# 54

# INTEROPERABILITY IN COLLABORATIVE NETWORK OF BIODIVERSITY ORGANIZATIONS

Ozgul Unal and Hamideh Afsarmanesh
*University of Amsterdam, THE NETHERLANDS*
*{ozgul, hamideh}@science.uva.nl*

*Schematic and semantic heterogeneity are two important types of heterogeneities that need to be resolved in order to enable interoperability and exchange of data among distributed and heterogeneous databases in a collaborative network of biodiversity nodes. This paper describes the SASMINT system, which performs schema matching and integration among databases. SASMINT identifies syntactic/semantic/structural similarities between two schemas as automatically as possible, resolving their heterogeneity and creating mappings among the pairs of matched components. Unlike other systems that are typically limited to specific algorithms, SASMINT combines a number of algorithms from the NLP and graph theory domains. After obtaining the user-input on validation/enhancement of matching results, SASMINT exploits the results of schema matching to automatically generate an integrated schema.*

## 1. INTRODUCTION

The number of organizations willing and interested to collaborate with others is increasing at a high pace. As a result, different types of collaborative networks have been formed in the last years, such as the supply chains and virtual organizations (Camarinha-Matos and Afsarmanesh, 2005). Although the first examples of collaborative networks come from the manufacturing domain, the need for collaboration has been recently well understood in most areas of science and industry, including the biodiversity domain, which is the main focus of this paper.

Increasing biodiversity conservation activities bring about a variety of new needs for collaborative networks in this domain. For instance, they entail producing more accurate results by comparison and/or merging different biodiversity analysis activities, and making better predictions about the global distribution of species. This, in turn requires the collaboration and data / resource sharing among the biodiversity centers, organizations, and individual researchers. Although importance of collaboration in biodiversity has become clear to most involved scientists, most biodiversity related organizations hesitate to actively cooperate. This is mostly due to the sensitivity of some specific data categories, such as endangered species. Therefore, new mechanisms and infrastructures are needed, supporting collaboration among organizations, while taking these types of criteria into account. With the existence of such a mechanism, organizations can more easily decide to collaborate.

Because of the growing number of heterogeneous databases, interoperability has become one most critical issue that such infrastructures for biodiversity, as well as for other domains, need to consider. Organizations typically design and use different structures for storage and processing of their data depending on their specific needs. Usually, even database schemas for identical concepts in two organizations have structural and naming differences. They might even use similar terms with completely different meanings. Since data sharing constitutes the main type of collaboration, the collaboration infrastructure has to consider such differences for providing effective mechanisms to integrate or inter-link and homogeneously access heterogeneous databases.

Semantic and schematic (structural) database schema heterogeneity are two main types of interoperability problems, where the former refers to differences in the meaning of data, and the latter is related to differences in the modeling and encoding of the concepts. In order to deal with these types of heterogeneity, schema matching and integration approaches have found considerable interest recently. However, most approaches typically require a large amount of manual work. Manual schema matching creates a major bottleneck due to the rapidly increasing number of heterogeneous data sources. As systems become able to handle more complex databases and applications, their schemas become larger, further increasing the number of matches needed to be performed. On the other hand, the suggested automatic resolution of semantic and schematic schema heterogeneity still remains challenging for provision of integrated data access/sharing among autonomous and distributed databases.

In order to address these problems concerning database interoperability in biodiversity domain, a Semi-Automatic Schema Matching and INTegration (SASMINT) system is proposed in the ENBI project (European Network for Biodiversity Information) (ENBI (2005)). The Collaborative Information Management System (CIMS) in ENBI aims at dealing with the information management related problems in biodiversity domain. Unlike other approaches to schema matching and integration, SASMINT requires minimal user input. It combines a number of linguistic and structure matching techniques from Natural Language Processing (NLP) and graph similarity research domains in a flexible way and (semi-) automatically matches the schemas. Then, after user validation/enhancement of matching results, SASMINT produces a new extended integrated schema. If there is an existing common schema in the collaborative network of nodes, it may not be needed to generate an integrated schema and the operation of SASMINT can be stopped after the identification of proper matches among the schema elements.

The rest of this paper is organized as follows. Section 2 provides an overview of related work. Section 3 describes the steps of the approach of SASMINT, addressing its syntactic and semantic similarity matching. Finally, Section 4 summarizes the main conclusions of this paper.

## 2. RELATED WORK

In this section, brief information about a number of related biodiversity projects is provided. Furthermore, as for the schema matching, an overview of main related work from database research domain is given.

