LEARNING–BASED GEOSPATIAL SCHEMA MATCHING

GUIDED BY EXTERNAL KNOWLEDGE

by

Jeffrey Partyka

APPROVED BY SUPERVISORY COMMITTEE:

_____

Dr. Latifur Khan, Co-Chair

_____

Dr. Bhavani Thuraisingham, Co-Chair

_____

Dr. Murat Kantarcioglu

_____

Dr. Weili Wu

To my parents, family, and loved ones

LEARNING-BASED GEOSPATIAL SCHEMA MATCHING GUIDED BY EXTERNAL KNOWLEDGE

by

JEFFREY PARTYKA, B.S, M.S.

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

December 2011

PREFACE

This dissertation was produced in accordance with guidelines which permit the inclusion as part of the dissertation the text of an original paper or papers submitted for publication. The dissertation must still conform to all other requirements explained in the "Guide for the Preparation of Master's Theses and Doctoral Dissertations at The University of Texas at Dallas." It must include a comprehensive abstract, a full introduction and literature review and a final overall conclusion. Additional material (procedural and design data as well as descriptions of equipment) must be provided in sufficient detail to allow a clear and precise judgment to be made of the importance and originality of the research reported.

It is acceptable for this dissertation to include as chapters authentic copies of papers already published, provided these meet type size, margin and legibility requirements. In such cases, connecting texts which provide logical bridges between different manuscripts are mandatory. Where the student is not the sole author of a manuscript, the student is required to make an explicit statement in the introductory material to that manuscript describing the student's contribution to the work and acknowledging the contribution of the other author(s). The signatures of the Supervising Committee which precede all other material in the dissertation attest to the accuracy of this statement.

ACKNOWLEDGEMENTS

First, I would like to offer my most sincere acknowledgement to my advisor, Dr. Latifur Khan, for devoting enormous amounts of his energy and time to support me in my research. His consistent expertise and peerless vision made the pursuit of a Ph.D. degree a worthwhile and enjoyable experience.

I would like to thank my committee members, Dr. Bhavani Thuraisingham, Dr. Murat Kantarcioglu and Dr. Weili Wu for their many suggestions regarding the improvement of my research and for taking out the time to listen to my proposal and dissertation defense.

I would like to thank Dr. Kevin Hamlen, Dr. D.T. Huynh, Dr. Yang Liu and Dr. Mohammad M. Masud for providing inspiration and guidance throughout my time as a Ph.D student.

I would like to thank all of my friends and colleagues in Dr. Khan's and Dr. Bhavani's research groups for their encouragement and contributions on my papers and projects. In particular, I would like to offer special recognition to Neda Alipanah, Pallabi Parveen and Sunitha Sriram.

Most importantly, I would like to thank my parents, my brothers, my relatives and all of my closest friends for their unerring love and support since I started in the Ph.D. program in August of 2007. I would not be here without them, and they share in my successes.

November 2011

LEARNING-BASED GEOSPATIAL SCHEMA MATCHING

GUIDED BY EXTERNAL KNOWLEDGE

Publication No. _____

Jeffrey Partyka, Ph.D.
The University of Texas at Dallas, 2011

Supervising Professors: Dr. Latifur Khan, Co-Chair
Dr. Bhavani Thuraisingham, Co-Chair

Resolving semantic heterogeneity across distinct data sources remains a highly relevant problem in the GIS domain requiring innovative solutions. Our approach, called GSim, semantically aligns tables from respective GIS databases by first choosing attributes for comparison. We then examine their instances and calculate a similarity value between them called entropy-based distribution (EBD) by combining two separate methods. Our primary method discerns the geographic types from instances of compared attributes. If successful, EBD is calculated using only this method. GSim further facilitates geographic type matching by using latlong values to further disambiguate between multiple types of a given instance and applying attribute weighting to quantify the uniqueness of mapped attributes. If geographic type matching is not possible, we then apply a generic schema matching method which employs normalized Google distance. In addition, we seek to address additional challenges encountered when employing clustering-based geospatial schema matching. We focus on three distinct challenges. First, many schema matching

algorithms, including GSim, rely only on one instance property. Second, a consistent score for an attribute match is not produced. Third, hierarchical relationships between the data are not considered. In order to meet these challenges, we develop a successor to GSim called GeoSim. GeoSim derives clusters from attribute instances based on their geographic and semantic properties and produces a high-quality clustering by optimizing an objective function. It also captures hierarchical relationships between the GTs representing the instances in compared tables and attributes. Finally, GSim possesses the ability to execute greedy 1:N matching that reveals relationships between several attributes. 1:N matching is defined as an optimization problem and is justified with a proof of correctness. With impressive results generated, we show the effectiveness of GSim and GeoSim over traditional mining-based similarity approaches across multi-jurisdictional datasets.

TABLE OF CONTENTS

LIST OF FIGURES

Number Page

CHAPTER 1

INTRODUCTION

The amount of geospatial data that is accumulating in gazetteers, geodatabases and many other geographic data sources continues to increase at a very fast pace. One of the results of this is the proliferation of independent and heterogeneous data repositories of geospatial data accumulated by an increasingly disparate set of processes. For instance, unmanned aerial vehicles may take snapshots of a land area to analyze its transformation over a period of time(Fonseca 2002, 232), and sensor networks commonly are employed to measure the water level of a river to analyze its potential for producing flooding conditions(Klien 2004).

Because of this, questions regarding the feasibility and potential applications for integrating geospatial data in these repositories have arisen. These questions are some of the most crucial questions regarding information integration, which extends far beyond the geospatial domain. It has been explored in the form of semantic similarity research from cognitive science, information retrieval and artificial intelligence conducted over the past few decades(Tversky 1977, 329-36;Nosofsky 1986, 39-45;Bouchon-Meunier 1996, 144-151). With regard to the geospatial domain, geospatial data inherently possess vagueness, uncertainty and varying levels of granularity(Tian 2008;Kuhn 2005). Different sets of data modeling the same geographic location may be represented by differing file formats, type representations, coordinate reference systems, projections, natural language text descriptions, and much more. As a result, measuring the semantic similarity of geospatial data is a uniquely challenging problem

that will continue to require innovative solutions that are increasingly sophisticated as the unique properties of geospatial data become better understood.

Semantic similarity in the geospatial domain has been successfully applied to numerous information retrieval and ranking problems, including geolocation(Fink 2009), text classification(Martineau 2009;Seco 2004), geospatial tagging, land cover similarity(Ahlqvist 2006), Web services composition(Xu 2010), ontology alignment (Cruz 2007, 230-52;Janowicz 2009;Albertoni 2006;Rodríguez 2003, 442-54;Euzenat and Shvaiko 1998, 73-115;Janowicz 2008, 651-58;Cruz 2009b;Maedche 2002;Noy 2003, 983-1007;Haeri 2007, 803-11), recreational tasks like route planning for mountain climbing(Wilkes 2008), more serious tasks like emergency response decision making(Rinner 2007) and much more. Furthermore, the success of the geospatial Semantic Web depends very much on semantic similarity algorithms being able to determine commonalities and differences between geospatial data and their data models (Finin 2010). Efforts such as the Data Web and LinkedGeoData(Auer 2009) represent transitional efforts progressing towards a geospatial Semantic Web, as they connect disparate geospatial datasets to better facilitate geographic information retrieval and semantic similarity.

The main focus of our research is determining the semantic similarity between geospatial data within compared schemas. Research into the problem of schema matching within the geospatial domain would seem to be integral to the above efforts. Relatively speaking, though, it has not received very much attention. A number of research efforts(Luiz 2009;Brauner 2007;Cruz 2009a, 1586-89;Melnik 2010;Cheng 2010;Yan 2001;Kang 2003;Chiticariu 2006;Isaac 2007;Wartena 2008;Wang 2004;Wang 2008;Doan 2001) have focused on instance-based schema matching methods that depend on the semantics embedded in structured information(Sunna

2007;Mazuel 2008), such as a domain ontology, to identify correct correspondences. However, if a domain ontology is not available, or if it is not designed well (either because it is incomplete or subjective), then these methods will not work very well. Other methods which make use of various schematic properties such as metadata, word count frequency in the data, the data types of attributes, etc. tend to rely heavily on the syntactic properties of the data within the schemas to determine similarity. However, if schemas of different languages were compared, or if the matching was designed to discover similar types attached to instances, then exact instance matching would fail, particularly in the geospatial domain. This is because many instances with different names and geographic properties may end up sharing certain some other property, such as a geographic type as defined by a gazetteer. We require a tool that aligns geographic data, not based on the syntactic similarity of the data, but based on its semantic similarity; in other words, we need to match attributes based on whether their respective instances are of the same type, rather than if their string values match with each other.

In this dissertation, we propose a tool specifically designed to perform geospatial schema matching over heterogeneous data sources using an instance-based approach that leverages multiple clustering techniques. The subsequent sections in this chapter will introduce the basic concepts incorporated into our approach, including geographic type matching, non-geographic type matching, our information theoretic semantic similarity measure known as entropy-based distribution (EBD), attribute weighting, geosemantic clustering and multi-attribute matching (also known as 1:N matching). We also discuss a security-related application relevant to Android permissions granted to mobile phone applications at runtime, which, if implemented properly, can exhibit numerous benefits across several domains, including the geospatial domain.

1.1 GSim Overview

In this dissertation, we introduce GSim(Partyka 2009a;Partyka 2010, 52-70;Partyka 2009b;Partyka 2008a;Partyka 2008b;Partyka 2011), an information-theoretic algorithm used to measure instance similarity between compared attributes in geospatial schemas. Unlike the above methods of geospatial schema matching, GSim does not require any structured information for assistance in deriving attribute matches. At a high level, it works by first comparing tables. This is done by first determining pairs of attributes between the tables that are to be compared. Comparing all attributes of the compared tables and their instances against one another would result in a significant time penalty. Therefore, as a preprocessing measure, we use attribute name and data type matching to reduce the space of possible attribute mappings. Second, for each pair, we examine the respective attributes' instance data using two separate instance-similarity methods. Third, we determine corresponding attributes across tables based on semantic similarity scores. Combining all of the scores from aligned attributes will determine similarity between the tables as a whole.

Regardless of the method GSim uses to match instances, it performs matching at the concept/attribute level by examining instances belonging to those compared attributes/concepts. We consider every attribute/concept matched by GSim to consist of a set of one or more instances. Therefore, if the similarity between the instances of compared attributes/concepts is high, then this implies that the attributes/concepts themselves share this similarity. On the other hand, if the similarity between the instances of compared attributes/concepts is low, then this again implies that the attributes/concepts themselves share this similarity.

1.2 Geographic Type Matching

GSim's primary approach for examining instance data determines the geographic types (GT) over the instances associated with compared attributes. This is done by leveraging an external data source known as a gazetteer(Zhou 2004;Newsam 2008). In a naïve geotyping algorithm, the gazetteer matches one or more GTs for a given instance – this might happen if the feature has a common name, such as "Johnson", as there might be "Johnson Road", "Johnson River", etc. A more advanced geotyping algorithm, which GSim features, is able to identify exactly one GT for any instance recognized by a gazetteer with the help of latlong values. Latlong values help in disambiguating several instances with the same name, such that the proper GT may be associated with the instance in the schema. Of course, the effectiveness of this approach depends on whether the feature-type thesaurus in the gazetteer contains a set of types that is able to represent all of the instances from our data. Thus, we have made an assumption in this paper that any instance in our data set that can be identified by a gazetteer has a type which can be represented by that gazetteer.

Whenever possible, GSim calculates similarity between attributes using GTs alone. If the dataset contains latlong values associated with each instance, then based on our type assumption we made above, it is possible to guarantee a 1:1 mapping between each instance and its GT as identified by a gazetteer. However, due to the great variability in how geographic data is stored and represented, not all geographic instances necessarily come with latlong values. Thus, when the instances are fed to a gazetteer, we may derive more than one GT for certain instances. In this case, the best that can be done is to derive 1:N mappings of these instances to their respective sets

of GTs. Subsequently, similarity is calculated using these mapped GTs after applying a pruning algorithm for disambiguation purposes.

1.3 Non-Geographic Type Matching

In the case where too many instances within the compared attributes lack GT information, then GSim resorts to its secondary approach, which uses a generic schema matching algorithm based on a semantic distance measure known as normalized Google distance (GD)(Cilibrasi 2007, 370-81;Gligorov 2007). GD, combined with K-medoid clustering of the instances of an attribute, yields a set of non-geographic types specific to that attribute which are then used to compute semantic similarity with another attribute in a different table. This method is generic because it is not dependent on geographic types at all. Despite the utility of GD, solely relying on it to determine similarity is unwise, particularly in the GIS domain. The reason is that a number of situations exist where the instances are determined to be similar due entirely to their close geographic proximity. One such situation is depicted in Section 3.3.4.

GD, which depends on the Web pages indexed by the Google search engine, was chosen because of its effective coverage of the GIS domain. This is in contrast to an external knowledge source such as WordNet(Fellbaum 1998, 10-25), a lexical database of English containing over 117,000 synsets and over 200,000 word-sense pairs. While the coverage of WordNet is quite extensive for various domains, for the GIS domain, it is not very extensive at all.

1.4 Entropy Based Distribution

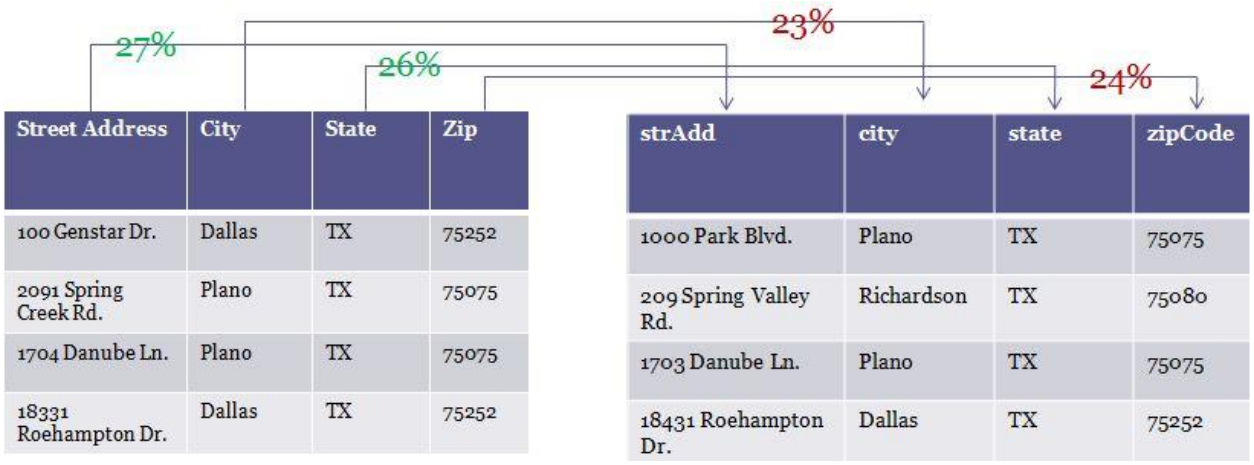
Once the instance type has been determined, similarity is calculated by considering the collection of types extracted from instances between the compared attributes. It is based on an information-theoretic measure known as entropy-based distribution (EBD), which is defined as the ratio of the conditional entropy within each type over a pair of compared attributes with the entropy taken over all types for that same pair. An EBD value has a range from 0 to 1, with 0 indicating no similarity whatsoever between the attributes, and 1 indicating identical attributes. The more similar that (1: the sets of GTs between the compared attributes are, and (2: the number of instances representing identical types between the compared attributes are, the higher the EBD will be, and vice versa. A formal definition of EBD is given in section 2.3.

The major advantage of using of an information-theoretic measure over other semantic similarity measures is its versatility and lack of constraints. Other similarity methods, such as those that use description logic (DL) (Rodríguez 2003, 442-56), NLP(Karalopoulos 2005;Vinyals 2010;Bui 2009) or network based matching(Albertoni 2006;Harrington 2010), require a strictly defined set of relationships between concepts and attributes. For example, calculating the difference in depths between two concepts (as one would do in network-based approaches), or determining a common parent between two concepts (as one might do in DL approaches) is only possible if the concepts are represented in a hierarchy, such as an ontology. NLP approaches are dependent upon the relationships between the words in a natural language description of a geographical concept. In turn, this depends on the presence of natural language text in a geographic concept, which is not guaranteed, the use of a language for which part of speech tagging can be confidently applied, etc. On the other hand, information-theoretic measures like

EBD do not require that data in attributes or concepts be organized in any way. In fact, a flat structure of attributes and concepts is not a problem for an information-theoretic measure. The only requirement that it has is for a probabilistic model to be applicable to the data being compared(Lin 1998). If this is the case, then information-theoretic measures can also be combined with any other semantic similarity technique, and it can be applied to various data models. For instance, although we applied the EBD measure in GSim to geodatabases, it can also be applied to ontology matching for Semantic Web applications. Since GSim uses instance-based matching to align attributes between tables in a 1-1 fashion, it can also be used to align the properties associated with the concept instances in a 1-1 fashion. This results in the alignment of concepts between ontologies.

1.5 Attribute Weighting

GSim also provides attribute weighting capabilities to emphasize semantic correspondences between attributes of compared tables where the attributes are not common, relative to the respective attributes sets across tables between their respective data sources. In the same way, attribute weighting may penalize strong semantic correspondences between tables resulting from attribute mappings where the attributes in the mapped pair commonly-occur across all of the tables in their respective databases. Doing this allows us to refine the semantic similarity score generated between tables by focusing on the compared attributes that are unique relative to attributes found throughout all tables.



Street Address	City	State	Zip
100 Genstar Dr.	Dallas	TX	75252
2091 Spring Creek Rd.	Plano	TX	75075
1704 Danube Ln.	Plano	TX	75075
18331 Roehampton Dr.	Dallas	TX	75252

strAdd	city	state	zipCode
1000 Park Blvd.	Plano	TX	75075
209 Spring Valley Rd.	Richardson	TX	75080
1703 Danube Ln.	Plano	TX	75075
18431 Roehampton Dr.	Dallas	TX	75252

Figure 1.1. Attribute Weighting Applied to Attribute Comparisons Between Tables

In Figure 1.1, we see four separate attribute comparisons between a pair of tables. By default, each attribute comparison within a given table comparison is considered to be as important as any other. With this weighting model, the attribute comparisons would share equal weight; in this case, being that there are 4 attribute comparisons, each one would be assigned a weight of 25% towards the final EBD calculation between the tables. However, in reality, some comparisons are likely more important than others, due to the uniqueness and relevance of the attributes relative to the other attributes involved in comparisons between the same table pair, or even between attributes matched between different tables. Figure 1.1 depicts a situation where two of the attribute pairs have been assigned higher weight, while two have been assigned lower weight. The sum of the weight percentages must always add to 100% for all attribute comparisons involving the same pair of compared tables.

1.6 Geosemantic Clustering and Hierarchical Matching

Despite the intricacies of the approaches inherent within GSim, a number of challenges regarding geospatial schema matching need to be addressed, particularly regarding the clustering phase. Here, we will consider the following three challenges. Three major challenges with geospatial schema matching remain largely unaddressed.

First, geospatial applications that calculate semantic similarity using instances often rely solely on one instance property; for some applications(Zhou 2007;Auer 2009), the property is geographic, and for others, the property is semantic(Janowicz 2009;Albertoni 2006). However, the idea of using both geographic and semantic properties for a geosemantic similarity algorithm has not been explored in detail to our knowledge. Using both properties can lead to better clustering.

Second, the semantic similarity score computed between compared attributes in most schema matching applications using clustering are based on inconsistent cluster quality. Because of this, the generated semantic similarity score for a given attribute comparison may differ from one trial to the next. In the case of our above work, GSim(Partyka 2009b), it uses clustering over the instances of a pair of compared attributes; however, this approach can lead to a large variance in the calculated similarity score, due to the inherent variability in the clustering process. Sometimes, this is due to factors such as the initial centroids (or medoids) chosen or the number of clusters. However, we will focus on score variability resulting from the varying semantic properties of instance samples from one clustering trial to the next for a given attribute comparison.

The third challenge is accounting for the presence of hierarchical relationships in the data. If the compared attributes are associated with GTs structured as a hierarchy (e.g. SWEET ontology in the earth sciences domain), then the semantic similarity score will be affected by the relationship between the attributes' GTs as indicated by the hierarchy.

These three challenges are addressed in the successor to GSim known as GeoSim. GeoSim will be discussed in detail in Chapter 5.

1.7 Multi-Attribute Matching

1:1 attribute matches can capture a great deal of the semantics between compared tables, and consequently, they have been explored extensively in the literature. However, many matches by their nature are not 1:1 and can only be characterized by a composite relationship involving > 2 attributes. Matches of this variety are known as multi-attribute matches or 1:N matches. Here, one attribute from one table is matched with N attributes from the other table, where N >= 2. Unlike 1:1 matching, multi-attributes matches have received little attention in the literature at this time. Figure 1.2 below depicts an example of a 1:N match.

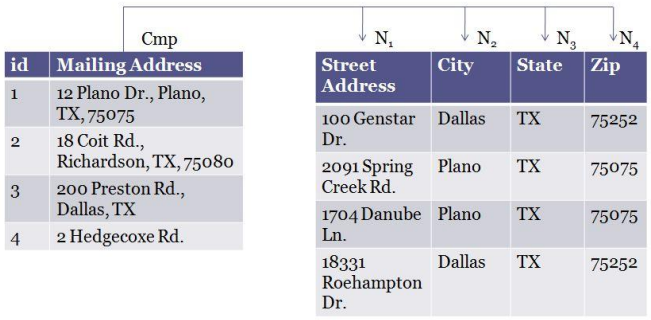


Figure 1.2. Example of multi-attribute (1:N) match

Here, a single attribute from one table (the attribute named Cmp) is matched up with N attributes from the other table. In this particular example, the N attributes represent constituent parts of the attribute Cmp, but there are other instances where the relationship is not quite as obvious. Additional questions to be explored regarding 1:N matching include the exact process by which semantic similarity is calculated, whether the match derived is the best 1:N match for the attribute Cmp, and whether the match is even correct to begin with. Chapter 6 discusses these issues in great detail.

1.8 Android Mobile Phone Security

In addition to geospatial research, we looked into work regarding the security of mobile phones sporting the Android operation system, due to its burgeoning popularity across a wide number of phone vendors. Across the world as of the end of 2010, there were 5.8 billion mobile phone users (MercoPress) leveraging the burgeoning capabilities of smart phones for everything from business conference calls, financial transactions, personalized communications such as email, SMS and Twitter, recreation such as mobile phone games and sports-related applications, and much more. Over 170 million smartphones were purchased globally in 2009(Gartner). As evidenced by their multifaceted capabilities, voice calling is no longer the main purpose of mobile phones – the fact that more than 5 trillion SMS messages were sent worldwide in 2009 serves as strong evidence(Portio Research).

With this newfound focus on applications in mobile phones comes a concomitant interest in the security of these applications, particularly at the application level. What makes this problem more difficult is the sheer diversity of applications that can be downloaded to mobile phones. For

example, (Ontang 2010) reports that the Apple App Store, which contains over 130,000 applications, recorded 280 million application downloads in December 2009. In particular, we will examine the problem of Android applications accessing what are potentially sensitive phone resources based on permissions granted to it by an antsy user upon the application's installation. Chapter 8 will explore this issue in more detail.

1.9 Contributions

Our contributions are as follows. First, we describe GSim, a method of aligning geospatial schemas using an information-theoretic measure to determine the semantic similarity of attributes. This is primarily accomplished via GT matching. Second, we propose a method of disambiguating among multiple possible GTs associated with an instance using an associated latlong value. Third, we introduce a method of attribute weighting that accounts for the uniqueness of the paired attributes relative to all others. This is done in order to improve the accuracy of the semantic similarity value between tables. Fourth, we provide a way to perform attribute matching using non-geographic types, in case insufficient GT information is available. Fifth, we introduce a geosemantic clustering paradigm that supports the recognition of both semantic and geographic properties within instances, reduces variance in the similarity score between the same attribute pair over multiple trial runs, and implements hierarchical matching over the GTs of instances. Sixth, we introduce a greedy 1:N matching algorithm proven to find optimal matches in O(n) time. Finally, we introduce a security permissions paradigms for use by applications installed on an Android mobile phone.

1.10 Organization of Dissertation

The dissertation is organized as follows. Chapter 2 states the background definitions, the problem to be solved, our proposed solution and the related work. Chapter 3 presents in detail the GSim algorithm, detailing both geographic type matching and non-geographic matching, along with experiments demonstrating the efficacy of both approaches. In chapter 4 we present our work regarding attribute weighting, along with its experimental results. In chapter 5, we present our work with GeoSim, which employs geosemantic clustering and hierarchical geographic type matching, along with its experimental results. Chapter 6 describes GSim's 1:N matching algorithm, along with its proofs of correctness and EBD optimality and experimental results. Chapter 7 describes our proposed Android application security framework related to the permissions associated with using phone resources. Finally, chapter 8 outlines our future work and directions in the aforementioned research areas.

CHAPTER 2

BACKGROUND

In this chapter, we provide an overview of the necessary background for understanding geospatial schema matching, the algorithms of GSim, and its semantic similarity measure, entropy based distribution. In addition, we provide an extensive analysis of related work depicting alternative schema matching and data source matching approaches that display similarities to ours, but rely on approaches that are syntactic or inappropriate in the context of the geospatial domain. Section 2.1 provides the definitions necessary to understand GSim's geographic type matching algorithm. Section 2.2 formally states the problem that we are trying to solve. Section 2.3 states our proposed solution to the problem and introduces the concept of entropy based distribution (EBD). Section 2.4 illustrates an example of EBD, and finally, Section 2.5 provides an extensive analysis of the related work that bears some similarities to GSim.

2.1 Definitions

Most of the existing techniques fall into the first category. First, we will provide definitions that will assist in defining the problem and describing GSim.

Definition 1 (attribute) *An attribute of a table T, denoted as att(T), is defined as a property of T that further describes it.*

Definition 2 (instance) *An instance x of an attribute att(T) is defined as a data value associated with att(T).*

Definition 3 (keyword) *A keyword k of an instance x associated with attribute att(T) is defined as a meaningful word (not a stopword) representing a portion of the instance*

roadName	City
Johnson Rd.	Plano
School Dr.	Richardson
Zeppelin St.	Lakehurst
Alma Dr.	Richardson
Preston Rd.	Addison
Dallas Pkwy	Dallas

Figure 2.1. Sample table containing 2 attributes and 6 instances per attribute

In figure 2.1 above, the two attributes for the given table are roadName and City, two instances from the roadName attribute are “Johnson Rd.” and “School Dr.”, and the two keywords associated with the instance “School Dr.” are “School” and “Dr.”.

Definition 4 (type) *A type t associated with attribute att(T) is defined as a class of related entities grouped together*

We define two kinds of types:

Definition 4a (geographic type (GT)) *A geographic type GT associated with attribute att(T) is defined as a class of instances of att(T) that represent the same geographic feature.*

Definition 4b (non-geographic type (NGT)) *A non-geographic type NGT associated with attribute att(T) is defined as a group of keywords from instances of att(T) that are semantically related to each other. An NGT is only derived for an instance when it cannot be associated with any geographic type from a gazetteer.*

Definition 5 (geographic type (GT) vector) *A geographic type vector* $T_x = \{GT_1, GT_2, \ldots GT_m\}$ *associated with an instance x of attribute att(T) is defined as a set of GTs.*

Definition 6 (geographic weight (GW) vector) *A geographic weight vector* $W_x = \{w_1, w_2, \ldots w_m\}$ *associated with a GT vector* $T_x = \{GT_1, GT_2, \ldots GT_m\}$ *for an instance x of attribute att(T) is defined as a list of real numbers between 0 and 1 representing the influence of a GT on the instance.*

Note that for all i, $GT_i \in T_x$ of any instance x has weight $w_i \in W_x$.

Definition 7 (geographic type set of attribute (T_{att})) *A geographic type set of attribute att(T), denoted* T_{att} *, is the set of GTs derived from the union of the types from all GT vectors for the instances of att(T).*

Definition 8 (non-geographic type set of attribute (NT_{att})) *A non-geographic type set of attribute att(T), denoted* NT_{att}*, is the set of NGTs associated with keywords from instances of att(T).*

Definition 9 (geographic type weight list (W_{att})) *A geographic type weight list W_{att} associated with attribute att(T) is the total type weights for each type in T_{att}.*

We will now proceed with our problem statement.

2.2 Problem Statement

The inputs to GSim are two data sources, S_1 and S_2, each of which is composed of a set of tables where $\{T_{11}, T_{12}, T_{13}\ldots T_{1k}\ldots T_{1m}\} \epsilon S_1$ and $\{T_{21}, T_{22}, T_{23}\ldots T_{2j}\ldots T_{2n}\} \epsilon S_2$, with $1 <= k <= m$ and $1 <= j <= n$. Next, similarity between T_{1k} and T_{2j} is computed by matching each attribute of T_{1k} with one attribute of T_{2j}. If T_{1k} and T_{2j} have different numbers of attributes, the table with less attributes has each attribute matched with an attribute in the compared table. The table with more attributes may have >= 1 attributes not involved in any matches. Once all 1:1 attribute pairs have been determined, then for each pair, we determine their similarity by examining their respective instances and assigning them to clusters. Once the instances have been clustered for an attribute pair, each cluster is now synonymous with a particular GT. Similarity between the attributes is then calculated based on the distribution of attribute instances for each GT. We use EBD to represent the similarity; EBD is described in section 2.3 and section 2.4. The similarity of tables T_{1k} and T_{2j} is the average of EBD values between all of their attribute pairs. At this point, we may perform 1:N matching similarity to determine additional matches.

