The Properties of Property Alignment on the Semantic Web

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

By

MICHELLE CHEATHAM
B.S., University of Kentucky, 2001
M.B.A., University of Kentucky, 2001
M.S., Wright State University, 2005

2014
Wright State University
Dayton, Ohio 45435-0001

WRIGHT STATE UNIVERSITY
SCHOOL OF GRADUATE STUDIES

July 15, 2014

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Michelle Cheatham ENTITLED The Properties of Property Alignment on the Semantic Web BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy in Computer Science and Engineering.

Pascal Hitzler, Ph.D.
Dissertation Director

Arthur Goshtasby, Ph.D.
Director, Computer Science and Engineering Ph.D. Program

Robert E.W. Fyffe, Ph.D.
Vice President for Research and Dean of the Graduate School

Committee on
Final Examination

Pascal Hitzler, Ph.D.

Isabel Cruz, Ph.D.

Krishnaprasad Thirunarayan, Ph.D.

Mateen Rizki, Ph.D.

ABSTRACT

Cheatham, Michelle. PhD Department of Computer Science and Engineering, Wright State University, 2014. The Properties of Property Alignment on the Semantic Web.

Ontology alignment is an important step in enabling computers to query and reason across the many linked datasets on the semantic web. This is a difficult challenge because the ontologies underlying different linked datasets can vary in terms of subject area coverage, level of abstraction, ontology modeling philosophy, and even language. The alignment approach presented here centers on string similarity metrics. Nearly all ontology alignment systems use a string similarity metric in one form or another, but it seems that the choice of a particular metric is often arbitrary. We begin this dissertation with the most comprehensive survey to date on the performance of string similarity metrics and string preprocessing strategies for ontology alignment. Based on this work we present practical guidelines for choosing string metrics in the face of different types of ontologies and different alignment goals. Additionally, we show that string similarity metrics alone can perform competitively with state-of-the-art alignment systems on the most popular benchmarks in the field.

One of the contributions of our string similarity metric survey is quantification of the difference in performance between aligning classes and aligning properties (relations between classes). Put simply: aligning properties is hard, and existing string similarity metrics are not of great help. We therefore take on the task of developing a new string-based alignment approach that performs better on properties. Unfortunately, evaluating that approach is difficult because the only existing alignment benchmark that includes properties is, in our view, unrealistic since all relations in the reference alignment are presented as completely certain. Human experts do not have this degree of confidence when asked to align an ontology. We therefore present a more nuanced version of this benchmark that we have created through a combination of expert survey and crowdsourcing. We then present our new string-based property alignment system and evaluate its performance on both the current benchmark and our proposed revision. Our property-centric string metric can be configured for either high precision or high recall. The results show a five-fold increase in precision and a doubling of recall over an approach based on the best current string metric. Finally, we apply our system to a real-world test case and analyze the results.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I am very fortunate to have had several "fans" who have supported me throughout my life. First among these are my parents, Curt and Lynn Andreen, and my husband Jason. They are very special people. Another is Mr. Rob Ehret, who hired me, taught me, and then showed me the door when it was time to move on. I would also like to thank my advisor, Dr. Pascal Hitzler, and my dissertation committee for their guidance and support.

This work was supported by the National Science Foundation under award 1017225 "III: Small: TROn—Tractable Reasoning with Ontologies."

1

Introduction

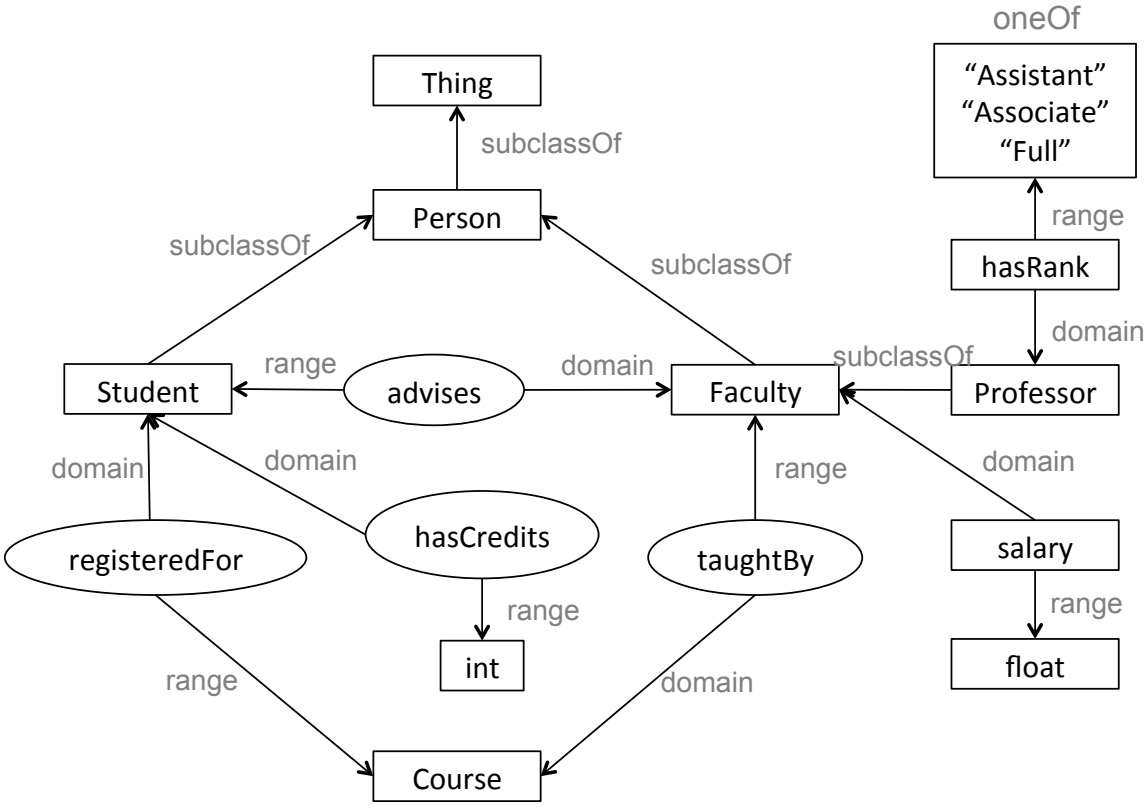
Tim Berners-Lee originally envisioned a much different world wide web than the one we have today – a Semantic Web that computers as well as humans could search for the information they need [Berners-Lee et al. 2000]. Due to an increasing number of individuals and organization publishing their data as linked data, things are moving in that direction. Linked data builds upon existing web standards such as HTTP, RDF, and URIs to create web pages that are machine-readable and, ideally, machine-understandable. According to Berners-Lee[1], the four rules of linked data are:

1. Use URIs to denote things.
2. Use HTTP URIs so that these things can be referred to and looked up ("dereferenced") by people and user agents.
3. Provide useful information about the thing when its URI is dereferenced, leveraging standards such as RDF, SPARQL.
4. Include links to other related things (using their URIs) when publishing data on the Web.

The last of these rules is particularly critical in order to fully leverage the information published on the Semantic Web. Links between related things, particularly related things from different datasets, are what enables applications to move beyond individual silos of data towards synthesizing information from a variety of data sources. Unfortunately, data publishers often don't explicitly specify links between their datasets and others. The field of ontology alignment attempts to discover these links in an automatic or semi-automatic way. Developing efficient and effective alignment systems is therefore an important step towards realizing the promise of the Semantic Web.

[1] http://www.w3.org/DesignIssues/LinkedData.html

Figure 1.1: A ontology snippet describing a university from a teaching perspective.

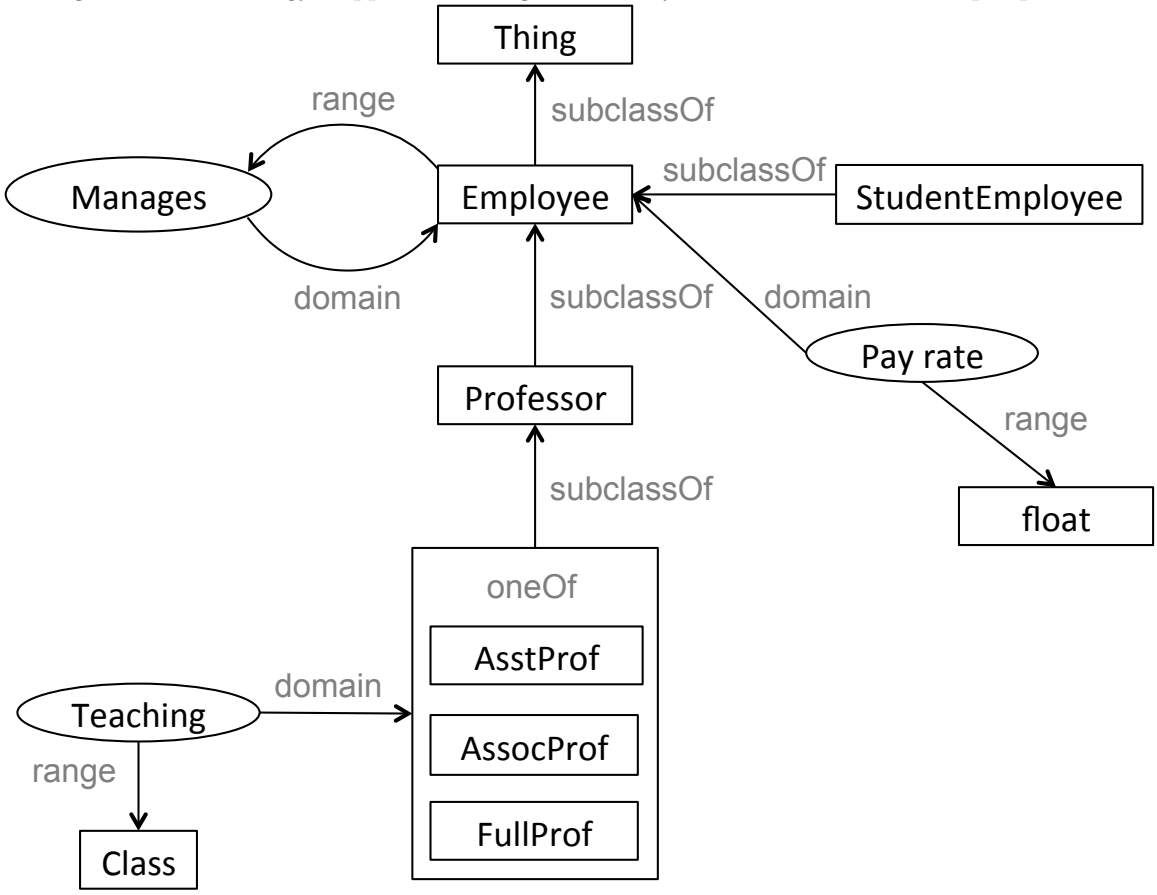


1.1 Fundamental Vocabulary

Linked data seamlessly combines raw data with schema information that describes the meaning of the data, in the form of an ontology. Tom Gruber, one of the early voices on knowledge representation (and the creator of Siri), defines an ontology as a "specification of a conceptualization." He elaborates that an ontology defines the concepts and relationships within a domain [Gruber 1993]. We briefly introduce the principle elements of ontologies here. Our goal is to present the minimum vocabulary needed to digest the work in this document in an approachable manner. A more formal and extensive treatment of these topics can be found in [Hitzler et al. 2011].

Figure 1.1 shows a portion of an ontology that describes a university. This example includes concepts such as Person, Course, and Student. These are called classes. A class represents a grouping of objects with similar characteristics. Classes are often arranged in a hierarchy using subclass relationships. For instance, in our example Faculty is a subclass of Person. An instance (or individual, we will use these terms interchangeably) is a particular object. An instance has a type that is some class within the ontology. For example, an instance of type Course may be MA 123

Figure 1.2: A ontology snippet describing a university from a human resources perspective.



and an instance of type Faculty may be Dr. John Doe. This is somewhat analogous to classes and instances of those classes in object-oriented programming languages, such as Java. Relationships between instances, such as registeredFor and hasCredits, are called properties. All properties are directed binary relations that map an instance with a type from the domain to something in the range. Properties that map an instance to another instance (e.g. registeredFor, which maps an instance of type Student to an instance of type Course) are object properties, whereas properties that map an instance to a literal value (e.g. salary, which maps an instance of type Faculty to a float value) are datatype properties. Common data types include integers, doubles, strings, and dateTime. Both object properties and data properties must involve an instance. A third type of property, called an annotation property, can be used to describe relationships between any types of entities (i.e. instances, classes or other properties). All of this information: classes, properties, and any restrictions on them, such as cardinality, disjointness, etc., are called the schema, or T-box (for terminology), of the ontology. Conversely, the instance data, or A-box (for assertions), contains

assertions about individuals using data from the T-box. Of the statements below, the first three are from the T-box and the last one is from the A-box. These statements are written in the Web Ontology Language (OWL)[2].

```
<owl:Class rdf:about="#Professor">
  <rdfs:subClassOf rdf:resource="#Faculty"/>
</owl:Class>

<owl:ObjectProperty rdf:about="#advises">
  <rdfs:range rdf:resource="#Student"/>
  <rdfs:domain rdf:resource="#Faculty"/>
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:ID="salary">
  <rdfs:domain rdf:resource="#Faculty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#double"/>
</owl:DatatypeProperty>

<owl:Thing rdf:ID="Dr\_John\_Doe">
 <rdf:type rdf:resource="#Professor"/>
  <advises rdf:resource="#Johnny\_Smith" />
  <salary rdf:datatype="\&xsd;double">90000</salary>
</owl:Thing>
```

Many of the ontologies available on the web today are heavily skewed towards either T-box or A-box. For instance, there is a project[3] to publish the content of the MusicBrainz online music catalog[4] as linked data using the Music Ontology[5]. The Music Ontology contains 54 classes, 153 properties, and essentially no logical axioms describing relations between these, other than domain, range, or subsumption. On the other hand, there are more than 75,000 instances represented. This combination of lightweight ontology combined with reams of instance data is typical of many datasets available on the Semantic Web. On the other hand, there are some ontologies that contain very complex models of a domain but little or no instance data. The biomedical community has

[2]http://www.w3.org/TR/owl2-overview/
[3]https://wiki.musicbrainz.org/LinkedBrainz
[4]https://musicbrainz.org
[5]http://musicontology.com

been at the forefront of developing such ontologies, with efforts like SNOMED CT[6].

1.2 Ontology Alignment

Engineering new ontologies is not a deterministic process – many design decisions must be made, and the designers' backgrounds and the application they are targeting will influence their decisions in different ways. The end result is that even two ontologies that represent the same domain will not be the same. They may use synonyms for the same concept or the same word for different concepts, they may be at different levels of abstraction, they may not include all of the same concepts, and they may not even be in the same language. And this is in the best case. In real-world datasets there are often problems with missing information, inconsistent use of the T-box when describing individuals, and logically inconsistent axioms. The goal of ontology alignment is to determine when an entity in one ontology is semantically related to an entity in another ontology (for a comprehensive discussion of ontology alignment, including a formal definition, see [Euzenat and Shvaiko 2007]).

An alignment algorithm takes as input two ontologies and produces a set of matches consisting of a URI specifying one entity from each ontology, a relationship (typically equality or subsumption), and an optional confidence value that is generally in the range of 0 to 1, inclusive. For example, an alignment system given the ontologies in Figures 1.1 and 1.2 might produce matches including:

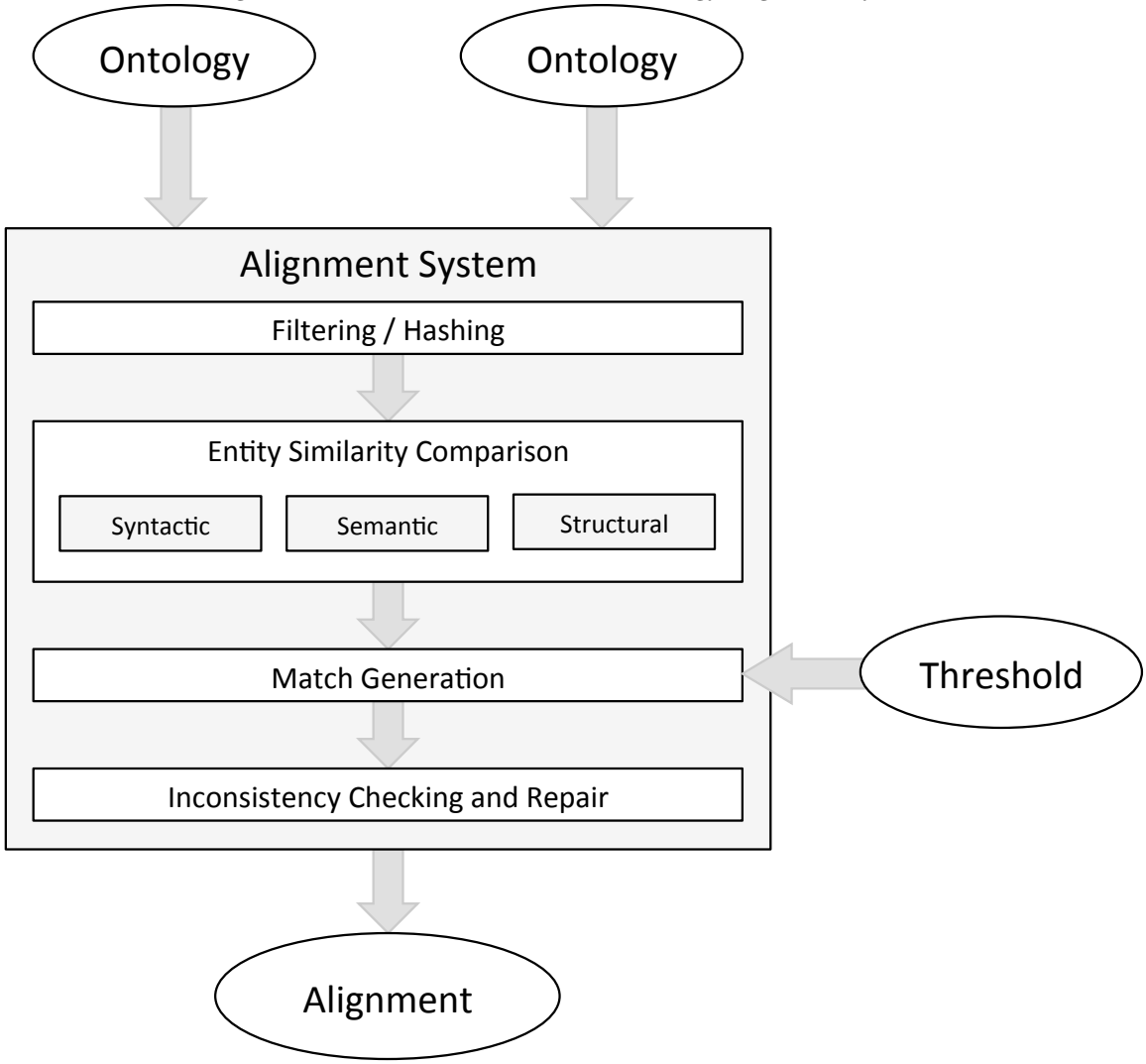
```
ont1:Course, ont2:Class, =, 0.9
ont1:Faculty, ont2:Employee, <, 0.6
```

The term ontology alignment can refer to aligning the entire ontology or just the entities in the T-box; aligning the instance data is generally called co-reference resolution. Though not necessary, in practice alignments are often interpreted under the closed world assumption, in the sense that any entity pairs not mentioned in an alignment are assumed to have no relationship.

Many alignment systems share a common general organization, shown in Figure 1.3. Because ontologies can contain millions of entities, it is often infeasible to compare every entity in one ontology to every entity in the other. Therefore, alignment systems sometimes employ a filtering or hashing step to determine which entities to compare [Duan et al. 2012; Hartung et al. 2013]. Alignment systems typically use a combination of three different approaches to evaluate entity similarity: syntactic, semantic, and structural similarity metrics. Syntactic metrics compare entities from each of the ontologies to be aligned based on strings associated with the entities. The strings are generally the entity label, but can also include comments or other annotations of the entity.

[6] http://www.ihtsdo.org/snomed-ct/

Figure 1.3: General structure of an ontology alignment system.



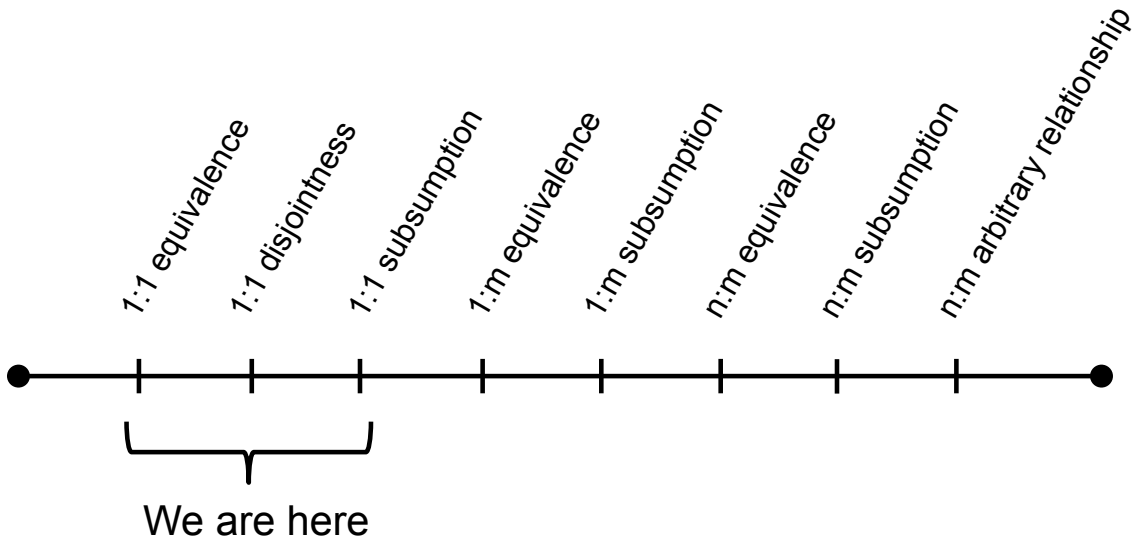
This type of metric is discussed in detail in Chapter 2. Semantic similarity metrics attempt to use the meanings of entity labels rather than their spellings. External resources such as thesauri, dictionaries, encyclopedias, and web search engines are often used to calculate semantic similarity [Jain et al. 2010; Taylor et al. 2005]. Structural techniques consider the neighborhoods of two entities when determining their similarity. For instance, two entities with the same superclass that share some common instances are considered more similar than entities that do not have these things in common. Graph matching techniques are often used for this [Gallagher 2006; Di Martino 2009]. An alignment system may use zero or more of each type of similarity metric. The values from multiple approaches may be combined to form a single measure of similarity, or they may be used in a serial fashion to filter potential matches down to the most likely candidates. At some point, a final list of

related entities is generated, frequently by including any matches with a confidence (similarity) value higher than some threshold. Additionally, alignment systems may use some form of inconsistency checking and repair after the matching process in order to ensure a merged ontology produced using the alignment is logically consistent [Meilicke 2011; Santos et al. 2013; Pesquita et al. 2013].

Ideally, alignment systems should be able to uncover any entity relationships across two ontologies that can exist within a single ontology. Such relationships have a wide range of complexity, as shown in Figure 1.4. The simplest type of relationship is 1-to-1 equivalence or disjointness of two entities (i.e. all instances of A are also instances of B or an instance of A is definitely not an instance of B). The relation ont1:Course = ont2:Class is an example of this type of match, as is ont1:registeredFor disjoint ont2:Teaching (i.e. someone cannot both register to take a course and teach it). The next complexity level is subsumption relationships, i.e. that an entity in one ontology is a subclass or superclass of an entity in another ontology. ont1:Faculty $\subset$ ont2:Employee is an example of this. Even harder to find are 1-to-many equivalence or subsumption relationships between entities, such as the union of ont2:AsstProf, ont2:AssocProf, and ont2:FullProf is equivalent to ont1:Professor. This causes a complexity problem. To find 1-to-1 relationships, an exhaustive search needs to compare every entity in the first ontology to every entity in the second ontology, which may be feasible for small ontologies. To find 1-to-m relationships an exhaustive approach would need to compare each entity in the first ontology to all possible combinations of m entities in the second ontology, which is not generally possible. Finding arbitrary n-to-m relationships is the most complex alignment task. By "arbitrary," we mean any type of relationship, not restricted to equivalence, disjointness, or subsumption. An example of this might be that a ont1:Professor with an ont1:hasRank value of "Assistant" is equivalent to an ont2:AsstProf. Such complex relationships would need to be expressed as logical rules or axioms.

Nearly all existing alignment systems fall at the simplest end of this scale. A few systems, including BLOOMS [Jain et al. 2010] and PARIS [Suchanek et al. 2011b], are capable of finding subsumption relationships across ontologies. CSR [Spiliopoulos et al. 2010] and TaxoMap [Hamdi et al. 2010] attempt to find 1-to-m equivalence and subsumption relationships. In general though, most research activity in the field of ontology alignment remains focused on finding 1-to-1 equivalence relations between ontologies. Current benchmarks for alignment encourage this to some degree, since they are focused on evaluating performance on finding equivalent pairs of classes and instances. Alignment systems have become quite proficient at this task. Figure 1.5 shows that the top systems on the widely-respected OAEI Conference test set are achieving f-measures of around 0.75. As we will show in Chapter 3, this is nearing the level of consensus that humans familiar with ontology design have for the alignment tasks in this benchmark. We feel that alignment research is in danger

Figure 1.4: Complexity range of entity relationships between ontologies.



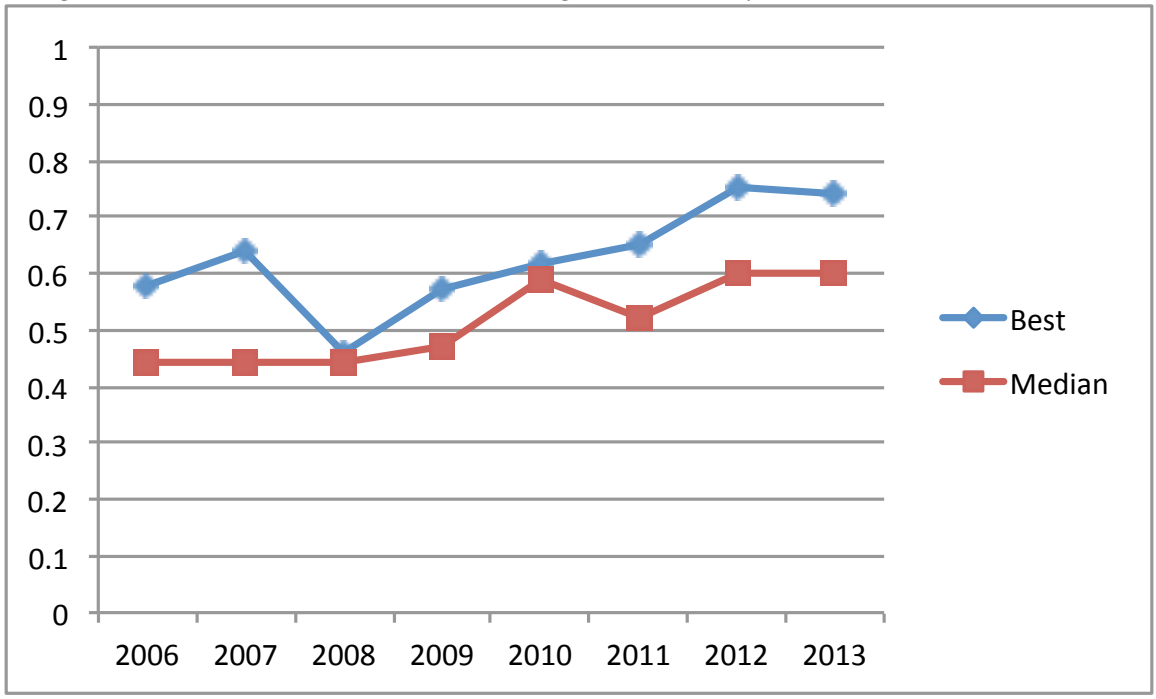
of becoming stuck in a "local maximum", and it may be time to make a concerted push towards discovering more complex semantic relationships. Before we can do that, however, one small problem remains: while the performance on finding equivalent pairs of classes and instances has become quite good, identifying equivalent properties remains challenging for most alignment systems. Exploring this issue and possible approaches to solving it is the focus of this dissertation.

1.3 Improving Performance on Property Alignment

In this work we explore the problem of poor performance of current alignment systems on aligning properties compared to that of aligning classes and instances. We begin by investigating the performance of string similarity metrics as applied to ontology alignment in Chapter 2. We show that, by intelligently choosing string similarity metrics based on the characteristics of the particular ontologies to be aligned, we can actually achieve performance quite competitive with current systems using *only* string similarity metrics. We also show that the performance of such string metrics when aligning classes is approximately four times better than that of aligning properties for a popular alignment benchmark. These results hold even if strategies proposed by other researchers to improve performance on properties, such as word stemming or stopword removal, are employed.

Having established the poor performance of current string similarity metrics for property alignment, we seek to develop a new string-based approach. Ideally, the approach should produce similarity values that correspond reasonably well to the degree of consensus among humans on the

Figure 1.5: Best and median f-measure throughout the history of the OAEI Conference track.



similarity of two properties. Unfortunately, there is only one established benchmark that contains matches between properties: the Conference track of the Ontology Alignment Evaluation Initiative. This benchmark was developed primarily by just three people, and the confidence value of every match is 1.0. We therefore find ourselves without a suitable test case with which to evaluate our method. The work presented in Chapter 3 seeks to remedy this. First, we use a survey of 13 experts together with Amazon's Mechanical Turk platform to crowdsource a more nuanced version of the Conference benchmark, with confidence values indicative of the degree of consensus among the surveyed individuals. Next, we extract a connected subgraph from the DBPedia and YAGO ontologies that contain all of the properties, together with the classes and instances related to them. The result is a pair of real-world ontologies suitable to evaluate property alignment that is of a size within the capabilities of most alignment systems. While we do not have a complete manually-curated reference alignment for this test set, the PARIS alignment system has produced some initial results for the complete versions of these ontologies, and in Chapter 4 we present our results. We use these initial alignments together with crowdsourced input from Mechanical Turk to conduct a preliminary assessment of the quality of results.

