Charles University in Prague

Faculty of Mathematics and Physics

MASTER'S THESIS

Michal Fiedler

Ontology Matching

Department of Software Engineering

Advisor: Mgr. Martin Nečaský

Study Program: Computer Science, Software Systems

Prague, December 2007

Contents

1. Introduction 7
2. The ontology matching task 9
  2.1. Background 9
  2.2. The matching process 10
  2.3. Comparison to schema matching 11
  2.4. Using the result 12
  2.5. Scenarios 12
    2.5.1. Ontology engineering 12
    2.5.2. Information integration 13
    2.5.3. Semantic web services integration 13
    2.5.4. Agent communication 14
3. Ontology matching techniques 15
  3.1. Element-level techniques 16
    3.1.1. String-based techniques 16
    3.1.2. Natural language processing techniques 17
    3.1.3. Thesauri and lexicons 18
    3.1.4. Constraint-based matching 19
  3.2. Structure-level techniques 20
    3.2.1. Specialization relations 20
    3.2.2. (General) relations 21
    3.2.3. Attributes 22
    3.2.4. Individuals 22
  3.3. Utilization of existing alignments 22
4. Matching systems 24

4.1. Matching paradigm 24
4.2. Combining elementary matchers 25
4.3. Dealing with non-local context 26
5. A survey of state of the art systems 28
5.1. Matching system evaluation 28
5.1.1. Evaluation measures 28
5.1.2. Ontology Alignment Evaluation Initiative 29
5.1.3. Method of this survey 29
5.2. Characteristics of surveyed systems 32
5.2.1. ASMOV 32
5.2.2. Falcon-AO 33
5.2.3. LILY 34
5.2.4. OLA2 35
5.2.5. Prior+ 36
5.2.6. RiMOM 37
5.3. Synthesis 37
6. Conclusions 39
6.1. Related work 39
References 40

Název práce: Ontology Matching

Autor: Michal Fiedler

Katedra: Katedra softwarového inženýrství

Vedoucí diplomové práce: Mgr. Martin Nečaský.

e-mail vedoucího: Martin.Necasky@mff.cuni.cz

Abstrakt:

Mnoho nezávislých systémů je postaveno nad ontologiemi. Takto vznikají ontologie, které často popisují stejnou oblast nebo se překrývají. Aby mohli uživatelé či samostaní agenti tyto ontologie využívat, musí nalézt vztahy mezi jejich prvky. To je úloha mapování ontologií.

Cílem této práce je nalézt nejúčinnější postupy používané při mapování ontologií. Za tímto účelem podává přehled metod užívaných pro mapování ontologií a zkoumá nejmodernější mapovací systémy. Na základě těchto poznatků pak identifikuje postupy společné nejúspěšnějším systémům.

Klíčová slova:

Ontologie, mapování ontologií

Title: Ontology Matching

Author: Michal Fiedler

Department: Department of Software Engineering

Supervisor: Mgr. Martin Nečaský

Supervisor's e-mail address: Martin.Necasky@mff.cuni.cz

Abstract:

Underlying heterogeneous systems, multiple ontologies are created that describe same or overlapping domains. Software agents (or tools in general) need to learn how to translate between such related ontologies. Only then will they be able to exploit the information from various sources simultaneously. This demand introduces the task of ontology matching. During the process of ontology matching, relationships holding between ontology entities are discovered.

The goal of this thesis is to discover the most successful approaches to ontology matching. To this end, a summary of matching techniques and approaches is presented, along with a survey that maps their usage in state of the art ontology matching systems and observes features common to recent systems.

Keywords:

Ontology, ontology matching, ontology alignment

1. Introduction

In his well-known article [1], Tim Berners-Lee proposed the concept of Semantic Web, a worldwide web of content meaningful for computers. In the following effort to fulfill this vision, ontologies were chosen to serve as means of storing information.

While the idea still remains largely unrealized, significant development was achieved in the field of ontology engineering [9][35]. Standards as RDF and OWL were presented, as well as a number of ontology processing tools. Areas such as government, libraries, healthcare or biotechnology have adopted the use of ontologies.

However, in this open and heterogeneous environment, multiple ontologies are created that describe same or overlapping domains. Software agents (or tools in general) need to learn how to translate between such related ontologies. Only then will they be able to exploit the information from various sources simultaneously. This demand introduces the task of ontology matching. During the process of ontology matching, relationships holding between ontology entities are discovered.

Over the past decade, approaches to ontology matching were studied intensely, building on techniques originating from the areas of database integration and knowledge representation. At present a number of systems is available that allow fully automated ontology matching.

The goal of this thesis is to discover the most successful approaches to ontology matching. To this end, a summary of matching techniques and approaches is presented, along with a survey that maps their usage in state of the art ontology matching systems and observes features common to recent systems.

The thesis is organized as follows.

Chapter 2 introduces the task of ontology matching and its various aspects.

Chapter 3 provides a classification and list of elementary ontology matching techniques.

Chapter 4 studies the architecture of ontology matching systems.

Chapter 5 introduces matching system evaluation, describes approaches used in state of the art systems in order to recognize their distinctive features.

Chapter 6 gives the conclusions and presents related work.

2. The ontology matching task

2.1. Background

Ontology is a data model that represents a set of concepts within a domain and the relationships between those concepts. Definitions of these entities include information about their meaning and constraints on their logically consistent application. Therefore, ontology may be used to reason about the objects within its domain.

