A Complete Transformation System for Polymorphic Higher-Order Unification

Ullrich Hustadt

MPI–I–91–228 December 1991

Author's Address

Ullrich Hustadt
Max-Planck-Institut für Informatik
Im Stadtwald
W-6600 Saarbrücken
Germany
E-mail: Ullrich.Hustadt@mpi-sb.mpg.de

Abstract

Polymorphic higher-order unification is a method for unifying terms in the polymorphically typed λ-calculus, that is, given a set of pairs of terms $S_0 = \{s_1 \stackrel{?}{=} t_2, \ldots, s_n \stackrel{?}{=} t_n\}$, called a unification problem, finding a substitution σ such that $\sigma(s_i)$ and $\sigma(t_i)$ are equivalent under the conversion rules of the calculus for all i, $1 \leq i \leq n$.

I present the method as a transformation system, i.e. as a set of schematic rules $U \Longrightarrow U'$ such that any unification problem $\delta(U)$ can be transformed into $\delta(U')$ where δ is an instantiation of the meta-level variables in U and U'. By successive use of transformation rules one possibly obtains a solved unification problem with obvious unifier. I show that the transformation system is correct and complete, i.e. if $\delta(U) \Longrightarrow \delta(U')$ is an instance of a transformation rule, then the set of all unifiers of $\delta(U')$ is a subset of the set of all unifiers of $\delta(U)$ and if $\mathcal{U}$ is the set of all unification problems that can be obtained from successive applications of transformation rules from an unification problem U, then the union of the set of all unifiers of all unification problems in $\mathcal{U}$ is the set of all unifiers of U.

The transformation rules presented here are essentially different from those in Snyder and Gallier (1989) or Nipkow (1990). The correctness and completeness proofs are in lines with those of Snyder and Gallier (1989).

Keywords

Higher-order unification, Polymorphic Lambda-Calculus

1 Introduction

Since the beginning of the 1970s a variety of higher-order theorem provers and higher-order programming languages have been invented. The languages used in these systems are based on the Simple Theory of Types defined in Church (1932). This theory is based on three sets of rules: The term formation rules define how to build expressions describing the application of a function to an object and the abstraction of a function from an expression describing an object. These rules are governed by the classification of objects into types. Types are either basic types or function types ($A \to B$) where A and B are again types. The conversion rules define equivalence classes on terms. Turning the conversion rules into reduction rules gives an description of application evaluation. Deduction rules describe how formulas, which are simply terms of a designated type, can be proved.

In theorem proving most implementations lift the resolution principle well-known for first-order theorem proving to the higher-order case. For example Gordon (1985) and Paulson and Nipkow (1990) respectively describe the theorem provers HOL and Isabelle. Higher-order programming languages augment some PROLOG-like language with the term formation and reduction rules of the Simple Theory of Types. See for example Miller and Nadathur (1986) and Pfenning (1989) for descriptions of the programming languages λProlog and ELF, respectively.

The core of these systems is an implementation of unification of terms in the Simple Theory of Types, higher-order unification for short. All these implementations are based on the algorithmic description of the correct and complete higher-order unification algorithm given in Huet (1975). In Snyder and Gallier (1989) this algorithmic description is reformulated using the method of transformations on system of terms, which is based on Herbrand's thesis and was adapted to first-order unification in Martelli and Montanari (1982). A unification problem is a set of pairs $S_0 = \{s_1 \stackrel{?}{=} t_1, \ldots, s_n \stackrel{?}{=} t_n\}$ of terms to be unified. By repeated application of various transformation rules one possibly obtains a solved system S_n whose unifier is obvious. The correctness of the transformation rules ensures that every unifier of a solved system S_n is also a unifier of S_0. A complete set of transformation rules allows to enumerate a complete set of unifiers by examing all possible transformation sequences.

Soon the limitations of the simple type theory for the formulation of axiom system and programs became apparent. Various extensions of the simple type theory were suggested. One of the most prominent extensions is the polymorphic λ-calculus. Beside the basic types it allows the usage of type variables in type formation, so it is possible to describe a family of types as instances of a single type.

To deal with these extensions the higher-order unification algorithm has to be modified. But none of the current implementations provides a complete algorithm. Nipkow (1990) presents a correct but incomplete transformation system for polymorphic higher-order unification. His transformation system is only able to enumerate a complete set of unifiers for sure if none of these unifiers instantiates a type variable with a function type. Dougherty (1991) presents a complete algorithm for unification in the polymorphic λ-calculus. In his approach the problem of higher-order unification is translated to the problem of unification with respect to extensional equality in combinatory logic. Solving a unification problem in the λ-calculus involves translating it into a system of combinatory terms, computing unifiers using his transformation rules, and translating back the unifiers into unifiers in the λ-calculus. Although this approach allows for a very elegant presentation of the problem and its solution, it basically uses narrowing and some additional rules to transform the systems of combinatory terms and inherits all weaknesses of narrowing.

We will present a correct and complete transformation system $\mathcal{HPT}$ for polymorphic higher-order unification. The transformation rules are essentially different from the rules in Snyder and Gallier (1989) or Nipkow (1990). The correctness and completeness proofs are in line with those of Snyder and Gallier (1989). We define a verification system $\mathcal{CPT}$, show that for every unifier for a given unification problem one can find a terminating sequence of transformations resulting in a solved system whose associated solution is this unifier. Then we show that every transformation in $\mathcal{CPT}$ corresponds to a sequence of transformations in $\mathcal{HPT}$. This enables us to enumerate a complete set of unifiers by systematically applying all possible transformations in $\mathcal{HPT}$ to the unification problem.

2 Polymorphic Terms and Substitutions

Definition 2.1 (Types)
Let $\mathcal{T}_0$ be a set of *base types* and $\mathcal{V}_\mathcal{T}$ a set of *type variables*. The set of *types* $\mathcal{T}$ is inductively defined as the smallest set containing $\mathcal{T}_0$ and $\mathcal{V}_\mathcal{T}$ such that if $S, T \in \mathcal{T}$ then $(S \to T) \in \mathcal{T}$. An element of either $\mathcal{T}_0$ or $\mathcal{V}_\mathcal{T}$ is called a *atomic type*. △

Definition 2.2 (Terms)
The set $\mathcal{RL}^{Poly}$ of *raw terms* is defined by the following abstract syntax

$$\mathcal{RL}^{Poly} = \mathcal{V} \mid (\Sigma{:}\,\mathcal{T}) \mid (\mathcal{V}{:}\,\mathcal{T}) \mid (\mathcal{RL}^{Poly} \cdot \mathcal{RL}^{Poly}) \mid \lambda\,\mathcal{V}{:}\,\mathcal{T}.\,\mathcal{RL}^{Poly}$$

where $\mathcal{V}$ is a set of term variables and Σ a set of constants. We suppose $\mathcal{V}$, Σ, and $\mathcal{V}_\mathcal{T}$ to be pairwise disjoint. To define the set $\mathcal{L}^{Poly} \subseteq \mathcal{RL}^{Poly}$ of well-typed terms, we use the following inference rules:

Bound variable: $$\frac{x{:}T \in \Gamma}{\Gamma \vdash x{:}T}, \quad \text{if } x \in \mathcal{V} \wedge T \in \mathcal{T}$$

Free variable: $$\frac{}{\Gamma \vdash (F{:}T){:}T}, \quad \text{if } F \in \mathcal{V} \wedge T \in \mathcal{T}$$

Constant: $$\frac{}{\Gamma \vdash (c{:}T){:}T}, \quad \text{if } c{:}T \in C \wedge T \in \mathcal{T}$$

Application: $$\frac{\Gamma \vdash M{:}(S \to T) \quad \Gamma \vdash N{:}S}{\Gamma \vdash (M \cdot N){:}T}$$

Abstraction: $$\frac{\Gamma \oplus x{:}S \vdash M{:}T}{\Gamma \lambda\, x{:}\,S.\, N{:}(S \to T)}$$

The set $\mathcal{L}^{Poly}$ is the set of all $M \in \mathcal{RL}^{Poly}$ such that $\epsilon \vdash M{:}T$ for some $T \in \mathcal{T}$. The set of all free variables is $\mathcal{FV}$. The type variables have no special meaning for type inference. They act like nullary type constructors. △

Definition 2.3 (type **and** range**)**
We define the function range : $\mathcal{T} \to \mathcal{T}$ as follows:

$$\begin{array}{lcll} \text{range}(T) & = & T & \text{if } T \in \mathcal{T}_0 \\ \text{range}(T) & = & T & \text{if } T \in \mathcal{V}_\mathcal{T} \\ \text{range}((S \to T)) & = & \text{range}(T) & \text{otherwise} \end{array}$$

The functions type : $\mathcal{L}^{Poly} \to \mathcal{T}$ and range : $\mathcal{L}^{Poly} \to \mathcal{T}$ are defined by

$$\begin{array}{lcll} \text{type}(M) & = & T & \text{iff } \epsilon \vdash M{:}T \\ \text{range}(M) & = & T & \text{iff } \epsilon \vdash M{:}S \text{ and } \text{range}(S) = T \end{array}$$

△

α-, β- and η-conversion on terms is defined in the usual way. We ignore α-conversion by working with α-equivalence classes of terms. The strong normalization property holds for β-reduction on polymorphically typed terms, so any term M has a unique β-normal form denoted $M{\downarrow}$. With any term M in β-normal form we can associate a unique η-expanded form $\eta[M]$. The set of all terms in η-expanded form is $\mathcal{L}_\eta^{Poly}$. Any term M in β-normal form can be represented in general form $\lambda\, x_1{:}\,S_1.\ldots.\lambda\, x_n{:}\,S_n.(((a \cdot P_1) \cdot \cdots) \cdot P_n)$ which is also denoted $\lambda\, x_1{:}\,S_1 \ldots x_n{:}\,S_n.\, a(P_1, \ldots, P_n)$ or $\lambda\, \overline{x_n{:}\,S_n}.\, a(\overline{P_n})$. We call $\overline{x_n{:}\,S_n}$ the *binder* of M, a the *head*, which is also denoted head(M), and $a(\overline{P_n})$ the *matrix* of M. If a is a constant or bound variable, e is called *rigid*, otherwise it is called *flexible*.

Definition 2.4 (Bound and free variables)
A bound variable x is called *bound in a term* M_1 if M_1 contains a subterm of the form $\lambda x{:}T.\, M_2$. The subterm M_2 is the *scope* of this binding occurrence of $x{:}T$. The set of all bound variables of a term M_1 is denoted $BV(M_1)$.

A free variable $(F{:}T) \in \mathcal{FV}$ is a *free variable of* M_1 or is said to occur *free in* M_1 if it is a subterm of M_1. The set of free variables of a term M_1 is denoted $FV^{\text{Term}}(M_1)$.

