A Vector Space Model for Semantic Similarity Calculation and OWL Ontology Alignment

Rubén Tous and Jaime Delgado

Distributed Multimedia Applications Group (DMAG)
Universitat Politècnica de Catalunya (UPC), Dpt. d'Arquitectura de Computadors
Universitat Pompeu Fabra (UPF), Dpt. de Tecnologia
rtous@ac.upc.edu, jaime.delgado@upf.edu

Abstract. Ontology alignment (or matching) is the operation that takes two ontologies and produces a set of semantic correspondences (usually semantic similarities) between some elements of one of them and some elements of the other. A rigorous, efficient and scalable similarity measure is a pre-requisite of an ontology alignment process. This paper presents a semantic similarity measure based on a matrix represention of nodes from an RDF labelled directed graph. An entity is described with respect to how it relates to other entities using N-dimensional vectors, being N the number of selected external predicates. We adapt a known graph matching algorithm when applying this idea to the alignment of two ontologies. We have successfully tested the model with the public testcases of the Ontology Alignment Evaluation Initiative 2005.

1 Introduction

1.1 Motivation

For many knowledge domains (biology, music, web directories, digital rights management, etc.) several overlapping ontologies (middle ontologies) are being engineered. Each one is a different abstraction and representation of the same or similar concepts. There are proliferating also a myriad of problem-specific ontologies (lower ontologies) for many applications, metadata repositories, personal information systems and peer-to-peer networks.

To enable collaboration within and across information domains, software agents require the semantic alignment (mapping) of the different formalisms. The alignment process will identify the equivalences between some entities (e.g. classes and properties) of the participating ontologies, and the different levels of confidence. These mappings are required before the querying of semantic data from autonomous sources can take place.

1.2 Ontology Alignment Formal Definition

Ontology alignment (or matching) is the operation that takes two ontologies and produces a set of semantic correspondences (usually semantic similarities) between some elements of one of them and some elements of the other. Several

S. Bressan, J. Küng, and R. Wagner (Eds.): DEXA 2006, LNCS 4080, pp. 307–316, 2006.

ontology alignment algorithms have been provided like GLUE [2], OLA [7] or FOAM [4]. A more formal definition, borrowed from [3], can be given:

Definition 1. Given two ontologies $\mathcal{O}$ and $\mathcal{O}'$, an *alignment* between $\mathcal{O}$ and $\mathcal{O}'$ is a set of correspondences (i.e., 4-uples): $< e, e', r, n >$ with $e \in \mathcal{O}$ and $e' \in \mathcal{O}'$ being the two matched entities, r being a relationship holding between e and e', and n expressing the level of confidence [0..1] in this correspondence.

It is typically assumed that the two ontologies are described within the same knowledge representation language (e.g. OWL [12]). Here we will focus on automatic and autonomous alignment, but other semi-automatic and interactive approaches exist.

1.3 Semantic Similarity Measures

The ontology alignment problem has an important background work in discrete mathematics for matching graphs [8][13], in databases for mapping schemas [14] and in machine learning for clustering structured objects [1]. Most part of ontology alignment algorithms are just focused on finding close entities (the "=" relationship), and rely on some *semantic similarity* measure.

A semantic similarity measure tries to find clues to deduce that two different data items correspond to the same information. Data items can be ontology classes and properties, but also instances or any other information representation entities. Semantic similarity between ontology entities (within the same ontology or between two different ones) may be defined in many different ways. The recently held Ontology Alignment Evaluation Initiative 2005 [11] has shown that the best alignment algorithms combine different similarity measures. [7] provides a classification (updating [14]) inherited from the study of similarity in relational schemas. This classification can be simplified to four categories when being applied to ontologies: Lexical, Topological, Extensional and Model-based.

1.4 Our Approach

The work presented in this paper takes a topological or structure-based semantic similarity approach. As ontologies and knowledge-representation languages evolve, more sophisticated structure-based similarity measures are required. In RDF (Resource Description Framework [15]) graphs, relationships are labeled with predicate names, and trivial distance-based strategies cannot be applied. Some works like [6] explore similarity measures based on structure for RDF equivalent bipartite graphs.

Our work focus also in RDF, but faces directly the natural RDF labelled directed graphs. The approach can be outlined in the following two points:

1. To compute the semantic similarity of two entities we have taken the common RDF and OWL predicates as a semantic reference. Objects are described and compared depending on how they relate to other objects in terms of these predicates. We have modeled this idea as a simple vector space.

