Mining Relatedness Graphs for Data Integration

Jeremy T. Engle
(jtengle@indiana.edu)
Ying Feng
(yingfeng@indiana.edu)
Robert L. Goldstone
(rgoldsto@indiana.edu)
Indiana University
Bloomington, IN. 47405 USA

Abstract

In this paper, we present the AbsMatcher system for schema matching which uses a graph based approach. The primary contribution of this paper is the development of new types of relationships for generating graph edges and the effectiveness of integrating schemas using those graphs. AbsMatcher creates a graph of related attributes within a schema, mines similarity between attributes in different schemas, and then combines all information using the ABSURDIST graph matching algorithm. The attribute-to-attribute relationships this paper focuses on are semantic in nature and have few requirements for format or structure. These relationships sources provide a baseline which can be improved upon with relationships specific to formats, such as XML or a relational database. Simulations demonstrate how the use of automatically mined graphs of within-schema relationships, when combined with cross-schema pair-wise similarity, can result in matching accuracy not attainable by either source of information on its own.

Keywords: Data integration; graph matching; ABSURDIST; semantic relatedness.

Introduction

Data integration has application to a wide variety of fields from e-commerce to bioinformatics. One of data integration's subtopics is the attribute matching problem which finds mappings between attributes in source and target data sets. This paper presents the AbsMatcher framework which concentrates on one-to-one attribute matches as an initial effort, leaving complex n-to-one matches to future work. AbsMatcher finds matching results based on graphs of within-schema attribute relationships and cross-schema comparisons of attribute similarity. The focus of in this paper is the process and relationships used to create a within-schema graph for each data set.

The AbsMatcher framework has two distinct phases. The first is a mining phase which produces a graph for each data set where edges relate within-schema attributes and an aggregated matrix of cross-schema attribute similarities measures. We refer to these graphs as internal information, because a graph only contains information relating attributes within the same schema. Information which is aggregated into the cross-schema matrix is referred to as external information because it involves a comparison between attributes in different schemas to determine how semantically similar they are.

Secondly, AbsMatcher's matching phase uses the ABSURDIST (Feng, Goldstone, & Menkov, 2004; Goldstone & Rogosky, 2002) algorithm to combine mined information and determine match correspondences using an iteratively converging global optimization algorithm. ABSURDIST was originally developed to translate between conceptual systems in a psychologically plausible manner. Additionally, ABSURDIST has a weighting ratio to determine the balance of influence on the outcome of internal and external information. Though we focus on specific sources of internal and external information in this paper both AbsMatcher and ABSURDIST were designed in a way so that additional sources could easily be added.

In a graph based approach to schema matching the matching process uses graph matching to determine mappings between attributes/nodes based on the similarity of their sets of relationships/edges. Edges of graphs are labeled with different relationship types, which represent different forms of information. Relationship types can broadly be divided into structural relationships which are based on how attributes are organized in a data set and semantic relationships which are based on meaning of the information associated with an attribute. An example of a structural relationship is a parent/child relationship between nested attributes from an XML data set. An example of a semantic relationship is a general/specific relationship for the concepts represented by two attributes. These examples highlight that one of the challenges in how to create graphs is that what relationships can be used is tied to the format of a data set and the available (meta)data.

Previous systems (Aumueller, Do, Massmann, & Rham, 2005; Giunchiglia & Shvaiko, 2003; Melnik, Garcia-Molina, & Rahm, 2002) address the problem of how to create graphs by only using metadata which can be intuitively translated to graph form. As a result, the graphs created by these translations predominately represent the structural design of a data set. Though structural relationships can be useful, their disadvantage is that factors such as missing metadata, differing metadata formats, or different database designers can remove these structural relationships' usefulness.

This paper presents a set of relationships which can be used as general practice but more importantly are still applicable when metadata is limited or datasets with differing formats are being integrated. Of particular contribution are Yahoo semantic relatedness relationships which leverage Yahoo query results to measure the semantic relatedness of attributes' names. Yahoo relationships are an improvement over the use of tools like Wordnet because of their ability to handle attribute names that have abbreviations, words unique to a domain, and/or multi-term phrases. All three of these factors commonly occur in data sets. Together, Yahoo relationships and the other relationships we present offer tools to use when structural metadata is absent or of no benefit.

We use the terms AbsMatcher and ABSURDIST throughout this work. AbsMatcher is the overall system which formulates graphs after mining internal information and aggregates mined sources of external similarity. ABSURDIST refers specifically to the matching phase which iteratively combines internal and external information to determine a set of correspondences.

