UFOM: Unified Fuzzy Ontology Matching

Yinuo Zhang, Anand Panangadan, Viktor K. Prasanna
University of Southern California
Los Angeles, CA 90089
Email: {yinuozha,anandvp,prasanna}@usc.edu

***Abstract*—Unified Fuzzy Ontology Matching (UFOM) is an ontology matching system designed to discover semantic links between large real-world ontologies populated with entities from heterogeneous sources. In such ontologies, several entities in different ontologies are expected to be related to each other but not necessarily with one of the typical well-defined correspondence relationships (equivalent-to, subsumed-by). In particular, we define a new kind of correspondence relation called Relevance that reflects the relation between entities when they share a certain amount of mutual information. UFOM uses fuzzy set theory as a general framework for fuzzy ontology alignment. The framework enables representation of multiple types of correspondence relations and characterization of the uncertainty in the correspondence discovery process. UFOM computes multiple measures of similarity among ontology entities – syntactic, semantic, and structural. These measures are composed in a principled manner for ontology alignment. The system is evaluated using publicly available ontologies provided by Ontology Alignment Evaluation Initiative (OAEI). The performance of the proposed system is comparable to the top performing ontology matchers in OAEI. We also evaluate the UFOM system on a dataset from an enterprise application domain.**

I. Introduction

The increasing amount of data that is captured in different information stores has led to the need for automatically discovering correspondences among these data sources. Of the different means of storing data, automatically linking ontologies created with Semantic Web technologies has generated particular research interest [1]. An ontology represents information as a set of formally defined concepts within a domain. Ontologies are semantically linked by discovering *alignments* between the entities contained in different ontologies. Researchers have developed different methods to discover such correspondences among ontologies [1], [2].

In this work, we address an issue that arises when semantically linking large real-world ontologies populated with entities from heterogeneous sources. A desirable feature for such ontology matching systems is the ability to represent and detect different types of relations that can exist between entities. In real-world datasets, entities in an ontology do not always correspond to single physical entities (even in cases where the schema is well-defined, the instance population process may not always follow the schema). In such cases, several entities in different ontologies are expected to be related to each other but not necessarily with one of the typical well-defined relationships (equivalent-to, is-a, part-of, subsumed-by). Thus, relations such as *equivalence* and *disjointedness* [1] do not represent all the possible relations that exist between entities in ontologies. Intuitively, there are entities that are related since they share a certain amount of mutual information. However, most of the existing systems for ontology matching focus on computing only the specific relations of equivalence [3]–[9] and subsumption [5].

We present a system called UFOM (Unified Fuzzy Ontology Matching) that is explicitly designed to discover entities that are *relevant* to each other across multiple ontologies (in addition to the previously defined relationships of equivalence and subsumption). We present a new type of relation called *Relevance* that is designed to represent relationships between entities that are not covered by strict definitions of equivalence and subsumption. The system uses fuzzy set theory to represent the inherent uncertainty in the discovered correspondences and generates a fuzzy ontology alignment.

A challenge in identifying such relations is that the degree of relatedness varies, i.e., it is not accurate to declare whether two entities are related or not. Instead, such relationships have a degree of uncertainty fundamentally associated with them. Recent research in Semantic Web and ontology matching has developed methods of handling uncertainty. In general, these methods are based on two alternate means of representing uncertainty: Probability Theory [10]–[12] or Fuzzy Set Theory [13]–[15].

Ontology matching systems capable of working on real-world datasets have to use a multiple similarity metrics to identify the variety of entities and relations between them. These include syntactic, semantic, and structural features [1]. UFOM computes multiple measures of similarity among ontology entities. This is because different types of semantic relationships require different means of detecting the relation. As the number of such similarity metrics increase, it becomes increasingly complex to aggregate the results of the individual metrics. We describe how UFOM composes these multiple similarity measures in a principled manner using fuzzy set theory for ontology alignment.

When computing the degree to which a particular relationship holds between two entities in different ontologies, it is important to evaluate the supporting evidence. For instance, if there is a large number of instances related to the two entities of interest and these instances show a high degree of similarity, then our degree of confidence in the relationship between the entities is high. In a real-world application, this confidence has to be reported along with the ontology alignment. In particular, this confidence should be used while querying linked ontologies in order to rank results by the certainty of correctness. We describe the multiple factors that are used in UFOM to compute the uncertainty in a correspondence.

The main contributions of this paper are 1) a formal definition of the concept of fuzzy ontology alignment, 2) introduction of a new type of correspondence relation, *Relevance*,

and demonstration of its use in a real-world application, and 3) development of a unified framework that integrates multiple measures of computing similarity using the principle of fuzzy set theory. We evaluate the system using publicly available ontologies provided by Ontology Alignment Evaluation Initiative (OAEI) campaigns. The accuracy of the matchings computed by UFOM is comparable with those of the best performing systems in the OAEI campaigns. We also show experimental results on a dataset from an enterprise application. To the best of our knowledge, UFOM is the first system to formally define a fuzzy representation of ontology matching and the new correspondence relation of *Relevance*. UFOM is also the first system based on a unified framework to generate ontology alignments for computing different types of correspondences.