The set of all type variables occurring in some type decoration in a term M_1 is denoted $FV^{\text{Type}}(M_1)$. The set of all free term and type variables of a term M_1 is denoted $FV(M_1)$. △

Definition 2.5 (Size of types and terms)
The *term size of a raw term* M, denoted $|M|^{\text{Term}}$, is the number of atomic subterms of M, defined by:

$$\begin{array}{lcl} |x|^{\text{Term}} & = & 1 \\ |F{:}\,S|^{\text{Term}} & = & 1 \\ |c{:}\,S|^{\text{Term}} & = & 1 \\ |(M \cdot N)|^{\text{Term}} & = & |M|^{\text{Term}} + |N|^{\text{Term}} \\ |\lambda\, x{:}\,S.\, M|^{\text{Term}} & = & |M|^{\text{Term}} \end{array}$$

The *type size of a type* T, denoted $|T|^{\text{Type}}$, is the number of atomic types of T, defined by

$$\begin{array}{lcl} |A|^{\text{Type}} & = & 1 \\ |s|^{\text{Type}} & = & 1 \\ |(S \to T)|^{\text{Type}} & = & |S|^{\text{Type}} + |T|^{\text{Type}} \end{array}$$

The *type size of a raw term* M, denoted $|M|^{\text{Type}}$, is the number of atomic types occurring in M, defined by:

$$\begin{array}{lcl} |x|^{\text{Type}} & = & 0 \\ |F{:}\,T|^{\text{Type}} & = & |T|^{\text{Type}} \\ |c{:}\,T|^{\text{Type}} & = & |T|^{\text{Type}} \\ |(M \cdot N)|^{\text{Type}} & = & |M|^{\text{Type}} + |N|^{\text{Type}} \\ |\lambda\, x{:}\,S.\, M|^{\text{Type}} & = & |S|^{\text{Type}} + |M|^{\text{Type}} \end{array}$$

△

Definition 2.6 (Substitutions)
A *polymorphic substitution* is a pair $\sigma = \langle \sigma_1, \sigma_2 \rangle$ consisting of a mapping σ_1 from type variables to types and a mapping σ_2 from free variables to terms defined in the usual way. $\widehat{\sigma_1}$ denotes the unique extension of σ_1 to a mapping from types to types and $\widehat{\sigma_2}$ denotes the unique extension of σ_2 to a mapping from λ-terms to λ-terms. We usually write σ or $\langle \sigma_1, \sigma_2 \rangle$ instead of $\langle \widehat{\sigma_1}, \widehat{\sigma_2} \rangle$. We write $\sigma_1(M)$ for the result of applying the type substitution σ_1 to all type variables occurring in the term M.

The identity substition ι is the pair consisting of the identity substitution ι_1 on types and the identity substitution ι_2 on terms.

The *union* of two polymorphic substitutions $\sigma = \langle \sigma_1, \sigma_2 \rangle$ and $\tau = \langle \tau_1, \tau_2 \rangle$ is the substitution $\sigma \cup \tau = \langle \sigma_1 \cup \tau_1, \sigma_2 \cup \tau_2 \rangle$ where the union of the type substitutions and the union of the term substitutions are defined in the usual way. For a set Z of type and free variables, we say that two substitutions σ and τ are *equal over* Z *with respect to some congruence relation* $=_E$ *on terms*, denoted $\sigma =_E \tau\,[Z]$, iff for all type variables $A \in Z$, $\sigma_1(A) = \tau_1(A)$, and for all free variables $(F{:}T) \in Z$, $\sigma_2(F{:}T) =_E \tau_2(F{:}T)$.

We say that σ is *more general than* τ *over* Z, denoted $\sigma \leq_E \tau\,[Z]$, iff there exists a substitution θ such that $\theta \circ \sigma =_E \tau\,[Z]$.

The *domain* of a substitution σ is

$$\mathcal{DOM}(\sigma) = \{A \in \mathcal{V}_{\mathcal{T}} \mid \sigma_1(A) \neq A\} \cup \{F \in \mathcal{FV} \mid \sigma_2(F) \neq F\}.$$

The set of variables *introduced by* σ is

$$\mathcal{I}(\sigma) = \bigcup_{F \in \mathcal{DOM}(\sigma)} FV(\sigma(F)).$$

A substitution σ is *normalized* if $\sigma(F) = \sigma(F)\downarrow_\beta$ for all $F \in \mathcal{DOM}(\sigma) \cap \mathcal{FV}$.

A substitution σ is a *renaming substitution away from* Z if $\sigma(F)\downarrow_\eta \in \mathcal{FV} \cup \mathcal{V}_\mathcal{T}$ for all $F \in \mathcal{DOM}(\sigma)$, $\mathcal{I}(\sigma) \cap Z = \emptyset$ and σ is injective. If σ is a renaming substitution and τ is some arbitrary substitution then $\sigma \circ \tau$ is called a *variant* of τ.

The *restriction* of a substitution σ to a set of variables Z, denoted $\sigma_{|Z}$, is the substitution

$$\sigma_{|Z}(F) = \begin{cases} \sigma(F), & \text{if } F \in Z \\ F, & \text{otherwise.} \end{cases}$$

△

3 Polymorphic Unification Problems

Definition 3.1 (Unification problems in $\mathcal{L}^{Poly}$)
An *equation in* $\mathcal{L}^{Poly}$ is a multiset of terms M and N in $\mathcal{L}_\eta^{Poly}$ such that all free variables in $\{M, N\}$ are uniquely annotated. For equations we will use the notation $M \stackrel{?}{=} N$. A *system* S *in* $\mathcal{L}^{Poly}$ is a multiset of equations in $\mathcal{L}^{Poly}$ such that that all free variables in the set of all terms in S are uniquely annotated. A *unification problem in* $\mathcal{L}^{Poly}$ is an ordered pair $\langle \sigma, S \rangle$ with σ a type substitution and S a system such that $\mathcal{DOM}(\sigma) \cap FV^{\text{Type}}(S) = \emptyset$. △

For a system

$$S = \{M_1 \stackrel{?}{=} N_1, \ldots, M_n \stackrel{?}{=} N_n\}$$

we write $S\downarrow$ instead of $\{\eta[M_1\downarrow] \stackrel{?}{=} \eta[N_1\downarrow], \ldots, \eta[M_n\downarrow] \stackrel{?}{=} \eta[N_n\downarrow]\}$. $S\downarrow$ is unique up to renaming of bound variables.

There are at least two views of the meaning of σ in a unification problem $\langle \sigma, S \rangle$.

1. The type substitution σ is a representation of some equations on types that have to be solved like any equation on terms in S. That means any type variable has to occur only once in $\langle \sigma, S \rangle$. So if $A \in \mathcal{DOM}(\sigma)$ and $A \in FV^{\text{Type}}(S)$ we have to eliminate A in S by applying $\sigma_{|\{A\}}$ to S.

2. The type substitution σ memorizes the type instantiation that has to be done to get a unifier for S. So typically we start with $\langle \iota_1, S \rangle$ and transform it into $\langle \sigma_1, S' \rangle$. The solved system S' represents a term substitution σ_2 and σ_1 a type substiution. Then $\langle \sigma_1, \sigma_2 \rangle$ is a unifier of S.

In this paper I choose the second view. Then the restriction $\mathcal{DOM}(\sigma) \cap FV^{\text{Type}}(S) = \emptyset$ on a pair $\langle \sigma, S \rangle$ ensures that we can memorize type instantiations on S. Now I will give definitions for the notions of unifier and solved system that I already used in this explanation.

Definition 3.2 (Unifier in $\mathcal{L}^{Poly}$)
A normalized substitution θ is called a *unifier in* $\mathcal{L}^{Poly}$ of two terms M and N from $\mathcal{L}_\eta^{Poly}$ iff $\theta(M) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \theta(N)$ holds. θ is a unifier of a system S in $\mathcal{L}^{Poly}$ iff it is a unifier for every equation in S and it is called a unifier of a unification problem $\langle \sigma, S \rangle$ iff it is a unifier of the system S and an instance of $\langle \sigma, \iota_2 \rangle$.

The set of all unifiers of a unification problem U is denoted $\text{SU}(U)$. △

Definition 3.3 (Complete set of unifiers)
Let U be a unification problem and Z a finite set of variables, called the set of *protected variables*. A set $\text{CSU}(U)[Z]$ of normalized substitutions is a *complete set of unifiers for* U *separated on* $FV(U)$ *away from* Z iff

(1) $\mathrm{CSU}(U)[Z] \subseteq \mathrm{SU}(U)$

(2) $\forall\, \phi \in \mathrm{SU}_E(\Gamma)\colon \exists\, \theta \in \mathrm{CSU}(U)[Z]\colon \theta \leq_\beta \phi\,[FV(U)]$

(3) $\forall\, \theta \in \mathrm{CSU}(U)[Z]\colon \mathcal{DOM}(\theta) \subseteq FV(U)$ and $\mathcal{I}(\theta) \cap (Z \cup \mathcal{DOM}(\theta)) = \emptyset$.

If Z is not significant we drop the notation $[Z]$. If $\mathrm{CSU}(U)$ consists of a single substitution we call this substitution a *most general unifier*. △

For any set of equations between types there exists at most one unifier up to variable renaming. If this unifier exists for a set of equations E, it will be denoted $\mathrm{mgu}(E)$.

Definition 3.4 (Idempotent Substitution)
A substitution σ is *idempotent* if $\sigma \circ \sigma = \sigma$. △

It is easy to show that if $\mathcal{I}(\sigma) \cap \mathcal{DOM}(\sigma) = \emptyset$ holds, σ is idempotent.

Lemma 3.5 *If $\theta \in \mathrm{CSU}(U)$ for some unification problem U then θ is idempotent.*

Proof: Follows directly from the last condition in the definition of a complete set of unifiers $\mathrm{CSU}(U)$.

Definition 3.6 (Solved Form)
An equation $M \stackrel{?}{=} N$ is in *solved form* in a unification problem $\langle \sigma, S \rangle$ if it is in the form $\eta[F\colon T] \stackrel{?}{=} N$ for some variable $F\colon T$ which occurs exactly once in U, and $F\colon T$ and N have the same type. A system S is *solved* if all of its pairs are solved. A unification problem $\langle \sigma, S \rangle$ is *solved* if S is solved.

With a system $S = \{F_1\colon T_1 \stackrel{?}{=} N_1, \ldots, F_n\colon T_n \stackrel{?}{=} N_n\}$ in solved form we associate a term substitution $\lceil S \rceil^{\mathrm{SUB}} = \{F_1\colon T_1/N_1, \ldots, F_n\colon T_n/N_n\}$. This substitution is unique up to variable renaming. With a unification problem $\langle \sigma, S \rangle$ in solved form we associate a substitution $\lceil U \rceil^{\mathrm{SUB}} = \langle \sigma, \lceil S \rceil^{\mathrm{SUB}} \rangle$. △

4 Problems with Unification in $\mathcal{L}^{Poly}$

As stated in Nipkow (1990) type variables cause new problems through the possibility of type instantiation. Assume we use the transformation system $\mathcal{HT}$ defined in Snyder and Gallier (1989). $\mathcal{HT}$ is correct and complete for higher-order unification in the simply typed λ-calculus. In the following example we consider a system of polymorphically typed λ-terms with $\mathcal{HT}$. We examine the effect of type variables on the applicability of the projection rule of $\mathcal{HT}$ which is

Projection

$$\{\lambda\,\overline{x_k\colon T_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\,\overline{x_k\colon T_k}.\, a(\overline{N_n})\} \cup D$$
$$\Downarrow$$
$$\{F \stackrel{?}{=} P, \lambda\,\overline{x_k\colon T_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\,\overline{x_k\colon T_k}.\, a(\overline{N_n})\} \cup \{F/P\}(D)\downarrow,$$

where

- D is a system,
- F is a free variable and a an arbitrary atom,
- P is a variant of a ith projection binding for $1 \leq i \leq m$, appropriate to the term $\lambda\,\overline{x_k\colon T_k}.\, F(\overline{M_m})$, that is, $P = \lambda\,\overline{y_m\colon S_m}.\, y_i(\lambda\,\overline{z_{p_q}\colon R_{p_q}}.\, H_q(\overline{y_m}, \overline{z_{p_q}}))$, and
- $\mathrm{head}(M_i) = a$, if $\mathrm{head}(M_i)$ is a constant.

