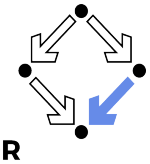
RISC



RESEARCH INSTITUTE FOR
SYMBOLIC COMPUTATION

JKU

JOHANNES KEPLER
UNIVERSITY LINZ

Higher-Order Pattern Unification Modulo Similarity Relations

Besik Dundua, Temur Kutsia

February 2025

RISC Report Series No. 25-03

ISSN: 2791-4267 (online)

Available at https://doi.org/10.35011/risc.25-03

JOHANNES KEPLER
UNIVERSITY LINZ
Altenberger Str. 69
4040 Linz, Austria
www.jku.at
DVR 0093696

Higher-Order Pattern Unification Modulo Similarity Relations

Besik Dundua[1] and Temur Kutsia[2]

[1] VIAM, Tbilisi State University, Georgia and Kutaisi International University, Georgia
[2] Research Institute for Symbolic Computation, Johannes Kepler University Linz, Austria

Abstract. The combination of higher-order theories and fuzzy logic can be useful in decision-making tasks that involve reasoning across abstract functions and predicates, where exact matches are often rare or unnecessary. Developing efficient reasoning and computational techniques for such a combined formalism presents a significant challenge. In this paper, we adopt a more straightforward approach aiming at integrating two well-established and computationally well-behaving components: higher-order patterns on one side and fuzzy equivalences expressed through similarity relations based on minimum T-norm on the other. We propose a unification algorithm for higher-order patterns modulo these similarity relations and prove its termination, soundness, and completeness. This unification problem, like its crisp counterpart, is unitary. The algorithm computes the most general unifier with the highest degree of approximation when the given terms are unifiable.

1 Introduction

Approximate reasoning involves making decisions or performing inferences based on vague or imprecise information, which is often modeled using fuzzy logic. Fuzzy similarity (and proximity) relations are key tools in this type of reasoning, allowing for working with uncertain or imprecise data. Fuzzy similarity refers to the degree to which two elements or objects are alike, using a value in the range of 0 to 1, where 0 represents no similarity and 1 represents complete similarity. These relations can be used to determine how similar a new situation is to past experiences or data. This is particularly useful in scenarios where precise comparisons aren't possible or where "similar enough" is acceptable. (See, e.g., [7, 12, 17, 19, 22, 27, 29, 30] in the context of using similarity relations in automated approximate reasoning.)

Fuzzy similarity and proximity relations have been incorporated into knowledge representation and inference processes in the area of logic programming [6, 8, 10, 11, 14, 16, 30], making it possible to reason or compute with vague concepts having imprecise boundaries. It helps deal with the ambiguity inherent in real-world data, where exact matches are rare, and allows systems to make reasonable inferences even with imprecise or incomplete information. In order this approach

to work, one needs a fundamental computational mechanism such as unification to make an "approximate inference step". Motivated by this application, first-order unification with fuzzy relations has been investigated both from theoretical and practical points of view, see, e.g., [1,7,9,13,15,17,18,21,27,28,30–32]. In [5], this technique was studied in a more general setting where the quantitative approximate information is specified using quantales, having the fuzzy quantale a special case.

The cited works focus on fuzzy reasoning and computation within a first-order framework. However, decision-making often requires higher-order reasoning across abstract layers of functions and predicates, where exact matches are rare or even unnecessary. For instance, when two functions are not identical but exhibit a high degree of similarity in behavior or output, fuzzy similarity relations would enable the system to recognize this resemblance and apply approximate reasoning for inference or prediction. This can be especially useful in fields such as natural language processing, knowledge representation, and complex decision-making systems, where human-like reasoning and the handling of uncertainty are essential. This idea is a primary motivation for our work.

It is important to note that the existing higher-order fuzzy logics (e.g., [20, 26]) are highly expressive formalisms, but because of this expressive power, developing efficient reasoning and computational techniques for them is a significant challenge. We adopt a more straightforward approach, concentrating on the integration of established and computationally well-behaving fragments: higher-order patterns on one hand, and fuzzy equivalences expressed through minimum T-norm-based similarity relations on the other. The first step in supporting reasoning for this integration is the development of fundamental computational techniques, such as unification, which is the subject of this paper.

Our contribution. We define the notion of similarity for simply typed lambda terms and study the unification problem for higher-order patterns à la Miller [23]. The equality relation modulo $\alpha\beta\eta$-equivalence is replaced by fuzzy similarity modulo $\alpha\beta\eta$ with the minimum T-norm. We develop a rule-based unification algorithm for such problems and prove its termination, soundness, and completeness. The unification problem, as its crisp counterpart, is unitary. The algorithm returns a most general pattern unifier of the given terms together with its approximation degree, which shows how similar the instances of the terms are with respect to the unifier. This degree should be at least the given fixed threshold (cut value). If the threshold is set to its maximal possible value 1, the algorithm computes the standard (crisp) unifiers. In this context, it can be seen as a rule-based version of the standard higher-order pattern unification algorithm.

2 Preliminaries

2.1 Term Language

Given a set of basic types, whose elements are denoted by δ, *simple types* are constructed using the grammar $\tau ::= \delta \mid \tau \to \tau$, where τ is associative to the

right. The alphabet of our language consists of the set $\mathcal{V}$ of variables and $\mathcal{F}$ of constants. They are disjoint and countably infinite and their elements have assigned types. Variables are denoted typically by $F, G, H, X, Y, Z, x, y, z, \ldots)$ and constants by $f, g, a, b, c, \ldots$. The set of *terms*, denoted by $\mathcal{T}(\mathcal{F}, \mathcal{V})$, is defined by the grammar $t ::= x \mid c \mid \lambda x.t \mid (t,\, s)$ where $(\lambda x.t)$ is called an *abstraction* and $(t\, s)$ is called an *application*. We denote terms by t, s, r.

The standard concepts of the simply typed λ-calculus, such as bound and free occurrences of variables, position in a term, α-conversion, β-reduction, and η-long β-normal form, are defined in the usual way, see, e.g., [2, 4]. The η-long β-normal form of a term t is denoted by $t\downarrow^{\eta}_{\beta}$. For any t, the term $t\downarrow^{\eta}_{\beta}$ has the form $\lambda x_1, \ldots, x_n.f(t_1, \ldots, t_m)$, where $n, m \geq 0$, f is either a constant or a variable, and each t_i (for $i = 1, \ldots, m$) follows the same structural form. Moreover, the term $f(t_1, \ldots, t_m)$ has a basic type. When we write an equality between two λ-terms, we mean that they are equivalent modulo α-, β-, and η-equivalence. The size of a term t, denoted $size(t)$, is defined recursively as $size(f(t_1, \ldots, t_n)) = 1 + \sum_{i=1}^{n} size(t_i)$ and $size(\lambda x.t) = 1 + size(t)$.

The sets of free and bound variables of a term t are denoted by $\mathtt{fv}(t)$ and $\mathtt{bv}(t)$ respectively. As a convention and for the sake of clarity, in what follows we distinguish bound and free variables syntactically. In particular, we use lowercase letters x, y, z for bound variables and capital letters X, Y, Z, F, G, H for free variables.

Definition 1 (Higher-order patterns). *A higher-order pattern is a term where, when written in η-long β-normal form, all free variable occurrences are applied to lists of pairwise distinct (η-long forms of) bound variables.*

A *substitution* σ is a mapping from variables to terms such that for each variable x of type τ, the term $\sigma(x)$ is of type τ, and all but finitely many variables are mapped to terms that are not equal to t (modulo $\alpha\beta\eta$). Each substitution σ is represented as a finite set of pairs $\{x_1 \mapsto \sigma(x_1), \ldots,\ x_n \mapsto \sigma(x_n)\}$ where the x's are variables for which $\sigma(x_i) \neq x_i$. The sets $Dom(\sigma) = \{x_1, \ldots, x_n\}$ and $Ran(\sigma) = \{\sigma(x_1), \ldots, \sigma(x_n)\}$ are called the *domain* and the *range* of σ, respectively. The set $\mathtt{fv}(\sigma)$ is defined as $\mathtt{fv}(\sigma) = Dom(\sigma) \cup \mathtt{fv}(Ran(\sigma))$ where $\mathtt{fv}(Ran(\sigma)) = \cup_{i=1}^{n} \mathtt{fv}(\sigma(x_i))$

The *application* of a substitution σ to t replaces each *free* occurrence of a variable x in t with $\sigma(x)$. It is defined inductively:

$$
\begin{aligned}
&x\sigma = \sigma(x), \text{ if } x \in Dom(\sigma).\\
&x\sigma = x, \text{ if } x \notin Dom(\sigma).\\
&f\sigma = f.\\
&(ts)\sigma = t\sigma\, s\sigma.\\
&(\lambda x.t)\sigma = \lambda x.t\sigma, \quad \text{if } x \notin \mathtt{fv}(\sigma).\\
&(\lambda x.t)\sigma = \lambda y.t\{x \mapsto y\}\sigma, \quad \text{if } x \in \mathtt{fv}(\sigma) \text{ and } y \text{ is fresh.}
\end{aligned}
$$

2.2 Fuzzy Relations

We define basic notions about fuzzy relations following [14, 30].

A binary *fuzzy relation* on a set S is a mapping from $S \times S$ to the real interval $[0, 1]$. If $\mathcal{R}$ is a fuzzy relation on S and μ is a number $0 < \mu \leq 1$ (called *cut value*), then the *μ-cut* of $\mathcal{R}$ on S, denoted $\mathcal{R}_\mu$, is an ordinary (crisp) relation on S defined as $\mathcal{R}_\mu := \{(s_1, s_2) \mid \mathcal{R}(s_1, s_2) \geq \mu\}$.

A T-norm $\wedge$ is an associative, commutative, non-decreasing binary operation on $[0, 1]$ with 1 as the unit element. Some of the most prominent T-norms are

- Gödel (or minimum) T-norm: $s_1 \wedge s_2 = \min(s_1, s_2)$,
- Product T-norm: $s_1 \wedge s_2 = s_1 * s_2$,
- Łukasiewicz T-norm: $s_1 \wedge s_2 = \max(0, s_1 + s_2 - 1)$.

