Canonical Forms and Similarity of Complex Concepts for Improved Ontology Alignment

Tejas Chaudhari, Uthayasanker Thayasivam and Prashant Doshi
THINC lab, Dept. of Computer Science
University of Georgia, Athens, USA
Email: {tejas,uthayasa,pdoshi}@cs.uga.edu

***Abstract*—Modern ontology languages such as the Web Ontology Language (OWL) allow defining complex concepts that involve restrictions, Boolean combinations, and exhaustive enumeration of individuals. Many of the current ontology alignment algorithms either do not consider the complex concepts in their alignment procedures or model them naively, thereby producing a possibly incomplete alignment. We introduce axiomatic and graphical canonical forms for modeling value and cardinality restrictions and Boolean combinations, and present a way of measuring the similarity between these complex concepts in their canonical forms. We integrate our approach in multiple ontology alignment algorithms. Our results indicate a significant improvement in the F-measure of the alignment. However, this improvement is at the expense of increased run time due to the additional concepts modeled.**

***Keywords*-ontology alignment; complex concepts; OWL;**

I. Introduction

Contemporary languages for describing ontologies such as the Web ontology language (OWL 2) and the resource description framework schema (RDFS) identify concepts using the internationalized resource identifier (IRI). RDF allows using *blank nodes* to represent resources that do not have an IRI. Analogously, OWL utilizes anonymous classes to represent certain class descriptions. These include concepts involving restrictions, Boolean combinations of classes and an exhaustive enumeration of individuals. We call these *complex concepts*; these are part of class expressions in OWL 2. Due to the absence of distinguishing labels, complex concepts are insufficiently utilized by many alignment algorithms. Being an important part of the conceptualization in ontologies, ignoring complex concepts often leads to an incomplete and inaccurate alignment. Modeling complex concepts for participation in the alignment is challenging. Though these concepts do not possess the attributes of a named class such as a label, comment and IRI, they contain semantics in the context of their respective ontology.

Understanding the semantics and structure of the complex concepts is a first step toward this goal. In theory, semantic approaches such as S-Match [1] have the potential to discover correspondences between the complex concepts. However, in practice, semantic approaches do not scale – S-Match is limited to small taxonomies – necessitating a combination of fast lexical techniques for discovery of correspondences with partial consideration of semantics, such as for validation as in LogMap. Nevertheless, LogMap by itself too did not identify the correspondences between the complex concepts in our example ontologies.

In this paper, we present a novel and general way of modeling complex concepts. We seek to find the similarity between the anonymous classes that appear in the definition of these concepts, so that it may be utilized by existing algorithms analogously to the other named entities. Alignment algorithms model OWL ontologies either as a set of axioms [1], [2], [3] or as a graph [4], [5], [6], [7]. Consequently, we introduce axiomatic representations of the different types of complex concepts in canonical forms, and additionally derive RDF graph-based canonical representations that model the associated OWL axioms without any loss in meaning. Subsequently, we compare the corresponding entities represented either axiomatically or as subgraphs in their canonical forms (canonicalized), in order to obtain a similarity between the anonymous classes or their graphical representation as blank nodes. This similarity is seamlessly integrated into ontology alignment algorithms.

We study the impact of our approach in the context of three ontology alignment algorithms: Falcon-AO [7], LogMap [2] and Optima+ [6]. Among these, Falcon-AO's lexical matcher utilizes a virtual document for an anonymous class, which is a string formed from concatenating neighboring concepts. The other two algorithms limit their focus to named entities only. Using 2 different testbeds, we demonstrate significant positive impact on the precision of the alignment, with improvement in the recall for some of the algorithms as well at the expense of computation time.

II. Background

In OWL, the restricted class is the only subclass of an anonymous class. The latter is a class that is devoid of an informative IRI and is the class of all individuals that satisfy the restriction. On declaring a restriction, an anonymous class associated with that restriction is created implicitly. Analogous to a restricted class, a Boolean class is a Boolean combination of two or more classes in the ontology. Boolean combination operators include the union, intersection and complement. The combination is implicitly an anonymous

class that is associated with a RDF list of all the classes involved using a property whose name is the Boolean operator. The Boolean class is named and is associated with the anonymous class using *owl:equivalentClass*.

