Coordinating Semantic Peers

P. Bouquet[1], L. Serafini[2], S. Zanobini[1]

[1] Department of Information and Communication Technology – University of Trento
Via Sommarive, 10 – 38050 Trento (Italy)
[2]ITC – IRST
Via Sommarive, 15 – 38050 Trento (Italy)
{bouquet,zanobini}@dit.unitn.it, serafini@itc.it

Abstract. The problem of finding an agreement on the meaning of heterogeneous schemas is one of the key issues in the development of the Semantic Web. In this paper, we propose a new algorithm for discovering semantic mappings across hierarchical classifications based on a new approach to semantic coordination. This approach shifts the problem of semantic coordination from the problem of computing linguistic or structural similarities (what most other proposed approaches do) to the problem of deducing relations between sets of logical formulas that represent the meaning of nodes belonging to different schemas. We show how to apply the approach and the algorithm to an interesting family of schemas, namely hierarchical classifications. Finally, we argue why this is a significant improvement on previous approaches.

Keywords: Semantic Web, Semantic Interoperability, Information retrieval, Automated Reasoning

1 Introduction and approach

One of the key challenges in the development of open distributed systems is enabling the exchange of meaningful information across applications which (i) may use autonomously developed schemas for organizing locally available data, and (ii) need to discover mappings between schema elements to achieve their users' goals. Typical examples are databases using different schemas, and document repositories using different classification structures. In restricted environments, like a small corporate Intranet, this problem is typically addressed by introducing shared models (e.g., ontologies) throughout the entire organization. However, in open environments (like the Web), this approach can't work for several reasons, including the difficulty of 'negotiating' a shared model of data that suits the needs of all parties involved, and the practical impossibility of maintaining such a shared model in a highly dynamic environment. In this kind of scenarios, a more dynamic and flexible method is needed, where no shared model can be assumed to exist, and mappings between elements belonging to different schemas must be discovered on-the-fly.

The method we propose assumes that we deal with a network of *semantic peers*, namely physically connected entities which can autonomously decide how to organize locally available data (in a sense, are semantically autonomous agents). Each peer organizes its data using one or more abstract schemas (e.g., database schemas, directories in a file system, classification schemas, taxonomies, and so on). Different peers may use different schemas to organize the same data collection, and conversely the same schemas can be used to organize different data collections. We assume to deal with schemas with meaningful labels, where 'meaningful' means that their interpretation is not arbitrary, but is constrained by the conventions of some community of speakers/users[1]. We also assume that semantic peers need to compute mappings between its local schema and other peers' schemas in order to exchange data.

The first idea behind our approach is that *mappings must express semantic relations*, namely relations with a well-defined model-theoretic interpretation[2]. For example, we want to state that the two elements of the schema are equivalent, or that one is more/less general, or that they are mutually exclusive. As we will argue, this gives us many advantages, essentially related to the consequences we can infer from the discovery of such a relation.

The second idea is that, to discover such semantic relations, one must *make explicit the meaning* implicit in each element of a schema. Our claim is that addressing the problem of discovering semantic relations across schemas, where meaningful labels are used, as a problem of matching abstract graphs is conceptually wrong.

[1] In the following we show how this problem is extremely different from the problem of determining the similarity across different graphs.

[2] This is an important difference with respect to approaches based on matching techniques, where a mapping is a measure of (linguistic, structural, . . .) similarity between schemas (e.g., a real number between 0 and 1). The main problem with such techniques is that the interpretation of their results is an open problem: should we interpret a 0.9 similarity as the fact that one concept is slightly more general or slightly less general than the other one, or that their meaning 90% overlaps? See [1] for a more detailed discussion.

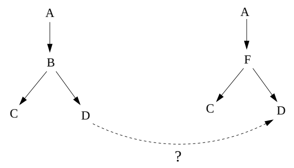

Fig. 1. Mapping abstract structures

To illustrate this point, consider the difference between the problem of mapping abstract schemas (like those in Figure 1) and the problem of mapping schemas with meaningful labels (like those in Figure 2). Nodes in abstract schemas do not have an implicit meaning, and therefore, whatever technique we use to map them, we will find that there is some relation between the two nodes `D` in the two schemas which depends only on the abstract form of the two schemas. The situation is completely different for schemas with meaningful labels. Intuitively, the semantic relations between the two nodes `MOUNTAIN` and the two nodes `FLORENCE` of structures in Figure 2 are different, despite the fact that the two pairs of schemas are structurally equivalent between them, and both are structurally isomorphic with the pair of abstract schemas in Figure 1[3]. This why we can make explicit a lot of information that we have about the terms which appear in the graph, and their relations (e.g., that Tuscany is part of Italy, that Florence is in Tuscany, and so on).