Example 4.1 Let A be a type variable and t some base type, such that $\mathrm{type}(F) = (A \to t)$ and $\mathrm{type}(G) = A$ for free variables F and G. We consider the unification problem U

$$\langle \iota_1, \{F(G) \stackrel{?}{=} a\}\rangle.$$

If we want to instantiate F with a projection binding the only possibility is to use the first projection binding because there's only one argument G. For the first projection binding to be appropriate for F we must have $\mathrm{range}(G) = \mathrm{range}(F)$. A complete set of type substitutions for A satisfying this equation is

$$\Theta = \{\{A/(A_1, \ldots, A_m \to t)\} \mid A_1, \ldots, A_m \text{ type variables}, m \geq 0\}.$$

For every substitution θ_1^m in this set the projection rule is applicable using the projection binding

$$\theta_2^m = \lambda\, y\colon (A_1, \ldots, A_m \to t).\, y(H_1(y), \ldots, H_m(y)),$$

where $\mathrm{type}(H_i) = ((A_1, \ldots, A_m \to t) \to A_i)$.

After applying θ_2^m to $\theta_1^m(F)$ we get the unification problem

$$S_m = \langle \theta_1^m, \{G(H_1(G), \ldots, H_m(G)) \stackrel{?}{=} a\}\rangle.$$

For the transformed unification problem S_m the substitution

$$\omega_2^m = \{G/\lambda\, \overline{x_m\colon A_m}.\, a\}$$

is a unifier. Depending on m we get an infinitary set of non-subsuming unifiers for the unification problem U. So if the transformation system for higher-order unification in $\mathcal{L}^{Poly}$ contains a projection rule like $\mathcal{HT}$ does, it will be infinitary branching even for preunification or it will be incomplete, if it only considers a finitary subset of Θ.

5 An Analysis of Polymorphic Unifiers

We analyse the process of polymorphical higher-order unification as follows. Without loss of generality we assume that M and N are two lambda terms in $\mathcal{L}_\eta^{Poly}$ and that θ is an idempotent normalized unifier of M and N. In other words M has the form

$$M \quad = \quad \lambda\, \overline{x_k\colon A_k}.\, a(M_1, \ldots, M_m)$$

where M has type $A = (A_1, \ldots, A_k \to A_0)$ and a has type $B = (B_1, \ldots, B_m \to A_0)$. N has the form

$$N \quad = \quad \lambda\, \overline{x_l\colon C_l}.\, b(N_1, \ldots, N_n),$$

where N has type $C = (C_1, \ldots, C_l \to C_0)$ and b has type $D = (D_1, \ldots, D_n \to C_0)$. And there exists some sequence of reductions to a β-normal form $\eta[\theta(M)] \xrightarrow{*}_\beta P \xleftarrow{*}_\beta \eta[\theta(N)]$. We analyse this sequence top-down. We have the following cases (which are intended to be mutually exclusive):

(A) $M = N$ and no unification is necessary. We assume M and N to be distinct in the remaining cases.

(B) Suppose $k < l$. Then θ instantiates the type of M such that a sequence of η-expansions enlarges the binder of M. This is only possible if A_0 is a type variable. Thus

$$\begin{array}{rcl} \theta(A_0) & = & (\theta(C_{k+1}), \ldots, \theta(C_l) \to \theta(C_0)) \\ & = & \theta((C+1, \ldots, C_l \to C_0)) \\ & = & \theta(\{A_0/(C_{k+1}, \ldots, C_l \to C_0)\}(A_0)) \end{array}$$

and θ is an instance of the most general unifier of A_0 and $(C_{k+1}, \ldots, C_l \to C_0)$.

(C) Suppose $k = l$ but $A_i \neq C_i$ for some i, $0 \leq i \leq k$. Then we have

$$\theta(A_i) \quad = \quad \theta(C_i)$$

and θ must be an instance of the most general unifier of A_i and C_i. In the remaining cases we assume M and N have the same type. This implies that the binders of M and N are identical up to α-conversion.

(D) Suppose $\mathrm{head}(M) = a\colon(B_1, \ldots, B_m \to A_0)$ and $\mathrm{head}(N) = a\colon(D_1, \ldots, D_m \to A_0)$ are atoms with distinct types. Then we have $B_i \neq D_i$ for some i, $1 \leq i \leq m$. Then θ possibly unifies B_i and D_i, i.e.

$$\theta(B_i) \quad = \quad \theta(D_i).$$

So θ is an instance of the most general unifier of B_i and D_i.

Note that B and D necessarily have the same number of argument types.

(E) No substitution takes place in the head of either term. In this case, $\mathrm{head}(M) = \mathrm{head}(N)$. Suppose

$$M \quad = \quad \lambda\overline{x_k\colon A_k}.\, a(M_1, \ldots, M_m)$$

and

$$N \quad = \quad \lambda\overline{x_k\colon A_k}.\, a(N_1, \ldots, N_m),$$

where either $a \in \Sigma$ or $a = x_i$ for some i, $1 \leq i \leq k$, or a is a free variable not in $\mathcal{DOM}(\theta)$. Here we must have

$$\eta[\theta(\lambda\overline{x_k\colon A_k}.\, M_i)] \xrightarrow{*}_\beta \lambda\overline{x_k\colon A_k}.\, P_i \xleftarrow{*}_\beta \eta[\theta(\lambda\overline{x_k\colon A_k}.\, N_i)]$$

for $1 \leq i \leq m$. That is, the subterms of M and N are pair-wise unifiable by θ.

(F) The two terms are

$$M \quad = \quad \lambda\overline{x_k\colon A_k}.\, F(\overline{x_k})$$

and

$$N \quad = \quad \lambda\overline{x_k\colon A_k}.\, b(N_1, \ldots, N_m)$$

for some free variable F c.f. $F \notin FV(N)$. In this case, we must have

$$\eta[\theta(\lambda\overline{x_k\colon A_k}.\, F(\overline{x_k}))] \xleftrightarrow{*}_\beta \lambda\overline{x_k\colon A_k}.\, b(N_1, \ldots, N_m)$$

and we know that $\sigma = \{F/\lambda\overline{x_k\colon A_k}.\, b(N_1, \ldots, N_m)\}$ is a most general unifier of this equation.

(G) Some substitution takes place in the head of only one term. Assume that this term is M. Then b is either a function constant, a bound variable, or a free variable not in $\mathcal{DOM}(\theta)$. Now in order for the two terms to unify, we must ensure that at some point in the sequence of β-reductions from $\theta(M)$ to P the head of M becomes b. There are two possibilities: either we imitate the head of N by substituting a term for F whose head is b or we substitute a term for F which projects up a subterm of M. We consider each of these possibilities in turn.

(Imitation) The substitution for F matches the head symbol of N by imitating the head symbol b where $b \in \Sigma$ or b is a free variable not in $\mathcal{DOM}(\theta)$. In contrast to the simply typed case we need to take the type substitution θ_1 into account. Suppose N is a rigid term of the form

$$N = \lambda\overline{x_k\colon A_k}.\, b\colon D(\overline{N_m})$$

and the application of θ_1 to D yields

$$\theta_1(D) = \theta_1((D_1, \ldots, D_m \to A_0)) = D' = (D'_1, \ldots, D'_m, A'_{k+1}, \ldots, A'_{k+p} \to A'_0).$$

Then we get the following reduction sequence

$$\begin{array}{rcl}\theta(N) & = & \theta_2(\lambda\,\overline{x_k\colon A'_k}.\,b\colon D'(\overline{N'_m}))\\ & = & \lambda\,\overline{x_k\colon A'_k}.\,b\colon D'(\overline{\theta_2(N'_m)})\\ & \xrightarrow{*}_{\beta} & \lambda\,\overline{x_k\colon A'_k}.\,b\colon D'(\overline{P_m})\\ & \xleftarrow{*}_{\eta} & \lambda\,\overline{x_{k+p}\colon A'_{k+p}}.\,b\colon D'(\overline{P_m}, x_{k+1},\ldots,x_{k+p})\end{array}$$

where N'_i is the result of applying θ_1 to N_i and P_i is the η-expanded normal form of $\theta_2(N'_i)$ for all i, $1 \le i \le m$.

We can assume that M has the form

$$M = \lambda\,\overline{x_k\colon A_k}.\,F\colon B(\overline{M_m})$$

and the result of applying θ_1 to the type of M and the type of the head of M will be

$$\theta_1(\text{type}(M)) = \theta_1((A_1,\ldots,A_k \to A_0)) = A' = (A'_1,\ldots,A'_k, A'_{k+1},\ldots,A'_{k+p} \to A'_0),$$

and

$$\theta_1(B) = \theta_1((B_1,\ldots,B_m \to A_0)) = B' = (B'_1,\ldots,B'_m, A'_{k+1},\ldots,A'_{k+p} \to A'_0).$$

Note that B must have the same number of argument types as D.

The η-expanded normal form of $\theta_1(M)$ will be

$$\lambda\,\overline{x_{k+p}\colon A'_{k+p}}.\,F\colon B'(\overline{M'_m}, x_{k+1},\ldots,x_{k+p})$$

where M'_i is the η-expanded normal form of $\theta_1(M_i)$ for all i, $1 \le i \le m$. Thus θ must take the form

$$\theta(F\colon B') = \lambda\,\overline{z_m\colon B'_m}, z_{m+1}\colon A'_{k+1},\ldots,z_{m+p}\colon A'_{k+p}.\,b\colon D'(\overline{Q_m}, z_{m+1},\ldots,z_{m+p})$$

for some terms $\overline{Q_m}$ of appropriate types. Note that none of the z_{m+1},...,z_{m+p} occurs in one of the Q_i, since none of them occurs in one of the $P_i = \eta[\theta_1(N_i)]$ for some i, $1 \le i \le m$.

This leaves us with a reduction sequence of the form

$$\begin{array}{rcl}\theta(M) & = & \theta_1(\lambda\,\overline{x_{k+p}\colon A'_{k+p}}.\,F\colon B'(\overline{M'_m}, \overline{x_{k+1\ldots k+p}}))\\ & \longrightarrow_{\beta} & \lambda\,\overline{x_{k+p}\colon A'_{k+p}}.\,b\colon D'(\overline{P_m}, \overline{x_{k+1\ldots k+p}})\\ & \xrightarrow{*}_{\eta} & \lambda\,\overline{x_k\colon A'_k}.\,b\colon D'(\overline{P_m})\\ & \xleftarrow{*}_{\beta} & \theta(\lambda\,\overline{x_k\colon A_k}.\,b\colon D(\overline{N_n})).\end{array}$$

Note that

$$\lambda\,\overline{z_m\colon B'_m}, z_{m+1}\colon A'_{k+1},\ldots,z_{m+p}\colon A'_{k+p}.\,b\colon D'(\overline{Q_m}, z_{m+1},\ldots,z_{m+p})$$

is an type-instance of

$$\lambda\,\overline{z_m\colon B_m}.\,b\colon D(\overline{Q_m}).$$

(Projection) The substitution for F projects up a subterm of M. When substituting a ith projection for the head of a flexible term $M = \lambda\,\overline{x_k\colon A_k}.\,F(\overline{M_m})$ we constrain the type of M_i as follows:

$$\begin{array}{rcl}\text{range}(\text{type}(\theta(M_i))) & = & \text{range}(\theta_1(B_i))\\ & = & \text{range}(\text{type}(\theta(F)))\\ & = & \text{range}(\theta_1((B_1,\ldots,B_m \to A_0)))\\ & = & \text{range}(\theta_1(A_0)).\end{array}$$

Assume $\theta_1((B_1,\ldots,B_m \to A_0)) = B' = (B'_1,\ldots,B'_m,B'_{m+1},\ldots,B'_p \to E_0)$ for some types $B'_0,\ldots,B'_p$ and $m \leq p$. The substitution for F takes the form $\{F\colon B'/\lambda\overline{y_p\colon B'_p}.\, y_i(\overline{L_q})\}$ for some terms $\overline{L_q}$ and some i in $\{1,\ldots,m\}$ such that $\mathrm{type}(y_i) = \mathrm{type}(\theta(M_i)) = (E_1,\ldots,E_q \to E_0)$. We have a reduction sequence of the form

$$\begin{array}{rcl}
\theta_2(\eta[\theta_1(M)]) & = & \theta_2(\lambda\overline{x'_{k+p-m}}.\, F\colon B'(M'_1,\ldots,M'_m,x'_{k+1},\ldots,x'_{k+p-m})) \\
& = & \lambda\overline{x'_{k+p-m}}.\,\lambda\overline{y_p}.\, y_i(\overline{L_q})(s'_1,\ldots,M'_m,x'_{k+1},\ldots,x'_{k+p-m}) \\
& \longrightarrow_\beta & \lambda\overline{x'_{k+p-m}}.\, M'_i(\overline{L_q}) \\
& \stackrel{*}{\longrightarrow}_\beta & \lambda\overline{x'_{k+p-m}}.\, a'(\overline{P_r}) \\
& \stackrel{*}{\longleftrightarrow}_\beta & \theta(N).
\end{array}$$

In this case, the head b of N may be a function constant, a free variable or a bound variable. Note that $i \leq m$ because b can not be equal to any of the $x'_{k+1},\ldots,x'_{k+p-m}$.