The recent **OWL 2 to RDF graph mapping** [8] provides a transformation, T, that can be used to translate any OWL 2 ontology axiom, O, into an RDF graph, $G = T(O)$, without loss of generality. A reverse mapping, T^{-1}, is also presented, which can be used to transform an RDF graph, G, satisfying certain restrictions into an OWL 2 DL ontology, OG. These transformations do not incur any change in the formal meaning of the ontology [8]. Formally, for any OWL 2 DL ontology O, let $G = T(O)$ be the RDF graph obtained by transforming O as specified, and let $OG = T^{-1}(G)$ be the OWL 2 DL ontology obtained by applying the reverse transformation to G; then O and OG are logically equivalent because they have exactly the same set of models. Blank nodes in RDF are used to represent a resource that is not provided with an IRI. Therefore, the anonymous classes of OWL are mapped to blank nodes in RDF.

III. Modeling Complex Concepts

Alignment algorithms either adopt an axiomatic model of an ontology or an intermediate RDF graph-theoretic model. Each axiom in OWL may be transformed to an equivalent RDF graph [8]. Thus, the OWL constructs that constitute complex concepts may also be represented using subgraphs within the full graphical representation of the ontology.

Our insight is that for the purpose of alignment, the axiomatic structural specifications of the differing types of restrictions and types of Boolean combinations may be partially standardized into canonical forms, and transformed into subgraph representations in canonical form, which are useful for equivalence comparisons. This is significant because there exist 12 different types of property restrictions and 3 different Boolean combination operators in OWL making their comparisons challenging.

A. Canonical Form for Value Restrictions

A value restriction on a class restricts the values of its property's range. The *ObjectAllValuesFrom* or *ObjectSomeValuesFrom* restrictions defined for an object property limits its values to individuals of a class, while data properties are restricted to a data range by *DataAllValuesFrom* or *DataSomeValuesFrom*. The *ObjectHasValue* limits the value of the object property to an individual, a, and *DataHasValue* limits the data property to a literal, lt. For example, the *ObjectAllValuesFrom(has_output_value, Spectral_Count)* restriction defined in the parasite experiment ontology (PEO) for the class *Proteome_analysis* restricts the range of *has_output_value* to take individuals from *Spectral_Count* only. Let us denote an object property expression as OPE and a data property expression as DPE. We refer to a class expression using CE and a datarange using DR.

We observe that the different types of value restrictions admit structural specifications, which may be represented equivalently (though their semantics differ). Subsequently, we introduce the generalized value restriction complex concept and define it in a canonical form, $CE_V = RE_V(PE_V, R_V)$, where $PE_V \in \{OPE, DPE, (DPE_1, \ldots, DPE_k)\}$, $k \geq 2$, is a property expression(s) and $R_V \in \{DR, CE, a, lt\}$ is the value that restricts the range. RE_V is one of the value restriction expressions in {*ObjectSomeValuesFrom*, *ObjectAllValuesFrom*, *ObjectHasValue*, *DataSomeValuesFrom*, *DataAllValuesFrom*, *DataHasValue*}.

Next, we introduce the general transformation, $\mathcal{T}(CE_V) = SG_V$, which is defined as transforming the canonical form value restriction, CE_V, into a subgraph for value restrictions, $SG_V = \langle V_V, E_V, \lambda_V \rangle$, in a canonical form. Here, the set of vertices, $V_V = \{T(PE_V), T(R_V), \text{owl:restriction}, \_ : x\}$, and the set of directed edges, $E_V = \{\langle \_ : x,\ T(PE_V)\rangle,\ \langle \_ : x,\ T(R_V)\rangle,\ \langle \_ : x,\ \text{owl:restriction}\rangle\}$, where $\_ : x$ is the blank node for the restriction and $T(\cdot)$ is the transformation mentioned in Section II. $\lambda_V : E_V \rightarrow L_V$ is the edge-labeling function defined below, where $L_V = \{$owl:onProperty, owl:onProperties, owl:allValuesFrom, owl:someValuesFrom, owl:hasValue, rdf:type$\}$:

$$\lambda_V = \begin{cases} \langle \_ : x, T(PE_V)\rangle & \rightarrow \text{ owl:OnProperty} \\ \langle \_ : x, R_V\rangle & \rightarrow \bar{T}(RE_V) \\ \langle \_ : x, \text{owl:restriction}\rangle & \rightarrow \text{ rdf:type} \end{cases}$$

Here, $\bar{T}(RE_V)$ maps the edge to either *owl:allValuesFrom*, *owl:someValuesFrom*, or *owl:hasValue*. We show the derived subgraph for value restrictions, SG_V, in Fig. $1(a)$. Note that the subgraph produced by $T(\cdot)$ for a specific value restriction canonicalizes to SG_V generated by $\mathcal{T}_V(\cdot)$, as we illustrate in Fig. $1(b)$.