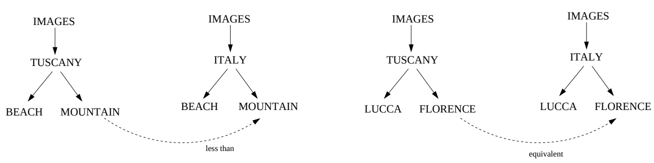

Fig. 2. Mapping schemas with meaningful labels

Using such an information, human reasoners (i) understand the meanings expressed by nodes: e.g., 'images of Tuscan mountains' (say P_1), 'images of Italian mountains' (say P_2), 'images of Florence in Tuscany' (say P_3) and 'images of Florence in Italy' (say P_4); and finally (ii) determine the semantic relations between nodes comparing the meanings, namely that $P_1 \subset P_2$ and $P_3 \equiv P_4$.

In [2], we claim that, for extracting such meanings and for comparing them, we need at least of three kinds of informations:

[3] Indeed, for the first pair of nodes, the set of documents we would classify under the node `MOUNTAIN` on the left hand side is a subset of the documents we would classify under the node `MOUNTAIN` on the right; whereas the set of documents which we would classify under the node `FLORENCE` in the left schema is exactly the same as the set of documents we would classify under the node `FLORENCE` on the right hand side.

Lexical knowledge: knowledge about the words used in the labels. For example, the fact that the word 'Florence' can be used to indicate 'a city in Italy' or 'a city in the South Carolina', and, conversely, to handle the synonymy;
World knowledge: knowledge about the relation between the concepts expressed by words. For example, the fact that Tuscany is part of Italy, or that Florence is in Italy;
Structural knowledge: knowledge deriving from how labeled nodes are arranged in a given schema. For example, the fact that the node labeled `MOUNTAIN` is below a node `IMAGES` tells us that it classifies images of mountains, and not, say, books about mountains.

Summarizing, the process of discovering semantic relations across meaningful labeled schemas can take advantage of exploiting the complex degree of semantic coordination implicit in the way a community uses the language from which the labels are taken[4]. The method is based on a procedure for explicitating the meaning associated to each node in a schema (notice that schemas such as the two classifications in Figure 2 are not semantic models themselves, as they do not have the purpose of defining the meaning of terms they contain; however, they presuppose a semantic model, and indeed that's the only reason why we humans can read them quite easily) and for comparing them. As we clearly show in the next section, this approach shifts the problem of semantic coordination from the problem of computing linguistic or structural similarities (what most other proposed approaches do) to the problem of deducing relations between sets of logical formulas representing the meaning of nodes belonging to different schemas.

2 The algorithm: CTXMATCH

In this section we show how to apply the general approach described in the previous section to the problem of coordinating *Hierarchical Classifications* (hereafter HCs), namely concept hierarchies [3] used for grouping documents in categories[5].

In our approach, we assume the presence of a network of semantic peers, where each peer is defined as a triple $\langle \mathcal{D}, \mathcal{S}, \langle L, O \rangle \rangle$, where: $\mathcal{D}$ is a set of documents; $\mathcal{S}$ represents the set of schemas used by the peer for organizing its data; and $\langle L, O \rangle$ is defined as a pair composed by a lexicon L and a world knowledge representation O. The structure of the semantic peer reflects the three levels of knowledge we showed before: $\mathcal{S}$ represents structural knowledge, L contains lexical knowledge, and O is world knowledge. Formally, L is a repository of pairs $\langle w, C \rangle$, where w is a word and C is a set of concepts. Each pair $\langle w, C \rangle$ represents the set of concepts C denoted by a word w. For example, a possible entry for a lexicon should express that the word 'fish' can denote

[4] Notice that the status of this linguistic coordination at a given time is already 'codified' in artifacts (e.g., dictionaries, but today also ontologies and other formalized models), which provide senses for words and more complex expressions, relations between senses, and other important knowledge about them. Our aim is to exploit these artifacts as an essential source of constraints on possible/acceptable mappings across structures.