ABSURDIST Background

ABSURDIST was developed to solve the general problem of translating between two conceptual systems. We adapt this approach to data integration by treating attributes as concepts to be matched. A complete discussion of ABSURDIST and how information factors into the iterative process can be found in Goldstone and Rogosky (2002). Information in ABSURDIST is classified as internal (within-schema) or external (cross-schema). External information provides the ability to input cross-schema similarity into the ABSURDIST algorithm. Different external sources are aggregated into an NxM matrix of values between 0 and 1, where N and M are the sizes of the schemas to be matched. The dividing line between internal and external is that internal information is relationships between attributes in the same schema, whereas external similarity is a comparison between attributes in two separate schemas.

ABSURDIST iteratively updates correspondences using internal and external information until reaching a stable point, terminates, and selects the final matches. ABSURDIST as an error minimization algorithm selects the set of matches that result in the least total link error. This section discusses the conceptual motivations of ABSURDIST and leaves specific examples of internal and external information for later sections.

Internal Information as Graphs

Internal information in ABSURDIST represents intra-system information about how nodes in each conceptual system relate to other nodes in the same system. Internal information for a system is independent of the system with which it is being aligned. For each schema, ABSURDIST takes internal information as input in the form of; information on relationship types, node types, node information, and a graph of relationships. Internal information factors into the *R* and *I* terms of Equation 1. A node in a conceptual system must have a unique identifier and a categorical type. If only one type exists then the effects of node types become irrelevant. Relationships in ABSURDIST represent a conceptual association between intra-system nodes creating a generalized interpretation of structure. A relationship type has a categorical label and is defined as being either directed or undirected. Relationships are instantiated as edges, which collectively form a graph of continuously valued weighted edges. If the same weight is used for every edge, these weights become irrelevant.

Iterative Algorithm

ABSURDIST is an iterative algorithm which updates an NxM matrix of correspondences where N and M refer to the number of attributes in the source schema, *A*, and target schema, *B*, respectively. Each cell in the correspondence matrix, $C_t(A_q,B_r)$, represents how strong a match is at iteration step *t* for attribute *q* in schema *A* and attribute *r* in schema *B*. The algorithm terminates when the matrix has converged or a maximum number of iterations is reached. For each iteration, ABSURDIST updates each $C_t(A_q,B_r)$ by a net input defined by

$$N(A_q, B_r) = \alpha E(A_q, B_r) + \beta R(A_q, B_r) - \chi I(A_q, B_r)$$

Equation 1. Correspondence Update Equation

Equation 1 shows how internal (*R* and *I*) and external (*E*) information combine to update the correspondence from attribute *q* in schema *A* to attribute *r* in schema *B*. The *E* term represents similarity based on external information, the *R* term represents similarity based on internal information, and the *I* term uses internal information to inhibit incorrect correspondences. As a global optimization algorithm, both *R* and *I* take into account the state of the system at each iteration t. α, β, and χ are weights that control the influence of forms of information, where α and β are set as a ratio to each other and χ is set independently of the others. For example, when α is one and β is zero only external information is used to find correspondences.

Related Research

A number of surveys have been done which cover the different aspects of the schema matching problem (Shvaiko & Euzenat, 2005). One of the established approaches to schema matching is to use candidate matchers to generate candidate matches which are aggregated into a final set. Graph-based systems, including AbsMatcher, have multiple modules to generate edges in the graph, multiple modules to generate the equivalent of external information, and then use a graph matching algorithm to generate correspondences based on graphs. It is possible that correspondences generated using a graph matching algorithm could be used as a candidate matcher in a system. Cupid (Madhavan, Bernstein, & Rahm, 2001), and Similarity Flooding (Melnik, Garcia-Molina, & Rahm, 2002) systems all use

graph matching to accomplish schema matching. COMA++ (Aumueller et al., 2005) is a generalized framework for schema matching which was used in the Similarity Flooding system to combine the results from graph matching with non-graph-oriented candidate matchers. The difference between AbsMatcher and these previous systems is the generality of AbsMatcher and generating graphs based on semantics instead of data model metadata.