2.3 Proposed Solution

We present GSim, an instance matching algorithm that generates semantic similarity values between compared attributes in different tables of a geodatabase. The derivation of

attribute mappings between a pair of compared tables is created in two separate stages. First, a preprocessing phase based on data type matching and attribute name matching determines the pairs of attributes that are most likely to be similar. These attribute pairs represent the attribute mappings whose collective similarity values will determine the similarity value between their tables. Second, instance-level matching is applied to each attribute pair in order to determine their similarity. Our instance-level matching is based on two main approaches. The primary approach assigns GTs to instances involved in compared attributes within two tables of the geodatabase with the help of a gazetteer. This results in a pair of GT sets, one for each attribute. The semantic similarity between the compared attributes is then computed using EBD over their respective GT sets. However, since gazetteers will not contain information about every instance, it is possible that attribute matching via geographic-type extraction will be ineffective. In this case, we apply a generic, non-geographic type (NGT) matching method, applicable over any knowledge domain, that is based on the extraction and clustering of instance keywords into NGTs, based on GD. We can further apply attribute weighting and 1:N matching to the results produced GT and NGT matching over instances of attribute pairs generated by GSim. Alternatively, we can match attribute pairs between tables of distinct data sources by using GeoSim, allowing for a match to occur using two distinct instance properties. If the GTs of the instances are arranged in a hierarchy or ontology, then we can also apply hierarchical matching using GeoSim_{H}.

2.4 Entropy Based Distribution

Since GSim's clustering and 1:N matching algorithms both lead to semantic similarity scores, we will first discuss EBD, our information-theoretic semantic similarity measure that was

inspired by the work of (Dai 2008). EBD is defined as a comparison of the conditional entropy of the attribute instances, given a particular GT, with the entropy over all GTs:

$$\mathbf{EBD = H(A|T) / H(A)} \quad (1)$$

Here, A is the attribute, coming from either table, and T stands for the type of instances (T=GT). An EBD has a value = [0,1], with 0 indicating no similarity between the attributes, and 1 indicating identical attributes. The greater the similarity in the distribution of GTs displayed by instances between the compared attributes, the higher the EBD and vice versa.

As an example, consider the attribute comparison Road(T_1)-Street(T_2). After analyzing the GD between all instance pairs and GTs of each instance, the clusters that result are as follows:

Cluster 1 (Interstates): {**25 instances ϵ Road(T_1), 20 instances ϵ Street(T_2)**}

Cluster 2 (County Roads): {**20 instances ϵ Road(T_1), 16 instances ϵ Street(T_2)**}

Cluster 3 (Local Roads): {**7 instances ϵ Road(T_1), 11 instances ϵ Street(T_2)**}

An EBD calculation proceeds as follows. First, the entropy is calculated by comparing the number of instances between Road(T_1) and Street(T_2) across all clusters (GTs). There are 52 instances from Road(T_1), 47 instances from Street(T_2), and a total of 99 instances. By inserting 52/99 and 47/99 in the equation for entropy, we calculate the entropy value to be .998. Next, the conditional entropy is calculated, which is represented as: $-\sum_{x\epsilon X; y\epsilon Y} p(x,y) \log p(y|x)$.

In this expression, the term p(y|x) represents the probability of an instance belonging to either Road(T_1) or Street(T_2), given its GT x. The term p(x,y) represents the probability that an

instance has a GT x and belongs to either Road(T_1) or Street(T_2). We calculate these terms for instances across all clusters (GTs). The conditional entropy value for the example above = .986. The final EBD value between the two attributes is (.986/.998) = .988. This makes sense, since the distributions of instances between the two attributes for each GT are similar.

2.5 Related Work

We will first present work related to schema and ontology matching. Second, we present work in the GIS domain making use of a gazetteer. Third, we present work making use of reverse geocoding. Fourth, we will present work done regarding attribute weighting. Fifth, we discuss other geospatial works that make use of a gazetteer to solve information retrieval problems. Next, we present related work in regards to 1:N matching, M:N matching, and other forms of complex matching as applied to schemas and ontologies. Finally, we contrast our work with another approach used to solve the schema matching problem.

A number of schema matching publications (Ralun 2001;Dai 2008;Bohannon 2006;Warren 2006) tailored to the database community influenced our work. The survey of approaches to automated schema matching by Rahm and Bernstein(Ralun 2001) includes a taxonomy which uses several criteria to categorize matching approaches such as schema and instance based methods, element-level and structure-level methods, and linguistic and constraint-based methods. While this work surveys a wide swath of approaches covered in schema matching literature, it does not present any approaches specifically tailored to the geospatial domain. Matching in the geospatial domain presents unique challenges, due to the properties inherent in geospatial data such as geometry, georeferenced coordinates, variations in formatting and coordinate systems,

and much more. The nature of geospatial data is complex enough such that many applications, including our current implementation of GSim, have only addressed a subset of its unique properties. Dai, Koudas et al. (Dai 2008) discuss instance-based schema matching using distributions of N-grams among compared attributes. The differences between our work and (Dai 2008) is discussed later in this section. Bohannon et al.(Bohannon 2006) investigate contextual schema matching, in which selection conditions and a framework of matching techniques are used to create higher quality mapping between attributes of compared schemas. Among their methods for deriving selection conditions is the training of a classifier on the attribute values from an attribute involved in a match. This would imply that the values of an attribute can be expressed by a pattern, such as a regular expression. However, this would not work in the geospatial domain because a number of attributes, such as 'City' and 'County' cannot have their attribute values described by a generalized expression. Thus, training classifiers on these attributes would not make a contribution towards a match with other similar attributes. Warren and Tompa (Warren 2006) propose an iterative algorithm that deduces the correct sequence of concatenations of column substrings in order to translate from one database to another without the use of a set of training instances. While this work addresses some of the same challenges that we do, our work is distinguished by the inclusion of attribute weighting to account for differences in the importance of certain attribute comparisons over others, and also by our use of latlong driven disambiguation as applied to geographic instances identified by gazetteers.

Within the AI community, a number of works in the schema matching area applied machine learning and statistical methods to learn attribute properties from data and examples. Li and Clifton (Li 2000, 51-73) describe a tool known as SEMINT, which uses neural networks to

determine match candidates by learning the metadata and data values patterns of attributes. From this, other attributes with similar metadata and data value patterns are sought in order to create 1:1 attribute mappings. However, their methods would not work in many cases for geospatial schema matching because several attributes in this domain share similar metadata and/or data value patterns, yet are completely different. For instance, the attributes "County" and "City" both could be characterized with the same SQL datatype (ie: CHAR (40)), and they may even share some identical data values. However, learning these characteristics would never amount to anything, because of the arbitrary nature of the names of counties and cities. Madhavan et al.(Madhavan 2001) present CUPID, a method exploiting linguistic matching between the attribute names and data types of compared schemas to compute semantic similarity. CUPID uses a clustering algorithm to model similarity between attribute names by a combination of direct comparison and the comparison of their representative clusters. Each cluster contains lexicographically similar metadata elements to the compared metadata element for a particular schema. However, their clustering method relies on a manual thesaurus lookup to determine the membership of a schema element in a particular cluster. Thus, clustering in CUPID is not fully automated. Also, CUPID does not make use of any instance information. A major problem with determining semantic similarity in schemas and ontologies are the inaccurate names often used by humans during the designing phase. The use of instance information can resolve such difficulties. Berlin and Motro(Berlin 2001) describe a tool known as Autoplex which uses supervised machine learning techniques such as Naïve Bayes classification for automating the discovery of new content for virtual database systems. While the versatility of the Naïve Bayes approach is widely known, its binary classification methodology is a problem for geospatial schema matching. In the Naïve

Bayes approach, an instance either belongs to an attribute or it does not. In geospatial schema matching, a finer grained approach is needed since instances often display degrees of membership to various attributes. GSim takes into the possibility of instances having multiple GTs. It attempts to reduce the number of GTs for an instance to one, but if this is not possible, then it takes into account all possible GTs for that instance into the final EBD calculation. Embley et al. (Embley 2004) explore both 1:1 and m:n schema mapping techniques by applying knowledge obtained from domain ontology snippets and data frames. However, if this method was applied to geospatial schema matching, then it would fail for the same reasons as Semint (Li 2000, 51-73) would fail. The problem is the assumption that the membership of an instance value to an attribute is based on a data pattern or a regular expression. In the geospatial domain, this is often not the case. Also, the use of domain ontology snippets for schema matching is highly subjective. The structure of the ontology is often dependent on the specific vision of its designers, which might differ from the vision of those individuals who designed the schemas being mapped. Furthermore, the choice of the ontological snippet to use is inevitably fraught with bias in one form or another. Pei et al.(Pei 2006) introduces a holistic schema matching approach over multiple schemas that relies on clustering at multiple stages. First, schema clustering is executed in order to aggregate schemas with similar contexts. Second, attribute clusters are formed from schemas in the same schema cluster. Third, attribute matches between different clusters are derived. Nonetheless, this approach heavily relies on syntactic matching. The formation of schema clusters is based on the exact matching of words associated with the attributes and their textual descriptions. Furthermore, the assignment of an attribute name to a particular cluster is based on the similarity between the term frequency vector of an attribute and the centroid term frequency vector.

The most closely related work in the GIS domain discusses instance matching over geodatabases, ontologies, thesauri and other geographic data sources. Cruz, Sunna et al.(Cruz 2007, 230-54) describe AgreementMaker, a visual tool that provides a user with the ability to perform mappings between ontologies using a multi-faceted strategy involving automated techniques as well as manual specifications. Albertoni et al.(Albertoni 2006) devised an instance based similarity measure that matches instances of ontological concepts based on two contextual layers: an ontology context, which is based on a comparison of the concepts' depth in a structured hierarchy as well as the number of attributes and relations they share, and an application context, which uses instance paths and set of predefined comparison operations between concepts to perform a match based on the specific needs of the user. Janowicz and Wilkes(Janowicz 2009) describe SIM-DLA, a DL based instance similarity measure that matches instances from a source concept, specified as a user query, with the instances from all target concepts that can satisfy the query. This is determined with the help of a context concept that is the superclass of all possible target concepts, along with a modified version of the tableau algorithm that is normally used in satisfiability checking. Unlike GSim, each of the above approaches requires that the instances of concepts or attributes belong to a sophisticated ontology replete with numerous relation types between the concepts and/or attributes. In a use case involving matching between two unstructured geospatial data sources, like flat sets of concepts, thesauri or an unstructured folksonomy (which might consist of satellite imagery of a geographic location, along with its keyword annotations) consisting of concepts annotated by a community of users, the methods above which depend on a defined structure of concepts will not be applicable. Karalopoulos et al.(Karalopoulos 2005) outline a method for using POS tagging and subsequent parsing to convert

a geographic concept description into a conceptual graph, which could then be used for various purposes including semantic similarity. Though this work does not explore semantic similarity, it also relies on a strict relation structure between the tagged words in the concept definition, as well as a strict grammatical structure of the definition itself. If the concept does not contain any annotations, then this method will not work. Furthermore, the successful creation of a conceptual graph depends on the definition containing an ordered grammatical triple consisting of a genus, differentia and an illustrative example. Obviously, many ontologies exist where concepts are annotated differently. Other work related to instance matching in the geospatial domain is as follows. Ahlqvist and Shortridge(Ahlqvist 2006) introduce semantic variograms, which can be used to determine the semantic similarity of multi-class land areas separated by a series of spatial lags. Leme, Casanova et al.(Luiz 2009) perform schema matching over GIS databases containing data represented by a dialect of OWL. Brauner, Intrator et al.(Brauner 2007) perform instance matching over the exported schemas of geographical database Web services and apply their technique over the GeoNames and ADL gazetteers. Brauner, Casanova et al.(Brauner 2006) leverage instance mapping between distinct terms in feature type thesauri used to classify data in gazetteers, for the facilitation of successful thesaurus migration from one gazetteer to another. The method described in (Ahlqvist 2006) works well for land cover classification, but would not work as well for geospatial schema matching, since its matching criteria only works over ordinal data. The aforementioned methods use co-occurrence statistics of pairs of keywords or types in order to derive attribute mappings. In many cases, this is an effective method; however, in order for it to work, it relies on a syntactic match between either instance names or the instance types. Often times, the names and properties of geospatial entities contain slight variations which require

methods beyond syntactic matching in order to determine a match with another entity. GSim relies on semantic matching by leveraging the GTs and latlong values of compared instances for geographic type matching. If geographic matching is not possible, instances can also be compared using a semantic NGT match via GD and K-medoid clustering.

Much work in the GIS community making use of a gazetteer for information lookup influenced also our work. Zhou, Frankowski et al. (Zhou 2004) apply a deterministic, density-based clustering algorithm to semi-automatically discover gazetteers from users' travel data, as well as disambiguate between uninteresting and interesting results from the gazetteer using temporal techniques. Newsam and Yang(Newsam 2008) integrate a gazetteer with high-resolution remote sensed imagery to automate geographic data management more completely, and they also demonstrate how gazetteers can be effectively used as a source of semi-supervised training data for geospatial object modeling. Pouliquen, Steinberger et al.(Pouliquen 2004) use a gazetteer lookup, as opposed to linguistic analysis, to search through natural language text and produce geographic maps and animations that represent the area referred to in the text. Despite the novelty of these works, they fail to address the challenges in geospatial matching that GSim is able to meet. The works just mentioned depend on performing exact matches between the user's data and data found in a gazetteer. A sophisticated semantic matching algorithm must discover similarity between heterogeneous sources, whether or not an exact word match exists between the compared data. Thus, the methods outlined here would be ineffective towards the application of aligning two geospatial ontologies that model the same geographic area, but using different languages. Meanwhile, the work in (Newsam 2008) focuses on using remote sensed imagery as training data in an effort to model geographic objects in a semi-supervised way; since it works with images as

opposed to text, it solves a different problem than GSim. However, even if it was applied for semantic matching over compared data sources that also contained representative image data, errors resulting from the variability of images, such as lighting, inclement weather, scale, etc. would cause a fairly high degree of error in identifying objects (or geographic features, in this case) from the images. Using GSim's type matching method, as long as a GT is associated with a geographic feature in a gazetteer, there will be no ambiguity about the type of a feature. Some work in the GIS community involving reverse geocoding is related to our research. Zhou and Frankowski (Zhou 2007) evaluate the accuracy of personal place discovery using reverse geocoding and clustering through a set of evaluation metrics and an interactive evaluation framework. Joshi and Luo(Joshi 2008) employ reverse geocoding using location coordinates from image data to obtain nearby points of interest connecting an image with its geographic location. Wilde and Kofahl(Wilde 2008) describe the use of reverse geocoding in retrieving location types as an essential component for a geo-enabled Web browser. Our work shares some tangential similarities with the above work (i.e the use of clustering), but differs fundamentally by using latlong information from gazetteers and attribute weighting to derive a more intelligent means of performing schema matching across data sources in the GIS domain.

Attribute weighting research has mostly focused on applications of machine learning, such as estimation by analogy and query ranking. To the best of our knowledge, it has never been applied to schema matching in the geospatial domain. Li and Ruhe(Li 2008, 1-23) performed a comparative study of five separate attribute weighting heuristics as a means of measuring software effort estimation. The heuristics are based on rough set analysis, which uses the notion of equivalence classes to construct approximations of a given set. This method, as stated in (Li

2008, 1-23), would not apply very well for our purposes to schema matching for two reasons. First, rough set analysis is designed to work with ordinal data, such as a list of categories (ie: {Low, Medium, High}). Our data sets consist of non-ordinal data, such as sets of county names or latlong values. Second, the methods described here depend on historical data sets to determine an analogous weighting scheme suitable to the current data set. However, there is nothing to suggest that these methods can handle new data values that have never appeared in any historical data set. In geospatial schema matching, it is common to encounter entirely new data values with the task of determining their similarity to another data set. Su et al(Su 2006) use attribute weighting to rank a list of results generated from a user query over an e-commerce database without the need for direct user feedback. However, in their approach, while it is true that they do not require a user to provide direct feedback on the attributes most important to him/her, this work determines the attribute weight largely based on implicit hints provided by the user query. For instance, in a web database of used cars consisting of attributes “Year”, “Price”, “Mileage” and others, if a user specifies a query, “Year > 2009”, then this work surmises that the user prefers a car with low “Mileage”, thus making this attribute more important than others. However, in our experiments, no user feedback whatsoever is available. Also, this work assumes that the “Price” attribute is always present in a database. For our experiments, we can never assume that a particular attribute is always present.

Among tools producing 1:N matches or other complex matches, the following influenced our work. Thor et al.(Thor 2007) propose an M:N ontology matching algorithm based on instance-based concept matching calculated via Dice’s coefficient. However, 1:1 instance similarity is determined by the exact matching of characteristic id values, a syntactic approach.

Since M:N matching is based on 1:1 matching, their M:N matching algorithm is also syntactic. Dhamankar et al.(Dhamankar 2004) describe iMap, a schema matching tool that discovers 1:N, M:N and other complex matches between attributes. It employs a set of searchers and exploits several kinds of domain knowledge to derive its matches. But iMap requires user feedback at various stages of the matching process, making it semi-automatic. Also, iMap uses a Naïve Bayes classifier to learn instance patterns associated with a given attribute. As with Semint, this approach will not work in the geospatial domain. Hu and Qu(Hu 2006) propose an algorithm that produces M:N matching between blocks of concepts of compared ontologies. The blocks are created by a partitioning algorithm that attempts to maximize the similarity between concepts within a given block while minimizing the similarity between concepts from different blocks. However, their 1-1 concept matches rely on syntactic criteria. The concepts are represented as virtual documents, created from keywords associated with the concept descriptions. When virtual documents are compared, similarity depends on the frequency of matching keywords.

We now seek to compare GSim against two other works. The first is the work of Dai, Koudas, et al(Dai 2008). They present a solution to the schema matching problem that makes use of N-grams, a syntactic matching method. The second work is that of Su et al.(Su 2006).

First, in regards, to the work described by (Dai 2008), we argue that GSim features an innovative instance matching algorithm that possesses a number of advantages over the N-gram approach, particularly in the GIS domain. An N-gram is a substring of length N consisting of contiguous characters. So for example, if N=2, then the word ‘GSim’ has N-grams ‘GS’, ‘Si’ and ‘im’. First, GSim determines GTs for instances via a gazetteer as part of the process of determining an overall semantic similarity value between attribute pairs containing those

instances. Because GSim uses domain-specific information to determine the GT for a given instance, it is better equipped than the N-gram approach to solve the information integration problem among geodatabases. N-grams cannot take advantage of domain knowledge, since they are only parts of words. Second, GSim can retrieve missing instance values in geodatabases by using associated latlong values to perform reverse geocoding. This ability is not available using solely the N-gram approach, because they cannot have latlong values associated with them. Third, in case the geographic lookup component is unsuccessful, GSim leverages clustering of types for use on distinct keywords found between compared attributes via GD. This approach is better able to capture the semantics of comparisons between attributes because words contain more implicit semantic information than N-grams. Using words, we can reference external data sources that allow for distance metrics to determine word relatedness. Finally, our new instance matching algorithm does not require a syntactic match between its instances, whereas N-grams does. For example, for two N-gram instances to match, they have to represent the same string (ie: “ab”). On the other hand, GT matching in GSim would be able to match instances such as Spring Valley Road and Canyon Creek Drive, based on their common geographic type.

Second, in regards to the work of (Su 2006) a schema matching tool called HSM is introduced that calculates similarity between multiple web databases via attribute matches from their representative Web query interfaces. To create matches, two scores are computed between every pair of attributes. First, a grouping score is computed between a pair of attributes, determining the likelihood that the attributes naturally co-occur in the same Web query interface (ie: ‘City’ and ‘State’). Second, a synonym score is computed between a pair of attributes to determine their similarity. Both of these measures are used in tandem to determine attributes

matches using a greedy algorithm. Despite their similarities, GSim provides the following benefits over HSM. First, synonym matches in HSM rely on exact keyword matching of attributes, a syntactic measure. GSim depends on semantic matching via GD and does not rely on syntactic matching of keywords in any way. Second, instances in HSM are not exploited. If this information were used, perhaps by domain query probing, then it would allow the definitions of the keywords to be broadened enough to allow for expanded synonymous matching. GSim depends on instance level matching to determine the similarity between compared attributes. Third, HSM's greedy algorithm produces a complex match designed to maximize its number of attributes. This means, however, that upon the addition of a new attribute to a match, the total similarity score between the attributes in the match may go down. This can happen because HSM regards any synonym match between attributes with a similarity score > 0 as valid. GSim counters this problem in two ways. First, a synonym match between two attributes can only occur if their similarity is >= .6. Second, in section 5.2, we formally show that any 1:N match in GSim produces an optimal similarity score, and that an attribute that does not generate a similarity value of .6 with a composite attribute cannot be included in the match.

CHAPTER 3

GSIM AND ITS MATCHING CAPABILITIES

In this chapter, we introduce GSim, which performs geospatial schema matching over heterogeneous data sources by deriving 1:1 attribute matches between compared tables. A broad overview of GSim and its different components is illustrated in Figure 3.1. GSim applies two separate algorithms to perform geospatial schema matching. First, it discerns the GTs of instances involved in a 1:1 attribute match using a gazetteer. If enough GTs are available, then EBD is calculated in the manner described in Sections 2.3 and 2.4. GT matching will be described in details in Section 3.2. However, if not enough GTs can be discerned for the instances involved in the comparison, then non-geographic matching, or NGT matching, is applied, which takes into account the Google Distance, or GD, between instances of the same type. This is a generic matching method that is based on co-occurrence of instances, and it will be discussed in detail in Section 3.3. Finally, we will present the results of experiments in geospatial schema matching over three different multi-jurisdictional datasets that demonstrate the efficacy of each approach.

3.1 GSim Overview

We begin by outlining the algorithm describing GSim, from a broad perspective considering both GT and NGT matching. For algorithm 1 below, represented as Figure 3.2, the input consists of the attributes $A_1 \in T$ in S_1 and $A_2 \in T'$ in S_2 and gazetteer G.

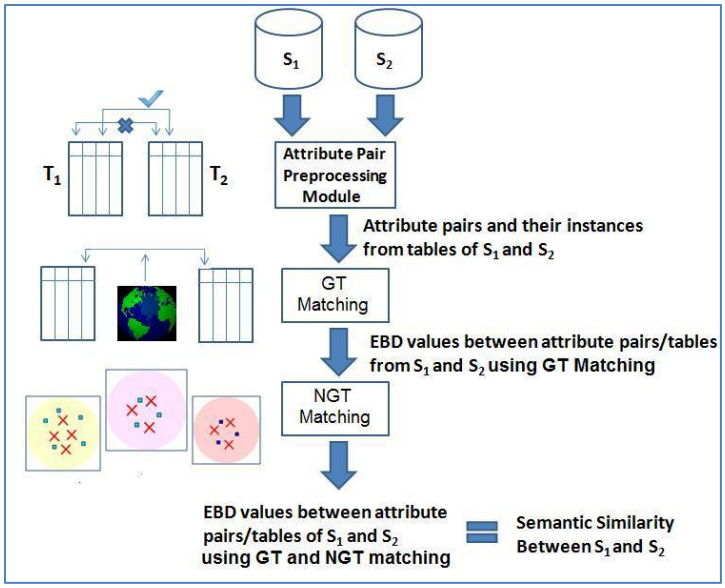


Figure 3.1. An Overview of GT and NGT matching in GSim

Line 1 initializes T_{gaz}, the set of all GTs recognized by gazetteer G, T_{A1} and T_{A2}, the GT vector lists for A_1 and A_2 respectively, NT_{A1} and NT_{A2}, the NGT vector lists for A_1 and A_2 respectively, and W_{A1} and W_{A2}, the GW vector lists for A_1 and A_2, respectively. Lines 2 and 3 extract the distinct instances from A_1 and A_2. Line 4 determines whether semantic similarity can be performed strictly by relying on GTs, or if GD similarity will be necessary. GT similarity is only possible if a gazetteer is available, and if it contains sufficient GT information about enough of the instances. For our purposes, we established a threshold, t_{min}, which represents the minimum number of instances that contain GT information in G. In our experiments, t_{min} was set to a value of .5. Therefore, if GT information can be found for a number of instances greater than or equal to t_{min} (at least 50% of the instances in the compared columns), then EBD is calculated using only GTs. This process is initiated in lines 5-8, where line 5 retrieves all available GTs, T_{gaz}, recognized by gazetteer G, lines 6-7 derives a GT vector list T_{A1} and its GW vector list (W_{A1} in line 6 and W_{A2} in line 7), consisting of GT vectors for each instance of A_1 and A_2.

Algorithm 1 GSim (A_1, A_2, G)

Input:
- attribute A_1 ε T in S_1 , attribute A_2 ε T' in S_2, gazetteer G
Output: Semantic similarity value between A_1 and A_2, expressed as EBD

1: $T_{gaz} = \Phi$, $T_{A1} = T_{A2} = \Phi$, $NT_{A1} = NT_{A2} = \Phi$, $W_{A1} = W_{A2} = \Phi$
2: IL_1 = ExtractInstances (A_1)
3: IL_2 = ExtractInstances (A_2)
4: if (geotypingIsPossible (G, IL_1, IL_2)) {
5: T_{gaz} = getGazetteerTypes(G)
6: (T_{A1}, W_{A1}) = lookupGeoTypes(T_{gaz}, IL_1)
7: (T_{A2}, W_{A2}) = lookupGeoTypes(T_{gaz}, IL_2)
8: } else {
9: (NT_{A1}, NT_{A2}) = NGDSim(IL_1, IL_2)
10: }//end if
11: EBD[A_1][A_2] = computeEBD(T_{A1}, T_{A2}, W_{A1}, W_{A2}, NT_{A1}, NT_{A2})
12: return EBD[A_1][A_2]

Figure 3.2. Algorithm 1: GSim, from a broad perspective

If however, in line 4 if geotypingIsPossible() returns false, then we need to rely on a more generic measure like GD to compute semantic similarity between the compared instances. This is done in line 9. The GD component of GSim will be described in section 3.1.4. Line 11 calculates the final EBD value between A_1 and A_2 given the combined type vector lists and weight vector lists of A_1 and A_2, and line 12 returns that EBD value.

We justify our usage of GSim as a semantic similarity metric by comparing it against an alternative semantic similarity metric derived from WordNet, a lexical dictionary for the English language. We decided against using it because of its shallow coverage of concepts relative to that which is covered by the combination of GSim for geographic matching and GD for non-geographic matching. For example, in comparing two street name attributes of the Road-Road table comparison for the GIS transportation dataset (see Section 5 for more information on the table comparisons), GSim+GD was able to compute 4776 out of 4992 (95.7%) distinct pairwise

distance values for the extracted keywords between the pair of attributes. For the same attribute pair, WordNet was only able to calculate 2,068 distinct pairwise values, only 43.3% of the number of values calculated by GSim+GD. Additionally, for a comparison of a street name attribute and a port name attribute between the Road table of S_1 and Ferry table of S_2 for the GIS transportation dataset, GSim+GD found 132 out of 161 (81.9%) distinct pairwise values between extracted keywords while WordNet only found 22 out of 161 (16.7%).

3.2 Geographic Type Matching

We will now describe geographic matching (GT matching) in detail. There are two different types to consider: (1: Naïve GT matching, where an instance may be mapped to more than 1 GT and (2: Latlong GT matching, where an instance is matched to exactly 1 GT.

3.2.1 Naïve GT Matching

The first step in Naïve GT matching is to leverage a gazetteer to help determine the GT of an instance. The gazetteer used for our purposes is GeoNames(GeoNames), containing information on over 8 million geographic names. The gazetteer classifies locations into different categories, or types. Some examples of GTs include city, county, state and a general feature with several sub-classifications, such as lake, port, school, etc. Instances with more commonplace names are likely to be listed under multiple types in the gazetteer. As a result, a single instance may be associated with a list of GTs = $\{GT_1, GT_2 \ldots GT_n\}$, where n is the number of GTs recognized by the gazetteer. However, as will be described in Algorithm 2, because an instance

may have multiple GTs, the weight of that instance for each of those types is divided proportionately. Finally, an EBD calculation over the different GTs is performed.

Formally, let $T_{gaz} = \{GT_1, GT_2,\ldots\ldots,GT_m\}$ be a set of GTs recognized by gazetteer G, with GT_i, $0 <= i <= m$, representing one of these types. For example, GT_i may be a county, city, state, etc. An arbitrary instance x associated with attribute att(T) will be associated with a GT vector $T_x = \{GT'_1, GT'_2,\ldots GT'_n\}$, $n <= m$ and $n > 0$. Let $W = (w_1, w_2,\ldots.w_n)$ be a GW vector, where each w_j is associated with each GT'_j in T_x for instance x, where $|W| > 0$ and all w_k in W for x have a value of $1 / |W|$. For example, if x was associated with three GTs, then the weight wj of each type for x would be 1/3.