(H) Substitutions take place at the head of both terms. Then let $M = \lambda\overline{x_k\colon A_k}.\, F(\overline{M_m})$ and $N = \lambda\overline{x_k\colon A_k}.\, G(\overline{N_n})$ for both F and G in $\mathcal{DOM}(\theta)$. Here we must eventually match the heads of the two terms, but we can do it in a large number of ways. In order to simplify our analysis if possible, we reduce it to the previous case. Let us (without loss of generality) focus on the binding for the variable F. There are two subcases.

(1) θ substitutes a non-projection term for F, e.g., $\theta(F) = \lambda\overline{y_p}.\, a(\overline{r_q})$, where $a \neq G$ is not a bound variable. By idempotency a is not a variable in $\mathcal{DOM}(\theta)$. The substitution (possibly) causes a β-reduction, after which we can analyse the result using **(G)**.

(2) θ substitutes a projection term for F (which obeys the typing constraints discussed above), e.g., $\theta(F) = \lambda\overline{y_p}.\, y_j(\overline{L_q})$. For further analysis we use **(G)** if the head symbol is either a function constant, a bound variable, or a variable not in $\mathcal{DOM}(\theta)$. We use **(H)** if the head is a variable in $\mathcal{DOM}(\theta)$.

It is straightforward to formulate transformation rules for cases **(A)**, **(C)**, **(D)**, **(E)**, and **(F)**. To formulate the transformation analysed in case **(B)**, we need the notion of a *binder expanding substitution*:

Definition 5.1 (Binder expanding substitution)
A type substition $\{A_0/(A_1 \to A_2)\}$ with type variables A_0, A_1, and A_2 is called a *binder expanding substitution* for a term of type $(B_1,\ldots,B_m \to A_0)$, where B_1, …, B_m are arbitrary types. △

The substitution $\{A_0/(C_{k+1},\ldots,C_l \to C_0)\}$ in case **(B)** is an instance of the composition of $l-(k+1)$ binder expanding substitutions.

The transformations associated with cases **(G)** and **(H)** can be formulated using partial binding substitutions. I use the definition due to Snyder and Gallier (1989).

Definition 5.2 (Partial binding)
A *partial binding* of type $(A_1,\ldots,A_n \to B)$ (for B atomic) is the η-expanded form of a term of the form

$$\lambda\overline{y_n\colon A_n}.\, a(H_1(\overline{y_n}),\ldots,H_m(\overline{y_n}))$$

for some atom a of type $(C_1,\ldots,C_m \to B)$, where $C_i = (D^i_1,\ldots,D^i_{p_i} \to C'_i)$ for $1 \leq i \leq m$. The H_i have type $(A_1,\ldots,A_n,D^i_1,\ldots,D^i_{p_i} \to C'_i)$ for $1 \leq i \leq m$. C'_1, …, C'_m are atomic types. The arguments of a partial binding will be called *general flexible terms*. △

We will now show that every partial binding can be constructed using matrix expanders and selectors.

Definition 5.3 (Matrix expander)
A *matrix expander* of type $(A_1,\ldots,A_{m+n} \to A)$ (where A is a base type) is a term of the form

$$\lambda\overline{y_{m+n}\colon A_{m+n}}.\, F(\overline{y_{m+n}}, G(\overline{y_m}))$$

with free variables F and G such that

- $\text{type}(F) = (A_1, \ldots, A_{m+n}, B \to A)$ and
- $\text{type}(G) = (A_1, \ldots, A_m \to B)$.

A substitution $\{H\colon D/M\}$ with M a matrix expander of type D is called a *matrix expanding substitution.* $\triangle$

Definition 5.4 (Selector)
A *selector* appropriate to type $(A_1, \ldots, A_m, B_1, \ldots, B_n \to C)$ is a term of the form

$$\lambda\, \overline{y_m\colon A_m}, \overline{z_n\colon B_n}.\, a(\overline{z_n})$$

where a is some arbitrary atom of type $(B_1, \ldots, B_n \to C)$. A substitution $\{H\colon D/M\}$ with M a selector of type D is called a *selector substitution.* $\triangle$

Lemma 5.5 *For any partial binding substitution*

$$\tau = \{F/\lambda\, \overline{y_m\colon A_m}.\, a(\lambda\, \overline{z^1_{p_1}\colon B^1_{p_1}}.\, H_1(\overline{y_m}, \overline{z^1_{p_1}}), \ldots, \lambda\, \overline{z^n_{p_n}\colon B^n_{p_n}}.\, H_n(\overline{y_m}, \overline{z^n_{p_n}}))\}$$

such that

- *F is free variable of type $(A_1, \ldots, A_m \to A)$,*
- *a is some arbitrary atom of type $(C_1, \ldots, C_n \to A)$ with $C_i = (B^i_1, \ldots, B^i_{p_i} \to B^i)$ for $1 \le i \le n$ and $B^i \in \mathcal{T}_0$, and*
- *$\text{type}(H_i) = (A_1, \ldots, A_m, B^i_1, \ldots, B^i_{p_i} \to B^i)$ for $1 \le i \le n$,*

there exist matrix expanding substitutions τ_1, ..., τ_n and a selector substitution τ_0, such that

$$\tau = \tau_0 \circ \tau_n \circ \cdots \circ \tau_1.$$

Proof: Let

$$\begin{aligned}
\tau_1 &= \{F/\lambda\, \overline{y_m\colon A_m}.\, F_1(\overline{y_m}, \lambda\, \overline{z^1_{p_1}\colon B^1_{p_1}}.\, H_1(\overline{y_m}, \overline{z^1_{p_1}}))\} \\
\text{type}(F_1) &= (A_1, \ldots, A_m, (B^1_1, \ldots, B^1_{p_1} \to B^1) \to A) \\
&= (A_1, \ldots, A_m, C_1 \to A) \\
\text{type}(H_1) &= (A_1, \ldots, A_m, B^1_1, \ldots, B^1_{p_1} \to B^1) \\
\tau_2 &= \{F_1/\lambda\, \overline{y_m\colon A_m}, x_1\colon C_1.\, F_2(\overline{y_m}, x_1, \lambda\, \overline{z^2_{p_2}\colon B^2_{p_2}}.\, H_2(\overline{y_m}, \overline{z^2}_{p_2}))\} \\
\text{type}(F_2) &= (A_1, \ldots, A_m, C_1, (B^2_1, \ldots, B^2_{p_2} \to B^2) \to A) \\
&= (A_1, \ldots, A_m, C_1, C_2 \to A) \\
\text{type}(H_2) &= (A_1, \ldots, A_m, B^2_1, \ldots, B^2_{p_2} \to B^2) \\
&\vdots \\
\tau_n &= \{F_{n-1}/\lambda\, \overline{y_m\colon A_m}, \overline{x_{n-1}\colon C_{n-1}}.\, F_n(\overline{y_m}, \overline{x_{n-1}}, \lambda\, \overline{z^n_{p_n}\colon B^n_{p_n}}.\, H_n(\overline{y_m}, \overline{z^n_{p_n}}))\} \\
\text{type}(F_{n-1}) &= (A_1, \ldots, A_m, C_1, \ldots, C_{n-1} \to A) \\
\text{type}(F_n) &= (A_1, \ldots, A_m, C_1, \ldots, C_{n-1}, (B^n_1, \ldots, B^n_{p_n} \to B^n) \to A) \\
\text{type}(H_n) &= (A_1, \ldots, A_m, B^n_1, \ldots, B^n_{p_n} \to B^n) \\
\tau_0 &= \{F_n/\lambda\, \overline{y_m\colon A_m}, \overline{x_n\colon C_n}.\, a(\overline{x_n})\} \\
\text{type}(a) &= (A_1, \ldots, A_m \to A)
\end{aligned}$$

We have

$$\begin{aligned}
\tau_0 \circ \tau_n \circ \cdots \circ \tau_1 &= \{F/\lambda\, \overline{y_m\colon A_m}.\, a(\lambda\, \overline{z^1_{p_1}\colon B^1_{p_1}}.\, H_1(\overline{y_m}, \overline{z^1_{p_1}}), \ldots, \lambda\, \overline{z^n_{p_n}\colon B^n_{p_n}}.\, H_n(\overline{y_m}, \overline{z^n_{p_n}}))\} \\
&= \tau
\end{aligned}$$

6 The Transformation System $\mathcal{HPT}$

We define the following transformation system on unification problems using the notions of binder expanding, matrix expanding and selector substitutions.

Definition 6.1 (Transformation system $\mathcal{HPT}$)
The rules for the transformation system $\mathcal{HPT}$ for unification problems in $\mathcal{L}^{Poly}$ are:
Trivial removal

$$\langle \sigma, \{M \stackrel{?}{=} M\} \cup S\rangle \Rightarrow \langle \sigma, S\rangle \qquad \mathcal{HPT}_1$$

Type unification

$$\begin{array}{c}\langle \sigma, \{M \stackrel{?}{=} N\} \cup S\rangle \\ \Downarrow \\ \langle \theta \circ \sigma, \theta(\{M \stackrel{?}{=} N\} \cup S)\rangle\end{array} \qquad \mathcal{HPT}_2$$

where

- M is a term of type A and N is term of type C,
- $A \neq C$, and
- $\theta = \mathrm{mgu}(\{A \stackrel{?}{=} C\})$.

Head type unification

$$\begin{array}{c}\langle \sigma, \{\lambda\overline{x_k\colon A_k}.\, a\colon B(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, a\colon D(\overline{N_m})\} \cup S\rangle \\ \Downarrow \\ \langle \theta \circ \sigma, \theta(\{\lambda\overline{x_k\colon A_k}.\, a\colon B(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, a\colon D(\overline{N_m})\} \cup S)\rangle\end{array} \qquad \mathcal{HPT}_3$$

where

- $B \neq D$,
- $\theta = \mathrm{mgu}(\{B \stackrel{?}{=} D\})$.

Decomposition

$$\begin{array}{c}\langle \sigma, \{\lambda\overline{x_k\colon A_k}.\, a(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, a(\overline{N_m})\} \cup S\rangle \\ \Downarrow \\ \langle \sigma, \bigcup_{1\leq i\leq m} \{\lambda\overline{x_k\colon A_k}.\, M_i \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, N_i\} \cup S\rangle\end{array} \qquad \mathcal{HPT}_4$$

where a is some arbitrary atom.
Variable elimination

$$\begin{array}{c}\langle \sigma, \{\lambda\overline{x_k\colon A_k}.\, F(\overline{x_k}) \stackrel{?}{=} N\} \cup S\rangle \\ \Downarrow \\ \langle \sigma, \{\lambda\overline{x_k\colon A_k}.\, F(\overline{x_k}) \stackrel{?}{=} N\} \cup \{F/N\}(S)\rangle\end{array} \qquad \mathcal{HPT}_5$$

where

- F is a free variable,
- $F \in FV(S)$ and $F \notin FV(N)$, and
- $\mathrm{type}(F) = \mathrm{type}(N)$.

