This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

Similarity-Based Reasoning with Order-Sorted Feature Logic

Gian Carlo Milanese, Gabriella Pasi

***Abstract*—Order-Sorted Feature (OSF) logic is a knowledge representation and reasoning language based on sorts – symbols that denote concepts ordered in a subsumption relation – and features – symbols that denote functional attributes. Reasoning with OSF logic is based on the unification of OSF terms, record-like structures that denote classes of objects and that are themselves ordered in a subsumption relation. OSF term unification aims to combine the constraints expressed by two terms in a consistent way, and it takes into account the subsumption relation between sort symbols, providing an efficient calculus of type subsumption. This paper presents an approach to define approximate reasoning with OSF logic by extending its language with a similarity relation on sorts. In order for the OSF term unification algorithm to take into account this similarity and its interaction with the subsumption relation, we propose to combine the two relations into a single fuzzy subsumption relation. The advantage is that the same unification rules of OSF logic can then be applied to this fuzzy setting. We conclude by discussing potential applications of OSF logic extended with a sort similarity relation.**

***Index Terms*—Approximate reasoning, Fuzzy unification, Similarity-based reasoning, Order-Sorted Feature logic, Knowledge representation**

I. Introduction

APPROXIMATE reasoning based on fuzzy relations – in particular similarity relations – has been researched extensively in fuzzy logic programming. Early work includes Ying's logic for approximate reasoning [1] and the first papers on similarity-based logic programming [2]–[5]. One motivation behind the similarity-based approaches is to model a form of reasoning that may be referred to as *reasoning by analogy or similarity* [5]. For instance, this may be achieved by relaxing the *equality* constraint on two functor symbols to a flexible constraint of similarity, when unifying two first-order terms (FOTs). For example, if the functors `thriller` and `horror` are assumed to denote similar concepts, then the term `thriller(X)` can unify with the term `horror("Psycho")` to some extent (degree), thus leading to approximate (not exact but similar) solutions to a query posed to a knowledge base. This kind of relaxed unification is generally referred to as *weak unification* (e.g., [5]).

This research line has been extended in several ways, including approaches that support multiple similarity relations [6], proximity relations [7]–[9], or related operations like matching and anti-unification [10], [11]. Moreover, weak unification has been implemented in fuzzy logic programming systems such as FASILL [12] and Bousi∼Prolog [13], [14], which has been employed in several applications such as text classification [15], [16], linguistic feedback in computer games [17], decision making [18], [19], and knowledge discovery [20]. Aït-Kaci and Pasi [21] have presented a procedure for weak unification that, besides tolerating different (but similar) functor symbols, also allows the unification of FOTs with a different number and possibly a different order of arguments. This work has been generalized to proximity relations [22], and a possible incorporation in Bousi∼Prolog has been proposed [23].

The work by Aït-Kaci and Pasi [21] was preliminary towards the definition of similarity-based reasoning with Order-Sorted Feature (OSF) logic, a knowledge representation and reasoning language developed by Aït-Kaci [24], which has found applications in constraint logic programming and computational linguistics (e.g., [25]). The interest in defining a similarity-based extension of OSF logic is that previous work has proved the efficiency of this language for knowledge representation and reasoning. For instance, one advantage of OSF logic is that its unification algorithm takes into account a *subsumption* (is-a) ordering between sorts, which enables a single unification step to potentially replace several resolution steps, possibly leading to more efficient computations [26], [27]. Moreover, it has been shown that, at the time of the experiments, the CEDAR Semantic Web reasoner based on OSF logic was consistently among the best reasoners in terms of concept classification, and several orders-of-magnitude more efficient in terms of terminological reasoning [28], [29]. Introducing a similarity relation to augment the flexibility of OSF logic, while preserving its efficiency, has the potential to significantly enhance the effectiveness and applicability of this framework, particularly in domains like the Semantic Web, where efficiency is of paramount concern.

In this work we show how to make the OSF term unification algorithm more flexible by considering a *similarity relation between sorts* besides a sort subsumption ordering. Rather than devising ad-hoc unification rules that deal with the similarity relation and its interaction with the sort subsumption ordering, we propose to *combine the two relations into a single fuzzy subsumption relation*. Intuitively, this is achieved by applying the following informal inference, inspired by the similarity-based approaches to logic programming (e.g., [5]):

$$\frac{\text{If the sort } s_0 \text{ is subsumed by the sort } s_1 \quad \text{and } s_1 \text{ is similar to the sort } s_2 \text{ with degree } \alpha}{\text{then } s_0 \text{ is subsumed by } s_2 \text{ with degree } \alpha.}$$

For example, if $slasher$s are $horror$ movies, and $horror$

Gian Carlo Milanese and Gabriella Pasi are with the Information and Knowledge Representation, Retrieval and Reasoning Lab, University of Milano-Bicocca, Milan, Italy. (e-mail: giancarlo.milanese@unimib.it; gabriella.pasi@unimib.it)

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

movies are similar to $thriller$s with degree 0.5, then we can conclude that $slasher$s are also $thriller$s with subsumption degree 0.5. As a consequence, queries aimed to retrieve $thriller$s from a knowledge base may also retrieve instances of $slasher$ (associated with an approximation degree), thus improving the flexibility of the retrieval process. Intuitively, the fuzzy subsumption relation encodes the information of both the crisp subsumption and the similarity. This procedure shifts the setting to that of *OSF logic with a fuzzy sort subsumption relation*, or *fuzzy OSF logic*, whose semantics has already been developed in [30]. The advantage is that in fuzzy OSF logic it is possible to apply the same unification algorithm of (crisp) OSF logic, with essentially the same computational cost, thereby retaining the efficiency inherent to this logical framework.

OSF logic [24] and fuzzy OSF logic [30] are briefly presented in Section II.

In Section III we argue why a similarity-based unification procedure for OSF terms should take into account the interaction between the sort subsumption and the sort similarity relations, and why combining them into a single fuzzy subsumption relation is an effective way to deal with this issue.

Section IV-A focuses on the issue of how to formally define a fuzzy subsumption relation – which should be a fuzzy partial order, or even a fuzzy lattice – starting from a (crisp) subsumption and a fuzzy similarity relation. We start by combining the two relations into a *fuzzy subsumption preorder*, which may however contain fuzzy subsumption cycles. This issue is discussed in Section IV-B, where we present a construction of a fuzzy partial order from a fuzzy preorder that generalizes a well-known result from order theory. In Section IV-C we also propose a definition of a completion of a fuzzy partial order into a fuzzy lattice, thus finalizing the transformation of a subsumption relation and a similarity into a fuzzy subsumption relation. In other words, Section IV ensures that the process of combining a crisp subsumption and a fuzzy similarity according to the intuition outlined above is sound, i.e., it indeed leads to a fuzzy lattice, as required by the unification rules of fuzzy OSF logic.

Finally, in Section V we discuss two potential applications of similarity-based reasoning with OSF logic: (i) an extension of the CEDAR reasoner [28], [29] which relies on a sort similarity relation to approximately answer queries posed to a knowledge base, and (ii) a fuzzy logic programming language based on OSF terms, which leverages a similarity relation between sorts, comparable to a similarity-based extension of the language LOGIN [26].

II. OSF Logic and Fuzzy OSF Logic

A. Order-Sorted Feature Logic

Order-Sorted Feature (OSF) logic is a Knowledge Representation and Reasoning language that originates in Hassan Aït-Kaci's work [31] and, similarly to Description Logics (DLs), it was initially meant as a formalization of Ron Brachman's structured inheritance networks [32], [33]. OSF logic and related formalisms – e.g., feature logic [34] or the logic of typed feature structures [25] – have been applied in computational linguistics [25] and implemented in constraint logic programming languages such as LOGIN [26], LIFE [24] and CIL [35] and, more recently, in the CEDAR Semantic Web reasoner [28], [29].

Due to their common origin, OSF Logic [24] and DLs [32] share a few similarities. Both formalisms are subsets of first-order logic designed to simplify its language in order to achieve computational tractability, while still providing enough expressive power for effective knowledge representation and reasoning. DLs and OSF logic are both based on set-denoting symbols (concepts and sorts, respectively) and symbols for expressing attributes: *relational roles* for DL, and *functional features* for OSF Logic. One of the most significant distinguishing aspects is that the semantics of OSF logic is based on the *closed world assumption*: for instance, if two sorts do not share a common subsort, then they are assumed to be disjoint. The relationship between the two languages has been explored thoroughly in works such as [33], [36]–[38].

As its name suggests, OSF logic is based on *sorts*, symbols that represent sets of objects such as $person$ or $movie$ and that are ordered by a subsumption relation, and *features*, symbols that denote functional attributes such as $title$ or $directed\_by$. These symbols are specified by an OSF signature, which is defined as follows.

Definition II.1 (OSF signature [24]). An *OSF signature* is a tuple $(\mathcal{S}, \mathcal{F}, \preceq)$ such that

1) $\mathcal{S}$ is a finite set of *sort symbols* (or *sorts*),
2) $\mathcal{F}$ is a finite set of *feature symbols* (or *features*),
3) $(\mathcal{S}, \preceq)$ is a finite lattice with least element $\bot$ and greatest element $\top$.

The greatest lower bound (GLB) of two sorts s and s' is denoted by $s \wedge s'$.

Sorts and features can be used to construct record-like structures called *OSF terms* that can describe more complex classes of objects.

Definition II.2 (OSF terms [24]). Let $\mathcal{V}$ be a countably infinite set of variables (or *coreference tags*, or simply *tags*). Let $X \in \mathcal{V}$, $s \in \mathcal{S}$ and $f_1, \ldots, f_n \in \mathcal{F}$. An *OSF term* is defined recursively as follows.

- A sorted variable $X : s$ is an OSF term.
- If $t_1, \ldots, t_n$ are OSF terms, then an attributed sorted variable $t = X : s(f_1 \to t_1, \ldots, f_n \to t_n)$ is an OSF term.

The set of variables occurring in t is denoted $Tags(t)$. The variable X is called the *root tag* of t and is denoted $Root(t)$.

Example II.3 (OSF terms). The following OSF term denotes the class of movies that are written and directed by the same person:

$$X : movie \begin{pmatrix} directed\_by \to Y : person, \\ written\_by \to Y \end{pmatrix}.$$

The variable Y is used as a coreference tag, i.e., it specifies that the values of the features $directed\_by$ and $written\_by$ should be the same. The variables in an OSF term are often omitted unless they are necessary to express this property. Any

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

sort s can be seen as an OSF term $X : s$ in which the variable is left implicit.

The definition of OSF terms given above does not rule out the presence of redundant or even contradictory information (e.g., consider the OSF term $s(f \to s_0, f \to s_1)$, which is contradictory if $s_0 \wedge s_1 = \bot$). OSF terms that are well-behaved to this regard are called *normal OSF terms* or *ψ-terms* [24], and they are also denoted ψ, ψ_i, and so on. Normal OSF terms are ordered in a subsumption lattice which extends the one on sort symbols[1] [24].

Reasoning with OSF logic is based on a unification procedure which aims to combine the constraints expressed by two OSF terms in a consistent way [24]. This is achieved by first translating the two terms into *OSF clauses*, an equivalent representation for OSF terms which is defined next.

Definition II.4 (OSF constraints and clauses [24]). An *OSF constraint* is an expression of the form $X : s$, $X \doteq X'$, or $X.f \doteq X'$. An *OSF clause* ϕ is a conjunction of OSF constraints. The set of variables occurring in ϕ is denoted $Tags(\phi)$, while $\phi[X/Y]$ is the OSF clause obtained by replacing all occurrences of Y with X.

