Jennifer Sleeman and Tim Finin, Cluster-based Instance Consolidation For Subsequent Matching, First International Workshop on Knowledge Extraction and Consolidation from Social Media, November 2012, Boston.

Cluster-based Instance Consolidation For Subsequent Matching

Jennifer Sleeman and Tim Finin

Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore. MD 21250 USA
jsleem1,finin@cs.umbc.edu

Abstract. Instance consolidation is a way to merge instances that are thought to be the same or closely related that can be used to support coreference resolution and entity linking. For Semantic Web data, consolidating instances can be as simple as relating instances using owl:sameAs, as is the case in linked data, or merging instances that could then be used to populate or enrich a knowledge model. In many applications, systems process data incrementally over time and as new data is processed, the state of the knowledge model changes. Previous consolidations could prove to be incorrect. Consequently, a more abstract representation is needed to support instance consolidation. We describe our current research to perform consolidation that includes temporal support, support to resolve conflicts and an abstract representation of an instance that is the aggregate of a cluster of matched instances. We believe that this model will prove flexible enough to handle sparse instance data and can improve the accuracy of the knowledge model over time.

1 Introduction

Though consolidation has been researched in other domains, such as the database domain, it is less explored in the Semantic Web domain. In relation to coreference resolution (also known as instance matching and entity resolution), once two instances or entities are designated as the same or coreferent, they are tagged in some way (using owl:sameAs) or consolidated into a single entity using various approaches. What has received less attention is how to merge instances with conflicted information and how to adapt consolidations over time. In this paper we describe our ongoing work that supports instance consolidation by grouping matched instances into clusters of abstract representations. We develop our consolidation algorithm to work with incremental online coreference resolution by providing a way to improve the instance data that will be used in subsequent matching. For example, in the case of sparse instances, a consolidated representation of features would be more likely to match newly discovered instances. As more instances are added to the cluster, the representation will become more enriched and more likely to match a wider number of instances in subsequent matches. When performing subsequent instance matching that includes both clusters and individual instances, the consolidated representation of clustered data, supported by our merging algorithm, can be used.

Figure 1 depicts an example of a consolidation when conflicts may occur. In this example, when we have a pair of attributes that are the same but their values differ, to consolidate we must determine whether both values are maintained, none of the values are maintained or one of the values is maintained. For the purposes of using the consolidated instance for future matching, the merging of instance data is incredibly important as it affects the performance of future matching. This is particularly true when working with data sets that are sparse.

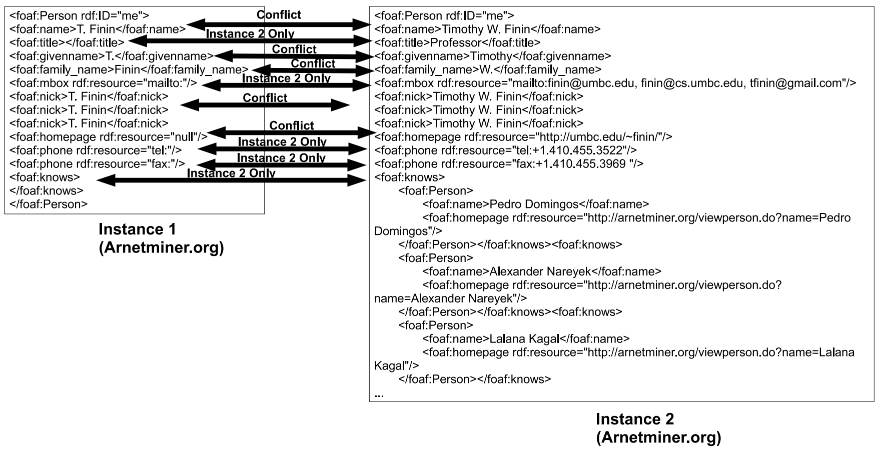

Fig. 1. Conflicts During A Merge

The temporal support is an important aspect to this problem since over time an entity's features may change. In Figure 2, the attribute population changes over time. This example highlights two complexities that are a natural effect of time. An instance can be thought of as a snapshot in time, therefore an instance captured at time $t-1$ may not be as relevant as an instance captured at time t. This affects how instances should be consolidated and is a good example of when a technique is required to resolve conflicts. Also, this implies that in certain cases, given enough time, two instances may no longer be coreferent, supporting the argument that temporal issues play a significant role in consolidation and subsequent processing.

