Generic pattern unification A categorical approach

Ambroise Lafont Neel Krishnaswami

University of Cambridge

December 2022

A quick introduction to unification

$$t \stackrel{?}{=} u$$

terms with metavariables M, N, \dots

Unifier = metavariable substitution σ s.t.

$$t[\sigma] = u[\sigma]$$

Most general unifier = unifier σ that uniquely factors any other

$$\forall \delta$$
, $t[\delta] = u[\delta] \Leftrightarrow \exists ! \delta'. \ \delta = \delta' \circ \sigma$

Goal of unification = find the most general unifier

Where is unification used?

First-order unification

No metavariable argument

Examples

- Logic programming (Prolog)
- ML type inference systems

$$(M \to N) \stackrel{?}{=} (\mathbb{N} \to M)$$

Second-order unification

 $M(\dots)$

Examples

- λ -Prolog
- Type theory, proof assistants

$$(\forall x.M(x,u)) \stackrel{?}{=} t$$

Undecidable

Pattern unification [Miller '91]

A decidable fragment of second-order unification.

Pattern restriction:

$$M(\underbrace{x_1,\ldots,x_n}_{\text{distinct variables}})$$

∃ unification algorithm [Miller '91]

- fails if no unifier
- returns the most general unifier
- linear complexity [Qian '96]

This work

A generic algorithm for pattern unification

- Parameterised by a signature
- Categorical semantics

Examples

- binding signatures
- Linear syntax (e.g., quantum λ -calculus)
- Intrinsic system F

See our preprint.

Related work: algebraic accounts of unification

First-order unification

- Lattice theory [Plotkin '70]
- Category theory
 - [Rydeheard-Burstall '88]
 - [Goguen '89]

Pattern unification

- Category theory
 - [Vezzosi-Abel '14] normalised λ -terms
 - This work

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- Categorical generalisation
 - A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- Categorical generalisation
 - ullet A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- Categorical generalisation
 - ullet A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Syntax (De Bruijn levels)

Metavariable context
$$(M_1:m_1,\dots)$$

$$\overbrace{\Gamma}; \underbrace{n}_{\mathsf{Variable context}} \vdash t$$

$$\frac{1 \le i \le n}{\Gamma; n \vdash v_i} \text{VAR} \qquad \frac{\Gamma; n \vdash t \quad \Gamma; n \vdash u}{\Gamma; n \vdash t \quad u} \text{APP} \qquad \frac{\Gamma; n + 1 \vdash t}{\Gamma; n \vdash \lambda t} \text{ABS}$$

$$\frac{(\textit{M}:\textit{m}) \in \Gamma \qquad 1 \leq \textit{i}_1, \ldots, \textit{i}_m \leq \textit{n} \qquad \textit{i}_1, \ldots \textit{i}_m \text{ distinct}}{\Gamma; \textit{n} \vdash \textit{M}(\textit{v}_{\textit{i}_1}, \ldots, \textit{v}_{\textit{i}_m})} \text{FLEX}$$

No β/η -equation

Metavariable substitution

Substitution σ **from** $(M_1: m_1, \ldots, M_n: m_n)$ **to** Δ :

$$(\sigma_1,\ldots,\sigma_p)$$
 s.t. $\Delta; m_i \vdash \sigma_i$

$$\Delta$$
; $m_i \vdash \sigma_i$

Notation

$$M_i(v_1,\ldots,v_{m_i})\mapsto\sigma_i$$

Term substitution

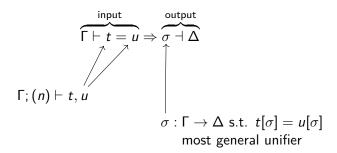
$$\Gamma$$
; $n \vdash t \mapsto \Delta$; $n \vdash t[\sigma]$

Base case:

$$M_i(x_1,\ldots,x_{m_i})\mapsto \sigma_i[v_i\mapsto x_i]$$

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- Categorical generalisation
 - ullet A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Unification algorithm