Previous graph-based schema matchers construct graphs based on the metadata for the data model. These systems have modules specifically built for translating different data models -- such as relational databases, XML, ontologies, or conceptual hierarchies -- into a graph form. This approach makes the graphs generated dependent on the thoroughness of the data set creator, and completely different graphs will be generated even when the same data set is stored in different data models. The advantage of these systems is that they leverage the effort of data set creators. For example considerable effort is generally put into the design phase of a relational database. Examples of using metadata would be creating a relationship between parent and child XML attributes or the fact that an attribute is a primary key in a relational database. The disadvantage of basing graphs on metadata is that derived relationships often have more to do with how data is stored and less about semantic relationships. The goal of the information sources we present in this paper is that they can be used regardless the data model and still generate semantic relationships.

The Semantic Matching (Giunchiglia & Shvaiko, 2003) system provides the closest comparison to AbsMatcher. It creates a graph based on metadata and a limited number of semantic relationships. Semantic Matching uses electronic thesauri in order to create overlap, mismatch, and general/specific relationships. The one issue with electronic thesauri is that they only work with words in their index and are unable to handle abbreviations or phrases which are often used to name attributes. AbsMatcher shares the same motivation as Semantic Matching, but uses the web to create semantic relatedness relationships and mines the data sets for statistical relatedness relationships. Additionally, ABSURDIST was designed with a general idea of relationships, which makes adding new forms of internal relationships a simple process.

We mine semantic relatedness using Yahoo query results (Bollegala et al., 2007) and Information Dependencies (Dalkilic & Robertson, 2000), however, neither has been used for schema matching.

Mining ABSURDIST Graphs

The focus of this paper is on the process and relationships types used to create within-schema graphs. The unifying characteristic for all of the relationships we present is that they are not specific to a data model nor represent structural information. We present two categories of relationships; ones which use the entropy of the data and the second which uses Yahoo query results based on attribute names to measure semantic relatedness.

Mining an ABSURDIST graph is a two-stage process. The first is mining edges of the desired relationship type and the second is filtering out noisy edges. Filtering is done by using thresholds to eliminate mined edges whose values are not statistically significant enough to represent something beyond noise. For brevity's sake we limit the discussion of filtering to describing what the threshold checks for each relationship type.

Entropy Relations

Entropy-based relationships use an information theoretic approach to look at the information content of attributes based on their data. The goal is to look for patterns which defy statistical trends and therefore are more likely to represent user intended relationships. We use the Information Dependency (InD) measure (Dalkilic & Robertson, 2000), which is based on Shannon's Entropy, to look at the information content of attributes. Entropy relationships require at least a sample of the data. The discussion of Entropy relationships includes approximate attribute entropy relationships, data set key relationships, and approximate functional dependencies.

Attribute entropy relationships measure the degree to which attributes resemble keys, which have a different value in each record in the data set for the attribute, or constants, which have the same value in each record in the data set for the attribute. An attribute being close to a key or constant is a unique statistical property which is a result of how data is created, e.g. an ISBN is purposefully defined as a key. Attributes in other data sets that are semantically similar are likely to also have similar statistical properties, so when keys or constants occur they are strong indicators of a likely match. In Table 1, PersonName is an example of a key and Gender is an example of an attribute that is almost a constant. Attribute Entropy relationships are filtered based on their entropy values and only kept when those values are either above (approximate key) or below (approximate constant) defined thresholds.

Data set keys are sets of attributes that together have a unique set of values for the data set and therefore form a key. Data set key relationships are created between pairs of attributes that together are close to forming, or do form, a data set key, but neither attribute is a key on its own. An example from Table 1 is that by combining Address and Gender a unique set of values exists for every row. The above example would result in an edge *PairKey*(Address, Gender) to be created in the graph. A data set key relationship creates undirected edges between attributes and uses the entropy value as the weight. Data set approximate key relationships are filtered using a threshold which defines how close to a primary key the attribute set must be.

The last Entropy relationship type uses Approximate Functional Dependencies (AFDs). AFDs are probabilistic rules, X➔Y, which measure the ability of values for a left hand side (LHS) attribute set to determine values of the right hand side (RHS) attribute set. The closer an AFD's measured value is to 1 the better the LHS is at predicting the

RHS. AbsMatcher's use of AFDs as an information source for schema matching presents a novel application for AFDs. We use AFDs which have a single attribute LHS and a single attribute RHS in creating dependency relationships. By only using single attributes on each side the search space is reduced from 2^{N+M} to NxM. Though Functional Dependencies (FDs), which AFDs extend, have been used in schema matching, this is to our knowledge the first use of AFDs. Filtering dependency relationships uses a threshold which parameterizes the number of standard deviations that an AFD's value must be away from the average value of all AFDs with the same LHS or RHS.