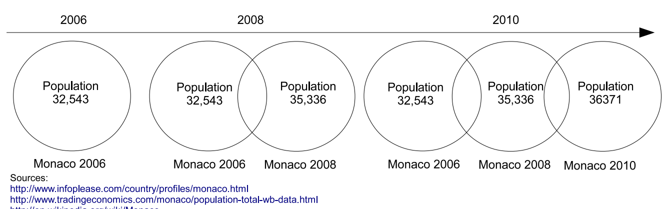

Fig. 2. Temporal Aspects of Consolidation

2 Background

Semantic Web data, which includes semantically tagged data represented using a Resource Description Framework (RDF) [1, 2] triples (subject, predicate, object) format, is often used as a way to commonly represent data. Data which conforms to an ontology, data exported from social networking sites, and linked data found on the Linked Open Data Cloud are often represented as triples. Attempting to match instances or entities among this type of data can be a challenge which is further complicated by noise and data spareness.

The topic of instance consolidation, the process of combining instances, is not novel. Previous research has addressed instance consolidation in relation to merging instances that are coreferent. What has received less attention is the temporal aspect of this problem, how to merge instances when conflicts are present and how using this method to support incremental coreference resolution can address issues related to spareness. For example, in Figure 3 we show two Friend of a Friend (FOAF) [3] documents representing a person. In the top document, information such as foaf:firstName, foaf:surname and foaf:name is absent. In the bottom document, these values exist and so a consolidation of these two documents would eliminate attributes that are missing values and increase the number of features that could be matched for subsequent matching.

```
<foaf:Person rdf:ID="jgolbeck">
<foaf:mbox_sha1sum>08445a31a78661b5c746feff39a9db6e4e2cc5cf</foaf:mbox_sha1sum>
<foaf:firstName></foaf:firstName> <foaf:surname></foaf:surname> <foaf:name> </foaf:name>
<foaf:homepage rdf:resource="http://trust.mindswap.org/cgi-bin/FilmTrust/foaf.cgi?user=jgolbeck"/>
<foaf:img rdf:resource=""/> <foaf:depiction rdf:resource=""/> <foaf:nick>jgolbeck</foaf:nick>
<foaf:holdsAccount> <foaf:OnlineAccount> <foaf:accountName>jgolbeck</foaf:accountName>
<foaf:accountServiceHomepage rdf:resource="http://trust.mindswap.org/FilmTrust/"/>
</foaf:OnlineAccount> </foaf:holdsAccount>
        http://trust.mindswap.org/cgi-bin/FilmTrust/foaf.cgi?user=jgolbeck
```

```
<swivt:Subject rdf:about="http://tw.rpi.edu/wiki/Special:URIResolver/Jennifer_Golbeck">
<rdfs:label>Jennifer Golbeck</rdfs:label>
<swivt:page rdf:resource="http://tw.rpi.edu/wiki/Jennifer_Golbeck"/>
<rdfs:isDefinedBy rdf:resource="http://tw.rpi.edu/wiki/Special:ExportRDF/Jennifer_Golbeck"/>
<rdf:type rdf:resource="http://tw.rpi.edu/wiki/Special:URIResolver/Category-
    3AAssistant_Professor"/>
<rdf:type rdf:resource="http://tw.rpi.edu/wiki/Special:URIResolver/Category-3APerson"/>
<property:Foaf-3Adepiction
    rdf:resource="http://tw.rpi.edu/wiki/Special:URIResolver/Anonymous.png"/>
<foaf:firstName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Jennifer</foaf:firstName>
<foaf:interest rdf:resource="http://tw.rpi.edu/wiki/Special:URIResolver/Category-
    3ASemantic_Web_Topic"/>
<foaf:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Jennifer Golbeck</foaf:name>
<foaf:surname rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Golbeck</foaf:surname>
                         http://tw.rpi.edu/wiki/Special:ExportRDF/Jennifer_Golbeck
```

