Lifting proof-relevant unification to higher dimensions

Jesper Cockx
Dominique Devriese

17 January 2017

The rewrite tactic: day one

$$\frac{\begin{array}{l} k : \mathbb{N} \\ n : \mathbb{N} \\ p : P\ k \\ e : k \equiv_{\mathbb{N}} n \end{array}}{? : P\ n}$$

The rewrite tactic: day one

$$
\begin{array}{l}
k : \mathbb{N} \\
n : \mathbb{N} \\
p : P\ k \\
e : k \equiv_{\mathbb{N}} n \\
\hline
? : P\ n
\end{array}
\quad \xrightarrow{\text{rewrite } e} \quad
\begin{array}{l}
k : \mathbb{N} \\
n : \mathbb{N} \\
p : P\ n \\
e : k \equiv_{\mathbb{N}} n \\
\hline
? : P\ n
\end{array}
$$

The unify tactic: day one

$$\frac{\begin{array}{l} k : \mathbb{N} \\ n : \mathbb{N} \\ p : P\ k \\ e : k \equiv_{\mathbb{N}} n \end{array}}{? : P\ n} \quad \xRightarrow{\text{unify } e} \quad \frac{\begin{array}{l} n : \mathbb{N} \\ p : P\ n \end{array}}{? : P\ n}$$

The rewrite tactic: day two

$$\frac{\begin{array}{l} k : \mathbb{N} \\ n : \mathbb{N} \\ p : P\ k \\ e : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n \end{array}}{? : P\ n}$$

The rewrite tactic: day two

$$\frac{\begin{array}{l} k : \mathbb{N} \\ n : \mathbb{N} \\ p : P\ k \\ e : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n \end{array}}{?\ : P\ n} \xRightarrow{\text{rewrite } e} \frac{\begin{array}{l} k : \mathbb{N} \\ n : \mathbb{N} \\ p : P\ k \\ e : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n \end{array}}{?\ : P\ n}$$

The unify tactic: day two

$$
\frac{\begin{array}{l} k : \mathbb{N} \\ n : \mathbb{N} \\ p : P\ k \\ e : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n \end{array}}{?\ :\ P\ n}
\quad \xRightarrow{\text{unify } e} \quad
\frac{\begin{array}{l} n : \mathbb{N} \\ p : P\ n \end{array}}{?\ :\ P\ n}
$$

The rewrite tactic: day three

$$\begin{array}{r@{\;:\;}l} k & \mathbb{N} \\ n & \mathbb{N} \\ xs & \mathtt{Vec}\ A\ (\mathtt{suc}\ k) \\ p & P\ k\ xs \\ e & \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n \\ \hline ? & P\ n\ (\mathtt{subst}\ (\mathtt{Vec}\ A)\ e\ xs) \end{array}$$

The rewrite tactic: day three

$$\begin{array}{r@{\;:\;}l} k & \mathbb{N} \\ n & \mathbb{N} \\ xs & \mathtt{Vec}\ A\ (\mathtt{suc}\ k) \\ p & P\ k\ xs \\ e & \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n \\ \hline ? & P\ n\ (\mathtt{subst}\ (\mathtt{Vec}\ A)\ e\ xs) \end{array} \quad \xRightarrow{\text{rewrite } e} \quad \text{error}$$

The unify tactic: day three

$$\frac{\begin{array}{rl} k & : \mathbb{N} \\ n & : \mathbb{N} \\ xs & : \texttt{Vec}\ A\ (\texttt{suc}\ k) \\ p & : P\ k\ xs \\ e & : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n \end{array}}{?\ : P\ n\ (\texttt{subst}\ (\texttt{Vec}\ A)\ e\ xs)} \xRightarrow{\text{unify } e} \frac{\begin{array}{rl} n & : \mathbb{N} \\ xs & : \texttt{Vec}\ A\ (\texttt{suc}\ n) \\ p & : P\ n\ xs \end{array}}{?\ : P\ n\ xs}$$