Definition 2 (Similarity relation). *A fuzzy relation $\mathcal{R}$ on a set S is called a* proximity relation*, if it is reflexive ($\mathcal{R}(s, s) = 1$ for all $s \in S$) and symmetric ($\mathcal{R}(s_1, s_2) = \mathcal{R}(s_2, s_1)$ for all $s_1, s_2 \in S$). A proximity relation is called a* similarity relation *if it is $\wedge$-transitive: $\mathcal{R}(s_1, s_2) \geq \mathcal{R}(s_1, s) \wedge \mathcal{R}(s, s_2)$ for any $s_1, s_2, s \in S$.*

In the role of S, we take the set of terms of our language. First, we assume having a fuzzy relation $\mathcal{R}_A$ defined on the alphabet $\mathcal{F} \cup \mathcal{V}$ in such a way that

- $\mathcal{R}_A(x, y) = 0$ for all $x, y \in \mathcal{V}$ with $x \neq y$,
- $\mathcal{R}(f, g) = 0$ for all $f, g \in \mathcal{F}$ such that f and g have different types.
- $\mathcal{R}(x, f) = \mathcal{R}(f, x) = 0$ for all $x \in \mathcal{V}$ and $f \in \mathcal{F}$.

Definition 3 (Fuzzy relation on terms). *Given a fuzzy relation $\mathcal{R}_A$ on the alphabet $\mathcal{F} \cup \mathcal{V}$, we define a fuzzy relation $\mathcal{R}$ on the set of terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$ using the T-norm $\wedge$.*

1. *If t and s are in η-long β-normal form, then $\mathcal{R}(t, s)$ is defined as follows:*

$$\begin{aligned}
&\mathcal{R}(a, b) = \mathcal{R}_A(a, b), \text{ where } a, b \in \mathcal{F} \cup \mathcal{V},\\
&\mathcal{R}(t_1 s_1, t_2 s_2) = \mathcal{R}(t_1, t_2) \wedge \mathcal{R}(s_1, s_2),\\
&\mathcal{R}(\lambda x.t, \lambda y.s) = \mathcal{R}(t\{x \mapsto z\}, s\{y \mapsto z\}),\\
&\quad \text{where } x, y, \text{ and } z \text{ have the same type and } z \text{ is a fresh variable,}\\
&\mathcal{R}(t, s) = 0 \text{ otherwise.}
\end{aligned}$$

2. *Otherwise, $\mathcal{R}(t, s) = \mathcal{R}(t{\downarrow_\beta^\eta}{\uparrow^\eta}, s{\downarrow_\beta^\eta}{\uparrow^\eta})$.*

It is easy to see that if $\mathcal{R}_A$ is a proximity relation on the alphabet, then $\mathcal{R}$ is a proximity relation on terms for any T-norm. This definition also implies that if $\mathcal{R}(t, s) > 0$, then

- t and s have the same type,
- $Pos(t{\downarrow_\beta^\eta}{\uparrow^\eta}) = Pos(s{\downarrow_\beta^\eta}{\uparrow^\eta})$,
- for each $p \in Pos(t{\downarrow_\beta^\eta}{\uparrow^\eta})$, symbols occurring at position p in $t{\downarrow_\beta^\eta}{\uparrow^\eta}$ and $s{\downarrow_\beta^\eta}{\uparrow^\eta}$ have the same type,

- a variable (resp. a constant) occurs at position p in $t\downarrow_\beta^\eta$ iff a variable (resp. a constant) occurs at position p in $s\downarrow_\beta^\eta$,
- a free variable X occurs at position p in $t\downarrow_\beta^\eta$ iff the same variable X occurs at position p in $s\downarrow_\beta^\eta$.

Note that if the T-norm is not idempotent, one can find a substitution σ such that $\mathcal{R}(t,s) > 0$ does not imply $\mathcal{R}(t,s) = \mathcal{R}(t\sigma, s\sigma)$. For instance, for the product T-norm, $\mathcal{R}(a,b) = 0.5$, and $\sigma = \{X \mapsto \lambda x.f(x,x)\}$ we have

$$\mathcal{R}(X(a), X(b)) = 0.5,$$
$$\mathcal{R}(X(a)\sigma, X(b)\sigma) = \mathcal{R}(f(a,a), f(b,b)) = 0.5 * 0.5 = 0.25.$$

However, this problem does not arise when the T-norm is idempotent, or when terms are higher-order patterns.

Theorem 1. *If $\mathcal{R}_A$ is a similarity relation on the alphabet, then $\mathcal{R}$ is a similarity relation on terms for any T-norm.*

Proof. We only need to show that for arbitrary terms t, s, r in η-long β-normal form and a T-norm $\wedge$, the $\wedge$-transitivity property holds: $\mathcal{R}(t,r) \geq \mathcal{R}(t,s) \wedge \mathcal{R}(s,r)$. Assume without loss of generality that $\mathcal{R}(t,s) \wedge \mathcal{R}(s,r) > 0$. Then by the monotonicity of T-norms we have $\mathcal{R}(t,s) > 0$ and $\mathcal{R}(s,r) > 0$, which implies that t, s, and r have the same structure. We proceed by induction on this structure.

- When $\{t, s, r\} \in \mathcal{F} \cup \mathcal{V}$, the property follows from the assumption that $\mathcal{R}_A$ is a similarity.
- Let $t = t_1 t_2$, $s = s_1 s_2$, and $r = r_1 r_2$. By definition of $\mathcal{R}$, we have $\mathcal{R}(t,r) = \mathcal{R}(t_1,r_1) \wedge \mathcal{R}(t_2,r_2)$, $\mathcal{R}(t,s) = \mathcal{R}(t_1,s_1) \wedge \mathcal{R}(t_2,s_2)$, and $\mathcal{R}(s,r) = \mathcal{R}(s_1,r_1) \wedge \mathcal{R}(s_2,r_2)$. By the induction hypothesis, we know $\mathcal{R}(t_i,r_i) \geq \mathcal{R}(t_i,s_i) \wedge \mathcal{R}(s_i,r_i)$, $i = 1,2$. Then, by monotonicity of T-norms, we get $\mathcal{R}(t_1,r_1) \wedge \mathcal{R}(t_2,r_2) \geq \mathcal{R}(t_1,s_1) \wedge \mathcal{R}(s_1,r_1) \wedge \mathcal{R}(t_2,s_2) \wedge \mathcal{R}(s_2,r_2)$, which implies $\mathcal{R}(t_1 t_2, r_1 r_2) \geq \mathcal{R}(t_1 t_2, s_1 s_2) \wedge \mathcal{R}(s_1 s_2, r_1 r_2)$ for any $\wedge$.
- Let $t = \lambda x.t'$, $s = \lambda y.s'$, and $r = \lambda z.r'$, where x, y, z have the same type. By definition of $\mathcal{R}$, without loss of generality, we can choose a fresh variable u of the same type such that $\mathcal{R}(\lambda x.t', \lambda z.r') = \mathcal{R}(t'\{x \mapsto u\}, r'\{z \mapsto u\})$, $\mathcal{R}(\lambda x.t', \lambda y.s') = \mathcal{R}(t'\{x \mapsto u\}, s'\{y \mapsto u\})$, and $\mathcal{R}(\lambda y.s', \lambda z.r') = \mathcal{R}(s'\{y \mapsto u\}, r'\{z \mapsto u\})$. By the induction hypothesis, we have $\mathcal{R}(t'\{x \mapsto u\}, r'\{z \mapsto u\}) \geq \mathcal{R}(t'\{x \mapsto u\}, s'\{y \mapsto u\}) \wedge \mathcal{R}(s'\{y \mapsto u\}, r'\{z \mapsto u\})$ for any T-norm, which implies $\mathcal{R}(\lambda x.t', \lambda z.r') \geq \mathcal{R}(\lambda x.t', \lambda y.s') \wedge \mathcal{R}(\lambda y.s', \lambda z.r')$ for any T-norm.

Given a similarity relation $\mathcal{R}$, cut value μ, and two terms t, s, we write $t \simeq_{\mathcal{R},\mu} s$ if $\mathcal{R}(t,s) \geq \mu$. The number $\mathcal{R}(t,s)$ is called the $\mathcal{R}$-*similarity degree* of t and s.

We say σ is $(\mathcal{R},\mu)$-*more general* than ϑ with variable restriction V, written $\sigma \preceq^V_{\mathcal{R},\mu} \vartheta$, if there exists a substitution τ such that $x\sigma\tau \simeq_{\mathcal{R},\mu} x\vartheta$ for all $x \in V$.

Definition 4 (Unification problem, unifier, unification degree, mgu). *An* $(\mathcal{R},\mu)$-unification equation *between terms* t *and* s *is written as* $t \simeq^?_{\mathcal{R},\mu} s$, *where* $\mathcal{R}$ *is a similarity relation and* μ *is a cut value. An* $(\mathcal{R},\mu)$-unification problem *is a finite set of* $(\mathcal{R},\mu)$*-unification equations. A substitution* σ *is a* unifier *(or a solution) of an* $(\mathcal{R},\mu)$*-unification problem* $\{t_1 \simeq^?_{\mathcal{R},\mu} s_1, \ldots, t_1 \simeq^?_{\mathcal{R},\mu} s_1\}$ *with* unification degree $\mathfrak{d}$ *if* $\mathcal{R}(t_1\sigma, s_1\sigma) \wedge \cdots \wedge \mathcal{R}(t_n\sigma, s_n\sigma) = \mathfrak{d} \geq \mu$.

Given an $(\mathcal{R},\mu)$*-unification problem* P, *we say that its* $(\mathcal{R},\mu)$*-unifier* σ *is a* most general $(\mathcal{R},\mu)$-unifier *of* P *if for any* $(\mathcal{R},\mu)$*-unifier* ϑ *of* P *we have* $\sigma \preceq^{\mathtt{fv}(P)}_{\mathcal{R},\mu} \vartheta$.

We consider a special case of this problem: all terms that appear in the unification problems and unifiers should be higher-order patterns. In the rest of the paper, we make the following assumptions (those related to terms follow [25]):

- min: T-norm $\wedge$ is the minimum T-norm,
- α: α-equivalent terms are identified,
- β: terms are β-normalized by default,
- η: terms are in η-expanded form except for the arguments of free variables, which are in η-normal form, i.e., bound variables.

