsf{nf}\ \lambda\, x.t} \qquad \frac{\mathsf{ne}\ t}{\mathsf{nf}\ t}$$

$$\frac{}{\mathsf{ne}\ x} \qquad \frac{\mathsf{ne}\ p}{\mathsf{ne}\ p._i} \qquad \frac{\mathsf{ne}\ t \qquad \mathsf{nf}\ u}{\mathsf{ne}\ t\ u}$$

That's just neutral terms!

Remember Lemma 3.5?

$$\frac{}{\mathsf{nf}\ *} \qquad \frac{\mathsf{nf}\ t \qquad \mathsf{nf}\ u}{\mathsf{nf}\ (t, u)} \qquad \frac{\mathsf{nf}\ t}{\mathsf{nf}\ \lambda\, x.t} \qquad \frac{\mathsf{ne}\ t}{\mathsf{nf}\ t}$$

$$\frac{}{\mathsf{ne}\ x} \qquad \frac{\mathsf{ne}\ p}{\mathsf{ne}\ p._i} \qquad \frac{\mathsf{ne}\ t \qquad \mathsf{nf}\ u}{\mathsf{ne}\ t\ u}$$

Why neutrals/normals?

That's just neutral terms!

Remember Lemma 3.5?

$$\frac{}{\mathsf{nf}\ *} \qquad \frac{\mathsf{nf}\ t \qquad \mathsf{nf}\ u}{\mathsf{nf}\ (t, u)} \qquad \frac{\mathsf{nf}\ t}{\mathsf{nf}\ \lambda\, x.t} \qquad \frac{\mathsf{ne}\ t}{\mathsf{nf}\ t}$$

$$\frac{}{\mathsf{ne}\ x} \qquad \frac{\mathsf{ne}\ p}{\mathsf{ne}\ p._i} \qquad \frac{\mathsf{ne}\ t \qquad \mathsf{nf}\ u}{\mathsf{ne}\ t\ u}$$

Why neutrals/normals?
Because they represent equivalence classes for $\equiv_\beta$

That's just neutral terms!

Remember Lemma 3.5?

$$\frac{}{\mathsf{nf}\ *} \qquad \frac{\mathsf{nf}\ t \qquad \mathsf{nf}\ u}{\mathsf{nf}\ (t, u)} \qquad \frac{\mathsf{nf}\ t}{\mathsf{nf}\ \lambda\, x.t} \qquad \frac{\mathsf{ne}\ t}{\mathsf{nf}\ t}$$

$$\frac{}{\mathsf{ne}\ x} \qquad \frac{\mathsf{ne}\ p}{\mathsf{ne}\ p._i} \qquad \frac{\mathsf{ne}\ t \qquad \mathsf{nf}\ u}{\mathsf{ne}\ t\ u}$$

Why neutrals/normals?
~~Because they represent equivalence classes~~
Because they have a good **subformula property**


I don't want invented formulas so I control their vocabulary


I don’t want invented formulas so I control their vocabulary


I don’t want invented types so I can decide typing

I don’t want invented formulas so I control their vocabulary

I don’t want invented types so I can decide typing

subformula property

bidirectional typing

Normal forms check, neutrals infer

$\boxed{\Gamma \vdash t \triangleleft A}$ Normal/checking forms $t, u, \cdots \in \mathsf{Nf}(\Gamma, A)$

$$\frac{}{\Gamma \vdash * \triangleleft \top} \qquad \frac{\Gamma \vdash t \triangleleft A \qquad \Gamma \vdash u \triangleleft B}{\Gamma \vdash (t, u) \triangleleft A \times B} \qquad \frac{\Gamma, x{:}\, A \vdash t \triangleleft B}{\Gamma \vdash \lambda\, x.t \triangleleft A \to B} \qquad \frac{\Gamma \vdash t \triangleright A \qquad A = B}{\Gamma \vdash \underline{t} \triangleleft B}$$

Normal forms check, neutrals infer

$\boxed{\Gamma \vdash t \triangleright A}$ Neutral/infering forms $t, u, \cdots \in \mathsf{Ne}(\Gamma, A)$

$$\frac{(x{:}\, A) \in \Gamma}{\Gamma \vdash x \triangleright A} \qquad \frac{\Gamma \vdash p \triangleright A_1 \times A_2}{\Gamma \vdash p._i \triangleright A_i} \qquad \frac{\Gamma \vdash t \triangleright A \to B \qquad \Gamma \vdash u \triangleleft A}{\Gamma \vdash t\; u \triangleright B}$$

Normalisation

For any term $\Gamma \vdash t : A$, there exists t' such that $\Gamma \vdash t' \triangleleft A$ and $t \rightsquigarrow^* t'$.

Logical relations/reducibility/…, rather standard

The plan

Normalisation

For any term $\Gamma \vdash t : A$, there exists t' such that $\Gamma \vdash t' \triangleleft A$ and $t \rightsquigarrow^* t'$.

