USING LINGUISTIC TECHNIQUES FOR SCHEMA MATCHING

Ozgul Unal, Hamideh Afsarmanesh
Informatics Institutes, University of Amsterdam, Kruislaan 403, Amsterdam, The Netherlands

Keywords: Schema Matching, Linguistic Matching, Collaborative Networks, Biodiversity.

Abstract: In order to deal with the problem of semantic and schematic heterogeneity in collaborative networks, matching components among database schemas need to be identified and heterogeneity needs to be resolved, by creating the corresponding mappings in a process called schema matching. One important step in this process is the identification of the syntactic and semantic similarity among elements from different schemas, usually referred to as Linguistic Matching. The Linguistic Matching component of a schema matching and integration system, called SASMINT, is the focus of this paper. Unlike other systems, which typically utilize only a limited number of similarity metrics, SASMINT makes an effective use of NLP techniques for the Linguistic Matching and proposes a weighted usage of several syntactic and semantic similarity metrics. Since it is not easy for the user to determine the weights, SASMINT provides a component called Sampler as another novelty, to support automatic generation of weights.

1 INTRODUCTION

Collaboration has drawn considerable attention in recent years and as a result, a large variety of collaborative networks have emerged, such as virtual organizations, supply chains, collaborative virtual laboratories, etc. (Camarinha-Matos and Afsarmanesh, 2005). As identified in the ENBI project (ENBI (2005)), need for collaboration as well as the mechanisms and infrastructures supporting advanced collaborations among pre-existing, heterogeneous, and autonomous organizations have been established in biodiversity domain also. One prominent requirement in a collaborative network is to share data with other organizations. Advances in the Internet technologies have made it possible to set up a high-speed infrastructure for sharing large amounts of data over long distances. However, it is still a big challenge to automatically resolve syntactic, semantic, and structural heterogeneity for providing integrated access to and data sharing among heterogeneous, autonomous, and distributed databases in biodiversity or in any other domains.

Therefore, an infrastructure supporting collaborative networks needs to consider this challenge to provide interoperability. In most previous approaches aiming to enable interoperability among databases, schema matching is performed manually, which is an error-prone and time consuming task. However, the SASMINT (Semi-Automatic Schema Matching and INTegration) system (Unal and Afsarmanesh, 2006), introduced as a part of the Collaborative Information Management System (CIMS) proposed in the ENBI project for the biodiversity domain, identifies the syntactic, semantic, and structural similarities automatically, and expects the user to accept, change, or reject the results. Furthermore, if there is a need to produce an integrated schema of the domain, it automatically generates the integrated schema using the results of schema matching.

The main processing steps of the SASMINT are shown in Figure 1. The steps, which are the subject of this paper are shown in bold. Since the focus of this paper is the Linguistic Matching, we will not go into the details of other steps. However, to give a general overview of the whole system, SASMINT consists of Schema Matching and Schema Integration components. Schema Matching involves the following main steps: 1) Preparation, which translates database schemas into the common Directed Acyclic Graph (DAG) format, 2) Comparison, which identifies the correspondences between the two schemas represented as DAGs, resolves the conflicts, and finds out the matches, using both Linguistic and Structure Matching, 3) Result Generation and Validation, in which matches are displayed to the end-user in order to make any necessary modifications and to save or reject the final result.

Unal O. and Afsarmanesh H. (2006).
USING LINGUISTIC TECHNIQUES FOR SCHEMA MATCHING.
In *Proceedings of the First International Conference on Software and Data Technologies*, pages 115-120