There have already been a number of projects from the biodiversity domain that aim at providing data from distributed and heterogeneous databases through a common access system. These projects typically assume that existing biodiversity data standards can be used as the common schema, such as Darwin Core (Darwin Core (2006)) and ABCD Schema (ABCD Schema (2006)), and data from each provider has to comply with this schema when providing it to the outside world. The BioCASE (A Biodiversity Collection Access Service for Europe) project aims at establishing a web-based information service providing researchers with unified access to biological collections in Europe (BioCASE (2006)). In this project, users are expected to manually map the related fields between their local schema and the ABCD Schema, by using a simple user interface. The Global Biodiversity Information Facility (GBIF) (GBIF (2006)) is an international initiative that aims at providing biodiversity data globally and freely available to all users. Similar to the BioCASE, it is again the responsibility of the data provider to provide his data using Darwin Core or ABCD Schema by doing required mappings manually.

It is clear from the examples above that semi-automatic schema matching has not yet been considered in biodiversity projects providing access to distributed and heterogeneous databases. Furthermore, they usually use existing biodiversity data standards as the common schema to represent data from provider nodes. However, existing standards are not extensive enough to represent all types of biodiversity data and in some cases, it is required to generate an integrated schema from local schemas of participating nodes.

In database research domain, the challenge of schema matching to support interoperability has already been addressed by a number of projects. Cupid (Madhavan, Bernstein et al., 2001) normalizes the element names and then exploits a combination of name and structure matcher. However, the normalization step in Cupid is not as comprehensive as our pre-processing step. Moreover, name matching involves a syntactic matching, which employs only one string similarity metric. The COMA system (Do and Rahm, 2002) provides a library of matchers that utilize element and structural properties of schemas. However, it does not support the pre-processing of elements' names. Similarity Flooding (Melnik, Garcia-Molina et al., 2002) converts diverse models into directed labeled graphs and then identifies the initial maps between elements of two graphs using only a simple string matcher. These initial maps are then used by a structure matcher. However, Similarity Flooding has no knowledge of edge and node semantics. Similarly, (Wang, Goguen et al., 2004) borrows the string similarity implementation of Similarity Flooding and thus suffers from the same limitations. The ONION system (Mitra, Wiederhold et al., 2001) uses a number of heuristic matchers, but it does not employ any combination of string similarity metrics. Moreover, it is assumed that the relationships among concepts are defined using a set of relationships with pre-defined semantics, requiring a lot of manual effort. GLUE (Doan, Madhavan et al., 2002) provides a name matcher and several instance-level matchers. It is different from our system in that it uses machine- learning techniques. However, in order to

train learners, ontologies need to be first mapped manually. Clio (Miller, Haas et al., 2000) generates alternative mappings as SQL view definitions based on the value correspondences defined by the user. For this reason, no linguistic matching techniques are used and a large amount manual work is required.

Although the importance of schema matching has been recognized in database interoperability research, previous approaches have some problems. They usually require substantial amounts of manual work and are limited in their solutions. Furthermore, these efforts mostly do not use linguistic techniques, which are needed to increase the overall accuracy of the schema matching system, when used effectively. Another problem is that none of these efforts considers how to use the result of schema matching for semi-automatic schema integration.

## 3. SEMI-AUTOMATIC SCHEMA MATCHING

Different organizations define their schemas differently. These definitions are frequently conflicting and thus making their matching and integration challenging. Among different types of heterogeneities, schematic (both syntactic and structural) and semantic heterogeneity are the most important obstacles to interoperability among databases.

As addressed in the Related Work section, most approaches in literature for resolving schematic and semantic heterogeneity require a large amount of user involvement. Furthermore, in related biodiversity projects aiming at interoperability, such as the BioCASE (BioCASE (2006)) and GBIF (GBIF (2006)), the idea of semi-automatic schema matching has not yet been considered and thus mappings between schemas remain to be identified manually. One key innovation of the approach suggested in SASMINT is that while it identifies the schematic and semantic heterogeneity and finds the "matches" between different schema elements, as automatically as possible, it also proposes to use the result of matching for schema integration. The SASMINT, can be used for two different cases of data sharing in a collaborative network of biodiversity nodes:

1. If a common schema such as ABCD Schema or Darwin Core is used in the network, SASMINT enables semi-automatic matching of the local schema of each node to the common schema.
2. If there is no common schema, SASMINT enables generation of a common integrated schema to represent data from participating biodiversity nodes. It helps to integrate local schemas of nodes by exploiting the result of schema matching.