Logical relations/reducibility/…, rather standard

Interpolation, generalised

Assume a partitioned context $\Gamma = \Gamma_s \cup \Gamma_t$. Given any $\Gamma \vdash t \triangleleft T$, there is

- a type M and terms $\Gamma_s \vdash l : M$ and $\Gamma_t.(x : M) \vdash r : T$
- such that $r[l] \rightsquigarrow^* t$
- and $\mathrm{voc}(M) \subseteq \mathrm{voc}(\Gamma_s) \cap (\mathrm{voc}(\Gamma_t) \cup \mathrm{voc}(T))$.

Normalisation

For any term $\Gamma \vdash t : A$, there exists t' such that $\Gamma \vdash t' \triangleleft A$ and $t \rightsquigarrow^* t'$.

Logical relations/reducibility/…, rather standard

Interpolation, generalised

Assume a partitioned context $\Gamma = \Gamma_s \cup \Gamma_t$. Given any $\Gamma \vdash t \triangleleft T$, there is

- a type M and terms $\Gamma_s \vdash l : M$ and $\Gamma_t.(x : M) \vdash r : T$
- such that $r[l] \rightsquigarrow^* t$
- and $\mathrm{voc}(M) \subseteq \mathrm{voc}(\Gamma_s) \cap (\mathrm{voc}(\Gamma_t) \cup \mathrm{voc}(T))$.

Mutual induction on inference/checking (*i.e.* neutrals/normals)

Extensions

Positive types

$$\frac{\Gamma \vdash t : \bot}{\Gamma \vdash \mathrm{raise}\, t : T} \qquad \frac{\Gamma \vdash s : A + B \qquad \Gamma.(x\colon A) \vdash b_l : T \qquad \Gamma.(x\colon B) \vdash b_r : T}{\Gamma \vdash \mathrm{if}(s, b_l, b_r) : T}$$

Positive types

> The elimination rules are very bad. What is catastrophic about them is the parasitic presence of a formula T which has no structural link with the formula which is eliminated.
>
> Girard – *Proofs and Types*

$$\frac{\Gamma \vdash t : \bot}{\Gamma \vdash \mathrm{raise}\, t : T} \qquad \frac{\Gamma \vdash s : A + B \qquad \Gamma.(x\colon A) \vdash b_l : T \qquad \Gamma.(x\colon B) \vdash b_r : T}{\Gamma \vdash \mathrm{if}(s, b_l, b_r) : T}$$

"**Cut-free proofs** have a good **subformula property**... but **sum elimination** complicates it."

"**Normal forms** have a good **bidirectional typing**... but **pattern-matching** complicates it."

Positive types

$$\frac{\Gamma \vdash t : \bot}{\Gamma \vdash \mathrm{raise}\, t : T} \qquad \frac{\Gamma \vdash s : A + B \qquad \Gamma.(x{:}\, A) \vdash b_l : T \qquad \Gamma.(x{:}\, B) \vdash b_r : T}{\Gamma \vdash \mathrm{if}(s, b_l, b_r) : T}$$

“**Cut-free proofs** have a good **subformula property**… but **sum elimination** complicates it.”

“**Normal forms** have a good **bidirectional typing**… but **pattern-matching** complicates it.”

Normalisation up to β and **commuting conversion**
Normalization by Evaluation for Call-By-Push-Value and Polarized λ-Calculus (Abel et al. 2019)

Equations & constants

- Relate reduction (β + cc) to equations ($\beta\eta$)
- Arbitrary constants
- Arbitrary equations

$\longrightarrow$ Semantic relationship to biCCC

See the paper for details!

Mechanisation

ROCQ APPROVED

Syntax, substitution	1.5 kLoC (>50% generated)
Typing, reduction, equational theory	1.2 kLoC
Meta-theory (incl. normalisation)	0.7 kLoC
Interpolation	0.9 kLoC
Total	4.5 kLoC

Basic facts

ROCQ APPROVED

Syntax, substitution	1.5 kLoC (>50% generated)
Typing, reduction, equational theory	1.2 kLoC
Meta-theory (incl. normalisation)	0.7 kLoC
Interpolation	0.9 kLoC
Total	4.5 kLoC

Constructive (and explicit): we can **compute interpolants**!

Pen and paper:

- vocabulary inclusion proofs are obvious ⟶ omitted
- typability proofs are obvious ⟶ omitted
- β-reduction proofs are obvious ⟶ omitted
- substitution calculus proofs are extremely obvious ⟶ not even mentioned

Can we do the same in mechanisation?