[5] Some well-known examples of HCs are web directories (see e.g. the Google$^{\mathrm{TM}}$ Directory or the Yahoo!$^{\mathrm{TM}}$Directory), file systems and document databases in content/knowledge management systems.

at least two concepts: ‘an aquatic vertebrate’ and ‘the twelfth sign of zodiac’. An important example of this kind of repository is represented by WORDNET [4]. A world knowledge O expresses the set of relations holding between different concepts. For example, a world knowledge O should express that the concept ‘an aquatic vertebrate’ denoted by the word ‘fish’ stays in a *IsA* relation with the concept of ‘animal’ (‘fish is an animal’) and that the concept ‘the twelfth sign of zodiac’ denoted by the same word ‘fish’ stays in a *IsA* relations with a geometrical shape (‘fish is a geometrical shape’). Formally, world knowledge is a logical theory written is a specific language, as for example Prolog clauses, RDF triples, DAML/OIL, OWL.

Our method is designed for the following scenario: a peer A (called the *seeker*) needs to find new documents relative to some category in one of its HCs, S. Imagine that peer A knew that peer B (the provider) owns interesting documents, and imagine that B classify its documents by means of a HC S'. This problem can be solved in a standard way coordinating the two HCs. Formally, we define the problem of coordinating S and S' as the problem of discovering a mapping $\mathcal{M} = \{\langle m, n, R\rangle \mid m \in S, n \in S'\}$, where R is a semantic relation between m and n. Five relations are allowed between nodes of different HCs: $m \supset n$ (m is more general than n); $m \subset n$ (m is less general than n); $m \equiv n$ (m is equivalent to n); $m \cap n$ (m is compatible with n); $m \perp n$ (m is disjoint from n).

Algorithm 1.1 CTXMATCH(S,$S' L, O$)
▷ *Hierarchical classifications* S, S'
▷ *Lexicon* L
▷ *World knowledge* O
VarDeclaration:
  contextualized concept $\langle \phi, \Theta\rangle$, $\langle \psi, \Upsilon\rangle$
  relation R
  mapping M
for each pair of nodes m, n, $m \in S$ and $n \in S'$ **do**
  $\langle \phi, \Theta\rangle \leftarrow$ SEMANTIC–EXPLICITATION(m,S, L, O);
  $\langle \psi, \Upsilon\rangle \leftarrow$ SEMANTIC–EXPLICITATION(n,S', L, O);
  $R \leftarrow$ SEMANTIC–COMPARISON($\langle \phi, \Theta\rangle$, $\langle \psi, \Upsilon\rangle$, O);
  $M \leftarrow M \cup \langle m, n, R\rangle$;
return M;

The algorithm CTXMATCH takes as **inputs** the HC S of the seeker and the HC S', the lexicon L and the world knowledge O of the provider[6]. As we will show in the following, the lexicon L and the world knowledge O play a major part in determining the mapping between schemas. But, from the definition of semantic peer follows that each peer has its own lexicon and world knowledge. A consequence of this consideration is that the mapping returned by the algorithm expresses the point of view (regarding the mapping) of the provider, and, consequently, is directional: the seeker, *mutata mutandis*, can find a different mapping.

[6] In the version of the algorithm presented here, we use WORDNET as a source of both lexical and world knowledge. However, WORDNET could be replaced by another combination of a linguistic resource and a world knowledge resource.

The **output** of the algorithm will be a set M of triples $\langle m, n, R\rangle$, where R is the semantic relation holding between the nodes m and n.

The algorithm has essentially the following two main macro steps.

Steps 2–3: in this phase, called *Semantic explicitation*, the algorithm tries to interpret pair of nodes m, n in the respective HCs S and S' by means of the lexicon L and the world knowledge O. The idea is trying to generate a formula approximating the meaning expressed by a node in a structure (ϕ), and a set of axioms formalizing the suitable world knowledge (Θ). Consider, for example, the node `FLORENCE` in left lower HC of Figure 2: steps 2–3 will generate a formula approximating the statement 'Images of Florence in Tuscany' (ϕ) and an axiom approximating the statement 'Florence is in Tuscany' (Θ). The pair $\langle\phi, \Theta\rangle$, called *contextualized concept*, expresses, in our opinion, the meaning of a node in a structure.

Step 4: in this phase, called *Semantic comparison*, the problem of finding the semantic relation between two nodes m and n is encoded as the problem of finding the semantic relation holding between two contextualized concepts, $\langle\phi, \Theta\rangle$ and $\langle\psi, \Upsilon\rangle$.