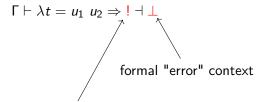


Examples

$$\Gamma, M: 2 \vdash M(x, y) = x \Rightarrow (M(v_1, v_2) \mapsto v_1) \dashv \Gamma$$

$$\Gamma, M: 2 \vdash M(x, y) = y \Rightarrow (M(v_1, v_2) \mapsto v_2) \dashv \Gamma$$

Impossible cases



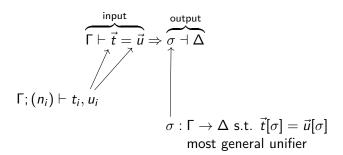
formal "error" substitution

Congruence

$$\frac{\Gamma \vdash t = u \Rightarrow \sigma \dashv \Delta}{\Gamma \vdash \lambda t = \lambda u \Rightarrow \sigma \dashv \Delta}$$

$$\frac{\Gamma \vdash "t_1, t_2 = u_1, u_2" \Rightarrow \sigma \dashv \Delta}{\Gamma \vdash t_1 \ t_2 = u_1 \ u_2 \Rightarrow \sigma \dashv \Delta}$$

Unifying lists of terms



Sequential unification

$$\frac{\Gamma \vdash t_1 = u_1 \Rightarrow \sigma_1 \dashv \Delta_1 \qquad \Delta_1 \vdash \vec{t_2}[\sigma_1] = \vec{u_2}[\sigma_1] \Rightarrow \sigma_2 \dashv \Delta_2}{\Gamma \vdash t_1, \vec{t_2} = u_1, \vec{u_2} \Rightarrow \sigma_1[\sigma_2] \dashv \Delta_2} \text{U-SPLIT}$$

Unifying a metavariable $M(\vec{x}) \stackrel{?}{=} \dots$

Three cases

$$M(\vec{x}) \stackrel{?}{=} M(\vec{y})$$

$$M(\vec{x}) \stackrel{?}{=} \dots M(\vec{y}) \dots$$

$$M(\vec{x}) \stackrel{?}{=} u \text{ and } M \notin u \text{ (non-cyclic)}$$

Unifying a metavariable with itself

$$M(x_1,\ldots,x_m)\stackrel{?}{=} M(y_1,\ldots,y_m)$$

Most general unifier

 $\vec{\mathbf{p}} = \text{vector of common positions: } (x_{\mathbf{p}_1}, \dots, x_{\mathbf{p}_n}) = (y_{\mathbf{p}_1}, \dots, y_{\mathbf{p}_n})$

$$\sigma: M(v_1,\ldots,v_m) \mapsto N(v_{p_1},\ldots v_{p_n})$$

Examples

$$\underbrace{\frac{\vec{p} = (2)}{M(x,y) = M(z,y)}}_{M(x,y) = M(z,x)} \Rightarrow M(v_1, v_2) \mapsto N(v_2)$$

$$\underbrace{M(x,y) = M(z,x)}_{\vec{p} = (1)} \Rightarrow M(v_1, v_2) \mapsto N$$

Deep cyclic case

$$M(\vec{x}) \stackrel{?}{=} \dots M(\vec{y}) \dots$$

No unifier

$$M(\vec{x}) = \dots M(\vec{y}) \dots \Rightarrow ! \dashv \bot$$

sizes cannot match after substitution

Non-cyclic case

$$M(\vec{x}) \stackrel{?}{=} u \ (M \notin u) \tag{1}$$

Most general unifier

(1) as the definition of M:

$$\sigma: M(v_1,\ldots,v_m) \mapsto u[x_i \mapsto v_i]$$

Side condition

$$fv(u) \subset \vec{x}$$

 $M(x) \stackrel{?}{=} y$ has no unifier $(x \neq y)$

Pruning

What about
$$M(x) \stackrel{?}{=} \underbrace{N(x,y)}_{u}$$
?