Ontology contains information about concepts, relationships, attributes and individuals. **Concepts** are abstract groups or types of objects. **Relationships** describe relations between objects, they include specialization (is-a) relation and meronymy (part-of) relation. **Attributes** describe concepts they belong to. **Individuals** are instantiations of concepts, usually representing real-world objects. They contain values (attribute instances) and are connected with relationship instances.

Ontology languages are formal languages used to encode ontology specifications. The **Web Ontology Language** (OWL) [38] is the recent standard for ontology specification in the domain of the semantic web. In OWL terminology, concepts are called *classes*. A class may be specified as a *subclass* of another class, thus implementing the specialization relationship. Properties are used to define the content of classes. Properties fall into two categories; *object properties* represent general relations between two classes, while *datatype properties* represent attributes. Individuals are still called *individuals*. OWL builds on other standards, XML Schema datatypes are used, and the RDF/XML syntax is used to exchange OWL ontologies.

Taxonomy is a special case of ontology. Taxonomies focus on classifying objects, offering a hierarchy of concepts and usually missing attributes. Taxonomies are common; examples are web content directories or electronic marketplace catalogues. Several ontology matching techniques and systems are specialized for taxonomy matching.

2.2. The matching process

Following the definition provided in [9] and accepted by ontology matching researchers [10], a **mapping element** is a 5-tuple: <*id, e1, e2, n, R*>. *Id* is an identifier of the mapping element; *e1* and *e2* are entities of the respective ontologies *o1* and *o2*. *R* is a relation that holds between the mapped entities, possible kinds of relations are *equivalence*, *subsumption*, *overlapping* or *disjointness*; *n* is a *confidence measure*, a number (usually from [0,1]) specifying the belief that the relation R holds between entities *e1* and *e2*.

An **alignment** is a set of mapping elements for ontologies *o1* and *o2*.

The task of **matching** is to create an alignment *A* between two ontologies, *o1* and *o2*. To this end, the matching process may utilize supplementary input in addition to the matched ontologies. This includes an *input alignment A0*, containing relations that hold and should be enhanced by the matching process, and *external resources* like thesauri or knowledge bases. To complete the setup, a set of matching *parameters* may be supplied.

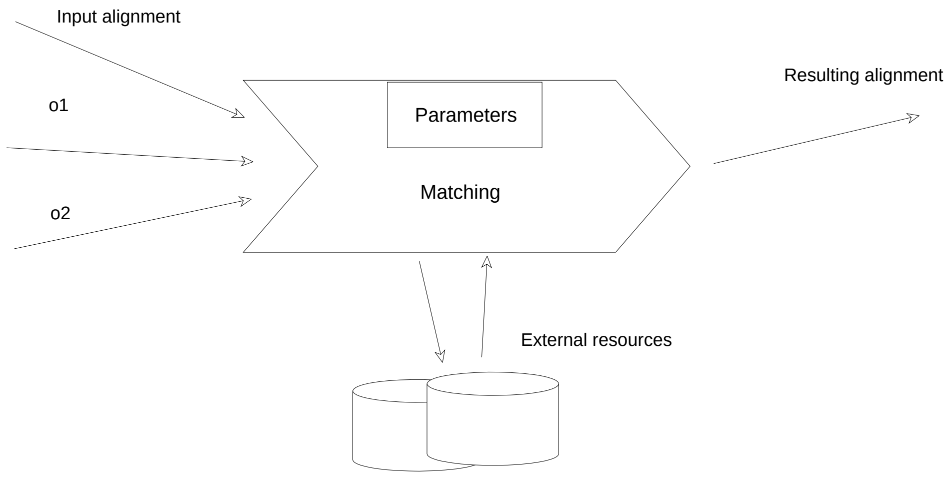

Figure 1. The matching task

Ontology maching system is a software system that performs the matching process. The degree of automation is an important characteristic of the matching system.

- **Interactive** matching is done by a human, who is given hints by the matching system.

- **Semiautomatic** matching systems present a set of mapping propositions to the user. The user approves them and may include additional mappings manually, this is repeated until the maching is complete.
- **Fully automatic** matching is done solely by the matching system; it is the only option, if user input is not available.

Present research focuses mainly to fully automatic matching, same applies to this thesis. In most scenarios, ontology matching is expected to be done without the presence of a qualified user, therefore fully automatic matching is required.

2.3. Comparison to schema matching

Before the need for ontology matching arose, matching was studied in other areas, particularly in the field of information integration. Information stored in information sources (typically databases or XML documents) is modeled according to a schema. In the process of data integration from various sources, schemas of the different sources must be matched. Numerous methods of schema matching techniques were developed in this area; a comprehensive list is given in [34].

Even though many techniques developed for schema matching prove useful for ontology matching as well, ontology matching differs in several characteristics. [31] compares the tasks of schema and ontology matching and discovers several differences, particularly

- *Explicit semantics*. Database schemas do not provide explicit semantics for their data, whereas in ontologies intended semantics is explicitly and formally specified.
- *Reusability*. Database schemas are usually defined over specific database, whereas ontologies are by nature reusable and frequently extend other ontologies.
- *Schema and data blending*. Databases make a clear distinction between schema and instance data, whereas in ontology modeling it is often difficult to distinguish between the ontology and the instances.

In addition to these differences in the nature of matching, schema and ontology matching differ in when they are applied. Schema matching is usually performed in the design stage, under the guidance of a schema designer. On the other hand, ontology matching is often performed dynamically in run time, when new ontologies are discovered and the need for alignment arises.

2.4. Using the result

Ontology matching is usually not the goal in itself. Possibilities of the further use of ontology alignments can be divided up into several classes; these are reasoning, merging, instance translation and mediation.

