Evolutionary Coincidence–Based Ontology Mapping Extraction

Vahed Qazvinian Hassan Abolhassani Seyed H. HAERI (Hossein)
Babak Bagheri Hariri

Web Intelligent Lab, Department of Computer Engineering, Sharif University of Technology
and School of Computer Science, Institute for Studies in Theoretical Physics and Mathematics (IPM)
Tehran, Iran
qazvinian@ce.sharif.edu, abolhassani@sharif.edu, shhaeri@ce.sharif.edu, hariri@ce.sharif.edu

Abstract

Ontology Matching is a process for selection of a good alignment across entities of two (or more) Ontologies. This can be viewed as a two phase process of: 1) applying a similarity measure to find the correspondence of each pair of entities from two ontologies, and 2) Extraction of an optimal or near optimal mapping. This paper is focused on the second phase and introduces our evolutionary approach for that. To be able to do so, we need a mechanism to score different possible mappings. Our solution is a weighting mechanism named *Coincidence-Based Weighting* – as explained in the paper. On that basis, a Genetic Algorithm is then introduced to create better mappings in successive iterations. We will explain how we code a mapping as well as our Crossover and Mutation functions. Evaluations of the algorithm is shown and discussed in the paper too.

1 INTRODUCTION

Semantic web is rather a new concept, and is defined so that the agents will be able to understand Web content and communicate through it, as like as human beings doing so now. Traditional knowledge-based systems were centralized, but on the other hand, semantic web is distributed and heterogenous. As a matter of fact, and according to (Mitra, Noy & Jaiswal 2003): "Information sources, even those from the same domain, are heterogenous in nature".
This heterogeneity has resulted in designing ontologies to lessen the difficulties of agents' understandings and communications. However, still another problem exists: ontologies themselves may have heterogeneity. This is when two ontologies are trying to express same knowledge or concepts but they use different languages or words (Euzenat & Valtchev 2004).
Ontology Alignment(OA) is a proposed solution to this problem by introducing a (proper) mapping of entities in two ontologies from two (different) domains. (Bouquet, Euzenat, Franconi, Serafini, Stamou & Tessaris 2004*a*) defines OA as:

> ... given two ontologies which describe each a set of discrete entities (which can be classes, properties, rules, predicates, etc.), find the relationships (e.g., equivalence or subsumption) holding between these entities.

Figure 1 shows a simplified OA framework. As shown in the figure, to extract an alignment, it is customary to first apply some measures (simple or complex) to reach to some initial similarity values. To this respect, there are already a vast amount of research works in the literature discussing about lexical and structural measures suitable for OA. Having such similarity values, the next problem is how to form an ideal mapping. We refer to this problem as Mapping Extraction. The goal is to find correspondence of entities among two ontologies such that the overall similarity value is maximal. Therefore it can be viewed as an optimization problem in which evolutionary approaches could be a legal solution.

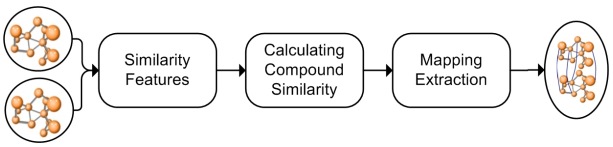

Figure 1: A Simplified Alignment Framework

In this paper we introduce a genetic based algorithm for the Mapping Extraction problem. First we have an explanation of the related works in section 2. Then in section 3 explanations about graph theoretical bases we use through the paper is given. To have a measure for calculation of how good a mapping is, the paper discusses coincidence based weighting in section 4. Our evolutionary algorithm is explained in section 5 and in section 6 evaluations are discussed. We also provide conclusions and explanations about our future works in Section 7.

2 RELATED WORKS

Unfortunately and as stated in (Bouquet, Euzenat, Franconi, Serafini, Stamou & Tessaris 2004*b*), works on ontology extractions are not so common. However, current researches on ontology mapping and its applications entails a large number of fields ranging from machine learning, concept lattices, and formal theories to heuristics, and linguistics. There are some similar works to match graphs, and trees (Hopcroft & Karp 1973, Papadimitriou & Steiglitz 1998), database schemas (Rahm & Bernstein 2001) and even in clustering compound objects with a machine learning technique (Bisson 1992). (Kalfoglou & Schorlemmer 2003) have come with a comprehensive review and presentations on the methods and approaches and the state of the art in ontology aligning.

Related works to our research are of the following two categories:

- The Ontology Alignment weighting and similarity measures. These works focus on introduction of new similarity measures between concepts of two ontologies, and a weight function to evaluate an alignment between two ontologies.
- The Ontology Mapping Extractions, in which the researches try to address the problem of alignment extraction and propose methods to find a

(proper) alignment among many different candidates.

There are also some works which address both problems simultaneously. We will have a quick review on each category in the following two subsections.

2.1 Similarity Measures and Ontology Alignment

There are considerable amount of previous works on similarity measures and ontology alignment. Some standards of metrics are acknowledged and defined as in the CommonKADS methodology (Schreiber, de Hoog, Akkermans, Anjewierden, Shadbolt & de Velde 2000), or OntoWeb EU thematic network (*OntoWeb. A survey on ontology tools.* 2002), which are partly endorsed by recognized bodies.
Also there have been some works on finding similarities of entities in two ontologies based on their structural standings. (Valtchev 1999) computes the dissimilarity of elements in a hierarchy based on their distances from closest common parent. The Upward Cotopic distance is introduced by (Maedche & Zacharias 2002) where they find dissimilarity of entities in hierarchies of ontologies. (Zhong, Zhu & Y. Li 1995) introduces a measure to calculate similarity of WordNet[1] concepts, i.e. a single hierarchy. In it, the similarity is computed based on the closest common parent and distance of the two entities from the root. On the other hands, some methods tend toward a trade-off between different features such as efficiency and quality, as in QOM (Ehrig & Staab 2003), and some have used approaches to integrate various similarity methods (Ehrig & Sure 2004).
Also compound metrics get use of simple measures by combining them, and hoping to improve the result of the mapping between two ontologies. One approach has been to define each measure as a dimension to find the Minkowski distance of two objects (Euzenat, Barrasa, Bouquet & Bo 2004). As introduced in (Euzenat et al. 2004) another approach for this problem has been a weighted average of features in which weights can be calculated by a machine learning technique. For example, Glue (Doan, Domingos & Halevy 2003) builds the similarity matrix by a machine learning approach. Also in APFEL (M. Ehrig 2005) weights for each feature is calculated using Decision Trees. The user only has to provide some ontologies with known correct alignments. The learned decision tree is then used for aggregation and interpretation of the similarities. Abolhassani et al. (Abolhassani, Haeri & Hariri 2006) introduces a new method for compound measure creation without any need to the mapping extraction phase.