Example 1. Higher-order pattern $(\mathcal{R},\mu)$-unification problems may have most general unifiers that are or are not degree-maximal, and may have degree-maximal unifiers that are or are not most general. We illustrate these facts with an $(\mathcal{R},\mu)$-unification problem between $t = \lambda x.\lambda y.f(F(x), F(y))$ and $s = \lambda x.\lambda y.g(a(G(y,x)), b(G(x,y)))$, where $\mathcal{R}$ is defined as $\mathcal{R}(f,g) = 0.8$, $\mathcal{R}(a,b) = 0.6$, $\mathcal{R}(b,c) = \mathcal{R}(a,c) = 0.5$, and the cut value is $\mu = 0.4$. Consider the following $(\mathcal{R},\mu)$-unifiers of t and s together with their unification degrees:

$$\begin{aligned}
\sigma_1 &= \{F \mapsto \lambda x.a(H(x)), G \mapsto \lambda y\lambda x.H(x)\}, && \mathfrak{d}_1 = 0.6.\\
\sigma_2 &= \{F \mapsto \lambda x.b(H(x)), G \mapsto \lambda y\lambda x.H(x)\}, && \mathfrak{d}_2 = 0.6.\\
\sigma_3 &= \{F \mapsto \lambda x.c(H(x)), G \mapsto \lambda y\lambda x.H(x)\}, && \mathfrak{d}_3 = 0.5.\\
\sigma_4 &= \{F \mapsto \lambda x.a(a(x)), G \mapsto \lambda y\lambda x.a(x)\}, && \mathfrak{d}_4 = 0.6.\\
\sigma_5 &= \{F \mapsto \lambda x.c(a(x)), G \mapsto \lambda y\lambda x.a(x)\}, && \mathfrak{d}_5 = 0.5.
\end{aligned}$$

From these unifiers, σ_1, σ_2 and σ_3 are most general ones on $V = \{F, G\}$. (They are equivalent to each other modulo similarity.) Hence, we have two most general unifiers σ_1, σ_2 with the maximum degree, and one with a lower degree. The unifier σ_4 is strictly less general on V than the first three, yet its unification degree is maximal. σ_5 is neither most general nor degree-maximal unifier.

Our goal is to design an algorithm that computes a degree-maximal most general unifier. Hence, for this problem, computing either σ_1 or σ_2 would be the desired outcome.

3 Similarity-based unification of higher-order patterns

In this section, we formulate a similarity-based unification algorithm in a rule-based manner. The rules operate on triples P; σ; $\mathfrak{d}$, referred to as unification

configurations, where P is a similarity-based unification problem, σ is the substitution computed so far, and $\mathfrak{d}$ is the approximation degree, also computed so far.

The rules are given below. The symbol $\uplus$ stands for disjoint union. The rule LF calls an auxiliary function VarElim (which is also defined below) with the sides of the selected equation as arguments.

Abs: **Abstraction**

$$\{\lambda x.t \simeq^?_{\mathcal{R},\mu} \lambda x.s\} \uplus P;\ \sigma;\mathfrak{d} \rightsquigarrow \{t \simeq^?_{\mathcal{R},\mu} s\} \cup P;\ \sigma;\mathfrak{d}.$$

Dec: **Decomposition**

$$\{f(t_1,\ldots,t_n) \simeq^?_{\mathcal{R},\mu} g(s_1,\ldots,s_n)\} \uplus P;\ \sigma;\mathfrak{d} \rightsquigarrow \{t_1 \simeq^?_{\mathcal{R},\mu} s_1,\ldots,t_n \simeq^?_{\mathcal{R},\mu} s_n\} \cup P;\ \sigma;\mathfrak{d} \wedge \mathcal{R}(f,g),$$

where $n \geq 0$ and $\mathcal{R}(f,g) \geq \mu$.

SV: **Same variables**

$$\{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} F(y_1,\ldots,y_n)\} \uplus P;\ \sigma;\ \mathfrak{d} \rightsquigarrow P\vartheta;\ \sigma\vartheta;\ \mathfrak{d},$$

where $n \geq 0$ and

- $\{z_1,\ldots,z_m\} = \{x_i \mid x_i = y_i, 1 \leq i \leq n\}$, $m \geq 0$,
- $\vartheta = \{F \mapsto \lambda x_1,\ldots,x_n.H(z_1,\ldots,z_m)\}$ for a fresh variable H of the appropriate type.

Ori: **Orient**

$$\{a(s_1,\ldots,s_m) \simeq^?_{\mathcal{R},\mu} F(x_1,\ldots,x_n)\} \uplus P;\ \sigma;\ \mathfrak{d} \rightsquigarrow \{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\ldots,s_m)\} \cup P;\ \sigma;\mathfrak{d},$$

where $n,m \geq 0$, F is a free variable and a is a constant or $a \in \{x_1,\ldots,x_n\}$.

LF: **Left-flex**

$$\{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\ldots,s_m)\} \uplus P;\ \sigma;\ \mathfrak{d} \rightsquigarrow P\vartheta;\ \sigma\vartheta;\ \mathfrak{d}$$

where $n,m \geq 0$ and

- $F \notin \mathtt{fv}(a(s_1,\ldots,s_m))$,
- a is a constant, free variable, or $a \in \{x_1,\ldots,x_n\}$,
- $\mathsf{VarElim}(F(x_1,\ldots,x_n), a(s_1,\ldots,s_m)) = \varphi$,
- $\vartheta = \varphi|_V$, where $V = \{F\} \cup \mathtt{fv}(a(s_1,\ldots,s_m))$.

In the rule LF, if a is a free variable, then $s_1,\ldots,s_m$ are distinct bound variables (since we work only with higher-order patterns). By this rule, in order to unify a flex-term t and a term s, the auxiliary function $\mathsf{VarElim}(t,s)$ is called, which creates an initial configuration $\{t \simeq^? s\};\varepsilon$ and applies the VE1 and VE2 rules below as long as possible. If the process stops with the configuration $\emptyset;\varphi$, we say that φ is the answer computed by VarElim for t and s and write $\mathsf{VarElim}(t,s) = \varphi$. Note that φ might contain bindings for variables introduced in the process of VarElim, which appear neither in t nor in s. Therefore, in ϑ, we keep only those bindings from φ that are relevant to t and s. Note also that VarElim is independent of $\mathcal{R}$ and does not involve the degree computation.

VE1: **Variable elimination 1**

$$\{F(x_1,\dots,x_n) \simeq^? a(s_1,\dots,s_m)\} \uplus P; \sigma \rightsquigarrow \{H_1(x_1,\dots,x_n) \simeq^? s_1,\dots,H_m(x_1,\dots,x_n) \simeq^? s_m\} \cup P; \sigma\vartheta,$$

where $n, m \geq 0$ and

- a is a constant or $a \in \{x_1,\dots,x_n\}$,
- $\vartheta = \{F \mapsto \lambda x_1,\dots,x_n.a(H_1(x_1,\dots,x_n),\dots,H_m(x_1,\dots,x_n))\}$ where the variables $H_1,\dots,H_m$ are fresh variables of appropriate types.
- (Note that if s_i, $1 \leq i \leq n$, is of function type, the term $H_i(x_1,\dots,x_n)$ needs to be η-expanded.)

VE2: **Variable elimination 2**

$$\{F(x_1,\dots,x_n) \simeq^? G(y_1,\dots,y_m)\} \uplus P; \sigma; \rightsquigarrow P\vartheta; \sigma\vartheta,$$

where $n, m \geq 0$, and

- $\{x_1,\dots,x_n\} \cap \{y_1,\dots,y_m\} = \{z_1,\dots,z_k\}$,
- $\vartheta = \{F \mapsto \lambda x_1,\dots,x_n.H(z_1,\dots,z_k), G \mapsto \lambda y_1,\dots,y_m.H(z_1,\dots,z_k)\}$ with H being a fresh variable of the appropriate type.

Note that when VarElim is invoked, it is always called with terms t and s where t has the form $F(x_1,\dots,x_n)$ such that the variable F does not occur in s. Applying VE1 or VE2 to the initial configuration $\{F(x_1,\dots,x_n) \simeq^? s\}; \varepsilon$ creates a new configuration in which F does not appear anymore, and if the variables introduced instead of it appear on the left-hand sides, then they are unique again. Hence, the configurations $P; \sigma$ to which VE1 and VE2 apply satisfy the following invariant:

- If an equation of the form $F(x_1,\dots,x_n) \simeq^? s$ appears in P, then F is unique.

Consequently, in VE1, it does not make sense to apply the generated substitution to the remaining unification problem. That's why we have P and not $P\vartheta$ on the right-hand side of VE1. On the other hand, we need $P\vartheta$ in VE2, because P might contain G.

Given a similarity relation $\mathcal{R}$, the cut value μ, and two terms t and s, to find an $(\mathcal{R},\mu)$-unifier of t and s, the algorithm HOPUS creates the initial configuration $\{t \simeq^?_{\mathcal{R},\mu} s\}; \varepsilon; 1$ and applies the rules above as long as possible. It means that HOPUS can be defined as the following strategy where nf stands for normal form:

$$\text{HOPUS} := \mathsf{nf}(\mathsf{Step}). \qquad \mathsf{Step} := \mathsf{choice}(\mathsf{Abs}, \mathsf{Dec}, \mathsf{SV}, \mathsf{Ori}, \mathsf{LF}).$$

It is easy to see that for each selected equation, only one rule applies. Hence, Step is deterministic: it transforms a given configuration C in only one way, to a configuration which we will refer to as $\mathsf{Step}(C)$.

Example 2. We illustrate the algorithm to solve an $(\mathcal{R},\mu)$-unification problem from Example 1. We create the initial configuration

$$\{\lambda x.\lambda y.f(F(x),F(y)) \simeq^?_{\mathcal{R},0.4} \lambda x.\lambda y.g(a(G(y,x)),b(G(x,y)))\}; \epsilon; 1$$

and apply the transformation rules as follows:

$\{\lambda x.\lambda y.f(F(x),F(y)) \simeq^?_{\mathcal{R},0.4} \lambda x.\lambda y.g(a(G(y,x)),b(G(x,y)))\};\epsilon;1 \rightsquigarrow_{\mathsf{Abs}}$

$\{\lambda y.f(F(x),F(y)) \simeq^?_{\mathcal{R},0.4} \lambda y.g(a(G(y,x)),b(G(x,y)))\};\epsilon;1 \rightsquigarrow_{\mathsf{Abs}}$

$\{f(F(x),F(y)) \simeq^?_{\mathcal{R},0.4} g(a(G(y,x)),b(G(x,y)))\};\epsilon;1 \rightsquigarrow_{\mathsf{Dec}}$

$\{F(x) \simeq^?_{\mathcal{R},0.4} a(G(y,x)), F(y) \simeq^?_{\mathcal{R},0.4} b(G(x,y))\};\epsilon;0.8 \rightsquigarrow_{\mathsf{LF}}$

Application of LF requires steps of the VarElim:

$\{F(x) \simeq^? a(G(y,x))\};\epsilon \rightsquigarrow_{\mathsf{VE1}}$

$\{H_1(x) \simeq^? G(y,x)\};\{F \mapsto \lambda x.a(H_1(x))\} \rightsquigarrow_{\mathsf{VE2}}$