Informally, the constraint $X : s$ means that the value assigned to X is of sort s; $X \doteq X'$ means that the same value is assigned to the variables X and X'; while $X.f \doteq X'$ means that applying the feature f to the value assigned to X returns the value assigned to X'. Every OSF term t can be translated into an equivalent OSF clause $\phi(t)$ [24].

Example II.5 (OSF clause). The OSF clause syntax for the OSF term of Example II.3 is $X : movie \;\&\; X.directed\_by \doteq Y \;\&\; Y : person \;\&\; X.written\_by \doteq Y$.

Fig. 2 shows the OSF constraint normalization rules [24] needed to unify two OSF terms. Each rule expresses that, whenever the (optional) condition in square brackets holds, the expression above the line can be simplified into the one below.

Proposition II.6 (OSF term unification [24]). Two OSF terms ψ_1 and ψ_2 can be unified by non-deterministically applying any applicable constraint normalization rule to the clause $\phi(\psi_1) \;\&\; \phi(\psi_2) \;\&\; Root(\psi_1) \doteq Root(\psi_2)$ until none applies. The resulting clause can be translated back into an OSF term ψ, called the unifier of ψ_1 and ψ_2 (the term $X : \bot$ if the unification fails). The term ψ is the GLB of ψ_1 and ψ_2 and is denoted $\psi_1 \wedge \psi_2$.

Example II.7 (OSF term unification). Consider the subsumption relation of Fig. 4a[2] and the terms $t_1 = movie(directed\_by \to person, genre \to horror)$ and $t_2 = movie(genre \to slasher)$. Their unifier is the term $t_3 = movie(directed\_by \to person, genre \to slasher)$, which combines the constraints of t_1 and t_2 in a consistent way.

[1]More precisely, the subsumption lattice is defined on equivalence classes of normal OSF terms, similarly to how first-order terms are partially ordered modulo variable renaming.

[2]A subsumption relation $\preceq$ is represented graphically as a directed acyclic graph, of which $\preceq$ is the reflexive and transitive closure. The bottom element $\bot$ is not drawn.

```
s2(X) :- s1(X).
...
sn(X) :- sn-1(X).
s1(a).
prop(a).
```

(a)

```
s1 < s2.
...
sn-1 < sn.
{a} < s1.
prop(a).
```

(b)

Fig. 1: Logic programs of Example II.8. (a) Prolog version. (b) OSF version.

Note that t_3 is subsumed by both t_1 and t_2, and it is in fact their GLB. In particular, this is because t_3 has the same root sort as the two other terms, and its values for the features $directed\_by$ and $genre$ are subsumed by those of t_1 and t_2.

An advantage of OSF term unification is that it takes into account a subsumption relation on sorts. This may cause a single unification step to potentially replace several resolution steps, as the following example from [26] shows.

Example II.8 (Prolog resolution and OSF unification). Consider the Prolog program of Fig. 1a. The query `?- sn(X), prop(X)` will require n resolution steps before matching `X = a`. The OSF version of the same program is depicted in Fig. 1b. The program involves declarations of shape `s < s'` for the sort subsumption relation, and of shape `{a} < s` for the instances of the sort symbols. Constants such as `a` are treated as singleton sorts (sorts that denote a single element), and thus as OSF terms themselves. The symbol `prop` is a predicate symbol which takes OSF terms as arguments. Since `a` is subsumed by `sn`, now the query `prop(X:sn)` (which aims to retrieve individuals `X` of sort `sn` and that satisfy the property `prop`) succeeds in a single unification step rather than n resolution steps.

The reader is referred to [24] for more details about OSF logic and its semantics.

Sort Intersection

$$\frac{\phi \;\&\; X : s \;\&\; X : s'}{\phi \;\&\; X : s \wedge s'}$$

Tag Elimination

$$\frac{[Y \in Tags(\phi)] \quad \phi \;\&\; X \doteq Y}{\phi[X/Y] \;\&\; X \doteq Y}$$

Inconsistent Sort

$$\frac{\phi \;\&\; X : \bot}{X : \bot}$$

Feature Functionality

$$\frac{\phi \;\&\; X.f \doteq Y \;\&\; X.f \doteq Y'}{\phi \;\&\; X.f \doteq Y \;\&\; Y \doteq Y'}$$

Fig. 2: OSF constraint normalization rules

B. Fuzzy Order-Sorted Feature Logic

Fuzzy OSF logic [30] generalizes OSF logic by considering a *fuzzy subsumption relation* between sorts and by interpreting sorts as *fuzzy sets*. More precisely, the set $\mathcal{S}$ of sorts is ordered by a fuzzy lattice[3] $\preceq : \mathcal{S} \times \mathcal{S} \to [0, 1]$, and the meaning of

[3]The definitions of fuzzy partial orders and lattices are recalled in Appendix A.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

a sort $s \in \mathcal{S}$ in an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a fuzzy set $s^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0,1]$, where $\Delta^{\mathcal{I}}$ is the domain or universe of the interpretation. We rely on the minimum as a t-norm (denoted $\wedge$) and on the maximum as a t-conorm (denoted $\vee$).

While in the crisp setting the subsumption $s \preceq s'$ denotes the inclusion of the sets denoted by s and s', we consider a fuzzy subsumption relation as a way to model a weaker notion of inclusion, which requires that every instance of $s^{\mathcal{I}}$ must also be an instance of $s'^{\mathcal{I}}$, but possibly with a lower membership degree. Thus, we define a fuzzy subsumption relation as a fuzzy lattice $\preceq : \mathcal{S}^2 \to [0,1]$ that associates a subsumption degree β with each pair of sorts $s, s' \in \mathcal{S}$ and having the following semantics:

$$\text{if } \preceq(s, s') = \beta, \text{ then } \forall d \in \Delta^{\mathcal{I}} : s^{\mathcal{I}}(d) \wedge \beta \leq s'^{\mathcal{I}}(d).$$

That is, any object d which is an instance of $s^{\mathcal{I}}$ with degree β' must also be an instance of $s'^{\mathcal{I}}$ with a degree that is *greater than or equal to the minimum of* β and β'. Note that Zadeh's inclusion of fuzzy sets [39] is a special case of this constraint where $\beta = 1$. An example of a fuzzy sort subsumption relation is given in Fig. 4c[4].

The interpretation of sorts as fuzzy sets can be extended to OSF terms, so that the denotation of an OSF term t in an interpretation $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a fuzzy set $[\![t]\!]^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0,1]$. Analogously, the fuzzy sort subsumption relation can be extended to a fuzzy subsumption between OSF terms, which is a fuzzy lattice on (equivalence classes of) OSF terms [30].

Like their crisp counterparts, fuzzy OSF logic and fuzzy DLs (e.g., [40], [41]) share a few common aspects. For instance, fuzzy DL concepts are also interpreted as fuzzy subsets of a domain. On the other hand, fuzzy DLs interpret roles as fuzzy relations, while fuzzy OSF logic maintains a crisp interpretation for feature symbols. Moreover, the notion of fuzzy subsumption defined above for fuzzy OSF logic differs significantly with that of graded subsumption in fuzzy DLs, which relies on the fuzzy implication operator.

With respect to computational complexity, the GLB of two OSF terms in the fuzzy subsumption lattice can be determined via their unification, through a constraint normalization procedure which is essentially the same as the one for crisp OSF logic [30]. The only difference involves the rule Sort Intersection, which in the fuzzy setting relies on the computation of the GLB of two sorts in a fuzzy lattice rather than a crisp one. Because of Proposition A.15, however, this computation can be reduced to the crisp setting, so that unifying two terms ψ_1 and ψ_2 in fuzzy OSF logic has the same complexity as in crisp OSF logic, which is $O(mG(m))$, where $m = |Tags(\psi_1) \cup Tags(\psi_2)|$ and the growth rate of the function G is of the order of an inverse of the Ackermann function ($G(m) \leq 5$ for all practical purposes) [26].

In fuzzy OSF logic, the unifier of two OSF terms is also associated with a unification degree, whose computation requires $O(m(|\mathcal{S}|+e))$ time, where $m = |Tags(\psi_1) \cup Tags(\psi_2)|$ and e is the number of edges in the graph representation of the fuzzy sort subsumption relation [30]. This computation depends on determining the subsumption degree $\preceq(s, s')$ between pairs of sorts. This operation can be performed with an approach that is analogous to solving the shortest paths problem in a directed acyclic graph, and that can be optimized through the application of graph encoding techniques [42].

Example II.9 (Fuzzy OSF term unification). Consider the fuzzy subsumption relation $\preceq$ represented in Fig. 4c. By employing the same constraint normalization rules of crisp OSF logic (with the only difference being that the GLB in the rule Sort Intersection is computed in a fuzzy lattice) it is possible, for example, to unify the terms $t_1 = movie(genre \to horror)$ and $t_2 = movie(genre \to thriller)$, resulting in the unifier $movie(genre \to slasher)$. The unifier is subsumed by t_1 with degree 1 (since $\preceq(slasher, horror) = 1$), and by t_2 with degree 0.5 (since $\preceq(slasher, thriller) = 0.5$). We thus say that the unification degree of t_1 and t_2 is 0.5.

III. Similarity-Based OSF Logic, Informally

```
horror(X) :- slasher(X).
slasher("Psycho").
thriller("Memento").
```

Fig. 3: Logic program of Example III.1

In a standard setting, two first-order terms can be unified if there is a substitution that makes them *equal*. A mismatch of functor symbols such as f and g in the terms $t_1 \stackrel{\text{def}}{=} f(X, a)$ and $t_2 \stackrel{\text{def}}{=} g(b, Y)$ causes their unification to fail. With similarity-based unification (e.g., [5]), it is possible to *weakly* unify the terms t_1 and t_2 provided that $f \sim_\alpha g$ with $\alpha \in (0,1]$. In this case the two terms *weakly unify with approximation degree* α. A similarity can also be considered when resolving clauses, leading to similarity-based SLD resolution [5].

Example III.1 (Similarity-based SLD resolution). Consider, for example, the Prolog program of Fig. 3 and assume that `horror` and `thriller` are similar to degree 0.5. Then the query `?- thriller(X)` will return, besides `X = "Memento"`, also the solution `X = "Psycho"` with approximation degree 0.5. This is due to the fact that, thanks to the similarity relation, the query will resolve with the first clause of the program, leading to an approximate solution. Intuitively, since `"Psycho"` is a `horror` movie as a consequence of the first rule, and `horror` movies are similar to `thrillers`, then, to some degree, `"Psycho"` is also a `thriller`.