Finally, step 5 generates the mapping simply by reiteration of the same process over all the possible pair of nodes $m \in S$ $n \in S'$ and step 6 returns the mapping.

The two following sections describe in detail these two top-level operations, implemented by the functions SEMANTIC–EXPLICITATION and SEMANTIC–COMPARISON.

2.1 Semantic explicitation

In this phase we make explicit in a logical formula[7] the meaning of a node into a structure, by means of a lexical and a world knowledge. In steps 1 and 2, the function EXTRACT–CANDIDATE–CONCEPTS uses lexical knowledge to associate to each word occurring in the nodes of an HC all the possible concepts denoted by the word itself. Consider the lower left structure of Figure 2. The label 'Florence' is associated with two concepts, provided by the lexicon (WORDNET), corresponding to 'a city in central Italy on the Arno' (`florence#1`) or a 'a town in northeast South Carolina' (`florence#2`). In order to maximize the possibility of finding an entry into the Lexicon, we use both a postagger and a lemmatizator over the labels.

In the step 3, the function EXTRACT–LOCAL–AXIOMS tries to define the ontological relations existing between the concepts in a structure. Consider again the left lower structure of Figure 2. Imagine that the concept 'a region in central Italy' (`tuscany#1`) has been associated to the node `TUSCANY`. The function EXTRACT–LOCAL–AXIOMS has the aim to discover if it exists some kind of relation between the concepts `tuscany#1`, `florence#1` and `florence#2` (associated to node `FLORENCE`). Exploiting world knowledge resource we can discover, for example, that '`florence#1` PartOf `tuscany#1`', i.e. that exists a 'part of' relation between the first sense of 'Florence' and the first sense of Tuscany.

[7] The choice of the logics depends on how expressive one wants to be in the approximation of the meaning of nodes, and on the complexity of the NLP techniques used to process labels. In our first implementation we adopted propositional logic, where each propositional letter corresponds to a concept (synset) provided by WORDNET.

Algorithm 1.2 SEMANTIC–EXPLICITATION(t, S, L, O)

▷ *t is a node in S*
▷ *structure S*
▷ *lexicon L*
▷ *world knowledge O*

VarDeclaration:
single concept $con[\,][\,]$
set of formulas Σ
formula δ

1 **for** each node n in S **do**
2 $\quad con[n] \leftarrow$ EXTRACT–CANDIDATE–CONCEPTS(n, L);
3 $\Sigma \leftarrow$ EXTRACT–LOCAL-AXIOMS$(t, S, con[\,], O)$;
4 $con[\,] \leftarrow$ FILTER–CONCEPTS$(S, \Sigma, con[\,])$;
5 $\delta \leftarrow$ BUILD–COMPLEX–CONCEPT$(t, S, con[\,])$;
6 **return** $\langle \delta, \Sigma \rangle$;

World knowledge relations are translated into logical axioms, according to Table 1. So, the relation '`florence#1` PartOf `tuscany#1`' is encoded as '`florence#1` $\rightarrow$ `tuscany#1`'[8].

WORDNET relation	axiom
`s#k` synonym `t#h`	`s#k` $\equiv$ `t#h`
`s#k` { hyponym | PartOf }`t#h`	`s#k` $\rightarrow$ `t#h`
`s#k` { hypernym | HasPart }`t#h`	`t#h` $\rightarrow$ `s#k`
`s#k` contradiction `t#h`	$\neg$(`t#k` $\wedge$ `s#h`)

Table 1. WORDNET relations and their axioms.

Step 4 has the goal of filtering out unlikely senses associated to each node. Going back to the previous example, we try to discard one of the senses associated to node `FLORENCE`. Intuitively, the sense 2 of 'Florence', as 'a town in northeast South Carolina' (`florence#2`), can be discarded, because the node `FLORENCE` refers clearly to the city in Tuscany. We reach this result by analyzing the extracted local axioms: the presence of an axiom such as '`florence#1` PartOf `tuscany#1`' is used to make the conjecture that the contextually relevant sense of Florence is the city in Tuscany, and not the city in USA. When ambiguity persists (axioms related to different senses or no axioms at all), all the possible senses are left and encoded as a disjunction.

Step 5 has the objective of building a complex concept (i.e., the meaning of a node label when it occurs in a specific position in a schema) for nodes in HCs. As described in [2], node labels are *singularly* processed by means of NLP techniques and translated into a logical formula[9]. The result of this first process is that each node has

[8] For heuristic reasons – see [2] – we consider only relations between concepts on the same path of a HC and their siblings.