For the sake *reasoning*, mapping elements contained in the alignment are understood as bridge axioms connecting the aligned ontologies. With this connection, reasoning is performed over both ontologies simultaneously.

Ontology *merging* is a task similar to database integration. A new ontology is created, that contains entities from both (or all) merged ontologies. Overlapping parts of the individual ontologies are merged according to the discovered mappings.

Instance *translation* converts data of the source ontology into the language of the target ontology. Usually, transformation rules (or a transformation program in general) are generated.

Mediation uses the alignment to produce a mediator; a component that can translate queries that use the language of the target ontology into queries to the source ontology. In addition, it may perform instance translation. Using a mediator, client that only understands the target ontology language may access information stored in the source ontology.

2.5. Scenarios

To show the broadness of ontology matching application, this section presents a selection of scenarios, where matching participates. Ontology engineering, ontology integration, and catalogue integration are long known applications with counterparts in the realm of database engineering and integration. In these scenarios, ontology matching is generated at design time. On the other hand, other newly emerged applications are located in a highly dynamic environment and require run time matching.

2.5.1. Ontology engineering

Ontology engineering involves tasks of ontology design, implementation and maintenance. In many cases ontology matching is performed and the assistance of a matching system is necessary. In application design, reuse of existing ontologies is recommended. Often, such standard ontologies cover only parts of the domain of interest and must be assembled together. If underlying ontologies evolve, mapping can be used to identify the changes and translate between ontology versions.

For example, for the design of hospital ontology, authors may decide to reuse ontologies for diseases, anatomy, medications, persons and processes. These independent ontologies may overlap and matching can identify that a `symptom` in the disease ontology corresponds to `indication` in the medication ontology, or that an `actor` of a process is a `person`.

2.5.2. Information integration

The task of information integration is to make information from various independent sources accessible through a united interface. The role of matching is to suggest the structure of the united schema/ontology and to aid in the merging or generate a broker that translates between operations on the common schema and operations on the information sources.

The individual sources may contain information of the same domain. An example of such situation is the integration of multiple accounting systems after a company merger, or connecting student evidence of multiple faculties.

If the individual sources contain information from different domains, the goal of data integration is to define a common schema and For example, sources of patient information may be connected to operation records. Whenever an operation is executed, a record is sent to the broker. The broker decides that the type and time of the operation are recorded to the operation records, whereas the current health state is recorded in the patients file.

Catalogue integration is a special case of information integration. Catalogues are taxonomies and therefore, specialized taxonomy matching systems may be applied. The task of catalogue information is common in B2B electronic marketplaces.

2.5.3. Semantic web services integration

Web services are software systems accessible from the web. Web service discovery and integration is a process of finding appropriate services and composing them in order to achieve a desired goal. Web services are developed independently and may therefore use different schemas. Without the exploitation of semantics, this process requires human supervision to ensure that the web services are compatible and to define translation between formats they use.

Semantic services, on the other hand, use ontology representation languages that provide ways to describe the purpose and language of the service more precisely. Within semantic web services, ontology matching may be used for both finding appropriate services and detecting their compatibility, and generating a mediator that will connect these services together.

2.5.4. Agent communication

Agents are autonomous software entities that have the ability of communication in order to pursue goals they were designed for. The inter-agent communication languages specify the structure of messages interchanged by the agents, but the contents of the message is expressed in a knowledge interpretation language and follows the agent's ontology. If the agents' ontologies differ, mapping is necessary to enable understanding between agents. Agents may perform ontology on their own, or cooperate on matching negotiation to achieve a mutual agreement on the alignment [22].

3. Ontology matching techniques

This chapter gives a listing of elementary matching approaches used in ontology matching tools or suggested by researchers. Usually, ontology matching systems use a combination of these elementary techniques (called *matchers*) in order to exploit various resources and achieve the best possible results. Methods of such combination are discussed in chapter 4.

[34] gives a listing of schema matching methods and presents a classification of element- and structure-level approaches.

- *Element-level* techniques match ontology entities on their very own, without considering the context other entities provide.
- *Structure-level* techniques match entities in the context of their surrounding, exploiting various information stored in the ontology structure.

[36] extends the listing with techniques specific for schema-based ontology matching and introduces another dimension of method classification. In addition to the granularity aspect, matchers are classified according to their *input interpretation*; syntactic, semantic and external methods are distinguished.

- *Syntactic* techniques interpret the input with regard to its sole structure following a clearly stated algorithm.
- *Semantic* techniques use formal semantics to interpret the input and infer the result.
- *External* methods exploit external resources to interpret the input.

Other possible classification of matching approaches is classification by the *kind of input* exploited.

- *Internal* input is the contents of matched ontologies. The use of instance data may be used to further classify schema- and instance-level approaches.
- *External* input includes knowledge sources and existing mappings.

Last classification dimension to mention is the dimension of *properties of output* alignment. Approaches vary in the types of relations (equivalence, subsumption…) they are able to detect, other distinctive feature is whether the technique produces a graded answer (giving a probability that a relation holds) or a yes-no answer.

This thesis adopts the granularity classification. Besides element- and structure-level techniques, methods that utilize existing alignments are treated separately, as they do not fit in these categories.

Other aspects are either considered less important, or are perceived as rather features of matching systems. This applies to the matching paradigm (chapter 4.1) or the approach to utilization of non-local context (chapter 4.3).

3.1. Element-level techniques

3.1.1. String-based techniques

String-based techniques utilize names or textual descriptions of ontology entities. They work with string in a strictly syntactic manner, considering them to be nothing more than sequences of letters. Typically, these techniques compute a distance between strings; the greater the distance, the less similar are the compared strings.