$\emptyset;\{F \mapsto \lambda x.a(H_2(x)), G \mapsto \lambda y\lambda x.H_2(x), H_1 \mapsto \lambda x.H_2(x)\}$

VarElim returns $\{F \mapsto \lambda x.a(H_2(x)), G \mapsto \lambda y\lambda x.H_2(x)\}$ and we continue:

$\{a(H_2(y)) \simeq^?_{\mathcal{R},0.4} b(H_2(y))\};\{F \mapsto \lambda x.a(H_2(x)), G \mapsto \lambda y\lambda x.H_2(x)\};0.8 \rightsquigarrow_{\mathsf{Dec}}$

$\{H_2(y) \simeq^?_{\mathcal{R},0.4} H_2(y)\};\{F \mapsto \lambda x.a(H_2(x)), G \mapsto \lambda y\lambda x.H_2(x)\};0.6 \rightsquigarrow_{\mathsf{SV}}$

$\emptyset;\{F \mapsto \lambda x.a(H(x)), G \mapsto \lambda y\lambda x.H(x), H_2 \mapsto \lambda x.H(x)\};0.6$

Hence, we obtained the substitution $\{F \mapsto \lambda x.a(H(x)), G \mapsto \lambda y\lambda x.H(x), H_2 \mapsto \lambda x.H(x)\}$ and the degree 0.6. It is easy to check that the substitution is indeed a unifier of the given terms with degree 0.6. Note also that not all the bindings from the substitution are relevant: we can restrict it only to the free variables of the given terms, getting $\{F \mapsto \lambda x.a(H(x)), G \mapsto \lambda y\lambda x.H(x)\}$, which corresponds to σ_1 from Example 1. If we applied the LF rule to the second equation $F(y) \simeq^?_{\mathcal{R},0.4} b(G(x,y))$ instead of the first one, we would get the unifier σ_2 from Example 1.

If we increase the cut value to, say, 0.7, the problem does not have a solution and the algorithm stops with a configuration to which no rule applies:

$$\{a(H_2(y)) \simeq^?_{\mathcal{R},0.7} b(H_2(y))\};\{F \mapsto \lambda x.a(H_2(x)), G \mapsto \lambda y\lambda x.H_2(x)\};0.8.$$

When $\mu = 1$, the problem is a crisp (i.e., standard) higher-order pattern unification problem, which obviously is not solvable due to the mismatch between f and g. The algorithm detects it, stopping after the application of the abstraction steps.

Theorem 2 (Termination of HOPUS). *HOPUS terminates for any input either with $\emptyset;\sigma;\mathfrak{d}$ or with $P;\sigma;\mathfrak{d}$, where $P \neq \emptyset$ but no rule applies to it.*

Proof. We first establish that $\mathsf{VarElim}(t,s)$ always terminates. This procedure starts by constructing the initial configuration $\{t \simeq^? s\};\varepsilon$ and then repeatedly applies the VE1 and VE2 rules. To each configuration $P;\sigma$ we associate a complexity measure as the multiset of the sizes of equations in P:

$$\{\!\{ size(t) \cup size(s) \mid t \simeq^? s \in P \}\!\}.$$

To order the measures, we use the multiset extension of the standard ordering on natural numbers (Dershowitz-Manna ordering [3]). Both VE1 and VE2

rules strictly decrease this measure, ensuring that the reduction process cannot continue indefinitely. Consequently, $\mathsf{VarElim}(t, s)$ must terminate.

Now we show termination of Hopus. For this, we define a complexity measure $cm(C)$ for a configuration $C = (P; \sigma; \mathfrak{d})$, and show that $cm(C) > cm(C')$ holds whenever $C' = \mathsf{Step}(C)$. The measure is the triple $cm(C) = \langle N_1, N_2, N_3 \rangle$ defined as follows:

- N_1 is the number of distinct variables in P,
- $N_2 := \{\!\{ size(t) \cup size(s) \mid t \simeq^{?}_{\mathcal{R},\mu} s \in P \}\!\}$ ($\{\!\{\}\!\}$ stands for a multiset),
- N_3 the number of rigid-flex equations in P.

Measures are compared lexicographically. For N_2, we use the Dershowitz-Manna ordering. The table below shows which rule reduces which component of the measure.

Rule	N_1	N_2	N_3
LF	$>$		
Abs, Dec, SV	$\geq$	$>$	
Ori	$\geq$	$\geq$	$>$

Hence, each rule strictly reduces $cm(C)$. Since the ordering is well-founded, it implies the termination of the algorithm. It is obvious that there are only two alternatives for the final configuration: either the first component is empty, or it contains an equation to which no rule applies. It finishes the proof.

When Hopus stops with $\emptyset; \sigma; \mathfrak{d}$, we say that Hopus *succeeds* with the *computed answer* $(\sigma, \mathfrak{d})$ (or computes $(\sigma, \mathfrak{d})$). If it stops with $P; \sigma; \mathfrak{d}$ where $P \neq \emptyset$ but no rule applies to it, we say that Hopus *fails*. For a similarity relation $\mathcal{R}$, a cut value μ, and two terms t and s, if Hopus computes $(\sigma, \mathfrak{d})$, we write $\textsc{Hopus}(t, s, \mathcal{R}, \mu) = (\sigma, \mathfrak{d})$.

Lemma 1. *Let $\mathcal{R}$ be a similarity relation, μ a cut value and $P; \sigma; \mathfrak{d}$ a configuration. If $\mathsf{Step}(P; \sigma; \mathfrak{d}) = (P'; \sigma\vartheta; \mathfrak{d} \wedge \mathfrak{d}')$ and τ is an $(\mathcal{R}, \mu)$-unifier of P' with degree $\mathfrak{d}''$, then $\vartheta\tau$ is an $(\mathcal{R}, \mu)$-unifier of P with degree $\mathfrak{d}' \wedge \mathfrak{d}''$.*

Proof. First, we prove the soundness of VarElim. This will be needed to show the soundness of the LF rule. However, note that the rules VE1 and VE2 are the standard variable elimination rules used in the crisp higher-order pattern unification [23,25] under the condition that variable in the left-hand side of the. Then their soundness follows from the soundness of the corresponding rules in the standard higher-order pattern unification.

Further, we proceed by case distinction on the inference rules of Hopus. We illustrate here three cases when the selected rules are Dec, SV, and LF. For the other rules, the lemma can be shown similarly.

- Step transforms the given configuration by the Dec rule, i.e., we have

$$\{f(t_1, \ldots, t_n) \simeq^{?}_{\mathcal{R},\mu} g(s_1, \ldots, s_n)\} \uplus P;\ \sigma;\ \mathfrak{d} \rightsquigarrow_{\mathsf{Dec}}$$

$$\{t_1 \simeq^?_{\mathcal{R},\mu} s_1, \ldots, t_n \simeq^?_{\mathcal{R},\mu} s_n\} \cup P;\ \sigma;\ \mathfrak{d} \wedge \mathcal{R}(f,g).$$

Then $\vartheta = \varepsilon$ and $\mathfrak{d}' = \mathcal{R}(f,g) \geq \mu$. Let τ be an $(\mathcal{R},\mu)$-unifier of the problem $\{t_1 \simeq^?_{\mathcal{R},\mu} s_1, \ldots, t_n \simeq^?_{\mathcal{R},\mu} s_n\} \cup P$ with the degree $\mathfrak{d}''$. This means,

$$\mathfrak{d}'' = \mathcal{R}(t_1\tau, s_1\tau) \wedge \cdots \wedge \mathcal{R}(t_n\tau, s_n\tau) \wedge d \geq \mu,$$

where $d = \bigwedge_{(t \simeq^?_{\mathcal{R},\mu} s) \in P} \mathcal{R}(t\tau, s\tau)$. Since $\mathfrak{d}' \geq \mu$ and $\mathfrak{d}'' \geq \mu$, we have $\mathfrak{d}' \wedge \mathfrak{d}'' \geq \mu$. Moreover,

$$\begin{aligned}\mathfrak{d}' \wedge \mathfrak{d}'' &= \mathfrak{d}' \wedge \mathcal{R}(t_1\tau, s_1\tau) \wedge \cdots \wedge \mathcal{R}(t_n\tau, s_n\tau) \wedge d \\ &= \mathcal{R}(f,g) \wedge \mathcal{R}(t_1\tau, s_1\tau) \wedge \cdots \wedge \mathcal{R}(t_n\tau, s_n\tau) \wedge d \\ &= \mathcal{R}(f(t_1,\ldots,t_n)\tau, g(s_1,\ldots,s_n)\tau) \wedge d,\end{aligned}$$

which implies that $\tau = \vartheta\tau$ is an $(\mathcal{R},\mu)$-unifier of the unification problem $\{f(t_1,\ldots,t_n) \simeq^?_{\mathcal{R},\mu} g(s_1,\ldots,s_n)\} \uplus P$ with degree $\mathfrak{d}' \wedge \mathfrak{d}''$.

– Step transforms the given configuration by the SV rule, i.e., we have

$$\{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} F(y_1,\ldots,y_n)\} \uplus P;\ \sigma;\ \mathfrak{d} \rightsquigarrow_{\mathsf{SV}} P\vartheta;\ \sigma\vartheta;\ \mathfrak{d}.$$

where $\vartheta = \{F \mapsto \lambda x_1,\ldots,x_n.H(z_1,\ldots,z_m)\}$, $\{z_1,\ldots,z_m\} = \{x_i \mid x_i = y_i, 1 \leq i \leq n\}$. We also have $\mathfrak{d}' = 1$. Let τ be an $(\mathcal{R},\mu)$-unifier of $P\vartheta$ with the degree $\mathfrak{d}''$. This means that $\mathfrak{d}'' = \bigwedge_{(t\vartheta \simeq^?_{\mathcal{R},\mu} s\vartheta) \in P\vartheta} \mathcal{R}(t\vartheta\tau, s\vartheta\tau) \geq \mu$. Then $\vartheta\tau$ is also unifier of $\{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} F(y_1,\ldots,y_n)\} \uplus P$ with degree $\mathfrak{d}''$. Indeed, we have

$$\begin{aligned}&\mathcal{R}(F(x_1,\ldots,x_n)\vartheta\tau, F(y_1,\ldots,y_n)\vartheta\tau) \wedge \bigwedge_{(t \simeq^?_{\mathcal{R},\mu} s) \in P} \mathcal{R}(t\vartheta\tau, s\vartheta\tau) \\ &= \mathcal{R}(H(z_1,\ldots,z_m)\tau, H(z_1,\ldots,z_m)\tau) \wedge \bigwedge_{(t\vartheta \simeq^?_{\mathcal{R},\mu} s\vartheta) \in P\vartheta} \mathcal{R}(t\vartheta\tau, s\vartheta\tau) \\ &= 1 \wedge \mathfrak{d}'' = \mathfrak{d}''.\end{aligned}$$