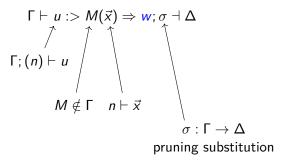
Most general unifier

$$N(v_1, v_2) \mapsto M(v_1)$$

- Side-condition $f_V(u) \subset \vec{x}$ is too pessimistic.
- \bullet Can be enforced by restricting metavariable arities in u.

$$N(x,y) \stackrel{\text{pruning}}{\Longrightarrow} N'(x)$$

Non-cyclic phase



Formal meaning

$$(\sigma, M(v_1, \ldots, v_m) \mapsto w) = \text{most general unifier for } M(\vec{x}) \stackrel{?}{=} u$$

Pruning a variable

$$\frac{y \notin \vec{x}}{\Gamma \vdash y :> M(\vec{x}) \Rightarrow !; ! \dashv \bot} \text{VAR-FAIL}$$

$$\overline{\Gamma \vdash x_n :> M(\vec{x}) \Rightarrow v_n; id_{\Gamma} \dashv \Gamma}$$

Pruning a metavariable

$$M(\vec{x}) \stackrel{?}{=} N(\vec{y}) \qquad (M \neq N)$$

Most general unifier

 $\vec{l}, \vec{r} = \text{vectors of common value positions:}$

$$(x_{l_1},\ldots,x_{l_p})=(y_{r_1},\ldots,y_{r_p})$$

Then,

$$M(v_1,\ldots,v_m)\mapsto P(v_{l_1},\ldots,v_{l_p})$$

 $N(v_1,\ldots,v_n)\mapsto P(v_{r_1},\ldots,v_{r_p})$

Examples

$$M(x,y) = N(z,x)$$
 \Rightarrow $M(v_1, v_2) \mapsto P(v_1)$
 $N(v_1, v_2) \mapsto P(v_2)$
 $M(x,y) = N(z)$ \Rightarrow $M(v_1, v_2), N(v_1) \mapsto P(v_2)$

Pruning operations

Divide & Conquer: a fresh metavariable for each argument.

$$\frac{\Gamma \vdash t :> M'(\vec{x}, \vec{v_{n+1}}) \Rightarrow w; \sigma \dashv \Delta}{\Gamma \vdash \lambda t :> M(\vec{x}) \Rightarrow \lambda w; \sigma \dashv \Delta} \qquad M = \lambda M'$$

$$\frac{\text{"}\Gamma \vdash t, u :> M_1(\vec{x}), M_2(\vec{x}) \Rightarrow w_1, w_2; \sigma \dashv \Delta\text{"}}{\Gamma \vdash t \ u :> M(\vec{x}) \Rightarrow w_1 \ w_2; \sigma \dashv \Delta} \qquad M = M_1 \ M_2$$

Non-cyclic unification of multi-terms

$$\Gamma \vdash u_1, \ldots, u_n :> M_1(\vec{x}_1), \ldots M_n(\vec{x}_n) \Rightarrow w_1, \ldots, w_n; \sigma \dashv \Delta$$

$$\Gamma; (n_i) \vdash u_i \qquad \Delta; m_i \vdash w_i \qquad \sigma : \Gamma \rightarrow \Delta$$

$$(M_i : m_i) \notin \Gamma \qquad \sigma : \Gamma \rightarrow \Delta$$

Formal meaning

$$(\sigma, M_i(v_1, \dots, v_{m_i}) \mapsto w_i) = \mathsf{most}$$
 general unifier for

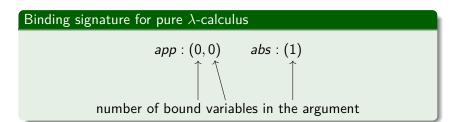
$$M_1(\vec{x}_1), \ldots M_n(\vec{x}_n) \stackrel{?}{=} u_1, \ldots, u_n$$