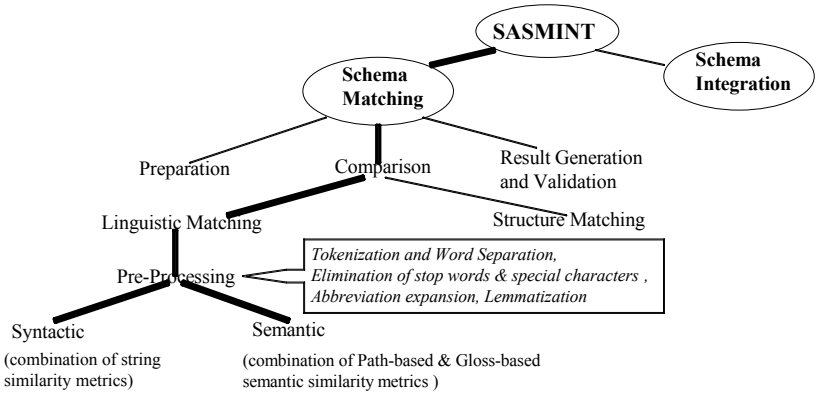

Figure 1: Processing Steps of SASMINT.

The rest of the paper is organized as follows. Section 2 is dedicated to the review of related work. Section 3 presents the details of similarity metrics used in the linguistic matching as well as the results of tests that we performed for the evaluation of the metrics. Section 4 briefly explains the Sampler component. Finally, Section 5 summarizes the main conclusions of this paper.

2 RELATED WORK

Schema matching has been the subject of a large number of efforts in the database research domain. However, these efforts usually require a large amount of manual work and are limited in their provided solutions. Furthermore, they provide insufficient functionalities for normalization of strings, referred to as pre-processing. Regarding the linguistic matching, they do not use the linguistic techniques effectively, but rather they typically use only one string similarity metric. Using a single similarity metric is not enough especially considering the fact that schemas usually contain element names with different characteristics and some metrics give more accurate results for certain types of strings.

Being the most similar system to ours, the COMA system (Do and Rahm, 2002) provides a library of matchers. However, it does not support the pre-processing of elements' names. Cupid (Madhavan, Bernstein et al., 2001) on the other hand, involves a normalization step. However, for the linguistic matching, Cupid exploits only a simple name matcher to identify syntactic similarities and nothing for determining semantic similarity. Similarity Flooding (Melnik, Garcia-Molina et al., 2002) identifies the matching between elements of two schemas represented as graphs using a simple string matcher and again no semantic similarity metric is used. The ONION (Mitra, Wiederhold et al., 2001) system uses a number of heuristic matchers in the matching process. However, it does not employ any combination of syntactic and semantic similarity metrics. GLUE (Doan, Madhavan et al., 2002) provides a name matcher and several instance-level matchers. However, compared to SASMINT, it requires a large amount of manual effort because ontologies need to be first mapped manually to train learners. Clio (Miller, Haas et al., 2000) requires the user to define the value correspondences, thus there is no use of any linguistic matching techniques and a lot of manual work is required.

In summary, the linguistic matching step of the schema matching process has not gained as much importance as the structural matching step. However, since the results of linguistic matching are used by the structure matching of schema matching, in our opinion it is imperative to obtain as accurate results as possible at this step. Therefore, using a combination of several syntactic and semantic similarity metrics has become one of the main goals of SASMINT.

3 LINGUISTIC MATCHING

The Linguistic Matching in SASMINT has two main goals: identifying both the syntactic and semantic similarity between element pairs. A combination of string similarity metrics are utilized to determine syntactic similarity, while semantic similarity algorithm uses the WordNet and considers a variety of relationships between the terms, such as synonymy and hyponymy, as well as the gloss information as explained in Section 3.3. Among different alternatives, a number of metrics, widely used in NLP and suitable for different string types,

are implemented in Java for the Linguistic Matching.

Before applying any linguistic matching algorithm, element names from two schemas need to be pre-processed to bring them into a common representation. Below, the main processes of the pre-processing step are introduced:

1. ***Tokenization and Word Separation****:* Strings containing multiple words are split into lists of words or tokens.
2. ***Elimination of stop words:*** Stop words are common words such as prepositions, adjectives, and adverbs, e.g. 'of', 'the', etc., that occur frequently but do not have much effect on the meaning of strings.
3. ***Elimination of special characters and De-hyphenation:*** The characters such as '/', '-', etc., are considered to be useless and removed from the names.
4. ***Abbreviation expansion:*** Since the abbreviations are mostly used in names, they need to be identified and expanded. For this purpose, a dictionary of well-known abbreviations is used.
5. ***Normalizing terms to a standard form using Lemmatization:*** Multiple forms of the same word need to be brought into a common base form. By means of lemmatization, verb forms are reduced to the infinitive and plural nouns are converted to their singular forms.