2.2 Mapping Extraction

Previous works, do not especially cover the problem of alignment extraction. A method for ontology alignment extraction is proposed by (Dieng & Hug 1998) which examines linguistical features to compare two ontologies on the basis of a *IS-A* relationship. In (Melnik, Garcia-Molina & Rahm 2002), to extract a reasonable extraction, *Stable Marriage* (Gibbons 1985) problem is discussed.
There are some other approaches, e.g. a machine learning approach to the problem is discussed in (Doan, Madhavan, Domingos & Halevy 2003), and (Mitra et al. 2003) describe a probabilistic based model.
Staab et al. (Staab & Mdche 2002) have focused on structural and taxonomic comparison of two trees to extract an alignment, in which dissimilarity of each two concepts is calculated based on their superclasses and subclasses. Stumme et al. (Stumme & Mdche 2001) uses shared instances of two ontologies that are to be mapped, however this work ignores the properties of classes.
Zhdanova et al. (Zhdanova & Shvaiko 2006) expand the notion of ontology matching to a community-driven approach to enable web communities to establish and reuse ontology mappings to achieve an adequate and timely domain representation.
(Johnson, Cohen, Baumgartner, Lu, Bada, Kester, Kim & Hunter 2006) models inter-ontology relationship detection as an information retrieval task, where relationship is defined as any direct or indirect association between two ontological concepts.
Wand et al. (Wang & Gasser 2002) presents a specific formalization and algorithm for local interpretation of shared representations to build global semantic coherence for the distributed actions of individual agents, known as "Mutual Online Ontology Alignment".
LOM, as described in (Li 2004), is a semi-automatic lexicon-based ontology-mapping tool that supports a human mapping engineer with a first-cut comparison of ontological terms between the ontologies to be mapped, based on their lexical similarity and simple heuristic methods.

[1] wordnet.princeton.edu

3 PRELIMINARIES AND NOTATIONS

In this section, we define some necessary mathematical concepts which are used throughout this paper.

3.1 Basic Definitions

A graph G_i, by definition, consists of two sets: $V(G_i), E(G_i)$, where $V(G_i)$ is the set of vertices, and $E(G_i)$ is the set of edges. The size of a graph is $|V(G_i)|$, which is denoted by $|G|$. Let us assume that labels assigned to nodes are chosen from a finite alphabet Σ. Let $\lambda \notin \Sigma$ be a null character, and $\Sigma_\lambda = \Sigma \cup \lambda$.

3.2 Metric Space

According to (Rudin 1976), a set of points X along with a function, is said to be a *Metric Space* if the function associates a real number with any pair of points p, q, denoted by $d(p, q)$, and called the *distance* p, q, such that:

$\forall x, y \in O, \delta(x, y) \geq 0$ (*positiveness*)
$\forall x \in O, \forall y, z \in O, \delta(x, x) \geq \delta(y, z)$ (*maximality*)
$\forall x, y \in O, \delta(x, y) = \delta(y, x)$ (*symmetry*)

Any function δ, satisfying the above conditions is said to be a distance function or a metric. In fact, the distance of two concepts belonging to two different ontologies is described as the distance of their labels in a metric space, and usually this metric distance is described by a *distance function* described above.

3.3 Typed Graph

An ontology O_i, in this paper, is considered as a *typed graph* G_i. A *typed graph*, as defined in (Haeri, Hariri & Abolhassani 2006), is denoted by $G_i(V, E, T)$, for which E is of type: $E : V \times V \rightarrow T$. Members of T are all from Σ_λ. In such a graph an edge e of type t between vertices v_{i_j} and v_{i_k} is denoted by: $e(v_{i_j}, v_{i_k}) : t$.
A homeomorphism from a typed graph $G(V, E, T)$ to another typed graph $G'(V', E', T)$ is a one-to-one correspondence between V and V'. In this paper, each ontology O_i is modeled using a typed graph G_i where concepts of O_i are nodes of G_i, and the relations/properties of O_i are *typed* edges of the graph.

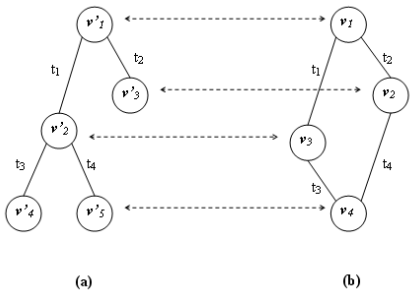

Figure 2: A sample alignment of two graphs G, G'

3.4 Edge Preservation

We will call an edge $e(v_{1_j}, v_{1_k}) : t \in E(G_{i_1})$ *preserved under the mapping* $\mathcal{M}$, if and only if there is an edge $e(\mathcal{M}(v_{1_j}), \mathcal{M}(v_{1_k})) : t \in E(G_{i_2})$. In other words, an edge e is preserved under mapping $\mathcal{M}$ if and only if $\exists e' \in E(G_{i_2}) : e' = (\mathcal{M}(v_{1_j}), \mathcal{M}(v_{1_k})), \mathcal{M}(e) = e'$, and is not preserved otherwise.

The preservation of edges between corresponding nodes is the key point to find an ideal mapping. In fact in an ideal alignment most of the edges of one ontology are preserved in the second one.

3.5 Ontology Alignment (OA)

In this section we will discuss our own understanding of a one to one alignment of two ontologies.
A one to one alignment of two ontologies O_{i_1}, O_{i_2} is denoted by $\mathcal{M} : O_{i_1} \rightarrow O_{i_2}$ and is a one to one correspondence between nodes of the two graphs of O_{i_1}, O_{i_2}: (G_{i_1}, G_{i_2}).
(Ehrig & Sure 2004) defines the mapping function in the following way:

- $\mathcal{M} : O_{i_1} \rightarrow O_{i_2}$
- $\forall v \in G_{i_1} : \mathcal{M}(v) = v'$ if $v' \in G_{i_2}$ and $\delta(v, v') < t$, for t being a threshold

v' is the *corresponding node* of v under the mapping $\mathcal{M}$.
Figure 2 illustrates a sample alignment for two example ontologies G, G'.

3.6 Our Formulation of OA

We denote the correspondence from ontology O_i to O_j while described by the concept of *typed graphs*, i.e. G_i to G_j by $\mathcal{M} : G_i \rightarrow G_j$. It is defined as follows:

1. $\forall v_i \in G_i$, v_i corresponds to only one vertex v_j in G_j (denoted by $\mathcal{M}(v_i) = v_j$), or does not correspond to any vertex in G_j (denoted by $\mathcal{M}(v_i) = null$). And if $v_1, v_2 \in V_i, v_1 \neq v_2, \mathcal{M}(v_1) \neq null, \mathcal{M}(v_2) \neq null$ then $\mathcal{M}(v_1) \neq \mathcal{M}(v_2)$
2. The correspondence of edges, is determined by the correspondence of nodes:
   $\forall e_i = (v_{i_1}, v_{i_2}) : t \in E(G_i)$: if $\mathcal{M}(v_{i_1}) = v_{j_1} \neq null$,$\mathcal{M}(v_{i_2}) = v_{j_2} \neq null$, and $e_j = (v_{j_1}, v_{j_2}) : t \in E(G_j)$ then e_i corresponds to e_j, $\mathcal{M}(e(v_{i_1}, v_{i_2})) = e(v_{j_1}, v_{j_2})$, else e_i does not correspond to any edge in G_j (denoted by $(\mathcal{M}(e(v_{i_1}, v_{i_2})) = null)$
3. Let $\mathcal{M}$ be a correspondence from G_i to G_j. We call $\mathcal{M}$ a *map* from G_i to G_j if $\forall v_i \in V_i, \mathcal{M}(v_i) \neq null, i.e. \mathcal{M}(v_i) \in V_j$
4. Each map, $\mathcal{M}$ from G_i to G_j has a *weight* and this weight is defined by the coincidence-based technique described in section 4.