2. To efficiently apply our similarity measure to the ontology alignment problem we have adapted it to the graph matching algorithm of [5].

2 Representing RDF Labelled Directed Graphs with a Vector Space Model (VSM)

In linear algebra a *vector space* is a set V of *vectors* together with the operations of addition and scalar multiplication (and also with some natural constraints such as closure, associativity, and so on). A vector space model (VSM) is an algebraic model introduced a long time ago by Salton [16] in the information retrieval field. In a more general sense, a VSM allows to describe and compare objects using N-dimensional vectors. Each dimension corresponds to an orthogonal feature of the object (e.g. weight of certain term in a document).

In an OWL ontology, we will compare entities taking into consideration their relationships with all the other entities present in the ontology - First we will focus on similarity within the same ontology, next we will study its application to the alignment of two ontologies -. Because relationships can be of different nature we will model them with a vector space. For this vector space, we will take as dimensions any OWL, RDF Schema, or other external predicate (not ontology specific) e.g. *rdfs:subClassOf*, *rdfs:range* or *foaf:name*. We can formally define the relationship of two nodes in the model:

Definition 2. Given any pair of nodes n_1 and n_2 of a directed labelled RDF graph $\mathcal{G}_\mathcal{O}$ representing the OWL ontology $\mathcal{O}$, the relationship between them, $rel(n_1, n_2)$, is defined by the vector $\{arc(n_1, n_2, p_1), ..., arc(n_1, n_2, p_N)\}$, where arc is a function that returns 1 if there is an arc labelled with the predicate p_i from n_1 to n_2 or 0 otherwise. p_i is a predicate from the set of external predicates $\mathcal{P}$ (e.g. {*rdfs:subClassOf*, *foaf:name*}).

$$rel(n_1, n_2) = \{arc(n_1, n_2, p_1), ..., arc(n_1, n_2, p_N)\} \mid n_1, n_2 \in \mathcal{G}_\mathcal{O} \ \wedge \ \forall i \in [0; N], p_i \in \mathcal{P}$$

$$arc(n_1, n_2, p_i) = \begin{cases} 1 & \text{if there is an arc labelled with } p_i \text{ from } n_1 \text{ to } n_2; \\ 0 & \text{otherwise.} \end{cases}$$

Example 1. Let us see a simple example. Take the following graph $\mathcal{G}_\mathcal{A}$ representing an ontology $\mathcal{O}_\mathcal{A}$. Imagine a trivial two-dimensional vector space to model the relationships between nodes. External predicates *rdfs:domain* and *rdfs:range* have been chosen for dimensions 0 and 1 respectively.

The relationship between the property *directs* and the class *director* will be described by $\{1, 0\}$. The relationship between the property *actsIn* and the class *movie* will be described by $\{0, 1\}$, and so on.

Now, the full description of an entity can be achieved with a vector containing the relationships between it and all the other entities in the ontology. Putting all

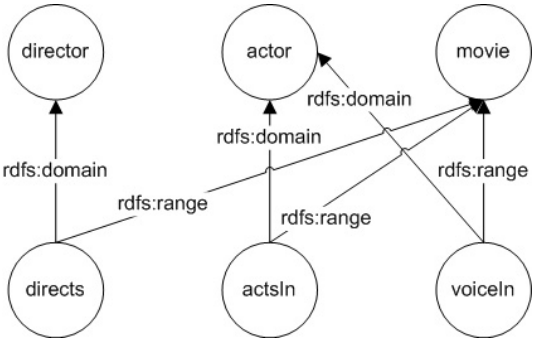

Fig. 1. $\mathcal{G}_{\mathcal{A}}$

the vectors together we obtain a three-dimensional matrix $\mathcal{A}$ representation of the labelled directed graph $\mathcal{G}_{\mathcal{A}}$ (row order: *director*, *actor*, *movie*, *directs*, *actsIn*, *voiceIn*):

$$\mathcal{A} = \begin{pmatrix} (0,0) & (0,0) & (0,0) & (0,0) & (0,0) & (0,0) \\ (0,0) & (0,0) & (0,0) & (0,0) & (0,0) & (0,0) \\ (0,0) & (0,0) & (0,0) & (0,0) & (0,0) & (0,0) \\ (1,0) & (0,0) & (0,1) & (0,0) & (0,0) & (0,0) \\ (0,0) & (1,0) & (0,1) & (0,0) & (0,0) & (0,0) \\ (0,0) & (1,0) & (0,1) & (0,0) & (0,0) & (0,0) \end{pmatrix}$$