After pre-processing element names, a variety of algorithms (metrics) are applied to identify syntactic and semantic similarities, as detailed below.

3.1 Syntactic Similarity

There has been a lot of past research work in Natural Language Processing (NLP) on comparing two character strings syntactically. SASMINT uses a combination of several main syntactic similarity metrics. Since each of these metrics is best suited for a different type of strings, we find it more appropriate for SASMINT to use several of them together, to make it a more generic tool to be used for matching schemas with different types of element names. These metrics are briefly explained below:

1. *Levenshtein Distance (Edit Distance):* Levenshtein Distance (Levenshtein, 1966), also known as Edit Distance, is based on the idea of minimum number of modifications required to change a string into another. The modification can be of type changing, deleting, or inserting a character. Levenshtein distance is a string-based distance metric, meaning that it does not consider the multi-word string as a combination of tokens but rather as a single string.
2. *Monge-Elkan Distance:* Monge and Elkan (Monge and Elkan, 1996) proposed another string-based distance function using an affine gap model. Monge-Elkan Distance allows for gaps of unmatched characters. Affine gap costs are specified in two ways: one with a cost for starting a gap and a second for the cost of continuation of the gap.
3. *Jaro* (Jaro, 1995), a string-based metric well known in the record linkage community, is intended for short strings and considers insertions, deletions, and transpositions. It also takes into account typical spelling deviations.
4. *TF*IDF (Term Frequency*Inverse Document Frequency)* (Salton and Yang, 1973) is a vector-based approach from the information retrieval literature that assigns weights to terms. For each of the documents to be compared, first a weighted term vector is composed. Then, the similarity between the documents is computed as the cosine between their weighted term vectors.
5. *Jaccard Similarity* (Jaccard, 1912) is a token-based similarity measure yielding a similarity value between 0 and 1. The Jaccard similarity between two strings A and B consisting of one or more words is defined as the ratio of the number of shared words of A and B to the number owned by A or B.
6. *Longest Common Substring (LCS):* is a special case of edit distance. The longest common substring of A and B is the longest run of characters that appear in order inside both A and B.

Syntactic Similarity Metrics Used in SASMINT

All the metrics described above are used in the Linguistic Matching component of our system. Considering that schemas usually consist of mixed set of element names (strings) with different characteristics and each metric might be suitable for different types of strings, we propose to use a weighted sum of the metrics as defined below, which yields better results.

$$sim_W(a,b) = w_{lv} * sm_{lv}(a,b) + w_{me} * sm_{me}(a,b) + w_{jr} * sm_{jr}(a,b) + w_{jc} * sm_{jc}(a,b) + w_{tf} * sm_{tf}(a,b) + w_{lc} * sm_{lc}(a,b)$$

where 'lv' stands for Levenstein, 'me' for Monge-Elkan, 'jr' for Jaro, 'jc' for Jaccard, 'tf' for TF-IDF, and 'lc' for Longest Common Sub-string.

Using a weighted sum of several metrics is one key innovation of our system, as depending on the characteristics of the element names, some metrics can be assigned higher weights. In addition to being

applicable for different string types, SASMINT system also proposes a recursive weighted metric as an alternative to the weighted metric, which is a modified version of Monge-Elkan's hybrid metric (Monge and Elkan, 1996) and better supports the matching of schema names when they contain more than one token. Our metric is different from that of Monge-Elkan in that ours applies more than one similarity metric to string pairs, while Monge-Elkan uses only one metric. Furthermore, unlike Monge-Elkan, it is symmetric generating the same result for sim(a,b) and sim(b,a), and thus making our pair matching associative. Given two strings a and b that are tokenized into $a = s_1, s_2, ..s_l$ and $b = t_1, t_2, ...t_m$, the recursive weighted metric is as follows:

$$sim(a,b) = \frac{1}{2l}\sum_{i=1}^{l}\max_{j=1}^{m} sim_W(a_i,b_j) + \frac{1}{2m}\sum_{j=1}^{m}\max_{i=1}^{l} sim_W(a_i,b_j)$$