Table 1. A sample data set of people

PersonName	Address	Gender
Santa Claus	100 North Pole	Male
Mrs. Claus	100 North Pole	Female
Jeremy Engle	215 Lindley Hall	Male
Rob Goldstone	338 Psychology	Male

Semantic Relationships

The premise behind using semantic relatedness is to create a relationship between attributes that are thematically related. A trivial example of this would be attributes for the first and last name of a person. If the respective attribute labels are "first" and "last" then a graph edge is created between these attributes based on the thematic association of these labels.

$$WebJaccard(P,Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \dfrac{H(P \cap Q)}{H(P) + H(Q) - H(P \cap Q)}, & H(P \cap Q) \geq c \end{cases}$$

Equation 2. WebJaccard Using Yahoo! Query Hits

One of the common tools for mining semantic relatedness is using WordNet (Fellbaum, 1998). Semantic relationships are found for two words according to their common membership in sets of synonyms, or synsets. Though WordNet has a large dictionary, the tools that rely on it fail when one of the two words is not in the dictionary. There are two common scenarios which increase the likelihood of WordNet failing. The first is that data sets commonly have domain specific terms that are less likely to be in a general dictionary like WordNet. The second problem is that data sets commonly have attribute names that are multiple words and/or use abbreviations. The tools making use of WordNet are not capable of handling either of these cases. In order to overcome these issues, we use tools that query the World Wide Web instead of WordNet.

We use the WWW as a source of information and adapt existing information retrieval measures to use the number of results from queries to compute similarity. Our semantic relatedness relationships are based on work by Bollegala et al. (2007) which queried Google and used the number of query results in computing existing similarity measures, however they only tested its use on single words.

The first step in mining semantic relatedness relationships is to tokenize attribute names. Attribute names are tokenized on occurrences of underscores and capital letters to create a multi-term query. Though not sophisticated these simple rules provide a best effort for creating multi-term queries. The relatedness of two attributes is then found using the WebJaccard measure as expressed in Equation 2, where P and Q are the multi-term queries for each attribute name. When available we also include the data set name as a query term to provide sense disambiguation. We use Yahoo as a source for querying because of the open availability of their search API. Yahoo semantic relatedness relationships are filtered to include edges only when the WebJaccard value is above a threshold.

Mining the External Similarity Matrix

We use existing sources of external information, and therefore only discuss them briefly. External information directly compares attributes in the source and target schemas to look for similar attributes. While mining external similarity both attribute names and values from the data are used. We tested basic sources of external information to investigate the effects of combining internal and external information. Two sources of external similarity were prototyped and tested.

The first source of external similarity is string edit distance, which is a lexical comparison of attribute names. String edit distance represents a method for finding matches that are "low hanging fruit." We use the jSimlib (https://jsimlib.dev.java.net/) library that normalizes string edit distance by the sum of the length of the two strings.

The second source of external similarity is cosine similarity, which is commonly used to compare the similarity of two free text documents. The similarity of the two documents is computed as the cosine value between the term frequency vectors for each document. For attribute-to-attribute schema matching, when the attributes contain text we treat them as documents and create term frequency vectors. The Lucene (http://lucene.apache.org/java/docs/index.html) framework was used to calculate the cosine similarity.

We tested three groups of data sets that vary in domain and size which come from the Illinois Semantic Integration Archive (ISIA) at http://pages.cs.wisc.edu/~anhai/wisc-si-archive/. The Courses data sets have listings of classes from four different universities, data sets sizes range from twelve to sixteen attributes. The second group of data sets is the Real Estate I (REI) data sets, which includes the homeseekers, nky, windermere, and yahoo data sets. Three of the data sets have sizes in the mid-thirties and the final one is in the sixties. The third group of data sets is the Real Estate Core (REC) data sets. REC data sets are the same as the REI data sets, but only include attributes that have a match in one of the other data sets. This reduced the number of attributes in the data sets to the low twenties, except one having twenty-eight attributes. The REC group is used to test the effects on matching performance when attributes with no matches are removed.

Validation Experiments

The goals in evaluating AbsMatcher are to look at the performance of internal information by itself and whether the combination of internal and external information provides better cumulative performance. WebJaccard and Entropy internal relationships are meant to provide a baseline ability for schema matching so performance is judged first by whether consistent evidence of an ability to find matches, and second by looking for evidence that combining internal and external information is better than only external information. Finding evidence of these two points would indicate matches being found which internal information can uniquely contribute to finding. Performance is measured using recall. Many schema matching systems provide statistical matches, as opposed to absolute matching, so we present recall for correct matches made and for the correct match being one of the top 3 best matches. This more liberal scoring criterion provides information on whether AbsMatcher has partial information that could be leveraged by future improvements to the algorithm or information sources. Precision is not included because currently AbsMatcher returns a match for each attribute in the smaller of the two schemas. This means that the number of matches returned for a pair of schemas will remain constant no matter what other parameters change. This point is discussed further in future work.