The problem which is addressed in this paper is formulated as follows:
INPUT: Two ontologies together with a matrix, rows and columns of which, stands for concepts of ontologies, and each cell shows the distance of the two concepts as given by a distance measure.
OUTPUT: A proper alignment.
In what follows, we explain a technique to score possible mappings of ontologies, so called coincidence–based mapping, in the next section and then use this weight function to extract a proper alignment with evolutionary approaches in section 5.

4 COINCIDENCE BASED WEIGHTING

In this section we introduce and discuss a new weighting model for an alignment, with which we will later design our genetic algorithm.

The coincidence based alignment weight function is sufficiently discussed in (Haeri et al. 2006), and here, we will have an overview of it. Before talking about the weight itself, lets take some time, and discuss the matter.

There is a set of properties that we believe any mapping should convince. Considering a mapping $\mathcal{M}$, between two ontologies with graphs G_{i_1}, G_{i_2}, and two nodes $v_{1_j}, v_{1_k} \in V(G_{i_1})$ and their matches $\mathcal{M}(v_{1_j}), \mathcal{M}(v_{1_k})$, the weighting system should result a high weight if v_{1_j} is close to $\mathcal{M}(v_{1_j})$ and also v_{1_k} is close to $\mathcal{M}(v_{1_k})$ and when $e = (v_{1_j}, v_{1_k}) : t \in E(G_{i_1})$ is preserved under M. This case is considered to be the most desired one and should be given the highest value.
An alternative is when the edge is not preserved. Here, a negligible negative point should be given. The reason for negative point is the fact that, the edge is not preserved and the structural matching of the graphs is interrupted. In this case the nodes are very close but the edge is missing.
The farther any of the nodes is, from its match, the lower should be the positive value of the mapping. If the edge is preserved, we give this mapping a low positive value. But when the edge is not preserved, in fact it is an undesired mapping. So we give it a negative point. In this case not only the nodes are far from their matches, but also the edge is not preserved.

According to the above considerations there should be six different categories which are graphically shown in the Figure 3 (In the following explanations we assume that G, G' are graphs of two ontologies O, O' to be aligned. a, b are concepts from G, and a', b' from G'):

- **Category I.** a and a' are too close[2], and b, b' are close as well, and the edge between a, b is preserved under the matching process. This category is of much importance. This is because actually the two edges coincide too much. To clarify the point suppose the case when a and b are "means of communication" and "mail" respectively, and a', b' are "communication" and "email". The fact that there is an edge, (i.e.

[2] in terms of a distance function described before

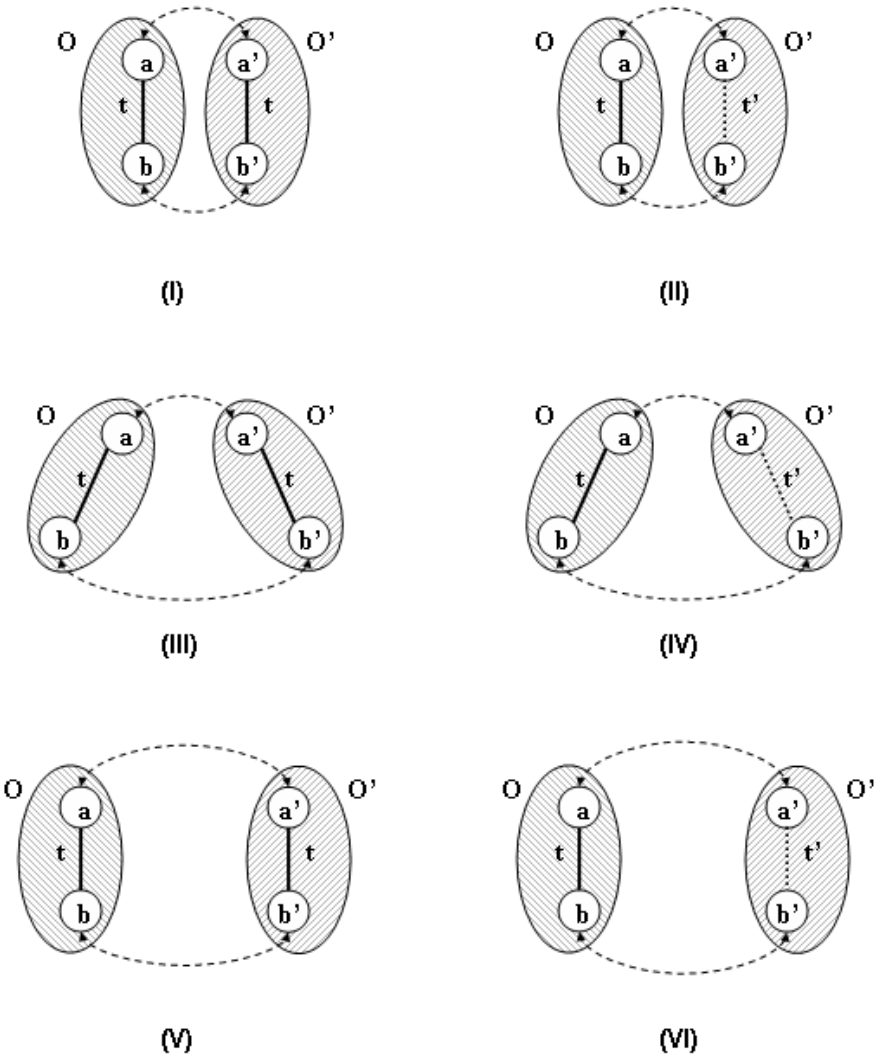

Figure 3: Different Properties of mappings in a metric space. Dotted edges with type t' show that there might be an edge of type t' or there might be no edge present. Dashed arrowed lines shows the mapping elements. (I) shows the first category where vertices (concepts) are close and the edge is preserved. (II) is the dual of category I, and different in that the edge is not preserved. (III) shows the case when the edge is preserved but only one of the endpoints of the two edges are close to each other. (IV) is dual case of category III, and different in that the edge is not preserved (V) The edge is preserved, but none of the endpoints are close to what they have been matched (VI) is the dual case of category V, but different in that the edge is not preserved.

$rdfs : type$) between both a, b and a', b' shows that the two edges coincide too much, and that the two ontolgoies are describing the same world.

- **Category II.** In this category, the two peers of an edge are close to their matches, that means, a is close to a' and b is close to b' as well. The only difference between this category and the previous one is that here, the edge is not preserved. As described in (Haeri et al. 2006) consider, e.g. when O describes the Glazing Technology, whilst O' is the ontology of simple glasses manufacturing studio. Let a, b be "Glass" and "Frame", and a' , b' the same respectively. Although $\delta(a, a'), \delta(b, b')$ are both small, the non-preservation of the edge (a, b) is a negative point. The fact that the vertices coincide, make us not to penalize this category much, because at least concepts are close to their matches and vertices coincide.