In Chapter 4 we further make the case that property alignment is an important topic by illustrating the much higher performance on classes versus properties of the current best-performing alignment systems. An analysis of false positives and false negatives commonly produced by these

systems shows that an improved similarity metric has the potential to eliminate many mistakes. We therefore introduce a new string-based alignment system, PropString, and evaluate its performance against a system based on soft TF-IDF, the best-performing existing string metric as identified in Chapter 2. The results are quite encouraging, with a precision-oriented version of PropString able to achieve perfect precision on the Conference benchmark while maintaining the recall of soft TF-IDF and a recall-oriented configuration that doubles the recall of soft TF-IDF while still achieving 50 percent better precision. Finally, we apply PropString to aligning the properties of two real-world ontologies and compare the results to that of soft TF-IDF and PARIS, a full-featured alignment system.

The main contributions of this work are:

- An extensive survey of the performance of string similarity metrics and string preprocessing strategies (e.g. stopword removal, stemming, translation, etc.) as applied to ontology alignment.
- Guidelines for choosing high-performing string similarity metrics based on easily-computed characteristics of the ontologies to be aligned.
- Quantification of the difference in performance of both string similarity metrics and current state-of-the-art alignment systems on classes versus properties.
- Creation of a more nuanced version of the only generally accepted alignment benchmark that involves property matches, backed by expert opinion and crowdsourcing.
- Creation of a real-world property-centric alignment test set that is scaled to be within the capabilities of most existing alignment systems.
- Introduction of a string-based property alignment approach designed for accurate similarity computation between properties, and evaluation of its performance.

2

String Similarlity

In researching past and present alignment systems, it became obvious that nearly all such systems make use of a string similarity metric. But despite the ubiquity of these metrics, there has been little systematic analysis on which string similarity metrics perform well when applied to ontology alignment. This chapter seeks to fill that gap by analyzing the performance of string similarity metrics in this domain, as well as the utility of string preprocessing approaches such as tokenization, translation, synonym lookup, and others.

This study leads naturally to a follow-up question: how much performance can we squeeze out of string-based techniques? We therefore consider how much an existing alignment algorithm can be improved by incorporating string similarity metrics that are optimized for the particular ontology matching problem at hand.

In particular, we seek to answer the following questions:

- What is the most effective string similarity metric for ontology alignment if the primary concern is precision? recall? f-measure?
- Does the best metric vary based on the nature of the ontologies being aligned?
- Does the performance of the metrics vary between classes and properties?
- Do string preprocessing strategies such as tokenization, synonym lookup, translations, or normalization improve ontology alignment results?
- What is the best we can do on the ontology alignment task using only string preprocessing and string similarity metrics?
- When faced with the task of aligning two ontologies, how can we automatically select which string similarity metrics and preprocessing strategies are best, without any training data available?

There has been some prior analysis of string similarity metrics in the context of ontology alignment as part of the development of a new string similarity metric designed specifically for this domain done by Stoilos and his colleagues [Stoilos et al. 2005]. They compared the performance of their own metric to that of Levenstein, Needleman-Wunsch (a weighted version of Levenstein), Smith-Waterman, Monge Elkan, Jaro Winkler, 3-gram, and substring on a subset of the OAEI Benchmark test set. The Benchmark test set is an older OAEI track that was phased out in 2010 in favor of a dynamically generated test set. Stoilos and his colleagues found that the Monge Elkan and Smith Waterman metrics performed very poorly on this task. The metric developed by the researchers performed the best. Another piece of work done in this area is a report produced by the Knowledge Web Consortium in 2004 that described of a variety string (terminological) metrics applied to the problem of ontology alignment [Euzenat et al. 2004]. This document also discussed string preprocessing strategies such as normalization and stemming.

When the area of interest is expanded to include string similarity metric studies for other domains, we find some additional interesting surveys. For instance, Branting looked at string similarity metrics as applied to the names of people, businesses, and organizations, particularly in legal cases [Branting 2003]. Nine categories of name variations were identified: punctuation, capitalization, spacing, qualifiers, organizational terms, abbreviations, misspellings, word omissions, and word permutations. His work evaluated the performance of various combinations of normalization, indexing (determining which names would be compared to one another) and similarity metrics. He found that string normalization was useful for this application and that a string similarity metric that he called RWSA (described below) resulted in the best performance. In addition, Cohen, Ravikumar, and Fienberg did a very thorough analysis of string similarity metrics as applied to name-matching tasks [Cohen et al. 2003]. They found that TF-IDF, Monge Elkan, and Soft TF-IDF performed well on the datasets they analyzed. In addition, they developed the SecondString Java library of string similarity metrics, which has become very widely used in the research community (including in our work here).

Some researchers have not set out to study string similarity metrics but have learned some interesting things about the topic while developing ontology alignment systems. For instance, the developers of Onto-Mapology tried Jaro, Jaro-Winkler, TF-IDF, and Monge Elkan in their alignment system and found Jaro-Winkler to have the highest performance [Bethea et al. 2006], and the developers of SAMBO, which focuses on biomedical ontologies, found that a weighted sum of n-gram, edit distance, and an unnamed set metric performed better than any of those metrics alone [Lambrix et al. 2008]. In addition, the X-SOM developers note that the optimal combination of metrics does not vary based on the domain of the ontologies but rather based on their design

characteristics [Curino et al. 2007].

While string similarity metrics are certainly not a new area of research, it remains unclear which string metric(s) are best for use in ontology alignment systems. In the OAEI competition algorithms surveyed for this work, 24 different string similarity metrics were used. In just the work cited above, Monge Elkan was found to be among the best performing metrics for name matching but among the worst performing for ontology alignment, yet several of the systems in the OAEI competition use Monge Elkan. Since nearly all alignment algorithms use a string similarity metric, more clarity in this area would be of benefit to many researchers. The work presented here expands on the previous efforts discussed above by considering a wider variety of string metrics, string preprocessing strategies, and ontology types. It also takes the work further by placing the string metrics into a complete ontology alignment system and comparing the results of that system to the current state of the art.

2.1 String Similarity Metrics

The Ontology Alignment Evaluation Initiative has become the primary venue for work in ontology alignment. Since 2006, participants in the OAEI competition have been required to submit a short paper describing their approach and results. All of these papers were surveyed to determine what lexical metrics were employed and what preprocessing steps were being used (or proposed). In cases where the paper was not explicit about the string similarity metric used, the code for the alignment algorithm was downloaded and examined when possible. The results of this survey are shown in Appendix B.

We can group string metrics along three major axes: global versus local, set versus whole-string, and perfect-sequence versus imperfect-sequence.

Global versus local refers to the amount of information the metric needs in order to classify a pair of strings as a match or a non-match. In some cases, the string metric needs to compute some information over all of the strings in one or both ontologies before it can match any strings. Such a metric is global. In other cases, the pair of strings currently being considered is all the input that is required. Such a metric is local. Global metrics can be more tuned to the particular ontology pair being matched, but that comes at the price of increased time complexity.

Perfect-sequence metrics require characters to occur in the same position in both strings in order to be considered a match. Imperfect-sequence metrics match characters as long as their positions in the strings differ by less than some threshold. In some metrics, this threshold is the entire length of the string. Imperfect-sequence metrics are thought to perform better when the word ordering

of labels might differ. This is common in biology-based ontologies. For instance, we would like to match "leg bone" with "bone of the leg." Imperfect sequence metrics are more likely to identify such matches. The drawback is that they also frequently result in more false positives. For instance, the words "stop" and "post" would be a perfect match for an imperfect-sequence metric if the threshold were the entire length of the string.

Largely orthogonal to these axes lie set-based string similarity metrics. A set-based string metric works by finding the degree of overlap between the sets of tokens contained in two strings. Tokens are most commonly the words within the strings. The set-based metric must still use a base string metric to establish if the individual tokens are equal (or close enough to be considered equal). This "helper" metric is often exact match, but it could be any non-set string metric. Word-based set metrics are generally thought to perform well on longer strings such as sentences or documents whereas they are assumed to give relatively high precision but low recall for shorter strings. Many ontologies have elements with short names that contain only a word or two, but ontologies in some domains may have longer labels. Also, the labels of individuals (versus classes or properties) in an ontology often have longer labels. Word-based set string similarity metrics may perform well in these situations.

The list below contains all string similarity metrics found in the review of OAEI participants and categorizes them based on the classifications described above. For set-based metrics, the underlying base metric used is given in parentheses. One combination does not contain any metrics: non-set/global/perfect-sequence. A subset of these metrics has been chosen for analysis related to various aspects of the ontology alignment problem. These metrics were chosen to reflect those most commonly used in existing alignment systems as well as to cover as fully as possible all combinations of the classification system provided. The chosen metrics are shown in bold.

- Set
  - Global
    - Perfect-sequence
      - Evidence Content (with exact)
      - **TF-IDF (with exact match)**
    - Imperfect-sequence
      - **Soft TF-IDF (with Jaro-Winkler)**
  - Local
    - Perfect-sequence

- **Jaccard (with exact match)**
- Overlap Coefficient (with exact)

* Imperfect-sequence
  - RWSA
  - **Soft Jaccard (with Levenstein)**

- Non-set
  - Global
    * Perfect-sequence
      - None
    * Imperfect-sequence
      - COCLU
  - Local
    * Perfect-sequence
      - **Exact Match**
      - **Longest Common Substring**
      - Prefix
      - Substring Inclusion
      - Suffix
    * Imperfect-sequence
      - Jaro
      - **Jaro-Winkler**
      - **Levenstein**
      - Lin
      - **Monge Elkan**
      - **N-gram**
      - Normalized Hamming Distance
      - Smith Waterman
      - Smith Waterman Gotoh
      - **Stoilos**
      - String Matching (SM)

The basic idea behind each metric is explained below. The list is organized alphabetically.

COCLU COCLU is short for Compression-based Clustering. The metric uses a Huffman tree to cluster the strings in one ontology and then matches each string in the second ontology to the appropriate cluster. Strings in the same cluster are considered equivalent. Whether to put a new string in a given cluster or create a new one is based on a distance metric called Cluster Code Difference (CCDiff), which is the difference between the summed length of the Huffman codes of all the strings in the cluster and the same with the new string added to the cluster. This has the effect of grouping together strings with the same frequent characters, regardless of the order of those characters. More information about COCLU can be found in [Valarakos et al. 2004].

Document Indexing The idea behind this approach is to use existing document indexing and retrieval tools as a string similarity metric. Each entity in the second ontology to be matched is treated as a document. The content of the document varies in different approaches. Options include any combination of an entity's label, name, id, comment, neighbors, ancestors, descendants, and instances. The documents (e.g. entities) are first indexed by a standard search engine tool such as Lucene or Indri. Then entities in the first ontology to be matched are treated as search queries over the second ontology. Matches are made to the best search results, provided that the quality is above a threshold set by the user.

Exact Match The most straightforward string similarity metric, exact match simply returns one if the two strings are identical and zero otherwise.

Evidence Content Evidence content is a cousin of the Jaccard metric. Rather than weighting each word equally, however, words are weighted based on their evidence content, which is the negative logarithm of the frequency of the number of entities a word appears in, relative to the entire ontology. See [Gaudan et al. 2008] for a discussion of this metric with respect to ontology alignment.

Hamming Distance (normalized) The Hamming distance is the number of substitutions required to transform one string into another. The normalized version divides this distance by the length of the string. This is similar to the Levenstein distance, but it only applies to strings of the same length.

Jaccard This is a classic string similarity metric. The formula is:

$Jaccard(s1, s2) = \frac{|A \cap B|}{|A \cup B|}$

The Jaccard metric is most commonly used as a set metric, where the union of A and B refer to all of the unique words in the two strings being compared and the intersection refers to the words common to both strings (as determined by simple string equality). It is also possible to use this metric as a base rather than set metric by considering individual letters instead of words in the strings.

Jaro This is another classic string similarity metric. The formula is:

$Jaro(s1, s2) = \frac{1}{3}(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m})$

where m is the number of "matching" characters and t is the number of transpositions. Two characters match if they are not further apart than $\lfloor(max(|s1|, |s2|)/2) - 1\rfloor$. Transpositions are cases where two characters match but appear in the reverse order.

Jaro-Winkler This variation on the Jaro metric gives preference to strings that share a common prefix. The thought is that many similar strings, particularly verbs and adjectives, have common roots but a variety of possible endings. The formula is:

$JaroWinkler(s1, s2) = Jaro(s1, s2) + (lp(1 - Jaro(s1, s2))$

where l is the length of the common prefix, up to four characters, and p is a weight for consideration of the common prefix (this must be less than 0.25 and is usually set to 0.1).

Longest Common Substring (LCS) This metric simply normalizes the length of the largest substring that the two strings have in common. The formula is:

$LCSSim(s1, s2) = \frac{2\cdot|\max CommonSubstring(s1,s2)|}{|s1|+|s2|}$

where $|x|$ is the number of characters in a string x.

Levenstein Edit Distance This is by far the most commonly used string similarity metric in ontology alignment systems. The Levenstein edit distance is the number of insertions, deletions, and substitutions required to transform one string into another. It can be normalized by dividing the edit distance by the length of the string (either the first string, to create an asymmetric metric, or the average of the lengths of both strings). Variations on this metric weight different types of edits differently.

Lin This metric is described in [Lin 1998]. The idea behind this metric is that the similarity between two things can be assessed by taking a measure of what they have in common and dividing by a measure of the information it takes to describe them. This definition has its basis in information

theory. They apply this intuition to determining string similarity using the following formula:

$$Sim(s1, s2) = \frac{2 \cdot \sum_{t|tri(s1) \cap tri(s2)} \log P(t)}{\sum_{t|tri(s1)} \log P(t) + \sum_{t|tri(B)} \log P(t)}$$

where tri enumerates the trigrams in a string and $P(t)$ is the probability of a particular trigram occurring in a string, which is estimated by their frequencies in the words (i.e. over all of the words in the ontologies).

Monge Elkan Monge and Elkan describe both a set-based similarity metric and a variant of the Smith-Waterman metric in their paper [Monge and Elkan 1996]. Different groups appear to refer to each of these as the "Monge Elkan metric." The SecondString library, a Java-based implementation of many different string similarity metrics, implements the Smith-Waterman variant as Monge Elkan, so that is what we will consider here.

This metric uses the Smith-Waterman approach with a match score of -3 for mismatched characters, +5 for the same characters (case insensitive), and +3 for "approximately" the same characters. This approximation is a variation on the original Smith-Waterman, along with the non-linear gap penalties used: 5 for a gap start and 1 for a gap continuation. The alphabet is upper and lower case letters, digits, period, comma, and space – all other characters are ignored.

Two characters are approximately equal if they fall into the same set:

- {d t}
- {g j}
- {l r}
- {m n}
- {b p v}
- {a e i o u}
- {. , space}

N-gram This metric converts each of the strings into a set of n-grams. For instance, if one of the words is "hello" and n is 3, the set of n-grams would be {hel, ell, llo}. The resulting sets for both strings are then compared using any set similarity metric (cosine similarity and Dice's coefficient are common). A variation is to have special characters to indicate "prior to the start of the string" and "after the end of the string." Using this approach, "hello" would result in the set {##h, #he,

hel, ell, llo, lo%, o%%}.

Overlap Coefficient This is very similar to the Jaccard metric. The formula is:

$Overlap(s1, s2) = \frac{|A \cap B|}{\min(|A|,|B|)}$

where A is the set of tokens (either words or characters) in the first string and B is the same for the second string.

Prefix This metric returns one if the first string is a prefix of the second, zero otherwise.

RWSA RWSA stands for Redundant, Word-by-word, Symmetrical, Approximate. This is based on the classification system for string similarity metrics presented in [Branting 2003]. Each string is indexed by the Soundex representation of its first and last words. Soundex is a phonetic encoding consisting of the first letter of the string followed by three digits representing the phonetic categories of the next three consonants, if they exist. The phonetic categories are:

1. B, P, F, V
2. C, S, K, G, J, Q, X, Z
3. D, T
4. L
5. M, N
6. R

When comparing two strings, a list of possible matches is retrieved by hashing the shorter of the two strings, and the remainder of the algorithm is run on these potential matches to find the best one and determine if it is above a threshold. These potential matches are all considered with respect to the indexing string. Both strings are broken into their component words. Two strings are considered to be a match if each word in the smaller of the two strings approximately matches a unique word in the larger string. An approximate match is one in which the edit distance is within a mismatch threshold. When computing the edit distance, there is a penalty of 1.0 for insertions, deletions and substitutions and a penalty of 0.6 for transpositions. The indexing to retrieve candidate matches may enable this metric to be used on larger ontologies than others that require each string to be compared against every other.

String Matching (SM) This metric was developed by Alexander Maedche and Steffen Staab and is described in [Maedche and Staab 2002]. It is essentially a normalized Levenstein edit distance in which the difference between the length of the shorter string and the edit distance is divided by the length of the shorter string.

Smith-Waterman This is a variant of the Needleman-Wunch metric and like that metric, it uses a dynamic programming algorithm [Durbin 1998]. To compare two strings, a matrix is created with the number of columns equal to the length of the first string and the number of rows equal to the length of the second string. The first row and first column are all zeros. All other elements i, j are set to the maximum of the following:

- 0
- $H(i-1, j-1) + w(s1(i), s2(j))$, for a match (mismatch) where w is the weight for a match (mismatch)
- $H(i-1, j) + w(deletion)$, for a deletion where $w(deletion)$ is the starting or continuing gap penalty
- $H(i, j-1) + w(insertion)$, for an insertion where $w(insertion)$ is the starting or continuing gap penalty

Once this matrix has been created, the distance between the two strings is found by starting with the highest value in the matrix and moving either up, left, or diagonally up and left, towards whichever value is highest. This is repeated until either a zero or the upper left corner of the matrix is reached. The distance is the sum of all of the values that were traversed.

Smith Waterman Gotoh This is a variation of the Smith-Waterman metric that has affine (non-linear) gap penalties. Because the length of the gaps doesn't matter in this version (a flat penalty is assessed for elongating an existing gap), a significantly faster implementation is possible [Gotoh 1982].

Soft Jaccard Unlike the Jaccard metric, soft Jaccard is a set metric *only*. It must be used in conjunction with a base similarity metric. First, the base similarity metric is run on all combinations of the words in both strings. The metric counts the number of these pairs in which the base metric result is greater than some threshold. This number is then divided by the number of words in the string with the higher word count. This is summarized in the formula below:

$SoftJaccard(s1, s2, t) = \frac{|sim(A_i, B_j) >= t|}{\max(|A|, |B|)}$

where A is the set of words in the first string, B is the set of words in the second string, t is the threshold for the base similarity metric, and sim is the base metric. The subscript i goes from 0 to the number of words in the first string and j does the same for the second string.

Soft TF-IDF This version of the TF-IDF metric is identical except that rather than requiring exact matches when computing the cosine similarity, words are considered matching if their similarity according to some base similarity metric is above a threshold. In our work, we have followed the lead of Cohen and his colleagues in [Cohen et al. 2003] and used Jaro-Winkler as the base metric.

Stoilos Metric for Ontology Alignment (SMOA) This string metric was specifically developed for use in ontology alignment systems. The main idea is to explicitly consider both the commonalities and differences of the two strings being compared. The formula is:

$Sim(s1, s2) = Comm(s1, s2) - Diff(s1, s2) + JaroWinkler(s1, s2)$

$Comm(s1, s2)$ finds the longest common substring, then removes it from both strings and finds the next longest substring, and so on until none remain. Then their lengths are summed and divided by the sum of the original string lengths. The formula for this is:

$Comm(s1, s2) = \frac{2 \cdot \sum |maxCommonSubString(s1, s2)|}{|s1| + |s2|}$

where $|x|$ is the number of characters in a string x.

$Diff(s1, s2)$ is computed using the following formula:

$Diff(s1, s2) = \frac{uLen(s1) \cdot uLen(s2)}{p + (1-p) \cdot (uLen(s1) + uLen(s2) - uLen(s1) \cdot uLen(s2))}$

where $uLen$ is the length of the unmatched part of the string from the first step divided by the length of the corresponding original string and p is the importance of the difference factor. The authors experimentally found 0.6 to be a good choice.

This metric ranges from -1 for completely different strings to +1 for identical strings. More information about this metric can be found in [Stoilos et al. 2005].

Substring Inclusion This metric returns one if the first string is contained within the second and zero otherwise.

Suffix This metric returns one if the first string is a suffix of the second one and zero otherwise.

TF-IDF/cosine TF-IDF stands for Term Frequency – Inverse Document Frequency. It is a technique used for document indexing in information retrieval systems. The term frequency is the

number of times a word appears in a document, divided by the number of words in the document. The inverse document frequency is the logarithm of the number of documents divided by the number of documents that contain the word in question. The idea behind using this approach for ontology alignment is that it is more indicative of similarity if two entities share a word that is rare in the ontologies than if they share a common word such as "the."

When computing the metric, the term frequency and inverse document frequency for each word in each document is computed (where document here means the same as described for the document indexing metric). This must be done for both ontologies before any entities can be compared. Then to compare two strings, each word's term frequency is multiplied by its inverse document frequency, creating a vector for each string. The string similarity is then the cosine similarity of the vectors.

2.2 String Preprocessing Strategies

This section describes all of the preprocessing approaches that were either tried or proposed by OAEI participants. The approaches mentioned by more than two participants are shown in bold italics – these will be examined in detail. Some operations are directly beneficial to the string similarity metric, while others primarily help with returning valid synonyms or translations (and so hopefully benefit the metric indirectly).

These approaches can be divided into two major categories: syntactic and semantic. Syntactic preprocessing methods are based on the characters in the strings or the rules of the language in which the strings are written. They can generally be applied quickly and without reference to an outside data store. Semantic methods relate to the meanings of the strings. These methods generally involve using a dictionary, thesaurus, or translation service to retrieve more information about a word or phrase.

- Syntactic
  - **tokenization**
  - split compound words
  - **normalization**
  - **stemming/lemmatization**
  - **stop word removal**
  - consider part-of-speech (i.e. weight functional words less)
- Semantic

- **synonyms**
- antonyms
- categorization
- language tag
- **translations**
- expand abbreviations and acronyms

Abbreviations and Acronyms Abbreviations and acronyms are particularly challenging for string similarity metrics. There have been several attempts to expand such shortcuts into their original representation by either looking them up in external knowledge sources or using language production rules. Reliable expansion of abbreviations and acronyms would be useful not just for the string metric, but also in improving synonym lookup and translations.

Antonyms Some similarity metrics consider differences as well as commonalities. A possible strategy for such metrics is to gather antonyms from a thesaurus in the same manner that synonyms are retrieved. These can then be used to determine that two strings are not equivalent.

Categorization In this approach, an external source containing a category hierarchy is used. Strings falling into the same category are considered more similar.

Compound Words It is possible that splitting compound words into their constituents can improve the performance of some set-based string similarity metrics.

Language Tag Ontology files sometimes use a language tag to specify the language of a particular string in the ontology. This can be used to avoid potentially misleading comparisons of words in different languages. It can also be used in conjunction with translations, to determine which language to translate from and which to translate to.

Normalization The idea behind normalization is to eliminate stylistic differences between strings as much as possible. This generally involves putting all characters into either upper or lower case, replacing punctuation characters with a space, and standardizing word order, often by alphabetizing the words within the string. Normalization might also involve transliterating characters not in the Latin alphabet to their closest equivalent.

Part-of-speech Similar in concept to stop word removal, it is possible to remove "functional" words such as articles, conjunctions, and prepositions from strings prior to assessing their similarity. Another possibility is to keep these words in the strings but weight them less than other words when a set-based string similarity metric is used.

Stemming/Lemmatization Stemming attempts to eliminate grammatical differences between words due to verb tense, plurality, and other word forms by finding the root of each word in the string. This topic has been studied in computer science since the sixties, and there are many existing algorithms. Stemming is both directly useful for string metrics and helpful in synonym lookup and translation.

Stop Word Removal Stop words are the most commonly used words in a language. The idea behind removing stop words from strings prior to computing their similarity is that very common words add little useful information. There are many lists of stop words available for different languages.

Synonyms In this preprocessing phase, strings are supplemented with their synonyms using either a general thesaurus such as WordNet or a domain-specific one such as UMLS for the biomedical domain. Biomedical ontologies also frequently have synonyms embedded in the ontology itself. Synonym lookup is by far the most often proposed preprocessing operation, but some who have actually implemented it report that it did not improve the performance of their system (e.g. SAMBO, GeRoMeSuite/SMB).

Tokenization Tokenization involves splitting strings into their component words. Word boundaries vary based on implementation, but often some combination of whitespace, underscores, hyphens, slashes, and lower-to-uppercase changes (to detect camelCase) is used. Tokenization is useful when comparing ontologies with different naming conventions, such as underscores versus hyphens to delineate words. This is particularly important for set-based string similarity metrics. In addition, tokenization is also needed for some other preprocessing steps, such as synonym lookup, translations, and word stemming.

Translation Translating strings when the ontologies to be matched are known to be in different languages has been suggested for a long time, but implementation has only become common in ontology alignment systems with the introduction of the Multifarm test set in the OAEI. The

language tag can be used to know which languages are involved, or a sample of the words in the ontologies can be analyzed to determine the languages.

2.3 Performance Evaluation

We have analyzed the performance of the selected string similarity metrics and preprocessing strategies on several OAEI test sets, the most popular ontology alignment benchmarks. Additionally, we have run the experiments on an analogous set of tests in order to evaluate the generalizability of the results. This section describes our experimental framework and presents an analysis of the results.

2.3.1 Experimental Setup

Here we describe the experimental framework and metric implementations in enough detail that others can reproduce our results. In addition, the source code for these experiments can be downloaded from http://www.pascal-hitzler.de/pub/StringMetricTester.jar.

The Ontology Alignment Evaluation Initiative (OAEI)[1] was started in 2004 with the goal of making it easier for researchers to compare the results of their ontology alignment algorithms. The organizers hold a contest each year in which participants run their algorithms on a variety of ontology matching problems and compare the results based on precision, recall, and f-measure. The OAEI features several tracks to test different types of ontology matching problems, three of which were used in this work.

The Conference track consists of finding equivalence relations among 16 real-world ontologies describing the same domain – conference organization. The ontologies are based on conference websites, software tools designed to support conference organization, and input from experienced conference organizers. These ontologies are all fairly small, with each one containing less than 200 classes and properties. The Multifarm track consists of the ontologies from the Conference track translated by native speakers into eight different languages: Chinese, Czech, Dutch, French, German, Portuguese, Russian, and Spanish (along with the original English). The goal is to align all combinations of languages. Finally, the Anatomy track consists of two ontologies from the biomedical domain: one describing the anatomy of a mouse and the other the anatomy of a human. As is common for biomedical ontologies, these are significantly larger than those found in the Conference track, with each containing around 3000 classes. More detail about the different OAEI tracks, particularly the Conference track, can be found in Chapters 3 and 4.

In order to get a sense of whether the results on the OAEI test sets generalize to similar cases, we

[1] http://oaei.ontologymatching.org/

have also run our tests on other ontology pairs of the same type. As an analog to the Conference test set, we have used two BizTalk files representing the domain of purchase orders: CIDX and Excel.[2] These were converted to an OWL format using a script that simply created a class for each entity (because we are only using the labels and not any relationship information, this is sufficient). The reference alignment for this dataset was created by domain experts. In addition, native speakers have assisted us in translating these schemas into German, Portuguese, Finish, and Norwegian so that we also have an analog for the OAEI Multifarm track. Finally, we have attempted to match the Gene Ontology[3] to the multifun schema[4], both of which cover topics from biomedicine (the Gene Ontology covers the general domain of genetics, while the multifun schema is a description of cell function). The multifun schema was put into an OWL format using the same procedure as the CIDX and Excel datasets. The reference alignment for this test set was also generated by domain experts.[5] The GO ontology and associated schema mappings are made possible by the work of the Gene Ontology Consortium [Ashburner et al. 2000].

Our test framework takes the two ontologies to be aligned and compares the label of every entity in the first ontology to every entity in the second ontology. The label is first considered to be the URI of the ontology entity, with the namespace (everything before the # character) removed. In the case that this approach results in an empty or null string, the "label" annotation of the entity is used instead. For each pair of labels, the metric being tested is run in both directions: metric.compute(labelA, labelB) and metric.compute(labelB, labelA). These results are put into two separate two dimensional arrays. Then the stable marriage algorithm is run on these two arrays to determine the best matches between the two ontologies. This algorithm finds a mapping such that, if A is mapped to B and C to D, it is guaranteed that A does not prefer D to B while D prefers A to C. This approach is used because in the OAEI reference alignments each entity is involved in at most one equality mapping. The version of the stable marriage algorithm used is deterministic. Finally, any mappings for which the minimum of metric.compute(labelA, labelB) and metric.compute(labelB, labelA) is less than one threshold or the maximum of those two values is less than a second threshold are thrown out. The resulting alignment is scored against the OAEI-provided reference alignment in terms of precision, recall, and f-measure.

Due to the nature of the test framework, each metric requires at least two parameters: the thresholds for the similarities between the two strings (in both directions). For each test, each metric had both of the parameters initially set at 1.0. The parameters were then both decreased

[2]http://disi.unitn.it/~accord/Experimentaldesign.html
[3]http://www.geneontology.org/GO.database.shtml
[4]http://genprotec.mbl.edu/files/MultiFun.html
[5]http://www.geneontology.org/GO.indices.shtml

in steps of 0.1 until the f-measure ceased to improve. Then the first parameter was decreased in steps of 0.1 while the second was held constant. Then the second parameter was decreased in steps of 0.1 while the first was held constant. Then the first parameter was increased in steps of 0.1 and finally the second parameter was increased in steps of 0.1. This entire process was repeated as long as improvements in f-measure continued to be made. In addition, the soft set metrics (soft Jaccard and soft TF-IDF) require an additional parameter. This was initially set at 0.9 (setting it at 1.0 would have negated the "soft" aspect of the metrics), the test was run according to the previous description, then the third parameter was set to 0.8 and the process was repeated. This process was repetitive in some cases, but it was a reasonably thorough search of the parameter space for each metric.

Below we discuss the implementation details of the string preprocessing approaches that were tested.

Tokenization Tokens are delimited by dash, underscore, space, camelCase, forward slash, and back slash. Each token is put into all lowercase. This is done to prevent camelCase labels from retaining a difference between tokens.

Stop Word Removal The Glasgow IR group's stop words list is used.[6] It consists of 318 common English words. Tokenization is done prior to stop word removal.

Stemming The Porter stemming algorithm is used [Porter 1980]. Tokenization is done prior to stemming to handle cases like runningTotal.