For the initial tests, we first explored schema matching using only the previously described internal relationships, in three combinations. The Entropy combination includes attribute entropy, data set key, and dependency relationships. The WebJaccard results consist of semantic relatedness relationships based on Yahoo results. Finally, the "All" combination includes both Entropy and WebJaccard relationships.

We first look at the extent to which schemas can be matched using only the mined graphs for the two data sets. When using only this limited source of information a high level of performance cannot be expected. However, this limitation is useful in making an initial judgment of whether mined graphs contain useful information. For each group of data sets we select the best performing parameters and present the results in Figure 1 for all three combinations of internal relationships and all three groups of data sets.

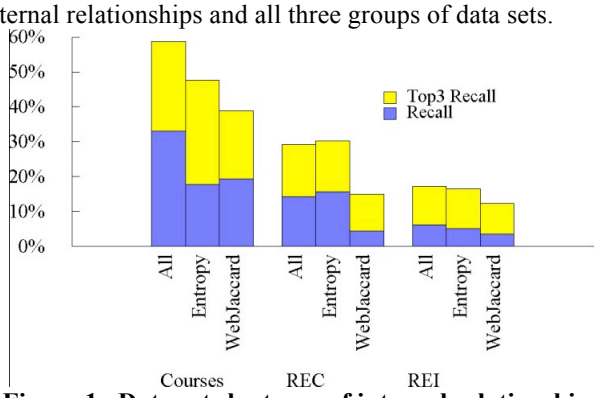

Figure 1. Data sets by types of internal relationships

The first result to examine is AbsMatcher's ability to find correct matches. Though the results in Figure 1 are relatively low in the context of overall performance of schema matching systems, the more appropriate context is as a source of matches which would be used in a broader system. In this context WebJaccard and Entropy relationships do show consistent ability to find at least some matches. The performance of the top 3 correspondences improves over just correct matches indicating that AbsMatcher can provide supporting evidence which would affirm or discredit correspondences from other candidate matchers. As seen in Figure 1 the top 3 correspondences can provide useful results on a third to half of all matches. The top 3 matches can be useful when considering that the weights of correspondences in the top 3 can often be very close.

The second result to examine is what sources or combinations of sources of internal relationships are the most effective. Neither the Entropy nor WebJaccard relationships were consistently the best between the different data set groups. Though neither was consistently the best, the positive result is that when combined in *All,* performance improved or matched the performance of the best performing source of internal relationships. The fact that adding sources of internal relationships does not degrade performance strengthens the potential that when other existing forms of internal relationships are added, performance could be improved.

For the second set of tests, we combined both internal and external sources of information. For some matches the information which best indicates the correct match is derived by comparing an attribute from each data set. In ABSURDIST this means the use of external information that is combined with internal information using Equation 1. In Equation 1 there are two weighting coefficients, α and β, which determine the balance between external and internal information. The α:β ratio represents the comparative weights of external:internal information. We tested AbsMatcher with different ratios, where each represented a different balance between external and internal information. Figure 2 presents results for a representative three of those ratios. The 0:1 data point represents using only internal information, which corresponds with the results in Figure 1. The 1:0 data point represents only using external information. The 3:1 data point tested the effort to combine the use of internal and external information. The goal in this evaluation is to determine whether combining internal and external information has a benefit over just using external similarity.

Figure 2 provides evidence that combining internal and external information can for some data sets provide better results than either one in isolation. Though the improvement for Courses and REC data sets is small the fact that it occurs for both supports the claim that internal structure can improve matching performance. It must be remembered that the results for Courses and REC represent the average performance across twelve different pairs of

data sets matched. The ability of internal structure to find correct matches and the additional beneficial effect that it can have when combined with external similarity indicates that internal structure is to some extent finding both unique and useful information for schema matching.