Proof-relevant unification

Unification of indexed data

Lifting unifiers to higher dimensions

Proof-relevant unification

Unification of indexed data

Lifting unifiers to higher dimensions

Proof-relevant unification

- Represent unification rules *internally*
- Rules get a *computational interpretation*
- Core of dependent pattern matching

See *Unifiers as Equivalences* (ICFP '16)

Proof-relevant unification: example

$$(k\ n : \mathbb{N})(e : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$

Proof-relevant unification: example

$$(k\ n : \mathbb{N})(e : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$
$$\wr|$$
$$(k\ n : \mathbb{N})(e : k \equiv_{\mathbb{N}} n)$$

Proof-relevant unification: example

$$(k\ n : \mathbb{N})(e : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$

$$\wr|$$

$$(k\ n : \mathbb{N})(e : k \equiv_{\mathbb{N}} n)$$

$$\wr|$$

$$(k : \mathbb{N})$$

Proof-relevant unification: example

$$(k\ n : \mathbb{N})(e : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$

$$\wr|$$

$$(k\ n : \mathbb{N})(e : k \equiv_{\mathbb{N}} n)$$

$$\wr|$$

$$(k : \mathbb{N})$$

Unifiers as equivalences

Goal: given some equations $\bar{u} \equiv_{\Delta} \bar{v}$ with free variables in Γ, find an equivalence f of type

$$\Gamma(\bar{e} : \bar{u} \equiv_{\Delta} \bar{v}) \simeq \Gamma'$$

Unification rules

$$(x : A)(e : x \equiv_A t) \simeq \top \qquad \text{(solution)}$$

$$(\mathtt{suc}\ x \equiv_{\mathbb{N}} \mathtt{suc}\ y) \simeq (x \equiv_{\mathbb{N}} y) \qquad \text{(injectivity)}$$

$$(\mathtt{left}\ x \equiv_{A \uplus B} \mathtt{right}\ y) \simeq \bot \qquad \text{(conflict)}$$

$$(n \equiv_{\mathbb{N}} \mathtt{suc}\ n) \simeq \bot \qquad \text{(cycle)}$$

Telescopic equality

The type of an equation can depend on previous equations:

$$(u : \mathtt{Vec}\ A\ k)(v : \mathtt{Vec}\ A\ n)$$
$$(e_1 : k \equiv_{\mathbb{N}} n)(e_2 : u \equiv_{\mathtt{Vec}\ A\ e_1} v)$$

This allows us to keep track of dependencies between equations.

Proof-relevant unification

Unification of indexed data

Lifting unifiers to higher dimensions

Injectivity for indexed data

Idea: simplify equations between indices together with equation between constructors:

$$(e_1 : i \equiv_I j)(e_2 : \texttt{c}\ u \equiv_{\texttt{D}\ e_1} \texttt{c}\ v)$$

$$\wr|$$

$$(e : u \equiv_A v)$$

Injectivity for indexed data

Idea: simplify equations between indices together with equation between constructors:

$$(e_1 : i \equiv_I j)(e_2 : \mathtt{c}\ u \equiv_{\mathtt{D}\ e_1} \mathtt{c}\ v)$$
$$\wr|$$
$$(e : u \equiv_A v)$$

Indices of $\mathtt{D}$ must be *fully general*: must be distinct equation variables.

Injectivity for indexed data: example

$$\begin{aligned} \mathtt{cons} \quad : \quad & (n : \mathbb{N})(x : A)(xs : \mathtt{Vec}\ A\ n) \\ & \to \mathtt{Vec}\ A\ (\mathtt{suc}\ n) \end{aligned}$$

Injectivity for indexed data: example

$$\begin{aligned}\texttt{cons} \;:\; & (n : \mathbb{N})(x : A)(xs : \texttt{Vec}\ A\ n) \\ & \to \texttt{Vec}\ A\ (\texttt{suc}\ n)\end{aligned}$$

$$(e_1 : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n)$$
$$(e_2 : \texttt{cons}\ k\ x\ xs \equiv_{\texttt{Vec}\ A\ e_1} \texttt{cons}\ n\ y\ ys)$$

