Similarity-based Set Matching

Temur Kutsia Mircea Marin Mikheil Rukhaia

Similarity relations are fuzzy counterparts of equivalence relations. A binary fuzzy relation $\mathcal{R}$ on a set S (a mapping from S to the real interval $[0,1]$) is a similarity relation if it satisfies

Reflexivity: $\mathcal{R}(s,s) = 1$ for all $s \in S$,

Symmetry: $\mathcal{R}(s_1,s_2) = \mathcal{R}(s_2,s_1)$ for all $s_1, s_2 \in S$,

Transitivity: $\mathcal{R}(s_1,s_2) \geq \mathcal{R}(s_1,s) \wedge \mathcal{R}(s,s_2)$,

where $\wedge$ is a T-norm: an associative, commutative, non-decreasing binary operation on [0, 1] with 1 as the unit element. In this paper we assume that the T-norm is minimum (Gödel T-norm). A fuzzy relation is a proximity relation if it is reflexive and symmetric but not necessarily transitive.

Basic operations for many deduction and computational formalisms are matching and unification. These are methods for solving systems of equations. In unification, variables can be replaced in both sides of equations. In matching, it is allowed only in one side. These techniques have been intensively investigated for the crisp (two-valued) case. In the presence of similarity relations, some references about equation solving (including also matching as a special case) are [1, 8–10, 13]. Equational matching and unification are important problems in this area, where equality is considered modulo background theories. However, unlike the crisp case, they have not attracted much attention in the fuzzy setting.

One such background theory is the theory of sets. In this context, a set is represented by a first-order term, called a set-term, using a special function symbol as its constructor. Set unification and set matching problems have been studied by several authors, see, e.g., [2, 3, 5, 6, 11, 12]. It can be also formulated as unification/matching modulo associativity, commutativity, and idempotence of the set constructor, together with its unit element (ACIU-unification/matching). These algorithms have found applications in e.g., deductive databases, theorem proving, static analysis, and rapid software prototyping, just to name a few.

In this paper, we propose extending set matching to similarity relations. In this way, we incorporate some background knowledge into solving techniques with similarity relations. Although our set terms are interpreted as (finite) classical sets, their elements (arguments of set terms) might be related to each other by a similarity relation, which induces also a notion of similarity between set terms. We design a matching algorithm and study its properties. It can be useful in applications where the exact set matching techniques need to be relaxed to deal with quantitative extensions of equality such as similarity relations.

This work can be further extended to several directions. A natural next step would be to allow approximate background knowledge expressed by, e.g., fuzzy sets or rough sets. Another direction would be to generalize the problem from matching to unification. Bringing in multisets together with sets in the theory, generalizing similarity to proximity relations would be also some other interesting extensions to investigate.

We follow [7] and define sets using so called *union-based representation*. We use the singleton constructor $\{\!|\cdot|\!\}$ and union constructor $\cup$, with intended meaning $S \cup T = \{x \mid x \in S \vee x \in T\}$. There is also a special symbol $\emptyset$ to denote the empty set.

Terms are defined by the grammar

$\chi := x \mid X$	common notation for individual and set variables
$\tau := t \mid S$	common notation for individual and set terms
$t := x \mid f(\tau_1, \dots, \tau_n)$	individual terms
$S := X \mid \emptyset \mid \{\!|\tau|\!\} \mid S_1 \cup S_2$	set terms

The symbol $\cup$ is associative, commutative, idempotent, and $\emptyset$ is its unit element. As a compact notation, we introduce $\{\!|\tau_1, \ldots, \tau_n|\!\}$ for $\{\!|\tau_1|\!\} \cup \cdots \cup \{\!|\tau_n|\!\}$ and assume that set terms are kept in the normal form modulo commutativity. Hence, every set term in normal form is either $\emptyset$ or has a form $\{\!|\tau_1, \ldots, \tau_n|\!\} \cup X_1 \cup \cdots \cup X_m$, where each τ_i itself is in normal form, $n, m \geq 0$ and $n + m > 0$. We say that a term is in normal form if all occurrences of set terms in it are in normal form.