The rest of the paper is organized as follows. Section II gives a brief overview of the related work on ontology matching. In Section III, we present background concepts and the provide the problem definition. Section IV shows how we design and develop UFOM and Section V briefly discusses how queries can be executed using the computed ontology alignment. In Section VI, we present the experimental evaluation on UFOM. We conclude the paper in Section VII.

II. Related Work

A. Ontology Matching

An ontology matching system aims to discover correspondences between entities in two ontologies. Most of existing systems adopt multiple strategies to fully utilize information contained in the ontologies. In general, these strategies can be classified as terminological, structural and instance-based.

[3] proposes an ontology matching system, SAMBO, for biomedical applications. They use n-grams and edit distances to compute the similarity between the names of two ontology entities. They also consider structural information of the ontology such as *is-a* and *part-of* to calculate the hierarchical similarity between two entities. In [4], the authors propose a divide-and-conquer approach to ontology matching. Specifically, the system partitions the ontologies into small clusters based on structural proximities between entities. Based on these pre-matched entities, similar blocks are discovered out of the clusters. Finally, a linguistic matcher and an iterative structural matcher are combined to find correspondences between the matched blocks. This system performs well in dealing with large ontologies. Another matching system handling large-scale ontologies is proposed in [5]. Specifically, they build a query graph for each ontology. Multiple terminological similarity approaches (e.g., Monger-Elkan and Jaccard distances) combined with Dempster-Shafer theory [16] are adopted to find the relevant graph fragments of two ontologies. Rule-based fuzzy inference is used in the final alignment generation phase. RiMOM [6] is a multi-strategy ontology matching system which considers both entity labels and instances. In order to capture structural properties, it adopts three similarity propagation strategies: class-to-class, property-to-property, and class-to-property. The choice of the strategy and the weights for similarity combination are dynamically determined based on the features of different ontologies. Semantic verification is introduced in ASMOV [7]. ASMOV is different from the above-described systems in that it examines five types of patterns (e.g., disjoint-subsumption contradiction, subsumption incompleteness) on the correspondences derived in the similarity calculation process. The verification process halts when no more correspondences are discovered. [8] introduces a concept called *anchor* which is a correspondence with exactly matched concepts. Given a set of anchors, the system gradually analyzes its neighbors to enrich the correspondence set. [9] considers user feedback in its system. It has a well-developed user interface which enables users to exercise more control over the ontology matching process.

Most of the above systems discover 1-to-1 correspondences. Only [7] and [9] consider n-to-m alignments. These systems also focus only on equivalence relation extraction except [5] which also computes the subsumption relation. Moreover, the result correspondences are all exact. This is in contrast to UFOM where we provide a single framework to compute multiple alignments based on different relations simultaneously.

B. Uncertain Ontology Matching

Mitra et al. [12] introduce a probabilistic ontology matching framework which uses Bayesian Networks to represent the influences between potential correspondences in different ontologies. They present a probabilistic representation of ontology matching rules and inference to refine the quality of an existing ontology alignment. Albagli et al. [17] propose a probabilistic framework for ontology matching based on Markov networks. In their framework, approximate reasoning is adopted to reduce the high computational complexity associated with their uncertainty model. Existing correspondences are used as training data to generate new ones and user feedback is incorporated to provide an interactive semi-automatic matching process.

Besides probability theory, fuzzy set theory has recently been used for ontology matching. Todorov et al. [15] propose an ontology matching framework for multiple domain ontologies. They represent each domain concept as a fuzzy set of reference concepts. Then, the matches between domain concepts are derived based on their corresponding fuzzy sets. In [13], a rule base provided by domain experts is used in the matching process. In that system, both Jaccard coefficient and linguistic similarity are first calculated for each pair of entities. Then, the system uses the rule base to generate the final similarity measure of each correspondence. The fuzzy set is used as a link between the preliminary similarities and the final one. Though both of the above-described works adopt fuzzy set theory for the ontology matching task, they do not provide a formal definition of a fuzzy representation of correspondence. Moreover, these systems are specific to equivalence type relations. As in the case of the systems described in Section II-A, there is no generic framework identifying correspondences for different types of relations. In this paper, we address these two issues.

III. Problem Definition

We first present the basic concepts of fuzzy set theory [18] which forms the basis for our framework. We then define the problem of fuzzy ontology matching.

A. Fuzzy Set Theory

Definition 1: A **fuzzy set** is a set of ordered pairs, given by $A = \{(x, \mu_A(x)) : x \in X\}$ where X is a universal set and $\mu_A(x)$ is the grade of membership of x in A which lies in [0,1].

For example, a fuzzy set BABY (representing the set of all babies) can be defined as BABY = $\{(x, \mu_{BABY}(x)\}$ where $\mu_{BABY}(x) = exp(-x)$. Here, x is the age of a person and as $x \to 0$, $\mu_{BABY}(x) \to 1$.

Definition 2: A **fuzzy relation** between sets X and Y is given by $R(x, y) = \{\mu_R(x, y)/(x, y)|(x, y) \in X \times Y\}$ where real number $\mu_R(x, y)$ denotes the fuzzy membership of relation $R(x, y)$.