In figure 2 below, the instances are "Victoria", "Anacortes", "Clinton" and "Edmonds". The GT 'City' represents the instances "Victoria" and "Clinton", The GT vector for "Victoria" = {City, State, Feature} and for "Anacortes", it is = {County}. The GW vector for "Victoria" is {1/3,1/3,1/3}, and for "Anacortes" it is {1}. If these four instances make up the entirety of attribute att, then T_{att} is {City, State, Feature, County}, and the GT weight list W_{att} is {1/3 + 1/2, 1/3, 1/3 + 1/2, 1 + 1}, or in simplified form, {5/6, 1/3, 5/6, 2}.

As an example of illustrating the weighting of GTs under naïve GT matching, consider figure 3.3, which depicts some instances along with the GTs that they match up with, as determined by a gazetteer.

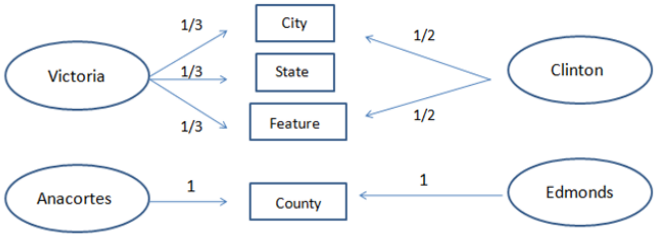


Figure 3.3. Example of Naïve GT Matching

Taking all of these instances into account, the total weighting for the types listed are as follows: “City” = (1/3 + 0 + 1/2 + 0) = 5/6, “State” = (1/3 + 0 + 0 + 0) = 1/3, “Feature” = (1/3 + 0 + 1/2 + 0) = 5/6, and “County” = (0 + 1 + 0 + 1) = 2 (Recall that for “County”, 1 is for Anacortes and 1 is for Edmonds).

3.2.2 Using Latlong Values

GSim also possesses the ability to leverage latlong values for the purposes of improving the accuracy of the semantic similarity measurement between two attributes, and ultimately, between two tables. This is accomplished by comparing latlong values associated with the instances of compared attributes, and comparing them against latlong values for those same instances in the gazetteer. This technique is intended for those instances associated with multiple GTs; using latlong values, it will be possible to identify the correct GT out of many within the GT set for an instance with a common name such as “Clinton”. At the same time, latlong values can also help disambiguate among the GTs of instances where the types do not match.

The process of using latlong values for further disambiguation is illustrated in figure 3.4.

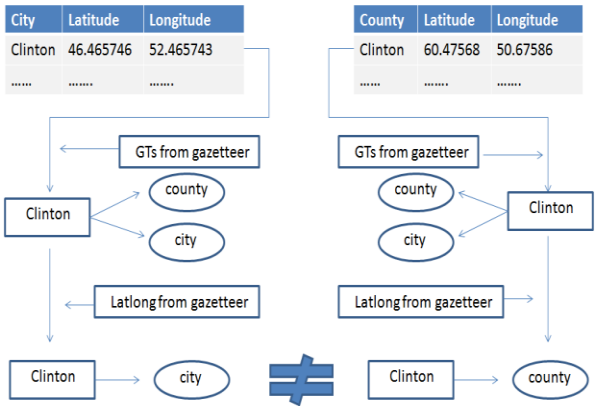


Figure 3.4. Example of Latlong GT Matching

Here two attributes are being compared against one another for the purposes of deriving a semantic similarity value. In particular, an instance of the City attribute from table T named “Clinton” is compared against an instance from table T’ from the ‘County” attribute, also named “Clinton”. Since “Clinton” is a common name, a query to a gazetteer by GSim for both instances is very likely to result in the return of > 1 GTs. Without the use of latlong information, we would not be able to definitively pare down the number of GTs for each instance, thus affecting the accuracy of the semantic similarity calculation. In figure 3.4, the instance “Clinton” in table T is associated with both the county and city GTs, and no further disambiguation is possible. The same is true with the instance “Clinton” from table T’. However, the use of latlong information, both from the instance data and the gazetteer itself, allows a comparison of the latlong values to be made so that the correct GT for each instance is chosen in an automated fashion. The end result of this is a more accurate semantic similarity calculation between attributes, and ultimately, between tables. In figure 3.4, using latlong information, it can now be determined unequivocally that the Clinton instances represent different GTs, and thus, should not be matched.

One crucial detail worth mentioning regarding the use of latlong values for GT identification is the natural variation in latlong values displayed by gazetteers. This may come about either because of differing numbers of significant digits in the coordinate values, cartographic projection, or due to differences in the scale, or level of detail, of geographic features. For instance, if our data contains an instance known as "Example Ave.", with a latitude value of "43.24323", and our gazetteer contains this instance at the same level of detail, but with a latitude value of "43.2432332", then in all likelihood, this should be considered a match. To solve this problem, we use a distance tolerance measure that discounts significant digits to the right of the

decimal point in the latlong value that are not deemed crucial for the match. The number of significant digits that are discounted depends on the features being matched, and their level of accuracy. Every time an instance in our data set is matched to an instance in the gazetteer, we first determine the geographic type of the instance. Afterwards, we classify the instance match according to 9 possible levels of accuracy, with the lowest level of detail (level 1) being country, and the highest level of detail (level 9) being "premise", which includes building names, property names, shopping centers, etc. We modeled our accuracy hierarchy after version 2 of Google's Reverse Geocoding API(Google). Using the level of detail of the feature data, in addition to the feature type of the instance, we can determine the number of significant digits to discount.

Algorithm 2 below, depicted as figure 3.5, outlines the final geographic type lookup algorithm, including both the naïve geographic type lookup algorithm and the more sophisticated version which matches exactly one type to each instance.

Algorithm 2 lookupGeoTypes ($\mathcal{T}_{gaz}$, IL)

Input:
-Set of geographic types $\mathcal{T}_{gaz}$ recognized by gazetteer
- List of instances IL associated with attribute att(T)
Output: an ordered pair ($\mathcal{T}_{att}$, $\mathcal{W}_{att}$) across all instances of att(T)

1: $\mathcal{T}_{att} = \Phi$, $\mathcal{W}_{att} = \Phi$
2: For each instance x ε IL {
3: T_x = typeLookup ($\mathcal{T}_{gaz}$, x)
4: if hasLatLong(x) {
5: prune(T_x)
6: $\mathcal{W}_x = w_1 = 1$
7: } else {
8: For each t ε T_x {
9: $w_t = 1 / | T_x |$
10: $\mathcal{W}_x = \{w_1 \ldots w_{last}\}$
11: }///end for
12: }//end if
13: $\mathcal{T}_{att} = \mathcal{T}_{att} \cup T_x$
14: $\mathcal{W}_{att} = \mathcal{W}_{att} \cup \mathcal{W}_x$
15:}//end for
16: return ($\mathcal{T}_{att}$, $\mathcal{W}_{att}$)

Figure 3.5. Algorithm 2 – GT matching in GeoSim

The input to algorithm 2 is the list of available GTs that are recognized by gazetteer G, along with IL, the list of instances associated with a given attribute and the gazetteer G itself, while the output is an ordered pair consisting of the GT vector list and GW vector list for the given attribute. Line 2 begins a loop that considers all instances in IL. Line 3 retrieves the set of GTs from T_{gaz} that instance x is associated with. Line 4 determines if instance x contains latlong information. If so, then it is possible to prune the number of possible geographic types for instance x to exactly one while assigning a weight of this type to be 1. This occurs on lines 5 and 6. If x does not contain any latlong information, then Lines 8-10 derive all possible types T_x for x and assign the weight of each type associated with the current instance. Lines 13-14 aggregate the GT and weight vectors computed for instance x to T_{att} and W_{att}, respectively. Finally, these vectors are returned as an ordered pair to GSim, which facilitates the EBD calculation between two compared attributes.

3.2.3 Accounting for Hierarchical GTs

Some gazetteers contain a hierarchical feature type thesaurus, many of which are freely accessible(Open Geospatial Consortium). One example is the ADL gazetteer(Alexandria Digital Library Project); figure 3.6 shows the segment of the feature type hierarchy that represents manmade features. As of now, GSim assigns the most general GT to a given instance. For a gazetteer with a flat feature type system, this is not a problem, as there will be no doubt about the GTs of instances when computing an EBD score between compared attributes. However, for a hierarchical type system, the final EBD score depends upon the specificity of the GT assignments of the instances. For example, in Figure 3.6 below, if instances are assigned GTs that are no more

specific than “Manmade Features”, as illustrated by Cutoff 1, then any instance that is an “Agricultural Site”, “Commercial Site”, etc. will have a GT of “Manmade Feature”. As a result, the calculated EBD between compared attributes with these instances is likely to be higher.

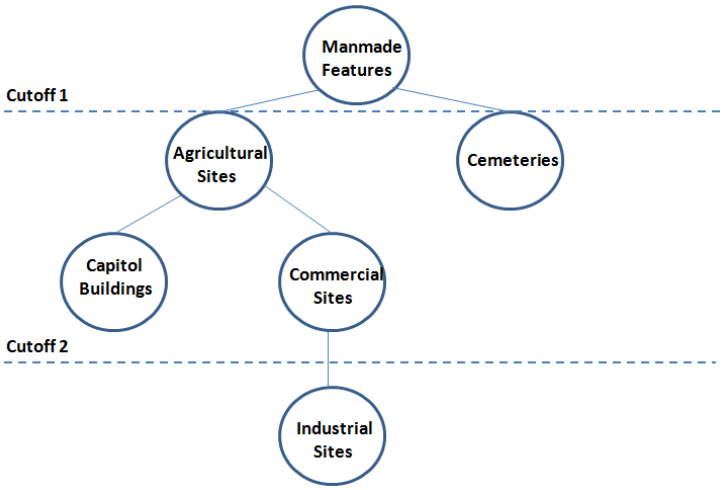


Figure 3.6. Segment of the ADL gazetteer’s GT hierarchy for manmade features.

In reality, though, the EBD value is more likely to be overestimated. On the other hand, if we assigned GTs to instances that may be as specific as “Commercial Sites” or “Capitol Buildings”, as indicated by Cutoff 2, then it is very possible that many of the instances that were labeled as “Manmade Features” using Cutoff 1 would now be labeled as a more specific GT, such as “Capitol Building” or a “Cemetery”. This would result in a final EBD value between compared attributes that is lower than if Cutoff 1 was used. However, in reality, the EBD is likely to be underestimated compared to an EBD score derived from a situation where the user was interested in assigning GTs no more specific than “Agricultural Site”. Although this problem is not the focus of our work, we are continuing to study it by carrying out additional experiments. The goal

is to determine the cutoff that yields the highest EBD value while sacrificing an acceptable amount of GT specificity.

3.3 Non-Geographic Matching

If GT matching between compared attributes is not possible, then a non-geographic semantic similarity measure is applied by GSim. The distance metric used for NGT matching is known as the normalized Google distance. The EBD is then calculated by extracting the keywords making up compared instances and assigning them generalized semantic types. These types are represented as clusters of keywords, whose semantic distance from each other is given by GD.

Figure 3.7 below illustrates each step involved with NGT matching.

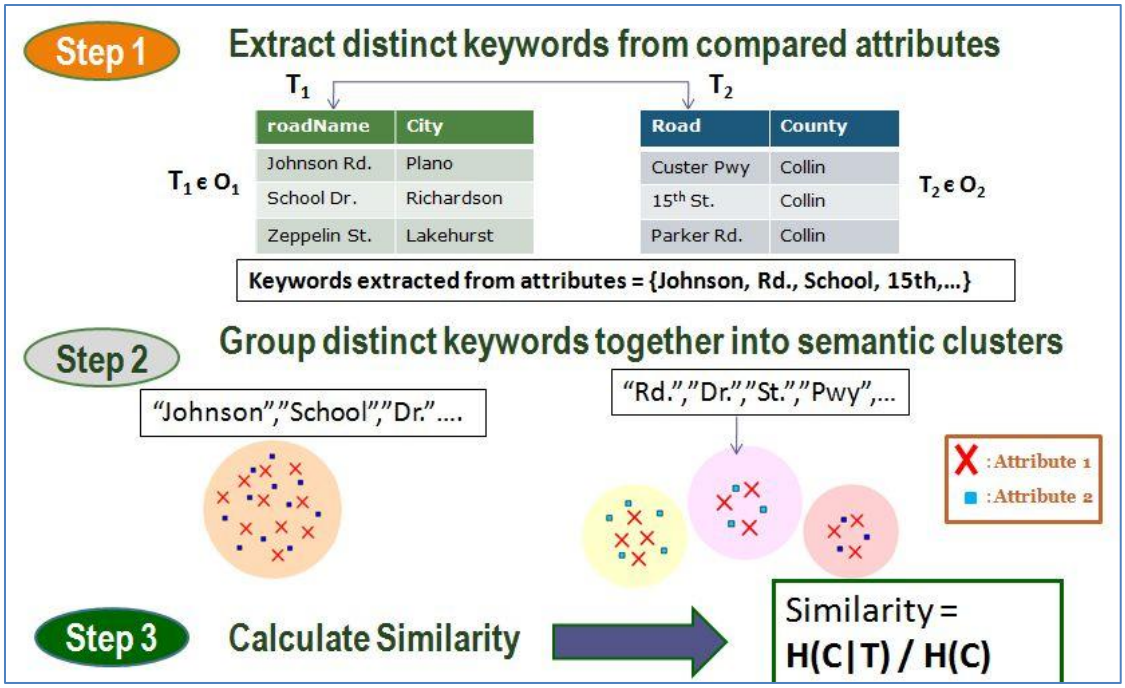


Figure 3.7. Overview of NGT matching

Step 1 illustrates the extraction of keywords of instances from a 1:1 attribute comparison. Step 2 illustrates the clustering step, which involves an algorithm known as K-medoid clustering and a semantic distance measure called Google Distance. Google Distance will be described in section 3.3.1 and K-medoid clustering will be described in section 3.3.3. Once the clustering is complete, EBD is calculated, where types in this case are represented by semantic clusters as deemed by GD.

3.3.1 Google Distance

GD is formally defined (Cilibrasi 2007, 370-83) as follows:

$$\mathbf{NGD}(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log M - \min\{\log f(x), \log f(y)\}} \quad (2)$$

In this formula, f(x) is the number of Google hits for search term x, f(y) is the number of Google hits for search term y, f(x,y) is the number of Google hits for the tuple of search terms xy, and M is the number of web pages indexed by Google. GD(x,y) is a measure for the symmetric conditional probability of co-occurrence of x and y. In other words, given that term x appears on a web page, GD(x,y) will yield a value indicating the probability that term y also appears on that same web page. Conversely, given that term y appears on a web page, GD(x,y) will yield a value indicating the probability that term x also appears on that page.

3.3.2 Algorithm Details

The algorithm for calculating the EBD between two compared attributes of tables in different data sources using NGT matching is as follows. The input is two compared attributes,

with each one originating from a separate table, while the output is an EBD value indicating the semantic similarity between the input attributes. First, the respective keyword lists for each input attribute are extracted. Second, the keyword lists are combined into a single list for the comparison. This list is dubbed as $L_{keywords}$. Third, all pairwise distances between the keywords are computed with the help of an external GD repository, resulting in a pairwise GD dictionary. Fourth, the K-medoid algorithm, which is described in the next section, is executed, yielding a set of clusters, or NGTs, that represent generic types. Finally, the calculation of EBD proceeds given the NGTs produced by K-medoid clustering.

3.3.3 K-Medoid Clustering

The algorithm begins by determining the number of clusters K based on the size of $L_{keywords}$ for each pair of compared attributes. Second, exactly one keyword from $L_{keywords}$ is assigned to each of the K clusters in a process called initial seeding. Each of these keywords is then considered a medoid for its particular clustering. Third, we continuously assign each remaining keyword in $L_{keywords}$ that is not a medoid to the cluster to which it is most semantically related. Once we have assigned all keywords in $L_{keywords}$, the algorithm determines if any cluster medoids need to be recomputed. To do this, we need to use the GD values between the keyword to be assigned to a cluster and all keywords already assigned to that same cluster. A given keyword, knew is assigned to the cluster associated with the smallest summation of the GD values between knew and the cluster's constituent keywords. After all keywords have been assigned to clusters, finally, we determine if the medoid for any cluster needs to be recomputed. This is accomplished by examining each of the keywords in a particular cluster and computing a GD

summation between a single keyword in that cluster and all other words in that cluster. The keyword in that cluster that produces the lowest GD summation will be assigned as the new medoid for that cluster. If no medoids have changed in any cluster, then the K-medoid algorithm is finished, and control proceeds to the calculation of the EBD between the compared attributes. However, if at least one medoid has changed in a particular cluster, then we begin a new clustering iteration.

3.3.4 Problem With Google Distance For NGT Matching

Despite the utility of GD over a number of domains, it tends to produce inaccurate results with regards to the GIS domain when the compared instances are geographically proximate, despite being completely different types. Figure 3.8 describes one particular example of this phenomenon. It serves as a justification of why GT matching is performed.

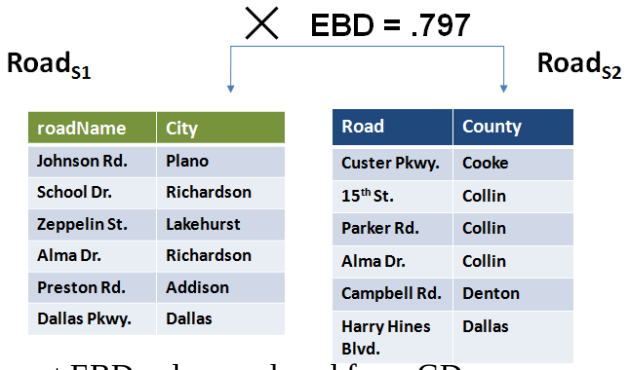


Figure 3.8. Incorrect EBD value produced from GD

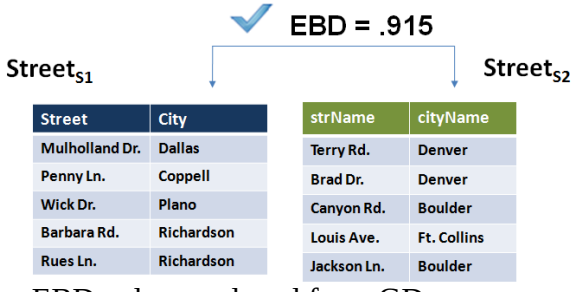


Figure 3.9. Correct EBD value produced from GD

The attribute “City”, associated with table $Road_{S1}$ is compared against the attribute “County” from table $Road_{S2}$. Although the instances are of different types, they are geographically proximate, as both the cities from “City” and the counties from “County” both describe the Dallas-Fort Worth area. As a result, even though the types are totally different, the exclusive usage of GD for NGT matching will deem that the “City” attribute is semantically similar to the “County” attribute. This happens because GD, by definition, is computed based on the probability of the co-occurrence of search terms x and y on a given web page indexed by the Google search engine. In many situations, a high probability of co-occurrence between x and y indicates that the terms are likely to be semantically similar to one another. However, as figure 3.8 shows, co-occurrence does not always imply similarity.

3.3.5 Proposed Solution to Google Distance Problem

We propose a solution to overcome the matching problem inherent in the GD method outlined in the previous section.

The proposed idea can be split into two separate parts. First, we try to resort to alternative means of acquiring the GT of an instance, if we cannot determine its type from GeoNames. We may use any number of other gazetteers to directly acquire the type from their type thesauruses, use Wikipedia to determine the type based on the Wikipedia category associated with the instance, or retrieve the top M highest-ranking Web pages from Google, where M is a threshold indicating a maximum number of Web pages, and use geotagging on the names of the instances. We could also integrate this step as part of our GT matching algorithm; this way, if we need to resort to

NGT matching, then we know that we have tried all possible geographic repositories to make GT matching work.

The second part of the solution would be executed if GT similarity was attempted, but was not able to determine the types of a sufficient number of instances (In our experiments, 50% of the total number of instances between the compared attributes having GTs is sufficient for GT matching). In this case, we resort to NGT matching and group the instances of the compared attributes into NGTs based on GD. Each NGT would be represented as a cluster of semantically related instances from both attributes. Among these instances in each cluster, some would have GTs that were explicitly determined from the previously attempted GT matching, and some would not have any GTs. During each 1-1 attribute mapping over NGTs, we would be able to use the instances with GTs from the previously attempted GT matching to verify whether GD has correctly clustered instances together, and thus, if NGT matching has produced a correct attribute match.

For each NGT, we are using those instances with GTs to guide us in determining its quality. Informally, if an NGT contains mostly instances associated with more than 1 GT, then the NGT is deemed impure. However, if an NGT contains mostly instances with associated with a single GT, then the NGT is deemed pure. If an acceptable number of the instances throughout all of the NGTs have been deemed pure (equal to or exceeding a predefined threshold), then the attribute match is verified to be correct. However, if too many instances across all NGTs have been deemed impure (below a threshold value), then the attribute match is verified as being incorrect. The result of this is a readjustment of the final EBD score between the attributes by changing the contribution that each NGT makes.

Figure 3.10 illustrates an impure NGT (on the left) and a pure NGT (on the right). As can be seen, each NGT represents instances of an attribute comparison between two attributes.

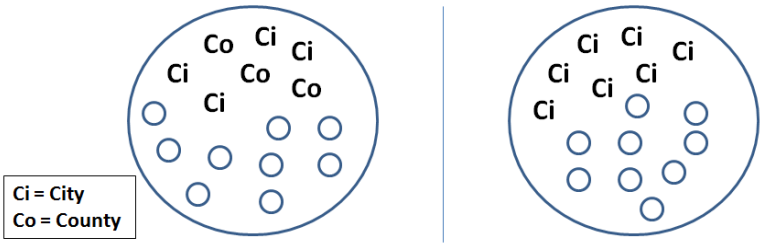


Figure 3.10. NGTs containing instances whose GTs were explicitly determined, and instances whose GTs are unknown.

The left NGT is derived from the attribute comparison depicted in Figure 3.8, while the right NGT is derived from the attribute comparison depicted in Figure 3.9. We will assume that in Figure 3.8, any instances in the City attribute with GTs have a GT of type "City", while any instances in the County attribute with GTs have a GT of type "County". In Figure 3.9, we will assume that all instances from both attributes that have GTs are of type "City". In both NGTs of Figure 3.10, an instance labeled by a gazetteer with "C_i" represents the GT "City", while an instance labeled by a gazetteer with "Co" represents the GT "County". The empty white circles indicate instances whose GT could not be determined explicitly by GSim. The NGT on the left, which results from the attribute comparison of Figure 3.8, is impure. To understand this, we first can see four city instances from the "City" attribute and three county instances from the "County" attribute. We also have a number of instances from both attributes whose GT cannot be determined. Since the instances collectively refer to more than one GT, we can infer that the NGT is impure. We may infer this even if the GD similarity between the two says otherwise. As a result of the impurity of the NGT, we may lower its weight in the EBD calculation between the

attributes. For the NGT on the right produced from the attribute comparison of Figure 3.9, all instances whose types are known share a single GT of type "City". Thus, the NGT is pure. If this is the only NGT between the compared attributes, we would conclude that the mapping between attributes in this case is correct.

3.4 Experiments

We now present three separate experiments that we conducted regarding geographic and non-geographic matching between distinct data sources in the GIS domain. The first experiment measured GSim's ability to compute semantic similarity between two pairs of GIS databases. The GT matching used here is naïve GT matching. The second experiment applied the use of latlong techniques to disambiguate between the GTs of instances in an attempt to improve our results. The third experiment illustrates GSim's NGT matching component, in a situation where GT matching is not possible, and we compare the results generated with those from the prior GT matching experiments.

3.4.1 Dataset Details

Figures 3.11, 3.12 and 3.13 below lists the details of three separate datasets to which we applied the GSim algorithm, along with some baseline methods of calculating semantic similarity.

Table Name	No. of Attributes	Areas Modeled	No. of Instances
Road(S_1), Road(S_2)	5,5	Fort Collins, CO Dallas, TX	1970, 1045
Ferry(S_1), Ferry(S_2)	4,3	Seattle, WA	24,42
Traffic Area(S_1), Traffic Area(S_2)	3,3	Virginia	329,108
Residential Area(S_1), Residential Area(S_2)	5,4	New Jersey, Texas	852,424

Figure 3.11. Description of GIS Transportation Dataset (GTD)

Table Name	No. of Attributes	No. of Instances
Flight Schools(S_1), Flight Schools(S_2)	7,8	930, 930
Schools(S_1), Schools(S_2)	3,3	11435, 11890
Piers(S_1)	6	1232
Indian Lands(S_1)	3	3160
Ports(S_2)	5	907
NavWaterways(S_2)	4	1377

Figure 3.12. Description of GIS Location Dataset (GLD)

Table Name	No. of Attributes	No. of Instances
Hospitals(S_1), Hospitals(S_2)	4,4	829 (for both)
Schools(S_1), Schools(S_2)	4,4	5768 (for both)
Streets1(S_1), Streets2(S_2)	6,6	1384 (for both)

Figure 3.13. Description of GIS POI Dataset (GPD)

In Figures 3.11, 3.12 and 3.13, tables from different data sources are listed either individually or in pairs. When they are listed in pairs, this implies that the tables are semantically similar, whereas if a table is listed individually (such as the table 'Indian Lands' in the GIS Location Dataset), then this implies that the table does not semantically match with another table. Also, for each table(s), the number of attributes and instances reflects the number involved in semantic matching, as opposed to the actual number of attributes or instances that exist within the table(s). Most of the attributes for each table remained unused either because they did not contain string data (and thus were not eligible for a match), or because they were not relevant enough to be used in our semantic matching experiments.

Now the details of each data set will be described. The first dataset, which we dubbed the GIS Transportation Dataset in figure 3.11, was created from instance data of the Road and Ferries package of a GIS data model known as GDF (Geographic Data Files)(Ertico). The tables vary in regards to number of attributes (with the smallest being in Ferry(S_1), 3, and the largest being in

Traffic Area(S_2) with 5), number of instances (smallest being Ferry(S_1) with 24, and the largest being Road(S_1) with 1970), and in regards to geographic area (data models 6 different states spread across the lower 48 states). We preferred data featuring a wide geographic dispersion with no shared instances. Therefore, similarity between tables would only be possible via a semantic match, as opposed to simple keyword matching. Furthermore, we considered this dataset to be multijurisdictional. The second dataset, which we dubbed the GIS Location Dataset in figure 3.12, details a wider assortment of location features across the United States and their associated data beyond merely transportation networks. Some of the location features in this dataset include flight schools, piers, navigable waterways and Indian lands. As with the GIS Transportation Dataset, the number of attributes and instances vary; for example, in the GIS location dataset, the Flight Schools table for S_2 has the largest number of attributes taking part in matching (8) and the both Schools tables and the Indian Lands table has the fewest (3). In regards to instances, Schools(S_2) contains 11890 instances, the most in the dataset, whereas Ports(S_2) contains the fewest number of instances at 907. As with the GIS Transportation Dataset, the instances in the tables of this dataset are multijurisdictional in nature. The third dataset, which we dub the GIS Point of Interest (POI) Dataset in figure 3.13, contains instances that extend beyond road networks and which are multijurisdictional in nature, much like the GIS Location dataset. The number of instances and locations modeled vary widely, which results in a dataset that requires semantic methods by an algorithm for any meaningful schema matching to occur.

3.4.2 Geographic Similarity Without Latlong Values

We will now present measurements, parameters and results regarding GSim's GT matching experiments.

3.4.2.1 Measurements and Parameters

We will use the results depicted in figure 3.14 and figure 3.15 to illustrate how the results are reported. Figure 3.14 depicts the alignment of S_1 and S_2 of their compared tables for the GIS transportation dataset, and figure 3.15 does the same, but over the GIS location dataset.