Fig. 3. Consolidating Instances

The research that exists today, has a tendency to use a methodology that relies upon inverse functional properties. For example, Hogan et al.[4] use inverse functional properties to determine instances in common and rewrite identifiers based on each equivalence chain. They require retrieval of the ontologies to identify inverse functional properties, this is not always possible. They describe a merge process that assumes agreement, i.e. no conflicts and they do not address how to handle data that does not use inverse functional properties. Shi et al.[5] describe instance consolidation as ’smushing’ and performs ’smushing’ by taking advantage of the inverse functional property. They work at the attribute level and calculate attribute level similarity measures. A property defined as inverse functional implies that the inverse of the property is functional; that it uniquely identifies the subject[2]. Again this work relies upon inverse functional properties and tends not to address how to resolve conflicts. Yatskevich et al.[6] address consolidation of graphs. They merge graphs if the instances belong to the same class, and if their string similarity is higher than a threshold. They describe special cases for particular types. This merge process does not address conflicts and there is no indication whether they could reverse a consolidated graph. In our previous work [7, 8] that explored our approach using simple merging heuristics and coreferent clustering of FOAF instances, particularly when working with sparse input, consolidation did positively affect subsequent coreferent pairing.

In our person data set, specifically using the FOAF ontology, we found a sizable percentage of the instances contained very few attributes. In Table 1, we show the number of instances originating from 3 different sources. Source ’vox’ had the highest number of instances and also the lowest number of attributes per instance. We have found this is prevalent among social networking sites and sites that support exports of user profile data using the FOAF ontology. This is not specific to FOAF instances and can present a problem for coreference resolution algorithms.

Source	Avg Number of Attributes	Number of Instances
vox	5.65	4492
journal	9.71	1259
ebiquity	19.78	217

Table 1. Average Number of Attributes

3 An Approach

We define an instance as an abstract representation that can be either a cluster of coreferent instances, or a single entity instance. A formal definition follows.

Definition 1. *Given a set of instances I and a set of clusters C, an abstract instance $A \in (I \cup C)$.*

Definition 2. *Given a pair of instances i_n and i_m, if the pair are coreferent or $coref\ (i_n, i_m)$, then a cluster C_{nm} is formed such that the cluster $C_{nm} = \{i_n, i_m\}$.*

Figure 4 depicts an example of a cluster that is formed with coreferent instances. Data relates to Monaco from three different sources (http://dbpedia.org/, http://www4.wiwiss.fu-berlin.de/, http://data.nytimes.com/) where each source represents a perspective of Monaco. Given a system that processes unstructured text that includes a reference to the instance Monaco, our work seeks to prove that we are more likely to recognize Monaco as an entity with the combined information taking the most relevant of features, rather than