– Step transforms the given configuration by the LF rule, i.e., we have

$$\{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\ldots,s_m)\} \uplus P;\ \sigma;\ \mathfrak{d} \rightsquigarrow_{\mathsf{LF}} P\vartheta;\ \sigma\vartheta;\ \mathfrak{d},$$

where ϑ is an $(\mathcal{R},\mu)$-unifier of $F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\ldots,s_m)$ with degree $\mathfrak{d}' = 1$. Let τ be an $(\mathcal{R},\mu)$-unifier of $P\vartheta$ with degree $\mathfrak{d}''$. Hence, $\mathfrak{d}'' = \bigwedge_{(t\vartheta \simeq^?_{\mathcal{R},\mu} s\vartheta) \in P\vartheta} \mathcal{R}(t\vartheta\tau, s\vartheta\tau) \geq \mu$. Then, since ϑ is an $(\mathcal{R},\mu)$-unifier of the equation $F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\ldots,s_m)$ with degree $\mathfrak{d}'$, so is $\vartheta\tau$. Hence, we get

$$\begin{aligned}&\mathcal{R}(F(x_1,\ldots,x_n)\vartheta\tau, a(s_1,\ldots,s_m)\vartheta\tau) \wedge \bigwedge_{(t \simeq^?_{\mathcal{R},\mu} s) \in P} \mathcal{R}(t\vartheta\tau, s\vartheta\tau) \\ &= \mathfrak{d}' \wedge \bigwedge_{(t\vartheta \simeq^?_{\mathcal{R},\mu} s\vartheta) \in P\vartheta} \mathcal{R}(t\vartheta\tau, s\vartheta\tau)\end{aligned}$$

$$= 1 \wedge \mathfrak{d}'' = \mathfrak{d}'',$$

which means that $\vartheta\tau$ is an $(\mathcal{R},\mu)$-unifier of the problem $\{F(x_1,\dots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\dots,s_m)\} \uplus P$ with degree $\mathfrak{d}''$.

Theorem 3 (Soundness of HOPUS). *Given a similarity relation $\mathcal{R}$, a cut value μ, and two terms t and s, if $\text{HOPUS}(t,s,\mathcal{R},\mu) = (\sigma,\mathfrak{d})$, then $\mathfrak{d} \geq \mu$ and $\mathcal{R}(t\sigma, s\sigma) = \mathfrak{d}$, i.e., σ is an $(\mathcal{R},\mu)$-unifier of t and s with degree $\mathfrak{d}$.*

Proof. From $\text{HOPUS}(t,s,\mathcal{R},\mu) = (\sigma,\mathfrak{d})$ we have the derivation

$$P_0;\, \sigma_0;\, \mathfrak{d}_0 \rightsquigarrow P_0;\, \sigma_0\vartheta_1;\, \mathfrak{d}_0 \wedge \mathfrak{d}_1 \rightsquigarrow^* P_n;\, \sigma_0\vartheta_1\cdots\vartheta_n;\, \mathfrak{d}_0 \wedge \mathfrak{d}_1 \wedge \cdots \wedge \mathfrak{d}_n,$$

where

$$P_0 = \{t \simeq^?_{\mathcal{R},\mu} s\}, \quad \sigma_0 = \varepsilon, \quad \mathfrak{d}_0 = 1,$$
$$P_n = \emptyset, \quad \sigma = \sigma_0\vartheta_1\cdots\vartheta_n = \vartheta_1\cdots\vartheta_n, \quad \mathfrak{d} = \mathfrak{d}_0 \wedge \mathfrak{d}_1 \wedge \cdots \wedge \mathfrak{d}_n = \mathfrak{d}_1 \wedge \cdots \wedge \mathfrak{d}_n.$$

Therefore, we get the desired result using simple induction with Lemma 1.

Lemma 2. *Let $\mathcal{R}$ be a similarity relation, μ a cut value and $P_0;\sigma_0;\mathfrak{d}_0$ a configuration. Assume τ is an $(\mathcal{R},\mu)$-unifier of P_0 with degree $\mu \leq \mathfrak{d}_\tau$ and $\sigma_0 \preceq^{Dom(\tau)}_{\mathcal{R},\mu} \tau$. Then we can make a step $\mathsf{Step}(P_0;\sigma_0;\mathfrak{d}_0) = (P_1;\sigma_1;\mathfrak{d}_1)$ such that*

- *$\mathfrak{d}_1 = \mathfrak{d}_0 \wedge \mathfrak{d}'_1$ for some $\mathfrak{d}'_1$,*
- *there exists a substitution φ_1 such that*
  - *$Dom(\varphi_1) = \mathtt{fv}(P_1) \setminus \mathtt{fv}(P_0)$ (the set of new free variables in P_1),*
  - *$\mathtt{fv}(Ran(\varphi_1)) \cap Dom(\tau) = \emptyset$,*
  - *$\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of P_1 with degree $\mu \leq \mathfrak{d}_{\varphi_1\tau}$ such that $\mathfrak{d}_\tau \leq \mathfrak{d}'_1 \wedge \mathfrak{d}_{\varphi_1\tau}$, and*
  - *$\sigma_1 \preceq^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$.*

Proof. Assume without loss of generality that τ is idempotent. We prove the lemma by case distinction on the rules applied by Step. For rules Abs and Ori it is simple. We concentrate on the other rules.

Dec: In this case, $P_0 = \{f(t_1,\dots,t_n) \simeq^?_{\mathcal{R},\mu} g(s_1,\dots,s_n)\} \uplus P$ and the application of the rule gives $P_1;\sigma_1;\mathfrak{d}_1$, where $P_1 = \{t_1 \simeq^?_{\mathcal{R},\mu} s_1,\dots,t_n \simeq^?_{\mathcal{R},\mu} s_n\} \cup P$, $\sigma_1 = \sigma_0$, and $\mathfrak{d}_1 = \mathfrak{d}_0 \wedge \mathcal{R}(f,g)$. We take $\mathcal{R}(f,g)$ as $\mathfrak{d}'_1$ and $\varphi_1 = \varepsilon$. Assume τ is an $(\mathcal{R},\mu)$-unifier of P_0 with degree $\mu \leq \mathfrak{d}_\tau$ and $\sigma_0 \preceq^{Dom(\tau)}_{\mathcal{R},\mu} \tau$. Then τ is also an $(\mathcal{R},\mu)$-unifier of P and we denote its degree by $\mathfrak{d}_P$. It implies that

$$\begin{aligned}\mu \leq \mathfrak{d}_\tau &= \mathcal{R}(f(t_1\tau,\dots,t_n\tau), g(s_1\tau,\dots,s_n\tau)) \wedge \mathfrak{d}_P\\ &= \mathcal{R}(f,g) \wedge \mathcal{R}(t_1\tau,s_1\tau) \wedge \cdots \wedge \mathcal{R}(t_n\tau,s_n\tau) \wedge \mathfrak{d}_P\\ &= \mathfrak{d}'_1 \wedge \mathcal{R}(t_1\varphi_1\tau,s_1\varphi_1\tau) \wedge \cdots \wedge \mathcal{R}(t_n\varphi_1\tau,s_n\varphi_1\tau) \wedge \mathfrak{d}_P.\end{aligned}$$

Denote $\mathcal{R}(t_1\varphi_1\tau,s_1\varphi_1\tau) \wedge \cdots \wedge \mathcal{R}(t_n\varphi_1\tau,s_n\varphi_1\tau) \wedge \mathfrak{d}_P$ by $\mathfrak{d}_{\varphi_1\tau}$. Then the previous inequality can be written as $\mu \leq \mathfrak{d}'_1 \wedge \mathfrak{d}_{\varphi_1\tau}$, from which we get

$\mu \leq \mathfrak{d}_{\varphi_1\tau}$, implying that $\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of $\{t_1 \simeq^?_{\mathcal{R},\mu} s_1, \ldots, t_n \simeq^?_{\mathcal{R},\mu} s_n\} \cup P$ with degree $\mathfrak{d}_{\varphi_1\tau}$. Hence, we got that $\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of P_1 with degree $\mu \leq \mathfrak{d}_{\varphi_1\tau}$ such that $\mathfrak{d}_\tau = \mathfrak{d}'_1 \wedge \mathfrak{d}_{\varphi_1\tau}$. Finally, from $\sigma_1 = \sigma_0$, $Dom(\varphi_1\tau) = Dom(\tau)$, and $\sigma_0 \preceq^{Dom(\tau)}_{\mathcal{R},\mu} \tau$, we conclude $\sigma_1 \preceq^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \tau$.

SV: In this case, $P_0 = \{F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} F(y_1,\ldots,y_n)\} \uplus P$ and the application of the rule gives $P_1;\sigma_1;\mathfrak{d}_1$ where $P_1 = P\vartheta$, $\sigma_1 = \sigma_0\vartheta$, and $\mathfrak{d}_1 = \mathfrak{d}_0$ for the substitution $\vartheta = \{F \mapsto \lambda x_1,\ldots,x_n.H(z_1,\ldots,z_m)\}$ where $\{z_1,\ldots,z_m\} = \{x_i \mid x_i = y_i, 1 \leq i \leq n\}$, $m \geq 0$.
Since τ is an $(\mathcal{R},\mu)$-unifier of P_0, we have $F\tau = \lambda x_1,\ldots,x_n.t$, where t is such a term that if some x_i or y_i, $1 \leq i \leq n$, occurs in t, then this variable should belong to $\{z_1,\ldots,z_m\}$, otherwise τ would not be a unifier of $F(x_1,\ldots,x_n) \simeq^?_{\mathcal{R},\mu} F(y_1,\ldots,y_n)$. It implies that $F\tau$ can be actually represented as

$$F\tau = (\lambda x_1,\ldots,x_n.H(z_1,\ldots,z_m))\{H \mapsto \lambda z_1,\ldots,z_m.t\}. \tag{1}$$

We take $\mathfrak{d}'_1 = 1$ and $\varphi_1 = \{H \mapsto \lambda z_1,\ldots,z_m.t\}$. Then

- $\mathfrak{d}_1 = \mathfrak{d}_0 = \mathfrak{d}_0 \wedge \mathfrak{d}'_1$,
- $Dom(\varphi_1) = \{H\} = \mathtt{fv}(P_1) \setminus \mathtt{fv}(P_0)$,
- $\mathtt{fv}(Ran(\varphi_1)) \cap Dom(\tau) = \emptyset$, since τ is idempotent.