$t \varepsilon S_1$	$t \varepsilon S_2$	P (N-gram)	R (N-gram)	F-measure (N-gram)	P (GSim)	R(GSim)	F-measure (GSim)
Road	Road	.50 (1/2)	.25 (1/4)	.20	.50 (2/4)	.50 (2/4)	.50
Road	Address Area	.25 (1/4)	1.00(1/1)	.40	1.00 (1/1)	1.00(1/1)	1.00
Road	Enclosed Traffic Area	.33 (1/3)	.50 (1/2)	.40	.50 (1/2)	.50 (1/2)	.50
Road	Ferry	0 (0/2)	0 (0/1)	0	.50 (1/2)	1.00 (1/1)	.67
Residential Area	Road	.25 (1/4)	1.00 (1/1)	.40	.50 (1/2)	1.00 (1/1)	.67
Residential Area	Address Area	.50 (2/4)	.50 (2/4)	.50	.75 (3/4)	.75 (3/4)	.75
Residential Area	Enclosed Traffic Area	.25 (1/4)	.50 (1/2)	.33	.50 (1/2)	.50 (1/2)	.50
Residential Area	Ferry	------ (0/0)	0 (0/1)	0	1.00 (1/1)	1.00 (1/1)	1.00
Traffic Area	Road	.33 (1/3)	.50 (1/2)	.40	.50 (1/2)	.50 (1/2)	.50
Traffic Area	Address Area	.50 (1/2)	.50 (1/2)	.50	1.00 (1/1)	.50 (1/2)	.67
Traffic Area	Enclosed Traffic Area	.50 (1/2)	.50 (1/2)	.50	1.00 (2/2)	1.00 (2/2)	1.00
Traffic Area	Ferry	.50 (1/2)	1.00 (1/1)	.67	1.00 (1/1)	1.00 (1/1)	1.00
Ferry	Road	.50 (1/2)	1.00 (1/1)	.67	1.00 (1/1)	1.00 (1/1)	1.00
Ferry	Address Area	1.00 (1/1)	1.00 (1/1)	1.00	1.00 (1/1)	1.00 (1/1)	1.00
Ferry	Enclosed Traffic Area	.50 (1/2)	1.00 (1/1)	.67	1.00 (1/1)	1.00 (1/1)	1.00
Ferry	Ferry	.50 (1/2)	.33 (1/3)	.4	.67 (2/3)	.67 (2/3)	.67
AVERAGE VALUES		.38	.52	.44	.70	.72	.71

Figure 3.14. Precision, recall and F-measure values between tables of S_1 and S_2 using N-grams and GSim relative to ground truth for GIS Transportation Dataset

From left to right, the first two are the precision and recall (denoted as P and R, respectively) produced using N-grams between an attribute from a table in data source S_1 and an attribute from a table in data source S_2. The last two values are the precision and recall values

produced by GSim between an attribute from a table in data source S_1 and an attribute from a table in data source S_2.

$t \varepsilon S_1$	$t \varepsilon S_2$	P (N-gram)	R (N-gram)	F-Measure (N-gram)	P (GSim)	R(GSim)	F-Measure (GSim)
Flight Schools	Flight Schools	1.00 (2/2)	.29 (2/7)	.45	1.00 (7/7)	1.00 (7/7)	1.00
Flight Schools	Schools	------ (0/0)	0 (0/3)	0	1.00 (2/2)	.67 (2/3)	.80
Flight Schools	Ports	------ (0/0)	0 (0/3)	0	.33 (1/3)	.33 (1/3)	.33
Flight Schools	NavWaterways	------ (0/0)	0 (0/3)	0	.33 (1/3)	.33 (1/3)	.33
Schools	Flight Schools	------ (0/0)	0 (0/3)	0	1.00 (3/3)	1.00 (3/3)	1.00
Schools	Schools	.66 (2/3)	.33 (1/3)	.44	1.00 (3/3)	1.00 (3/3)	1.00
Schools	Ports	------ (0/0)	0 (0/3)	0	1.00 (1/1)	.33 (1/3)	.50
Schools	NavWaterways	------ (0/0)	0 (0/3)	0	1.00 (3/3)	1.00 (3/3)	1.00
Indian Lands	Flight Schools	------ (0/0)	0 (0/3)	0	.33 (1/3)	.33 (1/3)	.33
Indian Lands	Schools	------ (0/0)	0 (0/3)	0	1.00 (1/1)	.33 (1/3)	.50
Indian Lands	Ports	------ (0/0)	0 (0/3)	0	1.00 (1/1)	.33 (1/3)	.50
Indian Lands	NavWaterways	------ (0/0)	0 (0/2)	0	1.00 (2/2)	1.00 (2/2)	1.00
Piers	Flight Schools	------ (0/0)	0 (0/5)	0	1.00 (2/2)	.40 (2/5)	.57
Piers	Schools	------ (0/0)	0 (0/3)	0	1.00 (1/1)	.33 (1/3)	.50
Piers	Ports	------ (0/0)	0 (0/3)	0	.50 (1/2)	.67 (2/3)	.57
Piers	NavWaterways	------ (0/0)	0 (0/2)	0	.67 (2/3)	1.00 (2/2)	.80
AVERAGE VALUES		.80	.06	.09	.80	.61	.68

Figure 3.15. Same as figure 3.14, but for GIS Location Dataset

As an example, for the comparison of Road from S_1 and Ferry from S_2 in figure 3.14, the precision and recall generated using N-grams are 0 and 0, respectively, while the precision and recall generated for GSim is .50 and 1.00, respectively. Also, for each cell containing a precision or recall value, there is a ratio. For precision, the top number of the ratio indicates the number of correct attribute mappings between the compared tables that were identified by the similarity method, while the bottom number indicates the total number of attribute mappings (both correct and incorrect) between the compared tables identified by the matching method. For recall, the top number of the ratio indicates the number of correct attribute mappings between the tables that were returned by the similarity method, while the bottom number of the ratio indicates the total number of correct attribute mappings that exist between the tables.

For instance, in figure 3.14, for the comparison of the Road table from S_1 with the Road table from S_2, the ratio in the cell for the precision of the N-gram method is "1/2", meaning that the N-gram method returned two attribute mappings between these two tables, but only one was correct. The cell to its right, which is the recall value produced by the N-gram method for the Road-Road table comparison, reads "2/4". This means that two correct attribute mappings were returned by the N-gram method, while there exists a total of 4 correct attribute mappings between the tables.

The values produced by both methods depend on a reference alignment, or ground truth, which contains the attribute pairs that are supposed to be semantically similar. The ground truth for both datasets was created by human experts knowledgeable in the area of GIS. For our experiments, we set two standards that affected the results. First, we decided that whenever an attribute pair produced a similarity value (an EBD value) measured to be greater than or equal to .6, then the method determined that pair to be a match. Second, we set N-grams to be of size 2, since any size greater than 2 would increase the number of possible N-grams by a margin significant enough such that the precision and recall values would almost always be too low to meet the match threshold for any dataset, thus rendering this method virtually useless as a semantic similarity measure for our experiments. Overall, the ground truth for the transportation dataset contained 29 correct mappings across all table comparisons, while the ground truth for the GIS location dataset contained 52 correct mappings across all table comparisons.

It should also be noted that in our experiments, valid attribute mappings were found even between tables that do not naturally correspond. For instance, in the GIS POI dataset, valid attribute mappings exist between disparate tables like Streets1 and Schools2 (the City attribute, in

this case). These mappings, and others which exist among the other datasets that we experimented upon, were included in our reference alignments.

3.4.2.2 Analysis of Results

Figure 3.14 shows the comparison of precision, recall and F-measure values using both GSim and the N-gram method for the transportation dataset. Note that the precision and recall values generated by GSim are never lower than those produced by N-grams for any table comparison. In total, the average precision produced by GSim was .70, and its average recall was .72. In contrast, the average precision of N-grams was .38, and its average recall was .52. GSim achieved a 32% improvement over N-grams in precision, and a 20% improvement in recall. Figure 3.15 depicts even more dramatic improvements made by GSim. The precision and recall values for GSim are always higher than those produced by the N-gram method for any table comparison. In total, the average precision produced by GSim was .80, and its average recall was .61. In contrast, while the average precision of N-grams is .80, the average recall is a staggeringly low value of .06. In fact, the reason why N-grams' precision was able to match GSim's precision was due to the extremely low recall. The reason for the low recall value was primarily due to the lack of identical instances between the compared attributes. As a result, most of the comparisons using the N-gram method were not able to reach the .60 threshold in semantic similarity. We did not lower the match threshold below .6 because we felt that a match threshold of a value that was lower, such as .5, would not be a realistic match threshold for determining whether two schemas were similar or not. The reason is that at lower thresholds, the precision and recall values generated by sophisticated and simplistic algorithms alike are not significantly

different. As a result of the higher threshold, GSim more clearly illustrates its more sophisticated semantic capabilities, largely resulting from GT extraction. This allows it to achieve a 55% improvement on recall versus a syntactic method such as N-grams.

In figure 3.14, the only reason why N-grams even performed somewhat competently was because of the large number of identical instances between many attribute pairs that happened to be similar. For the N-gram method to derive an attribute mapping, the instances between the compared attributes must share strings of length N. As an example, with N=2, the instances "Pasadena" and "El Paso" (from different attributes) would share a single 2-gram match on "Pa". Given enough matches of this sort between instances of two compared attributes, the N-gram method will derive a similarity score that meets the threshold of .60, registering as an attribute mapping. In table comparisons where the N-gram method derived a precision or recall value that was extremely low, the instances in the compared attributes shared few strings. This is what makes the N-gram method a syntactic method, as opposed to GSim. In figure 3.15, the N-gram method creates very few mappings that even reach the threshold of .60, because the GIS location dataset contains very few shared strings between valid attribute mappings of two tables.

Notice that when applying GSim to pairs of tables which seem incompatible (ie: Road-Address Area), it still yields some attribute matches, as evidenced by nonzero precision and recall scores. This is because valid attribute matches can exist between tables which are not compatible. An example of this is Road(S_1).County-AddressArea(S_2).Areaname – even though these tables are not related, they share this attribute, and thus, a match should exist. GSim is able to identify these kinds of attribute mappings, regardless of whether the compared tables seem compatible or not. This is evidenced by the 1.0 precision values between tables as different as Residential Area

(S_1) and Ferry (S_2) in the GIS transportation dataset, and between Schools (S_1) and Ports (S_2) in the GIS location dataset.

3.4.3 Geographic Similarity Without Latlong Values

As we did for naïve GT matching, we will first present the measurements and parameters used for the experiments involving latlong GT matching before we proceed with the analysis of the results.

3.4.3.1 Measurements and Parameters

Figure 3.16 below displays precision, recall and F-measure values in the GIS POI (point of interest) dataset comparing semantic similarity generated by the baseline N-gram method, GSim without the use of latong values, and GSim with the use of latlong values. The GIS POI dataset represents, as the name implies, a multijurisdictional collection of streets, schools and hospitals that are identified as points of interest in GeoNames. Like our previous experiments, the values produced by both N-grams and GSim in this dataset depend on a reference alignment which contains the attribute pairs that are semantically similar. The ground truth for both datasets was created by human experts knowledgeable in the area of GIS. However, in this experiment, we also directly compare the benefits that latlong values have on deriving similarity.

3.4.3.2 Analysis of Results

As figure 3.16 shows, not only does GSim produce markedly better results versus the N-gram approach, but when GSim has access to latlong values for the purposes of further

disambiguating between the GTs of instances, the results are even better. As can be seen, GSim without latlong values has an average precision of 1.00, while the average precision value for N-grams is .86.

$t \epsilon S_1$	$t \epsilon S_2$	P (N-gram)	R (N-gram)	F-Measure (N-gram)	P (GSim) w/o latlong	R(GSim) without latlong	F-Measure (GSim) without latlong	P (GSim) latlong	R(GSim) latlong	F-Measure (GSim) latlong
Streets1	Streets2	.50	.50	.50	1.00	1.00	1.00	1.00	1.00	1.00
Streets1	Schools2	0	0	0	0	0	0	1.00	.50	.67
Schools1	Streets2	0	0	0	0	0	0	1.00	1.00	1.00
Schools1	Schools2	1.00	.50	.67	1.00	1.00	1.00	1.00	1.00	1.00
Schools1	Hospitals2	1.00	.50	.50	1.00	.50	.67	1.00	1.00	1.00
Hospitals1	Schools2	1.00	.50	.67	1.00	.50	.67	1.00	.50	.67
Hospitals1	Hospitals2	1.00	.50	.67	1.00	1.00	1.00	1.00	1.00	1.00
AVERAGE VALUES		**.86**	**.29**	**.43**	**1.00**	**.76**	**.86**	**1.00**	**.90**	**.95**

Figure 3.16. Precision, recall and F-measure values between tables of S_1 and S_2 using N-grams, GSim without latlong values and GSim with latlong values relative to ground truth for GPD dataset

This amounts to a 16% improvement in precision by using GSim. As for average recall, GSim without latlong values produces a value of .76, while N-grams produces a value of .29. This represents a nearly threefold improvement in recall for GSim over N-grams. As for the average F-measure, GSim produces a value of .86, while N-grams produces a value of .43. In other words, GSim produces an F-measure that is twice as good as the F-measure for N-grams.

In addition to this, figure 3.16 shows that the use of latlong values in GSim produces further improvement. Using GSim with latlong values, average recall is measured at .90, an 18.4% increase over GSim without latlong values (.76). As for average F-measure, GSim with latlong values produces a value of .95, a 10.4% improvement over GSim without using latlong values (.86). Before the use of latlong values, a number of instances (especially those with common names) between any two compared attributes might possess GT sets of a size > 1. The end result of this was that instance that were genuinely of the same GT but were tagged with multiple semi-

overlapping GTs would have their similarity diminished unfairly, while instances that were genuinely not of the same GT but were tagged with multiple semi-overlapping GTs would have their similarity bolstered unfairly. However, using latlong values, if the instance is recognized by the gazetteer, then a 1:1 mapping between it and its correct GT is guaranteed to exist. Because of this, correct correspondences have their score raised, thus explaining the improved scores.

3.4.4 Non-Geographic Similarity

To illustrate the effectiveness of GSim's NGT matching component and to compare it to its GT matching component, we replaced instances from the GIS transportation dataset that were previously identified by a gazetteer with new instances whose type could not be discerned. Figure 3.17 shows the results of NGT matching applied to the GIS transportation dataset.

The precision, recall, and F-measure values are all better than what the N-gram method produced, but they are not as good as the results of GT matching on this dataset, as seen in figure 3.14. Specifically, the average precision produced by NGT was 45% higher than the precision produced by N-grams, but 21% lower than the precision produced using GT matching. The recall produced by NGT was 17% higher than that produced by N-grams, but 18% lower than the recall attained by GT matching.

3.4.5 GSim vs. Other Methods

We also sought to compare the effectiveness of GSim relative to two other widely accepted methods for determining the semantic similarity of sets of documents (or data sources) sing keyword frequency. These methods are known as nonnegative matrix factorization (NMF)

$t \varepsilon S_1$	$t \varepsilon S_2$	P (NGT)	R (NGT)	F-measure (NGT)
Road	Road	.50 (2/4)	.50(2/4)	.50
Road	Address Area	1.00 (1/1)	1.00(1/1)	1.00
Road	Enclosed Traffic Area	.50 (1/2)	.50 (1/2)	.50
Road	Ferry	0 (0/1)	0 (0/1)	0
Residential Area	Road	.33 (1/3)	1.00 (1/1)	.50
Residential Area	Address Area	.50 (2/4)	.50 (2/4)	.50
Residential Area	Enclosed Traffic Area	.50(1/2)	.50 (1/2)	.50
Residential Area	Ferry	1.00 (1/1)	1.00 (1/1)	1.00
Traffic Area	Road	.50 (1/2)	.50 (1/2)	.50
Traffic Area	Address Area	.50 (1/2)	1.00(1/1)	.67
Traffic Area	Enclosed Traffic Area	.50 (1/2)	.50 (1/2)	.50
Traffic Area	Ferry	.50 (1/2)	1.00 (1/1)	.67
Ferry	Road	1.00(1/1)	1.00 (1/1)	1.00
Ferry	Address Area	1.00 (1/1)	1.00 (1/1)	1.00
Ferry	Enclosed Traffic Area	1.00 (1/1)	1.00 (1/1)	1.00
Ferry	Ferry	.50 (1/2)	.33 (1/3)	.40
AVERAGE VALUES		.55	.61	.58

Figure 3.17. Precision, recall and F-measure values produced by the NGT matching component of GSim.

(Lee 2000, 556-62) and singular value decomposition (SVD) (GSL Team). NMF is an algorithm in linear algebra where a matrix X is factorized into two matrices, W and H. Formally, this is stated as: NMF(X) = WH. NMF differs from other matrix factorization methods in that all entries of W and H are to be nonnegative; this is especially applicable for applications of semantic similarity via keyword frequency, since the minimum frequency of any given keyword in a data source is 0. In SVD, the equation $M = U \Sigma V^*$ is satisfied, where U is an mxm unitary matrix over a field K, Σ is an n x m diagonal matrix with nonnegative real numbers along the diagonal, and V* is the conjugate transpose of V, a n x n unitary matrix over the field K. Though SVD has many uses, in regards to semantic similarity, it can be applied towards the implementation of latent semantic indexing (LSI). LSI uses SVD to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of text. LSI is based on the principle that words that are used in the same contexts tend to have similar meanings.

We have applied SVD (Singular value decomposition) and NMF on the same datasets that were being used in our experiments to find out the semantic similarity between the attribute pairs of any two tables. For this, first, we have generated a matrix X_{MxN} with m rows and n columns where the row represents distinct words and the column represents its attribute name from these two tables. We have two different implementations. In the frequency variant, each entry (i,j) of the matrix represents how many times the word i appears under a particular attribute j. On the other hand, in the binary variant, each entry (i,j) of the matrix represents the presence of word i under the particular attribute j. Thus if a word i appears under an attribute j, in the binary case, the value of the entry (i,j) is set to 1, whether word i appears 1 time or 100 times under attribute j.

We have used SVD to reduce the dimension of the matrix from n to k where k << n. SVD decomposes X_{MxN} into a product of three matrices as $X_{MxN} = USV^T$ where U is an m x n matrix, S is an n x n diagonal matrix, and V^T is also an n x n matrix. The definitions of S is as follows:

$$S = diag(\sigma_1, \sigma_2, \ldots \sigma_{\min(m,n)}), \quad \sigma_1 > \sigma_2 > \ldots \sigma_{\min(m,n)} \ \& \ \sigma_j > 0 \text{ for all j > rank } (X_{MxN}).$$

To reduce the dimension we generate a new matrix S^k by keeping the k largest singular values. Next, we have calculated the cosine similarity between attribute pairs by exploiting vectors in these reduced dimensional spaces. If the similarity is above a threshold (.5 for our experiments), we declare that to be a match for a 1-1 attribute comparison

We compared the effectiveness of GSim to NMF and SVD over the three datasets we experimented on (GIS Transportation Dataset, GIS Location Dataset, GIS POI Dataset) and obtained the following results. These are displayed in figure 3.18 below. The effectiveness of each semantic similarity measure with respect to a particular dataset was quantified using F-measure. Since F-measure takes into account both precision and recall, it represents the best

overall metric to measure the effectiveness of semantic similarity algorithms over a common dataset.

In figure 3.18, it can be seen that for the GIS Transportation Dataset, the F-measure generated by GSim outperforms that from the N-gram method by 61% (.71 to .44). The difference is even greater versus SVD and NMF, as GSim outperforms SVD .71 to .13 and outperforms NMF .71 to .25. For the GIS Location Dataset, GSim outperforms N-grams in terms of F-measure .68 to .09. The stark difference in similarity values between GSim and N-grams is a direct result of this dataset not containing any shared syntactic instances. As a result, only a method that can effectively measure semantic correspondences between instances is likely to be successful over this dataset. For this same dataset, GSim outperforms SVD in F-measure .68 to .17, and GSim outperforms NMF .68 to .22. As for the GIS POI Dataset, GSim outperforms its nearest competitor, NMF, in F-measure .86 to .49. Over the 3 datasets, GSim outperforms N-grams .75 to .32, SVD by .75 to .23 and NMF .75 to .37.

Datasets	N-grams			SVD			NMF			GSim		
	P	R	F	P	R	F	P	R	F	P	R	F
GTD	.38	.52	.44	.08	.29	.13	.17	.50	.25	.70	.72	.71
GLD	.80	.06	.09	.20	.15	.17	.26	.19	.22	.80	.61	.68
GPD	.86	.29	.43	.22	.66	.33	.35	.80	.49	1.0	.76	.86

Figure 3.18. Precision, recall and F-measure values between the three datasets in our experiments over N-grams, SVD, NMF and GSim

CHAPTER 4

ATTRIBUTE WEIGHTING

Up to now, we have assumed that all 1:1 attribute matches between any pair of tables were equally important to the overall semantic similarity score generated between the tables. However, in reality, this is often not the case. In this chapter, we will discuss attribute weighting, a mechanism within GSim where the weights of a set of attribute pairs between a pair of tables are adjusted based on the relevance and uniqueness of the attributes themselves, relative to the other attributes involved in their own pairings within the same data source. After providing an overview in section 4.1, we will describe attribute uniqueness in detail in section 4.2. We will then discuss how the adjusted weightings between a set of attribute pairs mapped between the same pair of tables is derived in section 4.3. Finally, we will show some experimental results illustrating the effectiveness of attribute weighting in section 4.4.

4.1 Attribute Weighting Overview

As stated before, attribute weighting provides the capability to reward unique semantic correspondences and penalize other common semantic correspondences, where the correspondences are 1:1 attribute pairs between tables where the attributes in the mapped pair commonly-occur across all of the tables in their respective databases. Doing this allows us to refine the semantic similarity score generated between tables by focusing on the compared attributes that are unique relative to attributes found throughout all tables. Let $S_1 = (T_{11},$

T_{12}…..T_{1M}) be the set of tables belonging to data source S_1, and let S_2 = (T_{21}, T_{22}….T_{2N}) be the set of tables belonging to data source S_2, and suppose T_{1J} and T_{2K} are being compared for semantic similarity. Further suppose for the sake of simplicity that pairings between attributes of T_{1J} and T_{2K} have been set such that for all i, attribute i of T_{1J} is matched with attribute i of T_{2K}, and T_{1J} and T_{2K} have the same number of attributes. Before attribute weighting is applied, similarity calculations between attribute i of T_{1J} and attribute i of T_{2K} occur. At this point, the EBD values of each attribute pair have equal weight. Recall that attribute-level EBD tells us which attributes are similar between compared tables. We will designate one such value between two attributes as EBD_{orig} (att(T_{1J}), att(T_{2K})).

As stated previously some attribute pairs in reality should be weighted higher than others. For example, given two tables, one called Road and another called Street, if the attribute ‘roadType’ in the Road table (let us call it Road.roadType) was mapped to an attribute ‘streetType’ in the Street table (let us call it Street.streetType), then this pair should contribute more substantially to the table similarity between Road and Street than a mapped attribute pair consisting of Road.roadName and Street.streetName. While Road.roadType and Street.streetType are two attributes that are not likely to be found in many other GIS tables, Road.roadName and Street.streetName are indeed likely to appear in other GIS tables, if, for example, these tables describe geographic objects that have some kind of street address such as a school, port or business. After deciding the weights of each attribute pair among a set of mappings across compared tables, the end result will be a more accurate EBD score. This is a result of the discriminative power of attribute weighting in that it can determine the attribute pairs that are most important to the table match.

A successful attribute weighting measure ensures that an attribute pairing att1-att2 between table T and table T' is weighted more heavily than other attribute pairs between T and T' because (1: from the pairing att1-att2, att1 and its instances are relevant to table T, and att2 and its instances are relevant to table T' (2: each attribute in the pairing att1-att2 is unique to its respective table. In other words, for att1-att2 to be weighted more heavily, the frequency by which each individual attribute is found in other tables across both data sources should be small relative to other attribute pairings. In addition, it should be noted that for attribute weighting to be successful, it needs to be executed after deriving EBD measures between all attribute pairings.

4.2 Attribute Uniqueness

The uniqueness of an attribute att1 found within table T for an attribute pairing att1-att2 is known as attribute uniqueness (AU). It is determined by applying hierarchical agglomerative clustering over all attribute names in tables present throughout all data sources. Figure 4.1 shows the basic outline of this method of clustering. In the first step, each attribute that takes part in an attribute pairing is contained within its own singleton cluster. Next, two singleton clusters are merged together to form a new one containing two attributes. Each merger of two distinct clusters is known as a cluster iteration (CI). Each subsequent step merges two distinct clusters until ideally, all related attributes across tables and data sources are grouped into distinct clusters.

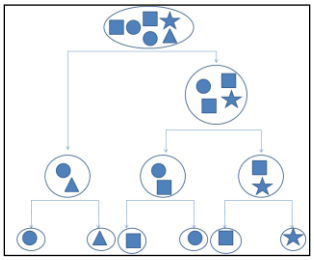


Figure 4.1. Conceptual Diagram of Hierarchical Agglomerative Clustering

The quality of the clustering, and thus the accuracy of AU values for any given attribute, depends on two factors: (1: the intercluster distance measure (2: the measure used to determine when to stop the clustering.

4.2.1 Intercluster Similarity

The intercluster similarity (ICS) measure used to determine the similarity between any two clusters A and B is expressed as follows:

$$ICS_{AB} = \frac{\sum_{a \epsilon A} \sum_{b \epsilon B} (S_N(a,b) + (S_{EBD}(a,b))}{|A| * |B|} \quad (3)$$

Here, a is an attribute name belonging to cluster A, b is an attribute belonging to cluster B, S_N is the name similarity between the names of attributes a and b, S_{EBD} is the EBD value generated between attributes a and b, |A| is the number of attributes in cluster A, and |B| is the number of attributes in cluster B. If no attribute pairing exists between attributes a and b, then we assume that the sum of S_N and S_{EBD} in this case is 0. This measure allows attribute similarity among sets of attributes within clusters to be based not only on the properties of the attribute names themselves, but also on their associated instances.

4.2.2 Cutoff Point and Calculation of Attribute Uniqueness

We add our own contribution to the standard hierarchical clustering technique through a specialized cluster stop criterion. Stopping the clustering at the most appropriate time is based on an intracluster distance measure applied after each cluster iteration over all clusters. We will refer to it as the cutoff point (CP) of the clustering. It is the average summation of the name and EBD

similarity between all valid pairings of attributes within a given cluster, taken over all clusters. It is expressed as follows:

$$CP = \frac{\sum_{A \epsilon C} \left(\frac{\sum_{a_1 \epsilon A; a_2 \epsilon A; table(a_1) \neq table(a_2)} \left(S_N(a_1, a_2) \right) + \left(S_{EBD}(a_1, a_2) \right)}{\binom{|A|}{2} - \kappa + 1} \right)}{\sum_{A' \epsilon C} |A'|} \quad (4)$$

In equation 4, C indicates the set of clusters, A is the cluster in C that contains the attributes a_1 and a_2 which are being considered for comparison, a_1 and a_2 are distinct attributes within a single cluster A, |A| is the number of attributes in cluster A, A' indicates an arbitrary cluster in C, the binomial coefficient that reads "|A| choose 2" indicates the number of possible subsets of attributes from A that are of size 2 (in other words, the number of possible pairings of attributes within cluster A), κ indicates the number of attribute pairings within A that are not possible, due to both attributes being from the same table (these do not include pairings between a_1 and a_2 of different tables that have a value = 0 for $S_N + S_{EBD}$), and |C| is the number of total clusters. The quantity, $\binom{|A|}{2} - \kappa + 1$, represents the total number of attribute pairings, with the addition of 1 to ensure that a divide by zero case never occurs.

The logic behind equation 4 is illustrated in figure 4.2. It displays a graph of the relationship between the number of cluster iterations (CI), located on the x-axis, and the cutoff point (CP), located on the y-axis. Once the average summation of S_N and S_{EBD} between all valid attribute pairs over all clusters reaches a maximum value, then the clustering is stopped, as we have attained an optimal clustering. According to Tan et al (Kumar, Steinbach and Tan 2006, 516), typical hierarchical agglomerative clustering cannot be viewed as globally optimizing an objective function.

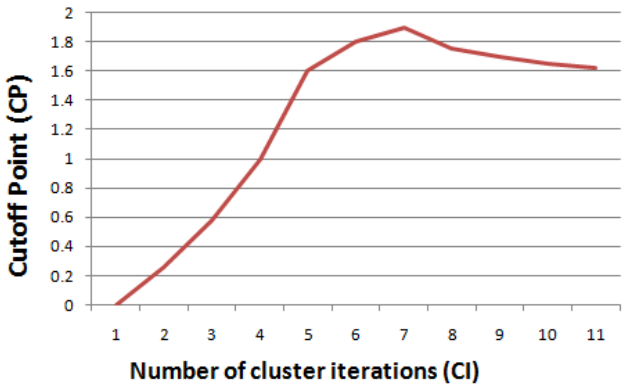


Figure 4.2. Cutoff Point vs. Number of Cluster Iterations

Rather, this type of clustering uses local criteria at each cluster iteration to merge two clusters. While standard hierarchical agglomerative clustering continues to merge clusters until the creation of one final cluster, encompassing all others, our technique uses the CP to stop the clustering prematurely, with multiple clusters remaining. Aside from how the clustering concludes, our clustering technique is identical to standard hierarchical agglomerative clustering. As a result, finding a global maximum for the CP value will be computationally infeasible. Hence, we say that the CP in Figure 4.2 represents a local maximum.

One question that naturally arises is the time complexity bottleneck that occurs as a result of the binomial coefficient term. However, since this process is executed offline, and because of the iterative algorithm reported by Manolopoulos(Manolopoulos 2002), we implemented this term to run in O(min(k, n-k)). In our case, n is equal to |A|, and k is equal to 2, making the execution time polynomial in the size of the cluster. Thus, for the reasons described above, the time complexity of this step is not a bottleneck.