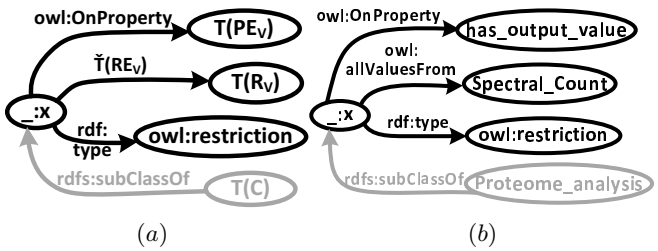


Figure 1. (a) The nodes and edges in bold constitute the canonical form RDF subgraph for value restrictions, while the grayed node is the restricted concept. (b) Canonicalized RDF subgraph for an extract from PEO. The specific value restriction, *owl:allValuesFrom*, on the *proteome_analysis* class restricts *has_output_value* property to take values from *Spectral_Count* only.

A reverse transformation function, $\mathcal{T}^{-1}(SG_V) = CSG_V$ is also defined which transforms a canonical form subgraph, SG_V, to a value restriction in the axiomatic structural specification, CSG_V. It applies $T^{-1}(.)$ as mentioned in Section II to each RDF triple in SG_V. Note that $T^{-1}(T(\cdot))$ produces an ontology that is logically equivalent to its input. The following theorem shows that the subgraph as

derived above may be used to represent any value restriction complex concept without loss in meaning.

Theorem 1 (Canonical value restriction subgraph): For any OWL 2 DL value restriction, CE_V, if SG_V is the canonical form subgraph obtained by transforming CE_V using $\mathcal{T}$, and CSG_V is the OWL 2 DL restriction in canonical form obtained by applying the reverse transformation, $\mathcal{T}^{-1}$, to SG_V, then CE_V and CSG_V are logically equivalent for any value restriction.

Proof: For each type of value restriction in CE_V, the RDF subgraph obtained by applying $T(\cdot)$ to the specific value restriction canonicalizes to SG_V. Furthermore, CSG_V is the canonical form of the OWL 2 ontology obtained by applying the reverse mapping, $T^{-1}(\cdot)$, to the RDF subgraph. The theorem holds because $T^{-1}(T(O))$ is equivalent in meaning to OWL 2 ontology, O, for any O including any value restriction, as we mention in Section II. ■

B. Canonical Form for Cardinality Restrictions

Cardinality restrictions declare the minimum, maximum and exact cardinality of a property range. A cardinality restriction expression is defined using a property and a cardinality value, n. For example, the cardinality restriction, *ObjectMaxCardinality(2, has_output_value, Standard_Deviation)*, defined in PEO for *Proteome_analysis* restricts the cardinality of property, *has_output_value*, to a maximum of 2 individuals of the *Standard_Deviation* class.

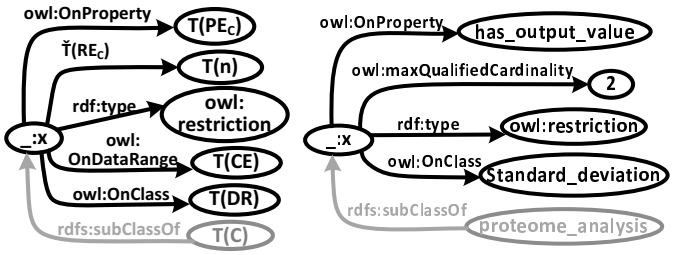
Analogously to value restrictions, different types of cardinality restrictions admit structural specifications, which may be represented equivalently. Using the cardinality value n and property expression, $PE_C \in \{OPE, DPE\}$, we introduce the generalized cardinality restriction complex concept in a canonical form, $CE_C = RE_C(n, PE_C, R_C)$, where RE_C is one of the cardinality restriction expressions in {*ObjectMinCardinality, ObjectMaxCardinality, ObjectExactCardinality, DataMinCardinality, DataMaxCardinality, DataExactCardinality*}. $R_C \in \{CE, DR\}$ is the specific class or datarange whose cardinality is restricted. R_C could be empty unless the cardinality is qualified.