Binder-Expansion

$$\begin{array}{c}\langle \sigma, \{\lambda\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, b(\overline{N_n})\} \cup S\rangle \\ \Downarrow \\ \langle \theta \circ \sigma, \eta[\theta(\{\lambda\overline{x_k\colon M_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon M_k}.\, b(\overline{N_n}) \cup S\})]\rangle,\end{array} \qquad \mathcal{HPT}_{6a}$$

where

- F is a free variable of type $(B_1, \ldots, B_m \to A_0)$,
- A_0 is a type variable, and
- $\theta = \{A_0/(C_1 \to C_2)\}$ for type variables C_1 and C_2.

Matrix-Expansion

$$\begin{array}{c}\langle \sigma, \{\lambda\,\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\,\overline{x_k\colon A_k}.\, b(\overline{N_n})\} \cup S\rangle \\ \Downarrow \\ \langle \sigma, \{F \stackrel{?}{=} Q\} \cup \{F/Q\}(\{\lambda\,\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\,\overline{x_k\colon A_k}.\, b(\overline{N_n})\} \cup S)\downarrow\rangle,\end{array} \quad \mathcal{HPT}_{6b}$$

where

- F is a free variable of type $(B_1, \ldots, B_m \to A_0)$;
- b is an arbitrary atom of type $\mathrm{type}(a) = (D_1, \ldots, D_n \to A_0)$;
- Q is a variant of a matrix expander of type $(B_1, \ldots, B_m \to A_0)$, i.e. $M = \lambda\,\overline{y_m\colon B_m}.\, G(\overline{y_m}, H(\overline{y_l}))$ with $l \leq m$.

Selection

$$\begin{array}{c}\langle \sigma, \{\lambda\,\overline{x_k\colon T_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\,\overline{x_k\colon T_k}.\, b(\overline{N_n})\} \cup S\rangle \\ \Downarrow \\ \langle \theta \circ \sigma, \{\theta(F) \stackrel{?}{=} Q\} \cup \{\theta(F)/Q\}(\theta(\{\lambda\,\overline{x_k\colon T_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\,\overline{x_k\colon T_k}.\, b(\overline{N_n})\} \cup S))\downarrow\rangle,\end{array} \quad \mathcal{HPT}_{7}$$

where

- F is a free variable of type $(\overline{B_m} \to A_0)$;
- b is some arbitrary atom of type $(\overline{D_n} \to D_0)$;
- a is some arbitrary atom of type $(E_1, \ldots, E_k \to E_0)$, for some k and some l, $k \leq m - l$, $0 \leq l \leq m$, such that $(B_{l+1}, \ldots, B_m \to A_0)$ and $\mathrm{type}(a)$ have a most general unifier θ;
- Q is a variant of a selector appropriate for type $\theta((B_1, \ldots, B_m \to A_0))$, i.e.

$$Q = \eta[\theta(\lambda\,\overline{y_m\colon B_m}.\, a(y_{l+1}, \ldots, y_m))],$$

such that $l \leq m$.

△

7 Correctness and Completeness of $\mathcal{HPT}$

7.1 Correctness of $\mathcal{HPT}$

Lemma 7.1 *Let τ be an idempotent type substitution and M, N two terms. If θ is an instance of τ then θ is a unifier of $\tau(M) \stackrel{?}{=} \tau(N)$ iff θ is a unifier of $M \stackrel{?}{=} N$.*

Proof: Because θ is an instance of τ there exists a substitution ρ such that $\theta = \rho \circ \tau$. So we have

$$\begin{array}{ll} & \theta(\tau(M)) = \theta(\tau(N)) \\ \text{iff} & \rho(\tau(\tau(M))) = \rho(\tau(\tau(N))) \\ \text{iff} & \rho(\tau(M)) = \rho(\tau(N)) \\ \text{iff} & \theta(M) = \theta(N). \end{array}$$

Lemma 7.2 *If $U \Longrightarrow_{\mathcal{HPT}} U'$ using one of the transformation rules $\mathcal{HPT}_1$, $\mathcal{HPT}_2$, or $\mathcal{HPT}_5$, then $\mathrm{U}(U) = \mathrm{U}(U')$.*

Proof: We consider each of the transformation rules in turn:

$\mathcal{HPT}_1$ If θ is a unifier for a unification problem $\langle \sigma, S\rangle$ then θ is also a unifier of $\langle \sigma, M \stackrel{?}{=} M \cup S\rangle$. The converse holds because $\theta(M) = \theta(M)$ holds for any substitution θ.

$\mathcal{HPT}_2$ Suppose τ is a unifier of $\langle \sigma, M \stackrel{?}{=} N \cup S\rangle$. Then τ_1 unifies the types of M and N and because there exists a most general unifier θ of type(M) and type(N), τ_1 must be an instance of θ. So a variant of τ is also a unifier of $\langle \theta \circ \sigma, \theta(M \stackrel{?}{=} N \cup S)\rangle$.
Suppose on the other hand τ is a unifier of $\langle \theta \circ \sigma, \theta(M \stackrel{?}{=} N \cup S)\rangle$. Then τ must be an instance of θ and therefore a variant of τ is a unifier of $\langle \sigma, M \stackrel{?}{=} N \cup S\rangle$.

$\mathcal{HPT}_5$ Let ρ be the substitution $\{F/N\}$. For any substitution τ if $\tau(F) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \tau(N)$ then $\tau =_{\beta\eta} \tau \circ \rho$, since $\tau \circ \rho$ differs from τ only at F but $\tau(F) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \tau(N) = \tau \circ \rho(F)$. We have $\tau \in \mathrm{U}(U)$ iff $\tau \circ \rho \in \mathrm{U}(U)$. Furthermore, since for any term P we have $\tau \circ \rho(P) = \tau(\rho(u)) \stackrel{*}{\longrightarrow}_{\beta} \tau(\rho(P)\downarrow)$, it can easily be shown that $\tau \circ \rho \in \mathrm{U}(U)$ iff $\tau \in \mathrm{U}(\rho(U)\downarrow)$. Thus

$$\begin{array}{ll} & \tau \in \mathrm{U}(\{F \stackrel{?}{=} N\} \cup S) \\ \text{iff} & \tau(F) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \tau(N) \text{ and } \tau \in \mathrm{U}(N) \\ \text{iff} & \tau(F) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \tau(N) \text{ and } \tau \circ \rho \in \mathrm{U}(U) \\ \text{iff} & \tau(F) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \tau(N) \text{ and } \tau \in \mathrm{U}(\rho(U)\downarrow) \\ \text{iff} & \tau \in \mathrm{U}(\{F \stackrel{?}{=} N\} \cup \rho(U)\downarrow). \end{array}$$

Lemma 7.3 *Let $U \Longrightarrow_{\mathcal{HPT}_3} U'$ where the transformed equation in U is*

$$\lambda\overline{x_k\colon A_k}.\, a\colon B(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, a\colon D(\overline{N_m}).$$

Let θ be any substitution. Then

(1) *$\theta \in \mathrm{U}(U')$ iff $\theta \in \mathrm{U}(U)$ if a is either a constant or a free variable not in $\mathcal{DOM}(\theta)$.*

(2) *$\theta \in \mathrm{U}(U')$ implies $\theta \in \mathrm{U}(U)$ if $a \in \mathcal{DOM}(\theta)$.*

Proof: Let $U' = \langle \tau \circ \sigma, D'\rangle$ and $U = \langle \sigma, D\rangle$. Let θ be a unifier of U'. Then θ is an instance of $\tau \circ \sigma$ where τ is an idempotent substitution. So following lemma 7.1 θ is a unifier of every equation in D. And because $\mathcal{DOM}(\sigma) \cap \mathcal{DOM}(\tau) = \emptyset$ the substitution θ is an instance of σ. So θ is a unifier of the unification problem U.

Now let θ be a unifier of U. Suppose a is either a constant or a free variable not in $\mathcal{DOM}(\theta)$. Then $\theta(a\colon B) = a\colon \theta_1(B) = a\colon \theta_1(D) = \theta(a\colon D)$. So θ is an instance of the most general unifier τ of B and D. Using lemma 7.1 we can conclude that θ must be a unifier of U'.

Lemma 7.4 *Let $U \Longrightarrow_{\mathcal{HPT}_4} U'$ where the transformed equation in U is $\lambda\overline{x_k\colon A_k}.\, a(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, a(\overline{N_m})$. Let θ be any substitution. Then*

(1) *$\theta \in \mathrm{U}(U')$ iff $\theta \in \mathrm{U}(U)$ if a is either a constant or a bound variable or a free variable not in $\mathcal{DOM}(\theta)$.*

(2) *$\theta \in \mathrm{U}(U')$ implies $\theta \in \mathrm{U}(U)$ if $a \in \mathcal{DOM}(\theta)$.*

Proof: If $\rho(\lambda\overline{x_k\colon A_k}.\, M_i) \stackrel{*}{\longleftrightarrow}_{\beta} \rho(\lambda\overline{x_k\colon A_k}.\, N_i)$ for $1 \leq i \leq m$, then clearly we have $\rho(\lambda\overline{x_k}.\, a(\overline{M_m})) = \rho(\lambda\overline{x_k}.\, a(\overline{N_m}))$. So for any atom a we have $\rho \in \mathrm{U}(U)$ whenever $\rho \in \mathrm{U}(U')$. If a is either a function constant, a bound variable or a variable not in $\mathcal{DOM}(\rho)$, then $\rho(a) = a$. It is easy to see that the converse direction holds as well.

Lemma 7.5 *If $U \Longrightarrow_{\mathcal{HPT}} U'$ using one of the transformation rules $\mathcal{HPT}_3$ or $\mathcal{HPT}_4$ then $\mathrm{U}(U') \subseteq \mathrm{U}(U)$.*

Proof: This is a consequence of lemma 7.3 and 7.4.

Lemma 7.6 *If $U \Longrightarrow_{\mathcal{HPT}} U'$ using transformation rule $\mathcal{HPT}_{6a}$ then $\mathrm{U}(U') \subseteq \mathrm{U}(U)$.*

Proof: In this case $U' = \theta(U)$ where θ is an idempotent type substitution. So we can use lemma 7.1 to prove $\mathrm{U}(U') \subseteq \mathrm{U}(U)$.

Lemma 7.7 *If $U \Longrightarrow_{\mathcal{HPT}} U'$ using transformations $\mathcal{HPT}_{6b}$ or $\mathcal{HPT}_7$ then $\mathrm{U}(U') \subseteq \mathrm{U}(U)$.*

Proof: These transformations add a pair $\{F \stackrel{?}{=} Q\}$ to the system S of the unification problem $U = \langle \sigma, S \rangle$ and then apply the substitution $\{F/Q\}$ to S. Since $S \subseteq \{F \stackrel{?}{=} Q\} \cup S$ we have $\mathrm{U}(\{F \stackrel{?}{=} Q\} \cup S) \subseteq \mathrm{U}(S)$. The application of the substitution can be seen as an application of $\mathcal{HPT}_5$ which has been shown to be sound in lemma 7.2.