Once we have completed the clustering, the attribute uniqueness AU_{att} of a given attribute is as follows:

$$AU_{att} = 1 - \left(\frac{|A| - 1}{\sum_{A' \in C} |A'|} \right) \quad (5)$$

AU_{att} always takes on a value in the range (0,1], with 0 indicating no attribute uniqueness, and 1 indicating the highest attribute uniqueness. A high AU_{att} value is achieved when attribute att appears infrequently across the tables of S_{att}, while a low value of AU_{att} occurs for an attribute that is commonly-occurring across the tables of S_{att}. An AU_{att} value of 1 indicates that an attribute is unique (in its own cluster by itself), while an AU_{att} value approaching 0 means that an attribute is one of many attributes in its own cluster. Note that an AU_{att} value for an attribute value att that has a value of 1 indicates that att has no other matching attribute in its cluster. As a result, att should not be involved in any match.

Recall that a single EBD value is between two attributes, and thus, to measure pairwise uniqueness, we need a measure that accounts for the AU_{att} value for both attributes in a pair. This measure is called pair uniqueness and designated as $PU_{att1,att2}$. It may be calculated by taking the arithmetic mean of the AU_{att} values for each attribute in a pair, the minimum AU_{att} value out of the pair, the maximum AU_{att} value out of the pair, and in a number of other ways. For our purposes, we achieved the most promising results when calculating $PU_{att1,att2}$ as the average of AU_{att1} and AU_{att2}. Like AU_{att}, the interval for $PU_{att1,att2}$ is (0,1], since it is based on AU_{att1} and AU_{att2}, both of which have values in the same interval.

4.2.3 Linear Time Clustering Heuristic

An alternative to using CP above which results in linear time clustering of attributes involves using the previously generated EBD values between all attributes in the compared data sources. Specifically, for each attribute, we keep track of all other attributes to which it is matched across all other tables other than its own. This means that we would need to perform attribute matching between tables within the same data source before performing the attribute clustering. Then, we would simply go through a list of all attributes, retrieve each attribute one by one (call this attribute a), retrieve its neighbors if they have not been retrieved before from any other attribute, and assign a and its neighbors to the same cluster. This ends up being a O(N) algorithm, because for a list of N attributes, there would be N checks to see if an attribute a has already been retrieved, with each check happening once per iteration. If a has not been retrieved, then its neighbors are retrieved. Attribute a and its neighbors are then inserted into a cluster; a's neighbors are retrieved only once. In the theoretical worst case, each attribute has a unique neighbor attribute, and each pair of attributes is inserted into its own cluster. For each iteration, there is a check on a to see if it is already part of a cluster. Also, exactly one operation per iteration for retrieving neighbors is executed; since each neighbor is referenced within the object for a, the act of retrieving a neighbor for a is done in time O(1). Thus, in this case there are N + N operations performed, which results in a O(2N) = O(N) algorithm.

4.3 Deriving Final Weighting

After calculating AU and PU, PU is then multiplied by the EBD_{orig} value produced by the pair to give a corrected value called EBD_{corr} :

$$EBD_{corr}(att1, att2) = EBD_{orig}(att1, att2) \times PU_{att1,att2} \quad (6)$$

Note that EBD_{corr} must be less than or equal to than EBD_{orig}, because $PU_{att1,att2}$ takes on a value between 0 and 1. The difference between EBD_{corr}(att1,att2) and EBD_{orig}(att1,att2), called pairwise semantic disparity ($PSD_{att1,att2}$), is then found between att1 and att2, and for all pairs of matching attribute pairs between two compared tables:

$$PSD_{att1, att2} = EBD_{orig(att1,att2)} - EBD_{corr(att1,att2)} \quad (7)$$

Next, the arithmetic mean of the PSD values, dubbed PSD_{avg}, amongst all of the attribute pairs for a table comparison is found. An attribute pair with a PSD value greater than PSD_{avg} indicates that a greater discrepancy exists between EBD_{orig} and EBD_{corr} relative to other attribute pairs. As a result, this pair should have the weight of its EBD_{orig} value reduced. In contrast, an attribute pair with a PSD value below PSD_{avg} indicates that relative to other pairs, its EBD discrepancy was less, and because of this, its attributes are more unique. Thus its EBD_{orig} value should contribute more substantially to semantic similarity between the tables. The new weight assigned to the attribute pair depends upon how far above or below the PSD value is relative to PSD_{avg}. For instance, an attribute pair that produces a PSD value that is .06 below PSD_{avg} is more unique than an attribute pair that produces a PSD value that is .03 below PSD_{avg}. Conversely, an attribute pair that produces a PSD value that is .06 above PSD_{avg} is less unique than an attribute pair that produces a PSD value that is .03 above PSD_{avg}.

Attribute weighting, as described above for a single table comparison, is illustrated in Algorithm 3 below, depicted as figure 4.3. Line 1 stores the attribute mappings that were

generated by GSim. Line 2 performs hierarchical agglomerative clustering described and assigns the derived set of clusters and their associated attributes taken from $M_{att(T),att(T')}$ to C. Lines 3-9 analyze each attribute mapping in $M_{att(T),att(T')}$ and ultimately calculate the PSD value between the attributes in the given mapping. Lines 4-5 calculate AU_{att1} and AU_{att2} for an attributes att1 and att2, respectively, and line 6 calculates the pairwise uniqueness between AU_{att1} and AU_{att2}. Lines 11-15 compare PSD_{avg} against the PSD value generated for a given attribute pair. If the PSD_{avg} is a higher value, then this means that the disparity in EBD values for this pair was less than the average, thus indicating that the pair is unique relative to other pairs.

Algorithm 3 attributeWeighting(T, T')

Input: Tables T and T', which are being semantically compared
Output: A weight vector $W_{att(T)-att(T')}$ containing normalized weights for each attribute pair among T and T'

1: $M_{att(T),att(T')}$ = getAttributeMappings(T, T')
2: C = performClustering($M_{att(T),att(T')}$)
3: For each attribute pair (att1(T), att2(T')) ε $M_{att(T),att(T')}$ {
4: $AU_{att1(T)}$ = calculateAU(att1(T), C)
5: $AU_{att2(T')}$ = calculateAU(att2(T'), C)
6: $PU_{att1(T),att2(T')} = (AU_{att1(T)} + AU_{att2(T')}) / 2$
7: $EBD_{corr}(att1(T),att2(T')) = EBD_{orig}(att1(T),att2(T')) \times PU_{att1(T),att2(T')}$
8: $PSD_{att1(T),att2(T')} = EBD_{corr}(att1(T),att2(T')) - EBD_{orig}(att1(T),att2(T'))$
9: }//end for
10: PSD_{avg} = computeAvg($M_{att(T),att(T')}$, $PSD_{att1(T),att2(T')}$)
11: For each attribute pair (att1(T), att2(T')) ε $M_{att(T),att(T')}$ {
12: if ($PSD_{att1(T),att2(T')} - PSD_{avg} > 0$)
13: $W_{att1(T),att2(T')}$ = reduceWeight(att1(T), att2(T'))
14: else
15: $W_{att1(T),att2(T')}$ = increaseWeight(att1(T), att2(T'))
16: }//end for
17: return $W_{att(T)-att(T')}$

Figure 4.3. Algorithm 3 – Attribute Weighting

This results in the pair's EBD value having a higher weight relative to other pairs in its table. On the other hand, if the PSD value generated between the attribute pair is higher, then the disparity of EBD values for this pair was more than average, indicating that the pair is not unique

relative to other pairs. This results in a deduction of weight for the pair's EBD value relative to other pairs in the table. Finally, line 17 returns the weights of all attribute pairs as a vector.

4.4 Experiments

We will now present the experiments involving GSim's application of attribute weighting. The datasets to which attribute weighting is applied are those from the GT and NGT matching experiments, namely the GIS Transportation Dataset, the GIS Location Dataset and the GIS POI Dataset.

4.4.1 Measurements and Parameters

To better illustrate the benefits of attribute weighting on matching tables, we preprocessed the attributes from tables of the GIS Transportation dataset and the GIS Location dataset to optimize GSim's ability to distinguish between commonly-occurring attributes and attributes that are more unique. The results of applying GSim's attribute weighting algorithm to the tables from the GIS Transportation dataset and the GIS Location dataset are shown below in figure 4.4 and 4.5, respectively. Figure 4.6 below illustrates EBD values produced between tables of the GIS POI dataset where all attribute mappings share equal weight while figure 4.7 illustrates the EBD values produced between these same tables where the attribute mappings now have attribute weighting applied to them. The table names along the vertical axis of the table belong to S_1, while the tables across the horizontal axis of the table belong to S_2.

	Road	Address Area	Enclosed Traffic Area	Ferry
Road	**.598**/.553	**.227**/.225	**.290**/.276	**.451**/.503
Residential Area	**.217**/.210	**.583**/.552	**.412**/.433	**.409**/.407
Traffic Area	**.151**/.136	**.206**/.209	**.962**/.958	**.218**/.235
Ferry	**.139**/.127	**.244**/.237	**.433**/.424	**.589**/.564

Figure 4.4. Two separate EBD values computed between a table from S_1 and a table from S_2 for the GIS Transportation Dataset. For each cell, the value right of the slash indicates the EBD value produced without attribute weighting, while the bolded value left of the slash is the EBD produced with the help of attribute weighting.

	Flight Schools	Schools	Ports	NavWaterways
Flight Schools	**.764**/.720	**.622**/.615	**.520**/.532	**.487**/.503
Schools	**.381**/.388	**.791**/.768	**.381**/.395	**.542**/.540
Indian Lands	**.473**/.489	**.506**/.513	**.488**/.486	**.522**/.533
Piers	**.490**/.496	**.469**/.489	**.639**/.633	**.626**/.616

Figure 4.5. Same as figure 4.4, but for GIS Location Dataset

	Streets2	Schools2	Hospitals2
Streets1	**.833**/.701	**.140**/.243	**.134** / .197
Schools1	**.173** / .201	**.842** / .735	**.236** / .269
Hospitals1	**.122** / .165	**.237** / .291	**.847** / .721

Figure 4.6. GIS POI dataset with latlong GT similarity applied, but no attribute weighting applied

	Streets2	Schools2	Hospitals2
Streets1	**.862**/.701	**.122**/.243	**.107**/ .197
Schools1	**.151** / .201	**.871** / .735	**.222** / .269
Hospitals1	**.111** / .165	**.216** / .291	**.869** / .721

Figure 4.7. GIS POI dataset with latlong GT similarity applied and attribute weighting applied

One last experimental parameter that should be mentioned is an attribute relevance parameter α that was applied to all attributes in tables from S_1 and S_2. Attribute relevance in GSim is executed as a preprocessing step that prevents any attribute that has a name or instance data which is not relevant to its containing table from taking part in a match with an attribute of another table. For instance, if table "Road" from S_1 is being compared with a table "Street" from S_2, then an attribute "Road.roadName", along with instance data containing road names, would be considered an attribute that is relevant to its containing table "Road". On the other hand, an attribute known as "Road.internalID", along with instances containing ID values of unknown significance, would likely not have any relevance to its containing table, "Road". The enforcement of attribute relevance is accomplished by taking the GD between the attribute name att1 and the name of the containing table T, added to the average GD between N instance values associated with att1 and the name of the containing table T. We set $\alpha = .90$ for all attribute weighting experiments.

4.4.2 Analysis of Results

The results of figure 4.4 and figure 4.5 show the effect that attribute weighting alone has on the EBD scores produced between tables in the GIS transportation dataset and GIS location dataset, respectively. The key observation in the results is that while attribute weighting consistently increases the EBD values between corresponding tables, it produces more arbitrary results among tables which do not naturally correspond. For these tables, latlong values in the data were not available, so the improvement in EBD is entirely the result of attribute weighting. In figure 4.4, the use of attribute weighting increased the EBD between pairs of corresponding tables (Road – Road, Residential Area – Address Area, Traffic Area – Enclosed Traffic Area, Ferry-Ferry) by 8.3%, 5.6%, 0.5% and 4.4%, respectively, when compared against GSim without latlong values and without attribute weighting. In figure 4.5, the use of attribute weighting increased the EBD between pairs of corresponding tables (Flight Schools – Flight Schools, Schools – Schools, Piers-Ports, Piers-NavWaterways) by 6.1%, 3.0%, 1.0% and 1.6%, respectively, when compared against GSim without latlong values and without attribute weighting. However, in both figures, EBD values neither consistently increased nor decreased when it came to pairs of tables that do not naturally correspond. The best results with attribute weighting were achieved in figure 4.7 with the POI dataset. Here, we also include figures 4.6 and 4.7 as a way to compare the improvement in EBD scores that resulted solely from the inclusion of latlong values (4.6) and the improvement garnered with the addition of attribute weighting (4.7). In figure 4.6, in each cell, the value in bold, to the left of the slash, indicates the EBD produced when taking into account latlong values only (without attribute weighting), while the value to the right of the slash indicates the EBD produced by GSim without latlong values or attribute weighting. In figure 4.7,

in each cell, the value to the left of the slash indicates the EBD score produced by GSim when using both latlong values and attribute weighting, while the value to the right of the slash, indicates the EBD produced when using neither latlong values nor attribute weighting. We see in figures 4.6 and 4.7 that the use of both latlong values and attribute weighting caused the EBD between corresponding tables to be strengthened more significantly and the EBD between dissimilar tables to be weakened consistently. The use of attribute weighting increase the EBD between pairs of corresponding tables (Streets1 – Streets2, Schools1 – Schools2, Hospitals1 - Hospitals2) by 22.9%, 18.5% and 20.5%, respectively, when compared against GSim without latlong values and without attribute weighting. Additionally, the combination of latlong values and attribute weighting was used to reduce the semantic similarity between dissimilar table pairs by an average of 19.1%. In analyzing the sole effects of attribute weighting, we can see that the EBD between Streets1-Streets increased by 3.5%, the EBD between Schools-Schools2 increased by 3.4%, and the EBD between Hosptals1-Hospitals2 increased by 2.6%. Furthermore, attribute weighting by itself also decreased the EBD values between non-corresponding tables in every case; the average reduction in EBD value due to attribute weighting for these tables was 11.7%.

Figure 4.8 below shows the progression of EBD scores when all of the approaches available in GSim are applied one at a time over tables of the POI dataset. For each cell (which represents a table comparison) there are four values. The value in the top row left of slash will be designated as (1), the value in the top row right of slash will be designated as (2), the value in the bottom row left of the slash will be designated as (3), and the value in the bottom row right of the slash will be designated as (4). The values are produced in the following ways: (1): GT matching + latlong + NGT matching + attribute weighting (2): GT matching + latlong + NGT matching (3):

GT matching + latlong (4): GT matching. It should be noted that typically, GSim only applies NGT matching if insufficient GT information exists within the data.

	Streets2	Schools2	Hospitals2
Streets1	.867/.862 .833/.701	.122/.230 .225/.243	.107/.133 .134/.197
Schools1	.151/.173 .173/.201	.872/.871 .842/.735	.222/.222 .236/.269
Hospitals1	.111/.124 .122/.165	.216/.216 .237/.291	.874/.869 .847/.721

Figure 4.8. EBD scores produced by GSim over the tables of the POI dataset, for naïve GT matching (bottom right value of cell), latlong GT matching (bottom left value of cell) latlong GT matching + NGT matching (top right value of cell), and latlong GT matching + NGT matchng

When it does, it is assumed that GT matching will not be applied, and that NGT matching is applied over all of the instances, including those that do possess GT information. However, for this experiment, we have adapted the NGT matching component of GSim such that it applies only to those instances without a GT. Doing this allows NGT matching to be applied directly on top of GT matching in a cumulative way.

This is similar, but slightly different from the approach employed in semi-supervised geosemantic clustering, in that the latter subjects all instances, whether they have a GT or not, to both a semantic similarity criterion via NGD and a geographic similarity criterion, such that the minimization of an objective function is achieved. Here, while instances with and without GTs may end up belonging to the same cluster, the instances with a GT are only clustered through their geography, while those without a GT are only clustered via NGD-based similarity with a cluster medoid.

As can be seen, taken over all cells, the largest average change in EBD occurs when latlong values are applied to disambiguate between multiple instances of the same name but different GTs. This accounts for 62.2% of the total EBD change from value (4) to value (1) in each cell. Another trend that can be observed is the general inefficacy of NGT matching when applied to instances in this dataset without GTs. In some cases, such as Streets1-Hospitals2 and Hospitals1-Streets2, NGT matching causes these incompatible table matches to slightly increase their EBD score. We believe that this occurs for two reasons. First, nearly all instances (about 98.3%) in the POI dataset have a GT identifiable by a gazetteer. Out of the three datasets we have experimented on with GSim, the POI dataset is the only one that contains latlong values associated with its instances. The fact that nearly all instances in the POI dataset having GTs and latlong values guarantees that NGT matching cannot make much of a contribution to the final similarity score. Second, in the cases where NGT matching slightly increases the EBD score between incompatible tables, this occurs because of the tendency of NGT matching to group together instances with more semantic disparity between them than GT matching would allow. NGT matching is based on co-occurrence embedded in the formula for GD. As a result, as long as two instances co-occur on a web page, regardless of their actual types, then they will be grouped together as part of the same generic type. Attribute weighting is responsible for 23.7% of the average change in EBD from value (4) to value (1) over all cells.

CHAPTER 5

GEOSEMANTIC CLUSTERING WITH GEOSIM

GSim addressed many challenges of geospatial schema matching that enable an accurate determination of semantic similarity between attribute pairs and table pairs. However, despite the variety of geospatial schema matching approaches explained above for GSim and the large variety of research on clustering detailed in the literature(Zhuang 2010;Li 2008, 1519-34;Zhao 2006;Bauckhage 2010;Dang 2010;Ping 2009, 1249-62;Wu 2010) several more challenges need to be addressed. As alluded to in section 1.6, in this chapter, we seek to address three additional challenges in performing geospatial schema matching: (1: Using both semantic and geographic instance properties in assigning instances to clusters (2: Reducing the variance in semantic similarity scores (EBD scores) over multiple trial runs of the same 1:1 attribute pair (3: Accounting for hierarchical relationships that might exist. We will discuss each one individually.

Up to now, when assigning an instance to a cluster for a given 1:1 attribute match, only one property of that instance was utilized. For instance, in GSim(Partyka 2010, 58-59) during GT matching, the GT of the instance determines whether it is assigned to a “River” cluster or to an “Airport” cluster. Other applications might use a geographic property other than GT(Ahlqvist 2006). In NGT matching the GD between an instance to be assigned and a cluster medoid determined instance membership in that cluster. Again, other applications may use a different semantic property(Partyka 2009a;Cruz 2009a). However, as mentioned previously the idea of using both geographic and semantic properties for a geosemantic similarity algorithm has not

been explored in detail to our knowledge. Using both properties can lead to better clustering.

Second, the semantic similarity score computed between compared attributes in most schema matching applications using clustering ends up varying from one trial run to the next, even for the same attribute comparison. In the case of GSim(Partyka 2009), it uses clustering over the instances of a pair of compared attributes. This approach can lead to a large variance in the calculated similarity score due to the inherent variability in the clustering process. Although this often results from the randomness of choosing initial centroids (or medoids), we will explore the cause of inconsistent EBD scores being from the varying semantic properties of instance samples from one clustering trial to the next for a given attribute comparison. Figure 5.1 below illustrates an example of this over two distinct clustering trials for the same attribute comparison.

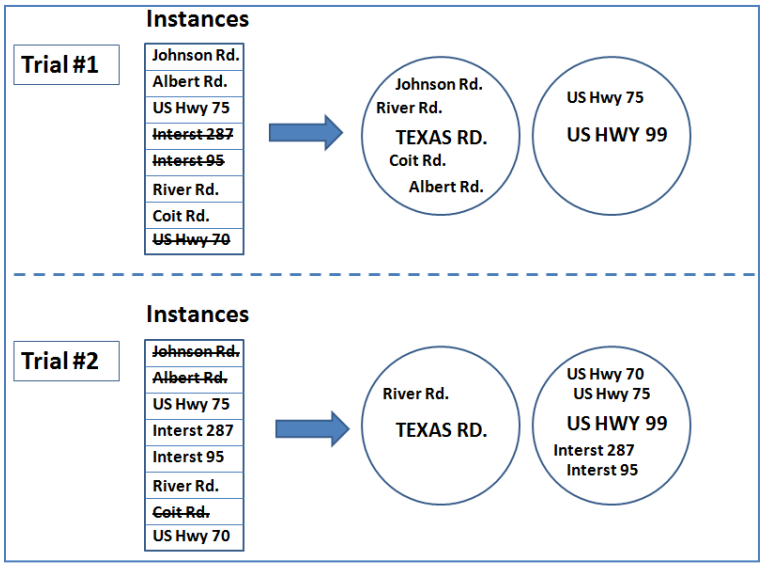


Figure 5.1. Clustering instances over two separate trials with inconsistent results

Here, instances from both attributes are considered for comparison, but only a sample of the instances will actually be clustered. We assume that instances are annotated by keywords, although this is not shown. So for example, the instance "Johnson Rd." might be annotated with

the keywords, “street, Plano, Texas, Collin-County”. A given instance is assigned to a cluster based on the highest number of overlapping keywords between it and the cluster medoid (indicated in CAPS). Two clusters are shown: a cluster of road instances and a cluster of highway instances. Note that the instances selected for clustering differ between the two trials, with the instances not selected for clustering indicated with a horizontal line through its text (strikethrough text). However, because keyword overlap is used as the semantic property by which instances are assigned to clusters, the clusters take on completely different instances between the trials. This will result in very different semantic similarity scores for the same attribute pair. To overcome this, we need to choose instance properties that result in more consistent clusterings. In turn, this will produce more consistent similarity scores.

The third challenge is accounting for the presence of hierarchical relationships in the data. While we have encountered some work examining formally analyzing hierarchical structures(Punera 2008), we know of none to our knowledge that have applied geospatial schema matching over geographic types. As mentioned previously, if the compared attributes are associated with GTs structured as a hierarchy (e.g. SWEET ontology in the earth sciences domain), then the semantic similarity score will be affected by the relationship between the attributes’ GTs as indicated by the hierarchy. Figure 5.2 depicts a situation involving compared attributes associated with the GTs ‘Rivers’ and ‘Springs’ (in an ontology, the GTs would be referred to as concepts). Both concepts are modeled after the hydrographic features portion of the GT hierarchy from the ADL gazetteer (Alexandria Digital Library Project). In figure 5.2, an attribute ‘River.name’ from a ‘River’ table is compared against ‘Rapid.Name’ from a ‘Rapid’ table. If the attributes are compared based on whether the GTs of the instances of ‘River.name’

match exactly the GTs of the instances of ‘Rapid.Name’, then no similarity between them exists since strictly speaking, a rapid is not exactly the same as a river.

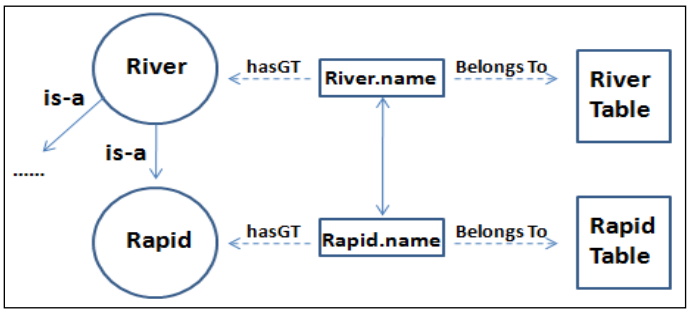


Figure 5.2. Comparing attributes using hierarchical relationships

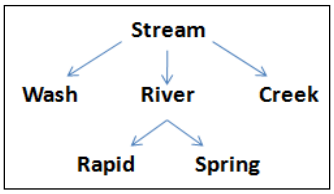


Figure 5.3. Part of ADL’s GT hierarchy for hydrographic features

However, when looking at the relationship between the ‘River’ concept (GT) and ‘Rapid’ concept (GT) in Figure 5.3, we can see that ‘Rapid’ is a direct subclass of ‘River’. Therefore, the hierarchy indicates that a ‘Rapid’ is a kind of ‘River’, and we would assign a higher similarity score between the compared attributes. However, taking into account hierarchical relationships appropriately requires a nuanced approach, as two kinds of mistakes can be made. The first is being overly specific in specifying GTs; this was illustrated above when the match between ‘Rapid’ and ‘River’ required an exact GT match. This would lower the similarity score too much.

The second is being overly general in specifying GTs. In figure 5.3, when comparing attributes ‘Stream.name’ and ‘River.name’, if all instances of the ‘Stream’ concept were considered equivalent to instances of the ‘River’ concept, then the similarity score would be 1.0,

which is too high. The correct approach would recognize a hierarchical relationship while avoiding overspecificity or overgeneralization. Many geospatial schema matching programs, including GSim, do not account for hierarchical relationships in the data.

5.1 GeoSim Overview

To address the above challenges, we introduce GeoSim, a geospatial schema matching tool that derives semantic similarity between heterogeneous schemas. We focus on the comparison between tables and their attributes across the schemas. GeoSim consists of $GeoSim_G$, a semi-supervised algorithm producing consistent, geosemantic clusters of instances, and $GeoSim_H$, responsible for hierarchical matching. Like our prior work GSim, GeoSim derives 1:1 matches between attribute pairs of the compared tables by using attribute name and data type matching to select the attribute pairs. For each attribute pair, their instances are assigned to clusters. Traditional clustering algorithms such as GSim rely on only one instance property, leading to inconsistent clustering and lower, more inconsistent similarity scores. To ensure higher, more consistent similarity scores, GeoSim employs $GeoSim_G$, which creates clusters using two properties associated with each instance. The first is a semantic distance measure based on Google Distance (GD) that computes the "meaning distance" between two distinct instances. The use of GD overcomes problems caused by instance annotations from figure 5.1. The second is a GT that associates an instance with a geographic feature, such as a lake or a road. We use these properties to ensure consistent clusters of instances from a given attribute pair. GTs act as labels on instances, making $GeoSim_G$ semi-supervised instead of unsupervised like GSim. The semantic similarity score between the attributes is then calculated based on the distribution of their

instances over all clusters. The similarity is based on an information-theoretic measure known as entropy-based distribution (EBD). Combining the scores from all attribute pairs between the two tables determines 1:1 table similarity. To account for hierarchical relationships between the compared attributes, GeoSim employs $GeoSim_H$. $GeoSim_H$ executes hierarchical matching that uses the GTs of the compared attribute instances to calculate the path length between these GTs in a geographic hierarchy like an ontology. Both measurements are then incorporated by GeoSim into the final EBD calculation as a weighted linear combination.

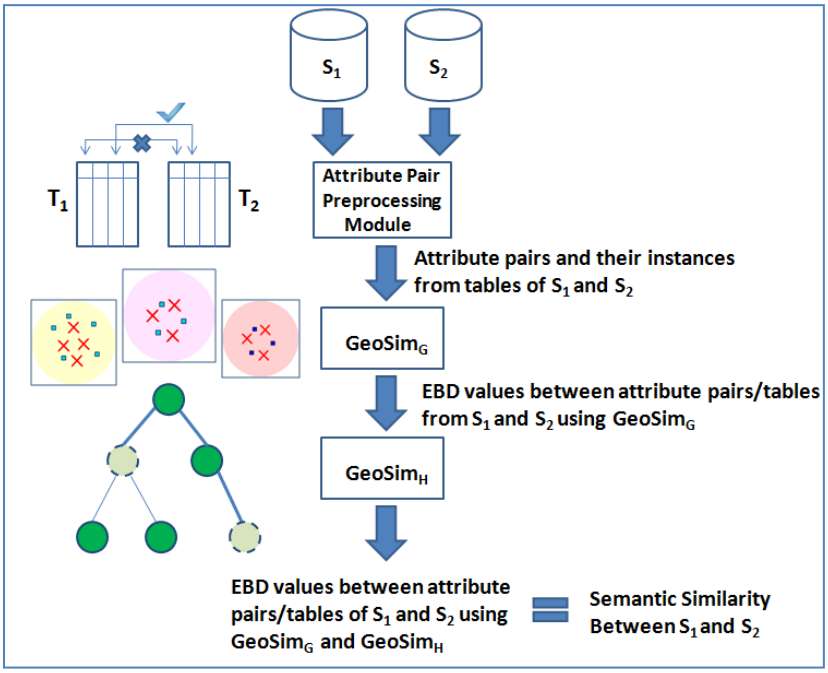


Figure 5.4. Flow of control in GeoSim

Figure 5.4 depicts GeoSim's typical flow of control. First, just as in GSim, the set of 1-1 attribute pairs is determined. From these attribute pairs, GeoSim evaluates the semantic similarity between each pair by examining the available instances. The first step in this process is to apply $GeoSim_G$, the clustering component of GeoSim, which functions by employing K-medoid clustering, GD and GT entropy. The end result of this is a similarity (EBD) value based on the

semantic and geographic properties of the instances associated with the compared attributes. Next, GeoSim leverages $GeoSim_H$, its hierarchical matching component, as well as the EBD scores computed for all 1-1 attribute pairs by $GeoSim_G$ to perform hierarchical matching. The final result of this is an EBD value between all attribute pairs that is based on geosemantic instance clustering and hierarchical matching.