3.2 Evaluation of Syntactic Similarity

In order to determine the accuracy of our syntactical similarity step, we carried out a number of tests. Since our aim was to evaluate only the accuracy, we did not consider any performance issues. We borrowed the test data from Similarity Flooding (Melnik, Garcia-Molina et al., 2002) and Clio (Miller, Haas et al., 2000). In our tests, similarities were calculated in three ways: 1) Using the metrics individually, *2)* using the SASMINT's weighted sum of the individual metrics, and *3)* using the SASMINT's modified recursive weighted metric. For calculating the weighted sum, we used the same weight for all similarity metrics.

All possible pairs of element names from the two schemas were compared using the metrics and each pair was assigned a similarity score between 0 and 1. If the similarity score was above the threshold, then the match was accepted. We performed the tests with the threshold value of 0.5. In the evaluation process, we used the concepts of precision and recall from the information retrieval field (Cleverdon and Keen, 1966). Precision (P) and Recall (R) are computed as follows:

$$P = \frac{x}{x+z} \text{ and } R = \frac{x}{x+y}$$

where x is the number of correctly identified similar strings, z is the number of strings found as similar, which are actually not similar (called as false positives), and y is the number of similar strings, which could not be identified by the system (called as false negatives).

Although precision and recall measures are widely used for variety of evaluation purposes, neither of them can accurately assess the match quality alone. Therefore, a measure combining precision and recall is needed. F-measure (Rijsbergen, 1979) is one such a measure, combining recall and precision as follows:

$$F = \frac{2}{\frac{1}{P}+\frac{1}{R}}$$

F-measure forms the base of our evaluation process. Although in Figures 2-7, the values of precision and recall are also shown, for the purpose of the assessment of the similarity metrics, values of F-measure are considered. The figures below demonstrate that while one metric among the six may perform well on one data set, it may not on another. This is due to the fact that different schema test sets from different domains contain elements with different characteristics, while at the same time each of the six metrics is usually suitable for specific type(s) of strings. However, SASMINT's approach is more generic, and in all our similarity evaluation tests we observed that the weighted sum of metrics consistently performed almost as good as the best metric among the six. For different types of schemas, SASMINT yields better results than other systems, as it uses a combination of metrics suitable for different element names.

3.3 Semantic Similarity

Semantic similarity algorithms used in SASMINT can be categorized into two, using the names of groups mentioned in (Pedersen, Banerjee et al., 2003): 1) path based measures and 2) gloss-based measures, which are briefly explained below.

1. Path-based Measures: Path-based measures are based on the idea of calculating the shortest path between the concepts in a IS-A hierarchy, such as the WordNet (Fellbaum, 1998), which is a lexical dictionary mostly used by similarity measures. Among different alternatives in this category, we use the measure of Wu and Palmer (Wu and Palmer, 1994). This measure focuses on verbs, though it is applicable for nouns also, and takes into account the lowest common subsumer of the concepts.

2. *Gloss-based Measures:* In this category of semantic similarity measures, gloss overlaps are used. *Gloss* refers to a brief description of a word. Lesk (Lesk, 1986) uses gloss overlaps for word sense disambiguation. We convert the algorithm of Lesk to compute the semantic similarity of two concepts c_1 and c_2 as follows: for each of the senses of c_1, we compute the number of common words between its glosses and the glosses of each of the senses of c_2.