Examples of string-based techniques are:

- **Edit distance** (or *Levenshtein distance*) – This measure computes number of letter manipulations (inserts, deletes, and substitutions) required to change one string to the other, normalized by the length of the longer string.

  For example, edit distance between terms `user` and `person` is 0.83 (5 manipulations: `user→ser→per→pers→perso→person;` normalized by 6), these terms are considered very distant by edit distance matching. Edit distance between `TopManager` and `Manager` is 0.3, these terms may be considered similar.

  This distance is used in [8] or [13]. Other similar distances, like *Hamming distance* or *Jaro-Winkler distance* are used in matching tools, too [28].

- **Prefix or suffix match** – This technique checks, whether one string is a prefix/suffix of the other or, alternatively, computes the length of their common prefix/suffix. Prefix matching is useful for detecting acronyms, while suffix matching helps detecting similarity between compound words (where, in English, the most important word comes last).

  E.g., the pair of term `sim` and `similarity` passes the prefix test, the pair of terms `WorkingMan` and `HardWorkingMan` passes the suffix test. Therefore, these pairs will be considered similar by these respective matchers.

  [8] and [13] make use of this technique.

- **N-gram** – This matcher counts the number of common n-grams (subsequences of n characters). It is proven that natural language strings are similar when the ratio of common n-grams is high. From the n-grams, distance may be computed as a cosine distance, or as the ratio of common n-grams.

  For example, trigrams (n-grams of the length of 3) of the term `human` are `hum`, `uma`, `man`. The distance between terms `human` and `woman` is 1/3, when using common trigram ratio.

  Examples of matching system featuring an n-gram matcher are [8] or [13].

3.1.2. Natural language processing techniques

Methods developed in the field of natural language processing utilize names or textual descriptions of ontology entities. In these strings, they detect words or sentences in a natural language (most often English).

Tokenization and lemmatization are *preprocessing techniques*, usually executed before the use of lexicons or before string matching.

- **Tokenization** – Tokenization divides names to sequences of tokens, every token is supposed to correspond to a word. This method uses punctuation, spaces and change of case to detect boundaries of tokens.

  Using this method, label `HardWorking_Man` will be transformed to the sequence of tokens <`hard`, `working`, `man`>.

  Tokenization is used in [13].

- **Lemmatization** – Lemmatization is a morphological method, which assigns basic form(s) to words.

  E.g., for the term `employees`, lemmatizer returns the form `employee`.

  [13] makes use of this technique.

- **Parsing sentences** – This technique builds a parse tree for given sentence. In ontology matching, the parse tree may be used to select more important words in entity descriptions.

  For example, in the sentence "`Account stores customer's precious money that he entrusted to the bank`", sentence parser may select the list of words <`account`, `stores`, `money`> for further processing.

  No recent systems are known to utilize sentence parsing.

- **Filtering** – This technique searches tokens for a list of known words. When found, such tokens may be discarded in the case of words not affecting the meaning, like articles. Other tokens (e.g. "and" or "non-") change the meaning of surrounding tokens and may be treated accordingly.

For example, [24] uses filtering to eliminate articles, prepositions and conjunctions. [4] chooses a gentler approach, interpreting coordinating conjunctions and commas as disjunctions, chosen prepositions (like `in` or `of`) as conjunctions and exclusive expressions (like `except` or `but not`) as negations.

3.1.3. Thesauri and lexicons

Thesauri and lexicons can be used to obtain meanings of terms, thus reconstructing the semantics contained in the name or textual description of an entity. A *thesaurus* is a listing of words with similar, related, or opposite meanings; a *semantic lexicon* is a listing of words, with their possible meanings. The usage of thesauri and lexicons belongs to natural language processing techniques, but the exploitation of terms' semantics sets it apart.

Most often, WordNet [27] is used – a semantic lexicon for the English language. WordNet lists *senses* (possible meanings of words) and groups them into *synsets* (groups of synonymous words). Synsets are connected with one another via a number of different relations, including *hyponymy* (kind-of or is-a relation) or *meronymy* (part-of relation). By hyponymy relationships, nouns and verbs are organized into hierarchies.