5.2 Details of $GeoSim_G$ and SSGS Algorithm

In $GeoSim_G$, semantic similarity is based on semi-supervised geosemantic (SSGS) clustering. The idea is to produce consistent clusters for a given attribute pair using the semantic and geographic properties of instances. The semantic property is the GD between a pair of instances. The geographic property is a GT representing the geographic feature associated with the instance. The clustering is semi-supervised because some of the instances are labeled with GTs. Using these properties, we model a semi-supervised, consistent clustering as the minimization of an objective function over all instances.

The SSGS algorithm begins by determining the number of clusters K using instances between the compared attributes. We refer to the set of instances over both attributes as I. Second, one instance from I is assigned to each cluster as a medoid. Third, we assign all remaining instances in I to the most closely related cluster. This is done by minimizing an objective function for each instance. In unsupervised K-medoid clustering, the instances have no labels. The objective function to be minimized here is based on the distance between an instance and its corresponding cluster medoid. This is formally illustrated as follows. Given K clusters $\{C_1$,

$C_2 \ldots C_k\}$ and a set of instances $I_n = \{i_1, i_2, \ldots i_n)$ to be assigned to the K clusters, the objective function to be minimized is:

$$O_{Kmedoid} = \sum_{i=1}^{K} \sum_{x \epsilon Ci} ||x - u_i|| \quad (8)$$

Here u_i is the medoid for cluster C and $||x - u_i||^2$ is the distance between instance x and u_i. This is the semantic distance between an instance and a medoid, or GD. Taken over all instances in a cluster, this value is known as the semantic purity of the cluster, or Imp_S.

We also take into account the GT when assigning each instance to its appropriate cluster. We want each cluster to (1: contain instances which are semantically related to one another and (2: be as geographically homogenous as possible. The minimization of the objective function consists of both the semantic purity measure, Imp_S, and the geographic purity measure, Imp_G. For a cluster C_i, Imp_G is expressed as:

$$\mathbf{Imp_G(C_i) =} \sum_{t=0}^{T} p_t^i \log(p_t^i) \textbf{ where } p_t^i = \frac{|Ci(t)|}{|Ci|} \quad (9)$$

where p_t^i is the prior probability of GT t. In equation 9, $|C_i(t)|$ indicates the number of instances in cluster C_i that have a GT of t. While T can represent any number of GTs that are represented in a cluster C_i, in our experiments, we set T = 2. This is because for determining geographic purity of a cluster, we only care about the number of instances with a GT matching the GT of the medoid. As an example, for a cluster with a medoid having a GT of "road", if 10 of the instances in the cluster have a GT of "road", while 5 of the instances have a GT of "lake" and 5 instances have a GT of "grassland", then we consider all of the instances to fit into two possible classes: those with a GT = "road", and those with a GT ≠ "road". The number of

actual GTs represented within the cluster is irrelevant. Because T = 2, this means that the maximum value of Imp_G (C_i) is 1. This gives Imp_G a range of [0,1]. Based on the work of (Masud 2008), the objective function used by $GeoSim_G$ to evaluate the clustering produced by the SSGS algorithm once all instances are assigned is:

$$O_{SSGS} = \sum_{i=1}^{K} Imp_S + \sum_{i=1}^{K} W_i \, Imp_G \quad (10)$$

Imp_S is the semantic purity of the cluster = $\sum_{x \epsilon Ci} ||x - u_i||^2$, Imp_G is the geographic purity of the cluster and W_i is a weighting factor for cluster i. In $GeoSim_G$, clusters with more instances are given more weight, since they are more invested in the EBD calculation. The weight assigned to a given cluster is $W_i = \sum_{x \epsilon Ci} ||x - u_i||^2$. This says that the weight assigned to a cluster is based on the semantic purity of its instances.

Once we have assigned all instances from I to clusters, the SSGS algorithm determines if any cluster medoids need to be recomputed. To do this, we examine a given instance in a given cluster C_i and compute Imp_S. After doing this for all instances, if the current medoid produces the lowest Imp_S value, then $GeoSim_G$ checks Imp_G for C_i. If $Imp_G < \lambda$, where λ is a geographic purity threshold, and the GTs shared by the largest class of instances in the cluster is the same as the GT of the medoid, then C_i has an acceptable clustering. In our experiments, we set $\lambda = .05$. If we achieve this over all clusters, then we have minimized O_{SSGS}. Otherwise, another iteration of SSGS needs to be executed. This process continues until all clusters collectively minimize O_{SSGS} and individually satisfy λ.

The time complexity of the SSGS algorithm is $O(K|C_i|^2)$, where K is the number of clusters and $|C_i|$ is the number of instances in cluster C_i. The most expensive step is determining if

a cluster is optimal. Computing Imp_G takes O(1) time, since we keep a vector of Imp_G values for each cluster that is updated when a new instance is added.

5.3 Details of $GeoSim_H$ and Hierarchical GT Matching

Until now, we have ignored whether the GTs of the instances from each of the compared attributes are related to each other. If the instances do not share any GTs, then this is not a concern. However, it must be considered for a comparison of att1-att2, where att1 belongs to a table 'Lake' and att2 belongs to a table 'Pond'. Although the GTs do not strictly match, a lake is closely related semantically to a pond. Therefore, the EBD score between the attributes should reflect this relationship.

To account for this, we introduce a term into the equation for calculating EBD that represents the structural similarity between the GTs of the instances making up the compared attributes. The equation is:

$$\mathbf{EBD_{rel}} = (\mathbf{W_{ebd}} * \mathbf{EBD}) + (\mathbf{W_{struct}} * \mathbf{Sim_{struct}}) \quad (11)$$

Here, EBD is the value computed by $GeoSim_G$, W_{ebd} is a weighting factor for the EBD value, Sim_{struct} is the structural similarity between the GTs of the compared attributes, and W_{struct} is its weighting factor. The weighting factors when summed together will equal 1.0.

Each attribute involved in a comparison is associated with >= 1 GTs, based on the GTs of its instances. If the GTs are represented in a hierarchy like an ontology (with each GT representing a concept), then Sim_{struct} between two attributes can be calculated by measuring the path length from one GT to another over all distinct GT pairings between the instances of the

compared attributes. The average path length over all distinct GT pairings is taken to be the final value of Sim_{struct}.

Though there are many ways to measure path length, we used a normalized, geospatial version of the Leacock-Chodorow method (LDC)(Jin 2005) called LDC_G. LDC takes into account the WordNet-based path length between two concepts, as well as the depth of the WordNet hierarchy from the root node to the most distant leaf node, to calculate structural similarity. In order to adapt the LDC measure for the geospatial domain, LDC_G calculates the depth of the geospatial ontology that models the GTs of the attribute instances. Other than this, the formula for LDC_G is calculated just like LDC.

Popular alternatives to LDC_G are Lin Similarity and the Banerjee and Pedersen Measure (BNP)(Jin 2005). Both measures are used in semantic similarity calculations in many domains. For the geospatial domain, we claim that LDC_G is more suitable than Lin and BNP, because they both rely on a WordNet based measure called Information Content (IC). IC is calculated based on word frequency in the British National Corpus (BNC). However, the coverage of geographic feature names in the BNC is poor, so the IC values used for Lin and BNP will not be accurate. Since LDC_G relies entirely on the structure of geospatial ontologies, it avoids these problems.

5.4 Experiments

We present three geospatial matching experiments involving GeoSim, our geosemantic clustering algorithm that applies the SSGS algorithm to create a high-quality clustering by minimizing an objective function. The first of these experiments compares the semantic similarity results of $GeoSim_G$ against four popular methods used in the data mining community over two

separate geospatial datasets. They are: (1: N-grams (2: nonnegative matrix factorization (NMF) (3: singular value decomposition (SVD) (4: GSim, which here only uses geographic clustering. The second experiment involving GeoSim measures the precision and recall variance of $GeoSim_G$ and compares it to the precision and recall variance calculated for the four data mining methods from the previous experiment. The third experiment measures the performance of $GeoSim_H$.

5.4.1 $GeoSim_G$ Dataset Details

The datasets used for the experiments involving $GeoSim_G$ are the GIS Transportation Dataset (GTD), the GIS Location Dataset (GLD), and the GIS POI dataset (GPD). These datasets were also used for the experiments in Chapters 3 and 4. In regards to the ground truths for these datasets, GTD has 29 correct attribute matches across all table comparisons, while the ground truth for GLD has 52 correct attribute matches. It should be noted that valid attribute matches exist between tables that do not semantically correspond.

5.4.2 $GeoSim_G$ Results

In the first experiment, we measured the effectiveness of $GeoSim_G$ versus N-grams, NMF, SVD and GSim for determining the semantic similarity of compared tables over different data sources. For all experiments involving N-grams, N=2. The results of this experiment are displayed in figure 5.5. In the figure, P = precision, R = recall and F = F-measure. The effectiveness of each similarity measure with respect to a particular data source was quantified using F-measure, since it takes into account both precision and recall. For GTD, the F-measure generated by $GeoSim_G$ outperforms N-grams .83-.44, SVD .83-.13, NMF .83-.25 and

$GeoSim_G$ .83-.71. For GLD, $GeoSim_G$ outperformed N-grams .79-.09, SVD .79-.17, NMF .79-.22, and GSim .79-68.

Datasets	N-grams			SVD			NMF			GSim			$GeoSim_G$		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
GTD	.38	.52	.44	.08	.29	.13	.17	.50	.25	.70	.72	.71	**.84**	**.81**	**.83**
GLD	.80	.06	.09	.20	.15	.17	.26	.19	.22	.80	.61	.68	**.85**	**.74**	**.79**

Figure 5.5. Effectiveness of $GeoSim_G$ vs. other methods for GTD and GLD datasets

The use of both semantic and geographic clustering criteria improved $GeoSim_G$'s F-measure over GSim in GTD and GLD by 17.0% and 16.2%, respectively.

In the second experiment, the clustering consistency of N-grams, SVD, NMF, GSim, and $GeoSim_G$ were determined by measuring the variance in their generated precision and recall scores over the GTD and GLD data sets. The variances were generated after 50 trial runs for each attribute comparison. Figure 5.6 shows the results. As can be seen, the variances for the precision and recall of $GeoSim_G$ in both datasets are less than those produced by the other methods. For precision in GTD, $GeoSim_G$'s variance is 60% less than N-grams (.10-.25), 33% less than SVD(.10-.15), 47% less than NMF(.10-.19), and 47% less than GSim(.10-.19). For precision in GLD, $GeoSim_G$'s variance is 82% less than N-grams (.08-.44), 47% less than SVD(.08-.17), 71% less than NMF(.08-.28), and 68% less than GSim(.08-.25). For recall in GTD, $GeoSim_G$'s variance is 84% less than N-grams (.06-.37), 78% less than SVD(.06-.27), 82% less than NMF(.06-.33), and 33% less than GSim(.06-.09). For recall in GLD, $GeoSim_G$'s variance is 33% less than N-grams (.04-.06), 80% less than SVD(.04-.20), 82% less than NMF(.04-.22), and 64% less than GSim(.04-.11).

Datasets	N-grams		SVD		NMF		GSim		$GeoSim_G$	
	P	R	P	R	P	R	P	R	P	R
GTD	.25	.37	.15	.27	.19	.33	.19	.09	**.10**	**.06**
GLD	.44	.06	.17	.20	.28	.22	.25	.11	**.08**	**.04**

Figure 5.6. Variance of $GeoSim_G$ vs. other methods for GTD and GLD datasets

5.4.3 $GeoSim_H$ Dataset Details

For the second experiment, we tested $GeoSim_H$ by matching table attributes over two data sources gleaned from the GNIS(USGS US Board of Geographic Names). The first data source, the POI dataset, consists of points of interests taken from US geographic areas. The second data source, the HYDRO dataset, focuses on natural and manmade hydrographic features in the US. The number of instances per table in each data source varies widely. In POI, the largest is Locale(49735) and the smallest is Trading_Post (69). In HYDRO the largest is Hydrographic_Structure (154532) and the smallest is Slough (131). Furthermore, using the ADL gazetteer and inherent relationships within the two data sources, we devised a GT ontology for both datasets. The GT ontology for POI is shown in Figure 5.7, and the GT ontology for HYDRO is shown in Figure 5.8. The ground truth for POI has 32 correct attribute mappings, while the ground truth for HYDRO has 27 correct attribute mappings.

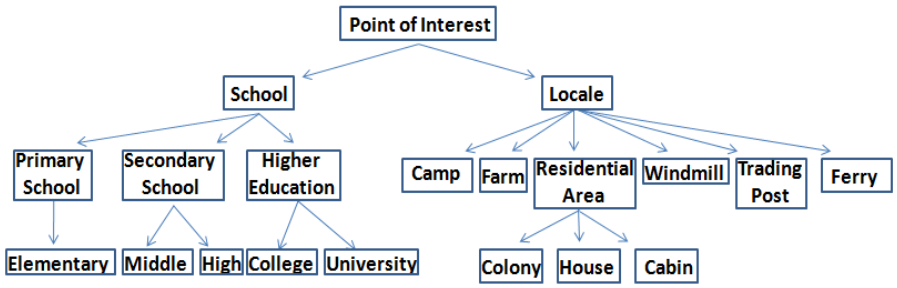


Figure 5.7. POI Ontology

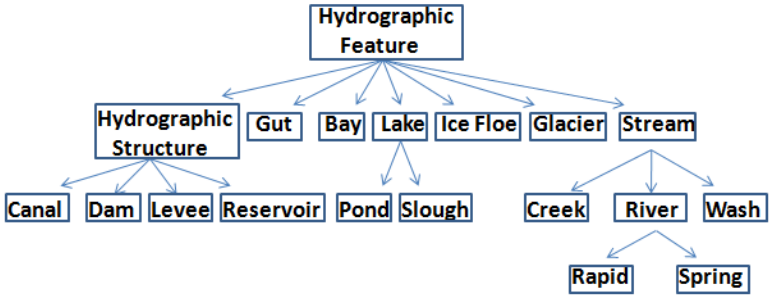


Figure 5.8. HYDRO Ontology

5.4.4 $GeoSim_H$ Results

First, we measured the improvement in semantic similarity by incorporating LDC_G into EBD. Here, $W_{ebd} = W_{struct} = .5$. Figure 5.9 below depicts the precision, recall and F-measure generated by $GeoSim_G$ alone(EBD) and $GeoSim_G + GeoSim_H$ (EBD + LDC_G, or EBD_{rel}) over POI and HYDRO. While the precision values remain largely unchanged, incorporating LDC_G into the semantic similarity calculation clearly increases recall. Thus, F-measure increased over POI and HYDRO. For the POI dataset, adding LDC_G increased recall by 38.7% (.406 to.563) and F-measure by 24.6% (.578-.720). For the HYDRO dataset, LDC_G increased recall by 274.7% (.068-.186) and F-measure by 243.9% (.127-.310).

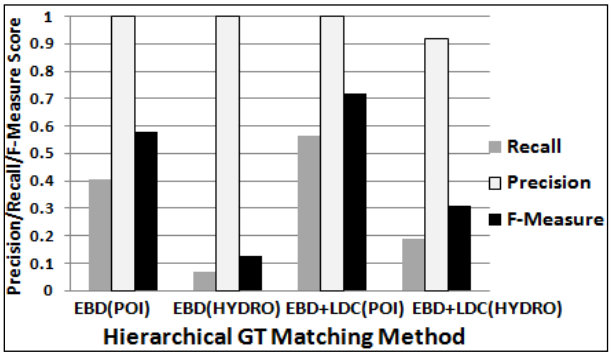


Figure 5.9. Precision, Recall and F-measure generated by EBD and EBD+LDC_G over the POI and HYDRO data sources

We also compared LDC_G to Lin similarity. Lin is a WordNet-based measure computing a ratio between the information content (IC) of the least common subsumer of the compared words and the IC values of the compared words.

While Lin is suitable generically, our experiments show that LDC_G is more suitable in the geospatial domain. Figure 5.10 shows the F-measure computed by EBD+LDC_G and EBD+Lin over the POI dataset. W_{ebd} took on values from 0 to .50. The remaining weight was assigned to either LDC_G or Lin. For each value of W_{ebd}, LDC_G outperforms Lin. The average F-measure of EBD+LDC_G was .825, while for EBD+Lin it was .522; LDC_G improved F-measure by 58%. Figure 5.11 shows F-measure computed by EBD+LDC_G and EBD+Lin over the HYDRO dataset. The average F-measure of EBD+LDC_G was .318, while for EBD+Lin, it was .191. EBD+LDC_G increased F-measure by 66%.

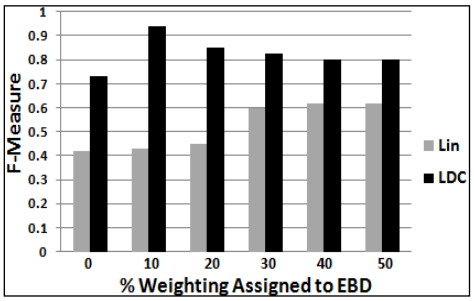


Figure 5.10. F-measure scores generated by EBD+LDC_G and EBD+Lin over 5 different values of W_{ebd} in the POI dataset data sources

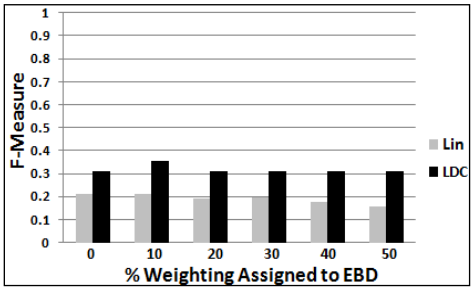


Figure 5.11. Same as figure 5.10, except applied to the HYDRO dataset

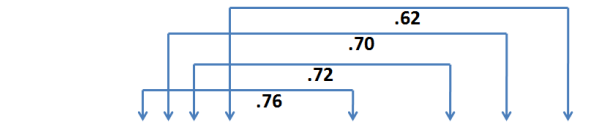
CHAPTER 6

MULTI-ATTRIBUTE (1:N) MATCHING

Up to this point, all of the matches that have been studies have been 1:1 matches, mostly between attribute pairs. The focus of this chapter will be 1:N matches, where 1 attribute of a table matches with N attributes of another table, with N > 1. In particular, great attention will be given to the correctness of our matching algorithm, and how it defines an optimal match. At the end of this chapter, we present an experiment to demonstrate the effectiveness of our algorithm.

6.1 Overview

Figure 6.1 and figure 6.2 illustrate 1:1 and 1:N matching, respectively, towards two compared tables in different schemas. Figure 6.1 depicts 1:1 attribute matching, while figure 6.2 depicts 1:N attribute matching.



val	Address
1	12 Plano Dr., Plano, TX, 75075
2	18 Coit Rd., Richardson, TX, 75080
3	200 Preston Rd., Dallas, TX
4	2 Hedgecoxe Rd., Frisco, TX

T_1

id	Street Address	City	State	Zip
1	100 Genstar Dr.	Dallas	TX	75252
2	2091 Spring Creek Rd.	Plano	TX	75075
3	1704 Danube Ln.	Plano	TX	75075

T_2

Figure 6.1. An instance of 1:1 matching between two tables

As can be seen, only 1:N matching can capture the match between Address(T_1) and the collection of attributes represented by the set {Street Address(T_2), City(T_2), State(T_2) and

Zip(T_2)}. This is because each instance within Address(T_1) contains subinstances, where each one has a type equal to the type for an instance found in either Street Address(T_2), City(T_2), State(T_2) or Zip(T_2).

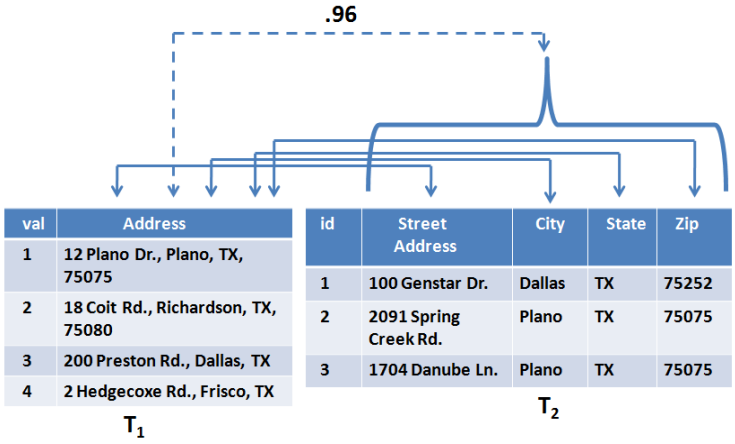


Figure 6.2. An instance of 1:N matching between the same two tables. Here, N attributes of T_2 match with a single attribute of T_1

Additionally, a score for an attribute match in both figures is listed along the arrow-headed line connecting the attributes. For our purposes here, let these numbers indicate the degree of semantic similarity, where 0 indicates no match at all and 1 indicate identical attributes. In figure 6.1, the 1:1 matches with Address(T_1) all possess similarity values in the range of [.62, .76], which we would consider fairly strong. In figure 6.2, however, notice that the 1:N match between Address(T_1) and {Street Address(T_2), City(T_2), State(T_2) and Zip(T_2)} is .96, which we would consider very strong. This demonstrates the ability of 1:N matching to identify strong matches for a given set of attributes that 1:1 matching cannot identify for the same attributes.

GSim (and not GeoSim) features a greedy 1:N matching algorithm that discovers relationships between a single composite attribute of one table and up to N attributes of another table, where N >= 2. 1:N matching scenarios arise often in various geospatial applications. For

instance, 1:N matching could improve land cover classification. If an area of land comprised of several different land types (ie: evergreen forest, wetlands, etc.) is separated by a spatial lag from another land area of interest, then 1:N matching could be used to more effectively calculate the similarity between those land areas.

We have added to GSim a greedy 1:N matching algorithm that can discover attribute correspondences like the one depicted in figure 6.2. It will be proved over the next few sections that our algorithm is correct, and what it means for our algorithm to generate an optimal EBD score.

6.2 Definitions and Assumptions

Before describing our 1:N matching algorithm and its proofs of correctness and optimality, we will specify some background definitions and properties. We assume that 1:1 matching between the tables of the compared data sources has been executed such that (1: all 1:1 attribute matches and their associated EBD scores have been derived (2: Every instance in an attribute involved in a 1:1 match is associated with >= 1 GTs.

Definition 10 (attribute of table) *An attribute of a table T, denoted as T(att), is a property of a table that describes it. It is made up of instances.*

Definition 11 (GT set of attribute) *The GT set of attribute T(att), denoted as GT(T(att)), is the set of GTs over all instances of T(att)*

Definition 12 (composite attribute) *A composite attribute of a table T, denoted as Cmp, is an attribute where |GT(T(Cmp))| >= 2.*

In figures 6.1 and 6.2, T_1(Address) is a composite attribute, because over all instances, GT(T_1(Address)) = {Street Address, City, State, Zip}. For simplicity purposes, we will assume from now on that the composite attribute Cmp in a 1:N match is located in table T_1 and that the N attributes that match Cmp are located in table T_2.

Definition 13 (EBD (T_1(att1), T_2(att2))) *A value denoted EBD (T_1(att1), T_2(att2)) between T_1(att1) and T_2(att2) is a real number in the range [0,1] indicating the 1:1 EBD value between T_1(att1) and T_2(att2).*

Definition 14 (C(T_2)) *The candidate attribute set, denoted by C(T_2), represents the set of attributes from T_2, ordered by EBD(Cmp, T_2(att)) from greatest to least, such that the following three conditions are satisfied:*

Condition 1: *EBD(Cmp, T_2(att)) >= minT for any att ϵ T_2. The constant minT is a match threshold value indicating the minimum value required for two attributes to be considered a match. We set minT = .6 in our experiments.*

Condition 2: *GT(T_2(att)) $\cap$ GT(Cmp) != Φ for any att ϵ T_2*

Condition 3: *No attribute in C(T_2) has been considered for inclusion into a 1:N match with Cmp.*

Condition 2 results from a proof below that demonstrates the correctness of the optimality of 1:N matching through the minimization of the GT sets over the instances of the participating

attributes. The formal description of condition 3 above will be stated below after Definition 15. Additional notation for definition 14 is as follows. To indicate that j attributes have been removed from $C(T_2)$, where $0 <= j < |C(T_2)|$, we use $C(T_2)^j$. From now on, $C(T_2) = C(T_2)^0$. From this, we derive a 1:N match property related to $C(T_2)$:

Property 1: $|C(T_2)| <= N$, where N is the number of attributes in T_2

Definition 15 ($F(T_2)$) The final attribute set, denoted by $F(T_2)$, represents the set of attributes from T_2, such that (1: They are in $C(T_2)$ at the time 1:N matching begins (2: The EBD between Cmp and all attributes in $F(T_2)$ when $|F(T_2)| = k$, $0 < k <= |C(T_2)^0|$ is greater than the EBD between Cmp and all attributes in $F(T_2)$ when $|F(T_2)| = k - 1$.

Definition 16 ($F_{EBD}(T_2)$) $F_{EBD}(T_2)$ indicates the total EBD between the attributes in $F(T_2)$ and Cmp.

The second condition of Definition 15 above says that the addition of an attribute from table T_2 to a 1:N match must raise the overall EBD between the Cmp and the matched attributes ϵ T_2. Thus, the EBD strictly increases for any given 1:N match. This shows that our conception of 1:N matching is based on optimizing the overall EBD score between the attributes, as opposed to optimizing the number of attributes involved in the match. Additional notation for $F(T_2)$ is as follows. $F(T_2)^k$ indicates that there are k attributes in $F(T_2)$, $F(T_2)^0$ indicates that $F(T_2)$ is empty, and $F(T_2)^C$ indicates that this is the complete set of attributes from T_2 involved in a 1:N match with Cmp. Definition 16 is not an average of the values of EBD(Cmp, $F(T_2)$), but rather, their

composite EBD. Using definition 16 and the notation in the second part of definition 15, the second condition of definition 15 can be restated as, "$F_{EBD}(T_2)^k > F_{EBD}(T_2)^{k-1}$, where $0 < k < |C(T_2)|$" Additional notation for definition 16 is as follows. $F_{EBD}(T_2)^k$ indicates the EBD value between the k attributes in $F(T_2)$ and Cmp, $F_{EBD}(T_2)^0 = 0$ and $F_{EBD}(T_2)^C$ is the EBD value between the attributes of $F(T_2)^C$ and Cmp.

Definition 17 (Cmax) Cmax indicates the attribute att from $C(T_2)$ with the highest value of EBD(Cmp, T(att)) that is to be included next into $F(T_2)^k$, where $0 <= k <= |C(T_2)|$

We will also the term $Cmax_x$, to indicate the set of x attributes selected from $C(T_2)$ where each one was Cmax relative to the other attributes in $C(T_2)$ upon its selection.

We are nearly ready to prove that our 1:N matching algorithm produces an optimal EBD score between the participating attributes by minimizing $GT(Cmp) - GT(F(T_2)^C)$. First, we need to state two additional properties of 1:N matching used by our algorithm.

Property 2: For all attributes in $F(T_2)^C$, the number of instances for each GT in $GT(F(T_2)^C)$ cannot exceed the number of instances of those same GTs in Cmp.

An example of property 2 would be as follows. Suppose GT(Cmp) = {River, Pond, Rapids} and $GT(F(T_2)^C)$ = {River, Rapids}. Further, let's suppose that the number of instances for each GT River in Cmp is as follows: River = 50, Pond = 20, Rapids = 60. Then the number of instances for the attributes in $F(T_2)^C$ for each GT must be as follows: River: <= 50, Pond: <=20, Rapids: <= 60. Two important consequences of property 2 are stated as properties 3 and 4:

Property 3: The number of instances in Cmp >= the number of instances in $F(T_2)^C$.

Property 4: If the number of instances for a GT in Cmp is 0, then over the attributes in $F(T_2)^C$, the number of instances with a GT not in GT(Cmp) must be 0.

One last property to mention is the following:

Property 5: For any attribute A in $F(T_2)^C$, $|GT(A)| = 1$.

Using the above definitions and properties, we will prove that our 1:N matching algorithm is correct, and that it produces an optimal EBD score between the attributes involved in the match. We will also use the above definitions and properties to derive the precise conditions under which our 1:N matching algorithm produces an optimal EBD score.