[9] Although in this paper we present very simple examples, the NLP techniques exploited in this phase allow us to handle labels containing complex expressions, as conjunctions, commas, prepositions, expressions denoting exclusion, like 'except' or 'but not', multiwords and so on.

a preliminary interpretation, called *simple concept*, which doesn't consider the position of the node in the structure. For example, the simple concept associated to the node FLORENCE of the lower left hand structure of Figure 2 is trivially the atom florence#1 (i.e. one of the two senses provided by WORDNET and not discarded by the filtering). Then, these results are combined for generating a formula approximating the meaning expressed by a node *into a structure*. In this version of the algorithm, we choose to express the meaning of a node n as the conjunction of the simple concepts associated to the nodes lying in the path from root to n. So, the formula approximating the meaning expressed by the same node FLORENCE *into the HC* is $(\texttt{image\#1} \vee \ldots \vee \texttt{image\#8}) \wedge \texttt{tuscany\#1} \wedge \texttt{florence\#1}$.

Step 6 returns the formula expressing the meaning of the node and the set of local axioms founded by step 3.

2.2 Semantic comparison

This phase has the goal of finding the semantic relation holding between two contextualized concepts (associated to two nodes in different HCs).

Algorithm 1.3 SEM–COMP($\langle\phi,\Theta\rangle$,$\langle\psi,\Upsilon\rangle$,$O$)
▷ *contextualized concept* $\langle\phi,\Theta\rangle$, $\langle\psi,\Upsilon\rangle$
▷ *world knowledge* O

VarDeclaration:
set of formulas Γ
semantic relation R

1 $\Gamma \leftarrow$ EXTRACT–RELATIONAL–AXIOMS(ϕ, ψ, O);
2 **if** $\Theta, \Upsilon, \Gamma \models \neg(\phi \wedge \psi)$ **then** $R \leftarrow \perp$;
3 **else if** $\Theta, \Upsilon, \Gamma \models (\phi \equiv \psi)$ **then** $R \leftarrow \equiv$;
4 **else if** $\Theta, \Upsilon, \Gamma \models (\phi \rightarrow \psi)$ **then** $R \leftarrow \subset$;
5 **else if** $\Theta, \Upsilon, \Gamma \models (\psi \rightarrow \phi)$ **then** $R \leftarrow \supset$;
6 **else** $R \leftarrow \cap$;
7 **return** R;

In Step 1, the function EXTRACT–RELATIONAL–AXIOMS tries to find axioms which connect concepts belonging to different HCs. The process is the same as that of function EXTRACT–LOCAL–AXIOMS, described above. Consider, for example, the senses italy#1 and tuscany#1 associated respectively to nodes ITALY and TUSCANY of Figure 2: the relational axioms express the fact that, for example, 'Tuscany PartOf Italy' ($\texttt{tuscany\#1} \rightarrow \texttt{italy\#1}$).

In steps 2–6, the problem of finding the semantic relation between two nodes n and m (line 2) is encoded into a satisfiability problem involving both the contextualized concepts associated to the nodes and the relational axioms extracted in the previous phases. So, to prove whether the two nodes labeled FLORENCE in Figure 2 are equivalent, we check the logical equivalence between the formulas approximating the meaning of the two nodes, given the local and the relational axioms. Formally, we have the following satisfiability problem:

Θ	`florence#1` $\rightarrow$ `tuscany#1`
ϕ	(`image#1` $\vee \ldots \vee$ `image#8`) $\wedge$ `tuscany#1` $\wedge$ `florence#1`
Δ	`florence#1` $\rightarrow$ `italy#1`
ψ	(`image#1` $\vee \ldots \vee$ `image#8`) $\wedge$ `italy#1` $\wedge$ `florence#1`
Γ	`tuscany#1` $\rightarrow$ `italy#1`

It is simple to see that the returned relation is '$\equiv$'. Note that the satisfiability problem for finding the semantic relation between the nodes `MOUNTAIN` of Figure 2 is the following:

Θ	$\emptyset$
ϕ	(`image#1` $\vee \ldots \vee$ `image#8`) $\wedge$ `tuscany#1` $\wedge$ `mountain#1`
Δ	$\emptyset$
ψ	(`image#1` $\vee \ldots \vee$ `image#8`) $\wedge$ `italy#1` $\wedge$ `mountain#1`
Γ	`tuscany#1` $\rightarrow$ `italy#1`

The returned relation is '$\subset$'.