Automation ❤

Pen and paper:

- vocabulary inclusion proofs are obvious ⟶ **set theory solver**
- typability proofs are obvious ⟶ **proof search**
- β-reduction proofs are obvious ⟶ **setoid rewriting**
- substitution calculus proofs are extremely obvious ⟶ **AutoSubst**

Can we do the same in mechanisation? **Yes!**

Pen and paper:

- vocabulary inclusion proofs are obvious ⟶ **set theory solver**
- typability proofs are obvious ⟶ **proof search**
- β-reduction proofs are obvious ⟶ **setoid rewriting**
- substitution calculus proofs are extremely obvious ⟶ **AutoSubst**

Can we do the same in mechanisation? **Yes!**

- concise proofs (progress + preservation in 30 LoC! normalisation in 500 LoC!)
- concentrate on the important stuff

Set theory solver

- from `stdpp` (`set_solver`)
- great, but slow

Solvers

Set theory solver

- from `stdpp` (`set_solver`)
- great, but slow

Substitution calculus solver

- from AutoSubst (`asimpl`)
- great, but slow
- by default incomplete (η for substitutions)

Set theory solver

- from `stdpp` (`set_solver`)
- great, but slow

Substitution calculus solver

- from AutoSubst (`asimpl`)
- great, but slow
- by default incomplete (η for substitutions)

Typability "solver"

- just a hint database
- naïve but works nicely in STLC

Solvers

Set theory solver

- from stdpp (`set_solver`)
- great, but slow

Substitution calculus solver

- from AutoSubst (`asimpl`)
- great, but slow
- by default incomplete (η for substitutions)

Typability "solver"

- just a hint database
- naïve but works nicely in STLC

Solvers are great! When they are (reasonably) **fast** and **complete**

Set theory solver

- from `stdpp` (`set_solver`)
- great, but slow

Substitution calculus solver

- from AutoSubst (`asimpl`)
- great, but slow
- by default incomplete (η for substitutions)

Typability "solver"

- just a hint database
- naïve but works nicely in STLC

Solvers are great! When they are (reasonably) **fast** and **complete**

Deeper integration? Lemma matching up to the substitution calculus?

Interpolants are non-trivial terms, and they reduce

$$r[l] \rightsquigarrow^* t$$

Typically: deep in a subterm

$$(\lambda\, x.(\lambda\, y.r')\, l', v) \rightsquigarrow^* (\lambda\, x.r'[l'], v) \rightsquigarrow^* (\lambda\, x.t', v) \rightsquigarrow^* \ldots$$

That's a job for setoid rewriting!

Interpolants are non-trivial terms, and they reduce

$$r[l] \rightsquigarrow^* t$$

Typically: deep in a subterm

$$(\lambda\, x.(\lambda\, y.r')\; l', v) \rightsquigarrow^* (\lambda\, x.r'[l'], v) \rightsquigarrow^* (\lambda\, x.t', v) \rightsquigarrow^* \ldots$$

That's a job for setoid rewriting!

We'd like the same for $\Gamma \vdash t \equiv u : A$, but it's not reflexive!

Interpolants are non-trivial terms, and they reduce

$$r[l] \rightsquigarrow^* t$$

Typically: deep in a subterm

$$(\lambda\, x.(\lambda\, y.r')\; l', v) \rightsquigarrow^* (\lambda\, x.r'[l'], v) \rightsquigarrow^* (\lambda\, x.t', v) \rightsquigarrow^* \ldots$$

That's a job for setoid rewriting!

We'd like the same for $\Gamma \vdash t \equiv u : A$, but it's not reflexive!

- hack setoid rewriting $\longrightarrow$ typing tactic to discharge reflexivity goals
- infrastructure that should be part of setoid rewriting? more general version of `Proper`?

The confluence proof

One pen and paper only proof = confluence for β+cc

Easy but very tedious

The confluence proof

One pen and paper only proof = confluence for β+cc

Easy but very tedious

1. Hindley-Rosen: confluence for β + confluence for cc + commutation
2. Confluence for β using parallel reduction
3. Confluence for cc by Newman's lemma
    - termination = smart measure
    - local confluence: 12 critical pairs, checked by CSI^{HO}

The confluence proof

One pen and paper only proof = confluence for β+cc

Easy but **very tedious**

1. Hindley-Rosen: confluence for **β** + confluence for **cc** + **commutation**
2. Confluence for β using **parallel reduction**
3. Confluence for cc by Newman's lemma
   - termination = smart measure
   - local confluence: **12 critical pairs**, checked by CSI$^{\text{HO}}$

One pen and paper only proof = confluence for β+cc

Easy but **very tedious**

1. Hindley-Rosen: confluence for **β** + confluence for **cc** + **commutation**
2. Confluence for β using **parallel reduction**
3. Confluence for cc by Newman's lemma
   - termination = smart measure
   - local confluence: **12 critical pairs**, checked by CSI^{HO}

Can we do better?

Wrapping up

Results

- A new, straightforward proof of relevant interpolation
- Insight on bidirectional typing from proof theory
- A basis for new results?

Results

- A new, straightforward proof of relevant interpolation
- Insight on bidirectional typing from proof theory
- A basis for new results?

Mechanisation

- Automation: very useful, and the toolbox is (mostly) there
- A potential source of inspiration?
- Can we do better? Scale to more complicated type systems?

Results

- A new, straightforward proof of relevant interpolation
- Insight on bidirectional typing from proof theory
- A basis for new results?

Mechanisation

- Automation: very useful, and the toolbox is (mostly) there
- A potential source of inspiration?
- Can we do better? Scale to more complicated type systems?

Thank you!