In this section, the approach of the SASMINT system for semi-automatic schema matching and integration is described. The main processing steps of the SASMINT are shown in Figure 1. The comparison step of the schema matching is detailed, addressing its Structural and Linguistic matching that further involves different syntactic and semantic similarity matching. At the end of Section 3, it is briefly explained how to use the result of schema matching to generate integrated schema.

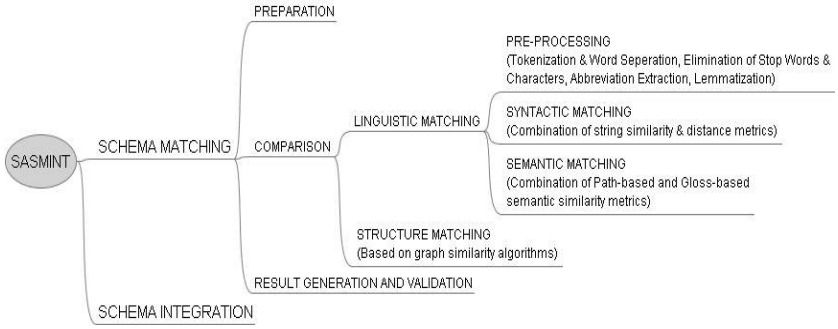


Figure 1- Processing Steps of SASMINT

### 3.1 Schema Matching

The Schema Matching itself consists of three steps: 1) Preparation, 2) Comparison, 3) Result Generation and Validation.

#### *3.1.1 Preparation*

In this step, SASMINT translates source schemas represented in different schema languages into a common Directed Acyclic Graph (DAG) format, which is necessary in order to compare two different types of schemas. The DAG has been chosen as the common format as we believe that it provides a balanced format among other alternatives supporting the representation of a relational schema, an object-oriented schema, etc., as a graph. Furthermore, existing graph theory concepts and algorithms can help us when comparing two graphs.

#### *3.1.2 Comparison*

After representing the schemas as DAGs, SASMINT automatically identifies correspondences between two schemas by resolving the structural as well as syntactic and semantic heterogeneities, referred to as *Comparison*. The comparison step consists of two kinds of matching: Linguistic and Structure Matching, as described below. Result of the Comparison Step is calculated as the weighted sum of Linguistic and Structure Matching. Formally, final similarity $Sim_F(a,b)$ for a pair of schema elements *a* and *b* is calculated by:

$$Sim_F(a,b) = ls_{a,b} * w_{ls} + ss_{a,b} * w_{ss}$$

where $w_{ls}$ is the weight for the linguistic matching, while $w_{ss}$ is the weight for structure matching. The sum of these values is equal to 1.0.

*3.1.2.1 Linguistic Matching*

Linguistic Matching involves both syntactic and semantic matching of element names from two schemas, which results in a linguistic similarity value (*ls*) between 0 and 1 for each possible name pairs. However, before any matching occurs, these names need to be pre-processed in order to bring them into a common

representation. The pre-processing involves the following operations: Strings containing multiple words (or tokens) are split into a list of tokens, for example "First Name" is split into "First" and "Name". Stop words, such as prepositions, adjectives, and adverbs, as well as special characters, such as '/' and '-' are eliminated. Since the abbreviations are mostly used in the names, they need to be expanded, for which SASMINT utilizes a text file containing some well-known abbreviations and their extensions. Finally, 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. SASMINT exploits WordNet (Fellbaum, 1998), a lexical dictionary, in order to find out the lemma's of words.

After pre-processing element names, a variety of algorithms (metrics) are applied to identify syntactic and semantic similarities. Below are the details of these similarity algorithms.

I. ***Syntactic Similarity***

There has been a lot of past research work in Natural Language Processing (NLP) on comparing two character strings syntactically. Unlike other schema matching approaches, which depend on only one metric, 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 SASMINT a more generic tool. The metrics used by SASMINT are 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. The costs of modifications are defined as 1 for each operation.
2. *Monge-Elkan Distance:* Monge and Elkan (Monge and Elkan, 1996) proposed another 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 secondly with a cost for continuation of a gap.
3. *Jaro (Jaro, 1995)*, a 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 document 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) 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 of words 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. Both A and B may have other extraneous characters along the way.