Normalization Any whitespace is replaced by a single space. Any letters with diacritic marks, umlauts, etc are replaced with the closest corresponding letter from the English alphabet. Russian characters are transliterated using the approach specified here: http://www.translit.cc/. We were unable to transliterate Chinese symbols. As a final step, the label is tokenized and the tokens are ordered alphabetically.

Synonyms This test was only run on the Conference and Anatomy test sets. The Multifarm test set could be included in the future if appropriate electronic thesauri could be found for each language. The labels are first tokenized and then synonyms are looked up either in WordNet, for the Conference set, or in the "synonyms" attribute of the entity itself, for the Anatomy set.

[6] http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words

There were some differences between using WordNet and the internal synonyms. The JWNL Java WordNet API was used to query WordNet. Tokenization must be done to get reasonable hit rates for synonym lookup from WordNet In addition, the WordNet API does its own form of word stemming internally, and that was left in place because it is the most common way to use WordNet in applications. A question arose on how to handle labels that are phrases when querying WordNet. Looking up "masters thesis" returned only the synonyms for "masters" and nothing for thesis or for the phrase as a whole. The same was true when masters_thesis was used as the query (even though WordNet will return items in a similar form, such as baseball_bat). Therefore the synonym set is generated by querying WordNet for each token in the label and aggregating the results. So for this example, the synonym set contains all synonyms for masters plus all synonyms for thesis.

A second question concerned how to compute the overall similarity value based on the similarity values of the synonyms. After some preliminary experimentation, it was determined that the synonyms provided within the Anatomy ontologies are far more specific and relevant than those arrived at by querying WordNet for the Conference ontologies. As a result, the best strategy for computing overall similarity in the Anatomy case was to take the maximum similarity value of the label of the first entity compared to that of the label and each of the synonyms of the second entity (and the same for the other direction). For the WordNet case the best result was obtained by employing a set similarity metric: the similarity values for the first entity's label and all of its synonyms were computed with respect to the second entity's label and all of its synonyms. All of these values were summed and divided by the size of the synonym set of the first entity, plus one for the label itself. We also tried using the same approach on the Conference test set that was used on the Anatomy test set, but the results were much worse than those resulting from the approach just outlined.

Translation Only the Multifarm test set was used for this experiment. Google Translate (via the Google Translate API) is used to translate from one language to another. Google Translate can handle translations between all of the languages in the OAEI test set. Unfortunately, it is not free. The cost is $20 per 2 million characters. It cost $12.08 to align all of the ontology pairs in the test set once. To avoid paying that for each metric multiple times (to optimize the parameter values), the results were cached and the cache was used after the first run. This does not affect the accuracy of the metrics. The service can also detect the language of the input it is provided with – the labels of ten randomly chosen entities were submitted to the translation service to detect the language. Google Translate does some internal preprocessing involving stemming, but Google does not provide details on this.

Following is a discussion of the implementation details for each string metric tested. We used existing implementations from open source libraries whenever possible.

Exact The Java String class's startsWith method is used for this metric. The reason for using startsWith rather than the equals method is that this makes the metric asymmetric. For instance, exact.compute("leg bone", "leg") returns 1 because "leg bone" starts with "leg" while exact.compute("leg", "leg bone") returns 0 since "leg" does not start with "leg bone".

Jaccard The SecondString library implementation of this metric is used.

Jaro-Winkler The SecondString library implementation of this metric is used.

Longest Common Substring This metric was coded based on the following logic: find the shorter of the two strings, for each character in that string, check to see if the longer string contains that character. If it does, find the length of the longest common substring starting with that character. The maximum of all of these lengths is then divided by the length of the first string and returned. This is an asymmetric metric.

Levenstein The distance between the two input strings is computed using the SecondString implementation of the Levenstein metric. The distance is then normalized by dividing it by the length of the first input string (creating an asymmetric metric). In order to have 1 rather than 0 represent a perfect match, the normalized distance is then subtracted from 1.

Monge Elkan The SecondString implementation of this metric is used.

N-gram After some initial experimentation, n was set equal to 3 for all tests. This metric was coded based on the following logic: construct all trigrams for each input string, representing characters prior to the beginning of the string with '#' and those after the end of the string with '%'. Return the number of trigrams common to the two strings, divided by the number within the first string (asymmetric metric), being sure to handle duplicate trigrams correctly.

Soft Jaccard A set of the unique words in the first string (where uniqueness is determined by a Levenstein distance less than a threshold from any word already in the set) and another set of the unique words in the second string are created. The intersection and union of these two sets are

computed, and the metric returns the size of the intersection divided by the size of the union. The Levenstein distance is computed using the Second String library implementation.

Soft TF-IDF The Second String library Soft TF-IDF class was used as the basis for this implementation. The internal metric used was the Second String implementation of Jaro Winkler. A SimpleTokenizer was created to split the strings into words with whitespace as the delimiter. The Soft TF-IDF dictionary was created using the words from all of the labels in both ontologies. If the test considered synonyms, sets of synonymous words were maintained and only one representative word from each set was added to the dictionary. If the test considered translation, the word was first translated to the appropriate language before being added to the dictionary. A BasicStringWrapperIterator was used to train the metric.

Stoilos This is coded based on the definition provided in the paper in which the metric was originally proposed [Stoilos et al. 2005]. There was a point of confusion related to the Jaro-Winkler-based improvement factor mentioned in that paper, however. The paper states that the metric should range from -1 to +1, but with that factor in it ranges from -1 to +2. We tried both approaches when attempting to reproduce the results mentioned in the paper but achieved at best an f-measure that was around 0.1 lower than what was reported. For instance, on the 301 benchmark test with a 0.6 threshold, our tests resulted in precision = .83 and recall = .68 for an f-measure of .75 while the authors report a precision of .98, recall of .79, and f-measure of .87. The results using the Jaro-Winkler improvement factor were significantly worse. The experiments here were conducted without that factor included.

TF-IDF The Second String library TF-IDF class was used as the basis for this implementation. A SimpleTokenizer was created to split the strings into words with whitespace as the delimiter. The TF-IDF dictionary was created using the words from all of the labels in both ontologies. If the test considered synonyms, sets of synonymous words were maintained and only one representative word from each set was added to the dictionary. If the test considered translation, the word was first translated to the appropriate language before being added to the dictionary. A BasicStringWrapperIterator was used to train the metric.

2.3.2 Results

In this section we review the results of the experiments described above.

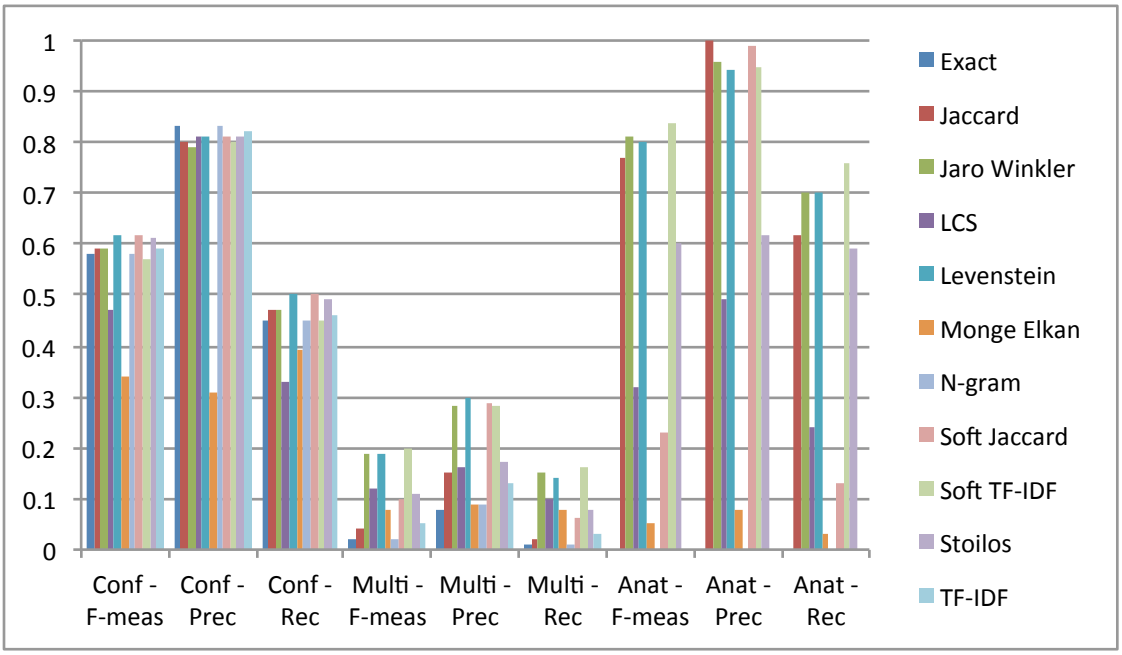


Figure 2.1: Results with no string preprocessing

2.3.2.1 OAEI Results

First, we look at the effect of the different string preprocessing strategies on precision, recall, and f-measure for the OAEI test sets. Figures 2.1 through 2.7 show the results for one string preprocessing strategy on all three OAEI datasets (Conference, Multifarm, and Anatomy). F-measure, precision, and recall are all shown on the graphs.

Figure 2.1 shows the results of when no string preprocessing is employed.

The Conference dataset does not reveal much disparity among the string similarity metrics. If we leave out the Monge Elkan and Longest Common Substring metrics, which perform very poorly, we are left with very little standard deviation for either precision or recall in this test set. Also, the optimal thresholds indicate that the best approach is to look for matches that are as exact as possible.

The Multifarm test set is much more challenging – both precision and recall are less than one fourth what they were for the same ontologies in English. This test set also reveals a much wider disparity among string similarity metrics. There is a large standard deviation for both precision and recall. This is likely because most ontologies, including these, contain the majority of the semantic information in labels. This intrinsic information is hidden from string similarity metrics by translation.

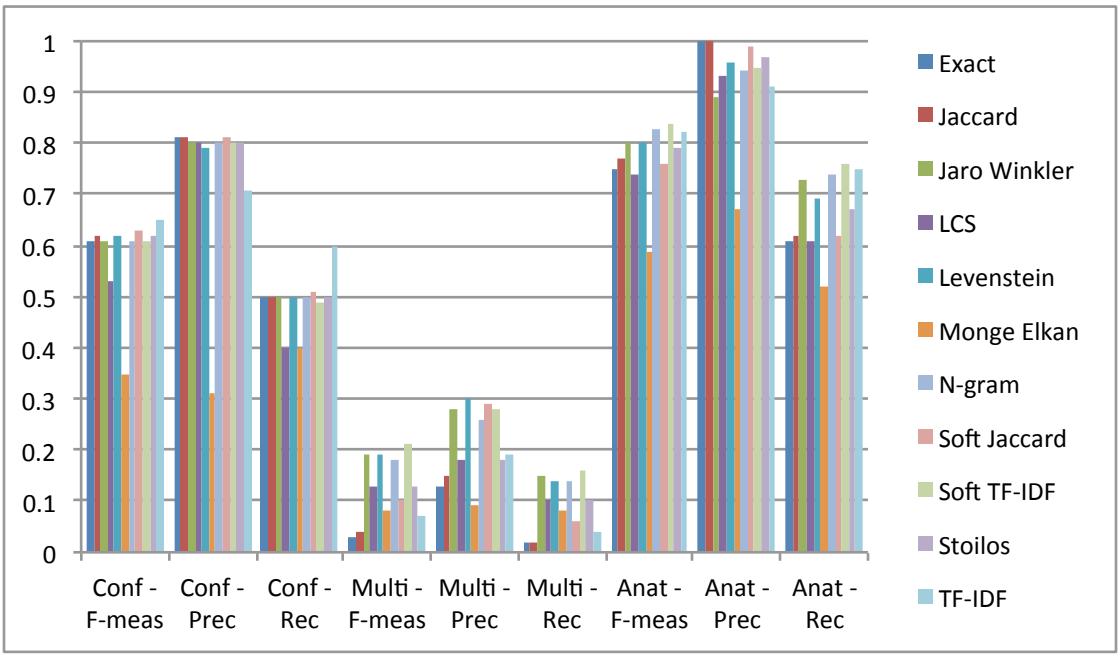


Figure 2.2: Results with tokenization

The exact, n-gram, and TF-IDF metrics cannot generate any matches for the Anatomy test set. Furthermore, the longest common substring, Stoilos, and Monge Elkan metrics perform very poorly. For the remaining metrics, we find that this test is easier for string similarity metrics than the Conference dataset. This is expected because biomedical datasets usually deal with a smaller, more regular vocabulary. There is often a small set of nouns with associated modifiers. The precision for this dataset is extremely high and the standard deviation of the precision is relatively low. There is a much higher standard deviation of recall. In particular, the metric with the next-to-highest precision (Soft Jaccard) has by far the lowest recall. This is a dataset where set metrics do particularly well, again because the biomedical domain frequently involves phrases that can be presented in different orders. The Anatomy test set is also interesting in that there is a more clear choice to be made between metrics that have good precision and those that have good recall – there is not a lot of overlap between these two sets.

Figure 2.2 shows the same results using tokenization.

On the Conference dataset, there were improvements in the recall of most metrics (a large one for LCS and TF-IDF) with little to no decrease in precision, except for TF-IDF. The recall of TF-IDF went from average to much better than any of the other metrics on this dataset with tokenization. This improvement in recall seems primarily due to the use of underscores as word separators in some

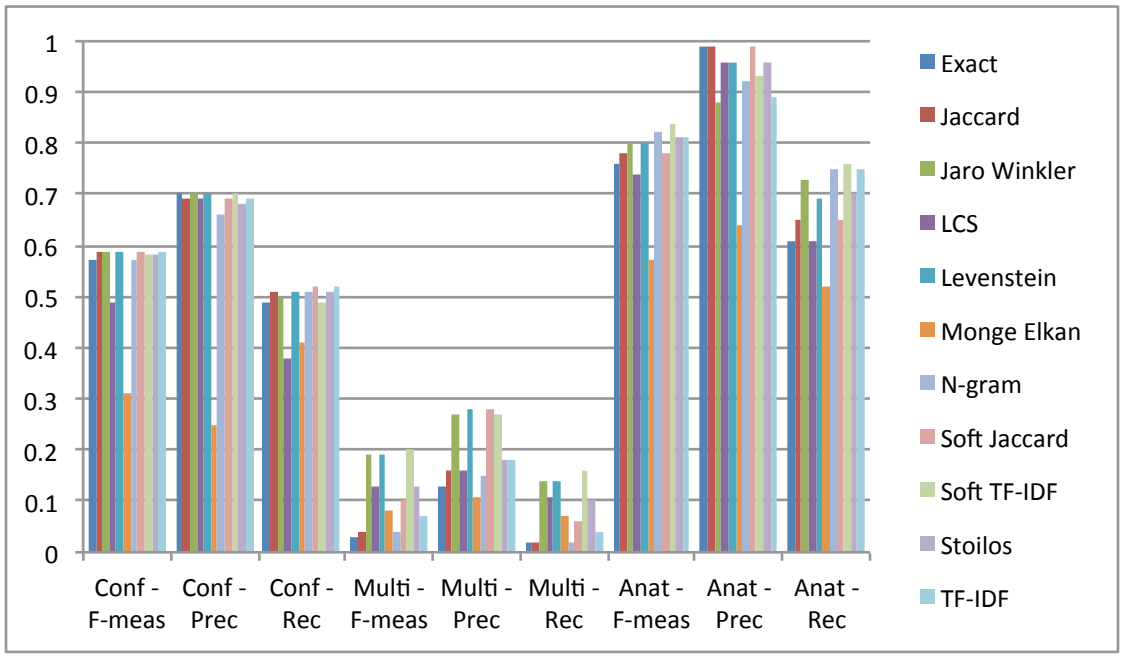


Figure 2.3: Results using stemming

ontologies and camelCase in others.

For the Multifarm test set there were no improvements in the best-performing metrics due to tokenization.

After tokenization, the exact, n-gram, and TF-IDF metrics are capable of producing results on the Anatomy dataset whereas without it they could not. In fact, exact now has perfect precision. The recall of most metrics was improved slightly or left unchanged. In the case of the n-gram metric this was significant enough to make it one of the best-performing metrics in terms of recall. These changes were enough to change the set of top performing metrics in terms of f-measure as well.

In general, it makes sense to perform tokenization as a preprocessing step – it improves overall performance (especially recall) slightly, particularly for metrics that involve exact match.

Figure 2.3 shows that when compared with tokenization as a baseline, stemming does not improve performance on any of the test sets (it actually slightly hurts precision on the Conference set). The only exception is that recall of the Soft TF-IDF metric is improved by 27% on the Conference test set, and that metric becomes the best choice for that test set in terms of recall. Recall of the n-gram metric on the Anatomy test set is hurt badly enough to move it out of the best-performing metrics for that category.

Removing stop words lowers precision on the Conference test set and has essentially no effect on

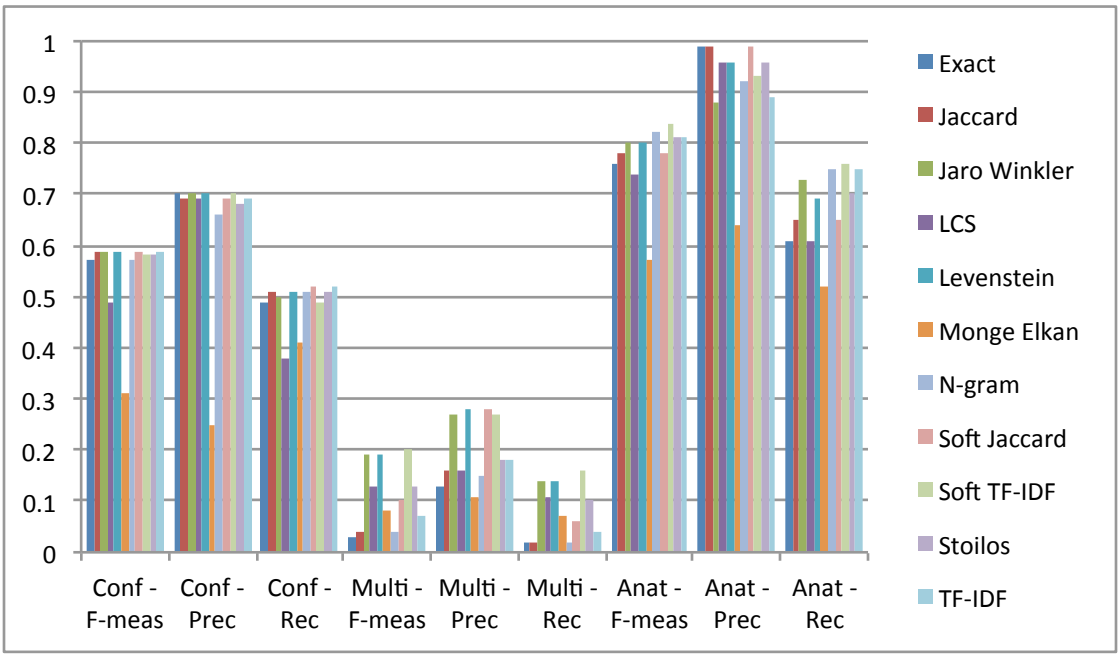


Figure 2.4: Results using stop word removal

the other two test sets when compared to tokenization (see Figure 2.4).

Figure 2.5 shows that normalization had little effect on the Conference test set. Performance in terms of both precision and recall on the Multifarm test set greatly improved with normalization (mostly due to the transliteration). Normalization had little effect on the precision or recall of any metric on the Anatomy test set.

Figure 2.6 shows the results when synonyms are considered. This hurts both precision and recall for all metrics on the Conference test set. On the Anatomy test set, the precision of all metrics was either flat or worse than for tokenization alone, but the recall of several metrics improved enough to raise the f-measure when synonyms were considered.

Translations result in huge improvements in both precision and recall for all metrics on the Multifarm test set, both over tokenization alone and over normalization (see Figure 2.7). There is a wide variation in the performance of metrics in this configuration on this test set. Also, the metrics with good precision have mediocre recall and vice versa.

2.3.2.2 Comparative Results

The next set of graphs presents the results of the same tests on the non-OAEI datasets. These were conducted in order to determine whether the results presented above are specific to the OAEI

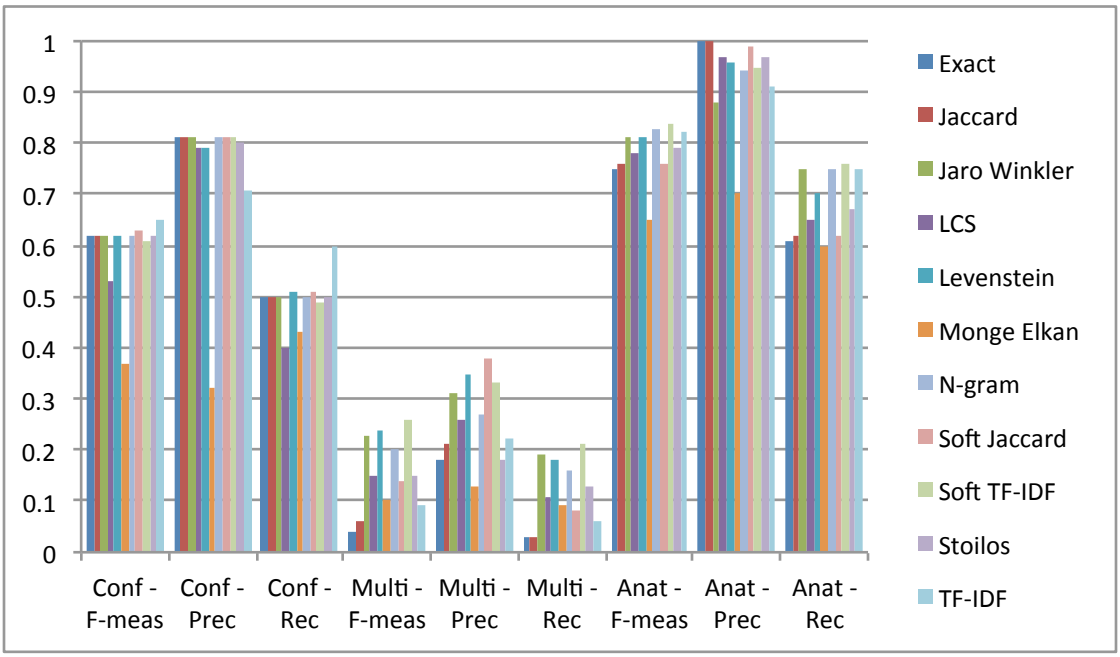


Figure 2.5: Results using normalization

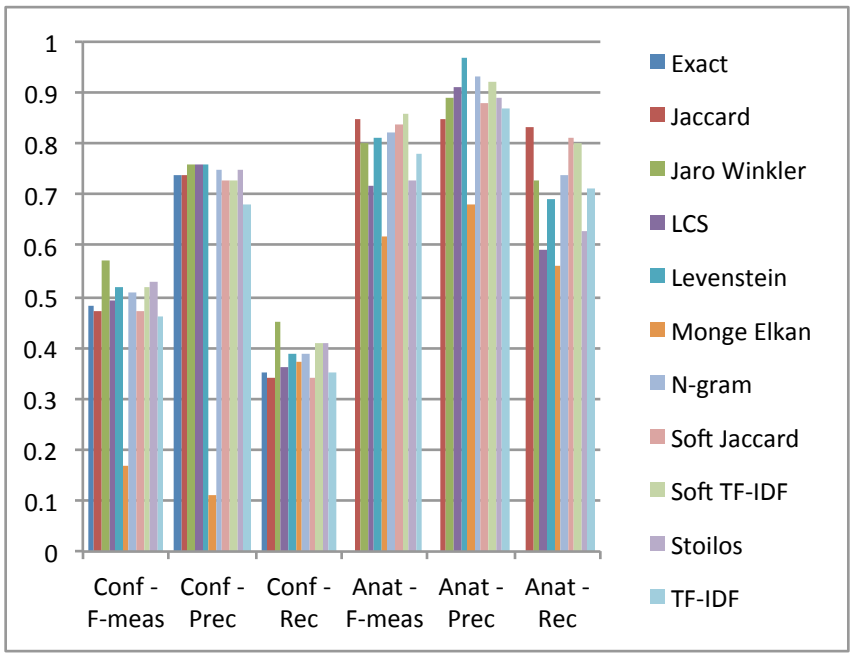


Figure 2.6: Results using synonyms

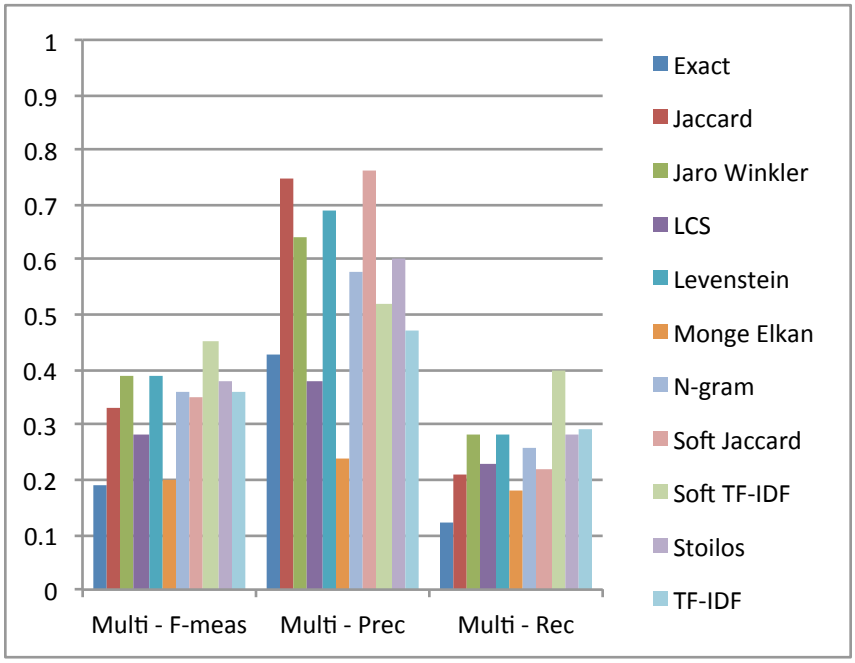


Figure 2.7: Results using translations

datasets or if they carry over to other ontologies of the same general type. Figure 2.8 shows the f-measure of the best-performing metric using each preprocessing strategy while Figure 2.9 shows the f-measure of each metric using the best-performing preprocessing strategy. The same information is shown for the analogous OAEI dataset for comparative purposes. Variations in the absolute heights of the bars between analogous datasets are to be expected because the overall difficulty of matching a particular ontology pair may vary considerably – what we are looking for is the same general shape of the bars for the adjacent sets (or a clear understanding of any differences).

For the most part the preprocessing strategies exhibit similar behavior on the analogous datasets, as shown in Figure 2.8. The only exception is that stemming improves performance on the Genes test set but not on Anatomy. Further analysis shows that this difference turns out to not be significant, however. The TF-IDF metric is the top-performing metric on the Genes test set. It turns out that many of the entities in the Genes dataset that are successfully matched using stemming contain the word "transport" or "transporter". When stemming is used, these two words are considered the same and the TF-IDF metric weights them less due to a more frequent occurrence in the ontologies, thereby allowing more correct results. In short, a single lucky break has resulted in a rather noticeable variation in performance.