Sequential non-cyclic unification

$$\frac{\Gamma \vdash t_1 :> M_1(\vec{x}) \Rightarrow u_1; \sigma_1 \dashv \Delta_1 \quad \Delta_1 \vdash \vec{t_2}[\sigma_1] :> \vec{M_2} \Rightarrow \vec{u_2}; \sigma_2 \dashv \Delta_2}{\Gamma \vdash t_1, \vec{t_2} :> M_1(\vec{x}), \vec{M_2} \Rightarrow u_1[\sigma_2], \vec{u_2}; \sigma_1[\sigma_2] \dashv \Delta_2}$$

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- Categorical generalisation
 - ullet A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Parameterisation by a signature



Example: $M(\vec{x}) \stackrel{?}{=} o(\vec{t})$, non-cyclic

$$o:(\alpha_1,\ldots,\alpha_p)$$

bound variables

$$\frac{\Gamma \vdash \vec{t} :> M_1(\vec{x}, \overbrace{v_{n+1}, \dots, v_{n+1+\alpha_1}}), \dots, M_p(\dots) \Rightarrow \vec{u}; \sigma \dashv \Delta}{\Gamma \vdash o(\vec{t}) :> M(\vec{x}) \Rightarrow o(\vec{u}); \sigma \dashv \Delta}$$

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- Generalisation to binding signatures
- 3 Categorical generalisation
 - A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- 3 Categorical generalisation
 - A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Pure λ -calculus as a functor

category of finite cardinals and injections between them



Pure λ -calculus as a functor $\Lambda : \mathbb{F}_m \to \operatorname{Set}$

$$\Lambda_n = \{t \mid \cdot; n \vdash t\}$$

injective renaming

$$M(\vec{x})[\sigma] = \sigma_M [v_i \mapsto x_i]$$

Pure λ -calculus as a fixpoint

$$\Lambda_n \cong \underbrace{\{1,\ldots,n\}}_{\text{variables}} + \underbrace{\Lambda_n \times \Lambda_n}_{\text{application}} + \underbrace{\Lambda_{n+1}}_{\text{abstraction}}$$

In fact,

$$\Lambda = \mu X.F(X)$$

Initial algebra of the endofunctor F on $[\mathbb{F}_m, \operatorname{Set}]$

$$F(X)_n = \{0, \dots, n-1\} + X_n \times X_n + X_{n+1}$$

Pure λ -calculus extended with a metavariable M:m

$$\Lambda_n^{M:m} = \{t \mid M: m; n \vdash t\}$$

As an initial algebra:

$$\Lambda^{M:m} = \mu X. (F(X) + \overbrace{arg^M}^{\text{metavariables}})$$
operations / variables
$$= T. (arg^M)$$

free monad generated by F

Pure λ -calculus extended with a metavariable M: m

$$arg^{M}: \mathbb{F}_{m} \to \operatorname{Set}$$
 $arg^{M}_{n} = \{M\text{-arguments in the variable context } n\}$
 $= \{\operatorname{choice of } m \text{ distinct variables in the context } n\}$
 $= \operatorname{Inj}(m, n)$
 $= \operatorname{hom}_{\mathbb{F}_{m}}(m, n) = ym_{n}$

$$\Lambda^{M:m} = T(ym)$$

Pure λ -calculus with metavariables

Given a metavariable context Γ , define

$$\underline{\Gamma} := \coprod_{(M:m) \in \Gamma} ym$$

$$T(\Gamma)_n = \{t \mid \Gamma; n \vdash t\}$$

Unification as a Kleisli coequaliser

Claims¹:

- hom $(yn, T\Gamma)$ = set of terms in context Γ ; n.
- $hom(\underline{\Gamma}, \underline{T\Delta}) = set$ of metavariable substitutions $\Gamma \to \Delta$.
- Most general unifier of t,u: coequaliser of $yn \xrightarrow{t} T\Gamma$ in $Th^{op}(F) \subset KI(T)$.