What remains to show is

- $\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of P_1 with degree $\mu \leq \mathfrak{d}_{\varphi_1\tau}$ such that $\mathfrak{d}_\tau \leq \mathfrak{d}'_1 \wedge \mathfrak{d}_{\varphi_1\tau} = \mathfrak{d}_{\varphi_1\tau}$, and
- $\sigma_1 \preceq^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$.

To show the first of them, we prove that $\vartheta\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of P_0 with degree $\mathfrak{d}_\tau$. For this, take a variable $X \in \mathtt{fv}(P_0)$. If $X \neq F$, then $X\vartheta\varphi_1\tau = X\tau$ since $X\vartheta\varphi_1 = X$. As for $X = F$, we have

$$\begin{aligned} F\vartheta\varphi_1\tau &= \big(\lambda x_1,\ldots,x_n.H(z_1,\ldots,z_m)\big)\varphi_1\tau = \text{ (by equality (1))} \\ &= F\tau\tau = \text{ (by the idempotence of } \tau) \ = F\tau. \end{aligned}$$

Hence, we can conclude that $\vartheta\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of P_0 with degree $\mu \leq \mathfrak{d}_{\vartheta\varphi_1\tau} = \mathfrak{d}_\tau$, implying that $\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of $P_1 = P\vartheta$ with degree $\mathfrak{d}_\tau$.
Now we should prove $\sigma_1 \preceq^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$. First, note that from $\sigma_0 \preceq^{Dom(\tau)}_{\mathcal{R},\mu} \tau$, there exists a substitution ψ_0 (whose domain is disjoint from the free variables in the range of τ) such that $X\sigma_0\psi_0 \simeq_{\mathcal{R},\mu} X\tau$ for all $X \in Dom(\tau)$. Define $\psi_1 = \varphi_1 \cup \psi_0$. (Note that ψ_0 can be chosen so that $Dom(\psi_0) \cap (Dom(\varphi_1) \cup \mathtt{fv}(Ran(\varphi_1))) = \emptyset$ and $Dom(\varphi_1) \cap (Dom(\psi_0) \cup \mathtt{fv}(Ran(\psi_0))) = \emptyset$.) Take a variable $X \in Dom(\varphi_1\tau) = \{H\} \uplus Dom(\tau)$ and show that $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$.

1. $X = H$. Then $X\sigma_1\psi_1 = X\psi_1 = X\varphi_1 = \lambda z_1,\ldots,z_m.t$, where t is such a term that $F\tau = \lambda x_1,\ldots,x_n.t$. But then, $\mathtt{fv}(t) \cap Dom(\tau) = \emptyset$ and we have $\lambda z_1,\ldots,z_m.t = \lambda z_1,\ldots,z_m.t\tau = X\varphi_1\tau$. Hence, $X\sigma_1\psi_1 = X\varphi_1\tau$.

2. $X = F$. Then

$$\begin{aligned} X\sigma_1\psi_1 = X\sigma_0\vartheta\psi_1 = X\vartheta\psi_1 &= \lambda x_1,\dots,x_n.H(z_1,\dots,z_m)\psi_1 \\ &= \lambda x_1,\dots,x_n.H(z_1,\dots,z_m)\varphi_1 = \lambda x_1,\dots,x_n.t. \\ X\varphi_1\tau = X\tau &= \lambda x_1,\dots,x_n.t. \end{aligned}$$

   Hence, also in this case, $X\sigma_1\psi_1 = X\varphi_1\tau$.
3. $X \in Dom(\tau) \setminus \{F\}$. We have two subcases:
   - $X\sigma_0\vartheta = X\sigma_0$. Then

$$\begin{aligned} X\sigma_1\psi_1 = X\sigma_0\vartheta\psi_1 = X\sigma_0\psi_1 = X\sigma_0\psi_0 &\simeq_{\mathcal{R},\mu} X\tau = \\ &\text{(by the assumption on } \psi_0) \\ &= X\varphi_1\tau \text{ (since } X \notin Dom(\varphi_1)). \end{aligned}$$

   - $X\sigma_0\vartheta \neq X\sigma_0$, then we should show $Z\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Z\varphi_1\tau$ for all $Z \in \mathtt{fv}(X\sigma_0\vartheta)$, which implies $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$.
     Proving $Z\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Z\varphi_1\tau$ for all $Z \in \mathtt{fv}(X\sigma_0\vartheta)$. From $X\sigma_0\vartheta \neq X\sigma_0$ we have $F \in \mathtt{fv}(X\sigma_0)$ and, therefore, $H \in \mathtt{fv}(X\sigma_0\vartheta)$. However, for H we already saw that $H\sigma_1\psi_1 = H\varphi_1\tau$. For any other $Y \in \mathtt{fv}(X\sigma_0\vartheta)$ (i.e., $Y \neq H$, $Y \neq F$) we should have $Y \in \mathtt{fv}(X\sigma_0)$, implying $Y \notin Dom(\sigma_0)$ (since Y appears in the range of σ_0 and σ's are idempotent). From $Y \notin Dom(\sigma_0)$ and $Y \neq F$ we get $Y \notin Dom(\sigma_1)$ and, consequently, $Y\sigma_1\psi_1 = Y\psi_1$. Then we have

$$\begin{aligned} Y\sigma_1\psi_1 = Y\psi_1 = \text{ (since } Y \neq H) \; Y\psi_0 = \text{ (since } Y \notin Dom(\sigma_0)) \\ = Y\sigma_0\psi_0 \simeq_{\mathcal{R},\mu} Y\tau = Y\varphi_1\tau. \end{aligned}$$

     It proves $Z\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Z\varphi_1\tau$ for all $Z \in \mathtt{fv}(X\sigma_0\vartheta)$.

   In both cases, we get $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$.

Hence, in all three cases, we proved $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$. It implies that $\sigma_1 \lesssim^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$.

LF: Here $P_0 = \{F(x_1,\dots,x_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\dots,s_m)\} \uplus P$ and the application of the rule gives $P_1;\sigma_1;\mathfrak{d}_1$ where $P_1 = P\vartheta$, $\sigma_1 = \sigma_0\vartheta$, and $\mathfrak{d}_1 = \mathfrak{d}_0$ for the substitution ϑ where ϑ is computed by $\mathsf{VarElim}(F(x_1,\dots,x_n), a(s_1,\dots,s_m))$. Note that $\mathsf{VarElim}$ applies (crisp) rules $\mathsf{VE1}$ and $\mathsf{VE2}$. From [25] it follows that repeated application of them leads to a most general pattern unifier of $F(x_1,\dots,x_n)$ and $a(s_1,\dots,s_m)$). Therefore, ϑ is such an mgu and we have $F(x_1,\dots,x_n)\vartheta = a(s_1,\dots,s_m)\vartheta$ and $\vartheta \lesssim^{Dom(\vartheta)}_{\mathcal{R},\mu} \tau$, where $Dom(\vartheta) = \{F\} \cup \mathtt{fv}(a(s_1,\dots,s_m))$. Note that all variables that appear in the range of ϑ are new because none of the rules retain the original variables from $a(s_1,\dots,s_m)$. Then there exists a substitution φ_1 such that
- $Dom(\varphi_1) = \mathtt{fv}(P_1) \setminus \mathtt{fv}(P_0) = \mathtt{fv}(Ran(\vartheta))$,
- $\mathtt{fv}(Ran(\varphi_1)) \cap Dom(\tau) = \emptyset$, and
- $X\vartheta\varphi_1 \simeq_{\mathcal{R},\mu} X\tau$ for all $X \in Dom(\vartheta)$.

Therefore, by the idempotence of τ, we get $X\vartheta\varphi_1\tau \simeq_{\mathcal{R},\mu} X\tau$ for all $X \in \mathtt{fv}(P_0)$. Moreover, we have

– $\mathcal{R}(F\vartheta\varphi_1\tau, a(s_1,\ldots,s_m)\vartheta\varphi_1\tau) \geq \mathcal{R}(F\tau, a(s_1,\ldots,s_m)\tau)$, and
– $X\vartheta\varphi_1\tau = X\tau$ for all $X \in \mathtt{fv}(P_0) \setminus Dom(\vartheta)$.

Therefore, $\vartheta\varphi_1\tau$ solves P with the degree $\mathfrak{d}_{\vartheta\varphi_1\tau}$ that is better than $\mathfrak{d}_\tau$. Note that $\mathfrak{d}_{\vartheta\varphi_1\tau}$ and the unification degree of $\varphi_1\tau$ for $P\vartheta$ (i.e., $\mathfrak{d}_{\varphi_1\tau}$) are the same since $s \simeq^?_{\mathcal{R},\mu} t \in P$ iff $s\vartheta \simeq^?_{\mathcal{R},\mu} t\vartheta \in P\vartheta$ and we have

$$\mathfrak{d}_\tau \leq \mathfrak{d}_{\vartheta\varphi_1\tau} = \bigwedge_{s\simeq^?_{\mathcal{R},\mu}t\in P} \mathcal{R}(s\vartheta\varphi_1\tau, t\vartheta\varphi_1\tau) =$$
$$= \bigwedge_{s\vartheta\simeq^?_{\mathcal{R},\mu}t\vartheta\in P\vartheta} \mathcal{R}(s\vartheta\varphi_1\tau, t\vartheta\varphi_1\tau) = \mathfrak{d}_{\varphi_1\tau}.$$

From $\mathfrak{d}_1 = \mathfrak{d}_0 = \mathfrak{d}'_1 \wedge \mathfrak{d}_0$ for $\mathfrak{d}'_1 = 1$, this implies that $\varphi_1\tau$ solves $P_1 = P\vartheta$ with degree $\mathfrak{d}_\tau \leq \mathfrak{d}_{\varphi_1\tau}$.

The last thing to prove is $\sigma_1 \lesssim^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$. From $\sigma_0 \lesssim^{Dom(\tau)}_{\mathcal{R},\mu} \tau$, there exists a substitution ψ_0 such that

– $X\sigma_0\psi_0 \simeq_{\mathcal{R},\mu} X\tau$ for all $X \in Dom(\tau)$,
– $Dom(\psi_0) \cap \mathtt{fv}(Ran(\tau)) = \emptyset$,
– $Dom(\psi_0) \cap (Dom(\varphi_1) \cup \mathtt{fv}(Ran(\varphi_1))) = \emptyset$, and
– $Dom(\varphi_1) \cap (Dom(\psi_0) \cup \mathtt{fv}(Ran(\psi_0))) = \emptyset$.