As seen in Section II-A, the unification of two OSF terms aims to combine the constraints expressed by the two terms in a consistent way into a single term. This procedure takes the sort subsumption relation $\preceq$ into account, so that it is possible, for instance, to unify the term $t_1 \stackrel{\text{def}}{=} s(f_1 \to s_1)$ with the term $t_2 \stackrel{\text{def}}{=} s'(f_2 \to s_2)$ even if their root sorts s and s' are different, as long as $s \wedge s' \neq \bot$. Moreover, as shown in Example II.8, this feature can allow a single unification step

[4]A fuzzy relation R is represented graphically as a weighted directed graph, of which R is the reflexive and transitive closure (see Definition A.9 in Appendix A). Edges with weight 1 are not labeled, and the bottom element $\bot$ is not drawn.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

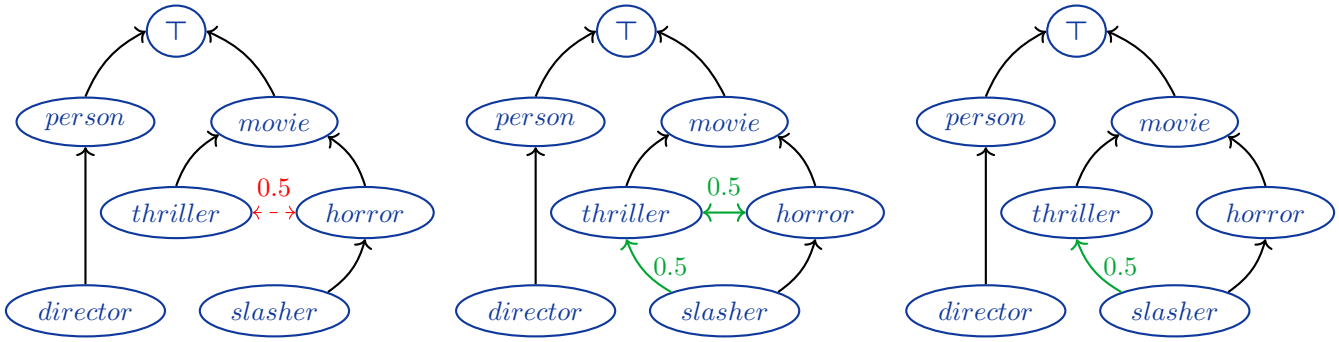

Fig. 4: Examples of (fuzzy) relations. (a) A subsumption (black) and a similarity (red, dashed). (b) A fuzzy preorder. (c) A fuzzy lattice.

```
s1(X) :- s0(X).
s3(X) :- s2(X).
...
s2n+1(X) :- s2n(X).
s0(a).
prop(a).
```

(a)

```
s0 < s1.
s2 < s3.
...
s2n < s2n+1.
{a} < s0.
prop(a).
```

(b)

Fig. 5: Logic programs of Example III.3 (a) Prolog version. (b) OSF version.

to replace several resolution steps, which can lead to more efficient computations.

Our goal is to define a more flexible, or *weaker*, unification procedure for OSF terms that also takes into account a *similarity relation* $\sim$: $\mathcal{S} \times \mathcal{S} \to [0,1]$ on sort symbols. Using this additional information it could be possible, for example, to unify the terms t_1 and t_2 even if $s \wedge s' = \bot$.

Example III.2 (Weak OSF term unification: similar sorts). Consider for example the set of sorts $\mathcal{S}$, the subsumption relation $\preceq \subseteq \mathcal{S}^2$ and the similarity $\sim : \mathcal{S}^2 \to [0,1]$ specified in Fig. 4a. Additionally, consider the sort `"Psycho"`, which only subsumes $\bot$ and is only subsumed by $\top$, and the OSF terms $\psi_1 \stackrel{\text{def}}{=} horror(title \to \texttt{"Psycho"})$ and $\psi_2 \stackrel{\text{def}}{=} thriller(title \to X : \top)$. Clearly the two terms cannot be unified since $horror \wedge thriller = \bot$, but they could be *weakly unified* if we additionally consider that $horror$ and $thriller$ are similar to degree 0.5. Intuitively, as ψ_1 and ψ_2 denote movies of similar genres (and all of the other constraints can be combined in a consistent way), then they could be weakly unified.

Besides matching similar sorts, weak OSF term unification could also consider the interaction between the subsumption and the similarity relations, as the following example shows.

Example III.3 (Weak OSF term unification: subsort of a similar sort). Continuing from Example III.2, consider the term $\psi_3 \stackrel{\text{def}}{=} slasher(title \to \texttt{"Psycho"})$ which unifies with ψ_1 since $slasher \preceq horror$. Moreover, as $horror \sim_{0.5} thriller$, then ψ_3 should also *weakly unify* with ψ_2. In other words, since the genre of ψ_3 is subsumed by a genre ($horror$) which is similar to the genre of ψ_2 (and also all of the other constraints can be combined in a consistent way), then the two terms could be *weakly* unified, achieving a result analogous to Example III.1.

Additionally, similarly to how a single OSF term unification step can replace several resolution steps (as in Example II.8), similarity-based OSF term unification should be capable of replacing several similarity-based resolution steps.

Example III.4 (Similarity-based resolution and weak OSF term unification). Consider the logic program of Fig. 5a and assume that $\sim$ is a similarity relation such that, for each $1 \leq i \leq n$, $\sim(\texttt{s}_{2i-1}, \texttt{s}_{2i}) = \alpha_i > 0$. Then the query `?- s2n+1(X), prop(X)` will require several similarity-based SLD resolution steps before returning the solution `X = a`. Now consider the OSF version of the same program of Fig. 5b and the query `prop(X:s2n+1)`. Because `a`[5] is subsumed by `s1`, which is similar to `s2`, which is subsumed by `s3`, and so on, up to `s2n+1`, then the query should succeed with a single unification of the terms `X:s2n+1` and `a`.

These examples show that a similarity-based notion of OSF term unification should consider *the interaction between the subsumption and similarity relations on sorts*, besides allowing sorts that are equal, similar, or with a non-bottom GLB.

In order to achieve this, we propose to combine the (crisp) subsumption $\preceq$ and the similarity $\sim$ into a single fuzzy subsumption relation $\precsim$. The advantage is that it would then be possible to employ the usual unification algorithm of OSF terms [24], [30], taking into account the fuzzy subsumption $\precsim$ which incorporates the information of both $\preceq$ and $\sim$ at the same time. Intuitively, the combination of $\preceq$ and $\sim$ is inspired by the following informal inference:

$$\frac{\begin{array}{l}\text{If the sort } s_0 \text{ is subsumed by the sort } s_1 \\ \text{and } s_1 \text{ is similar to the sort } s_2 \text{ with degree } \alpha\end{array}}{\text{then } s_0 \text{ is subsumed by } s_2 \text{ with degree } \alpha.}$$

[5]Recall that constants such as `a` are treated as singleton sorts, i.e., sorts that denote a single element.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

For example, considering the setting of Example III.3, since $\mathit{slasher} \preceq \mathit{horror}$ and $\mathit{horror} \sim_{0.5} \mathit{thriller}$ (Fig. 4a), then we could conclude that $\precsim(\mathit{slasher}, \mathit{thriller}) = 0.5$ (Fig. 4c). Thus, it would be possible to unify the terms ψ_1 and ψ_2, since, according to Fig. 4c, $\mathit{horror} \curlywedge \mathit{thriller} = \mathit{slasher}$ (where $\curlywedge$ denotes the GLB of two sorts in the fuzzy subsumption $\precsim$). The unification would result in the term $\mathit{slasher}(\mathit{title} \to \texttt{"Psycho"})$, which is subsumed by ψ_1 with degree 1, and by ψ_2 with degree 0.5 (and thus the overall unification degree is 0.5). In a similar manner, ψ_2 and ψ_3 can be unified with degree 0.5. Analogously, in the context of Example III.4, since

$$\mathtt{a} \preceq \mathtt{s}_1, \quad \sim(\mathtt{s}_1, \mathtt{s}_2) = \alpha_1, \quad \mathtt{s}_2 \preceq \mathtt{s}_3, \quad \sim(\mathtt{s}_3, \mathtt{s}_4) = \alpha_2,$$
$$\ldots, \quad \mathtt{s}_{2n\text{-}2} \preceq \mathtt{s}_{2n\text{-}1}, \quad \sim(\mathtt{s}_{2n\text{-}1}, \mathtt{s}_{2n}) = \alpha_n, \quad \mathtt{s}_{2n} \preceq \mathtt{s}_{2n+1}$$

then by repeatedly applying the same informal inference[6] we obtain $\precsim(\mathtt{a}, \mathtt{s}_{2n+1}) = \alpha = \min_{1 \le i \le n} \alpha_i$, thus enabling the weak unification of $\mathtt{a}$ with $\mathtt{X} : \mathtt{s}_{2n+1}$ with degree α.

The advantage of this approach is that it allows to seamlessly integrate the similarity relation into the unification rules of fuzzy OSF logic, taking the interaction between the similarity and the crisp subsumption into account, potentially allowing a single weak unification to replace several similarity-based SLD resolution steps, all the while maintaining the same computational complexity of crisp OSF logic for the computation of the unifier of two OSF terms. The only additional cost is the construction of the fuzzy subsumption, which is only required once, before the queries are processed.

The following section is devoted to the formal validation of this procedure. Specifically, it demonstrates that the fuzzy subsumption defined by combining a crisp subsumption and a similarity leads indeed to a fuzzy lattice, as required by the unification rules of fuzzy OSF logic.

IV. Similarity-Based OSF Logic, Formally

A. Combining a Subsumption and a Similarity

As a first step towards our formal definition of a fuzzy subsumption relation that combines a (crisp) subsumption relation and a sort similarity, we define a *fuzzy preorder* on sort symbols, which intuitively arises from the iterated application of the informal inference discussed above.

Definition IV.1 (Similarity-subsumption chain and $\precsim$). Let $(\mathcal{S}, \preceq)$ be a partial order, let $\sim\colon \mathcal{S}^2 \to [0,1]$ be a similarity relation, and let $s, s' \in \mathcal{S}$. A *similarity-subsumption chain of strength α* from s to s' is a sequence of sorts

$$s \sim_{\alpha_0} s_0 \preceq s'_1 \sim_{\alpha_1} s_1 \preceq s'_2 \cdots s_{n-1} \preceq s'_n \sim_{\alpha_n} s'$$

where $\alpha = \min_{0 \le i \le n} \alpha_n$. The fuzzy relation $\precsim$ on $\mathcal{S}$ is defined by letting

$$\precsim(s, s') \stackrel{\text{def}}{=} \max \left\{ \alpha \;\middle|\; \begin{array}{l} \text{there is a similarity-subsumption} \\ \text{chain of strength } \alpha \text{ from } s \text{ to } s' \end{array} \right\}.$$

Alternatively, it is possible to define $\precsim$ as the transitive closure of the composition of $\preceq$ and $\sim$.

[6]More precisely, a generalization of the rule where the subsumption in the antecedent is also fuzzy: if $\precsim(s_0, s_1) = \alpha$ and $\sim(s_1, s_2) = \beta$, then $\precsim(s_0, s_2) = \alpha \wedge \beta$.

Proposition IV.2 (Equivalent definition of $\precsim$). Let $(\mathcal{S}, \preceq)$ be a partial order[7], let $\sim$ be a similarity relation on $\mathcal{S}$, and let $\precsim$ be as in Definition IV.1. Then $\precsim = (\preceq \circ \sim)^{\oplus}$.

Example IV.3 (Similarity-subsumption chain). Fig. 4b represents the fuzzy preorder $\precsim$ obtained by combining the subsumption relation $\preceq$ and the similarity $\sim$ of Fig. 4a. The fuzzy subsumption edges added to $\preceq$ are represented in green.

Proposition IV.4 (Similarity-subsumption preorder). The fuzzy relation $\precsim$ of Definition IV.1 is a fuzzy preorder.

B. Dealing with Cycles

It is clear from Example IV.3 that the fuzzy relation $\precsim$ in general is not a fuzzy lattice, and not even a fuzzy partial order, as it is not antisymmetric. One reason is that $|\sim| \subseteq |\precsim|$ and $|\sim|$ is symmetric, and thus $\precsim$ will contain symmetric links (such as $\precsim(\mathit{thriller}, \mathit{horror}) = \precsim(\mathit{horror}, \mathit{thriller}) > 0$ in Example IV.3) that are due directly to the similarity relation. A solution in this case simply consists in deleting such similarity links, which is also justified by the fact that these are not needed anymore for the purpose of unifying two terms (for example, the fact $\sim(\mathit{thriller}, \mathit{horror}) > 0$ has already been used to define $\precsim(\mathit{slasher}, \mathit{thriller}) > 0$, which is enough, for instance, to make the terms ψ_2 and ψ_3 of Examples III.2 and III.3 weakly unifiable). According to this intuition, we define the fuzzy relation $\precapprox$ as follows.