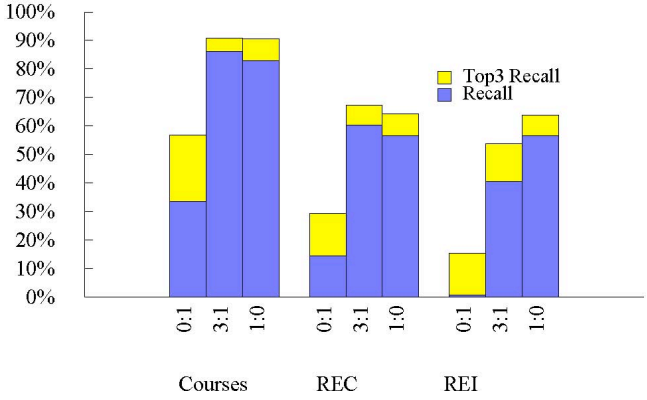

Figure 2. Data Sets by Ext:Int Ratio

The REI data sets do not benefit from internal information. This could in part be due to the fact that REI alignments leave more attributes unmatched. The REC data sets are versions of the REI data sets where attributes with no matches removed. They average 15.5 correct matches between a pair of data sets, meaning that on average half of the attributes in a data set for REI are not being matched, yet information is still mined for them. Courses and REC have a different scale yet both show similar trends in the ability to find correct matches. The only difference between REC and REI data sets is the existence of unmatched attributes, so the difference in performance can be unambiguously attributed to this. This indicates that information which indicates invalid matches could be an important feature to add to AbsMatcher.

Conclusions

The goal in developing AbsMatcher was to create a schema matching system that used a graph based approach, but was not reliant on a specific data model as a source of information. To this end, we propose Entropy and WebJaccard relationships which can be used even when more descriptive metadata, such as XML or metadata from a relational database, is unavailable. Additionally, these relationships emphasize non-structural relationships in an effort to create graphs which are more conceptual in nature. We then tested these graphs using the ABSURDIST graph matching system. ABSURDIST is ideally suited because of its ability to accept graphs with a wide variety of forms (weighted, unweighted, directed, undirected, labeled, and unlabeled) and ABSURDIST was designed specifically with the idea of combining internal and external information together.

The goals in testing AbsMatcher were to look at whether Entropy and WebJaccard relationships are useful for schema matching on their own and whether they have benefits when combined with external similarity. Experiments demonstrated that to varying extents the tested relationships are able to accomplish both of the goals. The results presented in this paper where aggregated over multiple individual experiments. The additive benefit of our sources of internal structure is important because it argues that internal structure holds unique information for finding correspondences.

These results were based on aggregating results from a number of matching pairs. It is important to note that there were outliers on both the positive and negative side. This is a common problem in schema matching, where sources of information perform well in certain scenarios and poorly in others. It is this point which motivated the approach of aggregating many disparate measures of similarity. This leads to the idea that by adding new information sources into AbsMatcher we can improve even beyond the baselines presented in this work.

Acknowledgments

This research was supported by National Science Foundation REESE grant 0910218, Lockheed Martin, and DARPA.

References

Aumueller, D., Do, H.-H., Massmann, S. and Rahm, E. (2005). Schema and ontology matching with COMA++ *Proceedings of the ACM SIGMOD international conference on Management of data*, ACM, Baltimore, Maryland.

Bollegala, D., Matsuo, Y. and Ishizuka, M. (2007). Measuring semantic similarity between words using web search engines *Proceedings of the 16th international conference on World Wide Web*, ACM, Banff, Alberta, Canada.

Dalkilic, M.M. and Roberston, E.L. (2000). Information dependencies *Proceedings of the 19th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, ACM, Dallas, Texas, United States.

Fellbaum, C. (1998). *Wordnet: An Electronic Lexical Database*. Bradford Books.

Feng, Y., Goldstone, R.L. and Menkov, V. (2004). ABSURDIST II: A Graph Matching Algorithm and its Application to Conceptual System Translation *Proceedings of the 17th International Florida Artificial Intelligence Research Symposium Conference (FLAIRS)*, AAAI Press, Miami Beach, Fla., USA, 640-645.

Giunchiglia, F. and Shvaiko, P. Semantic Matching (2003). *Knowedge Engingeering Review*, *18* (3). 265-280.

Goldstone, R.L. and Rogosky, B.J. (2002). Using Relations within Conceptual Systems to Translate across Conceptual Systems, *Cognition*, 295-320.

Madhavan, J., Bernstein, P.A. and Rahm, E. (2001). Generic Schema Matching with Cupid *VLDB*.

Melnik, S., Garcia-Molina, H. and Rahm, E. (2002). Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching *ICDE*.

Shvaiko, P. and Euzenat, J. (2005). A Survey of Schema-Based Matching Approaches *Journal on Data Semantics IV*, *3730*. 146-171.