Generic pattern unification

A categorical approach

Ambroise Lafont Neel Krishnaswami

University of Cambridge

October 2022

What is unification?

$$\underbrace{t}_{} \overset{?}{=} \underbrace{u}_{}$$

terms with metavariables $M, N, \ldots$

Unifier $=$ metavariable substitution σ s.t.

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

Most general unifier $=$ unifier σ that uniquely factors any other

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

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

$M(\dots)$

Example

- Type theory, proof assistants

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

Indecidable

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

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

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

Outline

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

Outline

Syntax (De Bruijn levels)

$$\overbrace{\Gamma}^{\text{Metavariable context } (M_1 : n_1, \dots)} ; \underbrace{n}_{\text{Variable context}} \vdash t$$

$$\frac{x < n}{\Gamma; n \vdash x}\textsc{Var} \qquad \frac{\Gamma; n \vdash t \quad \Gamma; n \vdash u}{\Gamma; n \vdash t\ u}\textsc{App} \qquad \frac{\Gamma; n+1 \vdash t}{\Gamma; n \vdash \lambda t}\textsc{Abs}$$

$$\frac{(M : n) \in \Gamma \qquad x_1, \dots, x_n < n \qquad x_1, \dots x_n \text{ distinct}}{\Gamma; n \vdash M(x_1, \dots, x_n)}\textsc{Flex}$$

Metavariable substitution

Substitution σ **from** $\overbrace{(M_1 : m_1, \ldots, M_p : m_p)}^{\Gamma}$ **to** Δ:

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

σ extends to terms:

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

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

Unification algorithm

$$\overbrace{\Gamma \vdash t = u}^{\text{input}} \Rightarrow \overbrace{\sigma \dashv \Delta}^{\text{output}}$$

$\Gamma; (n) \vdash t, u$

$\sigma : \Gamma \to \Delta$ s.t. $t[\sigma] = u[\sigma]$
most general unifier

Examples

$$\Gamma, M : 2 \vdash M(5,3) = 5 \Rightarrow (M \mapsto 0) \dashv \Gamma$$

$$\Gamma, M : 2 \vdash M(5,3) = 3 \Rightarrow (M \mapsto 1) \dashv \Gamma$$

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

$$\frac{\Gamma \vdash \text{''}t_1, t_2 = u_{1,} u_2\text{''} \Rightarrow \sigma \dashv \Delta}{\Gamma \vdash t_1\ t_2 = u_1\ u_2 \Rightarrow \sigma \dashv \Delta}$$

Unifying lists of terms

$$\overbrace{\Gamma \vdash \vec{t} = \vec{u}}^{\text{input}} \Rightarrow \overbrace{\sigma \dashv \Delta}^{\text{output}}$$

$\Gamma; (n_i) \vdash t_i, u_i$

$\sigma : \Gamma \to \Delta$ s.t. $\vec{t}[\sigma] = \vec{u}[\sigma]$
most general unifier

Examples (lists)

$$\Gamma \vdash () = () \Rightarrow id_\Gamma \dashv \Gamma$$

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

Impossible cases

$$\Gamma \vdash \lambda t = u_1 \ u_2 \Rightarrow ! \dashv \bot$$

formal "error" context

formal "error" substitution

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

Three cases

1. $M(\vec{x}) \stackrel{?}{=} M(\vec{y})$
2. $M(\vec{x}) \stackrel{?}{=} \ldots M(\vec{y}) \ldots$
3. $M(\vec{x}) \stackrel{?}{=} u$ and $M \notin u$

Unifying a metavariable with itself

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

Most general unifier: $M \mapsto M'(\vec{z})$

- $\vec{z}$ = vector of common positions: $x_{\vec{z}} = y_{\vec{z}}$

Formally,

$$\frac{"n \vdash \vec{x} = \vec{y} \Rightarrow \vec{z} \dashv p"}{\Gamma, M : n \vdash M(\vec{x}) = M(\vec{y}) \Rightarrow M \mapsto M'(\vec{z}) \dashv \Gamma, M' : p}$$