For example, the membership function may be $\mu_R(x, y) = exp[-(x-y)^2]$ where x and y are real numbers and the fuzzy relation $R(x, y)$ describes how close x is to y. In our general ontology matching framework, every type of relation that is to be discovered between entities in two ontologies have to be described using an appropriate fuzzy relation. For example, the equivalence relation, equ, between two numeric entities, x and y, may be defined as

$$\mu_{equ} = \exp\left[-\left(\sum_i \min_j |x_i - y_j| + \sum_j \min_i |x_i - y_j|\right)\right]$$

where x_i and y_j refer to the instances in x and y respectively. This membership function is maximized when every instance in x is also present in y and vice-versa. (Note that our UFOM system, described in detail in Section IV, uses a more complex membership function for the Equivalence relation.)

B. Ontology Matching

The ontology matching problem aims to find an alignment A for a pair of ontologies O_1 and O_2. An alignment is defined as follows [1],

Definition 3: An **ontology alignment** A is a set of *correspondences* between entities of the matched ontologies O_1 and O_2.

Definition 4: An **ontology correspondence** is a 4-tuple: $< id, E_1, E_2, r >$,

where id is the identifier for the given correspondence, E_1 and E_2 are the entities of the correspondence (e.g., properties in the ontology), and r is the relation between E_1 and E_2 (e.g, equivalence and disjointedness).

Note that in the above definition, the correspondence is exact, i.e., the relation r strictly holds between the ontology entities E_1 and E_2. However, in systems that have to automatically determine the set membership from real-world data, it is natural that a degree should be associated with the relation between the entities. The higher the degree is, the higher the likelihood that the relation holds between them. In order to represent the uncertainty in the correspondences, we present a fuzzy variant of ontology alignment.

Definition 5: A **fuzzy ontology alignment** is a set of **fuzzy correspondences** in the form of 6-tuple: $< id, E_1, E_2, r, s, c >$

where $s = \mu_r(E_1, E_2)$ which is the *relation score* denoting the membership of (E_1, E_2) in relation r, and c is the *confidence score* computed while generating the correspondence. With this definition of fuzzy correspondence, we can extend the relation type set with some other useful types. Next, we formally define a new type of relation called *Relevance*.

C. Relevance

We first motivate the new relation type with an example. Let O_1 be an ontology built for all products in a book store and O_2 be an ontology based on ISBN for all books. Let the *Description* property of O_1 contain the title and author information of a book. This information can also be found separately in the *Title* and *Author* properties in O_2. The relation between *Description* and *Title*, or *Description* and *Author* is neither subsumption nor part-of because *Description* is input by human beings and may not contain exact *Title* or *Author* of a book. Also *Description* may have many null values and we aim to model the degree of how two entities are associated.

In general, the same set of physical entities may be represented with multiple associations across different individuals in an ontology. We therefore define a new relation that we call *Relevance* as a general representation of the different possible associations between entities. In contrast with the typical relations used in existing ontology matching systems, we define relevance as a fuzzy relation instead of a Boolean one. We define *Relevance* as follows.

Definition 6: $Rel(E_1, E_2) = Pr\left(I(e_1) \subset I_2 | e_1 \in E_1\right)$

Here, $I(e)$ represents the set of physical entities referred to by an instance e and I_t represents the set of physical entities referred to by instances of entity E_t. The relevance score between properties E_1 and E_2 represents the probability that physical entities referred to by instances of E_1 are also referred to by instances in E_2. Note that the actual implementation of a Relevance relation between two entities will depend on the specific application in which they appear. In this work, we present one implementation of the Relevance relation (Section VI) and evaluate it on an enterprise application.

Next, we present the UFOM system to automatically generate a fuzzy ontology alignment. UFOM implements a unified framework for fuzzy ontology matching with different types of relations in the correspondences.

IV. Unified Fuzzy Ontology Matching (UFOM)

Our framework for computing fuzzy ontology alignments is based on computing both the similarity score and confidence score for every possible correspondence in the ontologies. In order to provide an extensible framework, every relation score (i.e., for every type of relation of interest) is computed from a set of pre-defined similarity functions. The framework is illustrated in the design of the UFOM system (Figure 1).

UFOM takes as input two ontologies and outputs a fuzzy alignment between them. UFOM consists of four components: Preprocessing Unit (PU), Confidence Generator (CG), Similarity Generator (SG), and Alignment Generator (AG). PU identifies the type of each entity in the ontology and classifies the entities based on their types. Different computation strategies are adopted for matching the entities with the most

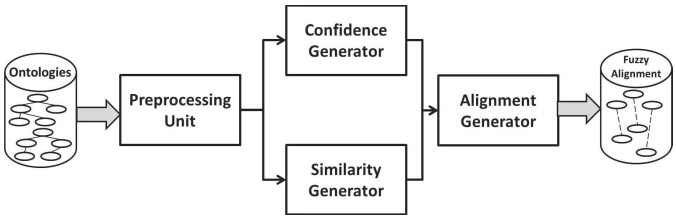

Fig. 1: Components of the UFOM system for computing fuzzy ontology alignment

appropriate type. CG quantifies the quality of the resources used to generate a potential match between two entities. Volume and variety are the two major factors considered in this step. SG computes multiple types of similarities for every pair of entities and generates a vector of similarity scores. These similarity scores are the component functions for computing any particular type of relation. Finally, AG calculates the relation score using the fuzzy membership functions for each relation type and constructs the correspondence based on both this relation score and confidence score. We next describe these components in greater detail. We use uppercase letters to denote an entity (T for a Class and E for a Property). We use lowercase letters to denote instances of an entity (t for class instances and e for property values).