Let $\psi_1 = \varphi_1 \cup \psi_0$. To prove $\sigma_1 \lesssim^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$, we show $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$ for all $X \in Dom(\varphi_1\tau)$. We have the following cases:

– $X \in Dom(\varphi_1)$. Then $X\sigma_1\psi_1 = X\sigma_0\vartheta\psi_1 = X\vartheta\psi_1 = X\psi_1 = X\varphi_1 = X\varphi_1\tau$ (since $\mathtt{fv}(Ran(\varphi_1) \cap Dom(\tau) = \emptyset$).
– $X \in Dom(\vartheta)$. Then $X\sigma_1\psi_1 = X\sigma_0\vartheta\psi_1 = X\vartheta\psi_1 = X\vartheta\varphi_1 \simeq_{\mathcal{R},\mu} X\tau = X\varphi_1\tau$.
– If $X \in Dom(\tau) \setminus Dom(\vartheta)$. We have two subcases:
  - If $X\sigma_0\vartheta = X\sigma_0$. then $X\sigma_1\psi_1 = X\sigma_0\vartheta\psi_1 = X\sigma_0\psi_1 = X\sigma_0\psi_0 \simeq_{\mathcal{R},\mu} x\tau = x\varphi_1\tau$.
  - If $X\sigma_0\vartheta \neq X\sigma_0$, then we should show $Z\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Z\varphi_1\tau$ for all $Z \in \mathtt{fv}(X\sigma_0\vartheta)$, which implies $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$.
  Proving $Z\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Z\varphi_1\tau$ for all $Z \in \mathtt{fv}(X\sigma_0\vartheta)$. For each $Y \in \mathtt{fv}(Ran(\vartheta)) = Dom(\varphi_1)$ we already saw that $Y\sigma_1\psi_1 = Y\varphi_1\tau$. For each $Y \in \mathtt{fv}(Ran(\sigma_0))\setminus(Dom(\vartheta))$ we have $Y\sigma_0\psi_0 \simeq_{\mathcal{R},\mu} Y\tau = Y\varphi_1\tau$. On the other hand, for such a Y we have $Y\sigma_0\psi_0 = Y\psi_0 = Y = Y\psi_1 = Y\sigma_1\psi_1$. Hence, in both cases we got $Y\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Y\varphi_1\tau$, which proves that $Z\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} Z\varphi_1\tau$ for all $Z \in \mathtt{fv}(X\sigma_0\vartheta)$.

Hence, in all cases, we get $X\sigma_1\psi_1 \simeq_{\mathcal{R},\mu} X\varphi_1\tau$. It implies $\sigma_1 \lesssim^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$.

Theorem 4 (Completeness of HOPUS). *Given a similarity relation $\mathcal{R}$, a cut value μ, and two terms t and s, if there exists an $(\mathcal{R},\mu)$-unifier τ of t and s with degree $\mathfrak{d}_\tau \geq \mu$, then* HOPUS$(t, s, \mathcal{R}, \mu)$ *computes an answer $(\sigma, \mathfrak{d})$ such that $\sigma \lesssim^{Dom(\tau)}_{\mathcal{R},\mu} \tau$ and $\mathfrak{d} \geq \mathfrak{d}_\tau$.*

Proof. First note that for any configuration $P;\sigma;\mathfrak{d}$, if P is $(\mathcal{R},\mu)$-unifiable then Step is defined. This is based on the observation that Step is not applicable

in only two cases: if P contains an equation $f(t_1,\dots,t_n) \simeq^?_{\mathcal{R},\mu} g(s_1,\dots,s_n)$ with $\mathcal{R}(f,g) < \mu$, or an equation $F(t_1,\dots,t_n) \simeq^?_{\mathcal{R},\mu} a(s_1,\dots,s_n)$ with $F \in \mathtt{fv}(a(s_1,\dots,s_n))$. But in none of these cases is P $(\mathcal{R},\mu)$-unifiable.

Hence, for the initial configuration $P_0;\sigma_0;\mathfrak{d}_0 = \{t \simeq^?_{\mathcal{R},\mu} s\};\varepsilon;1$ we have that τ is an $(\mathcal{R},\mu)$-unifier of P_0 with degree $\mu \le \mathfrak{d}_\tau$ and $\sigma_0 \precsim^{Dom(\tau)}_{\mathcal{R},\mu} \tau$. By Lemma 2, we can make $\mathsf{Step}(P_0;\sigma_0;\mathfrak{d}_0) = (P_1;\sigma_1;\mathfrak{d}_1)$ such that

- $\mathfrak{d}_1 = \mathfrak{d}_0 \wedge \mathfrak{d}'_1$ for some $\mathfrak{d}'_1$,
- there exists a substitution φ_1 such that
  - $Dom(\varphi_1) = \mathtt{fv}(P_1) \setminus \mathtt{fv}(P_0)$ (the set of new free variables in P_1),
  - $\mathtt{fv}(Ran(\varphi_1)) \cap Dom(\tau) = \emptyset$,
  - $\varphi_1\tau$ is an $(\mathcal{R},\mu)$-unifier of P_1 with degree $\mu \le \mathfrak{d}'_\tau$ such that $\mathfrak{d}_\tau \le \mathfrak{d}'_1 \wedge \mathfrak{d}'_\tau$, and
  - $\sigma_1 \precsim^{Dom(\varphi_1\tau)}_{\mathcal{R},\mu} \varphi_1\tau$.

The obtained configuration $P_1;\sigma_1;\mathfrak{d}_1$ and the substitution $\varphi_1\tau$ satisfy the conditions of Lemma 2. Hence, we can iterate applications of Step (finitely many times, due to Theorem 2), obtaining the maximal chain of configurations $P_1;\sigma_1;\mathfrak{d}_1 \rightsquigarrow \cdots \rightsquigarrow P_n;\sigma_n;\mathfrak{d}_n$. Then

- $P_n = \emptyset$, otherwise it would contradict the fact that it has a solution;
- $\sigma_n \precsim^{Dom(\varphi_1\cdots\varphi_n\tau)}_{\mathcal{R},\mu} \varphi_1\cdots\varphi_n\tau$;
- $\mathfrak{d}_n = \mathfrak{d}_0 \wedge \mathfrak{d}'_1 \wedge \cdots \wedge \mathfrak{d}'_n$, where $\mathfrak{d}_0 = 1$ and for all $1 \le i \le n$, $\mathfrak{d}'_i$ is such that $\mathfrak{d}_i = \mathfrak{d}_{i-1} \wedge \mathfrak{d}'_i$.

Hence, we obtained $\textsc{Hopus}(t,s,\mathcal{R},\mu) = (\sigma_n,\mathfrak{d}_n)$, where $\sigma_n \precsim^{Dom(\varphi_1\cdots\varphi_n\tau)}_{\mathcal{R},\mu} \varphi_1\cdots\varphi_n\tau$, and there exists $\mathfrak{d}'_\tau$ such that $\mathfrak{d}_\tau \le \mathfrak{d}'_1 \wedge \cdots \wedge \mathfrak{d}'_n \wedge \mathfrak{d}'_\tau = \mathfrak{d}_n \wedge \mathfrak{d}'_\tau \le \mathfrak{d}_n$. Since $Dom(\varphi_1\cdots\varphi_n) \cap Dom(\tau) = \emptyset$ and $\mathtt{fv}(Ran(\varphi_1\cdots\varphi_n)) \cap Dom(\tau) = \emptyset$, the composition $\varphi_1\cdots\varphi_n\tau$ can be also represented as a disjoint union of the set representations of these substitutions: $(\varphi_1\cdots\varphi_n) \uplus \tau$. Therefore, from $\sigma_n \precsim^{Dom(\varphi_1\cdots\varphi_n\tau)}_{\mathcal{R},\mu} \varphi_1\cdots\varphi_n\tau$ we get $\sigma_n \precsim^{Dom(\tau)}_{\mathcal{R},\mu} \tau$. We can take σ_n and $\mathfrak{d}_n$ in the role of σ and $\mathfrak{d}$, respectively, which finishes the proof.

4 Discussion

From fuzzy to crisp. When $\mu = 1$, we are essentially in the crisp case and our algorithm can be seen as a (rule-based) version of the higher-order pattern unification algorithm [23, 25]. It is closer to Nipkow's version [25] since with $\mathsf{VarElim}$, we provide a some kind of configuration transformation strategy. On the other hand, we are more flexible in selecting equations to be transformed since our rules work on sets and not lists. In [25], it was mentioned that using sets instead of lists would lead to divergence, but with our rules and control, it is not the case. We imitate the requirement from [25] to work at the head of the list, but only for the case when the left-hand side of the selected equation is a

flex-term. Here is an example of how our algorithm works on the problematic equations $\{F =^? c(G), G =^? c(F)\}$ from [25]:

$\{F \simeq^?_{\mathcal{R},1} c(G),\ G \simeq^?_{\mathcal{R},1} c(F)\}; \varepsilon; 1 \rightsquigarrow_{\mathsf{LF}}$
Calling VarElim :

$\{F \simeq^? c(G)\}; \varepsilon \rightsquigarrow_{\mathsf{VE1}} \{H \simeq^? G\}; \{F \mapsto c(H)\} \rightsquigarrow_{\mathsf{VE2}} \emptyset; \{F \mapsto c(G),\ H \mapsto G\}$
$\{G \simeq^?_{\mathcal{R},1} c(c(G))\}; \{F \mapsto c(G)\}; 1.$

The obtained configuration can not be transformed further by any rule. Hence, HOPUS fails, as expected.

It should also be mentioned that our rules allow a rather simple termination measure that facilitates the direct termination proof (instead of translating the problem into first-order unification).