Cyclic case

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

No unifier

$$\Gamma \vdash \underbrace{M(\vec{x}) = \ldots M(\vec{y}) \ldots}_{\text{sizes cannot match after substitution}} \Rightarrow\ ! \dashv \bot$$

Non cyclic case

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

Most general unifier: $M \mapsto u[\vec{x}^{-1}]$

- Requires

$$fv(u) \subset \vec{x} \tag{1}$$

$\Rightarrow$ **Pruning phase**: enforces (1) by restricting metavariable arities.

Example

$$\begin{array}{rcl} M(x) & \stackrel{?}{=} & N(x,y) \\ N(x,y) & \xrightarrow{\text{pruning}} & N'(x) \end{array}$$

Pruning phase

$$\Gamma \vdash u :> \underset{\substack{\uparrow \\ M \notin \Gamma, u}}{M}(\vec{x}) \Rightarrow \overbrace{v}^{u \text{ after pruning and renamed by } \vec{x}^{-1}} ; \underset{\substack{\uparrow \\ \sigma : \Gamma \to \Delta \\ \text{pruning substitution}}}{\sigma} \dashv \Delta$$

Intuition

$$u \stackrel{?}{=} M(\vec{x}) \quad \Rightarrow \quad (\sigma, M \mapsto v) = \text{most general unifier}$$

Pruning a metavariable

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

Most general unifier: $M \mapsto N'(\vec{l}),\ N \mapsto N'(\vec{r})$ such that

$$x_{\vec{l}} = y_{\vec{r}}$$

$$\frac{\text{"}n \vdash \vec{x} :> \vec{y} \Rightarrow \vec{l}; \vec{r} \dashv p\text{"}}{\Gamma, N : n \vdash N(\vec{x}) :> M(\vec{y}) \Rightarrow N'(\vec{l}); N \mapsto N'(\vec{r}) \dashv \Gamma, N' : p}$$

Pruning: other examples

$$\frac{}{\Gamma \vdash x_i :> M(x_0, \ldots, x_n) \Rightarrow i; id_\Gamma \dashv \Gamma} \qquad \frac{y \notin \vec{x}}{\Gamma \vdash y :> M(\vec{x}) \Rightarrow !; ! \dashv \bot}$$

$$\frac{\Gamma \vdash t :> M_1(\vec{x}, \overbrace{n}^{\text{bound variable}}) \Rightarrow v; \sigma \dashv \Delta}{\Gamma \vdash \lambda t :> M(\vec{x}) \Rightarrow \lambda v; \sigma \dashv \Delta}$$

$$\frac{\text{"}\Gamma \vdash t, u :> M_1(\vec{x}), M_2(\vec{x}) \Rightarrow v_1, v_2; \sigma \dashv \Delta\text{"}}{\Gamma \vdash t\ u :> M(\vec{x}) \Rightarrow v_1\ v_2; \sigma \dashv \Delta}$$

Pruning multi-terms

$$\Gamma \vdash u_1, \ldots, u_n :> M_1(\vec{x}_1), \ldots M_n(\vec{x}_n) \Rightarrow v_1, \ldots, v_n ; \sigma \dashv \Delta$$

$\Gamma ; (n_i) \vdash u_i$ $\qquad\qquad$ $\Delta ; {\color{blue} m_i} \vdash v_i$

$(M_i : {\color{blue} m_i}) \notin \Gamma, u_j$ $\qquad\qquad$ $\sigma : \Gamma \to \Delta$