- **Category III.** In this category the edge is preserved but only one of the a or b is close to its match. This is good but not as much as the previous category. Consider two ontologies, describing two different worlds. Suppose O is describing a "High Tech manufacturing" while O' is describing a "Supermarket". Let a, b be "Laptop Computer" and "Product" respectively and a', b' be "Laptops" and "On Sale Item" respectively. The two ontologies are describing two totally different domains, whilst a and a' are close. So it seems as if such ontologies are getting close " from the side of a". We would like to give such category a large weight, yet smaller than that of the category I.

- **Category IV.** As category II can be considered to be the dual of category I, this category can be the dual of category III. The reason origins in that only one peer of the edge in O is too close to what it is matched to, yet the edge is not preserved under the matching. As it is clear in Figure 3 IV the edge between a, b is not preserved, and b is far from b'. The only positive point of such a matching is the fact that a and a' are close. As an example, just to make things clearer, consider O to be describing a hotel's services, where a is "egg" and b is "omelette" and O' is describing a "Supermarket" and a', b' are "egg" and "shampoo" respectively. This matching, which maps a to a' and b to b' is not desired and is most probably getting into mistake. However, this mistake should not be penalized as much as the mistake in category VI.

- **Category V.** The last two categories describe the situation where none of the peers of an edge is close to what it is matched to. Even though the edge might be preserved (as in category V) the two edges do not coincide at either end points. In other words, in these categories, both a, a' and b, b' are far from one another, and the difference is in the preservation of edges. The fact that the vertices are not close to their matches, is quite enough to make us indifferent about the edge preservation. Because even if the edge is preserved, the two edges are not that much coincident. Both cases are not desired and should obtain low points. A clear example of category V, would be when a is "Plant", b is "Water", a' is "Car", and b' is "Fuel". No need to discuss that this mapping is not a desired one.

- **Category VI.** This case is even worse in category VI than that of category V, where the edge is not even preserved. An example would be when a, b are "mammal" and "elephant" in O which is describing a "zoo" and a', b' are "glasses" and "frame" respectively in O' which is incidently describing a "glasses manufacturing company". There is neither a similarity between endpoints of the two edges, nor is there any preservation. The vertices in this category are mapped to what have no similarities, semantically.

According to the above cases, the following weigh function is suggested:

$$w(\mathcal{M}) = w_0(\mathcal{M}) - w_l(\mathcal{M}) - w_r(\mathcal{M})$$

$$w_0(\mathcal{M}) = \sum_{(v_1,v_2):t\in E(G)\ ,\ (\mathcal{M}(v_1),\mathcal{M}(v_2)):t\in E(G')} f(v_1)+f(v_2)$$

$$w_l(\mathcal{M}) = \sum_{(v_1,v_2):t\in E(G)\ ,\ (\mathcal{M}(v_1),\mathcal{M}(v_2)):t\notin E(G')} g(v_1)+g(v_2)$$

$$w_r(\mathcal{M}) = \sum_{(v_1,v_2):t\notin E(G)\ ,\ (\mathcal{M}(v_1),\mathcal{M}(v_2)):t\in E(G')} g(v_1)+g(v_2)$$

The functions f and g, referred to as *Normalization Functions* (Haeri et al. 2006) , are in the form:

$$f : R \to R^+$$

$$g : R \to R^+$$

f, g are related to the distance function. In fact, f should be a positive decreasing function, so that if $\delta(v, \mathcal{M}(v))$ grows, it decreases to reduce the positive point. And on the other hand g should be a positive increasing function to grow with the growth of $\delta(v, \mathcal{M}(v))$ to increase the negative point for that match. In any other cases, in one of the above six categories w will misbehave. Normalization functions are defined by tuning the system. This will be described again later.

According to (Haeri et al. 2006): "no [much] work is so far done on the problem of Ontology Alignment or Ontology Matching in which graph theoretic backbone of problem is scrutinized.". With the use of graph theory and such a modeling we believe that there is a vast area for new work on the problem of ontology aligning. The coincidence measure explained in this section is a step forward in this direction. We believe it can be used in various ways for the Mapping Extraction problem. The sole usage of it in this paper is introduced in the next section.

5 GENETIC ALGORITHM (GA)

This section describes the developed genetic algorithm.
Matching two general graphs in polynomial running time algorithms is impossible, because the problem in its general case is MAX SNP-Hard (Arora, Lun, Motwani, Sudan & Szegedy 1992). So a random search algorithm could be a good idea when designed carefully. This leads us to the idea of using genetic algorithms.
Genetic Algorithmic solutions are evolutionary algorithms, which will approach the final state by gradually improving the solution. Any problem which is solved by a GA, should first be coded in a way that it

can be easily handled in different parts of the GA. Each coding of a solution will form an individual. Some individuals which are stored and processed in each iteration form the population. the population improves in each step with the use of crossovers and mutations and the best individual will finally be reported as the answer of the GA. In the following subsections we explain about different parts of our GA solution.

5.1 Coding a Mapping

To code a mapping we use hashmaps (Cormen, Leiserson, Rivest & Stein 2001) in which keys are concepts of one ontology and entries are concepts of another one. This data structure help us easily manipulate a one-to-one alignment, with a search of concepts in $O(1)$. Entry for each key is actually the corresponding node of that key in the mapping. That is, if $v_i \in O$ is mapped to $v'_i \in O'$ then in the hashmap h we'll have the v_i as the key, and $h(v_i) = v'_i$.

5.1.1 Pairs

According to the coincidence-based weighting, we define a *Pair*, as two concepts from one ontology, between which there is a relation (So there is an edge in the graph of that ontology between them). Figure 2 shows the alignment of two ontologies, in which $(v_1, v_2), (v_1, v_3), (v_3, v_4), (v_2, v_4)$ are pairs of G.
A pair also has a weight according to the alignment it involves in.
Clearly speaking, a pair is a function of the form:

$$P : V \times V \times T \longrightarrow R$$

where V is the set of vertices in ontology graph G and T is a set of labels in Σ_λ. So an ontology in a matching has a limited number of pairs.
The weight of a pair depends on the alignment in which the ontology is involved. Let G_{i_1}, G_{i_2} be two graphs of two aligned ontologies, and $v_{1_j}, v_{1_k} \in V(G_{i_1})$. Also assume an edge between v_{1_j}, v_{1_k} to be of type t, $e_{1_{jk}} = e(v_{1_j}, v_{1_k}) : t$.
$P(v_{1_j}, v_{1_k}, t)$ in the alignment of two ontologies is given by:

$$P(v_{1_j}, v_{1_k}, t) = \begin{cases} f(v_{1_j}) + f(v_{1_k}) & e_{1_{jk}} \text{ preserved} \\ -(g(v_{1_j}) + g(v_{1_k})) & e_{1_{jk}} \text{ not preserved} \\ -\infty & \text{if } e_{1_{jk}} \notin E(G_{i_1}) \end{cases}$$