All the metrics described above are implemented within the Linguistic Matching component of SASMINT. Considering that schemas usually consist of mixed set of element names (strings) with different characteristics and each metric may be suitable for different types of strings, we propose to use a weighted sum of the metrics as defined below, which yields better final 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 Levenshtein, 'me' for Monge-Elkan, 'jr' for Jaro, 'jc' for Jaccard, 'tf' for TF-IDF, and 'lc' for Longest Common Sub-string.

In addition to using a weighted sum of several metrics, another improvement of SASMINT over other schema matching systems is that it also utilizes the following new *recursive weighted* metric, which is a modified version of Monge-Elkan's hybrid metric (Monge and Elkan, 1996), to better support the matching of schema names when they contain more than one token. Therefore, the user can choose between the weighted sum metric and the recursive weighted metric based on his/her schema. 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:

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

*Identifying Weights Using the 'Sampler' Component*

If the characteristics of the schema are known by the user in advance, he can modify the weights of metrics accordingly. However, it may not be always easy for the user to carry out this task. For this purpose, as another contribution of the system, a *Sampler* component is proposed in SASMINT, to help the user with identifying the most suitable weights. The sampler operates as follows: The user is asked to provide up to ten known "similar pairs" from his/her schema domain (e.g. "student_name" , "name_of_student"). The sampler *first* runs the six metrics individually over these pairs, and determines their calculated similarities (between 0 and 1) for each pair. *Second*, using the F-measure (Rijsbergen, 1979), which is a combination of the Precision and the Recall methods from the information retrieval field (Cleverdon and Keen, 1966), the sampler calculates the accuracy level (F-measure) for each metric, in relation to these pairs provided by the user. *Third* step of the sampler is that based on the F-measures, it calculates the weight for each metric, higher F-measure meaning higher weight. *Fourth*, after all "weights" for metrics are determined by the sampler component, they are presented to the user, who can accept or modify them. The sampler component can also be used for determining weights for semantic similarity metrics, for which the user needs to provide semantically similar pairs, such as "employee" and "worker".

II. ***Semantic Similarity***

Similar to the syntactical similarity metrics, there are a number of semantic similarity algorithms from the NLP domain. The ones used in SASMINT can be categorized into two, using the names of groups mentioned in (Pedersen, Banerjee et al., 2005): 1) path based measures and 2) gloss-based measures, which are briefly explained below.

1. *Path-Based Measures:* These measures are based on the idea of calculating the shortest path between the concepts being compared in a IS-A hierarchy, such as the WordNet (Fellbaum, 1998). Among different alternatives in this category, we use the measure proposed by Wu and Palmer (Wu and Palmer, 1994). In addition to two concepts being compared in the IS-A hierarchy, this measure also takes into account the lowest common subsume of these concepts.
2. *Gloss-Based Measures: G*loss refers to a brief description of a word. In this category of semantic similarity measure, gloss overlaps are used. In SASMINT, we convert the algorithm of Lesk (Lesk, 1986) 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 $c_2$. A word can have different senses, depending on the context.

Both the measure of Wu and Palmer and the modified version of the measure of Lesk are used in the SASMINT system for determining semantic similarity. These measures utilize the WordNet. Similar to the case in the syntactical similarity, semantic similarity is also calculated as the weighted sum of the algorithms described above. Default value is 0.5 for each of them, but the user can run the sampler functionality of the system to determine the weights.

*3.1.2.2 Structure Matching*

Unlike Linguistic matching, which considers only the element names, structure matching takes into account the structural aspects of schemas also. Structure matching uses the results of Linguistic Matching and applies a number of graph similarity algorithms, which are based on the idea that if two elements have been found to be similar, their adjacent elements (parent and children nodes) may also match. Among different alternatives, following three algorithms were considered to be relevant and chosen for the Structure Matching component of the SASMINT.

1. Graph similarity algorithm proposed in (Blondel, Gajardo et al., 2004) computes the similarity of two graphs $G_A$ and $G_B$ with the vertices $n_A$ and $n_B$ and edges $E_A$ and $E_B$. For $i = 1,..,n_B$ and $j = 1,..,n_A$ the similarity scores are updated iteratively using the following equation:

$$Z_{k+1} = \frac{BZ_kA^T + B^TZ_kA}{\left\|BZ_kA^T + B^TZ_kA\right\|_F} \quad \text{k=0,1,....}$$

where $Z_k$ is the $n_B$ x $n_A$ matrix of entries $z_{ij}$ at iteration k, A and B are the adjacency matrices of $G_A$ and $G_B$, and and $A^T$ and $B^T$ are the transpose of A and B. The matrix norm $\|.\|_F$ used here is known as the Euclidean or Frobenius norm and equals to the square root of the sum of all squared entries. The matrix subsequences $Z_{2k}$ and $Z_{2k+1}$ converge to $Z_{even}$ and $Z_{odd}$.