We expand the general transformation function, $\mathcal{T}(CE_C) = SG_C$, to translate a canonical form cardinality restriction, CE_C, into a RDF subgraph in a canonical form, $SG_C = \langle V_C, E_C, \lambda_C \rangle$. Here, vertices $V_C = \{$n^^xsd:nonNegativeInteger, $T(PE_C), T(R_C)$, owl:restriction, $\_ : x\}$ and directed edges $E_C = \{\langle \_ : x,$ n$\rangle$, $\langle \_ : x,$ T(PE_C)$\rangle$, $\langle \_ : x,$ owl:restriction$\rangle$, $\langle \_ : x,$ $T(CE)\rangle\}$. Function, λ_C: $E_C \rightarrow L_C$ gives the edge labels, where $L_C = \{$owl:onProperty, owl:onClass, owl:onDataRange, owl:minCardinality, owl:maxCardinality, owl:cardinality, rdf:type$\}$, and is defined as:

$$\lambda_C = \begin{cases} \langle \_ : x, T(PE_C) \rangle & \rightarrow \text{owl:OnProperty} \\ \langle \_ : x, \text{T(CE)} \rangle & \rightarrow \text{owl:OnClass} \\ \langle \_ : x, \text{T(DR)} \rangle & \rightarrow \text{owl:onDataRange} \\ \langle \_ : x, \text{n^^xsd:nonNegativeInteger} \rangle & \rightarrow \bar{T}(RE_C) \\ \langle \_ : x, \text{owl:restriction} \rangle & \rightarrow \text{rdf:type} \end{cases}$$

Here, $\bar{T}(RE_C)$ maps RE_C to one of three corresponding restriction types: *owl:minCardinality*, *owl:maxCardinality*, *owl:cardinality*.

Subsequently, the reverse transformation function, $\mathcal{T}^{-1}(SG_C) = CE_C$ may also be defined by applying $T^{-1}(.)$ to each RDF triple in SG_C. The following theorem establishes that the transformation, $\mathcal{T}(\cdot)$, produces a general RDF subgraph, SG_C, that is a canonical form for cardinality restrictions with no loss in meaning.



(a) (b)

Figure 2. (a) Canonical RDF graph representation of cardinality restrictions. The nodes and edges in bold constitute the canonical form subgraph for a cardinality restriction. The restricted concept is grayed. (b) An example cardinality restriction obtained from PEO in the canonical form. A cardinality restriction on the *proteome_analysis* class restricts the *has_output_value* property to have a cardinality of 2 on the *Standard_deviation* class.

Theorem 2 (Canonical cardinality restriction subgraph): For any OWL 2 DL cardinality restriction CE_C, let $SG_C = \mathcal{T}(CE_C)$ be the canonical form subgraph obtained by transforming CE_C as specified previously, and let CSG_C be the OWL 2 DL restriction obtained by applying the reverse transformation $\mathcal{T}^{-1}$ to SG_C; then, CE_C and CSG_C are logically equivalent for any cardinality restriction.

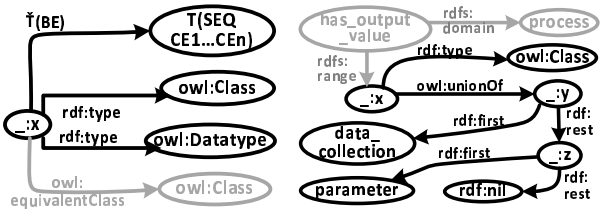
Proof: For each type of cardinality restriction in CE_C, the RDF subgraph obtained by applying $T(\cdot)$ to the specific cardinality restriction is identical to SG_C. Furthermore, the OWL 2 ontology, CSG_C, is the canonical form of the OWL 2 ontology obtained by applying the reverse mapping, $T^{-1}(\cdot)$, to the RDF subgraph. The theorem holds because $T^{-1}(T(O))$ is equivalent in meaning to OWL 2 ontology, O, for any O including any cardinality restriction. ■

We illustrate cardinality restrictions represented using the canonical RDF graph, and the previous example cardinality restriction canonicalized, in Fig. 2. The canonical form subgraph in Fig. 2(a) is not parsimonious. During canonicalization, either the edge $\langle \_ : x,$ T(CE)$\rangle$ or $\langle \_ : x,$ T(DR)$\rangle$ is retained while the other is absent based on whether the property is an object or data property, respectively.