Individual variables are denoted by x, y, z, individual terms by t, s, r, set variables by X, Y, Z, set terms by S, T, R, variables by χ, υ, and terms by τ, ϕ, ψ. Substitutions are mappings from individual variables to individual terms and from set variables to set terms that leave all but finitely many variables unchanged. They are extended to terms straightforwardly. We use $\sigma, \vartheta, \varphi$ for substitutions. The identity substitution is denoted by Id.

For a set term $S = \{\!|\tau_1, \ldots, \tau_n|\!\} \cup X_1 \cup \cdots \cup X_m$, we say that the term τ_i for each $1 \leq i \leq n$ belongs to S and write $\tau_i \in S$. We say that two set terms S and T are disjoint if $S = \emptyset$, or $T = \emptyset$, or $S = \{\!|\tau_1, \ldots, \tau_n|\!\} \cup X_1 \cup \cdots \cup X_m$, $T = \{\!|\phi_1, \ldots, \phi_k|\!\} \cup Y_1 \cup \cdots \cup Y_l$, and $\{\tau_1, \ldots, \tau_n\} \cap \{\phi_1, \ldots, \phi_k\} = \emptyset$ and $\{X_1, \ldots, X_m\} \cap \{Y_1, \ldots, Y_l\} = \emptyset$.

Definition 1 (Term similarity). Let $\mathcal{F}$ be a set of all function symbols and $\mathcal{S} = \{\emptyset, \{\!|\,|\!\}, \cup\}$. We assume that similarity relations are defined on the set $\mathcal{F} \cup \mathcal{S}$ so that for any such relation $\mathcal{R}$, we have $\mathcal{R}(f, g) = 0$ if f and g have different arity, and $\mathcal{R}(f, F) = 0$ if $f \in \mathcal{F}$ and $F \in \mathcal{S}$. Moreover, the set $\{g \mid \mathcal{R}(f, g) > 0\}$ is finite for each $f \in \mathcal{F}$.

Similarity relations are extended to terms as follows (terms are assumed in set normal form):

- $\mathcal{R}(\chi, \chi) = 1$.
- $\mathcal{R}(f(\tau_1, \ldots, \tau_n), g(\phi_1, \ldots, \phi_n)) = \mathcal{R}(f, g) \wedge \mathcal{R}(\tau_1, \phi_1) \wedge \cdots \wedge \mathcal{R}(\tau_n, \phi_n)$.
- Let $S = \{\!|\tau_1, \ldots, \tau_n|\!\} \cup X_1 \cup \cdots \cup X_m$ and $T = \{\!|\phi_1, \ldots, \phi_k|\!\} \cup X_1 \cup \cdots \cup X_m$, then

$$\mathcal{R}(S, T) = \bigwedge \Big(\Big\{ \max\{\mathcal{R}(\tau, \phi) \mid \phi \in T\} \mid \tau \in S \Big\} \cup \Big\{ \max\{\mathcal{R}(\phi, \tau) \mid \tau \in S\} \mid \phi \in T \Big\} \Big).$$

- In any other case, $\mathcal{R}(\tau, \phi) = 0$.

A couple of remarks about relating the notions in this paper to other notions:

1. Using $\mathcal{R}$, our set terms can be encoded as fuzzy sets: to each set term S we can associate its fuzzy version S_F. For each term τ, the membership degree of τ in S_F, written as $S_F(\tau)$, can be defined as $\max\{\mathcal{R}(\tau, \phi) \mid \phi \in S\}$.
2. The relation $\mathcal{R}$ can be related to the relation $\mathcal{M}(S_F, T_F) = \inf_\tau \min(I(S_F(\tau), T_F(\tau)), I(T(\tau), S(\tau)))$ (where I is the Lukasiewicz implicator $I(a, b) = \min\{1 - a + b, 1\}$ for all $a, b \in [0, 1]$), which induces a fuzzy similarity relation on fuzzy sets [4].