Objects: $\underline{\Gamma},\underline{\Delta},\ldots$ (finite coproducts of representable functors) "Lawvere theory"

¹well-known in the first-order case.

Outline

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- 2 Generalisation to binding signatures
- 3 Categorical generalisation
 - A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Signature

- \mathcal{A} small category of contexts (e.g., \mathbb{F}_m , 1)
- **Intuition:** objects = metavariable arities, morphisms = metavariable arguments.
 - ullet All morphisms in ${\cal A}$ are monic (pattern restriction).
 - \mathcal{A} has equalisers and pullbacks $(M(\vec{x}) \stackrel{?}{=} N(\vec{y}))$.
 - $oldsymbol{2}$ F endofunctor on $[\mathcal{A}, \operatorname{Set}]$ of the shape

$$F(X)_a = \coprod_{o \in O_a} X_{L_{o,1}} \times \cdots \times X_{L_{o,n_o}}$$

such that F restricts to an endofunctor on functors preserving finite connected limits.

Typing rules

Notation

$$\Gamma$$
; $b \vdash u$ means $u \in T(\underline{\Gamma})_b$
 $M_1: a_1, \dots, M_n: a_n$ $ya_1 + \dots + ya_n$

$$F(X)_{a} = \coprod_{o \in O_{a}} X_{L_{o,1}} \times \cdots \times X_{L_{o,n_{o}}}$$

$$\frac{\Gamma; L_{o,i} \vdash t_{i}}{\Gamma; a \vdash o(\vec{t})} \text{RIGID} \qquad \frac{x \in \text{hom}_{\mathcal{A}}(a, b)}{\Gamma, M : a \quad ; \quad b \vdash M(x)} \text{FLEX}$$

Semantics of unification

Claim: Given a signature (A, F), a coequaliser diagram in $Th^{op}(F)$ has a colimit as soon as there exists a cocone (i.e., a 'unifier').

Proof: By describing a unification algorithm.

End of this section: soundness proofs for 3 rules.

Interpreting the unification statements

Notations

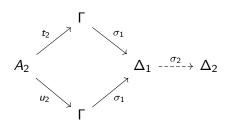
mostly used in $\mathsf{Th}^{op}(F)_{\perp} = \mathsf{Th}^{op}(F) + \mathsf{a}$ free terminal object \perp .

Soundness of U-SPLIT [Rydeheard-Burstall '88]

$$\frac{\Gamma \vdash t_1 = u_1 \Rightarrow \sigma_1 \dashv \Delta_1 \qquad \Delta_1 \vdash t_2[\sigma_1] = u_2[\sigma_1] \Rightarrow \sigma_2 \dashv \Delta_2}{\Gamma \vdash t_1, t_2 = u_1, u_2 \Rightarrow \sigma_1[\sigma_2] \dashv \Delta_2} \text{U-Split}$$

Diagramatically,

$$A_1 \xrightarrow[u_1]{t_1} \Gamma \xrightarrow{-\sigma_1} \Delta_1$$



$$A_1 + A_2 \xrightarrow[H_1, H_2]{t_1, H_2} \Gamma \xrightarrow{\sigma_2 \circ \sigma_1} \Delta_2$$

Soundness of U-FLEXFLEX

$$\frac{b \vdash x =_{\mathcal{A}^{op}} y \Rightarrow z \dashv c}{M : b \vdash M(x) = M(y) \Rightarrow M \mapsto M'(z) \dashv M' : c} \text{U-FLEXFLEX}$$

Diagrammatically,

$$\frac{a \xrightarrow{x} b - \stackrel{z}{\longrightarrow} c \quad \text{in } \mathcal{A}^{op}}{\mathcal{L}a \xrightarrow{\mathcal{L}_{x}} \mathcal{L}b - \stackrel{\mathcal{L}_{z}}{\longrightarrow} \mathcal{L}c \quad \text{in Th}^{op}(F)}$$