C. Canonical Form for Boolean Combinations

Complex concepts that are Boolean combinations are primarily defined using one of the set operators: union, intersection or complement. Union and intersection are applied to

a sequence of classes or datatypes while complement is applied on a single class or datatype. Structural specifications of these complex concepts may be represented identically in a canonical form as, $CE_B = BE(B)$. Here, the operand, $B \in \{(CE_1, \ldots, CE_k), CE, (DR_1, \ldots, DR_k), DR\}$, and the Boolean operator expression is denoted by, $BE \in$ {ObjectUnionOf, ObjectIntersectionOf, ObjectComplementOf, DataUnionOf, DataIntersectionOf, DataComplementOf}. An example Boolean combination from PEO defines the range of the object property *has_output_value* as a union of *data_collection* and *parameter*. Its structural specification in the canonical form is, *ObjectUnionOf(data_collection, parameter)*.



(a) (b)

Figure 3. (a) The nodes and edges in bold constitute the canonical form subgraph for a Boolean combination. (b) RDF graph representation of an example Boolean combination from PEO in its canonical form. Property *has_output_value* has a Boolean combination as its range, which is a *owl:unionOf* classes, *data_collection* and *parameter*.

Analogously to our previous approach, we expand the generalized transformation function, $\mathcal{T}$, to Boolean combinations, which when applied to CE_B yields a RDF subgraph SG_B that gives the canonical form, $\mathcal{T}(CE_B) = SG_B$. Graph, $SG_B = (V_B, E_B, \lambda_B)$ where the set of vertices is, $V_B = \{T(B),$ owl:class, rdfs:Datatype, _:x$\}$. $E_B = \{$ $\langle$_ : x, owl:class$\rangle$, $\langle$_ : x, rdfs:Datatype$\rangle$, $\langle$_ : x, T(B)$\rangle\}$ is a set of edges, and $\lambda_B : E_B \rightarrow L_B$, where L_B = {owl:unionOf, owl:intersectionOf, owl:complementOf, rdf:type}. λ_B labels the edge $\langle$_ : x, T(B)$\rangle$ with $\bar{T}(BE)$, which maps to *owl:unionOf*, *owl:intersectionOf* or *owl:complementOf*. Edges $\langle$_ : x, owl:class$\rangle$ and $\langle$_ : x, rdfs:Datatype$\rangle$ are both labeled with *rdf:type*. The corresponding canonical form subgraph for Boolean combinations is shown in Fig. $3(a)$. We also define a reverse transformation, $\mathcal{T}^{-1}(SG_B) = CE_B$, which transforms any canonical form Boolean combination subgraph back to a structural specification by applying the transformation, $T^{-1}(\cdot)$, to each RDF triple in the graph. The following theorem holds for complex concepts involving Boolean combinations as well.

Theorem 3 (Canonical Boolean subgraph): For any OWL 2 DL Boolean combination, CE_B, let $SG_B = \bar{T}(CE_B)$ be the canonical form subgraph and let CSG_B be the OWL Boolean combination obtained by using the reverse transformation, $\mathcal{T}^{-1}$, to SG_B; then, CE_B, CSG_B are logically equivalent for any combination.

Proof: For each type of Boolean combination in CE_B, the RDF subgraph obtained by applying $T(\cdot)$ to the specific Boolean combination canonicalizes to SG_B. Additionally, CSG_B gives the canonical form of the OWL 2 ontology obtained by applying the reverse mapping, $T^{-1}(\cdot)$, to the RDF subgraph. Because $T^{-1}(T(O))$ is equivalent in meaning to OWL 2 ontology, O, for any O including any Boolean combination, the theorem holds. ■

Note that the canonical form subgraph in Fig. $3(a)$ is not parsimonious. During canonicalization, either the edge $\langle$_ : x, owl:class$\rangle$ or $\langle$_ : x, rdfs:Datatype$\rangle$ is retained based on whether the specific concept is a Boolean combination of classes or data types. Well developed ontologies such as PEO have several Boolean combinations, as in Fig. $3(b)$.