For a couple of concepts which do not form a *pair* the value of P function is set to be $-\infty$. Definition of pairs is useful in crossover function which will try to improve the structural matching.
In the alignment of two ontologies, $O_{i_1}(G_{i_1})$, $O_{i_2}(G_{i_2})$, say $\mathcal{M} : O_{i_1} \rightarrow O_{i_2}$, we also define the *weight* of a single concept from one ontology, $W(v_{1_j})$ where $v_{1_j} \in V(G_{i_1})$, as follows:

$$W(v_{1_j}) = \sum_{\substack{\text{if } v_{1_j} \in V(G_{i_1}), \mathcal{M}(v_{1_j}) \in V(G_{i_2}) \\ \forall v \in G_{i_1} : e(v_{1_j}, v) : t \in E(G_{i_1})}} P(v_{1_j}, v, t)$$

5.1.2 An Example

To make things clear about the definition of *pair* and its corresponding weights described above, we give an example on how to compute these weights.
In the Figure 2 we have:

$$\begin{aligned} P(v_1, v_2, t_2) &= f(v_1) + f(v_2) \\ P(v_1, v_3, t_1) &= f(v_1) + f(v_3) \\ P(v_3, v_4, t_3) &= -g(v_3) - g(v_4) \\ P(v_2, v_4, t_4) &= -g(v_2) - g(v_4) \\ P(v_1, v_4, t_i) &= P(v_2, v_3, t_i) = -\infty \\ W(v_1) &= (f(v_1) + f(v_2)) + (f(v_1) + f(v_3)) \\ W(v_2) &= (f(v_2) + f(v_1)) - (g(v_2) + g(v_4)) \\ W(v_3) &= (f(v_3) + f(v_1)) - (g(v_3) + g(v_4)) \\ W(v_4) &= -(g(v_4) + g(v_3)) - (g(v_4) + g(v_2)) \end{aligned}$$

Now, with these definitions, it is the time, to describe the steps of our genetic algorithm.

5.2 Initialization

As of any other Genetic Algorithms, a primary population is needed. A population is made up of some individuals each of which is a solution to the problem (a mapping in this problem). The start population, is initialized randomly, with an initial size of 1000 individuals. The ideal mapping can be reached more quickly if the initial individuals, are made on the basis of the labels of concepts, that is if v_{1_j} in G_{i_1} and v_{2_j} in G_{i_2} have same labels, then let v_{1_j} corresponds to v_{2_j} in the initial mapping.

5.3 Selection

In each iteration, we sort the 1000 individuals according to their fitness described in section 4 (coincidence based weight function), and we select the 500 best individuals as parents of the next step. From these 500 individuals, with the use of crossover and mutation functions (as we will see later), 1000 new individuals are created. These 1000 individuals are sent to the next iteration as parents.

Two crossover functions are designed, one based on pairs, and the other based on solitary vertices. In the following two subsections we explain each one in detail.

5.4 Crossover I.

In the first crossover, Crossover I, the pairs are in primary concern and best pairs from the parents are preserved in offsprings.
For every single node in the first ontology graph, the pairs of that node are examined in the two mappings (i.e. two parents), and the best pair, which has the highest value of weight is copied in the offspring. If a matched node in ontology graph G' is already assigned to some other node of G, the assignment is done to a random unassigned node.
In figure 4 two mappings as parents and the result of crossover I are shown. The pair (v_i, v_j, t) in parent 1, has greater weight than in parent 2. It is resulted from computing $P(v_i, v_j, t)$ in both alignments: In parent 1 we have $P(v_i, v_j, t) = f(v_i) + f(v_j)$ because the edge $e(v_i, v_j) : t$ is preserved under $\mathcal{M}$, but in parent 2 it is $P(v_i, v_j, t) = -(g(v_i) + g(v_j))$ because the edge $e(v_i, v_j) : t$ is not preserved. So the pair in parent 1 is chosen for offspring. Since f, g are positive functions, in the offspring we have $\mathcal{M}(v_i) = v'_i, \mathcal{M}(v_j) = v'_j$.

It is clear that in this crossover the pairs in offsprings are not worse than those of parents.

5.5 Crossover II.

In this crossover function, single nodes are compared according to their weights. As we described before, the weight of a single node in a mapping is the sum of weights of pairs in which, that node is included.
Consider two parents in two ontology graphs G_{i_1}, G_{i_2}.

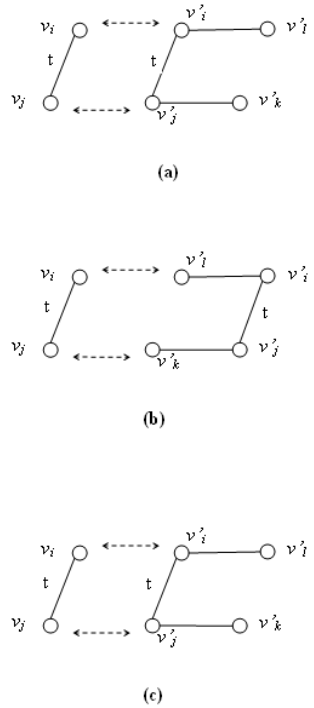

Figure 4: crossover I. (a) part of parent 1 mapping. (b) part of parent 2 mapping (c) part of offspring

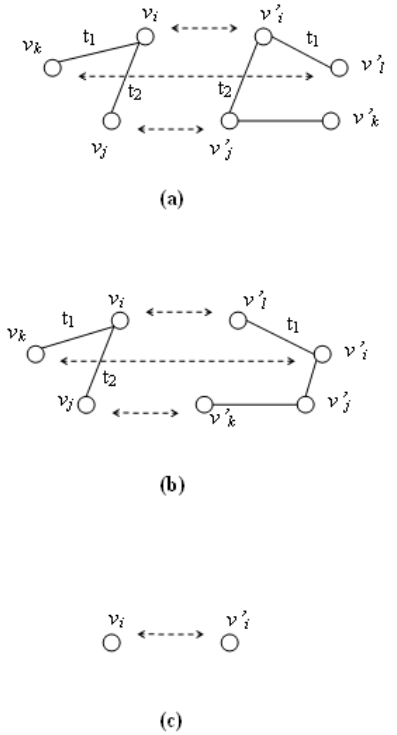

Figure 5: crossover . (a) part of parent 1 mapping. (b) part of parent 2 mapping (c) part of offspring

.

To make an offspring from two parents, for every node in G_{i_1}, say v_{1_j}, the mapping with larger $W(v_{1_j})$ is copied to the offspring. if $M(v_{1_j})$ in G_{i_2} is already assigned with some other node of G_{i_1}, then v_{1_j} is put in a reserved list. At the end of the complete iteration of nodes in G_{i_1}, the nodes in the reserved list are randomly mapped to the unassigned nodes of G_{i_2}. The random assignment is not done in the middle of an iteration to prevent nodes of G_{i_2} to be assigned to some random nodes that can be assigned to better nodes later in the iteration. So this random assignment is postponed until all nodes of G_{i_1} are examined to map to nodes in G_{i_2}.
As an example suppose $v_{1_m} \in V(G_{i_1})$ should be mapped to $v_{2_m} \in V(G_{i_2})$ and v_{2_m} is already mapped by some node from G_{i_1}, so if at that time we assign v_{1_m} to some random node like $v_{2_n} \in V(G_{i_2})$, it will prevent a possible good mapping of v_{1_n} to v_{2_n} later in the iteration. So this random assignment is delayed until no more assignment is possible.