The significantly smaller sizes of the non-OAEI test sets cause more variability in metric perfor-

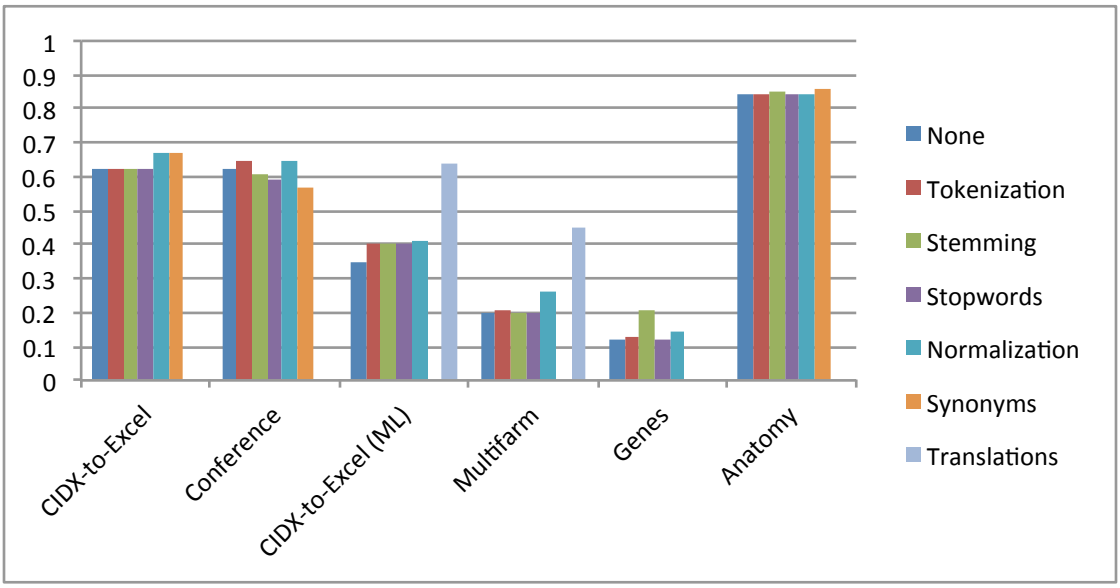


Figure 2.8: F-measures of the best-performing metric on all test sets for all string preprocessing strategies

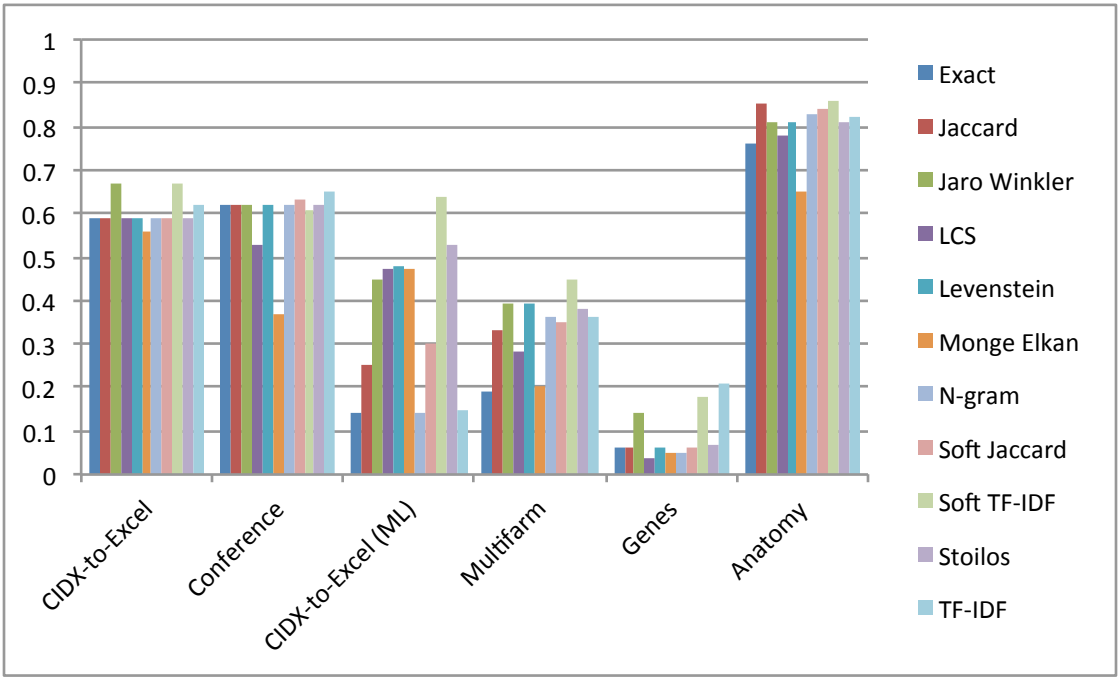


Figure 2.9: F-measures of all metrics on all test sets using the best-performing string preprocessing strategy

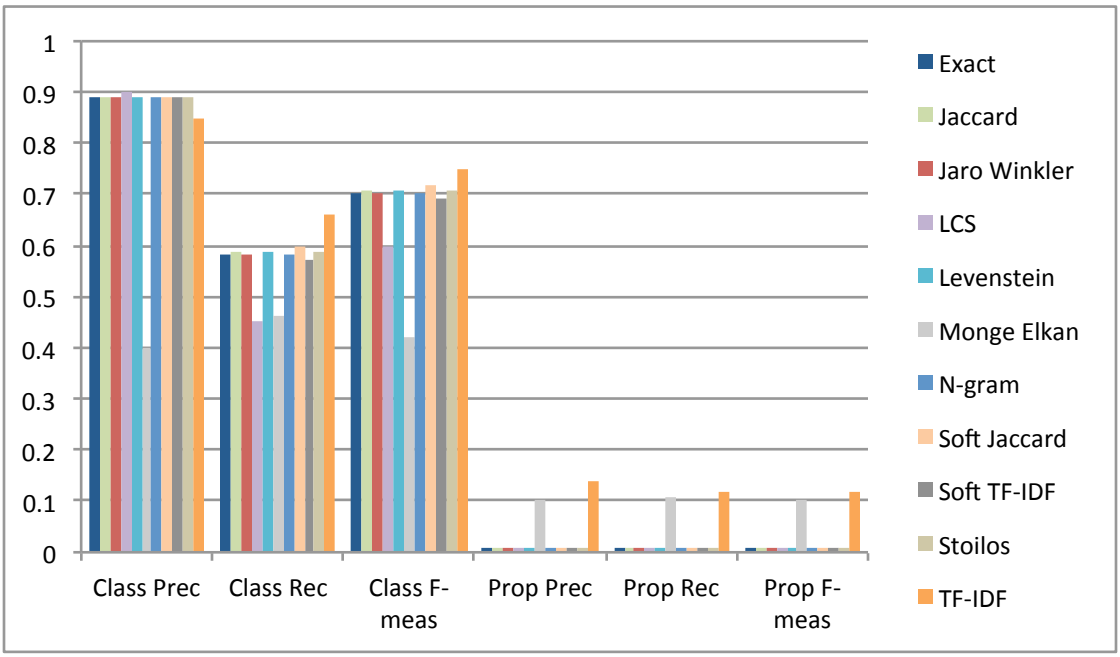


Figure 2.10: F-measures of all metrics on the classes and properties in the Conference dataset using string tokenization

mance (i.e. because there is a small number of matches, a metric is more heavily rewarded if it "gets lucky" on a particular match and more heavily penalized if it does not). However, we see from Figure 2.9 that choosing a string similarity metric is less important for "standard" ontologies because performance varies little among metrics. This is not the case for the multilingual and biomedical ontologies. In addition, we see that choosing a string similarity metric based on its performance on the OAEI test sets leads to good relative performance on analogous ontology matching problems.

2.3.2.3 Classes vs Properties

Others have found that human experts have a more difficult time agreeing on when properties match than on classes [Maedche and Staab 2002]. We seek here to determine if string similarity metrics also have particular difficulty with properties. In this section we look at the performance of the metrics on classes versus properties for the Conference and Multifarm datasets. There are no matching properties in the Anatomy test set.

From Figures 2.10 and 2.11 it is evident that string similarity metrics perform much worse on properties than on classes. It appears from an empirical analysis of the alignments that properties are particularly challenging for ontology alignment systems for several reasons. Properties

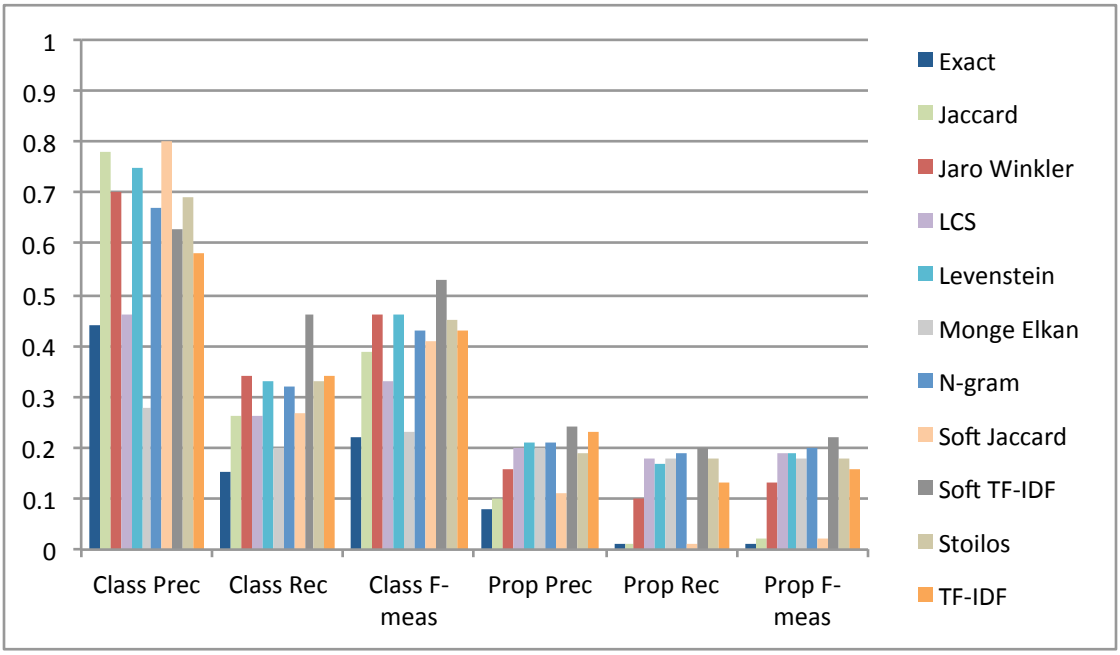


Figure 2.11: F-measures of all metrics on the classes and properties in the Multifarm dataset using string tokenization

frequently involve verbs, which can appear in a wider variety of forms than nouns (in addition to plurality/conjugation, verbs vary by tense). There are also often more functional words, such as articles and prepositions, in property names. Also, there are generally more common synonyms available for the (often very generic) verbs in property names than the (often more specific) nouns in class names. We therefore thought that stemming, stop word removal, or synonym lookup might be effective when matching properties. However, that turned out not to be the case. Figure 2.12 shows the effect of various preprocessing strategies in combination with the two metrics that performed the best on properties for the Conference test set: Monge Elkan and TF-IDF. Tokenization is required for the TF-IDF metric to work because it is a global set metric. Normalization improved the performance of Monge Elkan but not TF-IDF. Analysis of the results seems to indicate this is because putting the words into alphabetical order reduced the number of gap penalties for matching properties in Monge Elkan. This had no effect for TF-IDF because set metrics are not sensitive to word order.

It is curious that properties are more easily matched on the Mulitfarm dataset. This dataset consists of exactly the same ontologies as the Conference set, just translated into a variety of languages. It will be interesting to explore what is going on there, but the assistance of native speakers

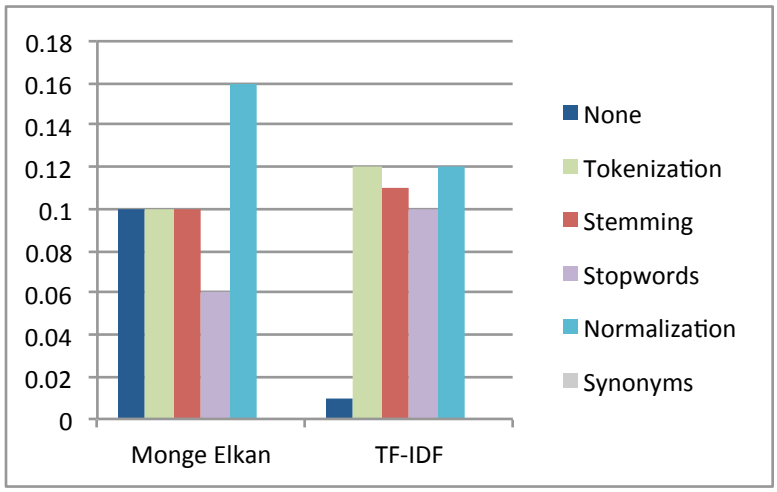


Figure 2.12: F-measures of Monge Elkan and TF-IDF on properties in the Conference dataset for all of the string preprocessing strategies

of some of the other languages will likely be required.

The above results were collected using the best thresholds found by optimizing the f-measure on the overall alignment problem (both classes and properties). In addition, we wanted to determine whether it was helpful to choose different thresholds for classes and properties. Figures 2.13 and 2.14 show the best results achieved for property matching on both the Conference and Multifarm datasets when the thresholds were optimized based solely on the f-measure for properties. The precision, recall, and f-measure when the thresholds were optimized for overall f-measure are reproduced here for ease of comparison. The results are better than in the previous case, indicating that for these datasets there is value in selecting different similarity metric thresholds for class and property comparisons.

2.3.3 Analysis

The results of the different metrics on the test sets reveal a potential trap for developers of ontology alignment systems. Results on the Conference test set, which is representative of many real-world ontologies, do not reveal much difference in the performance of the metrics in terms of f-measure. Basically, if you pick any string similarity metric and set the threshold high (i.e. between .9 and 1.0) then the results will be near optimal. However, the other test sets reveal that all string metrics are not created equal – performance of different metrics on the multilingual and biomedical test sets varied considerably. Choosing a string metric for use on these alignment tasks involves a significant impact on precision and recall. The moral of the story is that when choosing a string metric for use

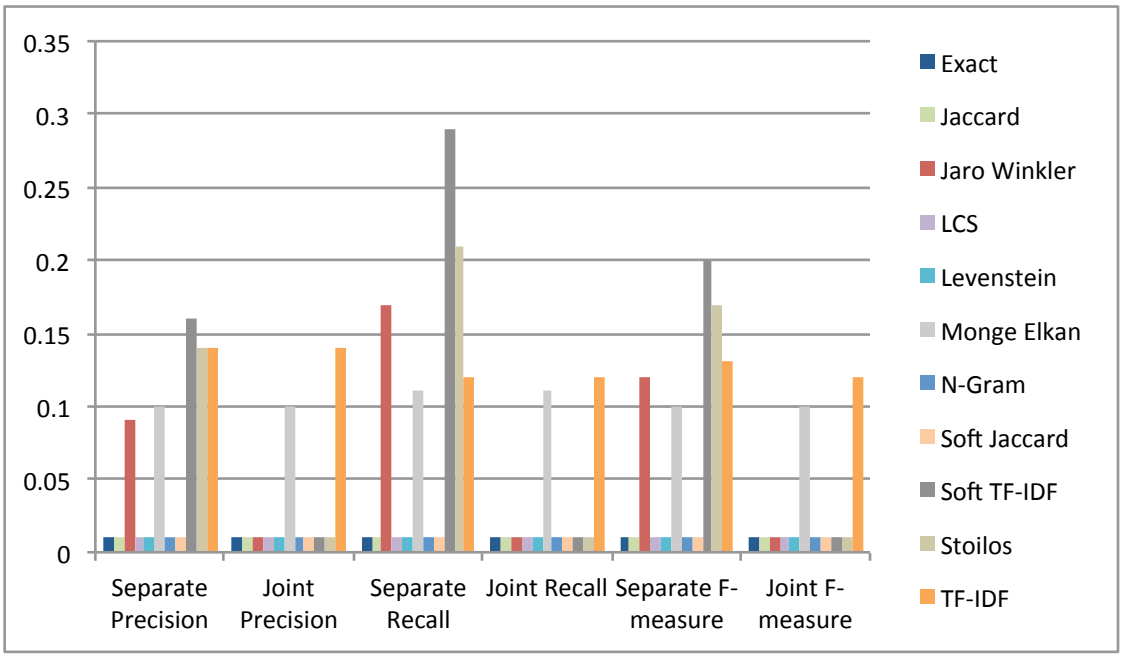


Figure 2.13: F-measures of all metrics using tokenization on the Conference dataset when the thresholds were optimized once for classes and properties together versus separately for properties

in an ontology alignment algorithm, one should consider the characteristics of the ontologies being aligned and whether precision or recall is more important for the algorithm. Below are some general guidelines:

- Standard ontology
  - Precision: All but Monge Elkan
  - Recall: TF-IDF
  - F-measure: All but Monge Elkan and LCS
- Multilingual
  - Precision: Soft Jaccard, Jaccard
  - Recall: Soft TF-IDF
  - F-measure: Soft TF-IDF
- Biomedical
  - Precision: Levenstein

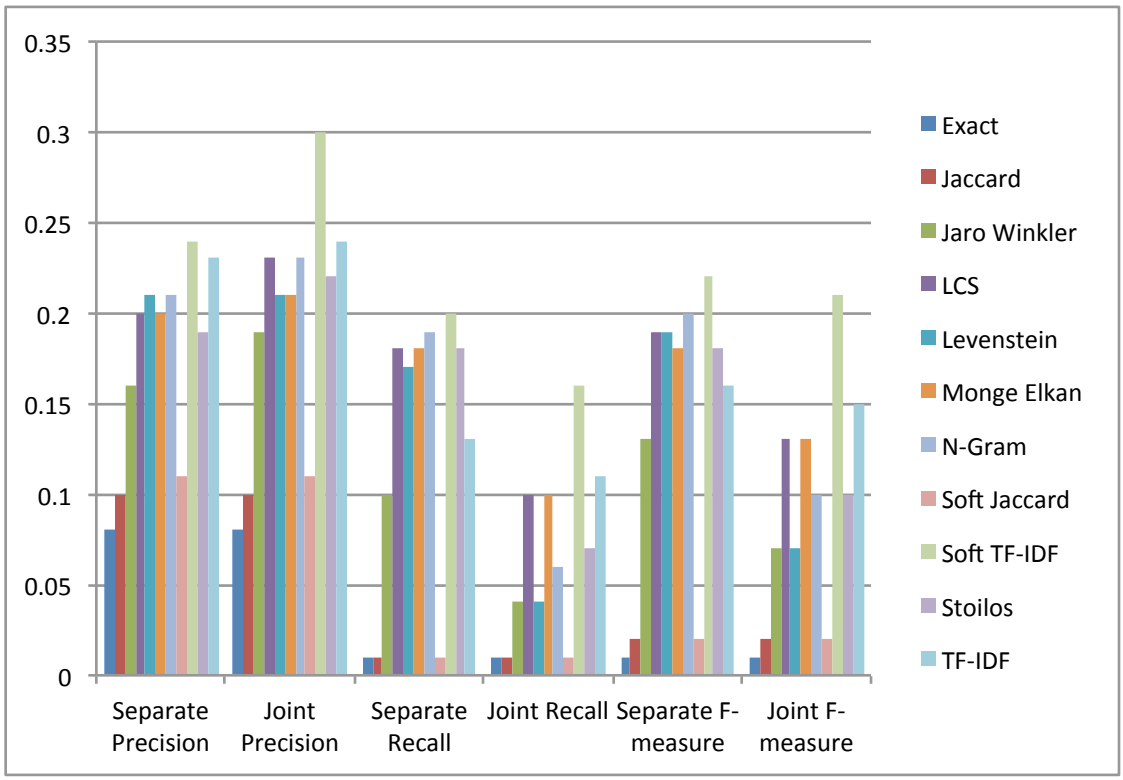


Figure 2.14: F-measures of all metrics using tokenization on the multiform dataset when the thresholds were optimized once for classes and properties together versus separately for properties

- Recall: Jaccard, Soft Jaccard, Soft TF-IDF
- F-measure: Soft TF-IDF, Jaccard, Soft Jaccard

Of the preprocessing strategies analyzed, few were beneficial. Tokenization is useful if the naming conventions differ between the ontologies (camelCase versus underscores to separate words, for example). Translation is very helpful when ontologies involve multiple languages. If translation is not available, normalization can be useful for multilingual ontology pairs, particularly if one of the languages uses a non-Latin alphabet and can be transliterated. Synonyms can be useful (particularly with respect to recall) for biomedical ontologies, where the synonyms are often embedded in the ontologies themselves.

2.4 String-based Ontology Alignment

With this analysis of string similarity metrics as applied to ontology alignment, we now turn to the question of how far we can get using *only* string metrics. To answer this question we developed a very simple ontology alignment algorithm. This algorithm works in the same way as our test framework – comparing every label in the first ontology to every label in the second and using the stable marriage algorithm to find the best mappings. The difference is that here we run the algorithm repeatedly: first with a high-precision metric and then with a high-recall metric. If a mapping in the second pass involves an entity that has already been used in the previous pass then it is ignored. Because string metrics were found to perform extremely poorly on properties, this approach does not attempt to match those (e.g. any property matches in the reference alignment are automatically false negatives). For the Anatomy test set, the approach used here first runs the high precision and high recall metrics on the entity labels themselves, and then considers synonyms. For this proof-of-concept, the algorithm is hard-coded with the optimal metrics and thresholds for the particular test set under consideration. The results are shown in Tables 2.1, 2.2, and 2.3, along with the results of the OAEI 2013 competitors. This approach is labeled StringsOpt, to indicate that this is the optimal configuration of the framework for each alignment task. Note that StringAuto, also listed in these tables, is a similar approach that will be discussed shortly.

Using only optimized string similarity metrics achieves an f-measure of .67 on the Conference dataset, which makes it the fifth highest performer. The strings only approach also significantly outperforms the baselines edna and StringEquiv, which are unrefined string metrics (StringEquiv uses string equality and edna is an edit distance metric with a threshold of .81). For the Multifarm test set, only the results of metrics that have been designed to handle multilingual alignment problems are shown. These results are divided into two groups: alignments of the same ontologies

in different languages (labeled "same") and alignments of different ontologies in different languages (labeled "dif"). The strings only approach is the second-best system in terms of f-measure on both the same and different ontologies. This approach ties for fifth on the Anatomy test set.

It is evident that these results compare very well with the current state-of-the-art in ontology alignment systems, but ***this is not an apples-to-apples comparison*** because this approach is not generic (due to the hard-coded metrics based on the test set). The next step is to add some means of selecting the appropriate string metrics and thresholds at runtime. Our goal is to develop a method that is fully autonomous and does not rely on any training data. We have started out with some basic observations based on the results we have gathered above:

- Precision does not vary widely among string similarity metrics for most standard ontologies.
- TF-IDF has high recall for most standard ontologies.
- When many entities in an ontology have labels that contain multiple words, it is best to use set-based string similarity metrics. Soft Jaccard and Soft TF-IDF often perform particularly well in these cases.
- Thresholds need to be lower when recall is of more concern than precision.
- Thresholds need to be lower when synonyms or translations are involved.

Based on these insights, we have developed an analysis module that runs before our main alignment algorithm to select the string metrics. This analysis module examines an ontology to find the answers to three simple questions:

- Is the ontology in English?
- What is the average number of words per entity label (after tokenization)?
- Does the ontology contain embedded synonyms?

The implementation of the analysis module is straightforward. The language of the ontology is determined by selecting ten entity labels and concatenating these into a single string, which is then used in a call to the "translate" function of the Google Translate API with English as the target language. This call is made with no value for the source language parameter, which causes Google to return its best guess as to the source language along with the translation. If the language is something other than English, the English translation is used in the remaining steps. The calculation of the average number of words is straightforward. Synonym detection is done simply by checking

the input files for mention of the word "synonym." The analysis module only considers classes – properties and instances are ignored.

Table 2.4 shows the average number of words per label for each dataset. From this we see that the labels in biomedical ontologies are typically made up of more words than those in standard ontologies. Also interesting is that the number of words per label is slightly greater for the multilingual version of a test set than the same ontologies in English (i.e. the metrics are higher for Multifarm than for Conference and for CIDX-to-Excel (ML) than for plain CIDX-to-Excel). This seems to be because Google Translate sometimes produces a multi-word phrase instead of a single word when performing translations.

Based on these results of the analysis module and whether precision or recall is currently of interest in the alignment process, a string metric is chosen. This is currently done using a hard-coded set of rules, but more research remains to be done in this area. When precision is the primary concern, it doesn't matter too much which metric we choose for most standard ontologies. We have decided to use Jaro-Winkler. In cases where multiple words per label are involved, word ordering often confuses the results. We therefore use the Soft Jaccard metric in these cases, with Levenstein as the base metric. When recall is the primary focus, we use TF-IDF for ontologies with predominantly one word per label and Soft TF-IDF for those with mostly multi-word labels. The thresholds used are shown in the decision tree below. In general, if translations or synonyms are involved then lower thresholds are appropriate. Note that these rules do not break cleanly among the different OAEI test sets – they are based on underlying features of the ontologies to be matched.

- Precision
  - Less than two words per label

    Jaro-Winkler 1, 1
  - Two or more words per label
    - Synonyms

      Soft Jaccard .2, .5 with Levenstein .9 base metric
    - No synonyms

      Soft Jaccard 1, 1 with Levenstein .8 base metric
- Recall
  - Less than two words per label

    TF-IDF .8, .8
  - Two or more words per label

* Synonyms

  Soft TF-IDF .5, .8 with Jaro-Winkler .8 base metric

* Different Languages

  Soft TF-IDF 0, .7 with Jaro-Winkler .9 base metric

* Other

  Soft TF-IDF .8, .8 with Jaro-Winkler .8 base metric

We have added this automatic metric selection step to our approach. The results for this are shown in tables 2.1, 2.2 and 2.3 under StringsAuto.[7] This fully-automated string-based approach to ontology alignment performs loses little performance when compared to the optimal version.

2.5 String Similarity Survey Summary

For some types of ontologies, the performance of different string similarity metrics varies greatly in terms of both precision and recall. It is important to be cognizant of this when selecting a string metric for a particular use. This chapter has established guidelines to assist researchers in making this selection. In addition, we have found that many string preprocessing strategies commonly used, such as stop word removal and word stemming are in many cases unhelpful and in some cases counter-productive. We have presented data on which preprocessing strategies are useful in particular situations. In addition, we have developed a basic system to automatically select an appropriate string similarity metric for a given pair of ontologies at runtime. Because nearly all ontology alignment algorithms make use of string similarity metrics, this work can similarly be integrated into other existing alignment algorithms and is therefore directly relevant to many researchers in this field. A conference paper based on this work has been published [Cheatham and Hitzler 2013].

There are several paths for future work based on the idea of pushing string similarity metrics as far as they can go in terms of ontology alignment. An important first step is to develop a string similarity metric that performs better on properties. This will be the focus of the remainder of this dissertation. However, in order to evaluate a property-centric string similarity metric we must use the OAEI Conference test set, as that is the only widely-accepted benchmark that includes matches between properties. There are some concerns related to the reference alignments in that benchmark though. The next chapter will explain these concerns and propose a modified version of the reference

[7] The results for StringsAuto on the Multifarm track differ from what is reported on the 2013 OAEI website because the workshop organizers were not able to connect to Google Translate when running their tests

alignments. Chapter 4 will then outline a new string-based approach to property alignment and evaluate its performance based on both the current and revised versions of the benchmark.

Metric	Precision	Recall	F-measure
YAM++	0.80	0.69	0.74
AML-bk	0.87	0.58	0.70
LogMap	0.80	0.59	0.68
AML	0.87	0.56	0.68
StringsOpt	**0.85**	**0.55**	**0.67**
ODGOMS1_2	0.74	0.60	0.66
StringsAuto	**0.78**	**0.54**	**0.64**
ServOMap	0.73	0.55	0.63
MapSSS	0.83	0.50	0.62
WeSeE-Match	0.85	0.47	0.61
ODGOMS1_1	0.76	0.51	0.61
HerTUDA	0.74	0.50	0.60
edna	0.76	0.49	0.60
WikiMatch	0.73	0.49	0.59
LogMapLite	0.73	0.50	0.59
IAMA	0.78	0.48	0.59
HotMatch	0.71	0.51	0.59
XMapSiG1_3	0.72	0.48	0.58
OntoK	0.77	0.47	0.58
CIDER_CL	0.75	0.47	0.58
XMapGen1_4	0.68	0.49	0.57
SYNTHESIS	0.77	0.45	0.57
StringEquiv	0.80	0.43	0.56
RiMOM2013	0.59	0.49	0.54
XMapSiG1_4	0.80	0.40	0.53
XMapGen	0.58	0.45	0.51
CroMatcher	0.52	0.50	0.51
MaasMatch	0.28	0.55	0.37

Table 2.1: Results of the StringsOpt and StringsAuto alignment algorithms together with the competitors from the OAEI 2013 competition on the Conference test set

Metric	Prec (dif)	Fms (dif)	Rec (dif)	Prec (same)	Fms (same)	Rec (same)
CIDER_CL	0.03	0.03	0.04	0.18	0.06	0.04
MapSSS	0.27	0.10	0.07	0.50	0.06	0.03
RiMOM2013	0.52	0.21	0.13	0.87	0.14	0.08
WeSeE-Match	0.22	0.15	0.12	0.56	0.16	0.09
WikiMatch	0.34	0.27	0.23	0.65	0.18	0.11
StringsAuto	**0.64**	**0.39**	**0.28**	**0.93**	**0.26**	**0.15**
StringsOpt	**0.58**	**0.40**	**0.31**	**0.90**	**0.38**	**0.24**
YAM++	0.51	0.40	0.36	0.91	0.60	0.50

Table 2.2: Results of the StringsOpt and StringsAuto alignment algorithms together with the competitors from the OAEI 2013 competition on the Multifarm test set

Metric	Precision	Recall	F-measure
AML-bk	0.95	0.93	0.94
GOMMA-bk	0.92	0.93	0.92
YAM++	0.94	0.87	0.91
AML	0.95	0.83	0.89
LogMap	0.92	0.85	0.88
StringsOpt	**0.88**	**0.87**	**0.88**
GOMMA	0.96	0.80	0.87
StringsAuto	**0.90**	**0.78**	**0.84**
LogMapLite	0.96	0.73	0.83
MapSSS	0.90	0.77	0.83
ODGOMS	0.98	0.71	0.82
WikiMatch	0.99	0.67	0.80
HotMatch	0.98	0.64	0.77
StringEquiv	1.00	0.62	0.77
XMapSig	0.86	0.67	0.75
ServOMap	0.96	0.62	0.75
XMapGen	0.81	0.70	0.75
IAMA	1.00	0.56	0.71
CIDER_CL	0.65	0.73	0.69
HerTUDA	0.69	0.67	0.68
WeSeE-Match	0.62	0.38	0.47
MaasMatch	0.36	0.48	0.41

Table 2.3: Results of the StringsOpt and StringsAuto alignment algorithms together with the competitors from the OAEI 2013 competition on the Anatomy test set

Test set	Words per label
Conference	1.85
CIDX-to-Excel	1.57
Multifarm	2.24
CIDX-to-Excel (ML)	1.61
Anatomy	2.64
Genes	4.11

Table 2.4: Comparison of datasets based on word length and number of words per label

3

Property Alignment Benchmarks

The Ontology Alignment Evaluation Initiative (OAEI) is now a decade old, and it has been extremely successful by many different measures: participation, accuracy, and the variety of problems handled by alignment systems have all increased, while runtimes have decreased [Euzenat et al. 2011]. The OAEI benchmarks have become *the* standard for evaluating general-purpose (and in some cases domain-specific or problem-specific) alignment systems. In fact, you would be hard-pressed to find a publication on an ontology alignment system in the last ten years that *didn't* use these benchmarks. They allow researchers to measure their system's performance on different types of matching problems in a way that is considered valid by most reviewers for publication. They also enable comparison of a new system's performance to that of other alignment systems without the need to obtain and run the other systems.

As of 2013 the OAEI has eight different tracks, each designed to exercise a different aspect of alignment systems.

- **Benchmark:** a synthetic benchmark to test general-purpose schema alignment by taking a root ontology and systematically modifying it (e.g. by removing hierarchy information, misspelling labels, etc.)
- **Anatomy:** tests alignment of two relatively small biomedical ontologies with a high degree of subject overlap
- **Conference:** tests general-purpose schema alignment on a group of small real-world ontologies describing the domain of conference organization
- **Multifarm:** a multi-lingual version of the Conference test set
- **Library:** tests alignments of two large real-world taxonomies related to the social science domain

- **Interactive:** same as the Conference track but allows the alignment system to request input from a user during the matching process
- **Large Biomedical:** tests the scalability of alignment systems by requiring alignment of three biomedical ontologies: the Foundational Model of Anatomy (FMA), SNOMED CT, and the National Cancer Institute Thesaurus (NCI), each of which contains tens of thousands of classes
- **Instance Matching:** a synthetic test set similar to the Benchmark track but focused on mapping instance data rather than schemas

While these tracks do test alignment systems in a range of contexts in which they might be used, there is not complete coverage of the current pressing issues in the field of ontology and linked data alignment (as described in Chapter 1). In addition, the details of some of the test sets have led to the incorporation of behaviors in many alignment systems that may not be optimal. For instance, in several OAEI tracks an entity can be involved in at most one match, which may not be realistic for some real-world datasets. Similarly, entities are only matched to other entities of the same type in some tracks. It could be argued that this is not realistic in many cases, particularly when the decision of when to represent something as an instance versus a class is not always clear cut.