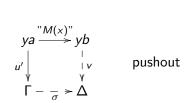
where

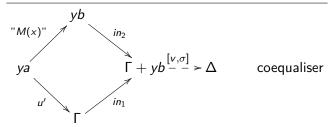
$$a \xrightarrow{x} b \xrightarrow{\mathcal{L} : \mathcal{A}^{op} \to \mathsf{Th}^{op}(F)} ya \xrightarrow{"M(x)"} T(\underline{M : b})$$

Soundness of U-NoCycle

$$\frac{u_{\mid \Gamma} = \underline{u'} \qquad \Gamma \vdash u' :> M(x) \Rightarrow v; \sigma \dashv \Delta}{\Gamma, M : b \vdash M(x) = u \Rightarrow \sigma, M \mapsto v \dashv \Delta} \text{U-NoCycle}$$

Diagrammatically,





Outline

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- ② Generalisation to binding signatures
- Categorical generalisation
 - ullet A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Types

Notation

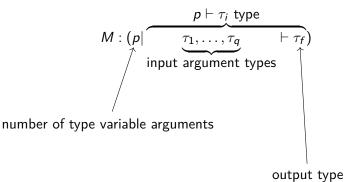
$$n \vdash \tau$$
 type \Leftrightarrow the type τ is wellformed in context n

$$\frac{1 \leq i \leq n}{n \vdash \eta_i \text{ type}} \text{Type-Var} \qquad \frac{n+1 \vdash \tau \text{ type}}{n \vdash \forall \tau \text{ type}} \text{Forall}$$
$$\frac{n \vdash \tau_1, \tau_2 \text{ type}}{n \vdash \tau_1 \to \tau_2 \text{ type}} \text{Arrow}$$

Metavariable arities

Metavariable application

type variables ground variables
$$M(\overbrace{\alpha_1,\ldots,\alpha_p}^{\text{type variables}} | \overbrace{x_1,\ldots,x_q}^{\text{ground variables}})$$



Signature

ullet Objects of $\mathcal{A}=$ metavariables arities

$$n|\vec{\tau} \vdash \sigma_f$$

• Endofunctor F on [A, Set] s.t.

$$\mu F(n|\vec{\tau} \vdash \sigma_f) = \{t \text{ s.t. } n|\vec{\tau} \vdash t : \sigma_f\}$$

The endofunctor for System F

Typing rule	$F(X)_{n\mid \vec{\tau}\vdash \sigma_f}=\coprod \ldots$
$\frac{\tau_i = \sigma_f}{n \vec{\tau} \vdash v_i : \sigma_f} \text{VAR}$	$ ec{ au} _{\sigma_f}$
$\frac{n \vec{\tau}, \sigma_1 \vdash t : \sigma_2}{n \vec{\tau} \vdash \lambda t : \sigma_1 \to \sigma_2} ABS$	$\coprod_{\sigma_1,\sigma_2 \text{ s.t. } \sigma_f = (\sigma_1 \to \sigma_2)} X_{n \vec{\tau},\sigma_1 \vdash \sigma_2}$
$ \frac{n \vec{\tau} \vdash t : \sigma \to \sigma_f n \vec{\tau} \vdash u : \sigma_f}{n \vec{\tau} \vdash t \ u : \sigma_f} \text{APP} $	$\coprod_{\sigma} X_{n \vec{\tau}\vdash \sigma\to\sigma_f} \times X_{n \vec{\tau}\vdash \sigma}$
$\frac{n+1 \vec{\tau}\vdash t:\sigma}{n \vec{\tau}\vdash \Lambda t:\forall \sigma} \text{T-Abs}$	$\coprod_{\sigma \text{ s.t. } \sigma_f = \forall \sigma} X_{n+1 \vec{\tau} \vdash \sigma}$
$\frac{n \vec{\tau} \vdash t : \forall \sigma_1}{n \vec{\tau} \vdash t \cdot \sigma_2 : \sigma_1[\sigma_2]} \text{T-App}$	$\coprod_{\sigma_1,\sigma_2 \text{ s.t. } \sigma_f = \sigma_1[\sigma_2]} X_{n \vec{\tau} \vdash \forall \sigma_1}$