IV. Similarity between Complex Concepts

The first step toward matching complex concepts present in an ontology pair is to identify them in each ontology, followed by canonicalizing them to the appropriate axiomatic or graph forms based on the concept type. We adopt caution in comparing the complex concepts for alignment. Specifically, due to the differing semantics of their interpretations, we do not match a value or cardinality restriction with a Boolean combination. This leads to a limitation of our approach: some concepts may admit descriptions using both restrictions and Boolean combinations, which may not be matched. Furthermore, we draw a strict distinction between cardinality and value restrictions by noting that their semantics are often complementary. Therefore, we do not seek a match between these different types of restrictions.

Let $\mathcal{CE}_{\mathcal{CC}}$ denote the set of all types of complex concepts, and Sim denote the similarity function between two complex concepts, $Sim : \mathcal{CE}_{\mathcal{CC}} \times \mathcal{CE}_{\mathcal{CC}} \rightarrow \mathbb{R}$. Then,

$$Sim(CE^1_{CC}, CE^2_{CC}) = \begin{cases} Sim_R(CE^1_{CC}, CE^2_{CC}) & CE^1_{CC}, CE^2_{CC} \in \{CE_V, CE_C\} \\ Sim_B(CE^1_{CC}, CE^2_{CC}) & CE^1_{CC}, CE^2_{CC} \in \{CE_B\} \\ -1 & \text{otherwise} \end{cases} \quad (1)$$

where Sim_R and Sim_B are the similarity functions that operate on restriction and Boolean complex concepts. Notice that we return a value of -1 instead of 0, which signifies that a match between the two concepts has not been attempted.

Similarity between property restrictions in their canonical form subgraphs is an aggregation of the similarities between their corresponding transformed property expressions, $T(PE^1)$ and $T(PE^2)$, corresponding transformed class expressions or dataranges, $T(R^1)$ and $T(R^2)$, and literals, n^1 and n^2. If one of the canonical representations has a nonempty set of literals while the other is empty indicating that the latter is a value restriction while the former is a restriction on cardinality, no similarity is computed.

$$Sim_R(CE^1_{V|C}, CE^2_{V|C}) = \begin{cases} w \cdot H\text{mean}\left(Sim'(T(PE^1_{V|C}), T(PE^2_{V|C})\right), & n^1, n^2 \neq \{\} \\ Sim'(T(R^1_{V|C}), T(R^2_{V|C})), Sim'(n^1, n^2)) & n^1, n^2 = \{\} \\ -1 & \text{otherwise} \end{cases} \quad (2)$$

Here, Sim' measures the similarity between the expressions. If the expressions are complex concepts themselves, this becomes a *recursive* call to the Sim function defined in Eq. 1, otherwise the lexical similarity is evaluated.

We utilize the weight, w, to emphasize the similarity in the types of value and cardinality restrictions. For example, the weight w between the same cardinality types could be 1, between a *owl:minCardinality* and a *owl:maxCardinality*, 0, and between the remaining cardinality type combinations, 0.75. Instead of taking a simple average, a modified harmonic mean, H_{mean}, is utilized. This mitigates the influence of extreme outliers in the similarity values and tends toward the lower values Sim' in the list.

In the context of Boolean combinations, we match complex concepts representing the same Boolean operators. Because property expressions and literals are not present in Boolean canonical subgraphs, Eq. 2 reduces to:

$$Sim_B(CE_B^1, CE_B^2) = w \cdot Sim'(\bar{T}(BE^1), \bar{T}(BE^2)) \quad (3)$$

Here, w becomes 0 if $\bar{T}(BE^1)$ and $\bar{T}(BE^2)$ are not the same indicating that the two canonicalized subgraphs contain different operators.

If the ontologies are modeled axiomatically with the complex concepts canonicalizing to structural specifications, Eqs. 2 and 3 measure the similarity between the participating expressions directly instead of their graph transformations.

V. Experiments

We integrate our approach for computing the similarity between the complex concepts described in the previous section within the representative alignment algorithms, Falcon-AO, LogMap and Optima+. Because many of the alignment algorithms precluded complex concepts while loading the ontologies, our first step is to update the ontology models of the algorithms to include complex concepts. We aim to integrate the similarity of complex concepts within the algorithms as seamlessly as possible. This allows the different alignment algorithms to treat the complex concepts analogously to the named concepts, thereby requiring minimal changes in the algorithms themselves.