Definition IV.5 (Fuzzy preorder $\precapprox$). Let $\preceq \subseteq \mathcal{S}^2$ be a partial order, $\sim : \mathcal{S}^2 \to [0,1]$ be a similarity relation and $\precsim$ be as in Definition IV.1. The fuzzy relation $\precapprox$ is defined by letting $\precapprox \stackrel{\text{def}}{=} ((\precsim \dot{-} \sim) \cup \preceq)^{\oplus}$, where the difference $\dot{-}$ is defined by letting, for two fuzzy sets F and G,

$$\mu_{F \dot{-} G}(x) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } \mu_G(x) > 0 \\ \mu_F(x) & \text{otherwise.} \end{cases}$$

Note that the fuzzy union of $(\precsim \dot{-} \sim)$ with $\preceq$ is necessary in case $|\sim| \cap \preceq \neq \emptyset$, and the transitive closure is needed to ensure the transitivity of $\precapprox$ in case transitive links are deleted by taking $(\precsim \dot{-} \sim)$.

Example IV.6 (Fuzzy preorder $\precapprox$). Fig. 4c represents the fuzzy relation $\precapprox$ obtained by combining the subsumption $\preceq$ and the similarity $\sim$ – resulting in the relation $\precsim$ represented in Fig. 4b – and then deleting the similarity links as per Definition IV.5. The result in this case is a fuzzy lattice which can be employed, for instance, for the unification of the terms ψ_2 and ψ_3 of Examples III.2 and III.3.

Deleting symmetric links may work for simple cases such as the one of Fig. 4, but in general the relation $\precapprox$ may still not be antisymmetric, as the next example shows.

Example IV.7 (Antisymmetry and $\precapprox$). Consider Figs. 6a, 6b and 6d, where $\mathcal{S} = \{a, b, c, d\}$, $a \preceq b$, $c \preceq d$, $a \sim d = \alpha > 0$ and $b \sim c = \beta > 0$. Then $a \preceq b \sim_\beta c$ and $c \preceq d \sim_\alpha a$ so that $a \precapprox_\beta c$ and $c \precapprox_\alpha a$, but $a \neq c$, violating fuzzy antisymmetry.

[7]Note that the crisp relation $\preceq$ can be treated as a fuzzy relation by identifying its characteristic function $\mathbf{1}_{\preceq}$ with the membership function $\mu_{\preceq}$.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

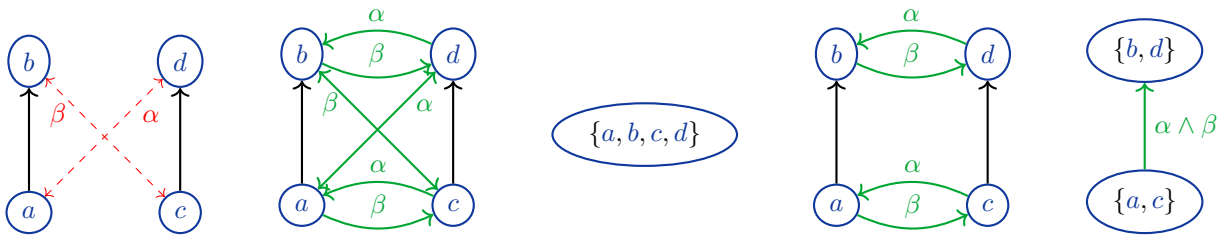

Fig. 6: The fuzzy relations of Examples IV.7 and IV.10. (a) $\preceq$ (black) and $\sim$ (red, dashed). (b) $\precsim$. (c) $\precsim$ on $\mathcal{S}_{/\approx}$. (d) $\precnsim$. (e) $\precnsim$ on $\mathcal{S}_{/\approx}$.

Proposition IV.8 (Fuzzy preorder $\precsim$). The fuzzy relation $\precsim$ of Definition IV.5 is a fuzzy preorder.

While cases such as the latter one could be blamed on an improper modeling of the subsumption and the similarity relations, we propose a solution that consists in the definition of a fuzzy partial order on equivalence classes of sorts starting from a fuzzy preorder on sorts. This construction generalizes to fuzzy set theory the well-known order theoretic transformation of a preorder (X, P) into a partial order on the quotient of X modulo the equivalence relation $x \approx y \Leftrightarrow (x, y) \in P$ and $(y, x) \in P$.

Definition IV.9 (Fuzzy partial order $\precnsim$ on $\mathcal{S}_{/\approx}$). Let $(\mathcal{S}, \preceq)$ be a fuzzy preorder on $\mathcal{S}$ and $\approx$ be the equivalence relation on $\mathcal{S}$ defined by letting $s \approx s' \Leftrightarrow (s, s') \in |\preceq|$ and $(s', s) \in |\preceq|$. The equivalence class of $s \in \mathcal{S}$ with respect to $\approx$ is denoted $[s]$. Let $\alpha_s \stackrel{\text{def}}{=} \bigwedge_{s' \in [s]} (\preceq(s, s') \wedge \preceq(s', s))$ for each $s \in \mathcal{S}$. The fuzzy binary relation $\precnsim$ on $\mathcal{S}_{/\approx}$ is defined by letting $\precnsim([s_0], [s_1]) \stackrel{\text{def}}{=} 1$ if $[s_0] = [s_1]$, and otherwise

$$\precnsim([s_0], [s_1]) \stackrel{\text{def}}{=} \alpha_{s_0} \wedge \alpha_{s_1} \wedge \bigvee_{s'_0 \in [s_0], s'_1 \in [s_1]} \preceq(s'_0, s'_1).$$

Note that the same symbol $\precnsim$ is used for the preorder on $\mathcal{S}$ and the fuzzy relation on $\mathcal{S}_{/\approx}$, since its meaning is clear from context.

Example IV.10 (Fuzzy partial order $\precnsim$ on $\mathcal{S}_{/\approx}$). Continuing from Example IV.7, Fig. 6e depicts the fuzzy partial order $\precnsim$ on the equivalence classes of $\mathcal{S}$ obtained from the fuzzy preorder $\precsim$ on $\mathcal{S}$ of Fig. 6d, as defined in Definition IV.9. Then, for example, the similarity-based unification of the OSF terms $b(f \to \top)$ and $c(f \to \top)$ could be performed by considering the fuzzy subsumption on the equivalence classes of the sorts, resulting in the unifier $t_1 = \{a, c\}(f \to \top)$, where $\{a, c\}$ is a *disjunctive sort*, and t_1 is a *disjunctive OSF term* [24]. Alternatively, the construction of Definition IV.9 could be applied to the fuzzy preorder $\precsim$ represented in Fig. 6b, leading to the partial order of Fig. 6c. In this case the similarity-based unification of the same two terms would result in $t_2 = \{a, b, c, d\}(f \to \top)$.

Proposition IV.11 (Fuzzy partial order $\precnsim$ on $\mathcal{S}_{/\approx}$). The fuzzy relation $\preceq$ of Definition IV.9 is a fuzzy partial order on $\mathcal{S}_{/\approx}$.

C. From a Fuzzy Poset to a Fuzzy Lattice

In (crisp) OSF logic, the sort subsumption relation is by definition required to be a finite lattice. In practice, however, it is enough for $(\mathcal{S}, \preceq)$ to be a partial order and to implicitly work on a *completion* of $(\mathcal{S}, \preceq)$ consisting of a lattice of sets of sorts, where singletons are treated as normal sorts, while sets of two or more sorts as *disjunctive sorts* [24]. Formally, the lattice is defined on the antichains of the sort subsumption partial order and, from an implementation standpoint, it does not need to be explicitly constructed [43].

Analogously, the fuzzy sort subsumption relation of fuzzy OSF logic is required to be a fuzzy lattice [30]. In this section we present a generalization to fuzzy set theory of the completion of a partial order into a lattice, i.e., we show how to construct a fuzzy lattice on the antichains of a fuzzy poset.

Definition IV.12 (Antichains in a fuzzy poset). Let $(\mathcal{S}, \preceq)$ be a fuzzy poset. Two elements $s, s' \in \mathcal{S}$ are said to be *incomparable* – denoted $s \parallel s'$ – if $\preceq(s, s') = 0$ and $\preceq(s', s) = 0$. An *antichain* is a subset $C \subseteq \mathcal{S}$ such that, for all $s, s' \in C$, $s \parallel s'$ if $s \neq s'$. The set of all antichains of $(\mathcal{S}, \preceq)$ is denoted $Antichains(\mathcal{S})$.

Definition IV.13 (Fuzzy antichain ordering). A fuzzy partial order $\preceq$ on $\mathcal{S}$ can be extended to a fuzzy relation $\preceq$ on $Antichains(\mathcal{S})$ by letting, for all $C, C' \in Antichains(\mathcal{S})$: $\preceq(C, C') \stackrel{\text{def}}{=} \min_{c \in C} \max_{c' \in C'} \preceq(c, c')$.

The same symbol $\preceq$ is used for the partial order on $\mathcal{S}$ and the fuzzy relation on $Antichains(\mathcal{S})$, since its meaning is always clear from context. Before showing that $\preceq$ on $Antichains(\mathcal{S})$ is a fuzzy lattice, we need the following definitions.

Definition IV.14 (Maximal elements and down-sets). Let $(\mathcal{S}, \preceq)$ be a fuzzy poset and $S \subseteq \mathcal{S}$. The set of *maximal elements of S* is defined as $\lceil S \rceil \stackrel{\text{def}}{=} \{s \in S \mid \forall s' \in S (s \preceq s', \Rightarrow s = s')\}$. The *(fuzzy) down-set of S* is defined as the set $S{\downarrow} \stackrel{\text{def}}{=} \{s \in \mathcal{S} \mid \exists s' \in S, s \preceq s'\}$.

Note that, for any $S \subseteq \mathcal{S}$, $\lceil S \rceil$ is an antichain.

Proposition IV.15 (Fuzzy antichain lattice). Let $(\mathcal{S}, \preceq)$ be a fuzzy poset. Then $(Antichains(\mathcal{S}), \preceq)$ from Definition IV.13 is a fuzzy lattice, where the GLB of $C, C' \in Antichains(\mathcal{S})$ is given by $C_1 \curlywedge C_2 \stackrel{\text{def}}{=} \lceil C_1{\downarrow} \cap C_2{\downarrow} \rceil$. Moreover, $(Antichains(\mathcal{S}), \preceq)$ preserves the GLBs that exist in $(\mathcal{S}, \preceq)$, i.e., if $s_0 \curlywedge s_1 = s$, then $\{s_0\} \curlywedge \{s_1\} = \{s\}$.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

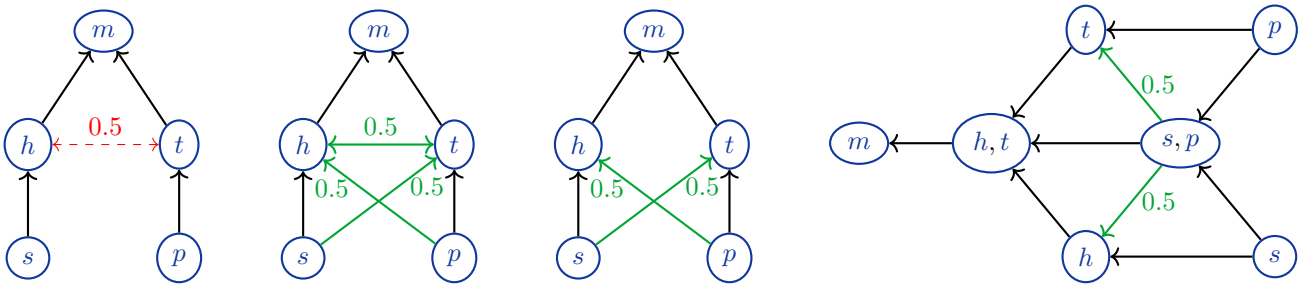

Fig. 7: The fuzzy relations of Example IV.16. (a) $\preceq$ (black) and $\sim$ (red, dashed). (b) $\precsim$. (c) $\precnsim$. (d) Completion of $\precnsim$.