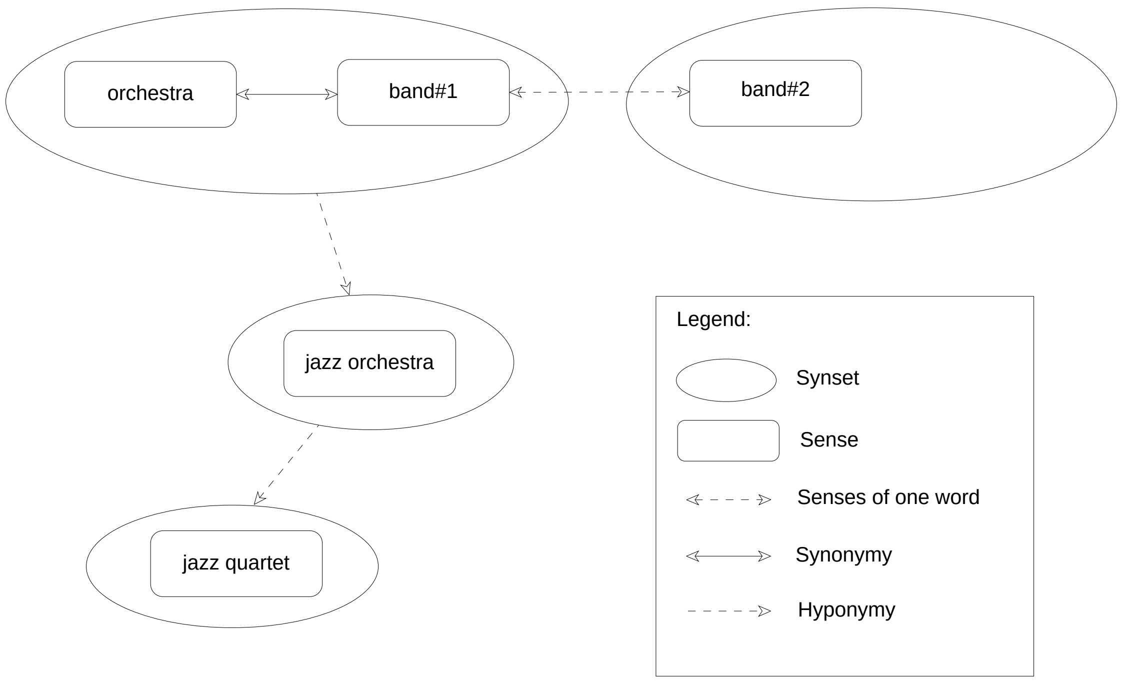

Figure 2. Thesaurus fragment

Figure 2 shows a thesaurus fragment. This fragment contains two senses of the word `band`. One of the senses is in a common synset with (the sense of) the word `orchestra`. `Jazz orchestra` is specified to be a kind of `orchestra` through the use of hyponymy relation. In addition to being hyponym of `orchestra`, `jazz orchestra` is the hypernym of `jazz quartet`.

Other lexicons may vary in the amount of the information contained. *Domain specific thesauri* are a special case [8], storing non-common knowledge, such as names of companies and products or abbreviations. Possible methods of lexicon utilization include:

- **Synonym lookup** – Synonyms are marked as equivalent. With the use of simple thesauri, this is the only method available.

  For example, the thesaurus may contain information that `student` and `pupil` are synonyms. Therefore, when these terms are encountered during matching, they are marked as equivalent by the synonym matcher.

- **Hierarchy-based matching** – A hierarchy-based matcher computes a distance between terms by measuring the number (and possibly the type) of arcs traversed between them in lexicon hierarchy. Different types of relation may be assigned different weights, the resulting distance being a function of the individual weights.

  Following the example from Figure 2 and hierarchy-based matching mechanisms presented in [6], the similarity of terms `band` and `jazz quartet` is computed as follows. Hyponymy relations holding between the pairs of synsets for `<band, jazz orchestra>` and `<jazz orchestra, jazz quartet>` are assigned a weight of 0.8. The resulting similarity is computed as the product of relation weights; the similarity value is 0.64.

- **Relationship composition** – This technique transforms the chain of relations between terms into a relation in the sense of ontology alignment. In some cases, like a sequence of hyponymy relations, relationship composition preserves more information than hierarchy based matching.

  Again using the example from Figure 2, the hyponymy relationships are composed, resulting in a hyponymy relationship between the terms band and jazz quartet. This relationship is then translated to a specialization relationship between ontology entities labeled by these terms.

  Relationship composition is used in [4].

3.1.4. Constraint-based matching

While the above described techniques are universal, ontology entities may contain further information that can be exploited specifically. Constraint definitions belong to this kind of information. Constraint-based matching is only possible for ontology definition languages that allow constraint specification.

- **Datatype comparison** – attributes are declared with a datatype they allow. Since the basic datatypes are specified by the used ontology language, they can be compared objectively, resulting in a datatype similarity measure. The similarity should be maximal when the types are the same, lower for compatible types where a conversion is possible (e.g. between `float` and `int`) and lowest when the types are incompatible.

  For example, [11] introduces a matching system tailored specifically for OWL. Datatype similarity is determined by comparison of sets of possible values, basic datatype relationships are taken from the definition of XML Schema [39].

  Consider an example, where attribute `age` is of type `PositiveInteger`, attribute `price` is of type `Float` and attribute `name` is of type `String`. Types `PositiveInteger` and `Float` are considered compatible, whereas types `String` and `PositiveInteger` are considered incompatible. Therefore, the attribute `age` is detected as more similar to `price`, than to `name`.

- **Multiplicity comparison** – In addition to datatype comparison, collection datatypes (lists, sets, multisets) may be compared regarding the cardinality constraints applied.

  [11] is an example of a matching system that utilizes multiplicity comparison. Both upper and lower bounds are compared, the resulting measure is 1 if both limits correspond, 0.5 if only one does, 0.35 if limits do not match, but there is inclusion between the intervals and 0 otherwise.

3.2. Structure-level techniques

The amount of information stored inside of an ontology entity itself is limited, as are methods for matching entities without the context of their surroundings. However, ontology entities are connected to each other and this information is exploited by structure-level techniques.

This section concentrates on matching concepts, which is the main goal in ontology matching. Aside from its name, a concept is merely a collection of attributes, individuals and relations to other concepts. Techniques will be presented grouped by the type of connections they exploit.

3.2.1. Specialization relations

- **Similarity transfer between sub- and super-concepts**. The specialization relation specifies that one entity is a super-concept of the other concept. If super-concepts are the same, the sub-concepts are similar to each other. Vice versa, if sub-concepts are the same, the super-concepts are similar to each other.

This technique is used in [11].