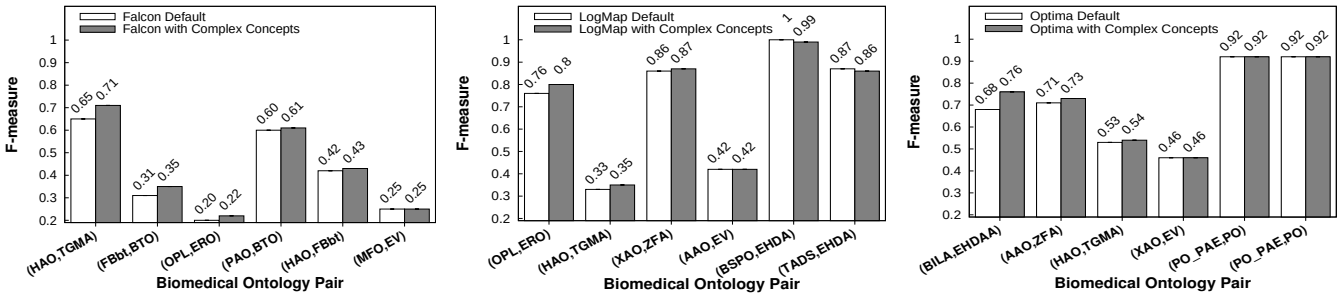
We analyze the improvements in precision and recall along with the associated trade off in the runtime by modeling complex concepts in various alignment algorithms. For evaluation, we use a comprehensive testbed of several ontology pairs spanning multiple domains. One of the testbed comprises of 25 pairs of ontologies from the 2012 edition of OAEI. We use 4 ontology pairs from its Benchmark track and 21 ontology pairs from its Conference track. The selection of these tracks is based on the fact that they include real-world ontologies for which the reference alignments are also provided by OAEI. This includes all ontology pairs in the 300 range of the Benchmark track, which relate to *bibliography*, and expressive ontologies in the *conference* domain all of which structure knowledge about conference organization. We created another novel testbed for evaluation using biomedical ontologies from the National Center for Biomedical Ontologies (NCBO). This testbed contains 35 ontology pairs organizing knowledge in various biomedical domains. The ontologies were selected based on having 10% percent or more of complex concepts and a good amount of reference correspondences available in NCBO (10% or more of each ontology's concepts are present in the reference). The biomedical testbed is available for use at http://tinyurl.com/aulcezm.

On modeling complex concepts, there is no change in the overall precision and recall for Falcon-AO across all the pairs in the *bibliography* and *conference* domains (precision=62%, recall=60%). For Optima+, we obtained an overall 1% improvement in the precision increasing it to 54% and 1% improvement in recall thereby making it 70%. LogMap's overall precision improved by 1% to 59% but modeling complex concepts did not affect its recall of 80%.

Increase in **runtime** caused by modeling complex concepts is minimal for each algorithm for these tracks. This is due to the scarcity of compatible complex concepts in the involved ontologies. LogMap and Falcon-AO consumed 1 and 8 and seconds more, respectively, than the default across all 25 pairs in the bibliography and conference domains. Optima+, which is slowest among the three, consumed 52 seconds in addition to the default.

For the large ontology pairs in our novel biomedical testbed, all the three algorithms benefit from modeling the complex concepts. The enhanced Falcon-AO identified a total of 88 less false positive correspondences thus improving precision significantly as shown in Fig. 4(a). It found 2 additional correct correspondences resulting in a small increase in overall recall of 0.12%. However, it also identified 4 more false positives. The overall improvement in F-measure for Falcon-AO to 31% (precision=48%, recall=23%) is significant (Student's paired t-test, $p < 0.05$). The enhanced LogMap pruned 81 false positives in addition to finding 3 more correct correspondences (see Fig. 4(b)). This increased its precision to 62% and recall to 35%, both of which increases are significant ($p < 0.05$). In particular, modeling complex concepts improved the F-measure of aligning the *OPL, ERO* pair by 4%. Note that LogMap is designed for aligning large biomedical ontologies. The enhanced Optima+ identified a total of 74 less false positives in addition to finding 9 more correct correspondences. Optima+ generates more useful samples by modeling complex concepts, which improves the recall noticeably for some of the pairs; for example, the *BILA, EHDA* pair gained recall by 6%. The overall F-measure improved by 4%. This improvement in F-measure to 56% (precision=55%,recall=56%) on the biomedical testbed is significant ($p < 0.01$).