Rules for multi-terms

$$\frac{}{\Gamma \vdash () :> () \Rightarrow (); id_\Gamma \dashv \Gamma}$$

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

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

Parameterisation by a *signature*

Binding signature for pure λ-calculus

$$\mathit{app} : (0, 0) \qquad \mathit{abs} : (1)$$

number of bound variables in the argument

Example: pruning an operation

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

$$\frac{\Gamma \vdash \vec{t} :> M_1(\vec{x}, \overbrace{n, \dots, n + \alpha_1 - 1}^{\text{bound variables}}), \dots, M_p(\dots) \Rightarrow \vec{u}; \sigma \dashv \Delta}{\Gamma \vdash o(\vec{t}) :> N(\vec{x}) \Rightarrow o(\vec{u}); \sigma \dashv \Delta}$$

Semantics

$$\begin{array}{rcrl} \Gamma \vdash t = u \Rightarrow \sigma \dashv \Delta & \Leftrightarrow & \sigma & = \textbf{coequaliser} \text{ of } t \text{ and } u \\ \Gamma \vdash t :> f \Rightarrow v; \sigma \dashv \Delta & \Leftrightarrow & \sigma, v & = \textbf{pushout} \text{ of } t \text{ and } f \end{array}$$

... in the category of metavariables contexts and substitutions between them (cf next section).

Outline

Outline

Pure λ-calculus as a functor

category of finite cardinals and injections between them

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

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

Pure λ-calculus as a fixpoint

$$\Lambda_n \cong \underbrace{\{0,\ldots,n-1\}}_{\text{variables}} + \underbrace{\Lambda_n \times \Lambda_n}_{\text{application}} + \underbrace{\Lambda_{n+1}}_{\text{abstraction}}$$

In fact,

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

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

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

Pure λ-calculus extended with one metavariable

$$\Lambda(M : m)_n = \{t \mid M : m; n \vdash t\}$$

As a fixpoint:

$$\Lambda(M : m) = \mu X.(\underbrace{F(X)}_{\text{operations / variables}} + \mathit{arg}^M)$$

$$\begin{aligned} \mathit{arg}^M{}_{\boldsymbol{n}} &= \{M\text{-arguments in the variable context } \boldsymbol{n}\} \\ &= \{\vec{x} \in \{0, \dots, n-1\}^m \ | x_1, \dots, x_m \text{ distinct}\} \\ &= \hom_{\mathbb{F}_m}(m, n) \end{aligned}$$

$$\Lambda(M : m) = \mu X.(F(X) + ym)$$

Pure λ-calculus with metavariables

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

As a fixpoint:

$$\Lambda(\Gamma) = \mu X.(F(X) + \underbrace{\coprod_{(M:m)\in\Gamma} ym}_{\underline{\Gamma}})$$

$$= \underbrace{T}_{\text{free monad generated by } F}(\underline{\Gamma})$$

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

Unification as a Kleisli coequaliser

Claims:

- $\hom(yn, T\underline{\Gamma})$ = set of terms in context $\Gamma; n$.
- $\hom(\underline{\Gamma}, T\underline{\Delta})$ = set of metavariable substitutions $\Gamma \to \Delta$.
- Most general unifier of t, u: coequaliser of $yn \underset{u}{\overset{t}{\rightrightarrows}} T\underline{\Gamma}$
  in $\mathrm{Th}_F \subset Kl_T$.

Objects: $\underline{\Gamma}$,$\underline{\Delta}$,...
(finite coproducts of representable functors)

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

1. $\mathcal{A}$ small category s.t.
   - all morphisms are monic (pattern restriction)
   - $\mathcal{A}$ has finite connected limits ($M(\vec{x}) \stackrel{?}{=} N(\vec{y})$)

Intuition: objects = metavariable arities

Example

$\mathbb{F}_m$

1. F endofunctor on $[\mathcal{A}, \mathrm{Set}]$ 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.

Semantics of unification

Claim: Given a signature $(\mathcal{A}, F)$, a coequaliser diagram in Th_F has a coequaliser as soon as there exists a cocone (i.e., a 'unifier')

Proof: By proving termination of the unification algorithm.