6.3 1:N Matching Algorithm

Program 1:N_Matching illustrates our algorithm. The algorithm depicts data structures such as $F(T_2)^N$ and $F_{EBD}(T_2)^N$ without subscripts denoting their size. This is because the for-loop eliminates the need to explicitly denote the sizes of these data structures. The inputs to 1:N_Matching (not depicted) are the composite attribute Cmp ϵ T_1 and the attributes of T_2. The outputs are $F(T_2)$ and $F_{EBD}(T_2)$. After initializations, the Cmp's GT set is retrieved using GT information from 1:1 matches between Cmp and attributes from T_2. Next, all candidate attributes are assigned to $C(T_2)$ based on its 3 membership conditions.

```
program 1:N_Matching (F(T2), FEBD(T2))
  var C(T2) = Φ; var F(T2) = Φ; FEBD(T2) = 0.0;
  GTCmp = getGTSet(Cmp);
  C(T2) = getMatchCandidates(Cmp, T2, GTCmp);
  C(T2) = orderByEBD(C(T2));
  For att A ϵ C(T2) with max value of EBD(Cmp,A){
     if (increaseEBD(Cmp, FEBD(T2)) {
        Cmax = A;
        F(T2) = F(T2) U Cmax;
        FEBD(T2) = addEBD(FEBD(T2), EBD(Cmp, Cmax))
     end if
     C(T2) = C(T2) – A;
   end for
```

After ordering the attributes in $C(T_2)$ by EBD score with Cmp, the attribute in $C(T_2)$ that currently possesses the maximum EBD score with Cmp in a 1:1 match is selected. After, A is tested for inclusion into $F(T_2)$. Membership into $F(T_2)$ is based on 3 conditions. The first 2 were tested as part of getMatchCandidates(). The third determines if the value of EBD(Cmp, $F(T_2)$) increases with the inclusion of attribute A. If so, then Cmax is assigned A and included into $F(T_2)$, $F_{EBD}(T_2)$ is recomputed and A is removed from $C(T_2)$. Finally, $F(T_2)$ and $F_{EBD}(T_2)$ are returned as outputs.

The time complexity of 1:N_Matching is O(n). The longest step occurs at line 6 in O(n) time through the use of count-sort to sort the EBD values from 1:1 matches of attributes in T_2 with Cmp. The analysis of the for-each loop is as follows. The selection of the next attribute to be examined takes O(1) time, since the sorting algorithm of line 6 ensures that attributes from T_2 are ordered in $C(T_2)$ in decreasing order of EBD with Cmp. Line 8 is also O(1), since it is just a GT. Line 9 takes O(1) time because all of the values in $F_{EBD}(T_2)$ have already been computed from 1:1 matching. Lines 10-14 are each O(1). Since the for-each loop examines all attributes in T_2 exactly once, it takes O(n) time.

6.4 Proof of Correctness

Theorem 1: (Proof of Greedy Choice Property for 1:N matching algorithm) – *All choices for $Cmax_x(T_2)$ will be present in an optimal 1:N match with Cmp ϵ T_1.*

Suppose that $F_{EBD}(T_2)^N$, for an arbitrary $F(T_2)^N$, produces an optimal EBD. Let us build a new set called $F2_{EBD}(T_2)^N$ from $F_2(T_2)^N$ such that every attribute included in $F2(T_2)^N$ is in $Cmax_x$ (T_2) for some x. Also, the cardinality of $F(T_2)^N$ and $F2(T_2)^N$ are equal, and every attribute between $F(T_2)^N$ and $F2(T_2)^N$ is identical, except for an arbitrary attribute indexed by r (r <= N) in $F2(T_2)^N$. Then by the definition of $Cmax_x$, the EBD value produced between Cmp and attribute r in $F2(T_2)^N$ is >= the EBD value produced between Cmp and attribute r given in $F(T_2)^N$. Since all other attributes are equal between $F(T_2)^N$ and $F2(T_2)^N$, then their associated 1:1 EBD scores with Cmp are also identical. Therefore, EBD(Cmp, $F2(T_2)^N$) >= EBD (Cmp, $F(T_2)^N$), but since $F(T_2)^N$ produces an optimal EBD with Cmp through $F_{EBD}(T_2)^N$, then EBD(Cmp, $F2(T_2)^N$) = EBD (Cmp, $F(T_2)^N$). Thus, $F2(T_2)^N$ also produces an optimal EBD with Cmp through $F2_{EBD}(T_2)^N$.

Theorem 2: (Proof of optimal substructure property) – *Let $F_{EBD}\ (T_2)^{N-1}$, N > 1, be the EBD score corresponding to the attribute match between Cmp ϵ T_1 and $F(T_2)^{N-1}$ ϵ T_2. If $F_{EBD}(T_2)^{N-1}$ is an optimal EBD score, and $F_{EBD}(T_2)^N$ is obtained by adding Cmax to $F(T_2)^{N-1}$, then $F_{EBD}(T_2)^N$ must also be an optimal EBD score.*

Assume that $F(T_2)^N$ was formed by adding Cmax to $F(T_2)^{N-1}$, but does not produce an optimal value of $F_{EBD}(T_2)^N$. Cmax represents the attribute with the highest EBD score with Cmp to be included in $F(T_2)^{N-1}$ with respect to all other attributes in $C(T_2)$. Then this means that $F(T_2)^{N-1}$ contains some attribute indexed by r (r <= N-1) whose EBD value is less than that of $Cmax_r$. Thus, $F_{EBD}(T_2)^{N-1}$ is not an optimal EBD score. This contradicts the statement above that $F_{EBD}(T_2)^{N-1}$ is an optimal EBD score. Therefore, if $F_{EBD}(T_2)^{N-1}$ is an optimal EBD score, and $F_{EBD}(T_2)^N$ is obtained by adding $Cmax_x$ to $F(T_2)^{N-1}$, then $F_{EBD}(T_2)^N$ must be an optimal EBD score.

Theorem 3: *Greedy 1:N matching produces a safe match with an optimal EBD score.*

This follows from Theorem 1 and Theorem 2.

6.5 1:N EBD Optimality Proof

To formally outline the conditions under which our 1:N matching algorithm produces an optimal EBD score, we will use the definitions and properties stated in section 6.2. We will begin by proving the following statement.

Theorem 4: *Suppose a 1:N match between Cmp and attributes of T_2 is being formed. Let $C(T_2)^j$ where $j< |C(T_2)^0|$, be the attributes of T_2 satisfying Conditions 1, 2 and 3 above. Let $F(T_2)^k$ and $F_{EBD}(T_2)^k$, where $k < |F(T_2)^C|$, be the set of attributes in T_2 already part of the 1:N match with*

Cmp and the associated EBD score between those attributes, respectively. Also assume that Properties 1,2 and 3 have been observed throughout the matching up to this point. If an attribute A from $C(T_2)^j$ *is added to* $F(T_2)^k$*, forming* $F(T_2)^{k+1}$ *and* $F_{EBD}(T_2)^{k+1}$*, such that GT(A) is not in GT(Cmp), violating properties 2 and 4, then* $F_{EBD}(T_2)^{k+1} < F_{EBD}(T_2)^k$.

Proof: To prove this property, note that in Equation 1, EBD is proportional to conditional entropy. Furthermore, property 3 guarantees that the number of instances in Cmp involved in a 1:N match >= the number of instances in $F(T_2)^C$ involved in the match. This means that the entropy, H(A), will always increase, regardless of whether attribute A's instances are of a valid GT. Therefore, Theorem 4 can be proved if we can show that the conditional entropy of $F(T_2)^{k+1}$ < the conditional entropy of $F(T_2)^k$.

To show this, let us state the equation for conditional entropy:

$$-\sum_{t\epsilon T; a\epsilon A} p(t,a)\log \mathrm{p}(a|t)$$

this can be rewritten in the following way:

$$-\sum_{t\epsilon T; a\epsilon A} p(t)p(a|t)\log\left(p(a|t)\right)$$

Suppose that GT(A) = {Newtype}. For any type (represented by t) other than Newtype, the value of p(t) over $F(T_2)^k$ is greater than the value of p(t) over $F(T_2)^{k+1}$. This is because p(t) represents the probability that an instance among the attributes of $F(T_2)^k$ (for $0 < k <= |F(T_2)^C|$) has a GT of type t. By adding attribute A, the total number of instances over $F(T_2)^{k+1}$ is greater than the total number of instances over $F(T_2)^k$. By Property 5, A does not introduce any instances with a GT other than Newtype, so the number of instances for type t remains the same. Thus, p(t)

is a lower value for $F(T_2)^{k+1}$ when A is added to $F(T_2)^k$. However, p(a|t) at $F(T_2)^k$ is equal to p(a|t) at $F(T_2)^{k+1}$ when A is added. As a result, for any type other than Newtype, $-p(t)p(a|t)\log(p(a|t))$ at $F(T_2)^{k+1} < -p(t)p(a|t)\log(p(a|t))$ at $F(T_2)^k$. When t = Newtype, $-p(t)p(a|t)\log(p(a|t)) = 0$ because p(a|t) = 0 when a = Cmp and log(p(a|t)) = 0 when a = $F(T_2)^{k+1}$. Taking the above into consideration, the conditional entropy produced among the attributes of $F(T_2)^k$ must be greater than the conditional entropy produced among the attributes of $F(T_2)^{k+1}$. Because the entropy must increase upon adding A to $F(T_2)^k$, $F_{EBD}(T_2)^{k+1} < F_{EBD}(T_2)^k$.

The contrapositive of Theorem 4 is also important and will be stated as Theorem 5.

Theorem 5: Given the same assumptions that we were given in Theorem 4, if $F_{EBD}(T_2)^{k+1} >= F_{EBD}(T_2)^k$, then when attribute A from $C(T_2)^j$ is added to $F(T_2)^k$, forming $F(T_2)^{k+1}$ and $F_{EBD}(T_2)^{k+1}$, GT(A) is in GT(Cmp).

Because Theorem 5 is the contrapositive of Theorem 4, this is a true statement. From this, there are only two ways that EBD(Cmp, $F(T_2)$k) can increase when adding a new attribute A that satisfies all aforementioned match properties and conditions (1: GT(A) was not in $GT(F(T_2)^k)$ but is in $GT(F(T_2)^{k+1})$ (2: GT(A) is in both $GT(F(T_2)^k)$ and $GT(F(T_2)^{k+1})$, but adds more instances to the GT to bring it closer to the total possessed by Cmp for that same GT. Based on this and theorems 4 and 5, we can state the following lemma describing 1:N matches that will yield optimum EBD values.

1:N EBD Optimality Lemma: *EBD(Cmp, $F(T_2)^C$) always attains its highest value when minimizing (1: GT(Cmp) – $GT(F(T_2)^C)$ (2: for each GT, the difference between the number of instances in Cmp and the number of instances in $F(T_2)^C$.*

As an example of how our algorithm performs a 1:N match using the definitions and theorems just described, we will look at figure 6.1. It depicts four separate 1:1 matches with each involving Address(T_1) and one of Street Address (T_2), City(T_2), State(T_2), Zip(T_2). Address(T_1) is a composite attribute, since its GT set over all of its instancs = {Street_Address, City, State, Zip}. GT(Street Address(T_2)) = {Street_Address}, GT(City(T_2)) = {City}, GT(State(T_2)) = {State} and GT(Zip(T_2)) = {Zip}. These attributes are not composite because the size of their GT sets is = 1. In order to determine if a 1:N match exists between Address(T_1) and the attributes of T_2, the GT sets of all attributes involved in the match, as well as the 1:1 EBD scores between Address(T_1) and the matching attributes from T_2 will be used. Since id(T_2) did not match with Address(T_2), it is not included in the 1:N match. The 1:1 EBD scores between Address(T_1) and the matching attributes from T_2 are sorted in decreasing order, such that $C(T_2)^0$ = {Street Address(T_2), City(T_2), State(T_2), Zip(T_2)}. These attributes were included into $C(T_2)^0$ because they met Conditions 1, 2 and 3 stated above. At this point, $F(T_2)^0$ = {} and $F_{EBD}(T_2)^0$ = 0. The match begins by choosing the first attribute in C(T_2), Street Address(T_2), since it had the highest 1:1 EBD score with Address(T_1). Here, Street Address(T_2) is Cmax. In order to determine if Street Address(T_2) should be included into F(T_2), $F_{EBD}(T_2)^1 > F_{EBD}(T_2)^0$. Since EBD(Address($T_1$), Street Address($T_2$)) = .76 > 0, $F(T_2)^1$ = {Street Address(T_2)} and $F_{EBD}(T_2)$ = .76. The process will continue until all attributes from C(T_2) have been considered for inclusion into F(T_2).

6.6 Experiments

This experiment analyzed GSim's 1:N matching capabilities over two pairs of matching tables. The first pair is derived from a database of travelling employees, while the second pair represents island groups.

For the employee tables, table T_1 contains the composite attribute 'Address', where GT('Address') = {'Street Address', 'City', 'State', 'Zip'}. Table T_2 contains individual attributes with street address, city, state and zip code. Because the tables contain information about travelling employees, some employees have more than one residence. As a result, T_2 contains two versions of each attribute (ie: street address is represented by 'stadd1' and 'stadd2'). The challenge was to match 'Address' ϵ T_1 with a combination of attributes ϵ T_2 such that the sets of GTs are equivalent. Also, our algorithm should not include two or more attributes ϵ T_2 with the same GT set.

For the islands database, table T_1 contains the composite attribute 'Island_Group' where each instance consists of a list of islands; GT('Island_Group') = {Island1, Island2, Island3, Island4, Island5}. Table T_2 contains individual attributes named 'Island1', 'Island2', 'Island3', 'Island4', and 'Island5'. T_2 also contains attribute 'Island6' whose GT = {'Island6'}. An instance of 'Island_Group' ϵ T_1 does not contain 'Island6' in its GT set. Here, the challenge is to identify the match between 'Island_Group' and the attributes 'Island1', 'Island2', 'Island3', 'Island4' and 'Island5' without including 'Island6'.

Attribute ϵ T_2	EBD w/'Address'	Current 1:N EBD
Street Address	.75	.75
City	.71	.84
State	.70	.90
Zip	.64	.95

Figure 6.3. EBD scores between 'Address' and attributes of T_2 for the employee tables

Attribute ϵ T_2	EBD w/'Island Group'	Current 1:N EBD
Island1	.82	.75
Island2	.80	.84
Island3	.73	.90
Island4	.66	.95
Island5	.61	.97

Figure 6.4. EBD scores between 'Address' and attributes of T_2 for the employee tables

Figure 6.3 displays the result of 1:N matching applied to the employee tables. The respective EBD values between 'Street Address', 'City', 'State' and 'Zip' with 'Address' are all > .6. The 1:N EBD score increases as additional attributes ϵ T_2 are matched with 'Address'. At the same time, the value of N increases. When it finishes, 'Street Address', 'City', 'State' and 'Zip' match with 'Address' with an EBD of .95. Through all simulations, our algorithm did not include 2 attributes ϵ T_2 with the same GT set in a 1:N match. Figure 6.4 below displays 1:N matching results for the tables containing island groupings. 'Island_Group' has valid 1:1 matches with 'Island1', 'Island2', 'Island3', 'Island4' and 'Island5'. No such match exists between 'Island_Group' and 'Island6'. Thus, 'Island6' cannot take part in a 1:N match with 'Island_Group'. The right column shows the progression of the EBD value as the 1:N match forms. The end result is a match between 'Island1', 'Island2', 'Island3', 'Island4', 'Island5' and 'Island_Group' with an EBD score of .97.

CHAPTER 7

ANDROID MOBILE PHONE SECURITY

In this chapter, we will take a departure from GSim and discuss mobile phone security in phones running on the Android operation system. In particular, we will examine the problem of applications being granted permissions to use any phone resource at any time, even though the use has no idea of the intent of the application. Additionally, by the time the application requests permissions to user certain phone resources, it is likely that the user has forgotten which permissions that application was given in the first place. This is due to the fact that permissions are granted to an application at install time. In the following pages, we will describe the problem with more background and detail and propose a solution that emphasizes the creation of a systems containing security authorization rules that only grant resource permissions to an application after passing a series of runtime checks. Our work is based on a previously proposed access control system, but we add a novel feature to it that adds greater flexibility and has the potential to incorporate a greater degree of user privacy.

7.1 Overview

Android(Google), introduced by Google, is the most popular open source mobile phone operating system, installed on 43% of smart phones in the United States as of September 26, 2011. This is significantly more than Apple's iOS (28%) and RIM's Blackberry (18%), due in part to the fact that Android is installed on phones created by multiple vendors, while iOS is installed

only on the iPhone(eWeek.com). However, Android is also extremely popular because it is open source and possesses an architecture based on customizable software stacks. It includes the OS itself, various system utilities such as a Traffic Monitor for measuring WiFi data traffic, a folder organizer, file explorer, etc., middleware represented by virtual machines associated with applications, and core applications including a web browser, dialer, calculator and more. Being open source, third party developers may build applications for Android and submit them to the Android Market where users can download and install them. While its open source nature allows for maximum user customization, it also implies a heightened security risk, since a malicious application that can request a series of permissions from other applications on the phone, such as system utilities, can be uploaded to the Android Market for subsequent download by unsuspecting users. As a result, when an application is installed, the user must simply trust that this application will not exhibit any malicious intent. To counter this threat, during the application installation process, Android presents to the user all of the permissions needed by the application. Unfortunately, if the user wants to be able to use the application at all, he/she must accept all permissions requested by the application at install time. Furthermore, an antsy user, thinking only of using the application, may not be aware or interested in the permissions being granted. As a final point, once the user agrees to the permission requests, there is no way for the user to later retract those permissions.

At this point, it is possible for other applications to access a phone resource or another application, regardless of whether the application is malicious or not. For instance, a user may download what he/she thinks is a weather application that uses the GPS and Internet resources to retrieve information about the local weather forecast in the user's present locale. However, it is

quite possible that the application, having access to the Internet, will also download ads and malicious software onto the user's phone, where it will unwittingly be part of a Botnet.

7.2 Runtime Permission Constraints

One solution to this problem was outlined by (Nauman 2010), where the authors created an access control specification language known as APEX (Android Permissions Extension) that allows for the creation of runtime constraints that applications need to fulfill in order to access the permissions that were already granted to it at install time. In other words, if a calling application wants access to use another application or a specific phone resource (ie: GPS), then it would not only depend on the user explicitly giving permission at runtime, but also on the calling application meeting certain runtime requirements specified by the user. This is done by the user creating a policy file, where the policy file would contain all of the necessary runtime constraints that an application would need to fulfill before accessing either a phone resource or an application. Formally, (Nauman 2010) defines a policy as follows:

Definition 18 (Policy): *A policy defines the conditions under which an application is granted a permission. It consists of two input parameters: (1: an application and (2: a permission. The policy also contains an authorization rule composed of predicates that specify the conditions under which the permission is granted/denied and a set of attribute update actions, which are to be performed if the conditions in the authorization rule are satisfied. Specifically:*

$$P(a, p): c_1 \text{ AND|OR } c_2 \text{ AND|OR } c_3 \text{ AND|OR } \ldots \text{ AND|OR } c_n \rightarrow \{permit, deny\}$$

Here, P is the policy, a is an Android application, p is a permission required by the calling application for its proper functioning, c is a runtime constraint that needs to be satisfied by the

calling application in order to be granted permission p. If all runtime constraints are satisfied, then permission p will be granted to the calling application. The entire conditional statement, including the parameters list (a,p), the constraints c_1 through c_n and the consequent {permit,deny}, is an authorization rule. Notice that an AND or OR connective can be used to connect any two runtime constraints. The complete formal specification of the APEX system is further described in (Nauman 2010).

A snapshot of some example authorization rules illustrated by APEX is in Figure 7.1 and 7.2 below(Nauman 2010). In figure 7.1, APEX creates a policy called *deny_gps*, which takes as the parameters the android application, "edu.apex.ringlet.Ringlet" and the permission, "android.permission.ACCESS_FINE_LOCATION". They are both given aliases ('Ringlet' for the application and 'GPS' for the permission) in order to simplify naming. For the *deny_gps* policy, one authorization rule is specified consisting of two runtime constraints.

```
deny_gps("edu.apex.ringlet.Ringlet" as Ringlet,
         "android.permission.ACCESS_FINE_LOCATION" as GPS):
  System.CurrentTime > 1700 ∨ System.CurrentTime < 0900
                            → deny(Ringlet, GPS);
```

Figure 7.1. The *deny_gps* policy from APEX

```
restrict_internet("edu.ringlet.Ringlet" as Ringlet,
                  "android.permission.INTERNET" as Net):
  true → deny(Ringlet, Net);
```

Figure 7.2. The *restrict_internet* policy from APEX

The first runtime constraint checks the current system time to see if it is at or past 1700 hours, while the second runtime constraint checks to see if the current system time is before 0900 hours. If either of these are true, then the Ringlet application is denied access to the GPS resource.

Figure 7.2 depicts a much simpler policy called the *restrict_internet* policy that simply denies the Ringlet application permission to access the Internet.

7.3 Extensions to Runtime Permissions

Building on the work of APEX briefly described in the previous section, we now discuss how we would extend their proposed system to more effectively and flexibly enforce access control of permissions within Android phones.

First, the authorization rules described in the previous section would need to be contained within a policy file made specifically for the Ringlet application, as they both impose an authorization rule associated with Ringlet. In order to locate this policy file, we would add markup to the manifest file for Ringlet to point to the policy. Figure 7.3 below illustrates the appropriate markup that would need to be included in the manifest file for Ringlet in order to use the authorization rule specified in Figure 7.1.

```
<policy>
  <file>edu.apex.waze.policies</file>
  <rule>deny_gps</rule>
</policy>
```

Figure 7.3. The markup in the manifest file for Ringlet that points to a policy containing the *deny_gps* authorization rule

While APEX appears formally sound and certainly represents a feasible specification for enforcing runtime permissions access, improvements are possible. The one which will serve as the focus here involves the inclusion of enumerations and function calls into an application's manifest file for usage within an authorization. Two separate examples will be given – one

involving the use of GPS that serves to cloak the location of a mobile user for privacy purposes, and another which restricts from where a particular ad supported application can downloads ads. This would be useful to ensure that a potentially untrusted application given the permission to use the Internet only downloads from a predefined set of URLs. At the moment, neither of these has been implemented; these are merely proposals for extending the work of APEX to include additional runtime constraints. However, work is progressing on implementing these ideas.

7.3.1 GPS Example

To illustrate the first improvement mentioned above, consider the following scenario. Say a user wanted to create a policy where his/her current location as indicated by GPS would dictate whether an application would post their current GPS location (either within an application on the phone or to a social networking site that the application is connected to), then authorization rules involving dynamically determined values would need to be created. For instance, in Figure 19a above, the values specified for the runtime constraints in the authorization rules are constant values (ie: System.CurrentTime > **1700** or System.CurrentTime < **0900**). For this rule, this is acceptable. However, if the user wanted to create a rule in which their GPS location would not be posted based on their current GPS value, then it would not be feasible in many cases for the user to specify a particular GPS value in the rule. This is because many users do not know what the value of their current GPS location might be at a given moment, thus making its static specification as a constant value in an authorization rule impossible. Furthermore, it would be more useful if the user was able to specify a range, or a bounding box of GPS values constituting an enclosed area where GPS values were not to be posted publicly by a particular application.

One potential scenario for the usage of such an authorization rule would be if a mother of a child concerned about stalkers wanted to hide her GPS location when she picked up her child from school. She might also want to conceal the GPS location of her workplace, the residences of her family members, a local playground, In both cases, however, she would want her phone to remain on, so that she could receive important phone calls from her family or workplace.

In such a case, we could create a policy involving enumerations and runtime function calls. Each enumeration would evaluate to one of several previously defined values at runtime, while a function call would be used to query a phone resource to acquire a value, and compare the returned value with all of the values in the defined enumeration. Figure 7.4 below illustrates such a policy.

```
<restricted_locations>
  <location>
    <value>"37.715183, -117.260489","37.715214, -117.260516"</value>
    <name>school</name>
  </location>
  <location>
    <value>"37.623190, -117.010488","37.625554, -117.062276"</value>
    <name>workplace</name>
  </location>
  <location>
    <value>"37.722587, -117.266192","37.722589, -117.266202"</value>
    <name>bus_stop</name>
  </location>
</restricted_locations>
```

Figure 7.4. Markup for the <restricted_locations> enumerations for use in an authorization rule that restricts the publication of GPS values

Here, we refer to an Android app known as Waze(Waze), which provides real-time traffic updates about road congestion by leveraging GPS values from various users' mobile phones in the area. In the case of the mother who wants her GPS value kept private while she is picking up her child from school, she could define within the manifest file for Waze the following enumeration called *restricted_locations.* Being that the enumeration is in the manifest file for Waze, we would have to define it as a series of markup tags.

We would then need to add markup for a runtime function call obtaining the current GPS value of the phone. The implementation of the function would be based on the LocationManager class of the android.location.* package(Android Developers), and it would be located in a class file within the directory for Waze. We would also need to specify the name of the function in the class file; we will use this same function name in the authorization rule. Figure 7.5 below illustrates the markup to be inserted into Waze's manifest file to leverage function calls in authorization rules.

```
<function>
   <name>GetGPS</name>
   <class>edu.apex.android.waze.wazepolicy</class>
</function>
```

Figure 7.5. Markup for defining the *GetGPS* function call used in the authorization rule defined in the Waze policy

Now we can redefine the policy depicted in Figure 7.1 to include GPS location restrictions; figure 7.6 illustrates this.

```
deny_gps ("edu.apex.waze.Waze" as Waze, android.permission.ACCESS_FINE_LOCATION" as GPS):
(System.CurrentTime > 1700  OR System.CurrentTime > 0900) AND GetGPS("restricted_locations") =
true --> deny(Waze, GPS)
```

Figure 7.6. Authorization rule *deny_gps*

7.3.2 Internet Example

We can also envision a situation where enumerations and function calls can be used to specify from where an application should downloads ads (if the app is add supported). This might be useful in a situation where the application is given permission to use the Internet, but the user wants to ensure that the application does not download ads from any other malicious locations. Here, we assume that the application provides all valid URLs from which it downloads ads.

As an example, say we wanted to limit from where a new weather-based app known as WeatherReport could download ads. First, we would define an authorization rule in WeatherReport's policy file called *restrict_internet* depicted in Figure 7.7.

```
restrict_internet("edu.weathereport.WeatherReport"  as WeatherReport
                  "android.Permission.INTERNET" as INTERNET):
IsValidURL("restricted_internet") = false --> deny(WeatherReport, INTERNET)
```

Figure 7.7. The improved *restrict_internet* authorization rule, now including the IsValidURL() function call that checks for valid ad URLs.

To make this rule work, we would then need to define an enumeration consisting of the valid URL's from which WeatherReport could download ads; Figure 7.8 shows this. We would also need to specify a function call (IsValidURL()) that compares the URL the application presently wishes to use to access the Internet with the valid set of URLs the application can use as defined in the *restricted_internet* enumeration from Figure 7.8.

```
<restricted_internet>
  <URL>
    <value>http://www.adlocation.com/loc1</value>
    <name>Location 1</name>
  </URL>
  <URL>
    <value>http://www.adlocation.com/loc2</value>
    <name>Location 2</name>
  </URL>
  <URL>
    <value>http://www.adlocation.com/backup/</value>
    <name>Backup</name>
  </URL>
</restricted_internet>
```

Figure 7.8. The *restricted_internet* enumeration

As a final note, it should be stated that the implementation of the Apex policies and the extensions to it discussed here involving the definitions of enumerations, function calls and policy files can only be realized with a change to the underlying Android codebase. More specifically, Android would need to keep track of which application was running, so that when the time comes for that application to acquire permissions to use a phone resource, it would need to do the following: (1: Find the <policy> tagset within the application's manifest file, so that the authorization rule can be located (2: Execute the authorization rule. If a function call is encountered, then refer to the <function> tagset defined in the manifest file to locate the function code. (3: The argument within the function call should be an enumeration whose name is defined as a tagset within the manifest file for the application. The current value that the application wants to use for a particular resource (i.e. GPS coordinates, or a URL) must then be compared against each enumeration value. The end result is a boolean value being evaluated for the function call, which when combined with the boolean values evaluated for the other runtime constraints, results in either the application either being granted or denied access to the resource.

CHAPTER 8

FUTURE WORK

In this dissertation, we have described our work with geospatial schema matching through our tools GSim and GeoSim. In particular, we have discussed algorithms associated with geographic type (GT) matching, non-geographic type (NGT) matching, attribute weighting, geosemantic clustering, multi-attribute matching (1:N matching) and Android mobile phone security. While these ideas were explored in great detail, many directions for future work on each of these ideas remain viable as potential research topics. We will touch on some of these possible directions in this chapter.

8.1 Geographic and Non-Geographic Matching

Future efforts to improve GSim's geographic and non-geographic matching components will focus on the following. First, we will refine our GT extraction techniques. This can be done in two ways. The first is to leverage multiple gazetteers making use of heterogeneous feature type thesauri while enhancing our recall of the correct type information. The second way is to apply pruning techniques to a given EBD calculation between two compared attributes. This way, geographic types represented by a very small number of instances are not considered in the final EBD calculation. The idea behind this is to correlate high EBD scores with high frequencies of instances across all present GTs. Second, we will work on supporting gazetteers, like ADL, that organize their feature type thesauri in an ontological fashion. Although we partially realized this

through GeoSim's hierarchical matching algorithm, it should be noted that this work was based on only 1 GT hierarchy (from the ADL gazetteer). Third, we plan on extending GSim such that geoontologies can be just as easily compared for similarity as geodatabases. To this end, we also plan on adapting suitable algorithms for comparing ontologies, such as structural and neighborhood matching techniques. We would then integrate them into a more sophisticated GT matching algorithm. Finally, we plan on implementing the algorithm outlined in section 3.3.5 to overcome inadequate attribute mappings produced by NGT matching by using the available GT information from instances.