Theorem 7.8 (Soundness) *If $U \stackrel{*}{\Longrightarrow}_{\mathcal{HPT}} U'$, with U' in solved form, then the substitution $\lceil U' \rceil^{\mathrm{SUB}} \restriction_{FV(U)} \in \mathrm{U}(U)$.*

Proof: By induction on the length of transformation sequences, and using the previous lemmas in the induction step, we can show that $\lceil U' \rceil^{\mathrm{SUB}} \in \mathrm{U}(U)$. Since the restriction to the free variables of U does not change the effect of the substitution on the terms in U, we see that $\lceil U' \rceil^{\mathrm{SUB}} \restriction_{FV(U)} \in \mathrm{U}(U)$.

7.2 Completeness of $\mathcal{HPT}$

In this section we will give a completeness proof for the transformation system $\mathcal{HPT}$.

Lemma 7.9 *Let U be any unification problem, ϕ any substitution, and Z any set of protected variables. Then, provided $\phi \in \mathrm{SU}(U)$, there exists some normalized substitution θ such that*

(1) *$\theta \in \mathrm{SU}(U)$;*

(2) *$\theta \leq_{\beta\eta} \phi\,[FV(U)]$ and $\phi \leq_{\beta\eta} \theta\,[FV(U)]$;*

(3) *$\mathcal{DOM}(\theta) \subseteq FV(U)$ and $\mathcal{I}(\theta) \cap (Z \cup \mathcal{DOM}(\theta)) = \emptyset$.*

Proof: Suppose $\phi \restriction_{FV(U)}$ is normalized and satisfies condition **(3)**. Then $\theta = \phi \restriction_{FV(U)}$ is the substitution we are looking for.

Otherwise, we assume that $\mathcal{I}(\phi) \cap \mathcal{FV} = \{F_1, \ldots, F_n\}$ holds. We can choose a set of free variables $\{G_1, \ldots, G_n\}$ such that $\mathrm{type}(G_i) = \mathrm{type}(F_i)$ for all i, $1 \leq i \leq n$, holds and the intersection with $Z \cup \mathcal{I}(\phi) \cup FV(U)$ is empty. Also we can assume that $\mathcal{I}(\phi) \cap \mathcal{V}_{\mathcal{T}} = \{A_1, \ldots, A_m\}$ holds. We can choose a set of type variables $\{B_1, \ldots, B_m\}$ such that the intersection with $Z \cup \mathcal{I}(\phi) \cup FV(U)$ is empty. Let

$$
\begin{array}{rcl}
\zeta_1 & = & \{A_1/B_1, \ldots, A_m/B_m\} \\
\rho_1 & = & \langle \zeta_1, \{\zeta_1(F_1)/\eta[\zeta_1(G_1)], \ldots, \zeta_1(F_n)/\eta[\zeta_1(G_n)]\}\rangle \\
\zeta_2 & = & \{B_1/A_1, \ldots, B_m/A_m\} \\
\rho_2 & = & \langle \zeta_2, \{G_1/\eta[F_1], \ldots, G_n/\eta[F_n]\}\rangle \\
\theta' & = & (\rho_1 \circ \phi) \restriction_{FV(S)} \\
\theta & = & \theta'{\downarrow_{\beta\eta}}
\end{array}
$$

We will show now that the substitution θ satisfies conditions **(1)**-**(3)**.

(1): For every $M \stackrel{?}{=} N \in S$ we have $\phi(M){\downarrow} = \phi(N){\downarrow}$ and for every term P we have $\theta'(P) \stackrel{*}{\longleftrightarrow}_{\beta\eta} \theta(P)$. Therefore

$$
\begin{array}{rcl}
\theta(M) & \stackrel{*}{\longleftrightarrow}_{\beta\eta} & \theta'(M) \\
 & = & \rho_1(\phi(M)) \\
 & \stackrel{*}{\longrightarrow}_{\beta\eta} & \rho_1(\phi(M){\downarrow}) \quad = \quad \rho_1(\phi(N){\downarrow}) \\
 & & \stackrel{*}{\longleftarrow}_{\beta\eta} \ \rho_1(\phi(N)) \\
 & & = \quad \theta'(N) \\
 & & \stackrel{*}{\longleftrightarrow}_{\beta\eta} \theta(N)
\end{array}
$$

Thus θ is an element of $\mathrm{SU}(U)$.

(2): Because

$$\theta =_{\beta\eta} \rho_1 \circ \phi\,[FV(U)]$$

we must have

$$\phi \leq_{\beta\eta} \theta\,[FV(U)].$$

Since $\rho_2 \circ \rho_1 =_{\beta\eta} \iota\,[FV(S) \cup \mathcal{I}(\phi)]$ holds, we have

$$\phi =_{\beta\eta} \rho_2 \circ \rho_1 \circ \phi\,[FV(U) \cup \mathcal{I}(\phi)].$$

Using $\theta =_{\beta\eta} \rho_1 \circ \phi\,[FV(U)]$ we get

$$\phi =_{\beta\eta} \rho_2 \circ \theta\,[FV(U)], \text{ and}$$

therefore

$$\theta \leq_{\beta\eta} \phi\,[FV(U)].$$

(3): is implied by the construction of θ.

Lemma 7.10 *Let $U = \langle \sigma, \{F_1 \stackrel{?}{=} M_1, \ldots, F_n \stackrel{?}{=} M_n\}\rangle$ be a unification problem in solved form. Then the set $\{\lceil U \rceil^{\text{SUB}}\}$ is a $\text{CSU}(U)[Z]$ for any finite set Z such that $Z \cap FV(U) = \emptyset$.*

Proof: We have to show that $\{\lceil S \rceil^{\text{SUB}}\}$ satisfies **(1)-(3)** of definition 3.3.

(1): Trivially, $\{\lceil U \rceil^{\text{SUB}}\} \subseteq \text{SU}(U)$.

(2): We must prove that for every normalized substitution $\theta \in \text{SU}(U)$ there exists a substitution in $\sigma \in \{\lceil U \rceil^{\text{SUB}}\}$ such that $\sigma \leq_\beta \theta\,[FV(U)]$ holds.
If $\theta \in \text{SU}(U)$ then $\theta =_\beta \theta \circ \lceil U \rceil^{\text{SUB}}$ because

$$\theta(F_i) \stackrel{*}{\longleftrightarrow}_\beta \theta(M_i) = \theta(\lceil U \rceil^{\text{SUB}}(F_i))$$

for every i, $1 \leq i \leq n$, and

$$\theta(F_i) = \theta(\lceil U \rceil^{\text{SUB}}(F_i)),$$

otherwise. For any type variable A we have

$$\theta(A) = \rho(\sigma(A))$$

for some type substitution ρ because θ is an instance of σ. Therefore $\lceil U \rceil^{\text{SUB}} \leq_\beta \theta$ and $\lceil U \rceil^{\text{SUB}} \leq_\beta \theta\,[FV(U)]$.

(3): We have to show that $\mathcal{DOM}(\phi) \subseteq FV(U)$, and $\mathcal{I}(\phi) \cap (Z \cup \mathcal{DOM}(\phi)) = \emptyset$ for all $\phi \in \{\lceil U \rceil^{\text{SUB}}\}$.
This is satisfied because $\mathcal{DOM}(\lceil U \rceil^{\text{SUB}}) = FV(U)$ and $\lceil U \rceil^{\text{SUB}}$ is idempotent.

Lemma 7.11 *If $M = \lambda\,\overline{x_n\colon A_n}.\, a(\overline{s_m})$ is any term of type $(\overline{A_n} \to A_0)$ then there exists a variant of a partial binding P and a substitution ρ such that $\rho(P) \stackrel{*}{\longrightarrow}_\beta M$.*

Proof: We distinguish the two cases $m = 0$ and $m > 0$.

$m = 0$**:** M has the form $\lambda\,\overline{x_n\colon A_n}.\, a$. According to the definition of partial bindings M is itself one. So choose $P = M$ and $\rho = \iota$.

$m > 0$**:** Let

$$P = \eta[\lambda\,\overline{x_n\colon A_n}.\, a(\overline{H_m(\overline{x_n})})]$$

be a partial binding appropriate for type $(A_1, \ldots, A_n \to A_0)$. ρ is the substitution

$$\langle \iota_1, \{H_1/\lambda\,\overline{x_n\colon A_n}.\, M_1, \ldots, H_m/\lambda\,\overline{x_n\colon A_n}.\, M_m\}\rangle.$$

Then

$$\rho(P) = \lambda\,\overline{x_n\colon A_n}.\, a(\overline{(\lambda\,\overline{x_n\colon A_n}.\, M_m)(\overline{x_n})}) \stackrel{+}{\longrightarrow}_\beta \lambda\,\overline{x_n\colon A_n}.\, a(M_1, \ldots, M_m)$$

as required.

So the term P and the substitution ρ have the desired properties.

Lemma 7.12 *If $\theta = \{F/M\} \cup \theta'$ then there exists a variant of a partial binding P appropriate to F and a substitution ρ such that*

$$\begin{aligned}\theta &= \{F/M\} \cup \rho \cup \theta' \,[\mathcal{DOM}(\theta)] \\ &=_\beta \rho \circ \{F/P\} \cup \theta' \,[\mathcal{DOM}(\theta)].\end{aligned}$$

Furthermore, if $\mathcal{DOM}(\theta) \cap \mathcal{I}(\theta) = \emptyset$ then $\theta'' = \{F/M\} \cup \rho \cup \theta'$ is a unifier of the equation $F \stackrel{?}{=} P$ and $\mathcal{DOM}(\theta'') \cap \mathcal{I}(\theta'') = \emptyset$.

Proof: Define P and ρ according to lemma 7.11. Therefore $\mathcal{DOM}(\rho) \cap \mathcal{DOM}(\theta) = \emptyset$ and $\rho(P) \stackrel{*}{\longrightarrow}_\beta M$. Thus

$$\{F/M\} = \{F/M\} \cup \rho =_\beta \rho \circ \{F/P\} \,[\mathcal{DOM}(\theta)].$$

Assume $\mathcal{DOM}(\theta) \cap \mathcal{I}(\theta) = \emptyset$. Because P is a variant we also have $\mathcal{DOM}(\rho) \cap \mathcal{I}(\theta) = \emptyset$ which implies $(\mathcal{DOM}(\theta) \cup \mathcal{DOM}(\rho)) \cap \mathcal{I}(\theta) = \emptyset$.

Because $\mathcal{DOM}(\theta) \cup \mathcal{DOM}(\rho) = \mathcal{DOM}(\theta'')$ and $\mathcal{I}(\theta) = \mathcal{I}(\theta'')$ we have $\mathcal{DOM}(\theta'') \cap \mathcal{I}(\theta'') = \emptyset$.

Finally, we show that θ'' is a unifier of F and P:

$$\theta''(P) = (\{F/M\} \cup \rho \cup \theta')(P) = \rho(P) \stackrel{*}{\longrightarrow}_\beta M = (\{F/M\} \cup \rho \cup \theta')(F) = \theta''(F).$$

Note that a can be a variable. If $\mathcal{DOM}(\theta) \cap \mathcal{I}(\theta) \neq \emptyset$ then $\theta = \{F/M, a/N\} \cup \theta'$ is possible. But in this case we have $\theta''(t) = (\{F/M\} \cup \rho \cup \{a/N\} \cup \theta')(P) \neq \rho(P)$. Therefore the condition $\mathcal{DOM}(\theta) \cap \mathcal{I}(\theta) = \emptyset$ is necessary.

These lemmas show that using partial bindings we can build arbitrary terms in an incremental way.

We define a transformation system $\mathcal{CPT}$ in which the applications of the transformation rules is controlled by a substitution θ. It is the goal of the transformations to get from the original unification problem U to a unification problem U' in solved form such that $\lceil U' \rceil^{\text{SUB}} \leq_\beta \theta$ holds.

Definition 7.13 (Transformation system $\mathcal{CPT}$)
With the following rules we define a transformation system on ordered pairs of normalized substitutions and unification problems.