Benchmarks in any field must be carefully chosen and revisited from time to time in order to make sure they are pushing the field forwards rather than constraining innovation. Interesting position papers on the qualities of a good benchmark include "Good Benchmarks are Hard to Find" [Dekhtyar and Hayes 2006], "The Art of Building a Good Benchmark" [Huppler 2009], and "Using Benchmarking to Advance Research" [Sim et al. 2003]. We believe that now is a good time to revisit the most popular existing alignment benchmarks and to develop new benchmarks to drive innovation in the field. In this chapter we begin this process by focusing on benchmarks related to property alignment.

3.1 The OAEI Conference Track

The OAEI Conference track is the only non-synthetic test set for alignment systems that has reference alignments containing matches between properties as well as classes. We therefore plan to use it to evaluate our string-based property alignment system. However, we have some concerns related to the reference alignments for this test set. In particular, we wonder about the ramifications on ontology alignment system evaluation of the rather strong claims made by the reference alignments within the Conference track, in terms of both the number of matches and the absolute certainty of each match. Other researchers have also expressed some concern related to this test set. In particular,

Ontology	Classes	Object Props	Data Props	Individuals
cmt	30	49	10	0
conference	60	46	18	0
confOf	39	13	23	0
edas	104	30	20	114
ekaw	74	33	0	0
iasted	141	38	3	4
sigkdd	50	17	11	0
median	60	33	11	0
total	558	259	96	118

Table 3.1: Characteristics of the ontologies in the OAEI Conference track

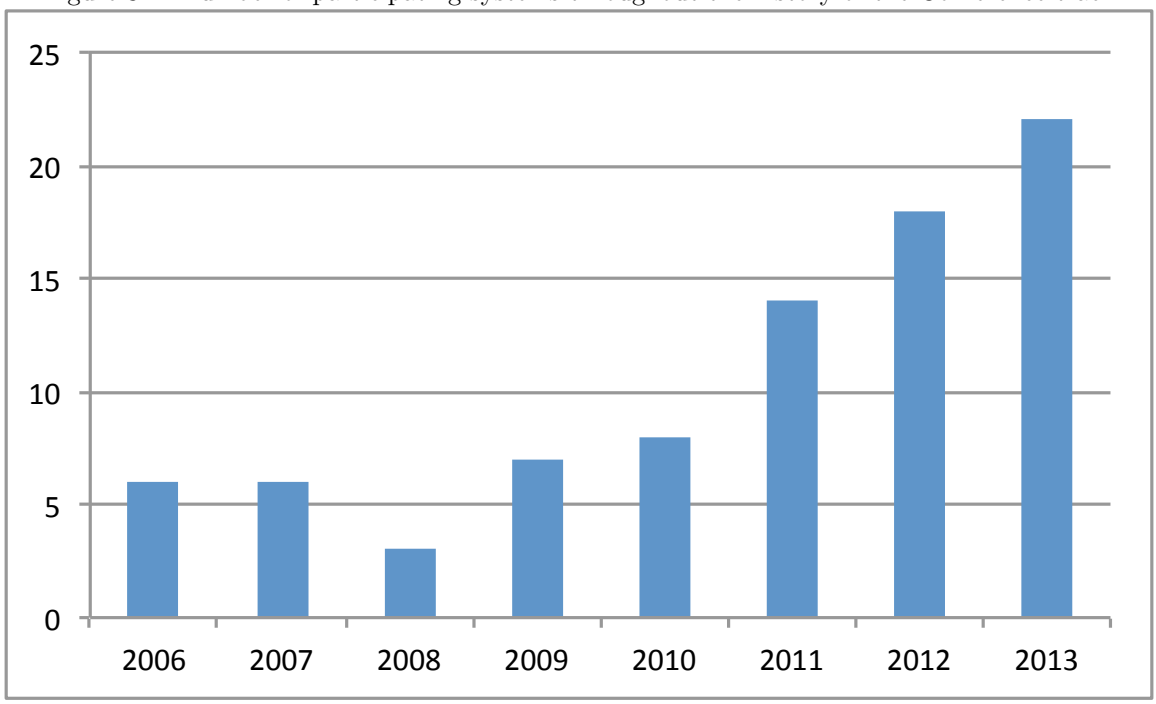
the developers of the CrowdMap alignment system indicated that some of the mappings from the reference alignments seemed suspect, including WelcomeTalk = Welcome_address, SocialEvent = Social_program and Attendee = Delegate (from the edas-iasted test case) [Sarasua et al. 2012].

As mentioned in Chapter 2, the complete Conference track consists of 16 ontologies covering the domain of conference organization. These ontologies were created to reflect the material on conference websites, software tools used for organizing conferences, and the knowledge of people involved in conference administration. Alignment systems are intended to generate alignments between each pair of ontologies, for a total of 120 alignments. Each system's output is evaluated against reference alignments in terms of precision, recall, and f-measure. Reference alignments are available for all pairwise combinations of a subset of seven of the ontologies (resulting in 21 reference alignments). The intent of the track is to provide real-world matching problems over ontologies covering the same domain. More detail about the track can be found at the OAEI website: http://oaei.ontologymatching.org

Table 3.1 shows the number of different types of entities in each of the seven ontologies for which reference alignments are available. While the number of properties is on par with the number of classes in these ontologies (355 versus 558), the reference alignments contain significantly fewer matches between properties than between classes (46 versus 259).

The ontologies that comprise the Conference track were developed in 2005 as part of the OntoFarm project [Šváb et al. 2005]. As explained in [Euzenat et al. 2011], the Conference track, together with the Anatomy track, was introduced to provide more realism and difficulty than that offered by the synthetically-generated Benchmark track. The history of the Conference track can be gleaned from the OAEI website. The track has been a part of every OAEI since 2006. For the first two years, reference alignments were unavailable and so alignments were evaluated using a combination of man-

Figure 3.1: Number of participating systems throughout the history of the Conference track



ual labeling by a group of experts (where each match was marked correct, incorrect, or unclear), data mining and logical reasoning techniques. Interesting or unclear matches were discussed in "Consensus Workshops." In 2008 the track organizers created a reference alignment for all possible pairs of five of the Conference ontologies. The reference alignments were based on the majority opinion of three evaluators and were discussed during the consensus workshop that year. The confidence value for all mappings in the reference alignments is 1.0. By 2009 the reference alignments contained all pairs for seven ontologies and the consensus workshop had been phased out. Additionally, as the number of participating systems grew (see Figure 3.1), the manual labeling was scaled back from one of correct, incorrect, or unclear to simply correct or incorrect. Further, this labeling was performed on the 100 matches in which the alignment systems had the highest confidence. By 2011 manual labeling was eliminated entirely and evaluation relied completely on the reference alignments and logical coherence. Each step in this history, while understandable due to the increasing number of participating systems, resulted in a loss of nuance in evaluation.

3.1.1 Expert Consensus

Today the reference alignments for the Conference track are being used to report precision and recall values for nearly all ontology alignment systems being developed. As was discussed in Chapter 1 (and

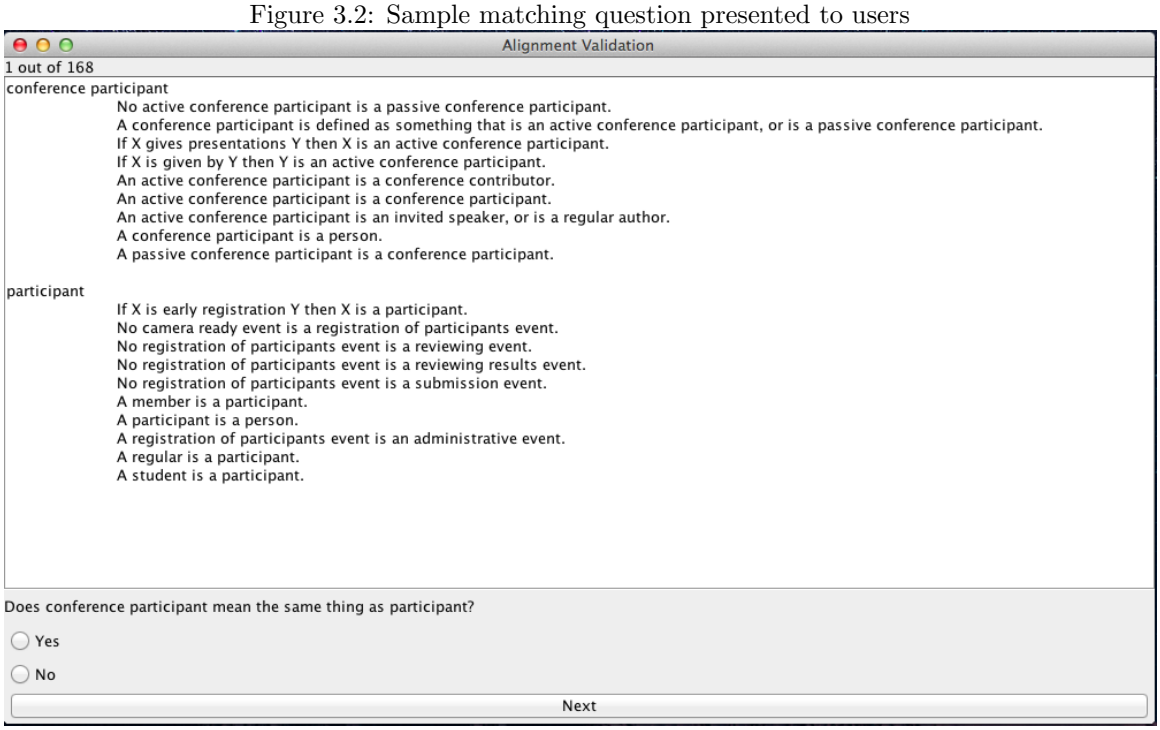
shown in Figure 1.5), performance has improved significantly over the existence of the track. Also, none of the matches in the reference alignments have been questioned in any of the ontology matching workshop papers submitted by tool developers from 2006 through 2013, and in the last three years of the ontology matching workshop none of the matches have come up for debate. However, it should be noted that these alignments were developed by just three individuals (with support from the consensus workshops). We wanted to determine the degree of consensus on these reference alignments from a group of experts. Initially we collected all of the matches in the reference alignments together with any match that was produced by at least one alignment system that competed in the 2013 OAEI. This resulted in 757 matches. We asked a group of people familiar with both ontologies and academic conferences to indicate whether or not they agreed with each match. The experts politely refused to opine on so many matches. In order to prune the question set, we adopted the approach described in [Sarasua et al. 2012] by attempting to verify the reference alignments using the consensus of existing alignment systems as a filter. In our case the alignment systems we consulted were the 2013 OAEI competitors that performed better than the baseline string similarity metric edna (edit distance). There were 15 such systems, which is a much larger sample than was used for the filtering step in [Sarasua et al. 2012]. We considered those matches in the reference alignments that at least one of the qualifying alignment systems disagreed on. This resulted in 168 matches that were presented to the experts for validation. The 141 matches that all of the alignment systems agreed upon were simple string equivalences. In fact, the Conference track seems quite challenging for current alignment systems, most of which are unable to identify the large majority of matches in the reference alignments that do not involve equivalent or nearly-equivalent strings. Additionally, there does not seem to be evidence of widespread overfitting despite the reference alignments being made available over five years ago. This is similar to the lack of overfitting discovered in an analysis of results on the Benchmark track after it had been available for a similar amount of time [Rosoiu et al. 2011], and encouraging for the field of ontology alignment.

The experts were given a link to download a Java program and accompanying data files. See Figure 3.2 for a screenshot of the program during execution. Note that the entity labels from each match were stripped of the URL, tokenized, and put into lower case. Additionally, in order to provide the experts with some context for the labels, all of the axioms in the ontologies were translated to English using Open University's SWAT Natural Language Tools.[1] Any axioms related to either of the entities in the match were displayed to the users. Users were then asked a question of the form "Does labelA mean the same thing as labelB?" and prompted to choose a yes or no answer.

We received input from 13 experts. Using a majority rules approach, the experts agreed with

[1] http://swat.open.ac.uk/tools/

Figure 3.2: Sample matching question presented to users



106 of the 168 matches. Assuming that the experts would also have concurred on all of the 141 matches that were not asked about because all of the alignment systems agreed upon them and that they would not have identified any additional mappings not in the reference alignments, their precision is 1.0. The second part of this assumption is admittedly more of a leap, but is somewhat reasonable considering that researchers are encouraged to bring matches that they believe are mistakenly categorized as false positives to the attention of the conference organizers, but no one has done so for over three years. Under these assumptions, the expert recall is 0.80 and their f-measure is 0.89. The f-measure of the individual experts ranges from 0.78 to 0.95 when computed against the OAEI reference alignment. This compares to an f-measure of 0.74 for the top-performing automated alignment system in 2013, while the median of this group of systems was 0.64.

One of the main things that stands out from the results of this experiment is the lack of consensus among the experts on these matches. For each match, we consider the *certainty* of our expert group as the difference between the percentage of people who answered "yes" and the percentage who answered "no." The average certainty over all matches was 43%, with a variance of 9%. There was total agreement on just 9 matches, while the experts were split 7-6 or 6-7 on 40 matches. Further, 6 of the 9 matches with complete consensus were exact or near lexical matches that were missed by one or more of the alignment systems for some reason (see Table 3.2). The experts deemed all of these matches to be valid – there were no cases in which the experts unanimously disagreed with a

Table 3.2: Matches on which all experts agreed

Entity 1	Entity 2	Test Name
email	E-mail	cmt-sigkdd
has_an_email	hasEmail	conference-confOf
hasSurname	hasLastName	confOf-edas
has_a_review	hasReview	conference-ekaw
hasAuthor	writtenBy	cmt-confOf
hasFirstName	hasFirstName	confOf-edas
has_the_last_name	hasLastName	conference-edas
CoffeeBreak	Coffee_break	edas-iasted
isReviewing	reviewerOfPaper	edas-ekaw

match.

It is also instructive to consider the degree of consensus on matches involving properties versus those involving classes. This information is shown in Table 3.3. Recall that to avoid requiring the experts to opine on an inordinate number of matches, the questions asked were limited to matches from the reference alignment on which at least one reasonably-performing alignment system disagreed. This was the case for *all* property matches, but for only about half of the class matches. The top half of Table 3.3 shows the results for only those matches that the experts were asked about, while the bottom half shows the same data for the complete reference alignments under the assumption that the experts would not have disagreed with any of the uncontroversial matches. We see that there was significantly more agreement among the experts on the property matches they were asked about than on the class matches. This is further indication that, while current alignment systems are reasonably proficient at aligning classes, many are stymied by rather straightforward matches between properties. Considering the reference alignments in their entirety, there was little difference between the average conference values for classes and properties.

3.1.2 Conference v2.0

In 2011 the developers of MapPSO pointed out that in the reference alignment for the Benchmark track (a separate test set offered alongside the Conference track) there were two matches resulting from the synthetic test set generation process that could not possibly be detected unequivocally from an information theoretic perspective. They argue that since neither humans nor machines could resolve these mappings, the confidence should be set at 50% for each [Bock et al. 2011]. We

Table 3.3: Expert consensus on classes versus properties

Controversial matches	
Average class confidence	0.54
Average property confidence	0.75
Classes with total confidence	1 (1%)
Properties with total confidence	8 (17%)
Classes with total confusion	35 (29%)
Properties with total confusion	5 (11%)
All matches	
Average class confidence	0.78
Average property confidence	0.75
Classes with total confidence	138 (53%)
Properties with total confidence	8 (17%)
Classes with total confusion	35 (14%)
Properties with total confusion	5 (11%)

claim that our results on the experiment discussed in the previous section show that a similar issue is occurring with the Conference track. It is less than ideal to evaluate automated alignment systems against a reference alignment with confidence values for all matches equal to 1.0 when the degree of consensus among human experts is actually quite different. Therefore, we have established another version of the Conference track reference alignments which has confidence values that reflect the percentage of agreement for each match among our group of experts. This alignment is available in the Alignment API format from http://www.michellecheatham.com/files/ConferenceV2.zip.

The first six columns of Table 3.4 show the results of the 2013 alignment systems that performed better than the string edit distance baseline on both the original (v1) and our revised (v2) versions of this benchmark. These columns show the traditional precision, recall, and f-measure metrics. In this evaluation approach, matches in the new version of the benchmark with a confidence of 0.5 or greater are considered fully correct and those with a confidence less than 0.5 are considered completely invalid. Thresholds for the matchers' results were set at a value that optimized f-measure for each system, in accordance with the evaluation procedure used by the OAEI. A hypothetical alignment system that perfectly agreed with the current version of the Conference track reference alignments would have a precision of 0.8 and a recall of 1.0 on version 2.0, yielding an f-measure of 0.89. All of the qualifying 2013 alignment systems saw an increase in traditional f-measure. In fact,

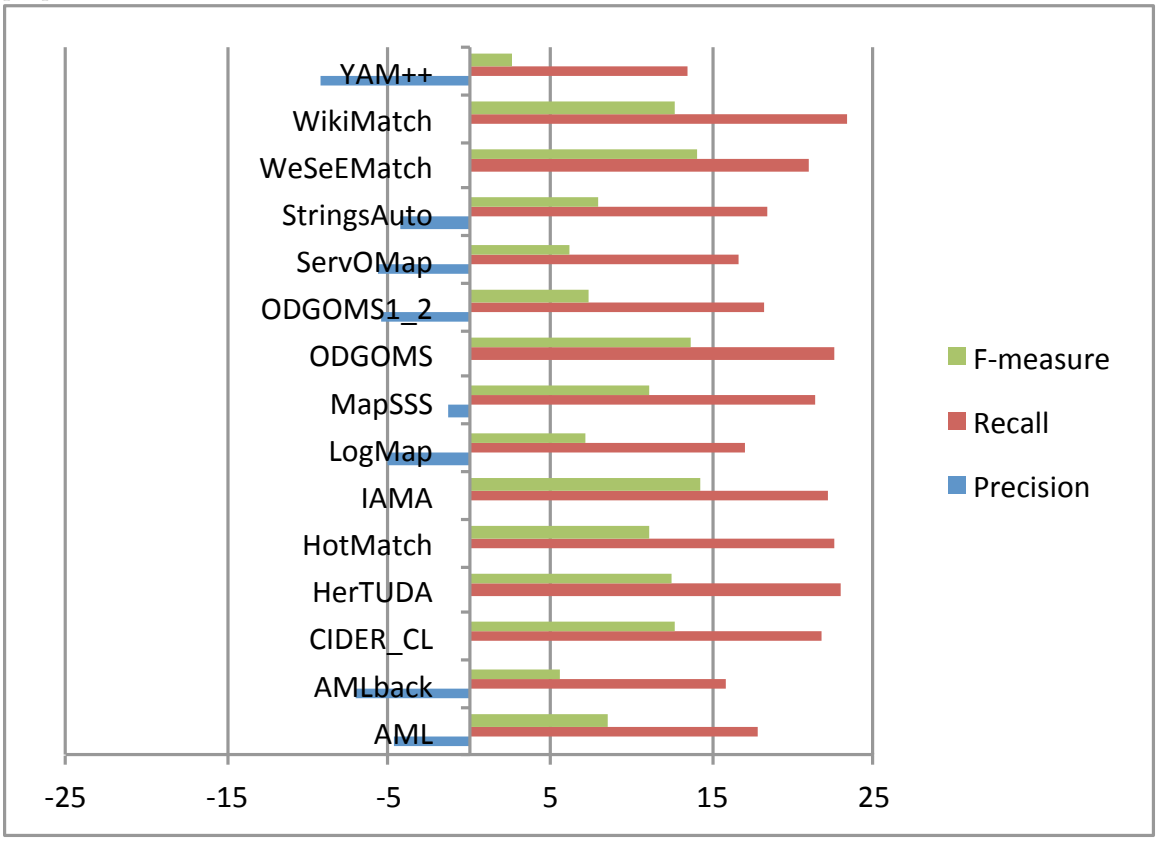
Table 3.4: Results of qualifying 2013 OAEI alignment systems on the traditional and proposed revision of the Conference track

System	Pr_{v1}	Rec_{v1}	Fm_{v1}	Pr_{v2}	Rec_{v2}	Fm_{v2}	Pr_{cont}	Rec_{cont}	Fm_{cont}
AML	0.87	0.56	0.68	0.83	0.67	0.74	0.88	0.65	0.75
AMLback	0.87	0.58	0.70	0.81	0.68	0.74	0.88	0.68	0.76
CIDER_CL	0.74	0.49	0.59	0.74	0.61	0.67	0.75	0.60	0.67
HerTUDA	0.74	0.50	0.60	0.74	0.63	0.68	0.75	0.66	0.70
HotMatch	0.71	0.51	0.60	0.71	0.64	0.67	0.71	0.66	0.68
IAMA	0.78	0.48	0.59	0.78	0.60	0.68	0.78	0.64	0.70
LogMap	0.80	0.59	0.68	0.76	0.70	0.73	0.83	0.56	0.67
MapSSS	0.74	0.50	0.60	0.73	0.62	0.67	0.72	0.64	0.68
ODGOMS	0.76	0.51	0.61	0.76	0.64	0.70	0.78	0.67	0.72
ODGOMS1_2	0.74	0.60	0.66	0.70	0.72	0.71	0.71	0.73	0.72
ServOMap	0.72	0.55	0.63	0.68	0.65	0.67	0.71	0.67	0.69
StringsAuto	0.71	0.54	0.61	0.68	0.65	0.66	0.67	0.67	0.67
WeSeEMatch	0.85	0.47	0.60	0.85	0.58	0.69	0.84	0.61	0.70
WikiMatch	0.73	0.49	0.59	0.73	0.62	0.67	0.73	0.65	0.69
YAM++	0.80	0.69	0.74	0.73	0.79	0.76	0.80	0.54	0.65

six systems saw a double-digit percentage improvement. In most cases precision remained constant or dropped slightly while recall increased significantly (see Figure 3.3). This is expected because no new matches were added to the reference alignments, but those that the experts did not agree on were removed. If we rank the systems in terms of f-measure, we see that the top five systems remain consistent across both versions. Also interesting to note, the rank of StringsAuto, the authors' own automated alignment system (see Chapter 2), dropped from the middle of the pack to next-to-last when evaluated under this version of the benchmark. This was by far the largest drop in rank of any system. The relative success of this approach on the existing version of the Conference track may indicate a bias towards exact or near-exact lexical matches in the benchmark.

Intuitively, it seems desirable to penalize an alignment system more if it fails to identify a match on which 90% of the experts agree than one on which only 51% of them agree. To do this, we evaluate the same group of 2013 systems based on modified precision and recall metrics that consider the confidence values of the matches, i.e., precision and recall metrics which are continuous versions of the traditional, discrete ones. Let us briefly reflect on how to do this. In order to follow the intuition

Figure 3.3: Percent difference in traditional precision, recall, and f-measure between the current and proposed revision of the Conference track



of the discrete (Boolean, two-valued) case, we would like to retain the usual definitions of precision, recall, and f-measure in terms of the numbers of *true positives* (tp), *false positives* (fp), and *false negatives* (fn), which are as follows.

$$\text{Precision} = \frac{\mathsf{tp}}{\mathsf{tp} + \mathsf{fp}}$$

$$\text{Recall} = \frac{\mathsf{tp}}{\mathsf{tp} + \mathsf{fn}}$$

$$\text{F-measure} = \frac{2 \cdot \mathsf{tp}}{2 \cdot \mathsf{tp} + \mathsf{fp} + \mathsf{fn}} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

It remains to obtain tp, fp, and fn for the case where both the benchmark and the results of the system to be evaluated are expressed in terms of confidence values for each alignment.

Given a potential match i (say, between "conference participant" and "participant"), let $b(i) \in [0, 1]$ denote the confidence value assigned to this match by the benchmark, and let $s(i) \in [0, 1]$ denote the confidence value assigned to this match by the system to be evaluated. Interpreting $b(i)$ and $s(i)$ as certainty values in the sense of fuzzy set theory [Nguyen and Walker 2005] – which is

reasonable from our perspective – we thus arrive at the formula

$$\mathsf{tp} = \sum_{i \in I} T(b(i), s(i)),$$

where T is some t-norm, i.e., a continuous-valued version of logical conjunction. The most obvious choices for the t-norm are arguably the product t-norm and the Gödel (or minimum) t-norm – it actually turns out that there is not much difference between these two with respect to our analysis. In fact the effect is within rounding error in most cases and maximally 3% (resulting in, e.g., f-measure of .65 rather than .67). In the following we will thus stick with the product t-norm.[2]

From this perspective, we thus arrive at the following.

$$\begin{aligned}
\mathsf{tp} &= \sum_{i \in I} b(i) \cdot s(i) \\
\mathsf{fp} &= \sum_{i \in \{j \in I | b(j) < s(j)\}} |b(i) - s(i)| \\
\mathsf{fn} &= \sum_{i \in \{j \in I | b(j) > s(j)\}} |b(i) - s(i)|
\end{aligned}$$

Note that all three revert to their original definition in a discrete (Boolean) setting in which only confidence values of 0 and 1 are used.

With these definitions, we thus obtain the following.

$$\begin{aligned}
\text{Precision} &= \frac{\mathsf{tp}}{\mathsf{tp} + \mathsf{fp}} = \frac{\sum_{i \in I} b(i) \cdot s(i)}{\sum_{i \in I} b(i) \cdot s(i) + \sum_{i \in \{j \in I | b(j) < s(j)\}} |b(i) - s(i)|} \\
\text{Recall} &= \frac{\mathsf{tp}}{\mathsf{tp} + \mathsf{fn}} = \frac{\sum_{i \in I} b(i) \cdot s(i)}{\sum_{i \in I} b(i) \cdot s(i) + \sum_{i \in \{j \in I | b(j) > s(j)\}} |b(i) - s(i)|} \\
\text{F-measure} &= \frac{2 \cdot \mathsf{tp}}{2 \cdot \mathsf{tp} + \mathsf{fp} + \mathsf{fn}} = \frac{2 \cdot \sum_{i \in I} b(i) \cdot s(i)}{2 \cdot \sum_{i \in I} b(i) \cdot s(i) + \sum_{i \in I} |b(i) - s(i)|}
\end{aligned}$$

Note that the f-measure is also rather intuitive: It is the sum $\sum_{i \in I} |b(i) - s(i)|$ of all differences in confidence, normalized (using tp) to a value between 0 and 1. The value for $(\mathsf{fp} + \mathsf{fn})$ is captured in this sum of differences.

A Java class that computes these metrics is included with the downloadable version of the reference alignments, together with a small driver program illustrating its use.

The last three columns of Table 3.4 show the results of the alignment systems when evaluated with these metrics. The continuous precision for most systems was slightly higher than that of the traditional precision metric on Conference v2. The average increase was about 3%. The continuous recall measures were also slightly higher (generally 3-5%) than the traditional version. Half of the alignment systems evaluated here created alignments that consisted entirely or predominantly of

[2]Note that the product t-norm also lends itself to a probabilistic interpretation.

matches with a confidence at or very near 1.0. If confidence values were stressed more as part of the alignment system evaluation, we would likely see larger differences between the continuous and discrete (traditional) precision and recall measures.

An interesting side note is that this method of evaluation does not involve setting any thresholds, either for the reference alignment or the matching systems. We argue that this is an improvement because it eliminates the need to artificially discretize a similarity assessment that is inherently continuous. It also considerably speeds up the evaluation process.

The performance of two systems in particular looks very different when these confidence-conscious versions of precision and recall are used to evaluate them. LogMap and YAM++ move from the top three to the bottom three systems when ranked by f-measure. These systems assign relatively low confidence values (e.g. 0.5-0.75) for many matches even when the labels of the entities involved are identical, which apparently does not correspond well to human evaluation of the match quality.

3.1.3 Crowdsourcing Alignment Benchmarks

While it is clearly valuable to have ontology alignment benchmarks that reflect the consensus opinions of a large number of experts, it is very difficult to persuade such experts to take the time necessary to create the required reference alignments. What if we could leverage the so-called "Wisdom of Crowds" for this task instead? We have investigated the use of Amazon's Mechanical Turk webservice for this purpose.

Amazon publicly released Mechanical Turk in 2005. It is named for a famous chess-playing "automaton" from the 1700s. The automaton actually concealed a person inside who manipulated magnets to move the chess pieces. Similarly, Amazon's Mechanical Turk is based on the idea that some tasks remain very difficult for computers but are easily solved by humans. Mechanical Turk therefore provides a way to submit these types of problems, either through a web interface or programmatically using a variety of programming languages, to Amazon's servers, where anyone with an account can solve the problem. In general, this person is compensated with a small sum of money, often just a cent or two. The solution can then be easily retrieved for further processing, again either manually or programmatically. While there are few restrictions on the type of problems that can be submitted to Mechanical Turk, they tend towards relatively simple tasks such as identifying the subject of an image, retrieving the contents of receipts, business cards, old books, or other documents that are challenging for OCR software, transcribing the contents of audio recordings, etc. As of 2010, 47% of Mechanical Turk workers, called "Turkers", were from the United States while 34% were from India. Most are relatively young (born after 1980), female, and have a Bachelors degree [Ipeirotis 2010]. It is possible for individuals asking questions via Mechanical Turk (called

Requesters) to impose qualifications on the Turkers who answer them. For instance, Requesters can specify that a person lives in a particular geographic area, has answered a given number of previous questions, has had a given percentage of their previous answers judged to be of high quality, or pass a test provided by the Requester. In addition, Requesters have the option to refuse to pay a Turker if they judge the Turker's answers to be of poor quality.

We used Mechanical Turk to ask 40 individuals their opinion on the same 168 matches presented to the group of experts. Each question was formatted in the same way as Figure 3.2, with the exception of the Next button. The questions were presented in 21 batches with 8 questions per batch. Respondents earned 16 cents for each batch and were paid regardless of the specific answers they gave. No qualifications were placed on who could work on the tasks.