Example 2. Let $\mathcal{R}(f, g) = 0.9$, $\mathcal{R}(a, b) = \mathcal{R}(b, c) = 0.6$, $\mathcal{R}(a, c) = 0.7$, and $\mathcal{R}(d, e) = 0.8$. Then for the set terms $S = \{\!|a, d, f(x, \{\!|d, e|\!\})|\!\} \cup X$ and $T = \{\!|b, c, d, g(x, \{\!|e|\!\})|\!\} \cup X$ we have (simplifying max-sets):

$$\begin{aligned}
&\mathcal{R}(\{\!|d, e|\!\}, \{\!|e|\!\}) = \bigwedge \big\{ \mathcal{R}(d, e), \mathcal{R}(e, e), \max\{\mathcal{R}(e, d), \mathcal{R}(e, e)\} \big\} = 0.8 \wedge 1 \wedge 1 = 0.8. \\
&\mathcal{R}(f(x, \{\!|d, e|\!\}), g(x, \{\!|e|\!\})) = \mathcal{R}(f, g) \wedge \mathcal{R}(x, x) \wedge \mathcal{R}(\{\!|d, e|\!\}, \{\!|e|\!\}) = 0.9 \wedge 1 \wedge 0.8 = 0.8. \\
&\mathcal{R}(S, T) = \bigwedge \big\{ \max\{\mathcal{R}(a, b), \mathcal{R}(a, c)\}, \mathcal{R}(d, d), \mathcal{R}(f(x, \{\!|d, e|\!\}), g(x, \{\!|e|\!\})), \mathcal{R}(b, a), \mathcal{R}(c, a) \big\} \\
&\qquad = 0.7 \wedge 1 \wedge 0.8 \wedge 0.6 \wedge 0.7 = 0.6.
\end{aligned}$$

Given two terms τ and ϕ, where ϕ does not contain variables, the problem of matching τ to ϕ ($\mathcal{R}$-matching problem of τ to ϕ) is a triple $(\tau, \phi, \mathcal{R})$, which is usually written as $\tau \preceq^?_{\mathcal{R}} \phi$. A substitution σ is a solution of this problem with similarity degree δ if $\mathcal{R}(\tau\sigma, \phi) = \delta > 0$, where $\tau\sigma$ is the term obtained by applying σ to τ.

Example 3. Our matching problems, if they are solvable, usually may have more than one (but finitely many) solutions modulo the ACUI properties of $\cup$. For instance, $\{\!|X, f(a, \{\!|d, x|\!\})|\!\} \cup X \preceq^?_{\mathcal{R}} \{\!|a, \{\!|b, c|\!\}, g(a, \{\!|e|\!\})|\!\}$, where $\mathcal{R}$ is defined in Example 2, has two solutions:

- $\sigma_1 = \{X \mapsto \{\!\!| b, c |\!\!\},\ x \mapsto e\}$ with degree 0.6 because

  $(\{\!\!| X, f(a, \{\!\!| d, x |\!\!\}) |\!\!\} \cup X)\sigma_1 = \{\!\!| \{\!\!| b, c |\!\!\}, f(a, \{\!\!| d, e |\!\!\}) |\!\!\} \cup \{\!\!| b, c |\!\!\} = \{\!\!| \{\!\!| b, c |\!\!\}, f(a, \{\!\!| d, e |\!\!\}), b, c |\!\!\}.$
  $\mathcal{R}(\{\!\!| \{\!\!| b, c |\!\!\}, f(a, \{\!\!| d, e |\!\!\}), b, c |\!\!\},\ \{\!\!| a, \{\!\!| b, c |\!\!\}, g(a, \{\!\!| e |\!\!\}) |\!\!\}) =$

  $\bigwedge \{\mathcal{R}(\{\!\!| b, c |\!\!\}, \{\!\!| b, c |\!\!\}), \mathcal{R}(f(a, \{\!\!| d, e |\!\!\}), g(a, \{\!\!| e |\!\!\})), \mathcal{R}(b, a), \mathcal{R}(c, a), \max\{\mathcal{R}(a, b), \mathcal{R}(a, c)\}\} =$
  $1 \wedge 0.8 \wedge 0.6 \wedge 0.7 \wedge 0.7 = 0.6.$