In Figure 5 two mappings between two ontologies O, O' are shown, and we want to decide the match node for $v_i \in V(G)$ in the offspring. In parent 1 we have:
$W(v_i) = P(v_i, v_j, t_2) + P(v_i, v_k, t_1) = (f(v_i) + f(v_j)) + (f(v_i) + f(v_k))$
and in parent 2 we have:
$W(v_i) = P(v_i, v_j, t_2) + P(v_i, v_k, t_1) = -(g(v_i) + g(v_j)) + (f(v_i) + f(v_k))$

Again it should be noted that f and g are positive functions so the value of $W(v_i)$ in parent 1 (Figure 5 (a)) is greater than that of in parent 2 (Figure 5 (b)). So as it is shown in part c of the Figure, the corresponding node of $v_i \in V(G)$ in offspring is chosen by the mapping from parent 1, and therefore is $v'_i \in V(G')$.

This kind of crossover seems reasonable because the mapping of a single node in the offspring is not worst than that of the two parents. So by this assumption, little by little, mappings of nodes will converge to ideal ones.

5.6 Mutation

A proportion of the population are mutated with some probability, different in various iterations. In mutation of a mapping of two ontologies with graphs G_{i_1}, G_{i_2}, two random nodes from G_{i_1} are chosen, and their matches in G_{i_2} are substituted with each other. Let $v_{1_j}, v_{1_k} \in V(G_{i_1})$ are chosen randomly, also let $\mathcal{M}(v_{1_j}) = v_{2_j} \in V(G_{i_2}), \mathcal{M}(v_{1_k}) = v_{2_k} \in V(G_{i_2})$. In the mutation process we just substitute the match nodes of the selected ones. So the new mapping will be $\mathcal{M}(v_{1_j}) = v_{2_k} \in V(G_{i_2}), \mathcal{M}(v_{1_k}) = v_{2_j} \in V(G_{i_2})$.

5.7 Continuation

One important issue with any Genetic Algorithm, is how to get use of the crossover and mutation functions, and how to create the new population, based on the old one. In our solution, the two previously explained crossover functions are invoked on the i^{th} and $i+1^{th}$ parents to create two offsprings. The last parent is mixed with the first one to produce last two offsprings.
Our population as described before, contains 500 individuals. These individuals are sorted decreasingly, and the sorted array forms the parents of the current step. In each iteration, these 500 parents, with the help of a series of crossovers and mutations (as explained before) produce 1000 new individuals. The resultant array of individuals is then sorted and the best 500 of them are selected as parents of the next step. Figure 6 shows this process.

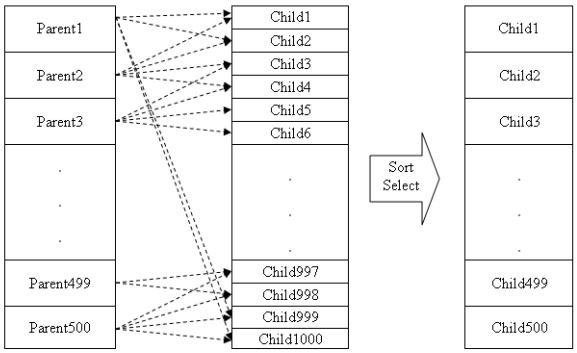

Figure 6: Population generation in each step of GA

5.8 End Condition

Basically there is no end for the execution of any evolutionary algorithm and specifically a genetic algorithm, where a user must pause the run process if she thinks the results are reasonable. However, to end the iteration of our GA, we used a threshold for convergence. The sequential GA is continued until the best mapping among all individuals in the population does not improve for more than 15 steps. Such mapping is reported as the answer for the problem of a proper alignment. The alignment process of two ontologies is then finalized.

6 EVALUATION

To evaluate our Genetic Algorithm, we designed three kinds of experiments. In the first experiment, we tested the Genetic Algorithm with diverse mutation probability. In the second experiment, we tried to align two identical ontologies (actually we aligned one ontology with itself). This experiment helped us examine the efficiency and accuracy of the algorithm, when two ontologies are more similar to each other. To verify our contribution, we came with a third experiment, in which we used a naive local search alignment method.

We already discussed about similarity measures in section 2. It is actually an important issue to pay attention in the ontology alignment and extraction process. There are some similarity measures proposed by some researchers. For example, (Euzenat & Altchev 2004) developed a similarity metric between concepts in OWL ontologies, which is a weighted combination of similarities of various features in OWL concept definitions: labels, domains, ranges of properties, restrictions on properties, types, IS–A relationships. (Mitra, Wiederhold & Decker n.d.) and (Mitra, Wiederhold & Decker 2001) use the combination of interactive specifications of mappings and heuristics to propose proper mappings. In this paper we used the Levenshtein (Levenshtein 1966) string–based similarity measure for the concepts, where the dissimilarity of any two concepts (from two ontologies) is calculated by the Levenshtein distance. This measure is suitable for our experiments since most of the heterogeneity in the Ontologies in our test collection comes from lexical difference. However, it should be noted that our coincidence-based weighting and hence our GA solution is independent of any similarity measure. To apply it to any other alignment, one can select another suitable measure for that domain and find similarity values to be used in our algorithm[3]

6.1 Limitations

The coincidence-based weighting has an innovative idea behind, however there are essential practical limitations to apply this method. The most important limitation is the available ontologies and test collections. Most of them do not have a large taxonomic structure and so the method does not have enough merit for them. However, in our search for a suitable test collection we found "Tourism" ontologies (*Tourism ontology FOAM* n.d.) a good one with approximately 340 classes and concepts.

6.2 Various Experiments Characteristics

For the tourism ontologies, an ideal alignment is included in the test collection. We use such information to calculate the precision (Baeza-Yates & Ribeiro-Neto 1999) for each experiment. Consider $\mathcal{M}$ to be a mapping: $\mathcal{M}O \rightarrow O'$. To find the accuracy of the method and calculate the precision, we need a *model* alignment $\mathcal{M}'$ which is formed and extracted by some expert. Let G, G' be the graphs for O, O' respectively. Also suppose, S is the set of vertices, v_i, in $V(G)$ where $\mathcal{M}(v_i) = v_i' = \mathcal{M}'(v_i)$. In other words:

$$\forall v_i \in V(G) : v_i \in S \Leftrightarrow \mathcal{M}(v_i) = \mathcal{M}'(v_i)$$

Now, the precision of the alignment $\mathcal{M}$ is given by:

$$Precision(\mathcal{M}) = \frac{|S|}{|V(G)|}$$

- Experiment 1
  As discussed previously, in this experiment we aligned "TourismA" with "TourismB". This is the main experiment to check the efficiency of our coincidence-based genetic algorithm.
  In this experiment, from each two parents, we made two offsprings one with the use of crossover I function and the other with crossover II. From the four different individuals (parents and the offsprings) we chose two best of them, to introduce as children of this amalgamation.
  Normalization Functions are as follows:

$$f(v) = \frac{1}{e^{\delta(v,M(v))}}$$

$$g(v) = \frac{1}{e^{\max(5,15-\delta(v,M(v)))}}$$

  These functions actually satisfy the characteristics expected from f, g (explained in Sec. 4). f is a decreasing function and decreases with the growth of δ and g is increasing. Exponential functions were chosen for f, g so that f, g would have close and comparable values. In fact, these functions match the discussions on positive and negative points for different categories of a coincidence based weight.