Interpreting the unification statements

Notations

$$\begin{array}{rcrl} \Gamma \vdash t = u \Rightarrow \sigma \dashv \Delta & \Leftrightarrow & \sigma & = \textbf{coequaliser} \text{ of } t \text{ and } u \\ \Gamma \vdash t :> f \Rightarrow v; \sigma \dashv \Delta & \Leftrightarrow & \sigma, v & = \textbf{pushout} \text{ of } t \text{ and } f \end{array}$$

mostly used in $\mathrm{Th}^*_F = \mathrm{Th}_F$ + a free terminal object $\perp$.

Next three slides: soundness proofs for 2 rules.

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

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

Diagramatically,

$$\begin{tikzcd} A_1 \arrow[r, shift left, "t_1"] \arrow[r, shift right, "u_1"'] & \Gamma \arrow[r, dashed, "\sigma_1"] & \Delta_1 \end{tikzcd}
\qquad
\begin{tikzcd} & \Gamma \arrow[dr, "\sigma_1"] & & \\ A_2 \arrow[ur, "t_2"] \arrow[dr, "u_2"'] & & \Delta_1 \arrow[r, dashed, "\sigma_2"] & \Delta_2 \\ & \Gamma \arrow[ur, "\sigma_1"'] & & \end{tikzcd}$$

$$\begin{tikzcd} A_1 + A_2 \arrow[r, shift left, "{t_1, t_2}"] \arrow[r, shift right, "{u_1, u_2}"'] & \Gamma \arrow[r, dashed, "\sigma_2 \circ \sigma_1"] & \Delta_2 \end{tikzcd}$$

Semantics of U-FlexFlex

$$\boxed{\dfrac{b \vdash x =_{\mathcal{A}^{op}} y \Rightarrow z \dashv c}{\Gamma, M : b \vdash M(x) = M(y) \Rightarrow M \mapsto M'(z) \dashv \Gamma, M' : c}\textsc{U-FlexFlex}}$$

Diagrammatically,

$$\dfrac{a \overset{x}{\underset{y}{\rightrightarrows}} b \overset{z}{\dashrightarrow} c}{\begin{tikzcd} & \mathcal{L}b \arrow[dr, "in_2"] & & \\ \mathcal{L}a \arrow[ur, "\mathcal{L}x"] \arrow[dr, "\mathcal{L}y"'] & & \Gamma + \mathcal{L}b \arrow[r, dashed, "\Gamma + \mathcal{L}z"] & \Gamma + \mathcal{L}c \\ & \mathcal{L}b \arrow[ur, "in_2"'] & & \end{tikzcd}}$$

where

$$a \xrightarrow{x} b \quad \xmapsto{\mathcal{L} : \mathcal{A}^{op} \to \mathrm{Th}_F} \quad ya \xrightarrow{"M(x)"} T(M : b)$$

Soundness of U-FlexFlex

$$\dfrac{\dfrac{a \overset{x}{\underset{y}{\rightrightarrows}} b \overset{z}{\dashrightarrow} c}{\mathcal{L}a \overset{\mathcal{L}x}{\underset{\mathcal{L}y}{\rightrightarrows}} \mathcal{L}b \overset{\mathcal{L}z}{\dashrightarrow} \mathcal{L}c}\ (\mathcal{L} \text{ preserves coequalisers})}{\begin{tikzcd} & \mathcal{L}b \arrow[dr, "in_2"] & & \\ \mathcal{L}a \arrow[ur, "\mathcal{L}x"] \arrow[dr, "\mathcal{L}y"'] & & \Gamma + \mathcal{L}b \arrow[r, dashed, "\Gamma+\mathcal{L}z"] & \Gamma + \mathcal{L}c \\ & \mathcal{L}b \arrow[ur, "in_2"'] & & \end{tikzcd}}$$