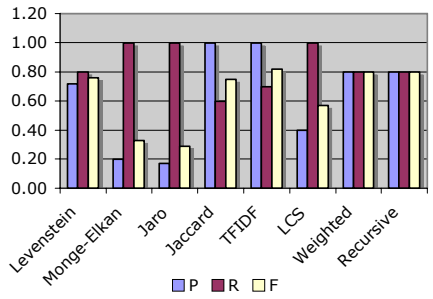

Figure 2: Test Data #1.

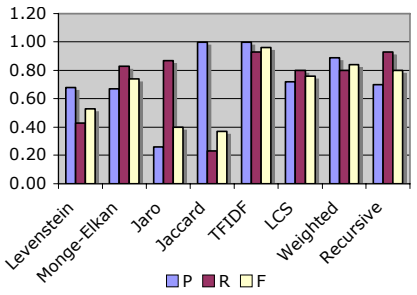

Figure 3: Test Data #2.

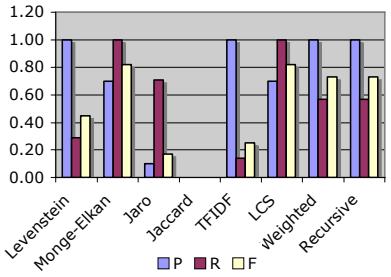

Figure 4: Test Data #3.

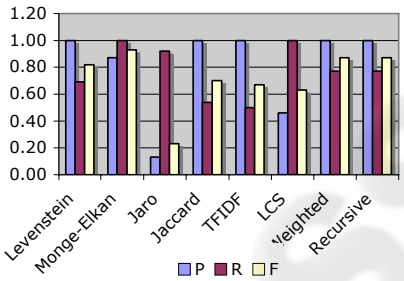

Figure 5: Test Data #4.

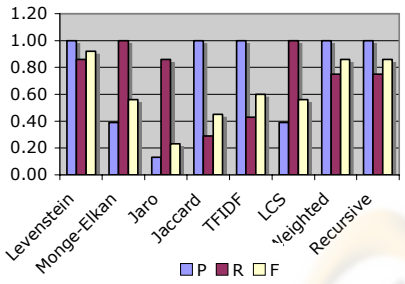

Figure 6: Test Data #5.

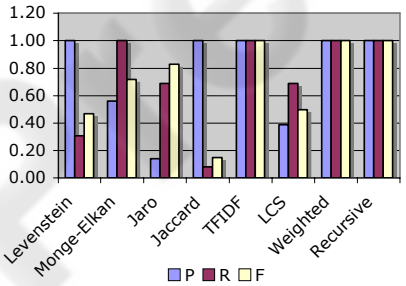

Figure 7: Test Data #6.

Semantic Similarity Metrics Used in SASMINT

Both of the semantic similarity measures mentioned above are used in the SASMINT system for identifying the semantic similarity of element names. These measures utilize the WordNet. Resulting semantic similarity is the weighted sum of path-based and gloss-based measures.

4 IDENTIFYING THE WEIGHTS AUTOMATICALLY

For schema matching among heterogeneous and autonomous sources of information, a variety of techniques can be applied. Nevertheless, the conclusion of our studies shows that depending on the characteristics of the strings in the domain, some metrics are more suitable for matching certain types of strings. Therefore, to produce more accurate results, we suggest assigning weights to the metrics and giving higher weights to those metrics that are more suitable. Although using a weighted sum of different metrics gives more accurate results in syntactic and semantic matching steps, it is not always easy to manually determine the weights. Furthermore, for some cases, it might not be appropriate to use equal weights for each of the metrics.

As another contribution of the system, SASMINT proposes a component called 'Sampler' to help the user with identifying the most suitable weights. The Sampler component discovers the weights through the following steps: 1) The user is asked to choose between syntactic or semantic matching and then to provide up to ten known sample "similar pairs" (syntactically or semantically similar, depending on which type of matching he

wants to determine the weights for) from his schema domain. 2) Sampler runs each metric over these pairs and determines the similarity for each pair between 0 and 1. 3) It calculates the accuracy of each metric using the F-measure as explained in Section 3.2. 4) Sampler assigns weights for each metric, based on the following formula, where $\sum F$ represents the sum of F-measure values resulted for all metrics used, and F_m represents the F-measure value calculated for metric 'm':

$$w_m = \frac{1}{\sum F} * F_m$$

5) Finally, weights are presented to the user who can accept or modify them. If the user does not run the sampler, then he either will define them, or averaging the metrics (equal weights for all) is the only mechanism that SASMINT can use, although it may not produce desirable results.