A. Preprocessing Unit

The Preprocessing Unit is a constraint-based classifier. It classifies the entities based on their types. We assume that the input ontologies are RDF or OWL files. We define three subtypes of the `DatatypeProperty`: `String`, `Datetime`, and `Numerical`. Specifically, an entity is classified as one of the following types: `ObjectProperty`, `String DatatypeProperty`, `Datetime DatatypeProperty`, and `Numerical DatatypeProperty`.

For DatatypeProperty, `rdfs:range` is used to decide which specific DatatypeProperty it belongs to. If the range is `xsd:decimal`, `xsd:float`, `xsd:double` or `xsd:integer`, it is a Numerical DatatypeProperty. If the range is `xsd:dateTime`, `xsd:time` or `xsd:date`, it is a `Datetime DatatypeProperty`. If the range is none of above, it is a `String DatatypeProperty`. Based on this entity classification, different matching algorithms will be applied during similarity generation (Section IV-C).

B. Confidence Generator

The Confidence Generator computes a confidence score for each correspondence which reflects the sufficiency of the underlying data resources used to generate this correspondence. For correspondence between properties, their instances are the main resources. Intuitively, the more instances that are used for computing similarity, the more confident we can be in the matching process. In order to quantify the sufficiency of the properties, we identify two metrics – *Volume* and *Variety*. The volume of resources related to property E of class T is defined as

$$VO(E) = \frac{|\{t \in T | t.E \neq \emptyset\}|}{|\{t \in T\}|} \tag{1}$$

where $|\{t \in T\}|$ is the number of instances of class T and $|\{t \in T | t.E \neq \emptyset\}|$ is the number of instances of class T with a non-null value for property E.

The variety of resources related to properties E_1 and E_2, $VA(E_1, E_2)$, quantifies the variety of the property instances using the concept of entropy. It is defined as

$$VA(E_1, E_2) = \frac{H(E_1) + H(E_2)}{\log(|v(E_1)| + |v(E_2)|)} \tag{2}$$

where $v(E)$ is the set of non-null values of entity E (i.e., $v(E_i) = \{e | e \neq \emptyset, e \in E_i\}$) and $H(E_i)$ is the entropy of the values of property E_i:

$$H(E_i) = -\sum_{x \in v(E_i)} p_i(x) \log(p_i(x)) \tag{3}$$

$p_i(x)$ is the probability of property value x appearing as E_i's instance:

$$p_i(x) = \frac{|\{t | t.E_i = x\}|}{|\{t | t.E \neq \emptyset\}|} \tag{4}$$

Intuitively, if the properties E_1 and E_2 have a large number of unique values relative to the number of class instances having those properties, then they have a large variety score.

Based on the volume and variety of the two properties, the confidence score of their correspondence is calculated as follows.

$$c(E_1, E_2) = \frac{VO(E_1) + VO(E_2) + VA(E_1, E_2)}{4} \tag{5}$$

C. Similarity Generator

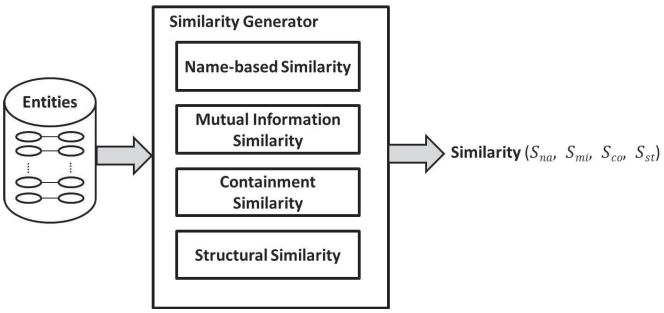

Fig. 2: Components of the UFOM Similarity Generator

The Similarity Generator generates a vector of similarities between two entities. The similarities form the basis for computing different types of relation correspondences (using their respective fuzzy membership functions). Each component of the vector represents a specific type of similarity. In UFOM, the vector consists of four values: Name-based Similarity, Mutual Information Similarity, Containment Similarity, Structural Similarity (Figure 2). Next, we describe these four types of similarity.

1) Name-based Similarity: The name-based similarity is calculated based on both the semantic similarity s_{se} and syntactic s_{sy} similarity of between the names of the two properties. Intuitively, the name denoting a property typically captures the most distinctive characteristic of the instances. For

semantic similarity, the following steps are used to generate s_{se},

1) Tokenize the names of both properties. Denote by $E_i.\text{TOK}_j$ as the j-th token in the name of property E_i.
2) Retrieve the synset of each token using Open Linked Data (WordNet)[1]. Denote by $syn(w)$ the WordNet synset of a word w.
3) Calculate the Jaccard similarity[2] on the synsets of each pair of tokens
4) Return the average-max Jaccard similarity as the semantic similarity:

$$s_{se}(E_1, E_2) = \frac{1}{n} \times$$

$$\sum_i \max_j Jac(syn(E_1.\text{TOK}_i), E_2.\text{TOK}_j))$$

Here, $Jac()$ represents the Jaccard similarity between two sets and n is the number of tokens in E_1's name.