3 Similarity of Entities Within the Same Ontology

In the general case, the correlation between two vectors x and y in an N-dimensional vector space can be calculated using the scalar product. We can normalize it by dividing this product by the product of the vector modules, obtaining the cosine distance, a traditional similarity measure. In our case, vectors describing entities in terms of other entities are composed by relationship vectors (so they are matrices). We can calculate the scalar product of two of such vectors of vectors V and W using also the scalar product to compute $V_i W_i$:

$$V \cdot W = \sum_{i=1}^{N} \sum_{j=1}^{M} V_{ij} W_{ij}$$

Applying this equation to the above example we can see that the scalar product of e.g. the vector describing *directs* and the vector describing *actsIn* is $directs \cdot actsIn = 1$. The scalar product of *actsIn* and *voiceIn* is $actsIn \cdot voiceIn = 2$, and so on. Normalizing these values (to keep them between 0 and 1) would allow to obtain a trivial similarity matrix of the ontology entities. However, we aim to propagate the structural similarities iteratively, and also to apply this

idea to the alignment of two different ontologies. In the following sections we will describe how to do it by adapting the ideas described in [5].

4 Applying the Model to an Ontology Alignment Process

To calculate the alignment of two ontologies represented with our vector space model we have adapted the graph matching algorithm of [5]. This adapted algorithm calculates entity similarities in an RDF labelled directed graph by iteratively using the following updating equation [1] :

Definition 3. $S_{k+1} = BS_kA^T + B^TS_kA, k = 0, 1, ...$
where S_k is the $N_B * N_A$ similarity matrix of entries s_{ij} at iteration k, and A and B are the $N_B * N_B * N_P$ and $N_A * N_A * N_P$ three-dimensional matrices representing $\mathcal{G}_\mathcal{A}$ and $\mathcal{G}_\mathcal{B}$ respectively. N_A and N_B are the number of rows of A and B, and P is the number of predicates selected as dimensions of the VSM.

Note that, as it is done in [5], initially the similarity matrix S_0 is set to 1 (assuming for the first iteration that all entities from $\mathcal{G}_\mathcal{A}$ are equal to all entities in $\mathcal{G}_\mathcal{B}$). If we start the process already knowing the similarity values of some pair of entities, we can modify this matrix accordingly, and keep the known values between iterations.

Example 2. Let's see a simple example. Take the following graphs $\mathcal{G}_\mathcal{A}$ and $\mathcal{G}_\mathcal{B}$. Figure 2 shows their corresponding RDF labelled directed graphs.

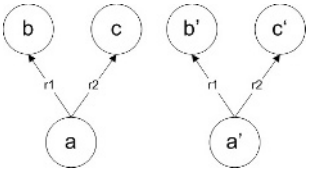

Fig. 2. $\mathcal{G}_\mathcal{A}$ (left) and $\mathcal{G}_\mathcal{B}$ (right)

$$A = \begin{pmatrix} (0,0) & (1,0) & (0,1) \\ (0,0) & (0,0) & (0,0) \\ (0,0) & (0,0) & (0,0) \end{pmatrix} B = \begin{pmatrix} (0,0) & (1,0) & (0,1) \\ (0,0) & (0,0) & (0,0) \\ (0,0) & (0,0) & (0,0) \end{pmatrix} S_0 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$S_1 = BS_0A^T + B^TS_0A = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

[1] The graph matching algorithm from [5] is exactly the same as the algorithm shown here, but using simple matrices of 1's and 0's instead of matrices of vectors. We omit more details to avoid redundancies and because the paper remains self-contained.