Example IV.16 (Greatest lower bounds in $\precnsim$). Consider the subsumption $\preceq$ and the similarity $\sim$ represented in Fig. 7a, and the fuzzy partial order $\precnsim$ obtained from their combination (according to Definition IV.5) represented in Fig. 7c. In this case $\precnsim$ is not a fuzzy lattice, since h and t do not have a GLB. Fig. 7d shows the completion of $\precnsim$ as defined in Definition IV.13, in which the GLB of $\{h\}$ and $\{t\}$ is $\{s, p\}$. We can thus use this fuzzy order for the unification, e.g., of the OSF terms $t_1 = m(f \to h)$ and $t_2 = m(f \to t)$, which would result in the disjunctive OSF term $m(f \to \{s, p\})$, which is subsumed by t_1 and t_2 with degree 0.5.

Similarly to the crisp setting, the completion of a fuzzy partial order $\preceq : \mathcal{S} \to [0, 1]$ does not need to be computed explicitly in order to find the GLB of two sorts in $Antichains(\mathcal{S})$, as the same encoding and decoding strategies of [43], [44] can be employed in the fuzzy case.

V. Potential Applications

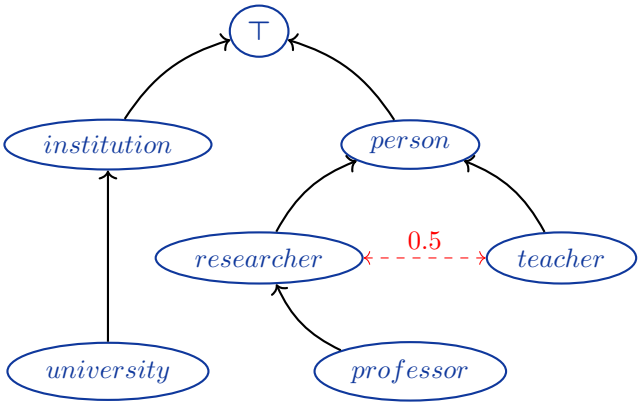

Fig. 8: The subsumption (black) and similarity (red, dashed) relations of Example V.1

A. Similarity-Based CEDAR

CEDAR [28], [29] is a Semantic Web reasoner based on OSF logic and OSF term unification, whose implementation relies on techniques that exploit the specificity of concept taxonomies, particularly the fact that subsumption orderings are central to all ontologies. The main capabilities of CEDAR are concept classification, Boolean query answering (i.e., answering queries formed by sort symbols and Boolean connectives), and answering queries represented as OSF terms (CEDAR features an extended OSF term syntax that also supports a few Description Logic constructs [29]).

Before being executed, a query – expressed as an OSF term in which the variables of interest are marked by a question mark – is first optimized according to both the knowledge expressed in the given ontology and the OSF constraint normalization rules. The resulting query is then translated into a SPARQL query and executed by a SPARQL engine. This process potentially leads to an optimized query that is more efficient to execute, by reducing the instance retrieval search space. Moreover, it also ensures the consistency of the input query against the ontology, so that no answer is provided if the query is inconsistent. The retrieval is further optimized thanks to a custom RDF triple indexing scheme based on OSF sort and attribute information [29]. The results of [28], [29] show that, at the time of the experiments, CEDAR was consistently among the best Semantic Web reasoners in terms of concept classification, and several orders-of-magnitude more efficient in terms of terminological reasoning.

Example V.1 (CEDAR query normalization). Consider the subsumption $\preceq$ and similarity $\sim$ represented in Fig. 8 (adapted from an example of [29]) and the query $?X : person(works\_at \to \top)$, which aims to retrieve people that work somewhere. Additionally, assume that the feature $works\_at$ applies to objects of sort $researcher$ and points to objects of sort $university$. Then, the CEDAR query optimization step [29] would transform the given query into the query $?X : researcher(works\_at \to university)$. The reason is that, since X is of sort $person$ and the feature $works\_at$ only applies to $researcher$s, then by the rule Sort Intersection the variable X must be of sort $person \wedge researcher = researcher$. An analogous reasoning applies to the value of the feature $works\_at$. The resulting query – which aims to retrieve researchers rather than people, and universities rather than entities of any sort – benefits from a much smaller search space, and is thus more efficient to execute.

On the other hand, CEDAR would find the query $?X : teacher(works\_at \to \top)$ to be inconsistent with the knowledge expressed in Fig. 8 (according to the subsumption $\preceq$, the sorts $teacher$ and $researcher$ are disjoint: $teacher \wedge researcher = \bot$), thus preventing the query from being

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

translated into SPARQL and executed.

While the consistency check is useful for preventing unnecessary computations, there may be scenarios where offering approximate answers to inconsistent queries becomes desirable. For instance, users often do not possess full knowledge of the extensive ontologies they query, making the inconsistency of a query, such as the second one in Example V.1, potentially surprising. Furthermore, the inconsistency of this query is due to the closed world assumption of OSF logic, which states that, since $teacher \wedge researcher = \bot$, the two sorts must denote disjoint classes. However, it may be desirable to relax this consistency requirement in contexts where this assumption does not hold, such as settings where domain knowledge can evolve or be incomplete.

With these considerations in mind, similarity-based reasoning with OSF logic could be implemented in an extension of CEDAR in order to relax the consistency requirement and provide approximate answers even when the input query is inconsistent with respect to the ontology. This would be achieved, for example, by enriching the subsumption relation of the given ontology with a similarity, which would then be taken into account during the query optimization phase, making query answering with CEDAR more flexible.

Example V.2 (Similarity-based CEDAR query normalization). Continuing from Example V.1, in a similarity-based extension of CEDAR we could consider the similarity relation $\sim$ of Fig. 8, which specifies that $researcher$ and $teacher$ are similar. The constraint normalization rules used in the query optimization phase could then be executed over the fuzzy subsumption $\precsim$ obtained by combining $\preceq$ and $\sim$ as described in the previous section, where $professor$ becomes the GLB of $researcher$ and $teacher$. The query $?X : teacher(works\_at \rightarrow \top)$, which was previously inconsistent, can now be simplified to $?X : professor(works\_at \rightarrow university)$. In this case, the answers to this query are associated with the approximation (satisfaction) degree 0.5. Alternatively, instead of providing approximate answers, the new query could be suggested to the user as a replacement to the inconsistent query.

B. Logic Programming with Similarity-Based OSF

Another possible application of our approach is the implementation of a logic programming language based on SLD resolution and OSF term unification (like LOGIN [26]), which additionally allows the specification of a sort similarity relation, similarly to how Bousi~Prolog [13], [14] allows the specification of a similarity (or proximity) relation between functors, constants and predicates[8]. The addition of a similarity relation allows the retrieval of approximate solutions to a query, improving the flexibility of query answering.

There are several potential advantages of using OSF logic in this context. First of all, as discussed in previous sections, the unification algorithm for OSF terms takes into account a (fuzzy) sort subsumption relation, which can lead to more efficient computations [26], [27]. For instance, Example II.8 shows how a single OSF term unification can replace several resolution steps, and analogously Example III.4 discusses how a single weak OSF term unification can replace several similarity-based resolution steps. Another advantage is the flexibility provided by OSF terms, which lack a fixed arity and can thus easily represent partial information, and are moreover simpler to interpret thanks to their use of features rather than positions to specify arguments [26].

The following example illustrates a similarity-based OSF logic program, showcasing the flexibility of our approach in retrieving approximate solutions to queries. The computation of a solution to a given query relies on the weak unification of OSF terms, and SLD resolution on predicate symbols. Predicate symbols in this context are distinct from the sorts and features of OSF terms. They work exactly as in Prolog, with the only difference that they accept OSF terms rather than first-order terms as arguments.

Example V.3 (Logic programming with OSF logic and a sort similarity). Consider the program of Fig. 9. A sort subsumption relation is specified by declarations of shape `s < s'`, while an expression like `{ c } < s` means that the constant `c` is an instance of the sort `s`. Constants such as `c` are treated as singleton sorts (sorts that denote a single element), and thus as OSF terms themselves. A similarity declaration of shape $x \sim y = \alpha$ specifies that the sorts `horror` and `thriller` are similar to degree 0.5. A few facts with the predicate `director_of` – whose arguments are OSF terms – are then specified and, finally, a rule involving the predicates `director_of` and `likes` states that `alinda` likes thriller movies. The query `?- likes(alinda, Y : movie)` is first reduced by resolution to the goal `director_of(X : person, Y : thriller)`, which is then resolved against the facts of the program, returning the following solutions:

- a solution binding `Y` to `memento(title -> "Memento")`, through the unification of `thriller` and `memento(title -> "Memento")`;
- a solution binding `Y` to `psycho` with approximation degree 0.5, through the weak unification of `thriller` and `psycho` (`psycho` is a `horror`, which is similar to `thriller`);
- a solution binding `Y` to `halloween(year -> 1979)` with approximation degree 0.5, through the weak unification of `thriller` and `halloween(year -> 1979)` (`halloween` is a `horror`, which is similar to `thriller`).

Behind the scenes, the computations would involve the unification of OSF terms over the fuzzy subsumption $\precsim$ obtained by combining, before the execution of the queries, the subsumption $\preceq$ and the similarity $\sim$ specified by the program.

Note that in this example a similarity was only specified between sort symbols in order to perform weak OSF term unification, but it could be possible to further extend our approach by also considering a similarity between predicate symbols in order to perform similarity-based SLD resolution. This would be analogous, for instance, to Bousi~Prolog,

[8]Note that Bousi~Prolog also supports proximity relations [13], [14] (i.e., reflexive and symmetric fuzzy binary relations, which can be more appropriate in some modeling contexts [7], [45], [46]), and linguistic variables [47].

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897


```
% Subsumption relation
director < person.
slasher < horror.
horror < movie.
thriller < movie.
% Similarity relation
thriller ~ horror = 0.5.
% Instances
{ alinda } < person.
{ hitchcock, carpenter, nolan  } < director.
{ psycho } < horror.
{ halloween } < slasher.
{ memento } < thriller.
% Facts
director_of(hitchcock, psycho).
director_of(carpenter, halloween(year -> 1979)).
director_of(nolan, memento(title -> "Memento")).
% Rules
likes(alinda, Y) :- director_of(X : person, Y : thriller).
```

Fig. 9: Logic program with OSF terms and a similarity relation

which supports a similarity relation on predicate symbols (in order to perform similarity-based SLD resolution) and on functor symbols (in order to perform weak first-order term unification).

We conclude with an example that showcases another advantage of our approach with respect to other similarity-based logic programming languages such as Bousi∼Prolog, in particular when dealing with cycles in the specification of the subsumption and similarity relations.