- **Disjointedness between sub-concepts**. Usually, there are multiple sub-concepts for a super-concept. In taxonomy, concepts with a common super-concept are conceived as disjointed. Taxonomy-matching tools thus explicate this quiet assumption. In a situation, where super-concepts are the same and multiple sub-concepts are similar to one sub-concept do not detect equivalence between the sub-concepts.

  [4] is an example of a matching system exploiting this technique.

- **Super-concept as a sum of sub-concepts**. In taxonomies, the set of sub-concepts defines the super-concepts. Machine learning methods may be used to match super-concepts using their sets of sub-concepts.

  This technique is used in [4]; [40] studies the use of naïve Bayes classifier for implementation of this technique in order to perform taxonomy matching.

3.2.2. (General) relations

- **Relation membership**. For a relation, range and domain are defined. Given a pair of relations and two pairs of concepts (ranges and domains) in two ontologies, if two of the pairs are similar, the third pair may be similar as well.

  As an example of further development of this technique, consider this situation. `Flat` and `person` are connected through the relation inhabits in first ontology, and `house` and `human` are connected through the relation `inhabitedBy`. If `flat` is detected to be similar to `house` and `person` to `human`, a matching system may discover, that `inhabitedBy` is in fact a reverse relation to `inhabits`.

  This technique is used in [11].

- **Composition**. Relations are often conceived as part-of relations, implementing composition. If the components of two concepts are similar, the concepts themselves are perceived structurally similar.

  [8], [11] or [24] are examples of matching systems that use this method.

- **Bounded path matching**. Given two pairs of concepts evaluated as similar, bounded path matchers study the paths between them. Concepts lying on the same position along these paths are then considered similar. Bounded path matching is applicable for both general and specialization relations.

  This technique was first introduced in [32].

3.2.3. Attributes

Attributes contribute to the concepts' structure. Same as with part-of relations, if structure of concepts is similar, they are considered similar as well.

Examples of matching systems utilizing this approach are [8], [11] or [24].

3.2.4. Individuals

In addition to entities constituting the schema, the ontology may contain individuals (instances). These may be instances of concepts (sometimes, the term individual is narrowed to this meaning), instances of relations and instances of attributes (or simply values). Instances may be matched on their own using most of the described techniques [11]. Furthermore, there are several methods of inferring entity similarity from the set of its instances.

- **Unique value count, Frequency distribution, Histogram**. These measures known from database management may be applied; the more similar the measures, the more similar are the corresponding entities.

  Use of these techniques is described in [20].

- **Instance set overlap**. In a situation when both ontologies describe the same individuals, sets of instances may be compared using a set similarity distance.

  [33] uses this kind of measure, namely the *Jaccard similarity coefficient*.

3.3. Utilization of existing alignments

If a matching was performed previously, or a mapping is supplied with the ontology, the existing alignment may be used as grounding for further matching. Several methods are proposed or used in recent tools. Some presume the existence of an (external) alignment repository, other expect the ontology itself to contain links to external concepts.

- **Alignment reuse**. An existing alignment of previously matched ontologies can be used when trying to match one of them to another ontology. This method may vary from as simple as finding the exact mapping in an *alignment repository* up to a detection of incomplete mapping paths between multiple ontologies. For example, when a matching of ontology O to O2 is requested and the alignment between ontologies O1 and O2 is supplied, the matcher may instead choose to match O to O1 first and only after that compute the resulting mapping as a composition. This approach follows the idea that if there is a mismatch between a pair of ontologies, it may not be present between another pair of ontologies, or it may be easier to detect. In addition, it may accelerate the matching process by eliminating unnecessary matching steps.

[1] is an example of a matching system utilizing alignment reuse.

- **Fragment-based matching** – For large ontologies, the *divide and conquer* paradigm may be applied. Ontology is divided into fragments, similar fragments are detected and matched an in the end, alignment of fragments is used as the input for whole ontology matching.

  This technique is used in [1].

- **Upper level ontologies** – Several upper level ontologies have been proposed, some of them are becoming accepted standards. These very general ontologies define concepts as persons, physical objects, events or space. Examples of such upper level ontologies are SUMO [29] or DOLCE [12]. The goal is to have domain-specific ontologies extend these general ontologies. In this manner, even if ontologies differ, they still have a common grounding in the upper level ontology and this grounding can be used to match their concepts.

  This approach is proposed in [30] and [36], but is not yet used in recent matching systems.

- **Mapping to a standard ontology** – Domain-specific standard ontologies exist in various areas. If two independent ontologies are both mapped to such reference ontology, matching between them is made much easier. For example, the use of *Process Specification Language* as a standard ontology for process representation ontologies is studied in [14].

4. Matching systems

The previous chapter introduced elementary techniques used in ontology matching systems. Every elementary matcher employs only a subset of information contained in the ontology or in external resources. To perceive the whole picture, matching systems need to use these techniques in a combination. This is a common feature of all state of the art ontology matching systems. However, they differ in several aspects. The degree of automation was already discussed in chapter 2.2. Other distinctive aspects will be examined in the rest of this chapter.

4.1. Matching paradigm

Matching paradigm is defining for the way matchers are combined (and implemented). Still, the techniques are the same, only the interpretation differs. The similarity paradigm is prevailing, but semantic paradigm has been successfully used, too.

Similarity paradigm considers ontology matching task as a task of measuring and maximizing similarity. Matchers measure similarity as a numeric value, combination of matchers constitutes of a mathematic calculation. At the end of the matching process (or at the end of its phases) pairs of entities with high similarity are designated as candidate mappings.

The similarity paradigm is prevailing among recent ontology matching systems. Examples of systems following this paradigm are [1], [6] or [11].