5 CONCLUSION

In this paper, we introduce a semi-automatic schema matching and integration tool called SASMINT and explain how it uses linguistic techniques to automatically resolve syntactical and semantic heterogeneities between database schemas. In order to identify the syntactic and semantic similarity between the elements' names from two schemas, unlike other schema matching efforts, the SASMINT system utilizes a combination of different types of string similarity and semantic similarity metrics from NLP. The use of a weighted and recursive weighted sum of these metrics are proposed, giving more accurate results. Furthermore, the Sampler component of SASMINT helps users to influence the weights for applying these metrics. A number of tests were carried out to measure the correctness of the metrics and the results are provided in this paper. Other tests are being planned to compare SASMINT with other similar systems.

REFERENCES

ENBI (2005). European Network for Biodiversity Information (IST 2001-00618). http://www.enbi.info.

Camarinha-Matos, L. M. and H. Afsarmanesh (2005). Collaborative networks: A new scientific discipline. *Journal of Intelligent Manufacturing* 16(4-5): 439-452.

Cleverdon, C. W. and E. M. Keen (1966). Factors determining the performance of indexing systems, vol 2: Test results, Aslib Cranfield Research Project. Cranfield Institute of Technology.

Do, H. H. and E. Rahm (2002). COMA - A System for Flexible Combination of Schema Matching Approaches. In 28th International Conference on Very Large Databases (VLDB).

Doan, A., J. Madhavan, et al. (2002). Learning to Map between Ontologies on the Semantic Web. In World-Wide Web Conf. (WWW-2002).

Fellbaum, C. (1998). An Electronic Lexical Database., Cambridge: MIT press.

Jaccard, P. (1912). The distribution of flora in the alpine zone. The New Phytologist 11(2): 37-50.

Jaro, M. A. (1995). Probabilistic linkage of large public health. Statistics in Medicine: 14:491-498.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: how to tell a pine code from an ice cream cone. In 5th SIGDOC Conference.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory 10(8): 707-710.

Madhavan, J., P. A. Bernstein, et al. (2001). Generic Schema Matching with Cupid. In 27th International Conference on Very Large Databases (VLDB).

Melnik, S., H. Garcia-Molina, et al. (2002). Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching. In 18th International Conference on Data Engineering (ICDE).

Miller, R. J., L. M. Haas, et al. (2000). Schema Mapping as Query Discovery. In 26th International Conference on Very Large Databases (VLDB).

Mitra, P., G. Wiederhold, et al. (2001). A scalable framework for the interoperation of information sources. International Semantic Web Working Symposium.

Monge, A. E. and C. Elkan (1996). The Field Matching Problem: Algorithms and Applications. In 2nd International Conference on Knowledge Discovery and Data Mining.

Pedersen, T., S. Banerjee, et al. (2003). Maximizing Semantic Relatedness to Perform Word Sense Disambiguation. Supercomputing Institute, University of Minnesota.

Rijsbergen, C. J. v. (1979). Information Retrieval, Butterworths, London.

Salton, G. and C. S. Yang (1973). On the specification of term values in automatic indexing. Journal of Documentation(29): 351-372.

Unal, O. and H. Afsarmanesh (2006). Interoperability in Collaborative Network of Biodiversity Organizations. In Proc. of PRO-VE'06 - Virtual Enterprises and Collaborative Networks, Accepted for Publication.

Wu, Z. and M. Palmer (1994). Verb Semantics and Lexical Selection. 32nd Annual Meeting of the Association for Computational Linguistics.