Example V.4 (Cycles in Bousi∼Prolog and similarity-based OSF logic). Consider the logic program enriched with a similarity relation of Fig. 10a, which follows the syntax of Bousi∼Prolog [13], [14]. The program starts by describing the similarity and the subsumption relations of Fig. 6a through declarations of shape $x \sim y = \alpha$ and two Horn rules, followed by a few facts that specify the instances of the predicates `a`, `b`, `c` and `d`. Loading this program in Bousi∼Prolog and querying `a(X)` leads to an infinite loop[9]. This is due to the fact that, since `a` and `d` are similar, then the query `a(X)` resolves with the clause `d(X) :- c(X)` though similarity-based SLD resolution, and the goal becomes `c(X)`. Then, since `b` and `c` are similar, the new goal resolves with the clause `b(X) :- a(X)` by similarity-based SLD resolution, resulting in the goal `a(X)` again, so that the cycle starts over.

Now consider the similarity-based OSF logic program of Fig. 10b, which describes the same setting. The subsumption and similarity defined in the program would be combined into a fuzzy subsumption relation before the execution of the query a(X), leading to the fuzzy partial order $\precsim$ of Fig. 6c, or the fuzzy partial order $\precsim$ of Fig. 6e (this could depend on an additional parameter passed to the program). In the first case, the answers to the query would include the instances of a, b, c and d (i.e., alice, bob, carol and david), while in the second case they would consist of the instances of a and c (i.e., alice and carol).

```
a ~ d = 0.4 .
b ~ c = 0.5 .
b(X) :- a(X).
d(X) :- c(X).
a(alice).
b(bob).
c(carol).
d(david).
```

(a)

```
a ~ d = 0.4 .
b ~ c = 0.5 .
a < b.
c < d.
{ alice } < a.
{ bob   } < b.
{ carol } < c.
{ david } < d.
```

(b)

Fig. 10: Logic programs with a similarity relation. (a) Bousi∼Prolog version. (b) OSF version.

VI. Conclusion

We have presented an approach to define approximate reasoning with OSF logic by extending its language with a similarity relation on sort symbols. By combining this similarity relation with the usual sort subsumption relation it is possible to define a fuzzy subsumption relation which intuitively combines the information of both relations and their interaction. The fuzzy subsumption is then taken into account when unifying two OSF terms, using the same rules of (crisp) OSF logic.

As discussed in Section V, similarity-based OSF logic could be applied, for example, in the implementation of a fuzzy logic programming language or a similarity-based extension

[9]The program of Fig. 10a was tested on Bousi∼Prolog 3.6.1.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

of the CEDAR reasoner [29], [29] which would be capable of returning approximate answer to queries.

ACKNOWLEDGMENTS

The authors express their gratitude and a fond thought to Hassan Aït-Kaci, who with Gabriella Pasi set out to define a fuzzy version of OSF logic. This paper originates from their work on the definition of similarity-based unification for OSF terms, extending the approach of [21].

This work was partially supported by the MUR under the grant "Dipartimenti di Eccellenza 2023-2027" of the Department of Informatics, Systems and Communication of the University of Milano-Bicocca, Italy.

APPENDIX A
FUZZY SET THEORY

We recall the basic definitions of fuzzy set theory [39] to fix the notation. Whenever possible we use the same notation for fuzzy sets and orders as the one used for crisp sets and orders, but with the addition of a dot $(\cdot)$ to avoid ambiguity. E.g., the symbol for the intersection of two crisp sets is $\cap$, while the symbol for the intersection of two fuzzy sets is $\dot\cap$. We adopt the minimum as a t-norm (denoted $\wedge$) and the maximum as a t-conorm (denoted $\vee$).

Definition A.1 (Fuzzy subset). A *fuzzy subset* F of a (crisp) set X is determined by its membership function $\mu_F : X \to [0,1]$.

Definition A.2 (Intersection and union of fuzzy subsets). The *intersection* $\dot\bigcap\mathcal{F}$ *of a set* $\mathcal{F}$ *of fuzzy subsets of a set* X is defined by letting $\mu_{\dot\bigcap\mathcal{F}}(x) \stackrel{\text{def}}{=} \inf(\{\mu_F(x) \mid F \in \mathcal{F}\})$. The *union* $\dot\bigcup\mathcal{F}$ *of a set* $\mathcal{F}$ *of fuzzy subsets of a set* X is defined by letting $\mu_{\dot\bigcup\mathcal{F}}(x) \stackrel{\text{def}}{=} \sup(\{\mu_F(x) \mid F \in \mathcal{F}\})$.

Definition A.3 (Support). The *support* of a fuzzy subset F of X is defined as $|F| \stackrel{\text{def}}{=} \{x \in X \mid \mu_F(x) > 0\}$.

Remark A.4 (Fuzzy set notation). Throughout the paper the membership function of a fuzzy subset F of X is simply written $F : X \to [0,1]$ instead of $\mu_F : X \to [0,1]$.

Definition A.5 (Fuzzy binary relation). A *fuzzy binary relation* R on a set X is a fuzzy subset of $X \times X$, i.e., it is a function $R : X \times X \to [0,1]$.

Definition A.6 (Fuzzy preorder). A fuzzy binary relation R on a set X is a *fuzzy preorder* if it satisfies:

$$\forall x \in X,\ R(x,x) = 1, \qquad \text{(Fuzzy Reflexivity)}$$

$$\forall x,y,z \in X,\ R(x,z) \geq R(x,y) \wedge R(y,z). \qquad \text{(Max-Min Transitivity)}$$

Definition A.7 (Fuzzy partial order). A fuzzy binary relation R on a set X is a *fuzzy partial order* if it satisfies (Fuzzy Reflexivity) and (Max-Min Transitivity) and

$$\forall x,y \in X, \text{if } R(x,y) > 0 \text{ and } R(y,x) > 0, \text{then } x = y. \qquad \text{(Strong Fuzzy Antisymmetry)}$$

The pair (X,R) is called a *fuzzy partially ordered set* (*fuzzy poset*).

Definition A.8 (Composition of fuzzy binary relations). The *(max-min) composition* of two fuzzy binary relations R and Q on a finite set X is the fuzzy binary relation $R \circ Q$ defined by the membership function $R \circ Q(x,z) \stackrel{\text{def}}{=} \bigvee_{y \in X}(R(x,y) \wedge Q(y,z))$. The *$n$-ary composition* of a fuzzy binary relation R with itself is defined by letting $R^1 \stackrel{\text{def}}{=} R$ and $R^n \stackrel{\text{def}}{=} R \circ R^{n-1}$ for $n > 1$.

Definition A.9 (Reflexive and transitive closure of a fuzzy binary relation). The *transitive closure* of a fuzzy binary relation R is defined as $R^{\oplus} \stackrel{\text{def}}{=} \dot\bigcup_{m \geq 1} R^m$. The *reflexive and transitive closure* $R^{\circledast}$ of a fuzzy binary relation R is obtained by letting $R^{\circledast}(x,y) \stackrel{\text{def}}{=} 1$ if $x = y$ and $R^{\circledast}(x,y) \stackrel{\text{def}}{=} R^{\oplus}(x,y)$ otherwise.

Definition A.10 (Similarity relation). A fuzzy binary relation $\sim$ on a set X is a *similarity* if it satisfies (Fuzzy Reflexivity) and (Max-Min Transitivity) and

$$\forall x,y \in X, {\sim}(x,y) = {\sim}(y,x). \qquad \text{(Fuzzy Symmetry)}$$

Remark A.11 (Infix notation). If R is a fuzzy binary relation, then $xR_\alpha y$ stands for $R(x,y) = \alpha$, and xRy stands for $R(x,y) > 0$.

We adopt the definitions of lower bounds, greatest lower bounds and fuzzy lattice from [48]–[50].

Definition A.12 (Lower bounds in a fuzzy poset). Let $(\mathcal{S}, \preceq)$ be a fuzzy poset and $S \subseteq \mathcal{S}$. The set of *lower bounds of* S is defined as $S^{fl} \stackrel{\text{def}}{=} \{s \in \mathcal{S} \mid \forall s' \in S, s \preceq s'\}$.

Definition A.13 (Fuzzy greatest lower bound). Let $(\mathcal{S}, \preceq)$ be a fuzzy poset and $S \subseteq \mathcal{S}$. The *greatest lower bound (GLB) of* S is the unique $s \in S^{fl}$ such that, for all $s' \in S^{fl}$, $s' \preceq s$. If the GLB of S exists, it is denoted $\dot\bigwedge S$, or simply $s \mathbin{\dot\wedge} s'$ in case $S = \{s, s'\}$.

Definition A.14 (Fuzzy lattice and bounded lattice). A fuzzy poset $(\mathcal{S}, \preceq)$ is a *fuzzy lattice* if every pair of elements has a GLB. A fuzzy lattice $(\mathcal{S}, \preceq)$ is *bounded* if there are elements $\bot, \top \in \mathcal{S}$ such that, for all $s \in \mathcal{S}$, $\preceq(\bot, s) = 1$ and $\preceq(s, \top) = 1$.

The following proposition states that the support $(\mathcal{S}, |\preceq|)$ of a fuzzy lattice $(\mathcal{S}, \preceq)$ is a lattice. Additionally, if s is the GLB of s_0 and s_1 in $(\mathcal{S}, \preceq)$, then s is also the GLB of s_0 and s_1 in $(\mathcal{S}, |\preceq|)$, and vice versa. This property ensures that the computation of GLBs in a fuzzy lattice can be reduced to the crisp setting. For instance, it would be possible to employ the techniques of [43], [44] on the weighted graph representation of a fuzzy lattice simply by ignoring the edge weights, preserving the same computational complexity.

Proposition A.15 (Fuzzy and crisp lattices). Let $(\mathcal{S}, \preceq)$ be a fuzzy lattice. Then $(\mathcal{S}, |\preceq|)$ is a (crisp) lattice on $\mathcal{S}$. Moreover, if $\wedge$ is the GLB operation for $(\mathcal{S}, |\preceq|)$, then $\dot\bigwedge S = \bigwedge S$ for every subset $S \subseteq \mathcal{S}$.

Proof. Let $\preceq \stackrel{\text{def}}{=} |\preceq|$ and $(\mathcal{S}, \preceq)$ be a fuzzy lattice. It follows that $\preceq$ is a reflexive, antisymmetric and transitive binary relation on $\mathcal{S}$.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

- (Reflexivity) Since $\preceq(s,s)=1$ for all $s\in\mathcal{S}$, then $s\preceq s$ for all $s\in\mathcal{S}$.
- (Antisymmetry) If $s\preceq s'$ and $s'\preceq s$, then $\preceq(s,s')>0$ and $\preceq(s',s)>0$, and by the antisymmetry of $\preceq$ it follows that $s=s'$.
- (Transitivity) If $s_1\preceq s_2$ and $s_2\preceq s_3$, then $\preceq(s_1,s_2)>0$ and $\preceq(s_2,s_3)>0$, so that by the transitivity of $\preceq$ it follows that $\preceq(s_1,s_3)\geq\min(\preceq(s_1,s_2),\preceq(s_2,s_3))>0$ and thus $s_1\preceq s_3$.