Trivial removal

$$\langle \Theta, \langle \sigma, \{M \stackrel{?}{=} M\} \cup S \rangle \rangle \Longrightarrow \langle \Theta, \langle \sigma, S \rangle \rangle \qquad \mathcal{CPT}_1$$

Term type unification

$$\begin{array}{c} \langle \Theta, \langle \sigma, \{M \stackrel{?}{=} N\} \cup S \rangle \rangle \\ \Downarrow \\ \langle \Theta, \langle \theta \circ \sigma, \theta(\{M \stackrel{?}{=} N\} \cup S) \rangle \rangle \end{array} \qquad \mathcal{CPT}_2$$

where

- M is a term of type B and N is term of type D,
- $B \neq D$, and
- $\theta = \text{mgu}(\{B \stackrel{?}{=} D\})$.

Head type unification

$$\begin{array}{c} \langle \Theta, \langle \sigma, \{\lambda \overline{x_k\colon A_k}.\, a\colon B(\overline{M_m}) \stackrel{?}{=} \lambda \overline{x_k\colon A_k}.\, a\colon D(\overline{N_m})\} \cup S \rangle \rangle \\ \Downarrow \\ \langle \Theta, \langle \theta \circ \sigma, \theta(\{\lambda \overline{x_k\colon A_k}.\, a\colon B(\overline{M_m}) \stackrel{?}{=} \lambda \overline{x_k\colon A_k}.\, a\colon D(\overline{N_m})\} \cup S) \rangle \rangle \end{array} \qquad \mathcal{CPT}_3$$

where

- $B \neq D$,
- $\Theta_1(B) = \Theta_1(D)$, and
- $\theta = \mathrm{mgu}(\{B \stackrel{?}{=} D\})$.

Decomposition

$$\begin{array}{c}\langle\Theta, \langle\sigma, \{\lambda\overline{x_k\colon A_k}.\, a(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, a(\overline{N_m})\} \cup S\rangle\rangle \\ \Downarrow \\ \langle\Theta, \langle\sigma, \cup_{1\leq i\leq m}\{\lambda\overline{x_k\colon A_k}.\, M_i \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, N_i\} \cup S\rangle\rangle,\end{array} \qquad \mathcal{CPT}_4$$

where a is some arbitrary atom.

Variable elimination

$$\begin{array}{c}\langle\Theta, \langle\sigma, \{\lambda\overline{x_k\colon A_k}.\, F(\overline{x_k}) \stackrel{?}{=} N\} \cup S\rangle\rangle \\ \Downarrow \\ \langle\Theta, \langle\sigma, \{\lambda\overline{x_k\colon A_k}.\, F(\overline{x_k}) \stackrel{?}{=} N\} \cup \{F/N\}(S)\downarrow\rangle\rangle,\end{array} \qquad \mathcal{CPT}_5$$

where

- F is a free variable,
- $F \notin FV(N)$ and $F \in FV(S)$,
- $\mathrm{type}(F) = \mathrm{type}(N)$, and
- F and $\mathrm{head}(N)$ are not in $\mathcal{DOM}(\Theta)$.

Partial binding

$$\begin{array}{c}\langle\{F/Q\} \cup \Phi, \langle\sigma, \{\lambda\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, N\} \cup S\rangle\rangle \\ \Downarrow \\ \langle\{F/Q\} \cup \rho \cup \Phi, \langle\tau \circ \sigma, \{F \stackrel{?}{=} P, \eta[\{F/P\}(\tau(\lambda\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, N) \cup S)]\}\rangle\rangle\end{array} \qquad \mathcal{CPT}_6$$

where

- F is a free variable of type $(\overline{B_m} \to B_0)$ which is not solved in $\{\lambda\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, N\} \cup S$,
- $Q = \lambda\overline{y_l\colon B'_l}.\, a\colon D(\overline{Q_p})$ with some terms $Q_1, \ldots, Q_p$ and $\Phi(\overline{B_m} \to B_0) = (\overline{B'_l} \to B'_0)$ where a is some atom of type $D = (\overline{D_p} \to B'_0)$,
- $\tau = \mathrm{mgu}(\{B_0 \stackrel{?}{=} (B_{m+1}, \ldots, B_l \to C_0)\})$ for some type variables $B_{m+1}, \ldots, B_l$, and C_0,
- $P = \eta[\tau(\lambda\overline{y_l\colon B_l}.\, a\colon C(\overline{H_p\colon (\overline{B_l} \to C_i)(\overline{y_l})}))]$ is a partial binding appropriate for F with the same head a as Q of type $C = (\overline{C_p} \to C_0)$ and free variables $H_i\colon (\overline{B_l} \to C_i)$ for all i, $1 \leq i \leq p$, and
- $\rho = \langle \mathrm{mgu}(\bigcup_{1\leq i\leq p}\{\tau(\overline{B_l} \to C_i) \stackrel{?}{=} D_i\}), \{H_1\colon D_1/\lambda\overline{y_l\colon B'_l}.\, Q_1, \ldots, H_p\colon D_p/\lambda\overline{y_l\colon B'_l}.\, Q_p\}\rangle$.

△

Lemma 7.14 *If* $\Theta_1 \in \mathrm{SU}(U_1)$ *for some unification problem* $\langle\sigma_1, S_1\rangle$ *and*

$$\langle\Theta_1, \langle\sigma_1, S_1\rangle\rangle \Longrightarrow_{\mathcal{CPT}} \langle\Theta_2, \langle\sigma_2, S_2\rangle\rangle$$

then there exists a corresponding sequence of $\mathcal{HPT}$*-transformations*

$$\langle\sigma_1, S_1\rangle \stackrel{+}{\Longrightarrow}_{\mathcal{HPT}} \langle\sigma_2, S_2\rangle.$$

Proof: The only problematic transformation rule is $\mathcal{CPT}_6$. Suppose F is a free variable of type $(\overline{B_m} \to B_0)$ and the partial binding substitution is $\langle\tau, \{\tau(F)/P\}\rangle$ with

$$\tau \;=\; \mathrm{mgu}(\{B_0 \stackrel{?}{=} (B_{m+1}, \ldots, B_l \to C_0)\})$$

and

$$P \;=\; \eta[\tau(\lambda\overline{y_l\colon B_l}.\, a\colon C(\overline{H_p(\overline{y_l})}))].$$

We define the substitutions $\theta_1, \ldots, \theta_{l-m}$, $\gamma_1, \ldots, \gamma_p$ and δ in the following way:

$$\begin{aligned}
\theta_1 &= \{B_0/(B_{m+1} \to F_1)\}\\
\theta_2 &= \{F_1/(B_{m+2} \to F_2)\}\\
&\vdots\\
\theta_{l-m} &= \{F_{l-(m+1)}/(B_l \to C_0)\}\\
\gamma_1 &= \{\theta_{l-m} \circ \ldots \circ \theta_1(F)/\lambda\overline{y_l\colon B_l}.\, G_1(\overline{y_l}, H_1(\overline{y_l}))\}\\
\gamma_2 &= \{G_1/\lambda\overline{y_l\colon B_l'}, z_1\colon C_1.\, G_2(\overline{y_l}, z_1, H_2(\overline{y_l}))\}\\
&\vdots\\
\gamma_p &= \{G_{p-1}/\lambda\overline{y_l\colon B_l}, \overline{z_p\colon C_p}.\, G_p(\overline{y_l}, H_p(\overline{y_l}))\}\\
\delta &= \{G_p/\lambda\overline{y_l\colon B_l}, \overline{z_p\colon C_p}.\, a\colon C(\overline{z_p})\}.
\end{aligned}$$

It is easy to see that $\theta_{l-m}\circ\cdots\circ\theta_1 = \tau\,[FV^{\mathrm{Type}}(\{B_0, B_{m+1}, \ldots, B_l, C_0\})]$ and $\delta\circ\gamma_m\circ\cdots\circ\gamma_1 = \{\tau(F)/P\}\,[\{F\}]$. Now define the systems $S_1, \ldots, S_{l-m+p+3}$ as follows:

$$\begin{aligned}
S_1 &= \{\lambda\overline{x_k\colon A_k}.\, F(\overline{M_m}) \stackrel{?}{=} \lambda\overline{x_k\colon A_k}.\, N\} \cup S\\
S_2 &= \eta[\theta_1(S_1)]\\
S_3 &= \eta[\theta_2(S_2)]\\
&\vdots\\
S_{l-m+1} &= \eta[\theta_{l-m}(S_{l-m})]\\
S_{l-m+2} &= \{\eta[F] \stackrel{?}{=} \gamma_1(\eta[F])\} \cup \gamma_1(S_{m-l+1})\downarrow\\
S_{l-m+3} &= \{G_1 \stackrel{?}{=} \gamma_2(G_1)\} \cup \gamma_2(S_{m-l+2})\downarrow\\
&\vdots\\
S_{l-m+p+2} &= \{G_{p-1} \stackrel{?}{=} \gamma_p(G_{p-1})\} \cup \gamma_p(S_{l-m+p+1})\downarrow\\
S_{l-m+p+3} &= \{G_p \stackrel{?}{=} \delta(G_p)\} \cup \delta(S_{l-m+p+2})\downarrow.
\end{aligned}$$

Then we get following transformation sequence:

$$\begin{aligned}
\langle\sigma_1, S_1\rangle &\Longrightarrow_{\mathcal{HPT}_{6a}} \langle\theta_1\circ\sigma_1, S_2\rangle\\
&\Longrightarrow_{\mathcal{HPT}_{6a}} \langle\theta_2\circ\theta_1\circ\sigma_1, S_3\rangle\\
&\quad\vdots\\
&\Longrightarrow_{\mathcal{HPT}_{6a}} \langle\theta_{l-m-1}\circ\cdots\circ\theta_1\circ\sigma_1, S_{l-m}\rangle\\
&\Longrightarrow_{\mathcal{HPT}_{6a}} \langle\tau\circ\sigma_1, S_{l-m+1}\rangle\\
&\Longrightarrow_{\mathcal{HPT}_{6b}} \langle\tau\circ\sigma_1, S_{l-m+2}\rangle\\
&\Longrightarrow_{\mathcal{HPT}_{6b}} \langle\tau\circ\sigma_1, S_{l-m+3}\rangle\\
&\quad\vdots\\
&\Longrightarrow_{\mathcal{HPT}_{6b}} \langle\tau\circ\sigma_1, S_{l-m+p+1}\rangle\\
&\Longrightarrow_{\mathcal{HPT}_{6b}} \langle\tau\circ\sigma_1, S_{l-m+p+2}\rangle\\
&\Longrightarrow_{\mathcal{HPT}_{7}} \langle\tau\circ\sigma_1, S_{l-m+p+3}\rangle
\end{aligned}$$

and $\langle \tau \circ \sigma_1, S_{l-m+p+3} \rangle = \langle \sigma_2, S_2 \rangle$ holds.

Lemma 7.15 *If $\Theta \in \mathrm{SU}(\langle \sigma, S \rangle)$ for some unification problem $U = \langle \sigma, S \rangle$ not in solved form then there exists a transformation*

$$\langle \Theta, \langle \sigma, S \rangle \rangle \Longrightarrow_{\mathcal{CPT}} \langle \Theta', \langle \sigma', S' \rangle \rangle,$$

such that

(1) $\Theta = \Theta'\,[W]$ *with W a set of variables,*

(2) *If $\mathcal{DOM}(\Theta) \cap \mathcal{I}(\Theta) = \emptyset$, then $\Theta' \in \mathrm{SU}(S')$ and $\mathcal{DOM}(\Theta') \cap \mathcal{I}(\Theta') = \emptyset$, and*

(3) $\langle \sigma, S \rangle \stackrel{+}{\Longrightarrow}_{\mathcal{HPT}} \langle \sigma', S' \rangle$.