For example, consider two properties named **complete_date** and **EndDate**. In the first step, the two names are tokenized into two sets {"complete", "date"} and {"End", "Date"}. In the second step, the synsets of "complete", "date" and "end" are retrieved from WordNet. In the third step, we calculate the Jaccard similarity for each pair of tokens ("complete"-"end", "complete"-"date", "date"-"end" and "date"-"date"). Finally, we find the maximum Jaccard similarity for each token in the first property name and average the similarity over all tokens in the first property name. In this example, $s_{se} = \frac{1}{2} \times (Jac(syn(\text{“}complete''), syn(\text{“}end'')) + Jac(syn(\text{“}date''), syn(\text{“}date'')))$.

For syntactic similarity, Levenshtein distance [19] is adopted as the distance metric between the names of the two properties. Formally, the syntactic similarity, $s_{sy}(E_1, E_2)$ between two properties E_1 and E_2 is

$$s_{sy}(E_1, E_2) = 1 - \frac{Lev(E_1.\text{Name}, E_2.\text{Name})}{max(|E_1.\text{Name}|, |E_2.\text{Name}|)} \quad (6)$$

where $E_i.\text{Name}$ denotes the name of property E_i, $|E_i.\text{Name}|$ is the length of the name string, and $Lev(w_1, w_2)$ is the Levenshtein distance between two words w_1 and w_2. In the above example, the syntactic similarity between **complete_date** and **EndDate** is $1 - \frac{8}{13}$=0.3846.

The name-based similarity is a weighted sum of the above two similarities:

$$s_{na}(E_1, E_2) = \omega_{se} s_{se}(E_1, E_2) + \omega_{sy} s_{sy}(E_1, E_2) \quad (7)$$

The weights are pre-defined as system parameters in UFOM. In our future work, we intend to use machine learning techniques to learn the weights from labeled datasets.

[1]http://wordnet.princeton.edu/

[2]http://en.wikipedia.org/wiki/Jaccard_index

2) Mutual Information Similarity: The Mutual Information Similarity aims to model the mutual information that exists between the *individuals* of one entity and the *domain* represented by the second entity. Intuitively, if two properties have a high proportion of instances shared between them, then this is indicative of these properties being highly related. Specifically, Mutual Information Similarity increases when all the components describing an individual of the first entity are also found in the universe of components defined by all the instance of the second entity. In UFOM, we model this by computing the probability that all words from an individual from the first entity appear in the set of words created by the union of the individuals of the second entity.

Let W_2 denote the set of words in the instances of E_2 and $|W_2|$ is the total number of such words. Let $n(w, E_2)$ denote the number of appearances of the word w in the instances of E_2. Then, the conditional probability of a word w appearing in some instance of E_2 can be estimated by

$$p(w|E_2) = \frac{n(w, E_2)}{|W_2|} \quad (8)$$

The joint probability of all words in a set S, also appearing in the word set of E_2, assuming independence of these word occurrences in E_1, is given by

$$p(S \subset W_2) = \prod_{w \in S} p(w|E_2) \quad (9)$$

Let $w \in e_1$ denote the words comprising the instance e_1. Then, the Mutual Information Similarity, $s_{mi}(E_1, E_2)$ between two properties E_1 and E_2 is defined as

$$s_{mi}(E_1, E_2) = \frac{1}{|v(E_1)|} \sum_{e_1 \in E_1} p(e_1 \subset W_2) \quad (10)$$

$$= \frac{1}{|v(E_1)|} \sum_{e_1 \in E_1} \prod_{w \in e_1} \frac{n(w, E_2)}{|W_2|} \quad (11)$$

3) Containment Similarity: Containment Similarity models the average level of alignment between an instance of property E_1 and the most similar instance in property E_2. Containment similarity is designed to detect pairs of properties that share a large number of common instances even if the instances are misaligned. Specifically, it is calculated as

$$s_{co}(E_1, E_2) = \frac{1}{|v(E_1)|} \sum_{e_1 \in E_1} \max_{e_2 \in E_2} SWS(e_1, e_2) \quad (12)$$

SWS(e_1,e_2) calculates the Smith-Waterman Similarity [20] between two instances e_1 and e_2 treated as a sequence of characters. The Smith-Waterman Similarity identifies local sequence alignment and finds the most similar instances of E_2 given an instance of E_1. For example, if e_1 is "Stephen W. Hawking" and e_2 is "A Brief History of Time by Stephen Hawking", then, SWS(e_1,e_2)=$\frac{15}{18}$=0.8333. Note that Containment Similarity is asymmetric. Intuitively, the greater the number of instances in E_2 compared to the number of instances in E_1, the more likely that E_1 is contained within E_2.