Injectivity for indexed data: example

$$\begin{aligned}\mathtt{cons} \;\; : \;\; & (n : \mathbb{N})(x : A)(xs : \mathtt{Vec}\ A\ n) \\ & \to \mathtt{Vec}\ A\ (\mathtt{suc}\ n)\end{aligned}$$

$$\begin{gathered}(e_1 : \mathtt{suc}\ k \equiv_{\mathbb{N}} \mathtt{suc}\ n) \\ (e_2 : \mathtt{cons}\ k\ x\ xs \equiv_{\mathtt{Vec}\ A\ e_1} \mathtt{cons}\ n\ y\ ys) \\ \wr\vert \\ (e_1' : k \equiv_{\mathbb{N}} n)(e_2' : x \equiv_A y) \\ (e_3' : xs \equiv_{\mathtt{Vec}\ A\ e_1} ys)\end{gathered}$$

Injectivity for indexed data: example

$$\texttt{cons} \; : \; (n : \mathbb{N})(x : A)(xs : \texttt{Vec}\ A\ n) \\ \to \texttt{Vec}\ A\ (\texttt{suc}\ n)$$

$$(e_1 : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n)$$
$$(e_2 : \texttt{cons}\ k\ x\ xs \equiv_{\texttt{Vec}\ A\ e_1} \texttt{cons}\ n\ y\ ys)$$
$$\wr|$$
$$(e_1' : k \equiv_{\mathbb{N}} n)(e_2' : x \equiv_A y)$$
$$(e_3' : xs \equiv_{\texttt{Vec}\ A\ e_1} ys)$$

What if the indices are not fully general?

$$(e : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ (\mathtt{suc}\ n)} \mathtt{cons}\ n\ y\ ys)$$

$$\wr|$$

$$???$$

Solution: generalizing the indices

$$(e : \texttt{cons}\ n\ x\ xs \equiv_{\texttt{Vec}\ A\ (\texttt{suc}\ n)} \texttt{cons}\ n\ y\ ys)$$

Solution: generalizing the indices

$$(e : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ (\mathtt{suc}\ n)} \mathtt{cons}\ n\ y\ ys)$$

$$\wr|$$

$$(e_1 : \mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$

$$(e_2 : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ e_1} \mathtt{cons}\ n\ y\ ys)$$

$$(p : e_1 \equiv_{\mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n} \mathtt{refl})$$

Solution: generalizing the indices

$$(e : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ (\mathtt{suc}\ n)} \mathtt{cons}\ n\ y\ ys)$$

$$\wr|$$

$$(e_1 : \mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$

$$(e_2 : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ e_1} \mathtt{cons}\ n\ y\ ys)$$

$$(p : e_1 \equiv_{\mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n} \mathtt{refl})$$

$$\wr|$$

$$(e'_1 : n \equiv_{\mathbb{N}} n)(e'_2 : x \equiv_A y)(e'_3 : xs \equiv_{\mathtt{Vec}\ A\ e'_1} ys)$$

$$(p : \mathtt{cong}\ \mathtt{suc}\ e'_1 \equiv_{\mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n} \mathtt{refl})$$

Solution: generalizing the indices

$$(e : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ (\mathtt{suc}\ n)} \mathtt{cons}\ n\ y\ ys)$$

$$\wr|$$

$$(e_1 : \mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n)$$

$$(e_2 : \mathtt{cons}\ n\ x\ xs \equiv_{\mathtt{Vec}\ A\ e_1} \mathtt{cons}\ n\ y\ ys)$$

$$(p : e_1 \equiv_{\mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n} \mathtt{refl})$$

$$\wr|$$

$$(e_1' : n \equiv_{\mathbb{N}} n)(e_2' : x \equiv_A y)(e_3' : xs \equiv_{\mathtt{Vec}\ A\ e_1'} ys)$$

$$(p : \mathtt{cong}\ \mathtt{suc}\ e_1' \equiv_{\mathtt{suc}\ n \equiv_{\mathbb{N}} \mathtt{suc}\ n} \mathtt{refl})$$