T-norms. Note that using the minimum T-norm (in fact, its idempotence property) is important for the completeness of the algorithm as well as for computing unifiers with the maximum approximation degree. (This is not specific to the higher-order case: it holds even in the first-order.) For non-idempotent T-norms, we should treat unification problems as multisets, not sets, and modify the degree condition of the decomposition rule into $(\mathfrak{d} \wedge \mathcal{R}(f,g)) > \mu$. Otherwise, the unification degrees are not computed correctly. However, even under these modifications, the algorithm would be incomplete. To illustrate this, consider a simple similarity relation $\mathcal{R}(a,b) = 0.5$ and the product T-norm defined as $s_1 \wedge s_2 = s_1 * s_2$. Let $t = f(X,X,X)$, $s = f(a,b,b)$, and the cut value $\mu = 0.3$. Then the unification problem $t \simeq^?_{\mathcal{R},\mu} s$ has a unifier $\{X \mapsto b\}$, because $\mathcal{R}(f(b,b,b), f(a,b,b)) = 0.5 > 0.3$. However, the (modified) algorithm fails to compute it:

$$\begin{aligned}
&\{\!\{f(X,X,X) \simeq^?_{\mathcal{R},0.3} f(a,b,b)\}\!\}; \varepsilon; 1 \rightsquigarrow_{\mathsf{Dec}}\\
&\{\!\{X \simeq^?_{\mathcal{R},0.3} a, X \simeq^?_{\mathcal{R},0.3} b, X \simeq^?_{\mathcal{R},0.3} b\}\!\}; \varepsilon; 1 \rightsquigarrow_{\mathsf{LF}}\\
&\{\!\{a \simeq^?_{\mathcal{R},0.3} b, a \simeq^?_{\mathcal{R},0.3} b\}\!\}; \{X \mapsto a\}; 1 \rightsquigarrow_{\mathsf{Dec}}\\
&\{\!\{a \simeq^?_{\mathcal{R},0.3} b\}\!\}; \{X \mapsto a\}; 0.5.
\end{aligned}$$

The computation stops here because the decomposition attempt of a and b would involve the check $0.5 * \mathcal{R}(a,b) \geq 0.3$, which fails since $0.5 * \mathcal{R}(a,b) = 0.5 * 0.5 = 0.25 < 0.3$.

For a lower cut value, e.g., $\mu = 0.1$, the algorithm would compute a solution $\{X \mapsto a\}$ with degree 0.25, failing to discover that there exists another unifier, $\{X \mapsto b\}$, with a better degree 0.3.

However, if we consider only linear unification problems (i.e., those where no free variable occurs more than once), the modified algorithm is complete and computes a unifier with the highest degree even for non idempotent T-norms.

It is possible to regain completeness for non idempotent T-norms for the general case by modifying the subprocedure VarElim: In the VE1 rule, allow the variable F to be replaced not only by a term

$$\lambda x_1, \ldots, x_n.a(H_1(x_1, \ldots, x_n), \ldots, H_m(x_1, \ldots, x_n))$$

but by any term

$$\lambda x_1, \ldots, x_n.b(H_1(x_1, \ldots, x_n), \ldots, H_m(x_1, \ldots, x_n))$$

(non-deterministically) where b is similar to a (provided that it still keeps the degree computed so far above the cut value). Such a modification would lead to multiple (finitely many) most general unifiers, out of which one would need to keep only those that have the highest unification degrees. A detailed discussion of this algorithm goes beyond the scope of this paper, though.

5 Conclusion

We studied unification of higher-order patterns modulo fuzzy similarity relations, using the minimum T-norm. This problem, like its classical (crisp) counterpart, is unitary. We developed a rule-based unification algorithm and proved its termination, soundness, and completeness. The computed most general unifier has the best approximation degree. Our work extends, on one hand, the well-known higher-order pattern unification [23, 25] to accommodate fuzzy similarity relations. On the other hand, it generalizes the weak unification algorithm [30] from first-order terms to simply-typed lambda terms.

There are some interesting directions for future work, which involve both practical applications and further theoretical developments:

- Incorporating the unification algorithm developed in this paper into a higher-order fuzzy logic programming formalism combining the powers of languages like Lambda-Prolog [24] and FASILL [11]. It can serve as a foundation of a flexible higher-order knowledge-based system.
- Relaxing similarity relations to adapt to more diverse scenarios. One possibility is, e.g., lifting the transitivity requirement and considering proximity relations. This can lead to the generalization of block-based [13] or class-based [18,28] approximate unification algorithms to the higher-order setting. Another option is defining approximation based on (Lawvereian) quantales like in quantitative algebras and generalizing the results from [5] to a higher-order case.

Acknowledgments. Partially supported by the Austrian Science Fund (FWF) under project P 35530 and by the Shota Rustaveli National Science Foundation of Georgia under project FR-21-16725.

References

1. H. Aït-Kaci and G. Pasi. Fuzzy lattice operations on first-order terms over signatures with similar constructors: A constraint-based approach. *Fuzzy Sets Syst.*, 391:1–46, 2020.
2. H. Barendregt and G. Manzonetto. *A Lambda Calculus Satellite*. College Publications, 2022.
3. N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *Commun. ACM*, 22(8):465–476, 1979.
4. G. Dowek. Higher-order unification and matching. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning (in 2 volumes)*, pages 1009–1062. Elsevier and MIT Press, 2001.
5. G. Ehling and T. Kutsia. Solving quantitative equations. In C. Benzmüller, M. J. H. Heule, and R. A. Schmidt, editors, *Automated Reasoning - 12th International Joint Conference, IJCAR 2024, Nancy, France, July 3-6, 2024, Proceedings, Part II*, volume 14740 of *Lecture Notes in Computer Science*, pages 381–400. Springer, 2024.
6. F. A. Fontana and F. Formato. Likelog: A logic programming language for flexible data retrieval. In B. R. Bryant, G. B. Lamont, H. Haddad, and J. H. Carroll, editors, *Proceedings of the 1999 ACM Symposium on Applied Computing, SAC'99, San Antonio, Texas, USA, February 28 - March 2, 1999*, pages 260–267. ACM, 1999.
7. F. A. Fontana and F. Formato. A similarity-based resolution rule. *Int. J. Intell. Syst.*, 17(9):853–872, 2002.
8. F. Formato, G. Gerla, and M. I. Sessa. Extension of logic programming by similarity. In M. C. Meo and M. V. Ferro, editors, *1999 Joint Conference on Declarative Programming, AGP'99, L'Aquila, Italy, September 6-9, 1999*, pages 397–410, 1999.
9. F. Formato, G. Gerla, and M. I. Sessa. Similarity-based unification. *Fundam. Inform.*, 41(4):393–414, 2000.
10. J. A. Guerrero, G. Moreno, J. A. Riaza, and J. Sanchez. Smart design of similarity relations for fuzzy logic programming environments. In *IEEE Symposium Series on Computational Intelligence, SSCI 2018, Bangalore, India, November 18-21, 2018*, pages 220–227. IEEE, 2018.
11. P. Julián Iranzo, G. Moreno, and J. A. Riaza. The fuzzy logic programming language FASILL: design and implementation. *Int. J. Approx. Reason.*, 125:139–168, 2020.
12. P. Julián Iranzo, G. Moreno, and J. A. Riaza. Some properties of substitutions in the framework of similarity relations. *Fuzzy Sets Syst.*, 465:108510, 2023.
13. P. Julián-Iranzo and C. Rubio-Manzano. Proximity-based unification theory. *Fuzzy Sets and Systems*, 262:21–43, 2015.
14. P. Julián Iranzo and C. Rubio-Manzano. A sound and complete semantics for a similarity-based logic programming language. *Fuzzy Sets and Systems*, 317:1–26, 2017.
15. P. Julián Iranzo and F. Sáenz-Pérez. An efficient proximity-based unification algorithm. In *2018 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2018, Rio de Janeiro, Brazil, July 8-13, 2018*, pages 1–8. IEEE, 2018.
16. P. Julián Iranzo and F. Sáenz-Pérez. Bousi∼Prolog: Design and implementation of a proximity-based fuzzy logic programming language. *Expert Syst. Appl.*, 213(Part):118858, 2023.

17. S. Krajci, R. Lencses, J. Medina, M. Ojeda-Aciego, and P. Vojtás. A similarity-based unification model for flexible querying. In T. Andreasen, A. Motro, H. Christiansen, and H. L. Larsen, editors, *Flexible Query Answering Systems, 5th International Conference, FQAS 2002, Copenhagen, Denmark, October 27-29, 2002, Proceedings*, volume 2522 of *Lecture Notes in Computer Science*, pages 263–273. Springer, 2002.
18. T. Kutsia and C. Pau. Solving proximity constraints. In M. Gabbrielli, editor, *Logic-Based Program Synthesis and Transformation - 29th International Symposium, LOPSTR 2019, Porto, Portugal, October 8-10, 2019, Revised Selected Papers*, volume 12042 of *Lecture Notes in Computer Science*, pages 107–122. Springer, 2019.
19. V. Loia, S. Senatore, and M. I. Sessa. Similarity-based SLD resolution and its role for web knowledge discovery. *Fuzzy Sets Syst.*, 144(1):151–171, 2004.
20. Y. Maruyama. Higher-order fuzzy logics and their categorical semantics: Higher-order linear completeness and Baaz translation via substructural tripos theory. In *30th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2021, Luxembourg, July 11-14, 2021*, pages 1–6. IEEE, 2021.
21. J. Medina, M. Ojeda-Aciego, and P. Vojtás. Similarity-based unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146(1):43–62, 2004.
22. G. C. Milanese and G. Pasi. Similarity-based reasoning with order-sorted feature logic. *IEEE Trans. Fuzzy Syst.*, 32(5):2797–2810, 2024.
23. D. Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. *J. Log. Comput.*, 1(4):497–536, 1991.
24. D. Miller and G. Nadathur. *Programming with Higher-Order Logic*. Cambridge University Press, 2012.
25. T. Nipkow. Functional unification of higher-order patterns. In *Proceedings of the Eighth Annual Symposium on Logic in Computer Science (LICS '93), Montreal, Canada, June 19-23, 1993*, pages 64–74. IEEE Computer Society, 1993.
26. V. Novák. Elements of model theory in higher-order fuzzy logic. *Fuzzy Sets Syst.*, 205:101–115, 2012.
27. C. Pau. *Symbolic Techniques for Approximate Reasoning*. PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University Linz, Austria, 2022.
28. C. Pau and T. Kutsia. Proximity-based unification and matching for fully fuzzy signatures. In *30th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2021, Luxembourg, July 11-14, 2021*, pages 1–6. IEEE, 2021.
29. S. A. Sandri, J. Mendonça, F. T. Martins-Bedé, R. Guimarães, and O. S. Carvalho. Weighted fuzzy similarity relations in case-based reasoning: A case study in classification. In *FUZZ-IEEE 2012, IEEE International Conference on Fuzzy Systems, Brisbane, Australia, June 10-15, 2012, Proceedings*, pages 1–7. IEEE, 2012.
30. M. I. Sessa. Approximate reasoning by similarity-based SLD resolution. *Theor. Comput. Sci.*, 275(1-2):389–426, 2002.
31. H. Virtanen. Vague domains, S-unification, logic programming. *Electr. Notes Theor. Comput. Sci.*, 66(5):86–103, 2002.
32. P. Vojtás. Declarative and procedural semantics of fuzzy similarity based unification. *Kybernetika*, 36(6):707–720, 2000.