To normalize the similarity matrix (to keep its values between 0 and 1) [5] divides all its elements by the Frobenius norm of the matrix, defined as the square root of the sum of the absolute squares of its elements.

$$S_1 = S_1/frobeniusNorm(S_1) = \begin{pmatrix} 0,816 & 0 & 0 \\ 0 & 0,408 & 0 \\ 0 & 0 & 0,408 \end{pmatrix}$$

Iterating the algorithm 4 times it converges to the following result:

$$S_4 = \begin{pmatrix} 0,577 & 0 & 0 \\ 0 & 0,577 & 0 \\ 0 & 0 & 0,577 \end{pmatrix}$$

So, as expected the entities a', b' and c' (rows) are similar to a, b and c (columns) respectively.

4.1 Computational Cost and Optimization

Because the number of selected external predicates $p_i \in \mathcal{P}$ can be small and it is independent of the size of the ontologies, operations involving relationships vectors can be considered of constant cost, and the general algorithm of order $O(N^2)$. Because the number of nodes can be considerably high, some optimizations are required to constraint the processing time. Inspired in [6], we have classified nodes into five types: Properties (p), Classes (c), Instances (i), External Classes (c') and External Instances (i'). Because nodes from one type cannot be similar to nodes of another type, the matrices can be rewritten (rows and columns correspond to types previously mentioned and in the same order):

$$A = \begin{pmatrix} A_{p-p} & A_{p-c} & A_{p-i} & A_{p-c'} & A_{p-i'} \\ A_{c-p} & A_{c-c} & A_{c-i} & A_{c-c'} & A_{c-i'} \\ A_{i-p} & A_{i-c} & A_{i-i} & A_{i-c'} & A_{i-i'} \\ A_{c'-p} & A_{c'-c} & A_{c'-i} & A_{c'-c'} & A_{c'-i'} \\ A_{i'-p} & A_{i'-c} & A_{i'-i} & A_{i'-c'} & A_{i'-i'} \end{pmatrix}$$

$$S_k = \begin{pmatrix} S_p & 0 & 0 & 0 & 0 \\ 0 & S_c & 0 & 0 & 0 \\ 0 & 0 & S_i & 0 & 0 \\ 0 & 0 & 0 & S_{c'} & 0 \\ 0 & 0 & 0 & 0 & S_{i'} \end{pmatrix}$$

Definition 4. The S_{k+1} equation can be decomposed into three formulas:

$$\mathbf{S_{p_{k+1}}} = B_{p-p}S_{p_k}A_{p-p}^T + B_{p-c}S_{c_k}A_{p-c}^T + B_{p-i}S_{i_k}A_{p-i}^T + B_{p-c'}S_{c'_k}A_{p-c'}^T + B_{p-i'}S_{i'_k}A_{p-i'}^T + B_{p-p}^T S_{p_k}A_{p-p} + B_{c-p}^T S_{c_k}A_{c-p} + B_{i-p}^T S_{i_k}A_{i-p} + B_{c'-p}^T S_{c'_k}A_{c'-p} + B_{i'-p}^T S_{i'_k}A_{i'-p}$$

$$\mathbf{S_{c_{k+1}}} = B_{c-p}S_{p_k}A_{c-p}^T + B_{c-c}S_{c_k}A_{c-c}^T + B_{c-i}S_{i_k}A_{c-i}^T + B_{c-c'}S_{c'_k}A_{c-c'}^T + B_{c-i'}S_{i'_k}A_{c-i'}^T + B_{p-c}^TS_{p_k}A_{p-c} + B_{c-c}^TS_{c_k}A_{c-c} + B_{i-c}^TS_{i_k}A_{i-c} + B_{c'-c}^TS_{c'_k}A_{c'-c} + B_{i'-c}^TS_{i'_k}A_{i'-c}$$

$$\mathbf{S_{i_{k+1}}} = B_{i-p}S_{p_k}A_{i-p}^T + B_{i-c}S_{c_k}A_{i-c}^T + B_{i-i}S_{i_k}A_{i-i}^T + B_{i-c'}S_{c'_k}A_{i-c'}^T + B_{i-i'}S_{i'_k}A_{i-i'}^T + B_{p-i}^TS_{p_k}A_{p-i} + B_{c-i}^TS_{c_k}A_{c-i} + B_{i-i}^TS_{i_k}A_{i-i} + B_{c'-i}^TS_{c'_k}A_{c'-i} + B_{i'-i}^TS_{i'_k}A_{i'-i}$$

$S_{c'_{k+1}}$ and $S_{i'_{k+1}}$ are diagonal matrices passed as input parameters. They are kept unchanged between iterations.