4) Structural Similarity: The fourth value in the vector of similarities is designed to capture the structural similarity between two properties as they are represented within their ontologies. We represent the ontology as a graph with properties as edges and classes as nodes. Intuitively, if two properties have similar domains and ranges (classes), then they are assigned high similarity. In turn, two classes that have similar properties should have higher similarity. The following iterative equations are used to refine the edge (property) similarity. At every iteration t, the node similarity, $s_N^t()$ is updated based on the edge similarity of iteration t, $s_E^t()$.

$$s_N^t(n_i, n_j) = \frac{1}{2N} \sum_{s(k)=i, s(l)=j} s_E^t(e_k, e_l) +$$

$$\frac{1}{2M} \sum_{t(k)=i, t(l)=j} s_E^t(e_k, e_l) \quad (13)$$

Then, the edge similarity of the next iteration $t+1$, $s_E^{t+1}()$ is calculated from the node similarity of iteration t, $s_N^t()$.

$$s_E^{t+1}(e_i, e_j) = \frac{1}{2}\left(s^t(n_{s(i)}, n_{s(j)}) + s^t(n_{t(i)}, n_{t(j)})\right) \quad (14)$$

In these equations, $s(k) = i$ and $t(k) = j$ denotes that edge e_k is directed from node n_i to n_j, i.e., property E_k has domain class T_i and range class T_j. N is the number of properties which have its domain as T_i multiplied by the number of properties which have its domain as T_j. Similarly, M is the number of properties which have its range as T_i multiplied by the number of properties which have its range as T_j.

The initial edge similarities are set from the weighted sum of the previously defined three similarities.

$$s_E^0(e_i, e_j) = \omega_{na}\delta(s_{na}(E_i, E_j)) +$$

$$\omega_{mi}\delta(s_{mi}(E_i, E_j)) + \omega_{co}\delta(s_{co}(E_i, E_j)) \quad (15)$$

where ω_{na}, ω_{mi} and ω_{co} are pre-defined weights in UFOM and

$$\delta(x) = \frac{1}{1 + e^{-k(x-\alpha)}}$$

A non-linear addition (δ function) is used to better distinguish between similarity values that are close to the median than a linear weighted sum. In this work, we have set $k = 5$. As in the case of name-based similarity, we will consider using machine learning techniques to learn these weights automatically from labeled ontology matching data.

The algorithm halts when $|s_E^{t+1}(e_i, e_j) - s_E^t(e_i, e_j)| < \epsilon$ where ϵ is a pre-defined parameter. The structural similarity of two properties are then set as

$$s_{st}(E_i, E_j)) = s_E^{t+1}(e_i, e_j) \quad (16)$$

Finally, the similarity generator outputs a vector of similarities for each pair of properties. The vector for property pair (E_i, E_j) is represented as $\overrightarrow{s(E_i, E_j)} = (s_{na}(E_i, E_j), s_{mi}(E_i, E_j), s_{co}(E_i, E_j), s_{st}(E_i, E_j))$.

D. Alignment Generator

The output of the Alignment Generator is a set of fuzzy correspondences in the form of 6-tuples: $< id, E_1, E_2, r, s, c >$. For the confidence score c, it has already been calculated by CG. In order to calculate the relation score s, a set of membership functions are pre-defined in UFOM. Each membership function corresponds to one type of relation. For example, the membership function for equivalence can be defined as a linear combination of all similarity values ($\mu_{equ}(\overrightarrow{s}) = \frac{1}{4}(s_{na} + s_{mi} + s_{co} + s_{st})$), the membership function for subsumption can be approximated by a combination of two similarity values ($\mu_{sub}(\overrightarrow{s}) = \frac{2}{3}s_{mi} + \frac{1}{3}s_{st}$) and the membership function for relevance can be written as a non-linear s-Function ($\mu_{rel}(\overrightarrow{s}) = S(s_{co}; 0.2, 0.5, 0.7)$). Then, the relation score s is computed by the corresponding membership function.

Once both s and c are derived, AG prunes the correspondences with s and c less than pre-defined cutoff thresholds s_δ and c_δ. Different applications will have different thresholds. For example, a recommendation system may have relatively low thresholds since false positives are tolerated, while a scientific application may have high thresholds.

Below are some examples of fuzzy correspondence:

$< 1, isbn : author, bkst : desc, relevance, 0.73, 0.92 >$

$< 2, isbn : author, bkst : desc, equivalence, 0.58, 0.92 >$

$< 3, isbn : author, bkst : pub, disjoint, 0.83, 0.75 >$

V. Query Execution

In this section, we briefly discuss how the fuzzy alignment helps during query execution over the matched ontology. Given two ontologies, O_1 and O_2, the query execution problem is as follows. Given an individual t in O_1, return all individuals of O_2 which are *relevant* to t. If there is no alignment, the only way to solve it is to compare each property value of t with all property instances in O_2. The time complexity of this approach is $O(|O_2||t||E_2|)$ where $|O_2|$ is the number of individuals in O_2, $|t|$ is the number of properties that t has and $|E_2|$ is the number of properties in O_2.

With the help of the fuzzy alignment, the computation burden can be reduced. One strategy is to find the fuzzy correspondences, setting E_1 as I's properties and r as *relevance*. Then each property value of t is compared only with the instances of properties appearing as E_2 in the correspondences. Specifically, the Smith-Waterman approximate substring matching algorithm can be used for comparison. The time complexity is $O(|O_2||t||E_2(I)|)$ where $|E_2(I)|$ is the average number of properties in O_2 which have correspondences with a property of I.

Another strategy is to have multi-layer references. The idea is to identify some intermediate classes with properties having correspondences with both I's properties and the target class' properties. Then the previous strategy is applied twice to find the answer instances of the target class.

VI. Experimental Evaluation

In this section, we present the evaluation of UFOM on various datasets. We mainly consider two types of relations – Equivalence(SectionVI-A) and Relevance(SectionVI-B).