Proof: Because U is not in solved form there is a pair $M \stackrel{?}{=} N$ which is not solved in U. We distinguish the following cases:

1. $M = N$. Then we may apply the transformation rules $\mathcal{CPT}_1$ or $\mathcal{CPT}_4$.
2. $\mathrm{type}(M) \neq \mathrm{type}(N)$. Then we can apply $\mathcal{CPT}_2$. In the remaining cases we may assume $\mathrm{type}(M) = \mathrm{type}(N)$.
3. $\mathrm{head}(M) = a\colon B$ and $\mathrm{head}(N) = a\colon D$ for some arbitrary atoms $a\colon B$ and $a\colon D$ such that $B \neq D$. We can apply $\mathcal{CPT}_3$.
4. $\mathrm{head}(M) = \mathrm{head}(N) \notin \mathcal{DOM}(\Theta)$. Then we can apply $\mathcal{CPT}_4$.
5. $M \neq N$, $\mathrm{head}(M) = a\colon B$, $\mathrm{head}(N) = b\colon D$, and one of the following two cases applies.
   (a) $\Theta_1(\mathrm{head}(M)) \neq \Theta_1(\mathrm{head}(N))$: Because there exists a unifier of S, not both M and N can be rigid. We can assume that M is a flexible term.
   (b) $\Theta_1(\mathrm{head}(M)) = \Theta_1(\mathrm{head}(N)) \in \mathcal{DOM}(\Theta)$: One of the terms is flexible too. Let's assume that M is flexible.

   Therefore
   $$M = \lambda\, \overline{x_k\colon A_k}.\, F(\overline{M_m})$$
   and
   $$N = \lambda\, \overline{x_k\colon A_k}.\, N'.$$
   Then we can apply $\mathcal{CPT}_6$ or if $M \stackrel{*}{\longrightarrow}_\eta F$ $\mathcal{CPT}_5$ is applicable too if it's application constraints are fulfilled.

Thus there exists a transformation

$$\langle \Theta, \langle \sigma, S \rangle \rangle \stackrel{+}{\Longrightarrow}_{\mathcal{CPT}} \langle \Theta', \langle \sigma', S' \rangle \rangle.$$

If we apply one of the transformation rules $\mathcal{CPT}_1, \ldots, \mathcal{CPT}_5$, the conditions hold because

(1) $\Theta = \Theta'$,

(2) of the correctness lemma.

(3) of the definition of $\mathcal{CPT}$.

If we apply the transformation rule $\mathcal{CPT}_6$, we can assume that

$$\Theta = \{F/Q\} \cup \Phi$$

and

$$\begin{aligned} \Theta' &= \{F/Q\} \cup \rho \cup \Phi =_\beta \rho \circ \{F/P\} \cup \Phi \\ S' &= \{F \stackrel{?}{=} P\} \cup \{F/P\}(\tau(S)) \\ \sigma' &= \tau \circ \sigma \end{aligned}$$

holds for partial binding P and substitutions ρ and τ, as given in the definition of $\mathcal{CPT}_6$.

Therefore the conditions are satisfied

(1) because of the construction of Θ' and lemma 7.12.

(2) If we assume $\mathcal{DOM}(\theta') \cap \mathcal{I}(\theta') = \emptyset$ then following lemma 7.12

$$\mathcal{DOM}(\Theta') \cap \mathcal{I}(\Theta') = \emptyset$$

and

$$\Theta'(P) = \rho(q) \xrightarrow{*}_\beta Q = \Theta'(F)$$

holds.

(3) because of lemma 7.14.

Corollary 7.16 *If $\theta \in \mathrm{SU}(U)$ and no transformation rule in $\mathcal{CPT}$ is applicable to $\langle \theta, U\rangle$, then U is in solved form.*

Theorem 7.17 (Completeness of $\mathcal{HPT}$) *Let U be a unification problem. If $\Theta \in \mathrm{SU}(U)$, then there exists a sequence of transformations*

$$U = U_0 \Longrightarrow_{\mathcal{HPT}} U_1 \Longrightarrow_{\mathcal{HPT}} U_2 \Longrightarrow_{\mathcal{HPT}} \cdots \Longrightarrow_{\mathcal{HPT}} U_n,$$

where U_n is in solved form and $\lceil U_n \rceil^{\mathrm{SUB}} \leq_\beta \Theta\,[FV(U)]$.

Proof: $\mathcal{DOM}(\Theta) \cap \mathcal{I}(\Theta) = \emptyset$ because of lemma 7.9. We define a complexity measure $\mu(\langle\langle\Theta_1, \Theta_2\rangle, \langle\sigma, S\rangle\rangle) = \langle l, m, n\rangle$ by

$$\begin{aligned} l &= \textstyle\sum_{x \in \mathcal{DOM}(\Theta_2) \setminus \mathrm{Solved}(S)} |\Theta_2(x)|^{\mathrm{Term}} \\ m &= \textstyle\sum_{(M \stackrel{?}{=} N) \in S} |M|^{\mathrm{Term}} + |N|^{\mathrm{Term}} \\ n &= \textstyle\sum_{x \in \mathcal{DOM}(\Theta_1)} |\Theta_1(x)|^{\mathrm{Type}} - |\sigma(x)|^{\mathrm{Type}}. \end{aligned}$$

Using lexicographic combination of the $<$-ordering on natural number on every component we get a notherian ordering $<$ on this measure. For every $\mathcal{CPT}$-transformation $U \Longrightarrow_{\mathcal{CPT}} U'$ we get $\mu(U') < \mu(U)$:

$\mathcal{CPT}_1$ reduces m without changing l.

$\mathcal{CPT}_2$ reduces n without changing l and m. Since $\theta \neq \iota$ and $|\sigma(x)|^{\mathrm{Type}} \leq |\theta \circ \sigma(x)|^{\mathrm{Type}} \leq |\Theta_1(x)|^{\mathrm{Type}}$ for all $x \in \mathcal{DOM}(\Theta_1)$.

$\mathcal{CPT}_3$ reduces n without changing l and m for the same reason as for $\mathcal{CPT}_2$.

$\mathcal{CPT}_4$ reduces m without changing l.

$\mathcal{CPT}_5$ reduces l, because a variable out of $\mathcal{DOM}(\Theta) \setminus \mathrm{Solved}(U)$ is erased.

$\mathcal{CPT}_6$ reduces l: The binding

$$\{F/\lambda\,\overline{y_l\colon B'_l}.\, a\colon D(\overline{Q_p})\}$$

does not increase l because F is solved in U', Rather

$$\langle \mathrm{mgu}(\bigcup_{1 \leq i \leq p} \{\tau(\overline{B_l} \to C_i) \stackrel{?}{=} D_i\}), \{H_1\colon D_1/\lambda\,\overline{y_l\colon B'_l}.\, Q_1, \ldots, H_p\colon D_p/\lambda\,\overline{y_l\colon B'_l}.\, Q_p\}\rangle.$$

is inserted and a is removed. In all l has decreased.

Hence every sequence of $\mathcal{CPT}$-transformations is finite. Therefore there exists a sequence

$$\langle \Theta, U\rangle = \langle \Theta_0, U_0\rangle \stackrel{+}{\Longrightarrow}_{\mathcal{CPT}} \langle \Theta_k, U_k\rangle,$$

such that no further transformations are applicable, and by induction over k using lemma 7.15 with $W = FV(S)$ we get $\Theta = \Theta_k\,[W]$ and $\Theta_k \in \mathrm{SU}(U_k)$. Due to lemma 7.14 there exists a corresponding sequence of $\mathcal{HPT}$-transformations

$$U = U_0 \stackrel{+}{\Longrightarrow}_{\mathcal{HPT}} U_p,$$

where as in Corollary 7.16 U_p is in solved form. Applying lemma 7.10 we get

$$\lceil U_p \rceil^{\mathrm{SUB}} \leq_\beta \Theta_k = \Theta\,[FV(U)].$$

Theorem 7.18 *For every unification problem U there exists a set*

$$G = \{\lceil U' \rceil^{\mathrm{SUB}}|_{FV(U)} \mid U \stackrel{+}{\Longrightarrow}_{\mathcal{HPT}} U' \text{ and } U' \text{ is in solved form}\}$$

which is a $\mathrm{CSU}(U)$. *Using some renaming substitution away from W this set is a* $\mathrm{CSU}(U)[W]$ *for arbitrary W.*

Proof: We have to check the conditions of Definition 3.3:

(1) $G \subseteq \mathrm{SU}(S)$ follows from the correctness of $\mathcal{HPT}$.

(2) The fact that for any normalized substitution $\theta \in \mathrm{SU}(U)$ there is substitution $\sigma \in G$, such that $\sigma \leq_\beta \theta\,[FV(U)]$, follows from the completeness of $\mathcal{HPT}$.

(3) The condition $\mathcal{DOM}(\sigma) \subseteq FV(U)$ follows from the construction of the substitutions in G. The condition $\mathcal{I}(\sigma) \cap (Z \cup \mathcal{DOM}(\sigma)) = \emptyset$ can be satisfied by appropriate renaming.

8 Future Work

The transformation system $\mathcal{HPT}$ introduced in this paper is based on the transformation system **UT** of Dougherty (1991) for unification in the polymorphically typed combinatory logic. In a forthcoming paper I present a detailed investigation of the relationship between $\mathcal{HPT}$ and **UT**.

It is not difficult to see that the search space of $\mathcal{HPT}$ is finitely branching. The search space of $\mathcal{HT}$, in contrast, is infinitely branching. However, although we have transformed the infinitely branching search space into a finitely branching one, (unfortunately) we have not reduced the search space. To this end the idea in Nipkow (1990) of introducing product types should be investigated. This amounts to developing a correct and complete unification algorithm for the λ-calculus with product types.

References

Alonzo Church, 1932. A Set of Postulates for the Foundation of Logic. *Annals of Mathematics*, Vol. 33, pp. 346–366.

Daniel J. Dougherty, 1991. Higher-Order Unificaiton via Combinators. Preprint.

M. Gordon, 1985. HOL: A Machine Oriented Formulation of Higher Order Logic. Technical Report 68, Computer Laboratory, University of Cambridge, Cambridge, England.

G.P. Huet, 1975. A Unification Algorithm for typed λ-Calculus. *Theoretical Computer Science*, Vol. 1, pp. 27–57.

S. Kaplan and M. Okada, editors, 1990. *Proceedings of the Second International Workshop on Conditional and Typed Rewriting Systems*, LNCS, vol. 516, Montreal, Canada. Springer-Verlag.

A. Martelli and U. Montanari, 1982. An Efficient Unification Algorithm. *ACM Transactions on Programming Languages and Systems*, Vol. 4, No. 2, pp. 258–292.

D. Miller and G. Nadathur, 1986. Higher-Order Logic Programming. In Shapiro (1986), pp. 448–462.

Tobias Nipkow, 1990. Higher-Order Unification, Polymorphism and Subsorts. In Kaplan and Okada (1990), pp. 229–237.

L. C. Paulson and T. Nipkow, 1990. Isabelle Tutorial and User's manual. Technical Report 189, Computer Laboratory, University of Cambridge, Cambridge, England.

Frank Pfenning, 1989. Elf: A language for logic definition and verifieed meta-programming. In *Fourth Annual Sumposium on Logic in Computer Science*, pp. 313–322. IEEE.

E. Shapiro, editor, 1986. *Proceedings of the Third International Logic Programming Conference*, LNCS, vol. 225, London. Springer-Verlag.

W. Snyder and J. Gallier, 1989. Higher-Order Unification Revisited: Complete Sets of Transformations. *Journal of Symbolic Computation*, Vol. 8, pp. 101–140.