Unification in system F: an example

$$M(\vec{\alpha}|\vec{x}) \stackrel{?}{=} M(\vec{\beta}|\vec{y})$$

Most general unifier

$$M(\eta_1,\ldots,\eta_p|v_1,\ldots v_m)\mapsto N(\vec{\gamma}|\vec{z})$$

where $\vec{\gamma}$ and \vec{z} are vectors of common positions

$$\alpha_{\vec{\gamma}} = \beta_{\vec{\gamma}}$$

$$x_{\vec{z}} = y_{\vec{z}}$$

Summary of the generic unification algorithm

$$\overline{\Gamma \vdash () = ()} \Rightarrow id_{\Gamma} \dashv \overline{\Gamma} \qquad \overline{\bot \vdash \vec{t} = \vec{u} \Rightarrow ! \dashv \bot}$$

$$\underline{\Gamma \vdash t_{1} = u_{1} \Rightarrow \sigma_{1} \dashv \Delta_{1}} \qquad \Delta_{1} \vdash \vec{t_{2}}[\sigma_{1}] = u_{2}^{2}[\sigma_{1}] \Rightarrow \sigma_{2} \dashv \Delta_{2}} \text{U-SPLIT}$$

$$\overline{\Gamma \vdash t_{1}, \vec{t_{2}} = u_{1}, \vec{u_{2}} \Rightarrow \sigma_{1}[\sigma_{2}] \dashv \Delta_{2}}$$

$$\underline{\Gamma \vdash \vec{t} = \vec{u} \Rightarrow \sigma \dashv \Delta} \text{U-RIGRIG} \qquad \frac{o \neq o'}{\Gamma \vdash o(\vec{t}) = o'(\vec{u}) \Rightarrow ! \dashv \bot}$$

$$\underline{u_{|\Gamma} = \underline{u'}} \qquad \Gamma \vdash u' :> M(x) \Rightarrow v; \sigma \dashv \Delta}_{\Gamma, M : b \vdash M(x) = u \Rightarrow \sigma, M \mapsto v \dashv \Delta} \text{U-NoCYCLE} \qquad + \text{sym}$$

$$\underline{b \vdash x =_{\mathcal{A}^{op}} y \Rightarrow z \dashv c}$$

$$\overline{\Gamma, M : b \vdash M(x) = M(y) \Rightarrow M \mapsto M'(z) \dashv \Gamma, M' : c} \text{U-FLEXFLEX}$$

$$\underline{u = o(\vec{t})} \qquad \underline{u_{|\Gamma} \neq \dots}_{\Gamma, M : b \vdash M(x) = u \Rightarrow ! \dashv \bot} \text{U-CYCLIC} \qquad + \text{sym}$$

Non cyclic phase

$$\frac{\Gamma \vdash () :> () \Rightarrow (); id_{\Gamma} \dashv \Gamma}{\bot \vdash t :> \vec{t} \Rightarrow !; ! \dashv \bot}$$

$$\frac{\Gamma \vdash t_{1} :> M_{1}(\vec{x}) \Rightarrow u_{1}; \sigma_{1} \dashv \Delta_{1} \quad \Delta_{1} \vdash t_{2}[\sigma_{1}] :> \vec{M_{2}} \Rightarrow \vec{u_{2}}; \sigma_{2} \dashv \Delta_{2}}{\Gamma \vdash t_{1}, \vec{t_{2}} :> M_{1}(\vec{x}), \vec{M_{2}} \Rightarrow u_{1}[\sigma_{2}], \vec{u_{2}}; \sigma_{1}[\sigma_{2}] \dashv \Delta_{2}}$$