Semantic paradigm interprets the information from the ontologies and external resources as facts of a logical model. In this model, deductive methods are used to discover the mappings. In semantic paradigm, elementary matchers perform transformation from internal or external information to facts of the chosen model. The combination of matchers is then a transformation from a set of facts to another set of facts.

Propositional satisfiability (SAT) is an example of approaches following the semantic paradigm. This approach, applied in [13] and [4] systems, translates the ontology and domain knowledge to a basis of logical formulas. Over this basis the validity of relations between pairs of ontology entities is checked using a SAT solver. In [5], the use of description logic techniques is studied as well.

4.2. Combining elementary matchers

There are two basic ways of matcher combination: composition and iteration. This combination may be done repeatedly, thus combining combined matchers.

Composition takes input from multiple matchers executed at the same time to compute the (combined) similarity.[1] In existing tools, various methods are used to compute the combined similarity. [11] uses a *weighted sum* with fixed weights acquired by experimentation, [2] and [28] use weighted sum with weights assigned by the *Dempster-Schafer* rule. [1] gives the option to use the weighted sum, *average, maximum or minimum* of measured similarities.

Above described approach to composition runs the individual matchers independently, the combination is external to the elementary matchers. Another option is to not combine results of the matchers, but the matching techniques themselves. In this case, the combination is internal and information from various sources can be exploited at the same time. Such matchers are often referred to as *hybrid matchers* [8].

Iteration combines matchers into a sequence. Results from the execution of one matcher are part of the input for the execution of the second matcher. Often, structure-level techniques are used to enhance mappings created by element-level matchers. For example, [32] first performs string-based matching and uses equivalence candidates as anchors (path boundaries) for the following path-based matching.

Semiautomatic tools usually work in iterations, after each matching phase they present candidate mappings to the user. Approved mappings (and disapproved mappings as well) are then used as input for the next iteration.

Figure 3 shows the architecture of an example matching system, the flow of the matching process is from up to down. The (hypothetic) matching system consists of four pipelines. In these pipelines basic matchers are connected via iteration. The first sequence handles language- and sense-based matching. The second line consists of only one string-based matcher; this part serves as fallback option for name matching. In the case when a term is not found in the thesaurus and hierarchy-based matching may not be finished, the result of edit distance

[1] The term similarity is not accurate for the semantic paradigm. However, it is used for simplicity, since the similarity-based approach is considered to be more intuitive

matcher is used instead. The other two sequences handle structure-level matching, namely composition. First of them matches attribute datatypes and distributes the similarity to concepts; the last sequence matches the composition of concepts by comparing part-of relations they include. In the end, these four sequences are connected via composition and the resulting similarity values are computed using weighted sum with fixed weights.

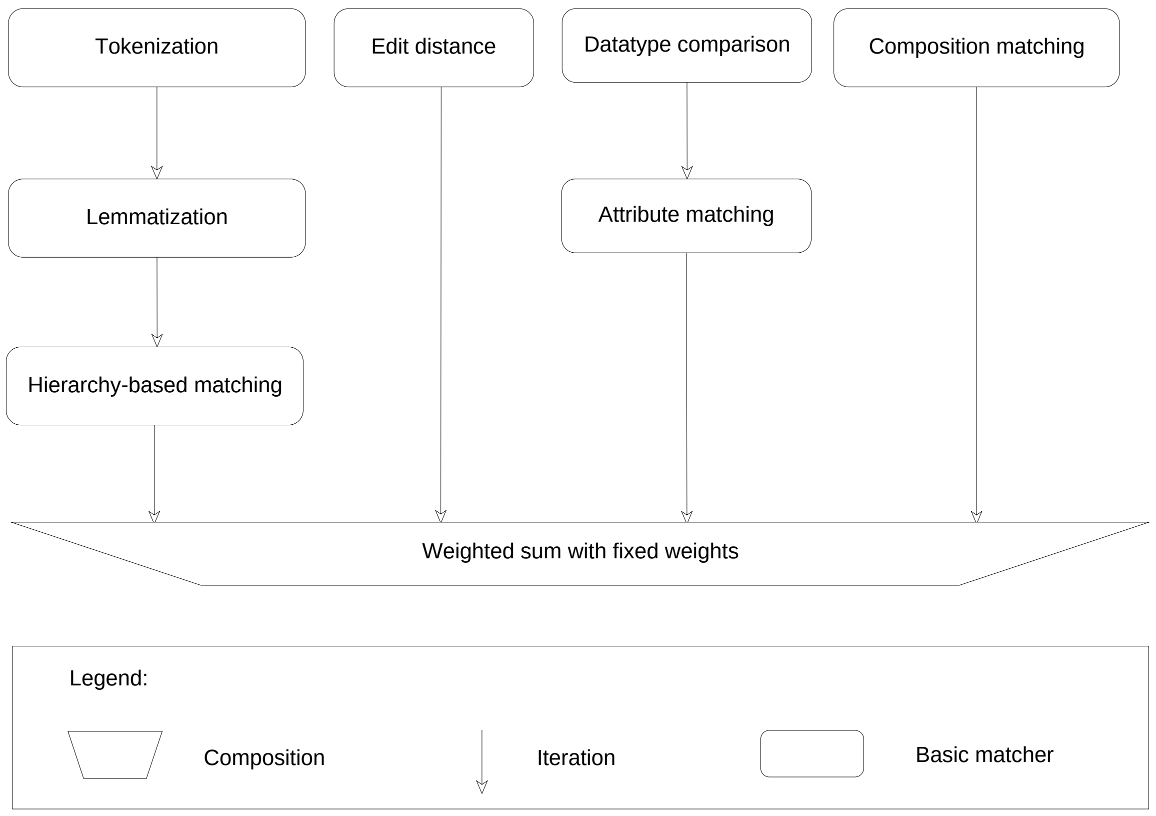

Figure 3. Matching system composition example

4.3. Dealing with non-local context

Variations of structure-level matching techniques may take into account different level of non-local context. Approaches in matching systems include