A. Equivalence

1) Dataset: We performed a set of experiments using two sets of ontologies provided by Ontology Alignment Evaluation Initiative (OAEI) 2013 Campaign. The first dataset is Conference Ontology[3] which contains 16 ontologies. In our experiments, we aim to find the alignment between one specific ontology *Cmt* and multiple ontologies including *confOf, sigkdd, conference, edas, ekaw, iasted*. In total, Cmt has 59 properties and the target ontology set has 188 properties. The second set is Instance Matching (IM) Ontology[4]. It has 5 ontologies and 1744 instances.

The weights for the name-based similarity and the structural similarity are set as $\frac{1}{2}$ and $\frac{1}{3}$. The membership function is constructed as

$$\mu_{equ}(\overrightarrow{s}) = \frac{s_{na} + s_{mi} + s_{co} + s_{st}}{4} \tag{17}$$

For Conference Ontology, no instances are provided. As a result, $s_{mi} = s_{co} = 0$ and $c = 0$. For Instance Matching Ontology, the confidence threshold c_δ is set as 0.6.

2) Results: For Conference Ontology, the relation score threshold is set as 0.25, 0.3, 0.35, and 0.4 (the upper bound of the score is 0.5 since $s_{mi} = s_{co} = 0$). We compute precision, recall, and F-measure to evaluate the performance of UFOM.

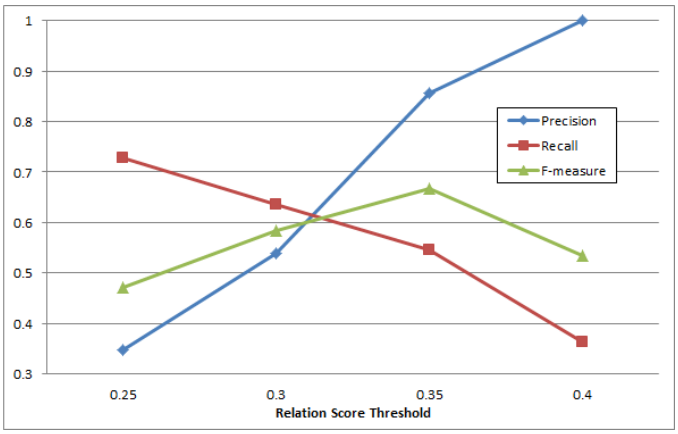

Fig. 3: Precision, recall, and f-measure on applying UFOM to the Conference Ontology matching problem

As the relation score threshold increases, precision increases while recall and F-measure decrease (Figure 3). Compared with the baseline matcher (*StringEquiv*) in OAEI 2013, when recall is the same (0.43), precision for UFOM is 0.92 while for the baseline is 0.8.

Similarly, we evaluate UFOM on Instances Matching Ontology. Since this ontology has instances, the upper bound of the score is 1.0. Thus, we tune the relation score threshold from 0.5 to 0.8 with step 0.1.

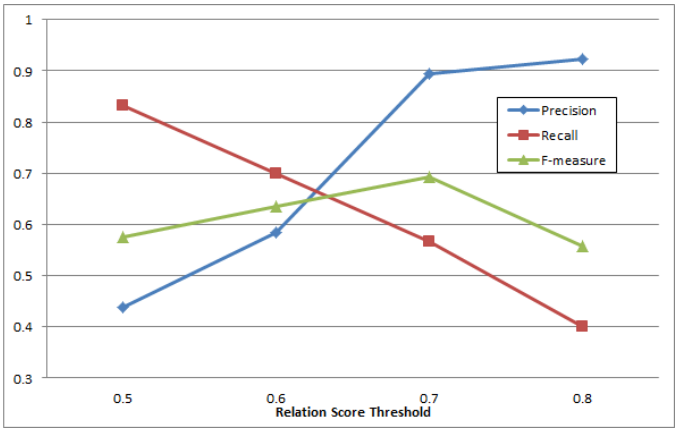

Fig. 4: Precision, recall, and f-measure on applying UFOM to the Instance Matching Ontology

A similar trend is shown in Figure 4. The difference is that precision, recall and F-measure are slightly higher than the results using Conference Ontology at most data points. The reason is that having instances in generating the correspondences can improve the quality of matching.

B. Relevance

1) Experimental Setting: For the evaluation of computing relevance alignment, we used two ontologies O_1 and O_2 from an enterprise-scale information repository. Each ontology focuses on a different application area within the enterprise, but they are related in terms of the entities that they reference. Ontology O_1 has 125865 triples. Ontology O_2 has 651860 triples. Due to privacy concerns, we do not expose the real names of the properties and ontologies.

The task is to find all relevant properties in O_2 given a property E in O_1. The weights for the name-based similarity and the structural similarity are set as $\frac{1}{2}$ and $\frac{1}{3}$. The relevance membership function is constructed as

$$\mu_{rel}(\overrightarrow{s}) = \frac{1}{4s_{na}} + \frac{1}{2s_{co}} + \frac{1}{4s_{st}} \tag{18}$$

2) Results: Table I shows the relevance score and confidence score between E and properties in O_2.