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

Types

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

$$\frac{x < n}{n \vdash \tau \text{ type}}\textsc{Type-Var} \qquad \frac{n+1 \vdash \tau \text{ type}}{n \vdash \forall\tau \text{ type}}\textsc{Forall}$$

$$\frac{n \vdash \tau_1, \tau_2 \text{ type}}{n \vdash \tau_1 \to \tau_2 \text{ type}}\textsc{Arrow}$$

Metavariable arities

Metavariable application

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

$$M : (p \mid \overbrace{\underbrace{\tau_1, \ldots, \tau_q}_{\text{input argument types}} \vdash \tau_f}^{p \vdash \tau_i \text{ type}})$$

number of type variable arguments

output type

Signature

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

$$n|\tau_1, \ldots, \tau_p \vdash \tau_f$$

- Need an endofunctor F on $[\mathcal{A}, \mathrm{Set}]$ s.t.

$$\mu F(n|\tau_1, \ldots, \tau_p \vdash \tau_f) = \{t \text{ s.t. } \mathsf{n}|\tau_1, \ldots, \tau_p \vdash t : \tau_f\}$$

The endofunctor for System F

Typing rule	$F(X)_{n\|\Gamma\vdash\tau} = \coprod \dots$
$\dfrac{x : \tau \in \Gamma}{n\|\Gamma \vdash x : \tau}\textsc{Var}$	$\|\Gamma\|_\tau$
$\dfrac{n\|\Gamma, x : \tau_1 \vdash t : \tau_2}{n\|\Gamma \vdash \lambda x.t : \tau_1 \to \tau_2}\textsc{Abs}$	$\coprod_{\tau_1,\tau_2 \text{ s.t. } \tau = (\tau_1 \to \tau_2)} X_{n\|\Gamma,x:\tau_1\vdash\tau_2}$
$\dfrac{n\|\Gamma \vdash t : \tau' \to \tau \quad n\|\Gamma \vdash u : \tau'}{n\|\Gamma \vdash t\ u : \tau}\textsc{App}$	$\coprod_{\tau'} X_{n\|\Gamma\vdash\tau'\to\tau} \times X_{n\|\Gamma\vdash\tau'}$
$\dfrac{n+1\|wk(\Gamma) \vdash t : \tau'}{n\|\Gamma \vdash \Lambda t : \forall\tau'}\textsc{T-Abs}$	$\coprod_{\tau' \text{ s.t. } \tau=\forall\tau'} X_{n+1\|wk(\Gamma)\vdash\tau'}$
$\dfrac{n\|\Gamma \vdash t : \forall\tau_1}{n\|\Gamma \vdash t \cdot \tau_2 : \tau_1[\tau_2]}\textsc{T-App}$	$\coprod_{\tau_1,\tau_2 \text{ s.t. } \tau=\tau_1[\tau_2]} X_{n\|\Gamma\vdash\forall\tau_1}$

Typing rule for metavariables

$$\frac{\alpha_1, \ldots, \alpha_p \text{ distinct}, < n \qquad x_1, \ldots x_q \text{ distinct}, < |\Gamma| \qquad \tau_i[\vec{\alpha}] = \Gamma_{x_i}}{\underbrace{\Delta, M : (p|\tau_1, \ldots, \tau_q \vdash \tau_f)}_{\text{Metavariable context}}; n|\Gamma \vdash M(\underbrace{\vec{\alpha}}_{\text{type variables}}, \vec{x}) : \tau_f[\vec{\alpha}]}$$

Unification in system F: an example

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

Most general unifier: $M \mapsto N(\vec{\gamma}, \vec{z})$, where

- $\vec{\gamma}$ maximal s.t.

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

- $\vec{z}$ maximal s.t.

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

Summary of the generic unification algorithm

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

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

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

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

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

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

Pruning phase

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

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

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

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

$$\frac{c \vdash_{\mathscr{D}} y :> x \Rightarrow y'; x' \dashv d}{\Gamma, M : c \vdash M(y) :> N(x) \Rightarrow M'(y'); M \mapsto M'(x') \dashv \Gamma, M' : d}\text{P-FLEX}$$