$$\frac{c \vdash_{\mathcal{A}^{op}} y :> x \Rightarrow l; r \dashv d}{\Gamma, M : c \vdash M(y) :> N(x) \Rightarrow M'(l); M \mapsto M'(r) \dashv \Gamma, M' : d} P\text{-FLEX}$$

$$\frac{\Gamma \vdash \vec{t} :> \mathcal{L}^{+} x^{o} \Rightarrow \vec{u}; \sigma \dashv \Delta \qquad o = x \cdot o'}{\Gamma \vdash o(\vec{t}) :> N(x) \Rightarrow o'(\vec{u}); \sigma \dashv \Delta}$$

$$\frac{o \neq x \cdot \dots}{\Gamma \vdash o(\vec{t}) :> N(x) \Rightarrow !; ! \dashv \bot}$$

Pruning $o(\vec{t})$: variable case

o is a variable \Rightarrow $\vec{t} = ()$.

$$\frac{\Gamma \vdash \overrightarrow{t} :> \dots \qquad o = \overrightarrow{x}_{o'}}{\Gamma \vdash o(\overrightarrow{t}) :> N(\overrightarrow{x}) \Rightarrow o'(\overrightarrow{u}); \sigma \dashv \Delta} \Leftrightarrow \frac{o = x_i}{\Gamma \vdash o :> N(\overrightarrow{x}) \Rightarrow i; id_{\Gamma} \dashv \Gamma}$$

$$\frac{o \neq x \cdot \dots}{\Gamma \vdash o(\overrightarrow{t}) :> N(x) \Rightarrow !; ! \dashv \bot} \Leftrightarrow \frac{o \notin \overrightarrow{x}}{\Gamma \vdash o :> N(\overrightarrow{x}) \Rightarrow !; ! \dashv \bot}$$

Pruning $o(\vec{t})$: operation case

Assume $o:(\alpha_1,\ldots,\alpha_p)$

$$\frac{\Gamma \vdash \vec{t} :> \overbrace{\mathcal{L}^{+} x^{o}}^{M_{1}(\vec{x}, n, \dots, n + \alpha_{1} - 1), \dots, M_{p}(\dots)} \Rightarrow \vec{u}; \sigma \dashv \Delta}{\Gamma \vdash o(\vec{t}) :> N(x) \Rightarrow o'(\vec{u}); \sigma \dashv \Delta}$$

$$\frac{o \neq x \cdot \dots}{\Gamma \vdash o(\vec{t}) :> N(x) \Rightarrow !; ! \dashv \bot}$$
never applies

Typing rule for metavariables

Typing judgement

$$\underbrace{\Gamma}_{\text{Type variable context}}; \qquad \underbrace{n}_{\text{Types of variables}} \\ |\underbrace{t_1, \dots, t_m}_{n \vdash t_i \text{ type}} \vdash u : t_f$$

Typing metavariables

$$\frac{0 < \overbrace{\alpha_1, \dots, \alpha_p}^{\text{distinct}} \leq n \quad 0 < \overbrace{x_1, \dots x_q}^{\text{distinct}} \leq m \quad \tau_i[\vec{\alpha}] = t_{x_i} }{\Gamma, M : (p | \tau_1, \dots, \tau_q \vdash \tau_f) \; ; \; n \mid t_1, \dots, t_m \vdash M(\underbrace{\vec{\alpha}}_{\text{type variables}} | \vec{x}) : \tau_f[\vec{\alpha}] }$$

Future directions

- Implementation
 - logic programming
 - verified in Coq/Agda (needs rephrasing using structural recursion)
- Unification modulo reduction
- Efficient pattern unification [Qian '96]
- Anti-unification