- $\sigma_2 = \{X \mapsto \{\!\!| a |\!\!\},\ x \mapsto e\}$ with degree 0.6 because

  $(\{\!\!| X, f(a, \{\!\!| d, x |\!\!\}) |\!\!\} \cup X)\sigma_2 = \{\!\!| \{\!\!| a |\!\!\}, f(a, \{\!\!| d, e |\!\!\}) |\!\!\} \cup \{\!\!| a |\!\!\} = \{\!\!| \{\!\!| a |\!\!\}, f(a, \{\!\!| d, e |\!\!\}), a |\!\!\}.$
  $\mathcal{R}(\{\!\!| \{\!\!| a |\!\!\}, f(a, \{\!\!| d, e |\!\!\}), a |\!\!\},\ \{\!\!| a, \{\!\!| b, c |\!\!\}, g(a, \{\!\!| e |\!\!\}) |\!\!\}) =$

  $\bigwedge \{\mathcal{R}(\{\!\!| a |\!\!\}, \{\!\!| b, c |\!\!\}), \mathcal{R}(f(a, \{\!\!| d, e |\!\!\}), g(a, \{\!\!| e |\!\!\})), \mathcal{R}(a, a)\} = 0.6 \wedge 0.8 \wedge 1 = 0.6.$

The problem $X \preceq^?_{\mathcal{R}} \{\!\!| a, c |\!\!\}$ has seven solutions, each obtained from a nonempty subset of $\{a, b, c\}$. Among them, those solutions that contain b have degree 0.6 (e.g. $\mathcal{R}(X\{X \mapsto \{\!\!| b |\!\!\}\}, \{\!\!| a, c |\!\!\}) = \mathcal{R}(\{\!\!| b |\!\!\}, \{\!\!| a, c |\!\!\}) = \bigwedge\{\max\{\mathcal{R}(b, a), \mathcal{R}(b, c)\}, \mathcal{R}(a, b), \mathcal{R}(c, b)\} = 0.6 \wedge 0.6 \wedge 0.6$), the degree of $\{X \mapsto \{\!\!| c |\!\!\}\}$ is 0.7, and the degree of $\{X \mapsto \{\!\!| a, c |\!\!\}\} = 1$.

Configurations are triples of the form $\mathcal{E}; \sigma; \alpha$, where $\mathcal{E}$ is a set of matching equations, σ is a substitution, and $\alpha \in (0, 1]$. To solve a matching problem $\tau \preceq^?_{\mathcal{R}} \phi$, we create what is called the initial configuration $\{\tau \preceq^?_{\mathcal{R}} \phi\}; Id; 1$ and apply non-deterministically the rules below.[1] The rule Gr has the priority: if it is applicable, the others are not applied. The algorithm terminates when there is no applicable rule. If in that case, the final configuration is $\emptyset; \sigma; \alpha$ (i.e. it is left empty), then σ is a solution of the given matching problem with approximation degree α.

Gr: $\{\tau \preceq^?_{\mathcal{R}} \phi\} \uplus \mathcal{E}; \sigma; \alpha \Longrightarrow \mathcal{E}; \sigma; \alpha \wedge \beta$, where τ is a ground term and $\mathcal{R}(\tau, \phi) = \beta > 0$.

Dec-I: $\{f(\tau_1, \ldots, \tau_n) \preceq^?_{\mathcal{R}} g(\phi_1, \ldots, \phi_n)\} \uplus \mathcal{E}; \sigma; \alpha \Longrightarrow \{\tau_i \preceq^?_{\mathcal{R}} \phi_i \mid 1 \le i \le n\} \cup \mathcal{E}; \sigma; \alpha \wedge \beta$,
where $\mathcal{R}(f, g) = \beta > 0$.