4.2 Comparison Against Algorithms Based on Bipartite Graphs

The use of an algorithm to measure similarity between directed graphs could lead to think that it would be better to directly apply it over the ontologies equivalent bipartite graphs (like it is done in [6]), instead of adapting it to RDF labelled directed graphs. However, our approach has some advantages; on one hand we reduce critically the number of nodes and the computational cost. On the other hand, in bipartite graphs the core predicates of OWL are treated as all the other nodes, while in our model they become the semantic reference to describe and compare entities. Figure 3 shows the equivalent bipartite version of the previous example with two graphs of three nodes.

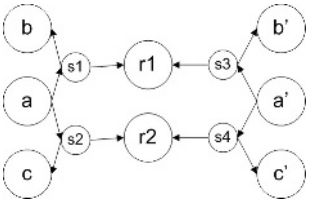

Fig. 3. $\mathcal{G}_\mathcal{A}$ (left) and $\mathcal{G}_\mathcal{B}$ (right)

Appying the alignment algorithm for bipartite graphs described in [6] we obtain the following similarity matrix between a', b', c' and a, b, c:

$$X_{22} = \begin{pmatrix} 0,405 & 0 & 0 \\ 0 & 0,153 & 0,05 \\ 0 & 0,05 & 0,153 \end{pmatrix}$$

As can be seen, the inclusion of statement nodes adds some symmetries not present in the original graphs, resulting in less precise results. Some similarities between nodes b' and c (and vice-versa) appear.

5 Results

To test our approach we have used the Ontology Alignment Evaluation Initiative 2005 testsuite [11]. The evaluation organizers provide a systematic benchmark test suite with pairs of ontologies to align as well as expected (human-based) results. The ontologies are described in OWL-DL and serialized in the RDF/XML format. The expected alignments are provided in a standard format expressed in RDF/XML and described in [11]. Because our model does not deal with lexical similarity, we have integrated our algorithm inside another hybrid aligner, Falcon [6] (replacing its structure similarity module by ours). This constraints the interest of the obtained results, but otherwise it hadn't been possible a comparative evaluation. Because most part of the tests include more lexical similarity than structural similarity challenges, our aligner and Falcon[2] obtain very similar results (the same for tests 101-104 and 301-304). The differences fall between tests 201-266, that we show in table 1.

Table 1. OAEI 2005 tests where our approach (vsm) obtains a different result than [6]

	vsm		falcon		foam		ola	
test	prec.	rec.	prec.	rec.	prec.	rec.	prec.	rec.
205	0.90	0.89	0.88	0.87	0.89	0.73	0.43	0.42
209	0.88	0.87	0.86	0.86	0.78	0.58	0.43	0.42
230	0.97	0.96	0.94	1.0	0.94	1.0	0.95	0.97
248	0.83	0.80	0.84	0.82	0.89	0.51	0.59	0.46
252	0.64	0.64	0.67	0.67	0.67	0.35	0.59	0.52
257	0.66	0.66	0.70	0.64	1.0	0.64	0.25	0.21
260	0.44	0.42	0.52	0.48	0.75	0.31	0.26	0.17
261	0.45	0.42	0.50	0.48	0.63	0.30	0.14	0.09
262	1.0	0.27	0.89	0.24	0.78	0.21	0.20	0.06
265	0.44	0.42	0.48	0.45	0.75	0.31	0.22	0.14
266	0.45	0.42	0.50	0.48	0.67	0.36	0.14	0.09

Rows correspond to test numbers, while columns correspond to the obtained values of precision (the number of correct alignments found divided by the total number of alignments found) and recall (the number of correct alignments found divided by the total of expected alignments).

From 50 tests our results just differ in 11 with respect to the aligner in which we have embedded our algorithm. We improve the results of Falcon in tests 205 and 206 (where labels have been replaced by synonyms), test 230 (where classes have been flattened) and test 262 (where everything except classes have been omitted). In test 257 (where names, comments and specialization hierarchy have been omitted) we just improve the recall value. In the other five tests our results are below the Falcon ones. We still cannot claim to outperform the original

[2] A description of all the tests can be obtained from [11]. Our results for tests not present in the table are the same as those of Falcon, and can be obtained in [6].

structural similarity algorithm of [6], but we can show that similar results (pretty good with respect to the other aligners) can be obtained by directly working over the RDF labelled directed graph, instead of working over the equivalent bipartite graph, that is bigger and can introduce symmetries not present in the original structure.