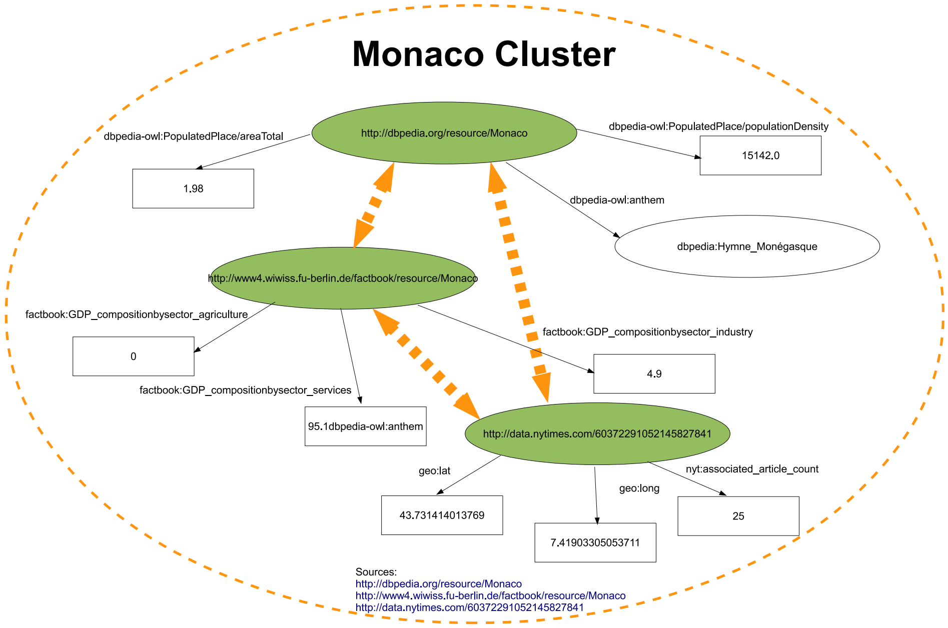

Fig. 4. Instance Cluster

using a single instance. We will also show how over time as attributes pertaining to these instances changes, the model can reflect these changes.
A consolidated representation is required in order to use the clustered data in subsequent matching. This consolidation can be as simple as a union between the sets of attributes of instances. However, as seen in Figure 1, this approach does not address situations where attributes are in conflict. Even in this simple example, conflicts are present that should be resolved. Our initial work includes the merging of instances and resolution of conflicts by using a basic set of rules and temporal information.

When evaluating two instances, for each attribute that is shared, if the values are equal, we retain one instance of the attribute. If two instances share the same attribute but their values differ, we try to resolve the conflict. If the two instances contain attributes that are not shared we include their unshared attributes. In resolving the conflict, we first try to determine if the two values are synonymous. If they are synonymous, we keep both values for the particular attribute. If not, then we will use additional analysis such as temporal information. As the same instance is processed over time given a particular URI, we track the changes among attributes for that instance. Given that attributes have changed for a particular instance we give the more recent values of the attributes a higher significance than a less recent values. We can then use this information to assist with resolving conflicts. When conflicts can not be resolved we keep both values for the unresolved attribute. We anticipate this approach will advance as we progress in our research.

Our cluster links are symbolic in nature. In order to support changes to the cluster over time, each instance in a cluster is linked and weighted to other instances in the cluster. How the weight is defined is based on the coreference resolution algorithm. In our work, we are using a clustering algorithm to cluster instances that are thought to be coreferent. The output of our calculation that supports our clustering method will also be used as an assessment of how closely the two instances are related for consolidation. Given the set of attributes for each instance in the cluster, we associate a score with each set of matched attributes. This score can be based on a distance function or based on a more complex representation. The goal of this step is to weight common features among pairs of coreferent instances in the cluster. Across all features in the cluster we wish to pick the most significant features to be used for subsequent matching. We are currently exploring feature reduction mechanisms to perform this step. This structure gives us the ability to compare coreferent relationships among instances over time, to remove coreferent relationships given changes over time, or to add and modify existing relationships given new instances that are added to the cluster.

4 Conclusion

We have presented a need for a more adaptive-based consolidation approach. A cluster-based consolidation provides a powerful model for instance matching algorithms. It is meant to adapt to change over time, is flexible and could potentially improve subsequent matching. Given the complexities of systems today, adaptation is a necessity. The challenge is developing a consolidation approach that is flexible enough to support the complexities of systems today, without incurring a large performance penalty.

References

1. Beckett, D.: Rdf/xml syntax specification. http://www.w3.org/TR/REC-rdf-syntax/ (2004)
2. Brickley, D., Guha, R.: Resource description framework (rdf) schema specification 1.0. http://www.w3.org/TR/rdf-schema/ (2004)
3. Brickley, D., Miller, L.: Foaf vocabulary specification .98 (August 2010) http://xmlns.com/foaf/spec/.
4. Hogan, A., Harth, A., Decker, S.: Performing object consolidation on the semantic web data graph. In: Proc. I3: Identity, Identifiers, Identification. Workshop at 16th Int. World Wide Web Conf. (February 2007)
5. Shi, L., Berrueta, D., Fernandez, S., Polo, L., Fernandez, S.: Smushing rdf instances: are alice and bob the same open source developer? In: Proc. 3rd Expert Finder workshop on Personal Identification and Collaborations: Knowledge Mediation and Extraction, 7th Int. Semantic Web Conf. (November 2008)
6. Yatskevich, M., Welty, C., Murdock, J.: Coreference resolution on rdf graphs generated from information extraction: first results. In: the ISWC 06 Workshop on Web Content Mining with Human Language Technologies. (2006)
7. Sleeman, J., Finin, T.: A machine learning approach to linking foaf instances. In: Spring Symposium on Linked Data Meets AI, AAAI (January 2010)
8. Sleeman, J., Finin, T.: Computing foaf co-reference relations with rules and machine learning. In: The Third International Workshop on Social Data on the Web, ISWC (November 2010)