Sol-I: $\{x \preceq^?_{\mathcal{R}} \phi\} \uplus \mathcal{E}; \sigma; \alpha \Longrightarrow \mathcal{E}\vartheta; \sigma\vartheta; \alpha$, where $\vartheta = \{x \mapsto \phi\}$.

Dec-S1: $\{S_1 \cup S_2 \preceq^?_{\mathcal{R}} T_1 \cup T_2\} \uplus \mathcal{E}; \sigma; \alpha \Longrightarrow \{S_1 \preceq^?_{\mathcal{R}} T_1, S_2 \preceq^?_{\mathcal{R}} T_2\} \cup \mathcal{E}; \sigma; \alpha$,
where S_1 and S_2 are disjoint and $S_i \neq \emptyset$, $i \in \{1, 2\}$.

Dec-S2: $\{\{\!\!| \tau |\!\!\} \preceq^?_{\mathcal{R}} \{\!\!| \phi |\!\!\}\} \uplus \mathcal{E}; \sigma; \alpha \Longrightarrow \{\tau \preceq^?_{\mathcal{R}} \phi\} \cup \mathcal{E}; \sigma; \alpha$.

Sol-S: $\{X \preceq^?_{\mathcal{R}} T\} \uplus \mathcal{E}\vartheta; \sigma; \alpha \Longrightarrow \mathcal{E}; \sigma\vartheta; \alpha$, where $\vartheta = \{X \mapsto T\}$.

Matching has many useful and important applications. It is used in query answering systems, where query contains variables and database information can be represented as a ground term. In the following example we demonstrate how matching can solve a graph coloring problem.

Example 4. Let us consider the following graph coloring problem:

$$\{\!\!| \{\!\!| x_1, \mathit{pink} |\!\!\}, \{\!\!| x_2, \mathit{rose} |\!\!\}, \{\!\!| x_1, x_2 |\!\!\} |\!\!\} \cup X \preceq^?_{\mathcal{R}} \{\!\!| \{\!\!| \mathit{red}, \mathit{green} |\!\!\}, \{\!\!| \mathit{green}, \mathit{blue} |\!\!\}, \{\!\!| \mathit{pink}, \mathit{blue} |\!\!\} |\!\!\}$$

and assume a similarity relation $\mathcal{R}(\mathit{pink}, \mathit{rose}) = 0.8$, $\mathcal{R}(\mathit{red}, \mathit{rose}) = 0.7$ and $\mathcal{R}(\mathit{red}, \mathit{pink}) = 0.7$.

We start with the Dec-S1 rule and since the algorithm is nondeterministic, there are many parallel computations leading (using the Sol-S rule) to $\vartheta = \{X \mapsto \emptyset\}$, or $\vartheta = \{X \mapsto \{\!\!| \mathit{red}, \mathit{green} |\!\!\}\}$, or $\vartheta = \{X \mapsto \{\!\!| \mathit{green}, \mathit{blue} |\!\!\}\}$, or $\vartheta = \{X \mapsto \{\!\!| \{\!\!| \mathit{red}, \mathit{green} |\!\!\}, \{\!\!| \mathit{green}, \mathit{blue} |\!\!\} |\!\!\}\}, \ldots$. In fact, all paths except $\vartheta = \{X \mapsto \emptyset\}$ ends with failure.

Now, we again apply Dec-S1 rule and the only path leading to a solution is the partitioning

$$\{\!\!| \{\!\!| x_1, \mathit{pink} |\!\!\}, \{\!\!| x_2, \mathit{rose} |\!\!\} |\!\!\} \preceq^?_{\mathcal{R}} \{\!\!| \{\!\!| \mathit{red}, \mathit{green} |\!\!\}, \{\!\!| \mathit{pink}, \mathit{blue} |\!\!\} |\!\!\} \text{ and } \{\!\!| x_1, x_2 |\!\!\} \preceq^?_{\mathcal{R}} \{\!\!| \mathit{green}, \mathit{blue} |\!\!\}$$

[1] $\uplus$ stands for disjoint union.