Higher-dimensional unification

$$(e_1' : n \equiv_{\mathbb{N}} n)(e_2' : x \equiv_A y)(e_3' : xs \equiv_{\texttt{Vec}\ A\ e_1'} ys)$$
$$(p : \texttt{cong}\ \texttt{suc}\ e_1' \equiv_{\texttt{suc}\ n \equiv_{\mathbb{N}} \texttt{suc}\ n} \texttt{refl})$$

Now we have to solve equations between equality proofs!

Proof-relevant unification

Unification of indexed data

Lifting unifiers to higher dimensions

How to solve higher-dimensional equations?

Existing unification rules do not apply...

How to solve higher-dimensional equations?

Existing unification rules do not apply...

We solve the problem in three steps:

1. lower the dimension of equations
2. solve lower-dimensional equations
3. lift unifier to higher dimension

Step 1: lower the dimension of equations

We replace all equation variables by regular variables: instead of

$$(e_1 : n \equiv_{\mathbb{N}} n)(e_2 : x \equiv_A y)(e_3 : xs \equiv_{\texttt{Vec}\ A\ e_1} ys)$$
$$(p : \texttt{cong}\ \texttt{suc}\ e_1 \equiv_{\texttt{suc}\ n \equiv_{\mathbb{N}} \texttt{suc}\ n} \texttt{refl})$$

let's first consider

$$(k : \mathbb{N})(u : A)(us : \texttt{Vec}\ A\ k)$$
$$(e : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n)$$

Step 2: solve lower-dimensional equations

This gives us an equivalence f of type

$$\begin{array}{c}(k : \mathbb{N})(u : A)(us : \texttt{Vec}\ A\ k) \\ (e : \texttt{suc}\ k \equiv_{\mathbb{N}} \texttt{suc}\ n) \\ \wr\! \mid \\ (u : A)(us : \texttt{Vec}\ A\ n)\end{array}$$

Step 3: lift unifier to higher dimension

We lift f to an equivalence $f^{\uparrow}$ of type

$$(e_1 : n \equiv_{\mathbb{N}} n)(e_2 : x \equiv_A y)$$
$$(e_3 : xs \equiv_{\texttt{Vec}\ A\ e_1} ys)$$
$$(p : \texttt{cong}\ \texttt{suc}\ e_1 \equiv_{\texttt{suc}\ n \equiv_{\mathbb{N}} \texttt{suc}\ n} \texttt{refl})$$
$$\wr|$$
$$(e_2 : x \equiv_A y)(e_3 : xs \equiv_{\texttt{Vec}\ A\ n} ys)$$

Lifting equivalences: (mostly) general case

Theorem. If we have an equivalence f of type

$$(x : A)(e : b_1\ x \equiv_{B\ x} b_2\ x) \simeq C$$

we can construct $f^{\uparrow}$ of type

$$\begin{gathered}(e : u \equiv_A v)(p : \mathtt{cong}\ b_1\ e \equiv_{r \equiv_{B\ e} s} \mathtt{cong}\ b_2\ e) \\ \wr\| \\ (e' : f\ u\ r \equiv_C f\ v\ s)\end{gathered}$$

Conclusion

Proof-relevant unification is useful to deal with many equality constraints.

Conclusion

Proof-relevant unification is useful to deal with many equality constraints.

To make it work on indexed datatypes, we need to solve *higher-dimensional equations*.

Conclusion

Proof-relevant unification is useful to deal with many equality constraints.

To make it work on indexed datatypes, we need to solve *higher-dimensional equations*.

We can reuse existing unification rules by *lifting* them to higher dimensions.