  - Experiment 1.1
    After the 1000 individuals are created, we mutate the **lower half** of them (with the mutation function described before) with a probability of **0.7**.

[3] as cited in related works section, there are some good works for selection of an appropriate measure for a domain which can be applied here.

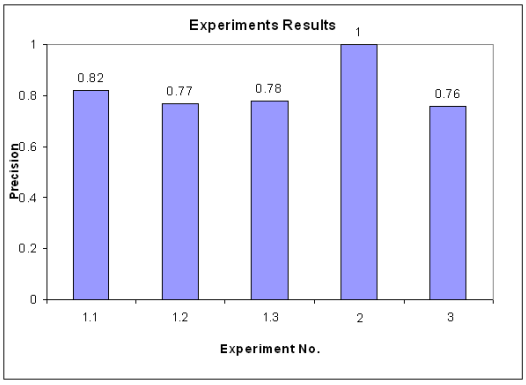

Figure 7: Precision result of experiments

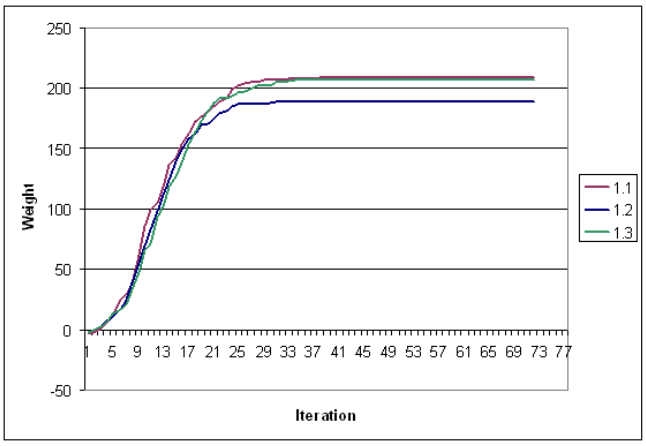

Figure 8: Convergence of results in GA

  - Experiment 1.2
    After the 1000 individuals is created, we mutate the **lower half** of the them (with the mutation function described before) with a probability of **0.3**.

  - Experiment 1.3
    Mutation was done on each one of the 1000 individual in the **all** of the 1000 children with the probability of **0.5**.

- Experiment 2
  In this experiment we are aligning "TourismA" with itself. This actually is a verification that shows how efficient the genetic algorithm will work, if two ontologies are more similar and actually more coincide.
  The generation summary is similar to the previous experiment, and mutation was done on the **lower half** of the individuals, with the probability of 0.5.
  Normalization Functions are similar to the previous experiment,

$$f(v) = \frac{1}{e^{\delta(v,M(v))}}$$

$$g(v) = \frac{1}{e^{\max(5,15-\delta(v,M(v)))}}$$

- Experiment 3
  This experiment actually provides a baseline comparison of the GA method with a naive local search method. In this part, we implemented a naive hill-climbing local search method. For the start point, we made an initial alignment. In this initial alignment, all concepts in "TourismA" is matched with concepts in "TourismB". For a node v_j in TourismA if there were a node v'_j with label $label(v_j)$ in TourismB, we matched v_j with v'_j. Otherwise we mapped v_j to a random node of TourismB.
  After that, in each iteration, the best single change (mutation) was preformed to improve the weight value of alignment. We iterated the method until almost 1000 steps, where, the results did not improve for more than 15 steps.

6.3 Results

Figure 7 shows the result of the above experiments according to the precision measure. As it is shown, with identical graphs, Genetic algorithm finds the best mapping and precision is 1. With other experiments, however, the result is a little inaccurate in comparison with the ideal alignment and precision is approximately 0.8.
In our experiments, the distance threshold, which we talked about in section 3.5, is set to be 4. We chose this number by experience, however there could be other solutions to determine this number, like machine learning techniques, etc. It is also possible to add one level of iteration to our genetic algorithm to test different values for distance threshold and select the threshold that results the best total weight for mapping. Since the labels for concepts in a typical Ontology normally does not exceed the length of an English (or other natural language) word the upper limit of the distance threshold can be said to be less than 20. So by execution of our genetic algorithm with different distance threshold vales (e.g. 1,2 to 20) we may find the best threshold value for an alignment task.
We also did an investigation on iteration number and convergence of the result in this genetic algorithm for Experiment 1. The results are shown in figure 8.

Figure 8 shows the weight of the best alignment, i.e. the best individual in the population, reached in each iteration of the running process in the Genetic Algorithm. The figure shows the best alignment weight for all three experiments, 1.1 in red, 1.2 in blue, and 1.3 in green. As it is clear, in all experiments the Genetic Algorithm converges before almost 35 iterations. The precision average for convergence is 79%. This means that the GA will not find a better solution, if it runs more than this number of iterations. Usually at such circumstances all individuals in the population, converge to a single individual and equal to each other, so that the combination of them in the form of crossovers will not improve the solution and will result in the same individuals. Besides, the mutation in most cases will result in a worse individual than a better one. The reason is quite clear. The optimal states (individuals) are much less than worse states, and the mutation is absolutely random. Mutation is mostly useful when the population is trapping in local maxima, and the mutation will move it to another point in the Genetic Algorithm.

It should be noted that in this area the main criteria of the goodness of an approach is the precision of the extracted mapping and reaching to a solution with the optimal time complexity is beyond the state of the researches.

7 CONCLUSION AND FUTURE WORK

In this paper, we discussed our method for Mapping Extraction in an Ontology Alignment framework. First, we introduced our modeling of an ontology with the concept of typed graphs. Then our coincidence-based measure for weighting of different candidate alignments were discussed. We would like to state that this experience gives us the impression that our solution which gets help from graph theory (and which was contrived for ontology alignment only) can be even further extended to a wider range of problems. Graph matching and similarity measures of graphs have many applications in machine vision, pattern recognition, bio–informatics, and etc.
We defined two cross-over functions based on coincidence-based measure. They are crafted in a way that ensure the new generations are not worse than previous ones. In our experiments our GA solution converges very rapidly, for example after approximately 40 iterations, which, in the order of magnitudes, is considerably better than a naive "candidate mapping generation and test" approach. This number can even be reduced more by choosing a biased initial population, where labels can be involved to choose better initial mappings.
There are also some weaknesses with Genetic Algorithms. One of them is the dependency of results to initial population. The more significant weakness is when the two ontologies are sparse graphs, or even worse is when there are like forests. In these cases the domain for crossover is not a soft one, and small changes in an individual in crossover or mutation might take it to a very far point. The reason for this anomaly is that in sparse graphs or jungles, the number of disconnected vertices is more, and therefore a small change in the alignment will map one node to another vertex which, more probably, will not form a coincident mapping. Roughly speaking, the more connected and taxonomic the two ontologies are, the better the results of the Genetic Algorithmic coincidence-based extraction will be.

In continuation of our research, work is now being done on tree-like ontologies (which seem being a most common form). Once we can align tree ontologies, we can model ontologies as trees and align the resulting trees. We are also interested in extending our theory and mechanisms for matching ontologies based on their various graphical shapes, properties of subgraphs, etc.