Next, applying the Dec-S1, Dec-S2 and Sol-I rules to the second equation gives us substitutions $\sigma_1 = \{x_1 \mapsto green, x_2 \mapsto blue\}$ or $\sigma_2 = \{x_1 \mapsto blue, x_2 \mapsto green\}$. Assume $\sigma = \sigma_1$, then we get:

$$\{\!\!\{\{\!\!\{green, pink\}\!\!\}, \{\!\!\{blue, rose\}\!\!\}\}\!\!\} \preceq^?_{\mathcal{R}} \{\!\!\{\{\!\!\{red, green\}\!\!\}, \{\!\!\{pink, blue\}\!\!\}\}\!\!\}$$

Repeated application of the Dec-S1 and Dec-S2 rules leads to (again we consider non-failing paths only)

$$\{green \preceq^?_{\mathcal{R}} green,\ blue \preceq^?_{\mathcal{R}} blue,\ pink \preceq^?_{\mathcal{R}} rose,\ pink \preceq^?_{\mathcal{R}} red\}$$

Now, we apply the Dec-I rule to these equations. the first and second equations does not change $\alpha = 1$, the third one reduces it to $\alpha = 1 \wedge 0.8 = 0.8$ and the last equation fixes $\alpha = 0.8 \wedge 0.7 = 0.7$. Thus we obtain the solution $\sigma = \{x_1 \mapsto green, x_2 \mapsto blue\}$ with approximation degree 0.7. In the same way, the path $\sigma = \sigma_2$ leads to the solution $\sigma = \{x_1 \mapsto blue, x_2 \mapsto green\}$ with approximation degree 0.7.

Acknowledgement. Supported by Shota Rustaveli National Science Foundation of Georgia, 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] N. Arni, S. Greco, and D. Saccà. Set-term matching in logic programming. In J. Biskup and R. Hull, editors, *Database Theory - ICDT'92, 4th International Conference, Berlin, October 14-16, 1992, Proceedings*, volume 646 of *Lecture Notes in Computer Science*, pages 436–449. Springer, 1992.

[3] N. Arni, S. Greco, and D. Saccà. Matching of bounded set terms in the logic language LDL++. *J. Log. Program.*, 27(1):73–87, 1996.

[4] I. Beg and S. Ashraf. Fuzzy similarity and measure of similarity with Lukasiewicz implicator. *New Mathematics and Natural Computation*, 04(02):191–206, 2008.

[5] A. Dovier, C. Piazza, E. Pontelli, and G. Rossi. Sets and constraint logic programming. *ACM Trans. Program. Lang. Syst.*, 22(5):861–931, 2000.

[6] A. Dovier, A. Policriti, and G. Rossi. A uniform axiomatic view of lists, multisets, and sets, and the relevant unification algorithms. *Fundam. Informaticae*, 36(2-3):201–234, 1998.

[7] A. Dovier, E. Pontelli, and G. Rossi. Set unification. *Theory and Practice of Logic Programming*, 6(6):645–701, 2006.

[8] F. A. Fontana and F. Formato. A similarity-based resolution rule. *Int. J. Intell. Syst.*, 17(9):853–872, 2002.

[9] 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.

[10] F. Formato, G. Gerla, and M. I. Sessa. Similarity-based unification. *Fundam. Informaticae*, 41(4):393–414, 2000.

[11] D. Kapur and P. Narendran. NP-completeness of the set unification and matching problems. In J. H. Siekmann, editor, *8th International Conference on Automated Deduction, Oxford, England, July 27 - August 1, 1986, Proceedings*, volume 230 of *Lecture Notes in Computer Science*, pages 489–495. Springer, 1986.

[12] D. Kapur and P. Narendran. Complexity of unification problems with associative-commutative operators. *J. Autom. Reason.*, 9(2):261–288, 1992.

[13] M. I. Sessa. Approximate reasoning by similarity-based SLD resolution. *Theor. Comput. Sci.*, 275(1-2):389–426, 2002.