We created alignments for the pairs of ontologies in the Conference track based on the results from the 40 Turkers. The confidence of each match was set to the percentage of Turkers who indicated the match was valid. These alignments were then evaluated against both the current and proposed revisions of the reference alignments. The results are shown in Table 3.5. The first line in the table shows that the recall is somewhat low on the current version of the Conference track. This is arguably an indication that the current version attempts to map too much. Remember from Section 3.1.2 that the performance of the experts, when taken as a group, was nearly identical (their precision was 1.0 and their recall was 0.80, yielding an f-measure of 0.89). Though further experimentation is necessary for confirmation, these results support the hypothesis that using Mechanical Turk to validate existing reference alignments yields essentially the same results as those produced by experts. Moreover, the third row in Table 3.5 indicates that the Turkers don't just agree with the experts in a binary context – the degree of consensus among them also closely corresponded to that of the experts, resulting in very similar confidence values. These results are quite encouraging – for $134.40 we generated a high-quality reference alignment in less than two days (over Easter weekend, no less). However, they may be somewhat overly optimistic, because the results were calculated on the reference alignments in their entirety, but 141 of the 309 matches in those alignments were trivial and therefore not included in our survey. If we compute the same metrics but restrict them to the subset of matches on which the Turkers and experts were surveyed, we arrive at the values in the last row of Table 3.5. These results are still quite strong, and we feel that this is a viable method of benchmark generation. This belief is supported by the fact that when the performance of the top alignment systems from the 2013 OAEI on the expert-generated reference alignments is compared to what it would be if the reference alignments were instead based solely on the results from the Turkers, there is little practical difference between the two. None of the continuous precision, recall, or f-measures differs by more than 0.02, and the vast majority are within 0.01.

Table 3.5: Performance of the Mechanical Turk-generated alignments on the traditional and proposed revision of the Conference track

Test Version	Prec.	Recall	F-meas.
Conference v1	1.00	0.81	0.90
Conference v2 (discrete)	0.88	0.89	0.88
Conference v2 (continuous)	0.98	0.96	0.97
Conference v2 subset (continuous)	0.94	0.88	0.91

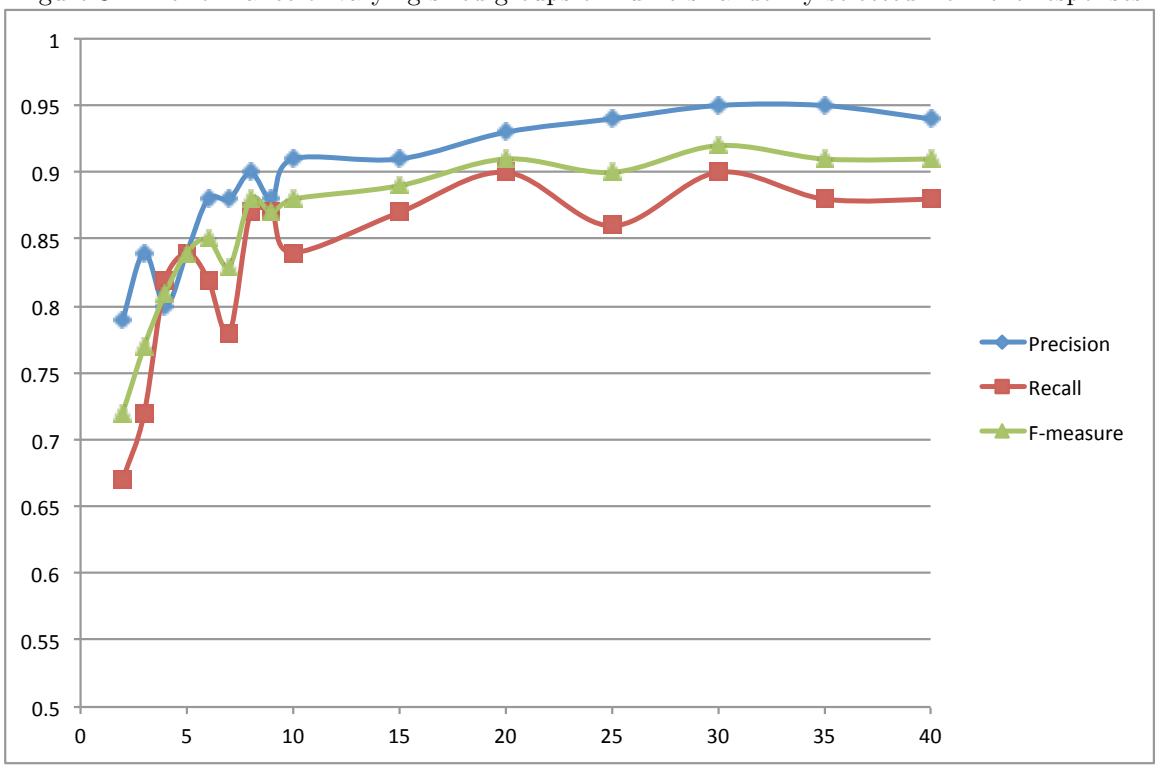
Other researchers have mentioned a problem with spammers on Mechanical Turk, who will answer questions randomly or with some other time-saving strategy in order to maximize their profit-to-effort ratio [Sarasua et al. 2012]. While we did not have this issue during our experiments, it might be possible to further optimize the crowdsourcing of reference alignments by reducing the number of Turkers recruited for the effort. It stands to reason that the fewer inputs that are collected, the higher quality each one needs to be in order to reap reasonable results. Amazon's Mechanical Turk Requester Best Practices Guide[3] suggests several potential ways to find high-quality Turkers, including using qualification tests or "golden questions." In an effort to identify high-performing individuals, we implemented the golden question approach, in which a Turker's answers are validated against a set of questions for which the answers are obvious. In this case, there were nine questions on which all of the experts agreed. There were 10 Turkers who agreed on either eight or nine of these golden questions. We call these respondents "Super Turkers." We created alignments using only the results of these Super Turkers and evaluated them with respect to the expert-generated reference alignments. If we evaluate their results over the whole of the Conference v2 reference alignments, we arrive at essentially the same result we achieved using the 40 regular Turkers. However, if we evaluate the Super Turker results over the subset of unclear matches, the performance is slightly worse that that of the entire group. Actually, it is roughly the same as the performance of a sample of the same size drawn randomly (see Figure 3.4, which shows the continuous precision, recall, and f-measure for varying numbers of randomly selected Turkers). So it does seem that the wisdom lies in the crowd rather than a few individuals in this instance.

The Java code to interact with Mechanical Turk and generate the reference alignments is available at http://www.michellecheatham.com/files/MTurk.zip. The program can be run from the command line and requires the following input:

- The ontologies to be aligned, in OWL or RDF format.

[3]http://mturkpublic.s3.amazonaws.com/docs/MTURK_BP.pdf

Figure 3.4: Performance of varying-sized groups of Turkers randomly selected from the responses



- A text file specifying the particular matches to be verified. One option would be to use one or more automated alignment algorithms to arrive at a set of possibilities.
- A text file containing the English translations of all of the axioms in both ontologies. This can be produced using the tool at http://swat.open.ac.uk/tools/.
- Two Mechanical Turk properties files containing information such as a Requester access key, the payment amount per question, and any qualifications required for Turkers to accept the assignments.

A Mechanical Turk Requester account with sufficient funds is required to submit questions to Amazon. There is a sandbox available from Amazon to test the assignments before submitting them.

There has been some related work involving using crowdsourcing for tasks related to ontologies that should be noted here. A group of researchers from Stanford University has recently published several papers on using Mechanical Turk to verify relationships within biomedical ontologies [Mortensen et al. 2013a; Noy et al. 2013; Mortensen et al. 2013b; Mortensen 2013]. This is clearly closely related to the work presented in this section, though our focus on generating reference alignments between pairs of ontologies and the potentially more "approachable" domain of conference

organization caused us to have slightly different experiences. In particular, when relationships to be verified come from separate ontologies rather than from within a single one, ontology design decisions can confuse this issue. Also, precise vocabulary such as that found in biomedical ontologies is less subject to different interpretations. The end result was that we did not need to qualify the Turkers who worked on our tasks in order to obtain good results as the group from Stanford did, but it was harder to judge the accuracy of the crowdsourced results due to the lack of strong consensus among both experts and Turkers.

There is also an alignment system called CrowdMap that uses Mechanical Turk to generate alignments between two ontologies [Sarasua et al. 2012]. The focus in that work is on generating alignments from scratch, which are then evaluated against the existing OAEI benchmarks (including the Conference track). While that is a topic we are interested in as well, we view the work presented here as complementary since our current goal is to establish a new version of the Conference track that more accurately reflects expert opinion.

There has also been research into using crowdsourcing in other contexts that bear some similarity to ontology alignment, such as natural language processing, information retrieval, and audio processing [Wichmann et al. 2011; Ul Hassan et al. 2012].

3.2 DBPedia and YAGO

In addition to the OAEI Conference test set, we would also like to analyze the performance of our proposed string-based property alignment system on another real-world alignment task. For this we have chosen DBPedia and YAGO. DBPedia is a linked data version of the information in Wikipedia. The dataset is currently on version 3.9 and can be downloaded at http://wiki.dbpedia.org/Downloads39. The YAGO knowledge base has been automatically extracted from Wikipedia, WordNet, and GeoNames by researchers at the Max Planck Institute for Computer Science. It can be downloaded from http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/

Both DBPedia and YAGO contain millions of instances and thousands of schema-level entities (see Table 3.6). We are specifically interested in aligning the properties of these two datasets, so we have extracted a cohesive subset of each one that will allow us to do this without requiring an inordinately long runtime. This was done using the following procedure:

1. For each property in YAGO, randomly choose five facts that involve the property. For properties with less than five facts, use all that are available.

2. Include the classes for the type of every instance mentioned in the facts from step 1.

Table 3.6: Characteristics of the DBPedia and YAGO samples

Dataset	DBPedia	YAGO
Classes	617	10962
Object Properties	1046	85
Data Properties	1407	37
Named Individuals	8685	1680
Datatypes	23	23
Annotations	77	125
Total Entities	11855	12912

3. Randomly select and add up to five other facts related to the instances from step 1.

4. Repeat step 2 for any additional instances added during step 3.

5. Compute the "closure" of this set of entities by recursively retrieving all schema-related axioms related to any entity within our sample.

The procedure for creating the DBPedia sample was analogous, except that instead of randomly choosing the facts in step 1, we selected facts with the same instances as our YAGO sample when available. This is possible because, since DBPedia and YAGO both represent information from Wikipedia, there is error-free mapping of instances that point to the same Wikipedia page. When there were no matching YAGO instances for the facts related to a particular DBPedia property, we reverted to randomly choosing facts. The characteristics of these dataset samples are shown in table 3.6.

This dataset sample may be of use to other researchers, so we have made it publicly available at http://www.michellecheatham.com/files/dbpedia-yago.zip. It should be noted that DBPedia and YAGO are much larger and have more of a "real-world" character than the ontologies in the OAEI Conference track. For instance, many properties defined in the ontologies are never used or are incompletely defined (e.g. missing domain or range definitions). Also, the definitions of some properties are spread across a datatype property, which specifies the range, and an annotation property, which specifies the domain (see below). Using domain axioms in conjunction with annotation properties is forbidden in the W3C Recommendation for OWL.[4] Also, some of the properties appear to be used inconsistently, or at least more broadly than they are defined. For instance, in DBPedia we see that the instance HAL 9000 has a gender property with a value of male and that Eaglet (Alice's

[4]http://www.w3.org/TR/owl-ref/#Annotations

Adventures in Wonderland) has a gender value female. In some cases the gender property is used differently, however. For instance, the instance Alexander has a gender property value of Alexandra, and the value for Maine North High School is mixed-sex education. While these issues can be a pain to work with, they are realistic concerns that ontology alignment systems will need to face for many application scenarios.

```
<!-- http://dbpedia.org/ontology/mayorFunction -->

<owl:DatatypeProperty
rdf:about="http://dbpedia.org/ontology/mayorFunction">
    <rdfs:label xml:lang="en">mayor function of a switzerland
    settlement</rdfs:label>
    <rdfs:range
    rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>

<owl:AnnotationProperty rdf:about="http://dbpedia.org/ontology/mayorFunction">
    <rdfs:label xml:lang="en">mayor function of a switzerland
    settlement</rdfs:label>
    <rdfs:domain rdf:resource=
        "http://dbpedia.org/ontology/SwitzerlandSettlement"/>
</owl:AnnotationProperty>
```

There is currently no curated alignment of the properties in the DBPedia and YAGO datasets. We would like to use the crowdsourcing approach outlined in Section 3.1.3 to create a complete reference alignment for the properties in these two datasets. It is not realistic to crowdsource opinion on all possible pairs of properties, however. A set of potential mappings is needed to bootstrap the crowdsourcing effort. Unfortunately, not many alignment systems have made results available for this pair of ontologies. The developers of the PARIS alignment system are the exception – they have produced and made public a set of subsumption relationships between properties [Suchanek et al. 2011a]. We can consider the cases where subsumption relations between two properties exist in both directions as indicative of an equivalence relation. These matches are shown in Table 3.7. We will use these matches, together with those produced by a basic string similarity metric and by our string-based property matcher described in the next chapter to begin the process of crowdsourcing a viable reference alignment. Due to the limited number of alignment approaches providing the potential matches to verify, this method will allow us to assess precision reasonably well but recall

Table 3.7: Equivalence mappings between DBPedia and YAGO properties as identified by the PARIS alignment system

DBPedia	YAGO
spouse	isMarriedTo
knownFor	isKnownFor
place	happenedIn
residence	livesIn
deathDate	diedOnDate
currency	hasCurrency
almaMater	graduatedFrom
isbn	hasISBN
birthDate	wasBornOnDate
doctoralAdvisor	hasAcademicAdvisor
motto	hasMotto
capital	hasCapital
numberOfStaff	hasNumberOfPeople
long[5]	hasLongitude
lat	hasLatitude

values are likely to be less accurate. While less than ideal, this is a common method of evaluation in the absence of an established reference alignment [Madhavan et al. 2001; Suchanek et al. 2011a; Ritze et al. 2009].

4

String-based Property Alignment

We have already seen indications that property alignment is difficult. Section 2.3.2.3 showed that traditional string similarity metrics perform much worse on properties than on classes. Furthermore, section 3.1.1 explains that current state-of-the-art alignment systems agree on many class equivalence relations in the conference test set but on no property equivalences. We will begin our development of a property-specific alignment system by further analyzing the performance of current systems with respect to properties. Table 4.1 shows the results of the top 2013 OAEI competitors on the Conference track, broken down into classes and properties[1]. The average f-measure for classes is more than three times that for properties.

Tables 4.2, 4.3, and 4.4 present the most common correct and incorrect property matches identified by the participants in the 2013 OAEI, along with the valid property matches that were most frequently omitted by those systems. The frequency column in these tables indicates the number of alignment systems out of the 15 qualifying systems that produced each match. The last column in Tables 4.2 and 4.4 shows the degree of confidence in each match. This value is based on the results of the survey of experts described in Chapter 3. Table 4.3 does not have this column because our survey focused on verifying the matches in the existing reference alignments for the Conference track, rather than attempting to identify missing matches. Table 4.2 shows that the equivalent properties that were most frequently correctly identified all have very high string similarity. Unfortunately, Table 4.3 shows that high string similarity is also the defining characteristic of the most common false positives. It may seem surprising that some of the matches in this table are not valid. The OAEI reference alignments are strongly biased towards achieving a logically consistent merged ontology. There are several reasons these matches may have been omitted from the reference alignments for the OAEI. In some cases the domain or range of the matched properties indicate that they are not being used in the same way. For instance, the domain of cmt:name is the union of Person and Conference

[1] MapSSS and StringsAuto do not attempt to align properties

System	Class Prec	Class Rec	Class Fms	Prop Prec	Prop Rec	Prop Fms
AML	0.86	0.62	0.72	1.00	0.20	0.33
AMLback	0.86	0.64	0.73	1.00	0.24	0.39
CIDER_CL	0.46	0.59	0.52	0.07	0.22	0.11
HerTUDA	0.84	0.56	0.67	0.26	0.20	0.23
HotMatch	0.81	0.57	0.67	0.24	0.20	0.22
IAMA	0.87	0.55	0.67	0.14	0.07	0.09
LogMap	0.82	0.65	0.73	0.62	0.28	0.39
MapSSS	0.74	0.59	0.66	0.00	0.00	0.00
ODGOMS	0.87	0.55	0.67	0.32	0.26	0.29
ODGOMS1_2	0.81	0.66	0.73	0.32	0.26	0.29
ServOMap_v104	0.74	0.65	0.69	0.00	0.00	0.00
StringsAuto	0.71	0.63	0.67	0.00	0.00	0.00
WeSeEMatch	0.85	0.54	0.66	0.50	0.02	0.04
WikiMatch	0.84	0.54	0.66	0.26	0.22	0.24
YAM++	0.82	0.71	0.76	0.68	0.57	0.62
Average	0.79	0.60	0.68	0.36	0.18	0.21

Table 4.1: Performance of the top 2013 OAEI competitors on classes versus properties

whereas the domain of sigkdd:Name is only Person and a separate property, Name_of_conference, is used to represent a conference's name. Presumably in other cases the match may make sense in isolation but would lead the merged ontology to be logically inconsistent. It is difficult to identify these cases, however. It would be helpful if the developers of the original Conference track reference alignments made them available in some form. Finally, we see in Table 4.4 that for many of the most frequently missed matches, humans have a difficult time agreeing. Out of the 31 matches, expert confidence was lower than 55 percent for 11 of them. The properties involved in these false negatives have a much lower string similarity, for instance cmt:hasBeenAssigned and ekaw:ReviewerOfPaper. In many of these cases, the domain and range *do* have strong syntactic similarity however, e.g. Reviewer and Paper for hasBeenAssigned and Possible_Reviewer and Paper for reviewerOfPaper. Further, there were some quite frequently missed equivalent properties that have strong clues in the labels themselves, such as cmt:writePaper and confOf:writes. Of the 31 common false negatives in Table 4.4, 13 have noticeable string similarity.

We have established that aligning properties is quite difficult, both for string similarity metrics alone and for the current top-performing alignment systems. In this chapter we will attempt to

Property 1	Property 2	Freq.	Conf.
cmt:email	confOf:hasEmail	11	0.92
confOf:hasFirstName	edas:hasFirstName	11	1.0
conference:has_an_email	confOf:hasEmail	9	1.0
cmt:email	conference:has_an_email	9	0.85
conference:has_the_last_name	edas:hasLastName	9	1.0
conference:has_a_review	ekaw:hasReview	9	1.0
conference:has_the_first_name	edas:hasFirstName	9	0.92
conference:has_the_first_name	confOf:hasFirstName	9	0.92

Table 4.2: Most common correct property matches identified by alignment systems in the 2013 OAEI

develop a string-based approach that exhibits better performance on property alignment and evaluate its performance.

4.1 Related Work

Empirical analysis of existing ontologies has shown that different naming conventions are used for different entity types. For instance, empirical analysis of existing ontologies has shown that object properties generally begin with a verb (e.g. attends, employs) or end with a preposition (e.g. friendOf, componentFor) while datatype properties are usually nouns (e.g. value, id, etc.). Additionally, the names of inverse properties were found to commonly follow one of two patterns: (1) active and passive forms of the same verb (e.g. wrote and writtenBy) or same noun phrase packed in auxiliary terms (e.g. memberOf and hasMember) [Svátek et al. 2009]. These different naming conventions may be one reason for the generally poor performance of string similarity metrics on properties.

In 2002 Melnik and his colleagues developed a strategy called "similarity flooding" to improve the performance of alignment systems. The general idea is that an initial pass is made through the datasets to establish a set of high precision anchor mappings, such as exact string matches. Then similarity values are propagated to adjacent nodes. If the similarity value of two nodes reaches a threshold, they are considered equivalent. The algorithm iterates until a fixed point is reached [Melnik et al. 2002]. This technique may improve the performance on property alignment by leveraging the increased accuracy of class and instance alignment. Similarity flooding in somewhat similar to extensional alignment techniques, in which two properties are judged more similar if the instances related by those properties within a dataset are similar [Gunaratna et al. 2013].

Property 1	Property 2	Freq.
iasted:pay	sigkdd:pay	9
confOf:hasEmail	edas:hasEmail	9
cmt:email	edas:hasEmail	8
cmt:name	sigkdd:Name	8
confOf:hasPhone	edas:hasPhone	8
confOf:hasStreet	edas:hasStreet	7
confOf:hasPostalCode	edas:hasPostalCode	7
iasted:obtain	sigkdd:obtain	7
confOf:hasTopic	edas:hasTopic	7
conference:has_an_email	edas:hasEmail	7
cmt:writtenBy	confOf:writtenBy	7

Table 4.3: Most common incorrect property matches identified by alignment systems in the 2013 OAEI

An ontology-centric version of the basic similarity flooding technique was first employed in RiMOM and subsequently adopted by many other ontology alignment systems. Rather than propagating similarity values to all neighbors in a graph structure, this approach considered sub-concepts, siblings, and properties for classes and sub-properties, range, and domain for properties [Li et al. 2009]. Suchanek et. al. recently applied this ontology-oriented similarity flooding approach in their PARIS alignment system, which identifies both equivalence and subsumption relations for classes and properties [Suchanek et al. 2011a]. They found that while class alignments didn't do much to facilitate alignment of properties or instances, there was significant interplay between the latter two. This was particularly true for functional or nearly functional properties, in which any domain value maps to only one range value.

There have been several attempts to modify the standard similarity flooding approach to further improve the performance on property matching. For example, comparison of instance data and datatype property range values can be improved by using different similarity metrics for strings, dates, integers, etc. [Zhao and Ichise 2013]. Further, in deference to the difficulty of matching properties, it is possible to propagate a fraction of the normal similarity values when adjacent properties are compatible rather than definite matches. This is the approach taken in [Pernelle et al. 2013] where compatibility for properties is defined as those with domains and ranges that are either the same or subtypes of one another.

Another particularly problematic aspect for property matching is the variety of design decisions

made when an ontology is created. For instance, some ontologies are class-centric while others are property-centric (e.g. SeasonTicketHolder versus holdsSeasonTicket) [Šváb 2007]. Intuitively we would like to say that if two ontologies had these entities, there should be some sort of mapping between them. Other ontology design decisions that impact property matching are how to handle part-whole relationships and when to reify properties [Noy and Hafner 1997]. Additionally, taxonomies of properties are much less common than those of classes [Svátek et al. 2009; Noy and Hafner 1997]. There has been some discussion in the literature of handling differences in design philosophy through ontology transformation, in which design patterns are recognized and translated into an analogous form [Sváb-Zamazal et al. 2009]. Ritze and her colleagues used this pattern-centric approach to find complex mappings between classes (and value restrictions) in one ontology and properties in another [Ritze et al. 2009].

4.2 String-based Approach

Our goal is to develop an entirely string-based approach to property alignment. This is in keeping with our belief that the label given to an entity is a very good indicator of its intended meaning and use, and that string-based approaches can often be competitive with much more complex and time-consuming methods. Specifically, we would like to develop an approach with the following characteristics:

- The approach should be elegant, in the sense that it is as simple as possible. It should not require setting a large number of parameters or utilize a store of rules or templates, which can be brittle.
- The approach should not make any unnecessary assumptions nor place any restrictions on how the metric can be used. In particular, it should not make arbitrary trade-offs between precision and recall, because the appropriate choice is very application-dependent.
- The approach should produce meaningful confidence values for each match, which correspond well to human opinion.

Considering these guidelines, we have arrived at the following approach, which we will call PropString.

Four strings are extracted for each property: the label, the core concept, the domain, and the range. All strings are tokenized and put into lower case.

The label is simply the entity's label, i.e. the portion of the URI after the last # or / or, failing that, the value in the rdf:label field.

The core concept is either the first verb in the label that is greater than four characters long or, if there is no such verb, the first noun in the label, together with any adjectives that modify that noun. For example, the label "wrote paper" has the core concept "wrote" because it is the first verb. The label "has corresponding author" has the core concept "corresponding author" because "has" is a verb less than five characters and "corresponding" is an adjective modifying the noun "author." We arrived at this technique through an analysis of common naming patterns for properties. While this approach is not perfect, it frequently allows us to detect the key part of property labels without the need for templates or rules. We used the Standford log-linear part of speech tagger to compute the core concept [Toutanova et al. 2003]. We used the model included with the software distribution, which was trained on English text from the Wall Street Journal.

The domain (resp. range) string is a concatenation of the labels of any classes in the domain (resp. range) of a property. In the case of datatype properties, the range is set to "literal." This was done rather than using the actual datatype because there are a variety of ways to represent the same data. In particular, many times information that is inherently numeric is encoded as a string. In order to facilitate strong recall, we take a liberal viewpoint on datatype similarity.

The similarity of each of these four pairs of strings is then computed. For the entity label and core concept, the soft TF-IDF metric, trained on the properties from both ontologies, is used. The internal threshold for that metric is set at 0.9. The similarity of the domain and range is computed using a standard TF-IDF metric trained on all entities from both ontologies, which was shown in Chapter 2 to have reasonable performance on classes in terms of both precision and recall. Because both the soft and regular versions of the TF-IDF metric are asymmetric, we compute the similarity values in both directions, i.e. sim(a, b) and sim(b, a), and average the two values.

While the vast majority of alignment systems use a string similarity metric, they use them in different ways. One approach is to find highly precise "anchor" matches which serve as the seed that the rest of the alignment grows out from. Another approach is to use a string metric to filter out any obviously incorrect matches in order to reduce computational complexity. This requires a string metric with high recall. To address both of these use cases, the PropString approach can be run in two configurations: precision-oriented and recall-oriented. In the precision-oriented mode, a pair of entities is considered a match if the similarity values for their core concepts, domains, **and** ranges are all greater than the threshold. In the recall-oriented mode, the pair is considered a match if the similarity values for their core concepts **or** their domains and ranges are greater than the threshold. One question that might be asked is "why not concatenate the domain and range information on to the property's label and consider the entire thing as one string?" The reason this is not done is because it frequently confuses inverse properties, in which the domain of one property is the same

as the range of the other and vice versa, as equivalent.

Requiring the core concept, domain, and range similarity to all be higher than the threshold results in very good precision and avoids many false positives of the variety that seem reasonable based on label but are used for different purposes by the two ontologies. However, allowing matches based solely on high similarity of domain and range in the recall-oriented configuration results in very low precision unless further steps are taken. To address this, we take two approaches that work together to reduce the number of false positives in this configuration. The first is the calculation of the confidence value: this is done by averaging the similarity values for the exact labels, their domains, and their ranges. The second is that we keep a list of each entity that is considered a match so far, along with the entity it maps to and the confidence value. Every time a new potential match between properties is identified, its confidence value is checked against any existing current matches involving those properties. If the new match has a greater confidence value, the old match is removed in favor of the new one. If the new match does not have greater confidence, it is ignored. Using the exact label similarity when computing the confidence values rather than the core concept eliminates the loss of precision associated with extracting the core concept, effectively breaking any ties in favor of the closer lexical matches. This approach is somewhat similar to the stable marriage algorithm used in the test framework in Chapter 2, but it is significantly more scalable. The effect of this approach is that any properties with the same domain and range act as a filter, with the specific match from that set chosen based on the actual property label. This is shown below, where the YAGO property "influences," with a domain and range of "Person," is being matched:

```
yago:influences = dbpedia:relative: 0.67
yago:influences = dbpedia:father: 0.67
yago:influences = dbpedia:mother: 0.67
yago:influences = dbpedia:spouse: 0.67
yago:influences = dbpedia:influencedBy: 0.93
yago:influences = dbpedia:influenced: 0.99
```

One thing to note: when looking at the mistakes made by an alignment system on established benchmarks, it is very tempting to develop additional checks to avoid them. For instance, if the system cannot determine that "has the last name" is equivalent to "has surname," it is very tempting to add a module that uses a thesaurus, web search engine, or other resource to evaluate the semantic similarity between labels. Similarly, if two properties have the same label, such as "has name," but one is used for people while the other is used only for conferences, it is tempting to use the instance data available to differentiate between these uses. We have tried both of these approaches, along with many, many others. The difficulty is that, while these approaches do indeed eliminate the

mistakes they were inspired by, they tend to cause many more. That is why it is important to evaluate each individual aspect of any proposed alignment method to make sure that they are all contributing positively to overall performance. This is the goal of the next section.

The source code for PropString is available at http://michellecheatham.com/files/PropString.zip.

4.3 Evaluation

In this section we will analyze the performance of the PropString approach when applied to both the existing and our proposed revision of the OAEI Conference track. We will also consider the effect on overall performance of each aspect of the approach. Finally, we will consider the results produced on the larger, real-world DBPedia-YAGO alignment task.

4.3.1 Conference

Table 4.5 shows the results of PropString on both versions of the Conference track reference alignments. The system was configured with a threshold of 0.9 and to only include matches in which both entities were in the namespace of the ontologies to be matched (in accordance with the OAEI guidelines). The results are compared with those of Soft TF-IDF with a threshold of 0.8. Recall from figure 2.13 that this was the best-performing string metric for property alignment. It is evident that PropString greatly outperforms Soft TF-IDF on the current version of the Conference track. The precision-oriented configuration of PropString quintuples the precision of Soft TF-IDF (to a perfect 1.0) while maintaining roughly the same recall. Analogously, the recall-oriented version doubles the recall of Soft TF-IDF while still achieving noticeably better recall. The f-measures for both the precision- and recall-oriented configurations are double that of Soft TF-IDF. The results for the expert-validated versions of the existing reference alignment tell essentially the same story for both the discrete and continuous evaluation approaches, with the caveat that the recall-oriented configuration does not have quite as high precision value in the continuous case.

Table 4.6 is a duplicate of 4.4, which shows the most commonly missed matches by the top-performing alignment systems from 2013. The extra column shows the confidence value that PropString (running in the recall configuration with a threshold of 0.9) assigned to each of these matches. PropString was able to correctly identify 9 of these 31 matches, including 8 of the 22 on which more than half of the experts agreed. This is quite encouraging considering that these were the most difficult matches for current alignment systems to identify. Several matches with limited or no label similarity were correctly found, such as edas:endDate = sigkdd:End_of_conference and conference:contributes = ekaw:authorOf. In addition, the confidence values assigned to the matches

that were found have quite reasonable correlation to the percentage of experts who agreed with the matches.

We now turn from a holistic evaluation of the performance of this approach to analyzing the effect of each aspect of the method on overall performance. This is somewhat difficult to do because the aspects do not stand alone – they influence one another. Therefore there is a combinatorial problem related to fully analyzing the behavior. We will take the approach of considering the impact of each design aspect when added to the basic Soft TF-IDF metric (Table 4.7), as well as the impact of each aspect when removed from the complete PropString approach (Table 4.8). For simplicity, results are shown only for version 2 of the Conference track reference alignments.