Falcon-AO's total run time of 20 minutes for aligning the entire biomedical testbed increases threefold to 57 minutes due to including complex concepts. Optima+ with complex



(a) Falcon-AO (b) LogMap (b) Optima+

Figure 4. Performance on the biomedical testbed. (a) Falcon-AO, (b) LogMap, and (c) Optima+ with complex concept modeling exhibits significantly improved precision (Student's paired t-test, $p < 0.05$) resulting in improved F-measure. We ran the algorithms on all 35 pairs of which we show the 3 ontology pairs that exhibit the highest and lowest differences in F-measure. Ontology names are NCBO abbreviations.

concepts took 25.6 hours compared to 18 hours consumed by the default resulting in a 1.5 times increase. LogMap with complex concepts modeled took 1.8 hours compared to 0.5 hour by the default, resulting in more than a threefold increase. The overhead associated with modeling and computing the similarity of complex concepts is evident in all three algorithms on the large biomedical testbed, which contain a large number of complex concepts. Consideration of the complex concepts affects the structural matching. The time complexity of the structural matchers of these algorithms is exponential in the size of the input ontologies, which exacerbated the total execution time. LogMap displays a particularly high increase in execution time because modeling complex concepts significantly increases the number of candidate correspondences that are considered. Falcon-AO takes more iterations to converge when complex concepts are modeled. As Optima+ performs a limited amount of structural matching focusing on the class hierarchy only, its increase in execution time is relatively less compared to other algorithms.

VI. Conclusion

We observed that different types of value restrictions could be modeled uniformly thereby allowing an axiomatic canonical form in OWL 2's structural specification and derived an equivalent RDF graph-based canonical form. Analogously, canonical forms were provided for different cardinality restrictions and the various Boolean combinations. This allowed us to improve ontology alignment by canonicalizing the complex concepts and providing a simple way to measure similarity between the anonymous classes.

Ideally, we seek to match composite complex concepts of different types such as one involving a value restriction and another containing a Boolean combination if they are semantically equivalent. Therefore, a single canonical representation that would identify the same concept despite being differently defined is preferred. This is challenging requiring robust DL inferencing. In this paper, we provide separate canonical representations for three types of complex concepts, which is a first step toward this goal. In our knowledge, this paper is the first in its explicit focus on modeling complex concepts for the alignment process.

Acknowledgement

This research is supported in part by grant number R01HL087795 from the NHLBI. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NHLBI and NIH.

References

[1] F. Giunchiglia, P. Shvaiko, M. Yatskevich, F. Giunchiglia, P. Shvaiko, and M. Yatskevich, "S-match: an algorithm and an implementation of semantic matching," in *European Semantic Web Symposium*, 2004, pp. 61–75.

[2] E. Jiménez-Ruiz, B. C. Grau, and Y. Zhou, "Logmap 2.0: towards logic-based, scalable and interactive ontology matching," in *4th International Workshop on Semantic Web Applications and Tools for the Life Sciences*, 2012, pp. 45–46.

[3] Y. R. Jean-Mary, E. P. Shironoshita, and M. R. Kabuka, "Ontology matching with semantic verification," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 3, pp. 235–251, 2009.

[4] Y. Li, J. Li, and J. Tang, "RiMOM: Ontology alignment with strategy selection," in *6th International and 2nd Asian Semantic Web Conference*, 2007, pp. 51–52.

[5] D. Ngo and Z. Bellahsene, "YAM++ : A Multi-strategy Based Approach for Ontology Matching Task," in *International Conference on Knowledge Engineering and Knowledge Management*, 2012, pp. 421–425.

[6] P. Doshi, R. Kolli, and C. Thomas, "Inexact matching of ontology graphs using expectation-maximization," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 2, pp. 90–106, 2009.

[7] W. Hu and Y. Qu, "Falcon-ao: A practical ontology matching system," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, pp. 237–239, 2008.

[8] P. F. Patel-Schneider and B. Motik, "Owl 2 web ontology language mapping to rdf graphs," http://www.w3.org/2007/OWL/wiki/Mapping_to_RDF_Graphs, w3C OWL wiki.