8 Acknowledgments

This research was in part supported by a grant from IPM (No. CS1385-4-01).

References

Abolhassani, H., Haeri, S. H. & Hariri, B. (2006), 'On ontology alignment experiments', *Webology* **3**(3).

Arora, A., Lun, C., Motwani, R., Sudan, M. & Szegedy, M. (1992), Proof verification and hardness of approximation problems, *in* '33rd IEEE symposium on the Foundations of Computer Science (FOCS)', pp. 14–23.

Baeza-Yates, R. & Ribeiro-Neto, B. (1999), *Modern Information Retrieval*, Addison Wesley.

Bisson, G. (1992), Learning in fol with similarity measure, *in* 'Proceedings of the 10th American Association for Artificial Intelligence conference, San-Jose (CA US)', pp. 82–87.

Bouquet, P., Euzenat, J., Franconi, E., Serafini, L., Stamou, G. & Tessaris, S. (2004*a*), Specification of a common framework for characterizing alignment, deliverable 2.2.1, Knowledge web NoE.

Bouquet, P., Euzenat, J., Franconi, E., Serafini, L., Stamou, G. & Tessaris, S. (2004*b*), 'Specification of common network framework for characterizing alignments', *Knowledge web NoE* **2.2.1**.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. & Stein, C. (2001), *Introduction to Algorithms, second edition*, MIT Press.

Dieng, R. & Hug, S. (1998), Comparison of "personal ontologies" represented through conceptual graphs, *in* '13th ECAI, Brighton (UK)', pp. 341–345.

Doan, A., Domingos, P. & Halevy, A. (2003), 'Learning to match the schemas of data sources: A multistrategy approach', *Machine Learning* **50**(3), 279–301.

Doan, A., Madhavan, J., Domingos, P. & Halevy, A. (2003), 'Ontology matching: A machine learning approach', *Handbook on Ontologies in Information Systems. Springer-Velag, 2003.* .

Ehrig, M. & Staab, S. (2003), Qom quick ontology mapping, *in* 'Proc. ISWC-2003.'.

Ehrig, M. & Sure, Y. (2004), Ontology mapping — an integrated approach, *in* '1st European Semantic Web Symposium (ESWS)', pp. 76–91.

Euzenat, J. & Altchev, P. (2004), Similarity–based ontology alignment in owl–lite, *in* 'The 16th European Conference on Artificial Intelligence'.

Euzenat, J., Barrasa, J., Bouquet, P. & Bo, J. (2004), 'State of the art on ontology alignment', *Knowledge Web, Statistical Research Division* .

Euzenat, J. & Valtchev, P. (2004), An integrative proximity measure for ontology alignment, *in* 'Proc. 3rd International Semantic Web Conference (ISWC2004) .'.

Gibbons, A. (1985), *Algorithmic Graph Theory.*, Cambridge University Press,.

Haeri, H., Hariri, B. & Abolhassani, H. (2006), Coincidence-based refinement of ontology alignment, *in* '3rd Intl. Conference on Soft Computing and Intelligent Systems'.

Hopcroft, J. & Karp, R. (1973), 'An n5/2 algorithm for maximum matchings in bipartite graphs', *SIAM Journal on Computing* **2**(4), 225–231.

Johnson, H. L., Cohen, K. B., Baumgartner, W. A., Lu, Z., Bada, M., Kester, T., Kim, H. & Hunter, L. (2006), 'Evaluation of lexical methods for detecting relationships between concepts from multiple ontologies', *Pac Symp Biocomput* pp. 28–39.

Kalfoglou, Y. & Schorlemmer, M. (2003), 'Ontology mapping: the state of the art', *The Knowledge Engineering Review* **18**, 1–31.

Levenshtein, V. I. (1966), 'Binary codes capable of correcting deletions, insertions and reversals.', *Sov. Phys. Dokl.* **6**, 707–710.

Li, J. (2004), Lom: A lexicon-based ontology mapping tool, *in* 'Proceeding of the Performance Metrics for Intelligent Systems (PerMIS04).Information Interpretation and Integration Conference (I3CON), Gaithersburg, MD.'.

M. Ehrig, S. Staab, Y. S. (2005), Bootstrapping ontology alignment methods with apfel., *in* 'Proceedings of the 4th International Semantic Web Conference (ISWC-2005), ser. Lecture Notes in Computer Science', pp. 186–200.

Maedche, A. & Zacharias, V. (2002), Clustering ontologybased metadata in the semantic web., *in* 'Proceedings of the 13th ECML and 6th PKDD'.

Melnik, S., Garcia-Molina, H. & Rahm, E. (2002), Similarity flooding: a versatile graph matching algorithm, *in* 'Proc. 18th International Conference on Data Engineering (ICDE), San Jose (CA US)', pp. 117–128.

Mitra, P., Noy, N. F. & Jaiswal, A. R. (2003), Omen: A probabilistic ontology mapping tool, *in* '4th international semantic web conference (ISWC 2005)', Vol. 3729, pp. 537–547.

Mitra, P., Wiederhold, G. & Decker, S. (2001), A scalable framework for interoperation ofinformation sources, *in* 'In The 1st International Semantic Web Working Symposium (SWWS01), Stanford University, Stanford, CA,'.

Mitra, P., Wiederhold, G. & Decker, S. (n.d.), 'The prompt suite: Interactive tools for ontology merging and mapping.', *International Journal of Human-Computer Studies, volume = .*

OntoWeb. A survey on ontology tools. (2002), available online : www.ontoweb.org/deliverable.htm.

Papadimitriou, C. H. & Steiglitz, K. (1998), *Combinatorial Optimization Algorithms and Complexity*, Prentice-Hall.

Rahm, E. & Bernstein, P. (2001), 'A survey of approaches to automatic schema matching', *VLDB Journal* **10**(4), 334–350.

Rudin, W. (1976), *Principles of Mathematical Analysis*, third edn, McGraw-Hill, New York.

Schreiber, G., de Hoog, R., Akkermans, H., Anjewierden, A., Shadbolt, N. & de Velde, W. V. (2000), *Knowledge Engineering and Management*, MIT Press.

Staab, S. & Mdche, A. (2002), 'Measuring similarity between ontologies', *Lecture notes in artificial intelligence* (2473), 251–263.

Stumme, G. & Mdche, A. (2001), Fca-merge: bottom-up merging of ontologies, *in* 'In Proc. 17th IJCAI, Seattle (WA US)', pp. 225–230.

Tourism ontology FOAM (n.d.). available online: http://www.aifb.uni-karlsruhe.de/WBS/meh/foam/ontologies/.

Valtchev, P. (1999), 'Construction automatique de taxonomies pour laide la reprsentation de connaissances par objets.', *Ph.D. Dissertation, Universite Grenoble. .*

Wang, J. & Gasser, L. (2002), Mutual online ontology alignment, *in* 'Proc. of the AAMAS 2002 Workshop'.

Zhdanova, A. V. & Shvaiko, P. (2006), Community-driven ontology matching., *in* 'Proc. of ESWC', pp. 34–49.

Zhong, J., Zhu, H. & Y. Li, a. Y. Y. (1995), Using information content to evaluate semantic similarity in a taxonomy, *in* 'Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)'.