Table 4.7 shows that no approach in isolation can achieve the overall precision, recall, or f-measure of the complete PropString metric. Also, the table shows that extracting the core concept from the property labels and considering domain and range information independent of the property label both significantly improve recall, as designed. Further, we see that considering domain and range in addition to the property label has a very large impact on precision. Finally, training the soft TF-IDF metric on only properties rather than all entities did not improve results, but it also does not negatively impact precision or recall, which is useful in the sense that it is more scalable and time-efficient.

Table 4.8 shows that there are not any superfluous aspects to the PropString metric – removing any element reduces performance. In particular, removing the idea of extracting the core concept from property labels has such a disastrous effect on recall that the precision-oriented configuration becomes useless. Similarly, removing either the best match filter or using simple label similarity for the confidence value rather than averaging label, domain, and range similarity cuts precision in half in the recall-oriented configuration. Consideration of domain and range in the similarity computation is shown to be the key to this approach. (Note that the precision and recall orientations are based on whether or not domain and range are required to be similar, so there is only one row in the table for this aspect.)

4.3.2 DBPedia-YAGO

In order to evaluate the performance of PropString on a larger and more realistic alignment task, we apply it to the DBPedia-YAGO test case described in Chapter 3. We compare the performance of PropString to that of the basic Soft TF-IDF similarity metric and the PARIS alignment system. PARIS is an acronym for Probabilistic Alignment of Relations, Instances, and Schema. The system approaches property alignment by considering the degree of overlap between the sets of instances involving each property [Suchanek et al. 2011b].

There is no established reference alignment for the DBPedia and YAGO ontologies. We begin the process of creating one by collecting the equivalent property relationships generated by PropString, Soft TF-IDF, and PARIS and using Mechanical Turk to verify their accuracy. In total, these three approaches produced 133 unique equivalence matches that involved properties. We formulated questions for each match of the form "Does property label A mean the same thing as property label B?" Respondents were instructed to choose one of four options:

1. They mean the same thing
2. One is a more general or more specific term than the other
3. They are related in some other way
4. There is no relation

We provided these more nuanced options rather than just yes or no because we would like to eventually develop a reference alignment useful for evaluating the performance of alignment systems that produce all types of matches. In order to provide some context, we provided information about the domain and range of each property and up to five examples of instances with values for each property. An example of one of these questions is shown below:

```
A "person" has a property called "directed" that involves a "thing." Examples are:
dario argento -> four flies on grey velvet
eldor magomatovich urazbayev -> tailcoat for shalopaya
masahiro shinoda -> gonza the spearman
jon monday -> the last straw film
d. w. griffith -> the fight for freedom

A "thing" has a property called "director" that involves a "person." Examples are:
la rabbia -> pier paolo pasolini
two living, one dead -> anthony asquith
sasneham -> sathyan anthikad
la rabbia -> giovannino guareschi
smart blonde -> frank mc donald (director)

Does "directed" mean the same thing as "director"?

Please choose the best answer
```

```
1. They mean the same thing
2. One is a more general or more specific term than the other
3. They are related in some other way
4. There is no relation
```

The 133 matches were grouped into 19 sets of 7 questions each, and we paid 25 cents for each set. Preliminary testing showed that the general response on these more nuanced questions were not as reliable as those asked when verifying the Conference track reference alignments (which were simply "yes or no" questions). We therefore asked only the Super Turkers from that experiment to participate in this one. There were ten of these individuals, and we got input from 6 or 7 of them for each match.

Table 4.9 shows a summary of the results; the complete set are included in Appendix B. The second column of Table 4.9 shows, out of the 133 suggested equivalence matches involving properties, the number of matches falling into each relationship category according to the consensus among the Turkers. The third column indicates the average percentage agreement among the Turkers on each category. Overall, the percentage of agreement with the consensus answer across all matches is 72 percent.

Rather than requiring precise agreement on the type of relationship (if any) for each potential match, it might make sense for our current purposes to consider a weaker sense of agreement. One way to do this is to consider two answers to be in agreement if they both either indicate some relationship exists or they both conclude there is no relation between the two properties. In this case, if one person indicated two entities are related in a sub/super relationship and another indicates that they are equivalent, these answers would be considered in agreement. Two answers would only be seen to disagree if one indicated there is no relation at all and the other disagreed. This way of interpreting the results might be useful for an alignment system if the results from this phase were being used to either find all types of relationships between entities or to gather all possible matches and use further processing to filter the set down to only equivalence relations. We will call this "recall-oriented." Using this measure, the consensus among the Turkers across all suggested matches is 88 percent.

Another possible way to interpret the results is to consider two answers to be in agreement only if they both conclude either that the entities are precisely equivalent or that they are not equivalent. Using this viewpoint, if one person indicates that two entities are related in a sub/super relationship and another indicates that they are precisely equivalent, these answers would be seen as disagreeing. If instead one person considered the match to be a sub/super relationship and another considered them to have no relationship at all, these two individuals would be seen as in agreement because

they both conclude that there is no equivalence relationship. This interpretation may be useful if an alignment system is attempting to find high-quality equivalence relations between entities, which it may subsequently use as a seed for further processing. We will refer to this as "precision-oriented." Coincidentally, the overall agreement among the Turkers using this definition is also 88 percent.

Figure 4.1 shows compares the results of PARIS, Soft TF-IDF, and PropString on the YAGO-DBPedia property alignment task. This figure measures precision, recall, and f-measure using the precision-oriented definition of correctness described above. PropString was run in both its precision and recall configurations. Note that this refers to the way PropString filters potential matches and is orthogonal to the precision- and recall-oriented performance metrics. The threshold for Soft TF-IDF was set to 0.8 and the threshold for PropString was set to 0.9, based upon the best-performing thresholds for these approaches on the Conference track. In addition, PropString was run in its recall configuration with a threshold of 0.5 because the developers of PARIS indicated that they needed to use a lower threshold on the DBPedia-YAGO task than on the OAEI alignment tasks. Figure 4.2 shows the same information using the recall-oriented definition of correctness for the computation of precision, recall, and f-measure.

Figures 4.1 and 4.2 reveal that the basic string metric Soft TF-IDF produces the highest precision, regardless of how is correctness is measured. Further, that precision is 0.79 and .96 (depending on evaluation approach), which is on par with the degree of agreement among the Turkers on these matches. So once again we see that a straightforward string metric can in some ways outperform more sophisticated alignment strategies. In fact, PARIS and the precision-based configuration of PropString have such low recall that they may not be of much utility for many application scenarios. This is surprising considering the strong performance of this PropString configuration on the properties within the Conference track. We argue that this wide variation in performance is further indication of the need for more benchmarks involving property alignment.

Another thing to note from these results is the very strong performance of the recall-based configuration of PropString, both relative to the other approaches and in an absolute sense. When PropString is run in its recall configuration with a threshold of 0.5, both the precision and recall are in the neighborhood of that produced by much more complex alignment systems on the simpler task of class equivalence in more structured test sets, such as the Conference track. Of course, the very preliminary nature of the YAGO-DBPedia reference alignment must be kept in mind. More work, hopefully involving results produced by many other alignment system on this pair of ontologies, is needed to confirm these results.

It is also interesting that the performance of the recall-based configuration of PropString varies more than the other approaches between the precision- and recall-oriented definitions of correctness.

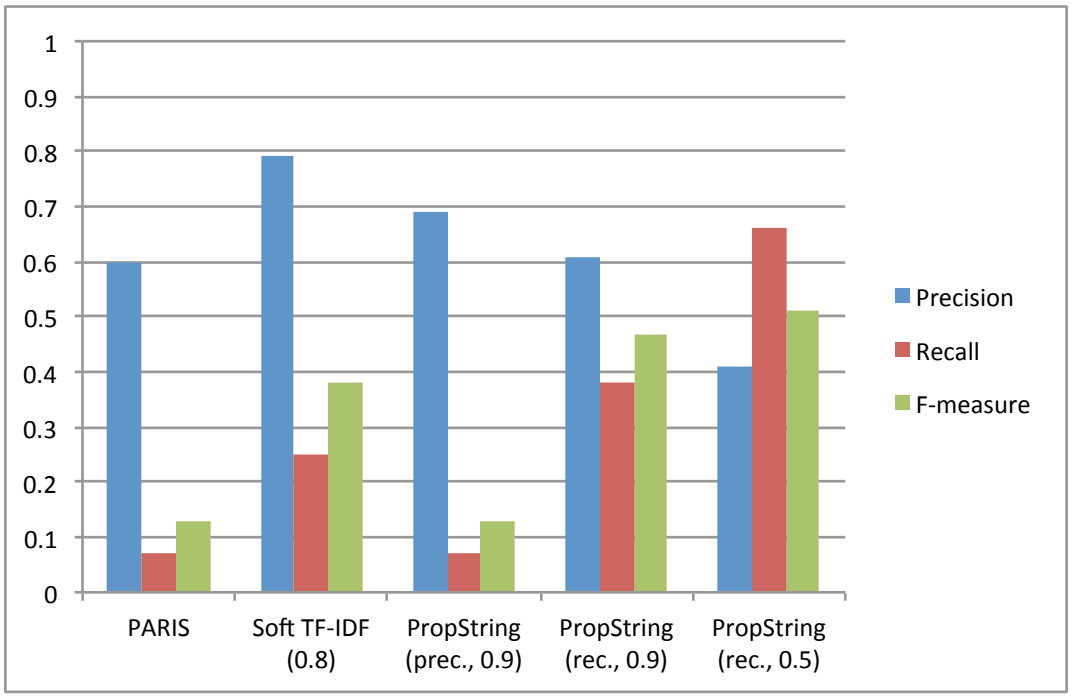


Figure 4.1: Precision-oriented evaluation of the results on the YAGO-DBPedia alignment task

This implies that this form of PropString is identifying many valid relationships but is incorrectly classifying them all as equivalence rather than correctly distinguishing sub-property/super-property relationships, inverse properties, etc.

In general, the performance of PropString when matching properties in both the OAEI Conference track and the YAGO-DBPedia alignment task compares favorably with existing approaches. However, further validation is needed, particularly with respect to more real-world alignment tasks.

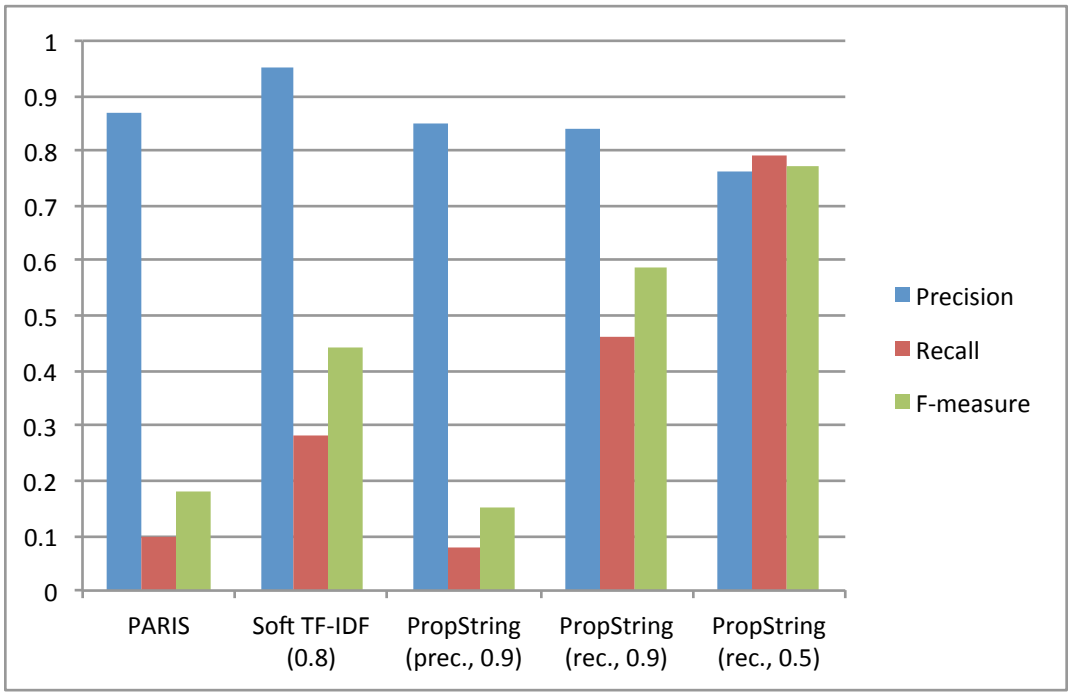


Figure 4.2: Recall-oriented evaluation of the results on the YAGO-DBPedia alignment task

Property 1	Property 2	Freq.	Conf.
cmt:hasBeenAssigned	ekaw:reviewerOfPaper	15	0.46
cmt:assignExternalReviewer	conference:invites_co-reviewers	15	0.54
cmt:assignedByReviewer	conference:invited_by	15	0.31
edas:endDate	sigkdd:End_of_conference	15	0.85
conference:is_given_by	sigkdd:presentationed_by	15	0.85
conference:has_a_track-workshop-tutorial_topic	confOf:hasTopic	15	0.31
conference:contributes	iasted:write	15	0.31
cmt:hasBeenAssigned	confOf:reviewes	15	0.62
conference:gives_presentations	sigkdd:presentation	15	0.46
conference:has_the_last_name	confOf:hasSurname	15	0.92
cmt:assignedTo	ekaw:hasReviewer	15	0.69
confOf:reviewes	edas:isReviewing	15	0.77
confOf:hasSurname	edas:hasLastName	15	1.0
conference:has_a_review_expertise	edas:hasRating	15	0.23
cmt:writtenBy	ekaw:reviewWrittenBy	15	0.69
cmt:hasSubjectArea	confOf:dealsWith	14	0.46
cmt:writePaper	confOf:writes	14	0.62
edas:isReviewedBy	ekaw:hasReviewer	14	0.92
cmt:hasAuthor	confOf:writtenBy	14	1.0
confOf:writes	edas:hasRelatedPaper	14	0.23
edas:hasCostAmount	sigkdd:Price	14	0.85
cmt:assignedTo	edas:isReviewedBy	14	0.92
edas:startDate	sigkdd:Start_of_conference	14	0.92
cmt:hasConferenceMember	edas:hasMember	14	0.54
cmt:hasBeenAssigned	edas:isReviewing	14	0.62
edas:hasLocation	ekaw:heldIn	14	0.92
edas:hasName	sigkdd:Name_of_conference	14	0.08
edas:isReviewing	ekaw:reviewerOfPaper	14	1.0
confOf:hasEmail	sigkdd:E-mail	13	0.92
conference:has_an_email	sigkdd:E-mail	13	0.92
conference:contributes	ekaw:authorOf	13	0.69

Table 4.4: Most common correct property matches omitted by alignment systems in the 2013 OAEI

System	Ver	Dis Prec	Dis Rec	Dis Fms	Con Prec	Con Rec	Con Fms
PropString (prec)	1.0	1.0	0.26	0.41	NA	NA	NA
PropString (rec)	1.0	0.34	0.5	0.4	NA	NA	NA
Soft TF-IDF	1.0	0.2	0.24	0.22	NA	NA	NA
PropString (prec)	2.0	0.92	0.3	0.45	0.89	0.28	0.43
PropString (rec)	2.0	0.32	0.59	0.42	0.25	0.51	0.33
Soft TF-IDF	2.0	0.2	0.3	0.24	0.16	0.23	0.19

Table 4.5: Results on the OAEI Conference track on both versions of the reference alignments

Property 1	Property 2	Expt.	Sys.
cmt:hasBeenAssigned	ekaw:reviewerOfPaper	0.46	0.0
cmt:assignExternalReviewer	conference:invites_co-reviewers	0.54	0.0
cmt:assignedByReviewer	conference:invited_by	0.31	0.0
edas:endDate	**sigkdd:End_of_conference**	**0.85**	**0.84**
conference:is_given_by	sigkdd:presentationed_by	0.85	0.0
conference:has_a_track-workshop-tutorial_topic	confOf:hasTopic	0.31	0.0
conference:contributes	iasted:write	0.31	0.0
cmt:hasBeenAssigned	confOf:reviewes	0.62	0.0
conference:gives_presentations	sigkdd:presentation	0.46	0.0
conference:has_the_last_name	confOf:hasSurname	0.92	0.0
cmt:assignedTo	ekaw:hasReviewer	0.69	0.0
confOf:reviewes	edas:isReviewing	0.77	0.0
confOf:hasSurname	edas:hasLastName	1.0	0.0
conference:has_a_review_expertise	edas:hasRating	0.23	0.0
cmt:writtenBy	**ekaw:reviewWrittenBy**	**0.69**	**0.75**
cmt:hasSubjectArea	confOf:dealsWith	0.46	0.0
cmt:writePaper	**confOf:writes**	**0.62**	**0.61**
edas:isReviewedBy	**ekaw:hasReviewer**	**0.92**	**0.67**
cmt:hasAuthor	confOf:writtenBy	1.0	0.0
confOf:writes	edas:hasRelatedPaper	0.23	0.0
edas:hasCostAmount	sigkdd:Price	0.85	0.0
cmt:assignedTo	edas:isReviewedBy	0.92	0.0
edas:startDate	**sigkdd:Start_of_conference**	**0.92**	**0.84**
cmt:hasConferenceMember	edas:hasMember	0.54	0.0
cmt:hasBeenAssigned	edas:isReviewing	0.62	0.0
edas:hasLocation	ekaw:heldIn	0.92	0.0
edas:hasName	**sigkdd:Name_of_conference**	**0.08**	**0.85**
edas:isReviewing	ekaw:reviewerOfPaper	1.0	0.0
confOf:hasEmail	**sigkdd:E-mail**	**0.92**	**0.87**
conference:has_an_email	**sigkdd:E-mail**	**0.92**	**0.86**
conference:contributes	**ekaw:authorOf**	**0.69**	**0.63**

Table 4.6: PropString performance on the most common correct property matches omitted by alignment systems in the 2013 OAEI

Configuration	Dis Prec	Dis Rec	Dis Fms	Con Prec	Con Rec	Con Fms
Soft TF-IDF	0.18	0.32	0.24	0.15	0.25	0.19
Property-trained	0.17	0.35	0.23	0.14	0.29	0.19
Core concept	0.09	0.46	0.15	0.08	0.44	0.14
Best match	0.2	0.3	0.24	0.16	0.23	0.19
Confidence calc.	0.37	0.35	0.36	0.25	0.27	0.26
Domain/range (prec)	0.86	0.16	0.27	0.81	0.14	0.23
Domain/range (rec)	0.2	0.38	0.26	0.15	0.28	0.2
PropString (prec)	0.92	0.3	0.45	0.89	0.28	0.43
PropString (rec)	0.32	0.59	0.42	0.25	0.51	0.33

Table 4.7: Impact of individual components added to Soft TF-IDF on performance on Conference version 2

Configuration	Dis Prec	Dis Rec	Dis Fms	Con Prec	Con Rec	Con Fms
PropString (prec)	0.92	0.3	0.45	0.89	0.28	0.43
PropString (rec)	0.32	0.59	0.42	0.25	0.51	0.33
Property-trained (prec)	0.92	0.3	0.45	0.89	0.28	0.43
Property-trained (rec)	0.33	0.59	0.42	0.25	0.48	0.33
Core concept (prec)	1.0	0.05	0.1	0.97	0.06	0.1
Core concept (rec)	0.34	0.46	0.39	0.3	0.4	0.34
Best match (prec)	0.92	0.3	0.45	0.89	0.28	0.43
Best match (rec)	0.14	0.49	0.22	0.1	0.37	0.16
Confidence calc. (prec)	0.92	0.3	0.45	0.87	0.3	0.44
Confidence calc. (rec)	0.14	0.43	0.22	0.14	0.41	0.2
Domain/range	0.37	0.49	0.42	0.26	0.4	0.32
Soft TF-IDF	0.18	0.32	0.24	0.15	0.25	0.19

Table 4.8: Impact of individual components removed from PropString on performance on Conference version 2

Relation Type	Number of Matches	Percent Agreement
Equivalent	53	79
Sub/super	11	66
Other relation	37	60
No relation	32	76

Table 4.9: Summary of Mechanical Turk results on the YAGO-DBPedia matches identified by PARIS, PropString, and Soft TF-IDF

5

Conclusion

Ontology alignment is essential to the development of applications that leverage the potential of the Semantic Web. Unfortunately, most current state of the art alignment systems are capable of identifying only the simplest of relationships between ontologies: 1-to-1 equivalence. Even more vexing is that the performance of finding such equivalence relations between properties lags far behind that for classes and instances. The work presented here addressed that issue in a systematic, data-guided manner.

Chapter 2 showed that string similarity metrics alone can achieve performance that is competitive with the top-performing full-featured alignment systems. The key is to choose the correct similarity metrics for a particular pair of ontologies and use case. We presented guidelines to assist alignment system developers in making this choice. This chapter also showed that the most commonly suggested string preprocessing strategies, such as stop word removal and consideration of synonyms, are not nearly as effective as hoped. This is why it is important to analyze every component of an alignment system in isolation: too often alignment system developers will throw in everything but the kitchen sink and quit as soon as they achieve good performance. It is important to go back and review whether or not every component that was added contributes positively to the overall performance of the alignment system. This was our methodology when developing PropString, an entirely string-based approach to aligning properties. PropString works by extracting the "core concept" of a property's label. The similarity of two properties' core concepts, domains, and ranges are used to identify potential matches. The system can be run in either a precision- or recall-oriented configuration. In the precision-oriented mode, the similarity of the core concept, domain, and range must all be greater than the threshold. In the recall-oriented mode, a match is considered viable if either the core concept *or* the domain and range have a similarity above the threshold. The confidence value of a match is the average of the string similarity of the properties' labels, domains, and ranges. This was shown to correlate well to the degree of expert agreement on property matches.

PropString was evaluated on the OAEI Conference track. Unfortunately, the current version of the reference alignments for that dataset have a confidence value of 1.0 for every match. We showed via a survey of experts that humans do not achieve this degree of consensus on these alignments. In order to generate realistic confidence values for the matches that could be used to validate those produced by PropString, we created a more nuanced version of the OAEI Conference track reference alignments via expert survey and crowdsourcing using Amazon's Mechanical Turk platform. We also created a scaled-down version of the DBPedia and YAGO ontologies that can also be used to assess the performance of property alignment systems. Our evaluation of PropString on these alignment tasks showed that it performs well in comparison to both Soft TF-IDF, which is the best-performing basic string similarity metric, and PARIS, a full-featured alignment system.

5.1 Lessons Learned

Over the course of this research effort, we learned several things that are applicable to the field of ontology alignment in general.

5.1.1 Words matter

One of the unique features of ontologies is the subject matter expertise encoded in the structure of the data, i.e. the restrictions and relationships between entities. Many alignment systems understandably try to leverage this information. However, the importance of entity labels should not be overlooked. While the developers of an ontology may become focused on their particular application and forget to include some relevant axioms, they generally give careful consideration to the labels they choose for entities. The labels therefore contain a lot of implicit meaning and should consequently be carefully considered by alignment algorithms.

5.1.2 Aligning properties is not so hard

The performance of the large majority of current alignment systems is quite poor on properties, in terms of both precision and recall. However, simple string similarity metrics can perform quite well on the property matching task. The important thing to consider is that property labels typically follow different naming patterns than classes and instances. In addition, while the key structural information important for identifying class equivalences is sub- and superclasses, and for co-reference resolution it is class type, for properties it is domain and range that are most critical.

5.1.3 Benchmarks can drive innovation

In the paper "Using Benchmarks to Advance Research: A Challenge to Software Engineering," the authors state that "The creation and widespread use of a benchmark within a research area is frequently accompanied by rapid technical progress and community building" and provide several examples [Sim et al. 2003]. The OAEI benchmarks have been a big boon for the field of ontology alignment, but new types of benchmarks are needed for progress to continue. In particular, benchmarks involving properties, complex (beyond 1-to-1 equivalence) relations, and "messy" real-world ontologies are needed. It would also be beneficial to have some reference alignments that are focused on applications which do not require a logically consistent merged ontology, such as querying multiple datasets as in ontology-based data access (OBDA) systems. Developing new benchmarks for a field is an arduous task, but the preliminary feasibility of crowdsourcing and the already-existing active collaboration among researchers in the ontology alignment field are positives indications that this necessary goal is attainable.

5.2 Future Work

Several aspects of the work presented here require further validation. In particular, additional experimentation regarding crowdsourcing reference alignments using Mechanical Turk needs to be done to verify the potential uses of the approach. For instance, our preliminary results showed that general users can often give good input on "yes or no" alignment verification tasks but that more complex questions regarding the type of relationship between two entities (e.g. equivalence, subsumption, inverse properties) is more difficult. It would be useful to develop guidelines for when and how to qualify users for different types of alignment tasks. More work in particular remains to be done in order to generate an established high-quality reference alignment for the DBPedia-YAGO alignment task. In order to do this, we need to generate results on this ontology pair using more alignment systems. These results can then be manually verified, either through Mechanical Turk or by experts. Additionally, we need to incorporate the PropString approach into a full-featured alignment system and evaluate the difference in performance. Unfortunately, it can be difficult to acquire the source code for a top-performing alignment system.

We also plan to explore the potential of raising the level of abstraction at which ontology alignment algorithms work by considering pattern-based alignment. Many times a dataset will contain a frequently-used grouping of conceptually related entities. Examples include groups of entities describing the trajectory of a moving object, such as a person, migrating bird, or ship, or information about an organization, such as its location, the people involved, the things it produces, etc. These

recurring concepts are often encoded as Ontology Design Patterns [Gangemi 2005]. An ODP is a self-contained partial ontology. It represents the core components of the concept it seeks to model, as identified by domain experts. ODPs avoid making any unnecessary ontological commitments in order to remain applicable in a diverse range of situations. ODPs can facilitate alignment by reducing the complexity inherent in dealing with two ontologies likely developed by different designers with different applications in mind. Rather than trying to align these two ontologies directly with one another, they can instead each be aligned to the application-neutral ODP. Such an approach is more likely to involve mappings at the less complex end of the spectrum presented in Figure 1.4. Furthermore, leveraging ODPs can enhance the scalability of an alignment algorithm by clustering entities into different ODPs. Individual entity relations would then need to be determined only within those in an ODP rather than across the whole of the ontologies.

We would also like to move towards applying our ontology alignment capabilities to several real-world applications. In particular, we will soon begin work on a GeoLink project that involves bringing together a wide variety of Earth science data and making it available to researchers from a single interface. This project will allow wider dissemination of Earth science research results, new insights through analysis of data in unexpected ways, and avoidance of costly repetition of experiments. We also plan to use the similarity metrics developed in this research in a linked data visualization system designed to allow users to interactively formulate SPARQL queries over multiple linked datasets. Similarity metrics will allow conceptually similar entities from different datasets to be placed near one another in the visualization. The similarity values can also be used to cluster entities at different levels of abstraction, which makes it easier for users to navigate large datasets and drill-down on topics of interest. We are also interested in applying ontology alignment techniques to issues related to the privacy concerns of Big Data. Currently, many linked datasets are anonymized before being made available on the Semantic Web. This anonymization process often involves ensuring k-anonymity, which requires that at least k individuals have all possible combinations of pseudo-identifier characteristics [Sweeney 2002]. For instance, if the dataset contains information about people's voting district, gender, and birth month and year, at least k people would be required to have all combinations of these attributes (if not, either fake data is added or the information is made more coarse, e.g. by providing only birth year rather than month and year). As the dimensionality of data increases (i.e. more features are available for each person), k-anonymity breaks down [Ohm 2009]. Often this happens when new datasets are released that can be joined with existing datasets through some public fields. Ontology alignment facilitates this process of joining different datasets. We would like to apply an alignment system to the problem of de-anonymizing data and explore new anonymization strategies that are resistant to this approach. Through these applications, we

hope to gain a greater insight into the practical requirements for ontology alignment systems and establish new techniques and methods that move the field forward.