8.2 Attribute Weighting

We plan on continuing our work on attribute weighting in the following ways. First, we will continue to apply attribute weighting over an increasing variety of schemas and domains. For instance, if we wanted to apply attribute weighting to object properties of concepts within ontologies, then we would need to take into account the relative positions of concepts in the ontological hierarchy and incorporate that into our weighting scheme. In this case, uniqueness and relevance of object properties would not be sufficient by themselves to properly weigh object property matches between concepts. Another factor that could alter the weights between matches between object properties in ontologies are the relationships between concepts themselves. We might decide that for a given pair of concepts related to each other via a superclass/subclass relationship, any 1:1 match of object properties between them might deserve more weight versus a meronymous match between a pair of concepts and their respective matching object properties. Also, as stated above, we would apply attribute weighting across domains beyond the geospatial

domain, as we would want to generalize our algorithm as much as possible to maximize its applicability.

8.3 Geosemantic Clustering

We will extend our work with geosemantic clustering in the following ways. First, we will address clustering situations where a majority of the instances (or most of the instances) do not possess a discernible GT. In this scenario, we would have to rely almost entirely on the NGD between instances and their associated medoid. However, if the instances did indeed possess some intrinsic geographic information, then the clustering might be changed significantly enough to alter the attribute match. For instance, the instances might not contain an explicit GT, but they may contain geometric information or related polygon information that can tell us the shape of the feature that this instance represents. Based on this, we can whittle down the possibilities as far as what GT the feature may represent. In addition, it may be possible to determine an instance's surrounding features, possibly using some form of region connected calculus. If this was possible, then it may be possible to infer the feature's GT. For instance, if we have a feature without an explicit GT, but it is known that it is located next to a harbor feature and is part of a port feature, then it is likely that this feature may be a wharf. Another way in which geosemantic clustering may be extended is to explore alternative clustering techniques such as density-based clustering, or even employ a limited rule set to improve the clustering in specific ways through association-rule mining.

8.4 Multi-Attribute Matching

We plan on extending our work on multi-attribute matching as follows. First, we plan on extending our algorithm from 1:N matching to M:N matching, where m, n > 1. This is an extremely difficult problem that has not been addressed in the literature very much due to the inherent computational intractability of a solution. However, by utilizing 1:1 matches previously generated by our greedy 1:N matching algorithm, we believe that this can reduce the complexity of the problem considerably. We do not know at the moment whether it is possible to create an polynomial time or sub-polynomial time algorithm using our approach, but our investigation will continue. In addition to this, we plan on performing additional experiments with our 1:N matching algorithm to generalize its effectiveness over an increasing number of matching situations.

8.5 Android Mobile Phone Security

As stated in chapter 7, our future work regarding our proposed Android runtime permissions policy engine is simply to implement and test it for viability. One of the major goals to this end is to minimize the number of changes that need to be made to the Android codebase. Other ideas regarding Android mobile phone security can extend from this. Two possible ideas are as follows. First, we can look into implementing a lattice-based access control model based on the work of Denning (Denning 1976, 236-43), where a partially-ordered set in this case would be the phone permissions, and the binary relation operating on pairs of permissions would be an adapted “flow to” operator which determines if an application in a given state (as defined by the set of permissions it contains at a particular point in time) can flow to another state, if granted

access to a particular phone resource. Second, we can employ ideas described in (Sweeney 2002, 557-70) regarding k-anonymity in order to grant mobile phone privacy to a user. K-anonymity in this case ensures that a particular user's identity on a wireless network, as indicated by their mobile phone, remains indistinguishable from the mobile identities of k-1 other users. This would helpful in a situation where a user needs to remain connected to a wireless network, but does not want to reveal their current location. This may pertain to either a concerned mother of a child who does not want to let others know where she picks up her child at school, or it could pertain to a government agent that needs to remain connected, but cannot announce his/her location publicly.

REFERENCES

Ahlqvist, Ola, and Ashton Shortridge. 2006. Characterizing land cover structure with semantic variograms. Paper presented at the 12th International Symposium on Spatial Data Handling, Vienna, Austria, July 10-12.

Albertoni, Riccardo, and Monica De Martino. 2006. Semantic similarity of ontology instances tailored on the application context. Paper presented at On the Move to Meaningful Internet Systems, Montpellier, France, October 29-November 3.

Alexandria Digital Library Project. ADL Gazetteer Development. http://www.alexandria.ucsb.edu/gazetteer (Accessed June 1, 2010).

Android Developers. LocationManager|Android Developers. http://developer.android.com/reference/android/location/LocationManager.html (accessed 9/1/2011).

Auer, Sören, Jens Lehmann, and Sebastian Hellmann. 2009. LinkedGeoData - adding a spatial dimension to the web of data. Paper presented at the 8th International Semantic Web Conference, Washington, D.C., USA, October 25-29.

Bauckhage, Christian, and Christian Thurau. 2010. Adapting information theoretic clustering to binary images. Paper presented at the 20th Conference of the International Association for Pattern Recognition, Istanbul, Turkey, August 23-26.

Berlin, J., and A. Motro. 2001. Autoplex: automated discovery of instance for virtual databases. Paper presented at the 9th International Conference on Cooperative Information Systems, Trento, Italy, September 5-7.

Bohannon, P., E. Elnahrawy, W. Fan, and M. Flaster. 2006. Putting context into schema matching. Paper presented at the 32nd International Conference on Very Large Data Bases, Seoul, South Korea, September 12-15.

Bouchon-Meunier, B., M. Rifqi, and S. Bothorel. 1996. Towards general measures of comparison of objects. *Fuzzy Sets and Systems* 84: 143-53.

Brauner, Daniela F., Chantal Intrator, João Freitas, and Marco A. Casanova. 2007. An instance-based approach for matching export schemas of geographical database web services. Paper presented at GeoInfo, Campos do Jordão, Brazil, November 25-28.

Brauner, Daniela F., Marco A. Casanova, and Ruy Luiz Milidiú. 2006. Towards gazetteer integration through an instance-based thesauri mapping approach. Paper presented at GeoInfo, Campos do Jordão, Brazil, November 19-22.

Bui, Trung H. Matthew Frampton, John Dowding, and Stanley Peters. 2009. Extracting decisions from multi-party dialogue using directed graphical models and semantic similarity. Paper presented at the 10th Annual SIGDIAL Meeting on Discourse and Dialogue, London, U.K., September 11-12.

Cheng, Reynold, Jian Gong, and David W. Cheung. Managing uncertainty of XML schema matching. 2010. Paper presented at the 26th International Conference on Data Engineering, Long Beach, CA, USA, March 1-6.

Chiticariu, L., and W.C. Tan. Debugging schema mappings with routes. 2006. Paper presented at the 32nd International Conference on Very Large Data Bases, Seoul, South Korea, September 12-15.

Cilibrasi R.L., and P.M.B. Vitanyi. 2007. The Google similarity distance. *IEEE Trans. Knowledge and Data Engineering*. 19(3): 370-83.

Cruz, Isabel F., Flavio Palandri Antonelli, and Cosmin Stroe. 2009. AgreementMaker: efficient matching for large real-world schemas and ontologies. *PVLDB* 2(2): 1586-1589.

Cruz, Isabel F., Flavio Palandri Antonelli, Cosmin Stroe, Ulas C. Keles, and Angela Maduko. Using AgreementMaker to align ontologies for OAEI 2009: overview, results, and outlook. 2009. Paper presented at the 4th International Workshop on Ontology Matching, Washington, D.C., USA, October 25.

Cruz, Isabel F., William Sunna, Nalin Makar, and Sujan Bathala. 2007. A visual tool for ontology alignment to enable geospatial interoperability. *J. Vis. Lang. Computing* 18(3): 230-54.

Dai, Bing Tian, Nick Koudas, Divesh Srivastava, Anthony K.H. Tung, and Suresh Venkatasubramanian. Validating multi-column schema matchings by type. 2008. Paper presented at the 24th International Conference on Data Engineering, Cancún, México, April 7-12.

Dang, Xuan Hong, and James Bailey: A hierarchical information theoretic technique for the discovery of non linear alternative clusterings. 2010. Paper presented at the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, D.C., USA, July 25-28.

Denning, Dorothy E. 1976. A lattice model of secure information flow. *Commun. ACM* 19(5): 236-43.

Dhamankar, Robin, Yoonkyong Lee, AnHai Doan, Alon Y. Halevy, and Pedro Domingos. iMAP: Discovering Complex Mappings Between Database Schemas. 2004. Paper presented at ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18.

Doan, A., P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: a machine-learning approach. 2001. Paper presented at the ACM SIGMOD International Conference on Management of Data 2001, Santa Barbara, CA, USA.

Embley, D.W., L. Xu, and Y. Ding. Automatic direct and indirect schema mapping: experiences and lessons learned. 2004. Paper presented at ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18.

Ertico. GDF - Geographic Data Files. http://www.ertico.com/gdf-geographic-data-files (accessed May 1, 2010).

Euzenat, Jérôme, and Pavel Shvaiko. 1998. *Ontology Matching*. New York: Springer-Verlag.

eWeek.com. Android Hits 43%, iOS Stuck on 28% Share: Nielsen. http://www.eweek.com/c/a/Mobile-and-Wireless/Android-Hits-43-iOS-Stuck-on-28-Share-Nielsen-848755/ (accessed 9/1/2011).

Fellbaum, Christiane, ed. 1998. *WordNet: An electronic lexical database*. Cambridge: The MIT Press.

Finin, Tim, and Zareen Syed. Creating and exploiting a web of semantic data. 2010. Paper presented the 2nd International Conference on Agents and Artificial Intelligence, Valencia, Spain, January 22-24.

Fink, Clayton, Christine D. Piatko, James Mayfield, Danielle Chou, Tim Finin, and Justin Martineau. The geolocation of web logs from textual clues. 2009. Paper presented at the 12th IEEE International Conference on Computational Science and Engineering, Vancouver, Canada, August 29-31.

Fonseca, Frederico T., Max J. Egenhofer, Peggy Agouris, and Gilberto Câmara. 2002. Using ontologies for integrated geographic information systems. *T. GIS* 6(3): 231-57.

Gartner. Gartner Says Worldwide Mobile Phone Sales to End Users Grew 8 Per Cent in Fourth Quarter 2009; Market Remained Flat in 2009. http://www.gartner.com/it/page.jsp?id=1306513 (accessed 9/1/2011).

GeoNames. GeoNames. http://www.geonames.org (accessed July 1, 2010).

Gligorov, R., W. Kate, Z. Aleksovski, and F. Harmelen. Using Google distance to weight

approximate ontology matches. 2007. Paper presented at the 16th International World Wide Web Conference, Calgary, Canada, May 8-12.

Google. Android Home Page. http://www.android.com (accessed 9/1/2011).

Google. Google Maps Geocoding API V2. http://code.google.com/apis/maps/documentation/geocoding/v2/index.html (accessed July 1, 2010).

GSL Team. GNU Scientific Library Reference Manual. http://www.gnu.org/software/gsl/manual/html_node/Singular-Value-Decomposition.html (accessed June 1, 2010).

Haeri, Seyed H., Hassan Abolhassani, Vahed Qazvinian, and Babak Bagheri Hariri. 2007. Coincidence-based scoring of mappings in ontology alignment. *Journal of Advanced Computational Intelligence and Intelligent Informatics*. 11(7): 803-16.

Harrington, Brian. Managing uncertainty, importance and differing world views in ASKNet semantic networks. 2010. Paper presented at the 4th IEEE International Conference on Semantic Computing, Pittsburgh, PA, USA, September 22-24.

Hu, Wei, and Yuzhong Qu. Block matching for ontologies. 2006. Paper presented at the 5th International Semantic Web Conference, Athens, Georgia, USA, November 5-11.

Isaac, A., L. van der Meij, S. Schlobach, and S. Wang. An empirical study of instance-based ontology matching. 2007. Paper presented at the 6th International Semantic Web Conference, Busan, Korea, November 11-15.

Janowicz, Krzysztof, and Marc Wilkes. SIM-DLA: A novel semantic similarity measure for description logics reducing inter-concept to inter-instance similarity. 2009. Presented at the 6th European Semantic Web Conference, Heraklion, Greece, May 31-June 4.

Janowicz, Krzysztof, Martin Raubal, Angela Schwering, and Werner Kuhn. 2008. Semantic similarity measurement and geospatial applications. *T. GIS* 12(6): 651-59.

Jin, Y., L. Khan, L. Wang, and M. Awad. Image annotations by combining multiple evidence & WordNet. 2005. Paper presented at ACM Multimedia, Hilton, Singapore, November 6-11.

Joshi, Dhiraj, and Jiebo Luo. Inferring generic activities and events from image content and bags of geo-tags. 2008. Paper presented at the 7th ACM International Conference on Image and Video Retrieval, Niagra Falls, NY, USA, July 7-9.

Kang, J., and J.F. Naughton. Schema matching with opaque column names and data values. 2003. Paper presented at the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, CA, USA, June 9-12.

Karalopoulos, Athanasios, Margarita Kokla, and Marinos Kavouras. Comparing representations of geographic knowledge expressed as conceptual graphs. 2005. Paper presented at the 1st International Conference on Geospatial Semantics, Mexico City, Mexico, November 29-30.

Klien, E., Udo Einspanier, Michael Lutz, and Sebastian Hübner. An architecture for ontology-based discovery and Retrieval of Geographic Information. 2004. Paper presented at 7th Conference on Geographic Information Science, Heraklion, Greece, April 29-May 1.

Kuhn, Werner. 2005. Geospatial semantics: why, of what, and how? *J. Data Semantics* 3: 1-24.

Kumar, Vipin, Michael Steinbach, and Pang-Ning Tan. 2006. *Introduction to Data Mining*. New York: Pearson Education, Inc.

Lee, Daniel D., and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*. 23: 556–62.

Li, M., M. Ng, Y. Cheung, and J. Huang. 2008. Agglomerative fuzzy k-means clustering algorithm with selection of number of clusters. *IEEE Trans. Knowl. Data Eng*. 20: 1519-34.

Li, Jingzhou, and Günther Ruhe. 2008. Software effort estimation by analogy using attribute selection based on rough set analysis. *International Journal of Software Engineering and Knowledge Engineering* 18(1): 1-23.

Li, W.S., and C. Clifon. 2000. Semint: a tool for identifying attribute correspondence in heterogeneous databases using neural networks. *J. Data Knowl. Eng.* 33(1): 49-84.

Lin, Dekang. An information-theoretic definition of similarity. 1998. Paper presented at the 15th International Conference on Machine Learning, Madison, WI, USA, July 24-27.

Luiz, André P., Leme Paes, Marco A. Casanova, Karin Breitman Koogan, and Antonio L. Furtado. Instance-based OWL schema matching. 2009. Paper presented at International Conference on Enterprise Information Systems, Milan, Italy, May 6-10.

Madhavan, Jayant, Phillip A. Bernstein, and Erhard Rahm. Generic schema matching with Cupid. 2001. Paper presented at Paper presented at the 27th International Conference on Very Large Databases, Roma, Italy, September 11-14.

Maedche, A., and S. Staab. Measuring similarity between ontologies. Paper presented at Knowledge Engineering and Knowledge Management. 2002. Ontologies and the Semantic Web, 13th International Conference, Siguenza, Spain, October 1-4.

Manolopoulos, Yannis. Binomial coefficient computation: recursion or iteration? 2002. Paper presented at the ACM Special Interest Group on Computer Science Education, Cincinnati, OH, USA, February 27-March 3.

Martineau, Justin, Tim Finin, Anupam Joshi, and Shamit Patel. Improving binary classification on text problems using differential word features. 2009. Paper presented at the 18th ACM Conference on Information and Knowledge Management, Hong Kong, China, November 2-6.

Masud, Mohammad M., Jing Gao, Latifur Khan, Jiawei Han, and Bhavani M. Thuraisingham. A practical approach to classify evolving data streams: training with limited amount of labeled data. 2008. Paper presented at IEEE International Conference on Data Mining, Pisa, Italy, December 15-19.

Mazuel, L., and N. Sabouret. Semantic relatedness measure using object properties in an ontology. 2008. Paper presented at the 7th International Semantic Web Conference, Karlsrhue, Germany, October 26-30.

Melnik, S., H. Garcia-Molina, and E. Rahm. Similarity flooding: a versatile graph matching algorithm and its application to schema matching. 2010. Paper presented at the 26th IEEE International Conference on Data Engineering, Long Beach, CA, USA, March 1-6.

MercoPress. Two Billion Internet Users Worldwide and Mobile Phone Users Increases. http://en.mercopress.com/2011/01/27/two-billion-internet-users-worldwide-and-mobile-phone-users-increases (accessed 9/1/2011).

Nauman, Mohammad, Sohail Khan, and Xinwen Zhang. Apex: extending Android permission model and enforcement with user-defined runtime constraints. 2010. Paper presented at the 5th ACM Symposium on Information, Computer, and Communications Security, Beijing, China, April 13-16.

Newsam, Shawn, and Yi Yang. Integrating gazetteers and remote sensed imagery. 2008. Paper presented at the 16th ACM International Workshop on Geographic Information Systems, Irvine, CA, USA, November 5-7.

Nosofsky, R.M. 1986. Attention, similarity, and the identification-categorization relationship. *Journal of Experimental Psychology* 115: 39-57.

Noy, N. F. and M. A. Musen. 2003. The PROMPT suite: interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies* 59: 983-1024.

Ongtang, Machigar, Kevin R. B. Butler, and Patrick Drew McDaniel. Porscha: policy oriented secure content handling in Android. 2010. Paper presented at the Annual Computer Security Applications Conference, Orlando, FL, USA, December 5-9.

Open Geospatial Consortium. http://www.opengeospatial.org/ (accessed August 1, 2010)

Partyka, Jeffrey, Latifur Khan, and Bhavani M. Thuraisingham. Content-based geospatial schema matching using semi-supervised geosemantic clustering and hierarchy. 2011. Paper presented at the 5th International Conference of Semantic Computing, Palo Alto, CA, USA, September 18-21.

Partyka, Jeffrey, Pallabi Parveen, Latifur Khan, Bhavani M. Thuraisingham, and Shashi Shekhar. 2010. Enhanced geographically-typed semantic schema matching. *Journal of Web Semantics* 9: 52-70.

Partyka, Jeffrey, Latifur Khan, and Bhavani M. Thuraisingham. Geographically-typed semantic schema matching. 2009. Paper presented at the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information, Seattle, Washington, USA, November 4-6.

Partyka, Jeffrey, Latifur Khan, and Bhavani M. Thuraisingham. Semantic schema matching without shared instances. 2009. Paper presented at the 3rd IEEE International Conference on Semantic Computing, Berkeley, CA, USA, September 14-16.

Partyka, Jeffrey, Neda Alipanah, Latifur Khan, Bhavani M. Thuraisingham, and Shashi Shekhar. Content-based ontology matching for GIS datasets. 2008. Paper presented at the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information, Laguna Beach, CA, USA, November 5-7.

Partyka, Jeffrey, Neda Alipanah, Latifur Khan, Bhavani M. Thuraisingham, and Shashi Shekhar. Ontology alignment using multiple contexts. 2008. Paper presented at the 7th International Semantic Web Conference, Karlsruhe, Germany, October 26-30.

Pei, Jin, Hong, Jun, and David A. Bell. A novel clustering-based approach to schema matching. Paper presented at Advances in Information Systems, Izmir, Turkey, October 18-20.

Luo, Ping, Hui Xiong, Guoxing Zhan, Junjie Wu, and Zhongzhi Shi. 2009. Information-theoretic distance measures for clustering validation: generalization and normalization. *IEEE Trans. Knowl. Data Eng.* 21(9): 1249-1262.

Portio Research. Mobile Messaging Futures 2010-2014: Analysis and Growth Forecsts for Mobile Messaging Markets Worldwide. http://www.portioresearch.com/MMF10-14.html (accessed 9/1/2011).

Pouliquen, Bruno, Ralf Steinberger, Camelia Ignat, and Tom De Groeve. Geographical information recognition and visualization in texts written in various languages. 2004. Paper presented at The 19th Annual ACM Symposium on Applied Computing, Nicosia, Cyprus, March 14-17.

Punera, Kunal, and Joydeep, Ghosh: Enhanced hierarchical classification via isotonic smoothing.

2008. Paper presented at the 17th International World Wide Web Conference, Beijing, China, April 21-25.

Ralun, E., and P.A. Bernstein. A survey of approaches to automatic schema matching. 2001. Paper presented at the 27th International Conference on Very Large Data Bases, Rome, Italy, September 11-14.

Rinner, C. Multi-criteria evaluation in support of emergency response decision making. 2007. Paper presented at the Joint CIG/ISPRS Conference on Geomatics for Disaster and Risk Management, Toronto, Ontario, Canada, May 23-25.

Rodríguez, M. Andrea, and Max J. Egenhofer. 2003. Determining semantic similarity among entity classes from different ontologies. *IEEE Trans. Knowl. Data Eng.* 15(2): 442-56.

Seco, N., T. Veale, and J. Hayes. An intrinsic information content metric for semantic similarity in WordNet. 2004. Paper presented at the 16th European Conference on Artificial Intelligence, Valencia, Spain, August 23-27.

Su, Weifeng, Jiying Wang, and Frederick H. Lochovsky. Holistic schema matching for web query interfaces. 2006. Paper presented at the 10th International Conference on Extending Database Technology, Munich, Germany, March 26-31.

Su, Weifeng, Jiying Wang, Qiong Huang, and Frederick H. Lochovsky. Query result ranking over e-commerce web databases. 2006. Paper presented at 15th ACM Conference on Information and Knowledge Management, Arlington, VA, USA, November 6-11.

Sunna, William and Isabel Cruz. Structure-based methods to enhance geospatial ontology alignment. 2007. Paper presented at the 2nd International Conference on Geospatial Semantics, Mexico City, Mexico, November 29-30.

Sweeney, Latanya. 2002. k-Anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10(5): 557-70.

Thor, Andreas, Toralf Kirsten, and Erhard Rahm. Instance-based matching of hierarchical ontologies. 2007. Paper presented at the 12th GI-Fachtagung für Datenbanksysteme in Business, Technologie und Web, Aachen, Germany, March 7-9.

Tversky, A. 1977. Features of similarity. *Psychological Review* 84(4): 327-52.

USGS US Board on Geographic Names. BGN: Domestic and Antarctic Names - State and Topical Gazetteer Download Files. http://geonames.usgs.gov/domestic/download_data.htm (accessed December 1, 2010)

Vinyals, Oriol, Gerald Friedland, and Nelson Morgan. Discriminative training for hierarchical clustering in speaker diarization. 2010. Paper presented at INTERSPEECH, Makuhari, Japan, September 26-30.

Wang, Shenghui, Gwenn Englebienne, and Stefan Schlobach. Learning concept mappings from instance similarity. 2008. Paper presented at the 7th International Semantic Web Conference, Karlsrhue, Germany, October 25-29.

Wang, Jiying, Ji-Rong Wen, Frederick H. Lochovsky, and Wei-Ying Ma. Instance-based schema matching for web databases by domain-specific query probing. 2004. Paper presented at the 30th International Conference on Very Large Databases, Toronto, Canada, August 29-September 3.

Warren, R.H., and F.W. Tompa. Multi-column substring matching for database schema translation. 2006. Paper presented at the 32nd International Conference on Very Large Data Bases, Seoul, South Korea, September 12-15.

Wartena, Christian, and Rogier Brussee. Instance-based mapping between thesauri and folksonomies. 2008. Paper presented at the 7th International Semantic Web Conference, Karlsrhue, Germany, October 26-30.

Waze. Free GPS Navigation with Turn by Turn – Waze. http://www.waze.com/ (accessed 9/1/2011).

Wilde, Eric, and Martin Kofahl. The locative web. 2008. Paper presented at the 1st International Workshop on Location and the Web, Beijing, China, April 22.

Wilkes, M., and K. Janowicz. A graph-based alignment approach to similarity between Climbing Routes. 2008. Paper presented at the 1st International Workshop on Information Semantics and its Implications for Geographic Analysis at GIScience, Park City, UT, September 23-26.

Wu, Qingyao, Yunming Ye, Michael K. Ng, Hanjing Su, and Joshua Zhexue Huang. Exploiting word cluster information for unsupervised feature selection. 2010. 11th Pacific Rim International Conference on Artificial Intelligence, Daegu, South Korea, August 30-September 2.

Xu, ChengZhi, Peng Liang, Taehyung(George) Wang, Qi Wang, and Phillip C.Y. Sheu. Semantic web services annotation and composition based on ER model. 2010. Paper presented at the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, SUTC 2010, and IEEE International Workshop on Ubiquitous and Mobile Computing, UMC 2010, Newport Beach, CA, USA, June 7-9.

Yan, L. L., R.J. Miller, L.M. Haas, and R. Fagin. Data-driven understanding and refinement of schema mappings. 2001. Paper presented at ACM SIGMOD International Conference on Management of Data 2001 in Santa Barbara, CA, USA.

Zhao, Na, Shu-Ching Chen, Mei-Ling Shyu, and Stuart Harvey Rubin. An integrated and interactive video retrieval framework with hierarchical learning models and semantic clustering strategy. 2006. Paper presented at the IEEE International Conference on Information Reuse and Integration, Waikoloa, HI, USA, September 16-18.

Zhou, Changqing, Dan Frankowski, Pamela J. Ludford, Shashi Shekhar, and Loren G. Terveen. 2007. Discovering personally meaningful places: An interactive clustering approach. *ACM Trans. Inf. Syst.* 25(3).

Zhou, Changqing, Dan Frankowski, Pamela J. Ludford, Shashi Shekhar, and Loren G. Terveen. Discovering personal gazetteers: an interactive clustering approach. 2004. Paper presented at the 12th ACM International Workshop on Geographic Information Systems, Washington, D.C., USA, November 12-13.

Zhuang, Fuzhen, Ping Luo, Hui Xiong, Qing He, Yuhong Xiong, and Zhongzhi Shi. Exploiting associations between word clusters and document classes for cross-domain text categorization. 2010. Paper presented at the 2010 SIAM International Conference on Data Mining, Columbus, OH, USA, April 29-May 1.

VITA

Jeffrey Partyka was born in Edison, New Jersey, on August 7, 1979, to his parents Maria and Carl Partyka. He was raised in Sayreville, New Jersey, not far from Edison, and proceeded to complete grades K-12, finally graduating from Sayreville War Memorial High School in June of 1997. Later that year he matriculated into The College of New Jersey in Ewing, New Jersey. In December of 2001, he was awarded a Bachelor of Science degree in Computer Science. From 2002 up to 2007, he worked in three different industries in an effort to launch a career as a Web developer and software developer. His first job was at Merck World Headquarters at Whitehouse Station, New Jersey from October of 2002 to November of 2003 as a Web Developer. He next worked as a Web Developer for a small legal newspaper in South Plainfield, New Jersey called New Jersey Lawyer, Inc., from January of 2004 to June of 2006. He then worked as a software developer at a medical marketing company known as Medical Marketing Studies in Scotch Plains, New Jersey, from June 2006 to May of 2007. He enrolled in The University of Texas at Dallas as a recipient of the Jonsson Distinguished Scholarship in August of 2007 and as an aspiring Ph.D. candidate under the tutelage of Dr. Latifur Khan. He earned his Master's Degree in Computer Science in December of 2009, and is expecting his Ph.D. in Computer Science in December 2011. Jeff is currently a software developer for Bank of America Merrill Lynch in Frisco, Texas. Once he receives his Ph.D., he plans to join Raytheon as a researcher in an Analytics group led by Donald Kretz and to continue exploring novel research ideas towards the strategic and economic advancement of the United States of America.

SELECTED PUBLICATIONS

[1] Partyka, Jeffrey, Latifur Khan, and Bhavani M. Thuraisingham. Content-based geospatial schema matching using semi-supervised geosemantic clustering and hierarchy. 2011. Paper presented at the 5th International Conference of Semantic Computing, Palo Alto, CA, USA, September 18-21.

[2] Partyka, Jeffrey, Pallabi Parveen, Latifur Khan, Bhavani M. Thuraisingham, and Shashi Shekhar. 2010. Enhanced geographically-typed semantic schema matching. *Journal of Web Semantics* 9: 52-70.

[3] Partyka, Jeffrey, Latifur Khan, and Bhavani M. Thuraisingham. Geographically-typed semantic schema matching. 2009. Paper presented at the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information, Seattle, Washington, USA, November 4-6.

[4] Partyka, Jeffrey, Latifur Khan, and Bhavani M. Thuraisingham. Semantic schema matching without shared instances. 2009. Paper presented at the 3rd IEEE International Conference on Semantic Computing, Berkeley, CA, USA, September 14-16.

[5] Partyka, Jeffrey, Neda Alipanah, Latifur Khan, Bhavani M. Thuraisingham, and Shashi Shekhar. Ontology alignment using multiple contexts. 2008. Paper presented at the 7th International Semantic Web Conference, Karlsruhe, Germany, October 26-30.

[6] Partyka, Jeffrey, Neda Alipanah, Latifur Khan, Bhavani M. Thuraisingham, and Shashi Shekhar. Content-based ontology matching for GIS datasets. 2008. Paper presented at the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information, Laguna Beach, CA, USA.