Let $S^l \stackrel{\text{def}}{=} \{s\in\mathcal{S}\mid s\preceq s', \forall s'\in S\}$ denote the set of lower bounds of $S\subseteq\mathcal{S}$ in $(\mathcal{S},\preceq)$, and note that $S^l=S^{fl}$ for any $S\subseteq\mathcal{S}$. Let $S\subseteq\mathcal{S}$. Then

$$\begin{aligned} s=\bigwedge S \quad &\Leftrightarrow \quad \text{(i) } s\in S^{fl} \text{ and (ii) } \forall s'\in S^{fl}, s'\preceq s \\ &\Leftrightarrow \quad \text{(i) } s\in S^{l} \text{ and (ii) } \forall s'\in S^{l}, s'\preceq s \\ &\Leftrightarrow \quad s=\bigwedge S. \end{aligned}$$

□

APPENDIX B
PROOFS OF RESULTS FROM SECTION IV

Proposition IV.2 (Equivalent definition of $\precsim$). Let $(\mathcal{S},\preceq)$ be a partial order[10], let $\sim$ be a similarity relation on $\mathcal{S}$, and let $\precsim$ be as in Definition IV.1. Then $\precsim=(\preceq\circ\sim)^{\oplus}$.

Proof. Let $s,s'\in\mathcal{S}$, and, for brevity, let $R\stackrel{\text{def}}{=}\preceq\circ\sim$. The fact that $\precsim(s,s')=R^{\oplus}(s,s')$ is given by the following equivalences:

$$\begin{aligned} \precsim(s,s')=\alpha \quad \Leftrightarrow_1 \quad & \alpha \text{ is the maximum such that there exists a sequence of sorts } s\sim_{\alpha_0} s_0\preceq t_1\cdots s_{p-1}\preceq t_p\sim_{\alpha_p} s' \text{ with } \alpha=\min_{0\leq i\leq p}\alpha_i \\ \Leftrightarrow_2 \quad & \alpha \text{ is the maximum such that there exists a sequence of sorts } s\,R_{\beta_0}\,s'_0\,R_{\beta_1}\,s'_1\ldots s'_{n-1}\,R_{\beta_n}\,s' \text{ with } \alpha=\min_{0\leq i\leq n}\beta_i \\ \Leftrightarrow_3 \quad & (\exists n\ R^n(s,s')=\alpha) \text{ and } (\forall m\geq 1,\ R^m(s,s')\leq\alpha) \\ \Leftrightarrow_4 \quad & R^{\oplus}\stackrel{\text{def}}{=}\bigcup_{m\geq 1}R^m=\alpha. \end{aligned}$$

The equivalences are given by:

1) Definition IV.1;
2) $R\stackrel{\text{def}}{=}(\preceq\circ\sim)$;
3) Definition A.8 and $\mathcal{S}$ is finite;
4) Definition A.9 and $\mathcal{S}$ is finite.

□

Proposition IV.4 (Similarity-subsumption preorder). The fuzzy relation $\precsim$ of Definition IV.1 is a fuzzy preorder.

Proof. Reflexivity holds since $s\sim_1 s$ for all $s\in\mathcal{S}$.

For transitivity, let $s_0,s_1,s_2\in\mathcal{S}$ be such that

$$s_0\precsim_{\alpha_0} s_1,\ s_1\precsim_{\alpha_1} s_2, \text{ and } s_0\precsim_{\alpha} s_2.$$

We want to prove that $\alpha\geq\min(\alpha_0,\alpha_1)$. By Definition IV.1 there exists chains

$$\begin{aligned} &s_0\sim_{\alpha_{(0,0)}} s_{(0,0)}\preceq\cdots\preceq s'_{(0,n)}\sim_{\alpha_{(0,n)}} s_1, \\ &s_1\sim_{\alpha_{(1,0)}} s_{(1,0)}\preceq\cdots\preceq s'_{(1,m)}\sim_{\alpha_{(1,m)}} s_2 \end{aligned}$$

such that $\alpha_0=\min_{0\leq i\leq n}\alpha_{(0,i)}$ and $\alpha_1=\min_{0\leq i\leq m}\alpha_{(1,i)}$. By joining them, these two chains constitute a single similarity-subsumption chain of strength $\min(\alpha_0,\alpha_1)$ from s_0 to s_2. By Definition IV.1, $\alpha\geq\min(\alpha_0,\alpha_1)$. □

Proposition IV.8 (Fuzzy preorder $\precsim$). The fuzzy relation $\precsim$ of Definition IV.5 is a fuzzy preorder.

Proof. Reflexivity is given by $\preceq$, and transitivity by the transitive closure. □

Proposition IV.11 (Fuzzy partial order $\precsim$ on $\mathcal{S}_{/\approx}$). The fuzzy relation $\preceq$ of Definition IV.9 is a fuzzy partial order on $\mathcal{S}_{/\approx}$.

Proof. First of all, note that, for all $x,x'\in\mathcal{S}$, whenever $x\in[x']$, then $\preceq(x,x')>0$, $\preceq(x',x)>0$, and $\alpha_x=\alpha_{x'}$. Also note that $\alpha_x>0$ for all $x\in\mathcal{S}$.

Reflexivity holds by definition.

For antisymmetry, let (a) $\preceq([x],[y])>0$ and (b) $\preceq([y],[x])>0$. We want to show that $[x]=[y]$. Let $x'\in[x]$, and we want to show that $x'\in[y]$. By (a), there are some $x''\in[x]$ and $y''\in[y]$ such that $\preceq(x'',y'')>0$. By (b), there are some $x'''\in[x]$ and $y'''\in[y]$ such that $\preceq(y''',x''')>0$. Then $x'\preceq x''\preceq y''\preceq y$ and $y\preceq y'''\preceq x'''\preceq x'$. Thus $x'\in[y]$. The fact that $[y]\subseteq[x]$ is proved similarly.

For transitivity, let

$$\begin{aligned} &\preceq([x],[y])=\alpha_x\wedge\alpha_y\wedge\bigvee_{x'\in[x],y'\in[y]}\preceq(x',y')=\alpha, \text{ and} \\ &\preceq([y],[z])=\alpha_y\wedge\alpha_z\wedge\bigvee_{y'\in[y],z'\in[z]}\preceq(y',z')=\beta. \end{aligned}$$

We want to show that $\preceq([x],[z])\geq\alpha\wedge\beta$.

Since $\mathcal{S}$ is finite, let $x_0\in[x]$, $y_0,y_1\in[y]$, and $z_1\in[z]$ be such that

$$\text{(c) } \preceq(x_0,y_0)=\bigvee_{x'\in[x],y'\in[y]}\preceq(x',y')$$

and

$$\text{(d) } \preceq(y_1,z_1)=\bigvee_{y'\in[y],z'\in[z]}\preceq(y',z').$$

Then:

$$\begin{aligned} \bigvee_{x'\in[x],z'\in[z]}\preceq(x',z') \ \geq_1\ & \preceq(x_0,z_1)\geq_2 \\ & \preceq(x_0,y_0)\wedge\preceq(y_0,z_1)\geq_3 \\ & \preceq(x_0,y_0)\wedge\preceq(y_0,y_1)\wedge\preceq(y_1,z_1)\geq_4 \\ & \preceq(x_0,y_0)\wedge\alpha_y\wedge\preceq(y_1,z_1)\geq_5 \\ & \bigvee_{x'\in[x],y'\in[y]}\preceq(x',y')\wedge\alpha_y\wedge\bigvee_{y'\in[y],z'\in[z]}\preceq(y',z'). \end{aligned}$$

The inequalities are given by:

1) Definition of the t-conorm $\vee$, $x_0\in[x]$, and $z_1\in[z]$;
2) Max-min transitivity of $\preceq$;
3) Max-min transitivity of $\preceq$;
4) Definition of α_y (Definition IV.9), $y_0,y_1\in[y]$;
5) (c) and (d).

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

Finally:

$$\preceq([x],[z]) \stackrel{\text{def}}{=} \bigvee_{x'\in[x],z'\in[z]} \preceq(x',z') \wedge \alpha_x \wedge \alpha_z \geq \bigvee_{x'\in[x],y'\in[y]} \preceq(x',y') \wedge \alpha_y \wedge \bigvee_{y'\in[y],z'\in[z]} \preceq(y',z') \wedge \alpha_x \wedge \alpha_z = \alpha \wedge \beta. \quad \square$$

Proposition IV.15 (Fuzzy antichain lattice). Let $(\mathcal{S},\preceq)$ be a fuzzy poset. Then $(Antichains(\mathcal{S}),\preceq)$ from Definition IV.13 is a fuzzy lattice, where the GLB of $C, C' \in Antichains(\mathcal{S})$ is given by $C_1 \curlywedge C_2 \stackrel{\text{def}}{=} \lceil C_1{\downarrow} \cap C_2{\downarrow}\rceil^{\cdot}$. Moreover, $(Antichains(\mathcal{S}),\preceq)$ preserves the GLBs that exist in $(\mathcal{S},\preceq)$, i.e., if $s_0 \curlywedge s_1 = s$, then $\{s_0\} \curlywedge \{s_1\} = \{s\}$.

Proof. First of all, note that the following holds by Definition IV.13:

$$\preceq(C,C') \geq \alpha \Leftrightarrow \forall c \in C\ \exists c' \in C' \text{ such that } \preceq(c,c') \geq \alpha.$$

This fact will be used throughout the proof.

Reflexivity of $(Antichains(\mathcal{S}),\preceq)$ is obvious, since, for any $C \in Antichains(\mathcal{S})$, for all $c \in C$: $\preceq(c,c) = 1$.

For transitivity, suppose that (i) $\preceq(C_0,C_1) = \alpha$ and (ii) $\preceq(C_1,C_2) = \beta$. To show that $\preceq(C_0,C_2) \geq \min(\alpha,\beta)$ it suffices to show that for any $c_0 \in C_0$ there exists a $c_2 \in C_2$ such that $\preceq(c_0,c_2) \geq \min(\alpha,\beta)$. Thus, let $c_0 \in C_0$ be arbitrary. Then by (i) there is some $c_1 \in C_1$ such that $\preceq(c_0,c_1) \geq \alpha$, and by (ii) there is some $c_2 \in C_2$ such that $\preceq(c_1,c_2) \geq \beta$. By max-min transitivity of $(\mathcal{S},\preceq)$ then

$$\preceq(c_0,c_2) \geq \min(\preceq(c_0,c_1),\preceq(c_1,c_2)) \geq \min(\alpha,\beta).$$

For antisymmetry, suppose that (iii) $\preceq(C_0,C_1) > 0$ and (iv) $\preceq(C_1,C_0) > 0$. To show that $C_0 = C_1$, let $c_0 \in C_0$. Then by (iii) there is some $c_1 \in C_1$ such that $\preceq(c_0,c_1) > 0$ and by (iv) there is some $c_2 \in C_0$ such that $\preceq(c_1,c_2) > 0$, implying $\preceq(c_0,c_2) > 0$. Because C_0 is an antichain, then $c_0 = c_2$, which implies $c_1 = c_0 = c_2$ by fuzzy antisymmetry of $(\mathcal{S},\preceq)$, so that $c_0 \in C_1$. The other direction is proved similarly.

Now let $C_0, C_1 \in Antichains(\mathcal{S})$, and $C \stackrel{\text{def}}{=} \lceil C_0{\downarrow} \cap C_1{\downarrow}\rceil^{\cdot}$. Then C is the GLB of C_0 and C_1 in $(Antichains(\mathcal{S}),\preceq)$.

- To prove that $C \preceq C_0$ it suffices to show that for any $c \in C$ there is some $c_0 \in C_0$ such that $\preceq(c,c_0) > 0$. This immediately follows from the fact that if $c \in C$, then in particular $c \in C_0{\downarrow}$. The fact that $C \preceq C_1$ is proved similarly.
- Suppose $C' \in Antichains(\mathcal{S})$ is such that (v) $C' \preceq C_0$ and (vi) $C' \preceq C_1$. To show that $C' \preceq C$, it is sufficient to show that for any $c' \in C'$ there is some $c \in C$ such that $\preceq(c',c) > 0$. Hence let $c' \in C'$ be arbitrary. By (v) and (vi) there are some $c_0 \in C_0$ and $c_1 \in C_1$ such that $c' \preceq c_0$ and $c' \preceq c_1$. Then $c' \in C_0{\downarrow} \cap C_1{\downarrow}$. Then by definition of C there is some $c \in C$ (possibly equal to c') such that $c' \preceq c$.

