Generic pattern unification A categorical approach

Ambroise Lafont Neel Krishnaswami

University of Cambridge

October 2022

What is 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

 $M(\dots)$

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

- 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
 - A case study: syntax of pure λ -calculus
 - Generic pattern unification
- 4 Example: System F

Syntax (De Bruijn levels)

Metavariable context
$$(M_1:n_1,...)$$

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

$$\frac{x < n}{\Gamma; \, n \vdash x} \mathrm{VAR} \qquad \frac{\Gamma; \, n \vdash t \quad \Gamma; \, n \vdash u}{\Gamma; \, n \vdash t \quad u} \mathrm{APP} \qquad \frac{\Gamma; \, n + 1 \vdash t}{\Gamma; \, n \vdash \lambda t} \mathrm{ABS}$$

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

Metavariable substitution

Substitution
$$\sigma$$
 from $(M_1: m_1, \ldots, M_p: m_p)$ to Δ :

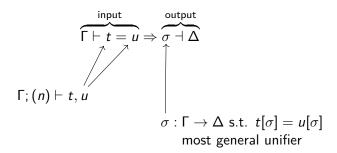
$$(\sigma_1,\ldots,\sigma_p)$$
 s.t. $\Delta; m_i \vdash \sigma_i$

 σ extends to terms:

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

- 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

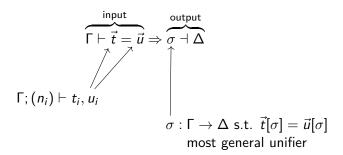
$$\Gamma, M: 2 \vdash M(5,3) = 5 \Rightarrow (M \mapsto 0) \dashv \Gamma$$

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

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

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

Unifying lists of terms

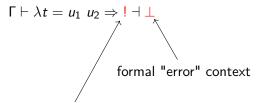


Examples (lists)

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

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

Impossible cases



formal "error" substitution

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

Three cases

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

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

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

Unifying a metavariable with itself

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

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

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

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

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

No unifier

$$\Gamma \vdash \underbrace{M(\vec{x})} = \underbrace{\ldots M(\vec{y}) \ldots} \Rightarrow ! \dashv \bot$$

sizes cannot match after substitution

Non cyclic case

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

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

Requires

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

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

Example

$$M(x)$$
 $\stackrel{?}{=}$ $N(x,y)$
 $N(x,y)$ $\xrightarrow{\text{pruning}}$ $N'(x)$

Pruning phase

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

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

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

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

Pruning: other examples

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

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

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

Pruning multi-terms

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

$$\Gamma; (n_i) \vdash u_i \qquad \Delta; m_i \vdash v_i$$

$$(M_i : m_i) \notin \Gamma, u_i \qquad \sigma : \Gamma \to \Delta$$

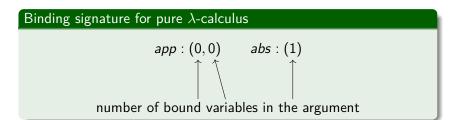
Rules for multi-terms

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

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

- 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: pruning an operation

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

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

Semantics

$$\Gamma \vdash t = u \Rightarrow \sigma \vdash \Delta \qquad \Leftrightarrow \qquad \sigma = \text{coequaliser of } t \text{ and } u$$

$$\Gamma \vdash t :> f \Rightarrow v; \sigma \vdash \Delta \qquad \Leftrightarrow \qquad \sigma, v = \text{pushout of } t \text{ and } f$$

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

- 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

- 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
- Example: System F

Pure λ -calculus as a functor

category of finite cardinals and injections between them

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

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

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 $[\mathbb{F}_m, \operatorname{Set}]$

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

Pure λ -calculus extended with one metavariable

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

As a fixpoint:

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

$$arg^{M}_{n} = \{M\text{-arguments in the variable context } n\}$$

$$= \{\vec{x} \in \{0, \dots, n-1\}^{m} \mid x_{1}, \dots, x_{m} \text{ distinct}\}$$

$$= \text{hom}_{\mathbb{F}_{m}}(m, n)$$

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

Pure λ -calculus with metavariables

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

As a fixpoint:

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

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

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

Unification as a Kleisli coequaliser

Claims:

- hom $(yn, T\Gamma)$ = 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 $yn \xrightarrow{t} T\underline{\Gamma}$ in $Th_F \subset KI_T$.