2. Structure matching of Similarity Flooding (Melnik, Garcia-Molina et al., 2002) is based on a fix point computation. It does not use node or edge semantics and is based on the assumption that whenever any two elements are found to be similar, the similarity of their adjacent elements increases. Over a number of iterations, the initial similarity of any two nodes propagates through the graphs. The algorithm terminates after the similarities of all model elements stabilize.

3. Structure Matching in Cupid exploits a TreeMatch algorithm, which is based on the following perceptions (Madhavan, Bernstein et al., 2001): 1) Atomic elements in the two trees are similar if they are individually similar and if their ancestors and siblings are similar. 2) Two non-leaf elements are similar if they are linguistically similar and the subtrees rooted at the two elements are similar. 3) Two non-leaf schema elements are structurally similar if their leaf sets are highly similar, even if their immediate children are not.

All the above algorithms form the base for the structure matching component of SASMINT. Similar to the method followed in linguistic matching, structure matching uses the weighted sum of the three structural similarity algorithms introduced above resulting in a structural similarity value (*ss*) between 0 and 1 for each possible name pairs.

*3.1.3 Final Result Generation and Validation*

After the correspondences between the graph elements are determined, the resulting matches need to be displayed for the user, both because it is not possible to determine all possible matches automatically and also because not all the identified matches may be correct or desirable. The user can modify the matches and then save (or discard) the final results.

### 3.2 Schema Integration

After schema matching, user has the option to generate an integrated schema of the two schemas being compared. SASMINT aims to facilitate schema integration, by exploiting the validated results of semi-automatic schema matching.

The schema integration component is now under development and ultimately will enable iterative development of a common integrated schema for a collaborative network of nodes, two schemas at a time. This can be achieved as follows: First schemas $S_1$ and $S_2$ of two nodes are chosen and after the schema matching component identifies the mappings and the result is validated by the user, they are integrated by the schema integration component into $S_{\text{int1}}$ and the result is saved. Then, the user selects $S_{\text{int1}}$ and the schema of another node $S_3$ integrating them into $S_{\text{int2}}$. This process continues until the schemas of all nodes are integrated, resulting in a final integrated schema $S_{\text{int}}$. The result of schema integration is stored using a derivation language (Afsarmanesh, Wiedijk et al., 1994).

## 4. CONCLUSION

This paper addresses an important challenge to data sharing in collaborative network of biodiversity organizations: automatic resolution of syntactic/semantic/structural schema heterogeneity. In order to deal with this challenge, SASMINT system is proposed. SASMINT semi-automatically identifies the mappings between the local schemas of nodes and the common schema of the network. It simultaneously uses a number of NLP algorithms that together with the structure matching enable achievement of a more generic schema matching. Furthermore, a sampler tool is

provided for users to influence the weights for these algorithms. If there is no common schema available in the collaborative network, SASMINT also supports the generation of an integrated schema from the schemas of participating nodes. The automatic use of the results produced by schema matching for generation of a new extended integrated schema is a part of the contribution of our work presented in this paper.

## 5. REFERENCES


ABCD Schema (2006). Access to Biological Collections Data (ABCD) Schema, http://bgbm3.bgbm.fu-berlin.de/TDWG/CODATA/Schema/default.htm.

BioCASE (2006). Biological Collection Access Services (BioCASE), http://www.biocase.org.

Darwin Core (2006). Darwin Core, http://darwincore.calacademy.org/.

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

GBIF (2006). Global Biodiversity Information Facility (GBIF), http://www.gbif.org.

Afsarmanesh, H., M. Wiedijk, et al. (1994). The PEER Information Management Language User Manual. Amsterdam, Department of Computer Systems, University of Amsterdam.

Blondel, V., A. Gajardo, et al. (2004). "A measure of similarity between graph vertices: applications to synonym extraction and web searching." Journal of SIAM Review 46(4): 647-666.

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.

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. In 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. (2005). "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.

Wang, G., J. Goguen, et al. (2004). Critical Points for Interactive Schema Matching. In 6th Asia Pacific Web Conference.

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