Finally, suppose s is the GLB of s_0, s_1 in $(\mathcal{S},\preceq)$. Then $\{s_0\} \curlywedge \{s_1\} = \lceil s_0{\downarrow} \cap s_1{\downarrow}\rceil^{\cdot} = \{s\}$. $\square$

REFERENCES

[1] M. Ying, "A logic for approximate reasoning," *The Journal of Symbolic Logic*, vol. 59, no. 3, pp. 830–837, 1994.

[2] G. Gerla and M. I. Sessa, *Similarity in Logic Programming*. Boston, MA: Springer US, 1999, pp. 19–31. [Online]. Available: https://doi.org/10.1007/978-1-4615-5261-1_2

[3] F. Formato, G. Gerla, and M. I. Sessa, "Similarity-based unification," *Fundamenta Informaticae*, vol. 41, no. 4, pp. 393–414, 2000. [Online]. Available: http://www.dipmat2.unisa.it/people/gerla/www/Down/similarity-based.pdf

[4] F. Arcelli-Fontana and F. Formato, "A similarity-based resolution rule," *International Journal of Intelligent Systems*, vol. 17, pp. 853–872, 2002.

[5] M. I. Sessa, "Approximate reasoning by similarity-based SLD resolution," *Theor. Comput. Sci.*, vol. 275, no. 1, pp. 389–426, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397501001888

[6] B. Dundua, T. Kutsia, M. Marin, and C. Pau, "Constraint Solving over Multiple Similarity Relations," in *5th International Conference on Formal Structures for Computation and Deduction (FSCD 2020)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), Z. M. Ariola, Ed., vol. 167. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, pp. 30:1–30:19. [Online]. Available: https://drops.dagstuhl.de/opus/volltexte/2020/12352

[7] P. Julián-Iranzo and C. Rubio-Manzano, "Proximity-based unification theory," *Fuzzy Sets Syst.*, vol. 262, pp. 21 – 43, 2015, theme: Logic and Computer Science. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165011414003133

[8] P. Julián-Iranzo and F. Sáenz-Pérez, "Proximity-based unification: an efficient implementation method," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 5, pp. 1238–1251, 2020.

[9] T. Kutsia and C. Pau, "Solving proximity constraints," in *Logic-Based Program Synthesis and Transformation*, M. Gabbrielli, Ed. Cham: Springer International Publishing, 2020, pp. 107–122.

[10] ——, "A framework for approximate generalization in quantitative theories," in *Automated Reasoning*, J. Blanchette, L. Kovács, and D. Pattinson, Eds. Cham: Springer International Publishing, 2022, pp. 578–596.

[11] ——, "Matching and generalization modulo proximity and tolerance relations," in *Language, Logic, and Computation*, A. Özgün and Y. Zinova, Eds. Cham: Springer International Publishing, 2022, pp. 323–342.

[12] P. Julián-Iranzo, G. Moreno, and J. A. Riaza, "The fuzzy logic programming language FASILL: Design and implementation," *Int. J. Approx. Reason.*, vol. 125, pp. 139–168, 2020.

[13] P. Julián-Iranzo, C. Rubio-Manzano, and J. Gallardo-Casero, "Bousi∼Prolog: a prolog extension language for flexible query answering," *Electronic Notes in Theoretical Computer Science*, vol. 248, pp. 131 – 147, 2009, proceedings of the Eighth Spanish Conference on Programming and Computer Languages (PROLE 2008). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1571066109002874

[14] P. Julián-Iranzo and F. Sáenz-Pérez, "Bousi∼Prolog: Design and implementation of a proximity-based fuzzy logic programming language," *Expert Syst. Appl.*, vol. 213, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422018760

[15] F. P. Romero, P. Julián-Iranzo, A. Soto, M. Ferreira-Satler, and J. Gallardo-Casero, "Classifying unlabeled short texts using a fuzzy declarative approach," *Language resources and evaluation*, vol. 47, pp. 151–178, 2013.

[16] S. H. Al-Sayadi, P. Julian-Iranzo, F. P. Romero, and F. Sáenz-Pérez, "A fuzzy declarative approach to classify unlabeled short texts based on automatically constructed wordnet ontologies," in *Computational Intelligence and Mathematics for Tackling Complex Problems 3*. Springer, 2022, pp. 157–164.

[17] C. Rubio-Manzano and G. Trivino, "Improving player experience in computer games by using players' behavior analysis and linguistic descriptions," *International Journal of Human-Computer Studies*, vol. 95, pp. 27–38, 2016.

[18] E. Çakir and H. Z. Ulukan, "A fuzzy logic programming environment for recycling facility selection." in *IJCCI*, 2019, pp. 367–374.

[19] E. Çakır and Z. Ulukan, "A fuzzy linguistic programming for sustainable ecotourism activities," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2020, pp. 0121–0126.

[20] C. Rubio-Manzano and P. Julián-Iranzo, "Incorporation of abstraction capability in a logic-based framework by using proximity relations," *Journal of Intelligent & Fuzzy Systems*, vol. 29, no. 4, pp. 1671–1683, 2015.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3362897

[21] 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.*, vol. 391, pp. 1–46, 2020, computer Science. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016501141930199X

[22] C. Pau and T. Kutsia, "Proximity-based unification and matching for fully fuzzy signatures," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2021.

[23] M. E. Cornejo, J. Medina Moreno, and C. Rubio-Manzano, "Towards a full fuzzy unification in the Bousi∼Prolog system," in *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2018, pp. 1–7. [Online]. Available: https://doi.org/10.1109/FUZZ-IEEE.2018.8491514

[24] H. Aït-Kaci and A. Podelski, "Towards a meaning of life," *J. Log. Program.*, vol. 16, no. 3, pp. 195–234, 1993. [Online]. Available: http://www.sciencedirect.com/science/article/pii/074310669390043G

[25] R. L. Carpenter, *The Logic of Typed Feature Structures: With Applications to Unification Grammars, Logic Programs and Constraint Resolution*, ser. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1992.

[26] H. Aït-Kaci and R. Nasr, "Login: a logic programming language with built-in inheritance," *J. Log. Program.*, vol. 3, no. 3, pp. 185–215, 1986. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0743106686900130

[27] A. G. Cohn, "Taxonomic reasoning with many-sorted logics," *Artif. Intell. Rev.*, vol. 3, no. 2-3, pp. 89–128, 1989.

[28] H. Aït-Kaci and S. Amir, "Classifying and querying very large taxonomies with bit-vector encoding," *J. Intell. Inf. Syst.*, vol. 48, no. 1, pp. 1–25, 2017. [Online]. Available: https://doi.org/10.1007/s10844-015-0383-2

[29] S. Amir and H. Aït-Kaci, "An efficient and large-scale reasoning method for the semantic web," *J. Intell. Inf. Syst.*, vol. 48, no. 3, pp. 653–674, Jun. 2017. [Online]. Available: https://doi.org/10.1007/s10844-016-0435-2

[30] G. C. Milanese and G. Pasi, "Fuzzy order-sorted feature logic," *Fuzzy Sets Syst.*, vol. 477, p. 108800, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165011423004451

[31] H. Aït-Kaci, "A lattice theoretic approach to computation based on a calculus of partially ordered type structures," Ph.D. dissertation, University of Pennsylvania, 1984.

[32] F. Baader, I. Horrocks, and U. Sattler, "Description logics," in *Handbook of Knowledge Representation*, ser. Foundations of Artificial Intelligence, F. van Harmelen, V. Lifschitz, and B. Porter, Eds. Elsevier, 2008, vol. 3, pp. 135–179. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574652607030039

[33] H. Aït-Kaci, "Data models as constraint systems: a key to the semantic web," *Constraint Processing Letters*, vol. 1, 2007.

[34] G. Smolka, "A feature logic with subsorts," *LILOG-Report*, vol. 33, 1988.

[35] K. Mukai, "Anadic tuples in prolog," ICOT, Tokyo, Japan, Tech. Rep. TR-239, 1987.

[36] B. Nebel and G. Smolka, "Representation and reasoning with attributive descriptions," in *Sorts and Types in Artificial Intelligence*, K. H. Bläsius, U. Hedtstück, and C.-R. Rollinger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 111–139.

[37] ——, *Attributive description formalisms ... and the rest of the world*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 439–452. [Online]. Available: https://doi.org/10.1007/3-540-54594-8_74

[38] H.-U. Krieger and U. Schäfer, "Dl meet fl: a bidirectional mapping between ontologies and linguistic knowledge," in *Coling 2010: Posters*, 2010, pp. 588–596.

[39] D. Dubois and H. Prade, *Fuzzy Sets and Systems. Theory and Applications.*, ser. Mathematics in Science and Engineering. Elsevier, 1980, vol. 144. [Online]. Available: https://www.sciencedirect.com/bookseries/mathematics-in-science-and-engineering/vol/144/

[40] U. Straccia, *Foundations of Fuzzy Logic and Semantic Web Languages*. New York: Chapman and Hall, 2014.

[41] S. Borgwardt and R. Peñaloza, "Fuzzy description logics–a survey," in *International Conference on Scalable Uncertainty Management*. Springer, 2017, pp. 31–45.

[42] G. C. Milanese and G. Pasi, "Conjunctive reasoning on fuzzy taxonomies with order-sorted feature logic," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2021, pp. 1–7. [Online]. Available: https://doi.org/10.1109/FUZZ45933.2021.9494474

[43] H. Aït-Kaci and S. Amir, "Classifying and querying very large taxonomies," LIRIS, Département d'Informatique, Université Claude Bernard Lyon 1, Villeurbanne, France, Tech. Rep., 2013, CEDAR Technical Report Number 2.

[44] H. Aït-Kaci, R. Boyer, P. Lincoln, and R. Nasr, "Efficient implementation of lattice operations," *ACM Trans. Program. Lang. Syst.*, vol. 11, no. 1, pp. 115–146, Jan. 1989. [Online]. Available: https://doi.org/10.1145/59287.59293

[45] S. Shenoi and A. Melton, "Proximity relations in the fuzzy relational database model," *Fuzzy Sets Syst.*, vol. 100, pp. 51 – 62, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165011499800062

[46] F. Klawonn, "Should fuzzy equality and similarity satisfy transitivity? comments on the paper by m. de cock and e. kerre," *Fuzzy Sets and Systems*, vol. 133, no. 2, pp. 175 – 180, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165011402002439

[47] C. Rubio-Manzano and P. Julián-Iranzo, "A fuzzy linguistic prolog and its applications," *Journal of Intelligent & Fuzzy Systems*, vol. 26, no. 3, pp. 1503–1516, 2014.

[48] I. Chon, "Fuzzy partial order relations and fuzzy lattices," *Korean J. Math.*, vol. 17, no. 4, pp. 361–374, 2009.

[49] I. Mezzomo, B. Bedregal, and R. H. N. Santiago, "Operations on bounded fuzzy lattices," in *2013 Joint IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS)*, 2013, pp. 151–156.

[50] I. Mezzomo and B. C. Bedregal, "On fuzzy α-lattices," in *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2016, pp. 775–781. [Online]. Available: https://doi.org/10.1109/FUZZ-IEEE.2016.7737766

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/