3 Conclusions and related work

In this paper we presented a new approach to semantic coordination in open and distributed environments, and an algorithm that implements this method for hierarchical classifications. The algorithm, already used in a peer-to-peer application for distributed knowledge management (the application is described in [5]), has been tested on real HCs (i.e., pre-existing classifications used in real applications) and the results are described in [6].

CTXMATCH faces the problem of semantic coordination deducing semantic relations between sets of logical formulas. Under this respect, to the best of our knowledge, there are no other works to which we can compare ours. However, there are three other families of approaches that we want to compare to: graph matching, automatic schema matching and semi-automatic schema matching. For each of them, we will discuss the proposal that, in our opinion, is more significant. The comparison is based on the following dimensions: (i) if and how structural, lexical and world knowledges are used; (ii) the type of returned relation. The general results of our comparison are reported in Table 2.

	graph matching	CUPID	MOMIS	CTXMATCH
Struct. knowl.	•	•	•	•
Lex. knowl.		•	•	•
Dom. knowl.				•
Relation returned	id of nodes	Value in $[0, 1]$	Value in $[0, 1]$	Semantic relation

Table 2. Comparing CTXMATCH with other methods

In graph matching techniques, a concept hierarchy is viewed as a labeled tree, but the semantic information associated to labels is substantially ignored. Matching two graphs G and G' means finding an isomorphic sub-graph of G' w.r.t. G. Some examples of this approach are described in [7, 8]. CUPID [9] is a completely automatic algorithm for schema matching. Lexical knowledge is exploited for discovering linguistic similarity between labels (e.g., using synonyms), while the schema structure is used as a matching constraint. That is, the more the structure of the subtree of a node s is similar to the structure of a subtree of a node t, the more s is similar to t. In case of equivalent concepts occurring in completely different structures, and completely independent concepts that belong to isomorphic structures, the match fails. MOMIS [10] is a set of semi–automatic tools for information integration of (semi-)structured data sources, whose main objective is to define a global schema that allows an uniform and transparent access to the data stored in a set of semantically heterogeneous sources. This integration is performed by exploiting knowledge in a Common Thesaurus together with a combination of clustering techniques and Description Logics. The approach is very similar to CUPID and presents the same drawbacks in matching hierarchical classifications.

References

1. Giunchiglia, F., Pavel, S.: Semantic matching. Proceedings of the workshop on Semantic Integration Sanibel Island (Florida, USA) (2003)
2. Bouquet, P., Serafini, L., Zanobini, S.: Semantic coordination: a new approach and an application. In Sycara, K., ed.: Second International Semantic Web Conference (ISWC-03). Lecture Notes in Computer Science (LNCS), Sanibel Island (Florida, USA) (2003)
3. Büchner, A., Ranta, M., Hughes, J., Mäntylä, M.: Semantic information mediation among multiple product ontologies. In: Proc. 4th World Conference on Integrated Design & Process Technology. (1999)
4. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. The MIT Press, Cambridge, US (1998)
5. Bonifacio, M., Bouquet, P., Mameli, G., Nori, M.: Kex: a peer-to-peer solution for distributed knowledge management. In Karagiannis, D., Reimer, U., eds.: 4th Intern. Conf. on Practical Aspects of Knowledge Management (PAKM-2002), Vienna (Austria) (2002)
6. Magnini, B.M., Serafini, L., Doná, A., Gatti, L., Girardi, C., , Speranza, M.: Large–scale evaluation of context matching. Technical Report 0301–07, ITC–IRST, Trento, Italy (2003)
7. Zhang, K., Wang, J.T.L., Shasha, D.: On the editing distance between undirected acyclic graphs and related problems. In Galil, Z., Ukkonen, E., eds.: Proc. of 6th Symposium on Combinatorial Pattern Matching. Volume 937., Finland, Springer-Verlag, Berlin (1995)
8. Pelillo, M., Siddiqi, K., Zucker, S.W.: Matching hierarchical structures using association graphs. Lecture Notes in Computer Science **1407** (1998)
9. Madhavan, J., Bernstein, P.A., Rahm, E.: Generic schema matching with cupid. In: The VLDB Journal. (2001) 49–58
10. Bergamaschi, S., Castano, S., Vincini, M.: Semantic integration of semistructured and structured data sources. SIGMOD Record **28** (1999) 54–59