Appendices

A

OAEI String Metric Survey

Algorithm	Lexical Metrics	Preprocessing
AgreementMaker [Sunna and Cruz 2007; Cruz et al. 2009; Cruz et al. 2010; Cruz et al. 2011]	edit distance, LCS, TF-IDF	stemming, stop words, synonyms, normalization
Anchor-flood [Seddiqui and Aono 2008; 2009]	JaroWinker, SMOA, SM	tokenization, abbrev/acronym expansion, stop words, categorization (WordNet)
AROMA [David 2008; 2011]	JaroWinkler, exact match	stemming
ASMOV [Jean-Mary et al. 2010]	exact match, Levenstein, set similarity metric for comments	tokenization, normalization, synonyms, part-of-speech
AUTOMS [Kotis et al. 2006]	COCLU	
BLOOMS [Pesquita et al. 2010]	exact match, Jaccard, evidence content	tokenization, stop words, synonyms, normalization, stemming, spelling variants (proposed)
CIDER [Gracia and Mena 2008; Gracia et al. 2011]	exact match, Levenstein	normalization, synonyms

Cluster-based similarity aggregation [Tran et al. 2011]	edit distance, TF-IDF/cosine similarity	
CODI [Noessner and Niepert 2010; Huber et al. 2011]	Levenstein, cosine, Jaro-Winker, Smith-Waterman Gotoh, overlap coefficient, Jaccard	tokenization, normalization, stop words
COMA++ [Massmann et al. 2006]		
Cross-lingual Dutch to English alignment [Bouma 2009]		stemming, split compound words, synonyms
DSSim [Nagy et al. 2006; 2007; Nagy et al. 2008]	Monge-Elkan, Jaccard, Jaro-Winkler	synonyms, split compound words, language tag, translations, abbrev expansion
Eff2Match [Chua and Kim 2010]	exact match	tokenization, stemming, synonyms, normalization
Falcon-AO/ObjectCoref [Hu et al. 2006; Hu et al. 2007; Hu et al. 2010]	SMOA, edit distance	synonyms (proposed), translations (proposed)
GeRoMeSuite/SMB [Quix et al. 2008; Quix et al. 2010]	Levenstein, Jaro-Winkler, SMOA, soft TF-IDF	synonyms
HMatch [Castano et al. 2006]		
Hybrid alignment strategy for anatomical ontologies [Zhang and Bodenreider 2007]	exact match	synonyms, normalization
JHU/APL Onto-Mapology [Bethea et al. 2006]	Jaro-Winkler, 2-gram, document indexing	stop words
KOSIMap [Reul and Pan 2009]	Jaro-Winkler, Q-gram, SMOA, Monge-Elkan	stop words, stemming

LDOA [Kachroudi et al. 2011]	Levenstein, Jaro-Winkler, soft Jaccard	
LILY [Wang and Xu 2007; 2008; Wang 2011]	Levenstein, edit distance	
LN2R [Saıs et al. 2010]	soft Jaccard	synonyms (proposed)
LogMap [Jiménez-Ruiz et al. 2011]	exact match, SMOA	synonyms, alternate words forms (i.e. reverse stemming)
MaasMatch [Schadd and Roos 2011]	document vectors	stemming, stop words
MapPSO [Bock et al. 2009; Bock et al. 2011]	SMOA, TF-IDF	
NBJLM [Wang et al. 2010]	exact match	synonyms
NLM Anatomical Ontology Alignment System [Zhang and Bodenreider 2006]	exact match	normalization, synonyms, stemming
OACAS [Zghal et al. 2011]	Levenstein, Q-gram, Jaro-Winkler	
OKKAM [Stoermer and Rassadko 2009]	Levenstein	
OLA [Djoufak-Kengue et al. 2007]	LCS, normalized Hamming distance, edit distance	tokenization, synonyms
OMReasoner [Shen et al. 2011]	edit distance, prefix, suffix	split compound words (proposed)
OntoDNA [Kiu and Lee 2007]	Levenstein	normalization, stop words
Optima [Thayasivam and Doshi 2011]	Smith-Waterman	

OWL-CM [Yaghlane and Laamari 2007]	edit distance	
OWL-CtxMatch [Niedbała 2006]		synonyms
PRIOR/PRIOR+ [Mao and Peng 2006; 2007]	cosine similarity, Levenstein, document indexing	synonyms (proposed), translations (proposed)
RiMOM [Li et al. 2006; Li et al. 2009; Zhang et al. 2008; Wang et al. 2010]	edit distance, KNN, TF-IDF/cosine distance	tokenization, stemming, stop words, synonyms
SAMBO/SAMBOdtf [Tan and Lambrix 2007; Lambrix et al. 2008]	n-gram, edit distance	tokenization, stemming, stop words, synonyms
SEMA [Spiliopoulos et al. 2007]	COCLU	synonyms
SERIMI [Araujo et al. 2011]	RWSA	tokenization, normalization
SILAS [Ossewaarde 2007]	exact match	
SOBOM [Xu et al. 2010]	edit distance, SMOA	
SODA [Zghal et al. 2007]	Jaro-Winkler, Monge-Elkan	
Spider [Sabou and Gracia 2008]		
TaxoMap [Zargayouna et al. 2007; Hamdi et al. 2008; Hamdi et al. 2009; Hamdi et al. 2010]	Lin's similarity measure, exact match, substring inclusion	stop words, part-of-speech, translation, synonyms (proposed)
X-SOM [Curino et al. 2007]	Jaro, Levenstein	

YAM++ [Ngo et al. 2011]	Levenstein, Smith-Waterman, Jaro, Jaro-Winkler, Monge-Elkan, prefix, suffix, LCS, SMOA	
Zhishi.links [Niu et al. 2011]	exact match	

B

YAGO-DBPedia Entity Relations

YAGO entity	DBPedia entity	Relationship
hasWikipediaAbstract	abstract	3
directed	director	3
subjectEndRelation	skos/core#subject	3
isLeaderOf	leader	1
isLeaderOf	leadership	1
hasNumberOfWikipediaLinks	numberOfRooms	4
hasNumberOfWikipediaLinks	numberOfNeighbourhood	4
hasLanguageCode	languageCode	1
hasLanguageCode	fdaUniiCode	4
hasLanguageCode	millsCodeBE	4
hasGeoCoordinates	capitalCoordinates	3
rdf-schema#subClassOf	class	2
permanentRelationToObject	projectObjective	3
objectEndRelation	relation	2
created	related	4
hasAirportCode	targetAirport	3
hasNeighbor	neighbourhood	3
hasNeighbor	neighboringMunicipality	3
diedOnDate	deathDate	1
participatedIn	participant	3
_hasTypeCheckPattern	canBaggageChecked	4
hasPopulationDensity	PopulatedPlace/populationDensity	1

hasAcademicAdvisor	parent	4
hasAcademicAdvisor	doctoralAdvisor	3
hasAcademicAdvisor	academicAdvisor	1
hasWebsite	websiteLabel	3
rdf-schema#domain	domain	1
placedIn	placeOfBurial	4
hasArea	area	1
hasArea	PopulatedPlace/area	1
hasArea	Lake/areaOfCatchment	3
hasGivenName	foaf/0.1/givenName	1
playsFor	plays	3
livesIn	residence	1
rdf-schema#range	range	1
isCitizenOf	citizenship	1
isCitizenOf	stateOfOrigin	1
hasDuration	duration	1
hasDuration	SpaceMission/stationVisitDuration	3
isKnownFor	knownFor	1
hasBudget	budget	1
rdf-syntax-ns#type	type	1
hasNumberOfWikipediaLinks	numberOfSportsEvents	4
isMarriedTo	spouse	1
isMarriedTo	parent	4
hasOfficialLanguage	officialLanguage	1
hasOfficialLanguage	officialName	4
hasOfficialLanguage	language	2
hasPopulationDensity	GeopoliticalOrganisation/populationDensity	1
hasHeight	Person/height	1
hasHeight	MeanOfTransportation/height	1
hasHeight	AutomobileEngine/height	1
objectStartRelation	relation	1
skos/core#prefLabel	prefectMandate	4
skos/core#prefLabel	prefect	4
skos/core#prefLabel	prefix	4

hasPredecessor	predecessor	1
hasLatitude	geo/wgs84_pos#lat	3
hasChild	child	1
hasChild	rightChild	3
isAffiliatedTo	affiliate	3
hasWeight	AutomobileEngine/weight	1
hasWeight	Person/weight	1
hasPages	wikiPageID	4
hasPages	grayPage	4
hasSuccessor	successor	1
wasBornOnDate	birthDate	1
extractionSource	source	3
graduatedFrom	almaMater	4
wasCreatedOnDate	related	4
hasWeight	MeanOfTransportation/weight	1
hasWeight	Weapon/weight	1
hasISBN	isbn	1
hasCapital	capital	1
hasTitleText	title	1
hasTitleText	titleDate	4
hasMotto	motto	1
hasLength	Weapon/length	1
hasLength	MeanOfTransportation/length	1
hasLength	Infrastructure/length	1
hasThreeLetterLanguageCode	languageCode	2
hasThreeLetterLanguageCode	licenceLetter	4
rdf-schema#comment	rdf-schema#comment	1
rdf-schema#comment	comment	1
_timeToLocation	location	3
hasRevenue	revenue	1
hasWikipediaCategory	category	2
hasMusicalRole	role	2
hasExternalWikipediaLinkTo	linkedSpace	3
hasExternalWikipediaLinkTo	wikiPageExternalLink	1

hasLength	AutomobileEngine/length	1
hasGini	giniCoefficientAsOf	3
happenedIn	place	3
hasGeonamesClassId	classes	2
hasConfidence	sourceConfluence	4
influences	influenced	3
influences	influencedBy	1
actedIn	dateAct	3
rdf-syntax-ns#type	spurType	2
isPoliticianOf	belgiumPoliticalSeats	4
isPoliticianOf	politicalPartyInLegislature	3
isLocatedIn	locatedInArea	3
endedOnDate	endDate	1
endedOnDate	productionEndDate	3
hasHeight	Weapon/height	1
hasNumberOfPeople	numberOfStaff	3
linksTo	regionLink	3
startedOnDate	startDate	1
startedOnDate	startReign	1
startedOnDate	startCareer	3
startedOnDate	activeYearsStartDate	3
hasCurrency	currency	1
placedIn	place	3
hasImdb	imdbId	3
hasCitationTitle	title	2
hasLongitude	geo/wgs84_pos#long	4
relationLocatedByObject	relation	3
hasContext	continentRank	4
hasContext	continent	4
hasFamilyName	family	3
owns	ons	4
hasWikipediaArticleLength	mayorArticle	4
rdf-schema#label	rdf-schema#label	1
hasNumberOfPeople	numberOfRooms	4

hasNumberOfPeople	numberOfFilms	4
hasNumberOfPeople	numberOfSportsEvents	4
permanentRelationToSubject	subjectOfPlay	4
hasGender	gender	1
subjectStartRelation	skos/core#subject	2
extractionTechnique	technique	2
worksAt	work	3
isInterestedIn	interest	3
isInterestedIn	restingPlacePosition	4

References

Araujo, S., de Vries, A., and Schwabe, D. 2011. Serimi results for oaei 2011. *Ontology Matching*, 212.

Ashburner, M. et al. 2000. Gene Ontology: tool for the unification of biology. *Nature Genetics 25,* 1, 25–29.

Berners-Lee, T., Fischetti, M., and Foreword By-Dertouzos, M. L. 2000. *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. HarperInformation.

Bethea, W. L., Fink, C., and Beecher-Deighan, J. 2006. Jhu/apl onto-mapology results for oaei 2006. In *Ontology Matching*. 144.

Bock, J., Danschel, C., and Stumpp, M. 2011. Mappso and mapevo results for oaei 2011. In *Proc. 6th ISWC workshop on ontology matching (OM), Bonn (DE)*. 179–183.

Bock, J., Liu, P., and Hettenhausen, J. 2009. Mappso results for oaei 2009. In *OM*. Citeseer.

Bouma, G. 2009. Cross-lingual dutch to english alignment using eurowordnet and dutch wikipedia. In *Proceedings of the 4th International Workshop on Ontology Matching, CEUR-WS*. Vol. 551. Citeseer, 224–229.

Branting, L. K. 2003. A comparative evaluation of name-matching algorithms. In *Proceedings of the 9th International Conference on Artificial Intelligence and Law*. ACM, 224–232.

Castano, S., Ferrara, A., and Messa, G. 2006. Results of the hmatch ontology matchmaker in oaei 2006. In *Ontology Matching*. 134.

Cheatham, M. and Hitzler, P. 2013. String similarity metrics for ontology alignment. In *The Semantic Web–ISWC 2013*. Springer, 294–309.

Chua, W. W. K. and Kim, J.-J. 2010. Eff2match results for oaei 2010. *Ontology Matching*, 150.

Cohen, W. W., Ravikumar, P., Fienberg, S. E., et al. 2003. A comparison of string distance metrics for name-matching tasks. In *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03)*. Vol. 47.

Cruz, I., Antonelli, F. P., Stroe, C., Keles, U. C., and Maduko, A. 2009. Using agreementmaker to align ontologies for oaei 2009: Overview, results, and outlook. In *Proceedings of the ISWC 2009 Workshop on Ontology Matching*. Citeseer, 135–146.

Cruz, I. F., Stroe, C., Caci, M., Caimi, F., Palmonari, M., Antonelli, F. P., and Keles, U. C. 2010. Using agreementmaker to align ontologies for oaei 2010. In *ISWC International Workshop on Ontology Matching (OM). CEUR Workshop Proceedings*. Vol. 689. 118–125.

Cruz, I. F., Stroe, C., Caimi, F., Fabiani, A., Pesquita, C., Couto, F. M., and Palmonari, M. 2011. Using agreementmaker to align ontologies for oaei 2011? In *ISWC international workshop on ontology matching (OM)*. Vol. 814. 114–121.

Curino, C., Orsi, G., and Tanca, L. 2007. X-som results for oaei 2007. In *Proceedings of the Second International Workshop on Ontology Matching*. Citeseer, 276–285.

David, J. 2008. Aroma results for oaei 2008. In *Proc. 3rd ISWC workshop on ontology matching (OM)*. 128–131.

David, J. 2011. Aroma results for oaei 2011. *Ontology Matching*, 122.

Dekhtyar, A. and Hayes, J. H. 2006. Good benchmarks are hard to find: Toward the benchmark for information retrieval applications in software engineering.

Di Martino, B. 2009. Semantic web services discovery based on structural ontology matching. *International Journal of Web and Grid Services 5,* 1, 46–65.

Djoufak-Kengue, J.-F., Euzenat, J., Valtchev, P., et al. 2007. Ola in the oaei 2007 evaluation contest. In *Proc. 2nd ISWC 2007 workshop on ontology matching (OM)*. 188–195.

Duan, S., Fokoue, A., Hassanzadeh, O., Kementsietsidis, A., Srinivas, K., and Ward, M. J. 2012. Instance-based matching of large ontologies using locality-sensitive hashing. In *The Semantic Web–ISWC 2012*. Springer, 49–64.

Durbin, R. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.

Euzenat, J. et al. 2004. State of the art on ontology alignment. *Knowledge Web Deliverable D 2*, 2–3.

Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., and Trojahn, C. 2011. Ontology alignment evaluation initiative: Six years of experience. In *Journal on Data Semantics XV*. Springer, 158–192.

Euzenat, J. and Shvaiko, P. 2007. *Ontology matching*. Vol. 18. Springer Heidelberg.

Gallagher, B. 2006. Matching structure and semantics: A survey on graph-based pattern matching. *AAAI FS 6*, 45–53.

Gangemi, A. 2005. Ontology design patterns for semantic web content. In *The Semantic Web–ISWC 2005*. Springer, 262–276.

Gaudan, S., Yepes, A. J., Lee, V., Rebholz-Schuhmann, D., et al. 2008. Combining evidence, specificity, and proximity towards the normalization of gene ontology terms in text. *EURASIP Journal on Bioinformatics and Systems Biology 2008*.

Gotoh, O. 1982. An improved algorithm for matching biological sequences. *Journal of Molecular Biology 162,* 3, 705–708.

Gracia, J., Bernad, J., and Mena, E. 2011. Ontology matching with cider: evaluation report for oaei 2011. *Ontology Matching*, 126.

Gracia, J. and Mena, E. 2008. Matching with cider: Evaluation report for the oaei 2008. In *3rd Ontology Matching Workshop (OM?08) at the 7th International Semantic Web Conference (ISWC?08), Karlsruhe, Germany*.

Gruber, T. R. 1993. A translation approach to portable ontology specifications. *Knowledge acquisition 5,* 2, 199–220.

Gunaratna, K., Thirunarayan, K., Jain, P., Sheth, A., and Wijeratne, S. 2013. A statistical and schema independent approach to identify equivalent properties on linked data. In *Proceedings of the 9th International Conference on Semantic Systems*. ACM, 33–40.

Hamdi, F., Safar, B., Niraula, N., Reynaud, C., et al. 2009. Taxomap in the oaei 2009 alignment contest. In *The Fourth International Workshop on Ontology Matching*.

Hamdi, F., Safar, B., Niraula, N. B., and Reynaud, C. 2010. Taxomap alignment and refinement modules: Results for oaei 2010. In *Proceedings of the 5th International Workshop on Ontology Matching (OM-2010) Collocated with the 9th International Semantic Web Conference (ISWC-2010), CEUR-WS*. 212–220.

Hamdi, F., Zargayouna, H., Safar, B., and Reynaud, C. 2008. Taxomap in the oaei 2008 alignment contest, ontology alignment evaluation initiative (oaei) 2008 campaign-int. In *Workshop on Ontology Matching*.

Hartung, M., Kolb, L., Gross, A., and Rahm, E. 2013. Optimizing similarity computations for ontology matching-experiences from gomma. In *Data Integration in the Life Sciences*. Springer, 81–89.

Hitzler, P., Krotzsch, M., and Rudolph, S. 2011. *Foundations of semantic web technologies*. CRC Press.

Hu, W., Chen, J., Cheng, G., and Qu, Y. 2010. Objectcoref & falcon-ao: results for oaei 2010. *Ontology Matching*, 158.

Hu, W., Cheng, G., Zheng, D., Zhong, X., and Qu, Y. 2006. The results of falcon-ao in the oaei 2006 campaign. In *Ontology Matching*. 124.

Hu, W., Zhao, Y., Li, D., Cheng, G., Wu, H., and Qu, Y. 2007. Falcon-ao: Results for oaei 2007. In *OM*.

Huber, J., Sztyler, T., Noessner, J., and Meilicke, C. 2011. Codi: Combinatorial optimization for data integration–results for oaei 2011. *Ontology Matching*, 134.

Huppler, K. 2009. The art of building a good benchmark. In *Performance Evaluation and Benchmarking*. Springer, 18–30.

Ipeirotis, P. G. 2010. Demographics of mechanical turk.

Jain, P., Hitzler, P., Sheth, A. P., Verma, K., and Yeh, P. Z. 2010. Ontology alignment for linked open data. In *The Semantic Web–ISWC 2010*. Springer, 402–417.

Jean-Mary, Y. R., Shironoshita, E. P., and Kabuka, M. R. 2010. Asmov: Results for oaei 2010. *Ontology Matching 126*.

Jiménez-Ruiz, E., Morant, A., and Grau, B. C. 2011. Logmap results for oaei 2011. *Ontology Matching*, 163.

Kachroudi, M., Moussa, E. B., Zghal, S., and Ben, S. 2011. Ldoa results for oaei 2011. *Ontology Matching*, 148.

Kiu, C.-C. and Lee, C.-S. 2007. Ontodna: Ontology alignment results for oaei 2007. In *OM*.

Kotis, K., Valarakos, A. G., and Vouros, G. A. 2006. Automs: Automated ontology mapping through synthesis of methods. In *Ontology Matching*. 96.

Lambrix, P., Tan, H., and Liu, Q. 2008. Sambo and sambodtf results for the ontology alignment evaluation initiative 2008. In *Proceedings of the Third International Workshop on Ontology Matching*. 190–198.

Li, J., Tang, J., Li, Y., and Luo, Q. 2009. Rimom: A dynamic multistrategy ontology alignment framework. *Knowledge and Data Engineering, IEEE Transactions on 21,* 8, 1218–1232.

Li, Y., Li, J.-Z., Zhang, D., and Tang, J. 2006. Result of ontology alignment with rimom at oaei'06. In *Ontology Matching*. 181.

Lin, D. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*. Vol. 1. San Francisco, 296–304.

Madhavan, J., Bernstein, P. A., and Rahm, E. 2001. Generic schema matching with cupid. In *VLDB*. Vol. 1. 49–58.

Maedche, A. and Staab, S. 2002. Measuring similarity between ontologies. In *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*. Springer, 251–263.

Mao, M. and Peng, Y. 2006. Prior system: Results for oaei 2006. In *Ontology Matching*. 173.

Mao, M. and Peng, Y. 2007. The prior+: Results for oaei campaign 2007. In *OM*.

Massmann, S., Engmann, D., and Rahm, E. 2006. Coma++: Results for the ontology alignment contest oaei 2006. In *Ontology Matching*. 107.

Meilicke, C. 2011. Alignment incoherence in ontology matching. Ph.D. thesis, Universitätsbibliothek Mannheim.

Melnik, S., Garcia-Molina, H., and Rahm, E. 2002. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proc. 18th ICDE Conf.(Best Student Paper award)*.

Monge, A. E. and Elkan, C. 1996. The field matching problem: Algorithms and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. 267–270.

Mortensen, J. M. 2013. Crowdsourcing ontology verification. In *The Semantic Web–ISWC 2013*. Springer, 448–455.

Mortensen, J. M., Musen, M. A., and Noy, N. F. 2013a. Crowdsourcing the verification of relationships in biomedical ontologies. In *AMIA Annual Symposium (submitted, 2013)*.

Mortensen, J. M., Musen, M. A., and Noy, N. F. 2013b. Ontology quality assurance with the crowd. In *First AAAI Conference on Human Computation and Crowdsourcing*.

Nagy, M., Vargas-Vera, M., and Motta, E. 2006. Dssim-ontology mapping with uncertainty.

Nagy, M., Vargas-Vera, M., and Motta, E. 2007. Dssim-managing uncertainty on the semantic web.

Nagy, M., Vargas-Vera, M., Stolarski, P., and Motta, E. 2008. Dssim results for oaei 2008. In *Proceedings of the Third International Workshop on Ontology Matching*. 147–159.

Ngo, D. H., Bellahsene, Z., Coletta, R., et al. 2011. Yam++–results for oaei 2011. In *ISWC'11: The 6th International Workshop on Ontology Matching*. Vol. 814. 228–235.

Nguyen, H. T. and Walker, E. A. 2005. *A First Course in Fuzzy Logic*, 3rd ed. Chapman and Hall / CRC.

Niedbała, S. 2006. Owl ctxmatch in the oaei 2006 alignment contest. *Ontology Matching*, 165.

Niu, X., Rong, S., Zhang, Y., and Wang, H. 2011. Zhishi. links results for oaei 2011. *Ontology Matching*, 220.

Noessner, J. and Niepert, M. 2010. Codi: Combinatorial optimization for data integration–results for oaei 2010. *Ontology Matching*, 142.

Noy, N. F. and Hafner, C. D. 1997. The state of the art in ontology design: A survey and comparative review. *AI magazine 18,* 3, 53.

Noy, N. F., Mortensen, J., Musen, M. A., and Alexander, P. R. 2013. Mechanical turk as an ontology engineer?: using microtasks as a component of an ontology-engineering workflow. In *Proceedings of the 5th Annual ACM Web Science Conference*. ACM, 262–271.

Ohm, P. 2009. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA L. Rev. 57*, 1701.

Ossewaarde, R. 2007. Simple library thesaurus alignment with silas. In *OM*.

Pernelle, N., Saïs, F., Safar, B., Koutraki, M., and Ghosh, T. 2013. N2r-part: identity link discovery using partially aligned ontologies. In *Proceedings of the 2nd International Workshop on Open Data*. ACM, 6.

Pesquita, C., Faria, D., Santos, E., and Couto, F. M. 2013. To repair or not to repair: reconciling correctness and coherence in ontology reference alignments. In *OM*. 13–24.

Pesquita, C., Stroe, C., Cruz, I., and Couto, F. M. 2010. Blooms on agreementmaker: results for oaei 2010. *Ontology Matching*, 134.

Porter, M. F. 1980. An algorithm for suffix stripping. *Program: Electronic Library and Information Systems 14,* 3, 130–137.

Quix, C., Gal, A., Sagi, T., and Kensche, D. 2010. An integrated matching system: Geromesuite and smb–results for oaei 2010. *Ontology Matching*, 166.

Quix, C., Geisler, S., Kensche, D., and Li, X. 2008. Results of geromesuite for oaei 2008. In *Proceedings of the Third International Workshop on Ontology Matching*. 160–166.

Reul, Q. and Pan, J. Z. 2009. Kosimap: ontology alignments results for oaei 2009. In *Proceedings of the ISWC 2009 Workshop on Ontology Matching*. Citeseer, 177–185.

Ritze, D., Meilicke, C., Sváb-Zamazal, O., and Stuckenschmidt, H. 2009. A pattern-based ontology matching approach for detecting complex correspondences. In *ISWC Workshop on Ontology Matching, Chantilly (VA US)*. Citeseer, 25–36.

Rosoiu, M., dos Santos, C. T., Euzenat, J., et al. 2011. Ontology matching benchmarks: generation and evaluation. In *Proc. 6th ISWC workshop on ontology matching (OM)*. 73–84.

Sabou, M. and Gracia, J. 2008. Spider: Bringing non-equivalence mappings to oaei. In *Proceedings of the Third International Workshop on Ontology Matching*.

Sais, F., Niraula, N., Pernelle, N., and Rousset, M.-C. 2010. Ln2r–a knowledge based reference reconciliation system: Oaei 2010 results. *Ontology Matching*, 172.

Santos, E., Faria, D., Pesquita, C., and Couto, F. 2013. Ontology alignment repair through modularization and confidence-based heuristics. *arXiv preprint arXiv:1307.5322*.

Sarasua, C., Simperl, E., and Noy, N. F. 2012. Crowdmap: Crowdsourcing ontology alignment with microtasks. In *The Semantic Web–ISWC 2012*. Springer, 525–541.

Schadd, F. C. and Roos, N. 2011. Maasmatch results for oaei 2011. *Ontology Matching*, 171.

Seddiqui, M. H. and Aono, M. 2008. Alignment results of anchor-flood algorithm for oaei-2008. In *Proceedings of Ontology Matching Workshop of the 7th International Semantic Web Conference, Karlsruhe, Germany*. 120–127.

Seddiqui, M. H. and Aono, M. 2009. Anchor-flood: results for oaei 2009. In *Proceedings of the ISWC 2009 Workshop on Ontology Matching*. Citeseer, 127–134.

Shen, G., Jin, L., Zhao, Z., Jia, Z., He, W., and Huang, Z. 2011. Omreasoner: using reasoner for ontology matching: results for oaei 2011. *Ontology Matching*, 197.

Sim, S. E., Easterbrook, S., and Holt, R. C. 2003. Using benchmarking to advance research: A challenge to software engineering. In *Proceedings of the 25th International Conference on Software Engineering*. IEEE Computer Society, 74–83.

Spiliopoulos, V., Valarakos, A. G., Vouros, G. A., and Karkaletsis, V. 2007. Sema: Results for the ontology alignment contest oaei 2007. In *OM*.

Spiliopoulos, V., Vouros, G. A., and Karkaletsis, V. 2010. On the discovery of subsumption relations for the alignment of ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web 8,* 1, 69–88.

Stoermer, H. and Rassadko, N. 2009. Results of okkam feature based entity matching algorithm for instance matching contest of oaei 2009. *Ontology Matching*, 200.

Stoilos, G., Stamou, G., and Kollias, S. 2005. A string metric for ontology alignment. In *The Semantic Web–ISWC 2005*. Springer, 624–637.

Suchanek, F., Abiteboul, S., and Senellart, P. 2011a. Ontology alignment at the instance and schema level. *arXiv preprint arXiv:1105.5516*.

Suchanek, F. M., Abiteboul, S., and Senellart, P. 2011b. Paris: Probabilistic alignment of relations, instances, and schema. *Proceedings of the VLDB Endowment 5,* 3, 157–168.

Sunna, W. and Cruz, I. F. 2007. Using the agreementmaker to align ontologies for the oaei campaign 2007. In *OM*. Citeseer.

Šváb, O. 2007. Exploiting patterns in ontology mapping. In *The Semantic Web*. Springer, 956–960.

Šváb, O., Svátek, V., Berka, P., Rak, D., and Tomášek, P. 2005. Ontofarm: Towards an experimental collection of parallel ontologies. *Poster Track of ISWC 2005*.

Sváb-Zamazal, O., Svátek, V., Scharffe, F., et al. 2009. Pattern-based ontology transformation service. In *Proc. 1st IK3C international conference on knowledge engineering and ontology development (KEOD)*. 210–223.

Svátek, V., Šváb-Zamazal, O., and Presutti, V. 2009. Ontology naming pattern sauce for (human and computer) gourmets. In *Workshop on Ontology Patterns*. 171–178.

Sweeney, L. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10,* 05, 557–570.

Tan, H. and Lambrix, P. 2007. Sambo results for the ontology alignment evaluation initiative 2007. In *OM*.

Taylor, J. M., Poliakov, D., and Mazlack, L. J. 2005. Domain-specific ontology merging for the semantic web. In *Fuzzy Information Processing Society, 2005. NAFIPS 2005. Annual Meeting of the North American*. IEEE, 418–423.

Thayasivam, U. and Doshi, P. 2011. Optima results for oaei 2011. In *Proc. of 6th OM Workshop*. 204–211.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 173–180.

Tran, Q.-V., Ichise, R., and Ho, B.-Q. 2011. Clusterbased similarity aggregation for ontology matching. In *Proc. 6th ISWC workshop on ontology matching (OM), Bonn (DE)*. 142–147.

Ul Hassan, U., ORiain, S., and Curry, E. 2012. Towards expertise modelling for routing data cleaning tasks within a community of knowledge workers. In *Proceedings of the 17th International Conference on Information Quality*.

Valarakos, A. G., Paliouras, G., Karkaletsis, V., and Vouros, G. 2004. A name-matching algorithm for supporting ontology enrichment. In *Methods and Applications of Artificial Intelligence*. Springer, 381–389.

Wang, P. 2011. Lily results on seals platform for oaei 2011. In *Proc. of 6th OM Workshop*. 156–162.

Wang, P. and Xu, B. 2007. Lily: the results for the ontology alignment contest oaei 2007. In *Proceedings of the Second International Workshop on Ontology Matching*. Citeseer, 179–187.

Wang, P. and Xu, B. 2008. Lily: Ontology alignment results for oaei 2008. In *Proceedings of the Third International Workshop on Ontology Matching*. 167–175.

Wang, S., Wang, G., and Liu, X. 2010. Results of nbjlm for oaei 2010. *Ontology Matching*, 187.

Wang, Z., Zhang, X., Hou, L., Zhao, Y., Li, J., Qi, Y., and Tang, J. 2010. Rimom results for oaei 2010. *Ontology Matching 195*.

Wichmann, P., Borek, A., Kern, R., Woodall, P., Parlikad, A. K., and Satzger, G. 2011. Exploring the crowdas enabler of better information quality. In *Proceedings of the 16th International Conference on Information Quality*. 302–312.

Xu, P., Wang, Y., Cheng, L., and Zang, T. 2010. Alignment results of sobom for oaei 2010. In *OM*.

Yaghlane, B. B. and Laamari, N. 2007. Owl-cm: Owl combining matcher based on belief functions theory. In *OM*. Citeseer.

Zargayouna, H., Safar, B., Reynaud, C., et al. 2007. Taxomap in the oaei 2007 alignment contest. In *Proceedings of The Second International Workshop on Ontology Matching (OM'07)*. 268–275.

Zghal, S., Kachroudi, M., Yahia, S. B., and MEPHU, E. 2011. Oacas: results for oaei 2011. *Ontology Matching*, 190.

Zghal, S., Yahia, S. B., Nguifo, E. M., Slimani, Y., et al. 2007. Soda: an owl-dl based ontology matching system. In *OM*.

Zhang, S. and Bodenreider, O. 2006. Nlm anatomical ontology alignment system. results of the 2006 ontology alignment contest. In *Ontology Matching*. 153.

Zhang, S. and Bodenreider, O. 2007. Hybrid alignment strategy for anatomical ontologies: Results of the 2007 ontology alignment contest. In *OM*.

Zhang, X., Zhong, Q., Li, J., Tang, J., Xie, G., and Li, H. 2008. Rimom results for oaei 2008. In *Proceedings of the 3rd International Workshop on Ontology Matching*. Vol. 431. 182.

Zhao, L. and Ichise, R. 2013. Instance-based ontological knowledge acquisition. In *The Semantic Web: Semantics and Big Data*. Springer, 155–169.