Objects: $\underline{\Gamma},\underline{\Delta},\ldots$

(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

- $oldsymbol{0}$ \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 [A, 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 (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

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

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

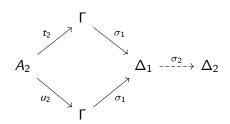
Next three slides: soundness proofs for 2 rules.

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,

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



$$A_1 + A_2 \xrightarrow[H_1, H_2]{t_1, t_2} \Gamma \xrightarrow{\sigma_2 \circ \sigma_1} \Delta_2$$

Semantics of U-FLEXFLEX

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

Diagrammatically,

$$a \xrightarrow{x} b - \stackrel{z}{-} > c$$

$$Lb$$

$$La$$

$$\Gamma + Lb \xrightarrow{\Gamma + Lz} \Gamma + Lc$$

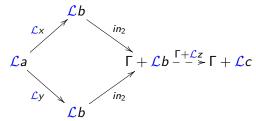
$$in_2$$

where

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

Soundness of U-FLEXFLEX

$$\frac{a \xrightarrow{x} b - \frac{z}{> c}}{\mathcal{L}a \xrightarrow{\mathcal{L}x} \mathcal{L}b - \frac{\mathcal{L}z}{> \mathcal{L}c}} \quad (\mathcal{L} \text{ preserves coequalisers})$$



Outline

- 1 Pattern unification for pure λ -calculus
 - Syntax
 - Unification algorithm
- Generalisation to binding signatures
- Categorical generalisation
 - ullet 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

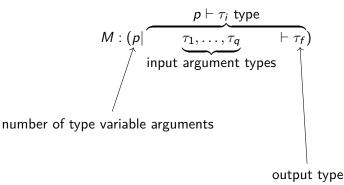
$$\frac{x < n}{n \vdash \tau \text{ type}} \text{Type-Var} \qquad \frac{n + 1 \vdash \tau \text{ type}}{n \vdash \forall \tau \text{ type}} \text{Forall}$$

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

Metavariable arities

Metavariable application

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



Signature

• Objects of A = metavariables arities

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

• Need an endofunctor F on [A, Set] s.t.

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

The endofunctor for System F

Typing rule	$F(X)_{n\mid\Gamma\vdash\tau}=\coprod\ldots$
$\frac{x:\tau\in\Gamma}{n \Gamma\vdash x:\tau}\mathrm{VAR}$	$ \Gamma _{ au}$
$\frac{n \Gamma, x : \tau_1 \vdash t : \tau_2}{n \Gamma \vdash \lambda x . t : \tau_1 \to \tau_2} ABS$	$\coprod_{\tau_1,\tau_2 \text{ s.t. } \tau = (\tau_1 \to \tau_2)} X_{n \Gamma,x:\tau_1 \vdash \tau_2}$
$\frac{n \Gamma \vdash t : \tau' \to \tau n \Gamma \vdash u : \tau'}{n \Gamma \vdash t \ u : \tau} APP$	$\coprod_{\tau'} X_{n \Gamma \vdash \tau' \to \tau} \times X_{n \Gamma \vdash \tau'}$
$\frac{n+1 wk(\Gamma)\vdash t:\tau'}{n \Gamma\vdash \Lambda t:\forall \tau'}\text{T-Abs}$	$\coprod_{\tau' \text{ s.t. } \tau = \forall \tau'} X_{n+1 wk(\Gamma) \vdash \tau'}$
$\frac{n \Gamma \vdash t : \forall \tau_1}{n \Gamma \vdash t \cdot \tau_2 : \tau_1[\tau_2]} \text{T-App}$	$\coprod_{\tau_1,\tau_2 \text{ s.t. } \tau=\tau_1[\tau_2]} X_{n \Gamma \vdash \forall \tau_1}$

Typing rule for metavariables

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

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

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

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

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

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

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

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

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

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

Pruning phase

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

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

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

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

 $c \vdash_{\mathscr{D}} v :> x \Rightarrow v' : x' \dashv d$

 $\overline{\Gamma, M : c \vdash M(y) :> N(x) \Rightarrow M'(y'); M \mapsto M'(x') \dashv \Gamma, M' : d} \operatorname{P-FLEX}_{4 \boxtimes b \land 4 \boxtimes 4 \boxtimes b \land 4$