We manually labeled each discovered correspondence as being "Highly relevant", "Relevant", or "Irrelevant" based on our understanding of the underlying entities. As can be seen from Table I, the relevance scores align well with the manually assigned labels. When the relevance score threshold is set as 0.2, precision and recall reach 1.0. We also evaluated query execution using the same experimental setup. We selected 10 representative individuals from O_1 and retrieve their relevant instances from O_2 using the fuzzy alignment. Both precision and recall achieve 1.0 after we verified the results with the ground truth obtained by manually examining the ontologies for each of the automatically retrieved entities.

[3]http://oaei.ontologymatching.org/2013/conference/index.html
[4]http://www.instancematching.org/oaei/

TABLE I: Computing Relevance relation between entities in the enterprise application

O_2 Property	Relevance	Confidence	Domain Expert
E_1	73.93%	92.85%	Highly Relevant
E_2	38.94%	87.73%	Relevant
E_3	38.89%	93.21%	Relevant
E_4	30.89%	87.73%	Relevant
E_5	23.95%	97.98%	Relevant
E_6	0.67%	77.31%	Irrelevant
E_7	0.67%	76.26%	Irrelevant
E_8	0.67%	72.98%	Irrelevant

VII. Conclusion

We formally defined a fuzzy set representation of an ontology alignment system. The unified framework for fuzzy ontology matching can derive fuzzy correspondences with different relation types and is able to discover a fuzzy alignment between two ontologies. We also introduced a novel correspondence relation type called Relevance. In the experiments performed on publicly available datasets, we showed that our proposed framework achieves high precision, recall and F-measure for generating the correspondences of both equivalence and the newly-defined relevance relation types.

In our future work, we intend to formalize the query execution process on UFOM. At the same time, various query optimization strategies will be developed to facilitate efficient query execution.

Acknowledgment

This work is supported by Chevron U.S.A. Inc. under the joint project, Center for Interactive Smart Oilfield Technologies (CiSoft), at the University of Southern California.

References

[1] P. Shvaiko and J. Euzenat, "Ontology matching: State of the art and future challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 1, pp. 158–176, 2013.

[2] N. Choi, I.-Y. Song, and H. Han, "A survey on ontology mapping," *SIGMOD Record*, vol. 35, no. 3, pp. 34–41, 2006.

[3] P. Lambrix and H. Tan, "Sambo - a system for aligning and merging biomedical ontologies," *J. Web Sem.*, vol. 4, no. 3, pp. 196–206, 2006.

[4] W. Hu, Y. Qu, and G. Cheng, "Matching large ontologies: A divide-and-conquer approach," *Data Knowl. Eng.*, vol. 67, no. 1, pp. 140–160, 2008.

[5] M. Nagy and M. Vargas-Vera, "Multiagent ontology mapping framework for the semantic web," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 41, no. 4, pp. 693–704, 2011.

[6] J. Li, J. Tang, Y. Li, and Q. Luo, "Rimom: A dynamic multistrategy ontology alignment framework," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 8, pp. 1218–1232, 2009.

[7] Y. R. Jean-Mary, E. P. Shironoshita, and M. R. Kabuka, "Ontology matching with semantic verification," *J. Web Sem.*, vol. 7, no. 3, pp. 235–251, 2009.

[8] M. S. Hanif and M. Aono, "An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size," *J. Web Sem.*, vol. 7, no. 4, pp. 344–356, 2009.

[9] I. F. Cruz, F. P. Antonelli, and C. Stroe, "Agreementmaker: Efficient matching for large real-world schemas and ontologies," *PVLDB*, vol. 2, no. 2, pp. 1586–1589, 2009.

[10] S. Albagli, R. Ben-Eliyahu-Zohary, and S. E. Shimony, "Markov network based ontology matching," *J. Comput. Syst. Sci.*, vol. 78, no. 1, pp. 105–118, 2012.

[11] M. Niepert, C. Meilicke, and H. Stuckenschmidt, "A probabilistic-logical framework for ontology matching," in *AAAI*, 2010.

[12] P. Mitra, N. F. Noy, and A. R. Jaiswal, "Omen: A probabilistic ontology mapping tool," in *International Semantic Web Conference*, 2005, pp. 537–547.

[13] S. Fernández, J. R. Velasco, I. Marsá-Maestre, and M. A. López-Carmona, "Fuzzyalign - a fuzzy method for ontology alignment," in *KEOD*, 2012, pp. 98–107.

[14] V. Cross and X. Hu, "Fuzzy set and semantic similarity in ontology alignment," in *FUZZ-IEEE*, 2012, pp. 1–8.

[15] K. Todorov, P. Geibel, and C. Hudelot, "A framework for a fuzzy matching between multiple domain ontologies," in *KES (1)*, 2011, pp. 538–547.

[16] G. Shafer, *A Math. Theory of Evidence*. Princeton Univ. Press, 1976.

[17] S. Albagli, R. Ben-Eliyahu-Zohary, and S. E. Shimony, "Markov network based ontology matching," in *IJCAI*, 2009, pp. 1884–1889.

[18] "Fuzzy sets and relations," in *Computational Intelligence*. Springer Berlin Heidelberg, 2005, pp. 37–66. [Online]. Available: http://dx.doi.org/10.1007/3-540-27335-2_2

[19] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals," *Soviet Physics-Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[20] T. Smith and M. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195 – 197, 1981. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0022283681900875