6 Related Work

The initial work around structure-based semantic similarity just focused on is-a constructs (taxonomies). Previous works like [10] measure the distance between the different nodes. The shorter the path from one node to another, the more similar they are. Given multiple paths, one takes the length of the shortest one. [17] finds the path length to the root node from the least common subsumer (LCS) of the two entities, which is the most specific entity they share as an ancestor. This value is scaled by the sum of the path lengths from the individual entities to the root. [9] finds the shortest path between two entities, and scales that value by the maximum path length in the is–a hierarchy in which they occur.

Recently, new works like [5] define more sophisticated topological similarity measures, based on graph matching from discrete mathematics. These new graph-based measures suit the particularities of the new ontologies, built with more expressive languages like OWL [12]. Our work is based on the previous work in [5], and also in its adaptation to OWL-DL ontologies alignment in [6]. This last work describes a structural similarity strategy called GMO (Graph Matching for Ontologies). Differently from our work, GMO operates over RDF bipartite graphs. It allows a more direct application of graph matching algorithms, but also increases the number of nodes and reduces scalability.

7 Conclusions

We have presented here an approach to structure-based semantic similarity measurement that can be directly applied to OWL ontologies modelled as RDF labelled directed graphs. The work is based on the intuitive idea that similarity of two entities can be defined in terms of how these two entities relate to the world they share (e.g. two red objects are similar with respect to the colour dimension, but their similarity cannot be determined in a general way). We describe and compare ontological objects in terms of how they relate to other objects. We model these relationships with a vector space of N dimensions being N the number of selected external predicates (e.g. *rdfs:subClassOf*, *rdfs:range* or *foaf:name*). We have adapted the graph matching algorithm of [5] to these idea to iteratively compute the similarities between two OWL ontologies. We have presented also an optimization of the algorithm to critically reduce its computational cost. The good results obtained in the tests performed over the Ontology Alignment Evaluation Initiative 2005 testsuite has proven the value of the approach in situations in which structural similarities exist.

Acknowledgements

This work has been partly supported by the Spanish administration (DRM-MM project, TSI 2005-05277).

References

1. M. Bisson. Learning in fol with a similarity measure. In *Proceedings of the 10th American Association for Artificial Intelligence conference, San-Jose (CA US), pages 8287*, 1992.
2. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web, 2002. citeseer.ist.psu.edu/doan02learning.html.
3. M. Ehrig and J. Euzenat. Relaxed precision and recall for ontology matching. http://km.aifb.uni-karlsruhe.de/ws/intont2005/intontproceedings.pdf.
4. M. Ehrig and S. Staab. Qom - quick ontology mapping. citeseer.ist.psu.edu/727796.html.
5. Vincent D. Blondel et al. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Rev.*, 46(4):647–666, 2004.
6. Wei Hu et al. Gmo: A graph matching for ontologies. In *Integrating Ontologies*, 2005.
7. J. Euzenat and P. Valtchev. Similarity-based ontology alignment in owl-lite. In *Proc. of ECAI 2004, pages 333–337, Valencia, Spain, August 2004*, 2004.
8. J. E. Hopcroft and R.M. Karp. An $\mathcal{O}(n^{5/2})$ algorithm for maximum matching in bipartite graphs. *SIAM J. Comput.*, 4:225–231, 1973.
9. C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. *WordNet: An Electronic Lexical Database*, 49(2):265–283, 1998.
10. J. H. Lee, M. H. Kim, and Y. J. Lee. Information retrieval based on conceptual distance in is-a hierarchies. *Journal of Documentation*, 49(2):188–207, 1993.
11. Ontology alignment evaluation initiative, 2005. http://oaei.inrialpes.fr/2005/.
12. Owl web ontology language overview. w3c recommendation 10 february 2004. See http://www.w3.org/TR/owl-features/.
13. C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.
14. E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal: Very Large Data Bases*, 10(4):334–350, 2001. citeseer.ist.psu.edu/rahm01survey.html.
15. Resource description framework. See http://www.w3.org/RDF/.
16. G. Salton. The smart retrieval system. *Experiments in Automatic Document Processing*, 1971.
17. Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, Morristown, NJ, USA, 1994. Association for Computational Linguistics.