Computing similarity locally is a simple approach that does not utilize non-local context at all. Local computing concentrates on the internal structure of the studied entity. Information stored in neighboring entities may be used. Nevertheless, it is considered to be a local context, since the similarity computation is performed from the view of the (local) entity. The key distinction is that similarities computed in neighboring entities do not affect each other. For example, [6] compares concepts locally. It exploits concept name, names of relations and names of surrounding concepts.

Detecting leaf similarity is based on the idea that relations are used to implement composition. From the view of this approach, immediate surrounding of the entity may not be decisive. Instead, a concept is defined by its bottommost components, or leaves in graph terminology. Therefore, the set of both directly and indirectly belonging attributes is collected for every concept and used for similarity computation. This approach is used in the [8] tool.

Graph-based **similarity propagation** makes full advantage of non-local context. Ontology is perceived as a graph, relations form its edges. Similarities between concepts' neighbors contribute to the similarity of concept in different ontologies. This dependency is recursive and results in computationally expensive problem. It is usually solved by iterative fix-point algorithms; examples of tools using similarity propagation are [11] and [26].

5. A survey of state of the art systems

5.1. Matching system evaluation

Evaluation is performed to identify matching system strengths and weaknesses. In a long term, such continued evaluation helps designers to improve their systems and achieve better results.

The process of evaluation consists of running matching over a test suite and comparing achieved results to the desired ones, so-called *reference alignments*. Test suites may include benchmark tests, specially crafted for the purpose of evaluation, and real world cases.

5.1.1. Evaluation measures

For the use in ontology alignment evaluation, methods from the field of information retrieval were adopted [7], particularly *precision* and *recall*. Precision represents the correctness of found mappings; recall measures the completeness of the alignment. Precise definition of these measures follows:

$$P(A,R) = \frac{|R \cap A|}{|A|}$$

$$R(A,R) = \frac{|R \cap A|}{|R|}$$

Where A is the alignment created with the matching system and R is the reference alignment.

Fallout computes the ratio of false positives in the returned alignment:

$$F(A,R) = \frac{|A| - |R \cap A|}{|A|}$$

High precision tends to lead to lower recall and vice versa. Therefore, comparison based on only one of these measures is not predicative enough. On the other hand, comparison based on two values is less transparent. Hence, measures that aggregate precision and recall may be more useful for the comparison of matching systems.

The *F-1 measure* gives a harmonic mean of precision and recall:

$$F1(A,R) = \frac{2 \times P(A,R) \times R(A,R)}{P(A,R) + R(A,R)}$$

5.1.2. Ontology Alignment Evaluation Initiative

The ontology alignment community performs evaluation on a yearly basis since 2004 in the form of the Ontology Alignment Evaluation Initiative[2]. Along with results provided by their respective matching tools, the participants present papers with their observations and conclusions from the campaign. Both the evaluation results and these papers are valuable sources of insight.

The evaluation campaign consists of several tracks. In 2007 [10], these tracks were:

- The benchmark track is designed to reveal areas, where the given matching system is strong or weak.
- The excessive ontologies track is a task to match a pair of large real world anatomy ontologies.
- The directory track contains of several real world directory and thesaurus matching tasks.
- The workshop track contains free work over a set of conference organization ontologies

5.1.3. Method of this survey

To select representative matching systems for this survey, the performance in benchmark track was selected as the main criterion. Reasons for this selection are foremost:

- The tests of the benchmark track cover the possibilities of ontology heterogeneity well.
- The tests are not too domain-specific and do not prefer systems developed specifically for one area.
- Reference alignments are well defined.

[2] http://oaei.ontologymatching.org

- The continuity with previous years is preserved. Thus, the progression of the matching systems is observable.

The benchmark track is built around one reference ontology; a number of ontologies are given to align with. The ontologies are described in OWL-DL and serialized in the RDF/XML format. The track consists of three groups of tests [10]:

- Simple tests such as comparing the reference ontology with itself, with another irrelevant ontology (the wine ontology used in the OWL primer) or the same ontology in its restriction to OWL-Lite.
- Systematic tests that were obtained by discarding features from some reference ontology. It aims at evaluating how an algorithm behaves when a particular type of information is lacking.
- Real-life ontologies of bibliographic references

Results are presented in the form of mean values of precision and result for the groups of tests. In addition, precision/recall graphs are given that show the dependency between precision and recall. These graphs cut the results given by the participants under a threshold necessary for achieving n% recall and compute the corresponding precision.

Reproduction of this graph is shown in Figure 4. In addition to the 13 participants of the benchmark track in 2007, the graph shows results of best systems from previous year, namely Falcon from 2005 and RiMOM from 2006. `Refalign` represents the values for the reference alignment (that keep a precision of 1). `Edna` shows values for a simple edit distance matcher, given for comparison.

These results single out a group of matching systems, ASMOV, Lily, Falcon 0.7, OLA2, Prior+ and RiMOM which seem to perform these tests at the highest level of quality. Of these, ASMOV, Lily and RiMOM seem to have slightly better results than the three others.

These six systems will participate in the survey; their characteristics are investigated in the following section.

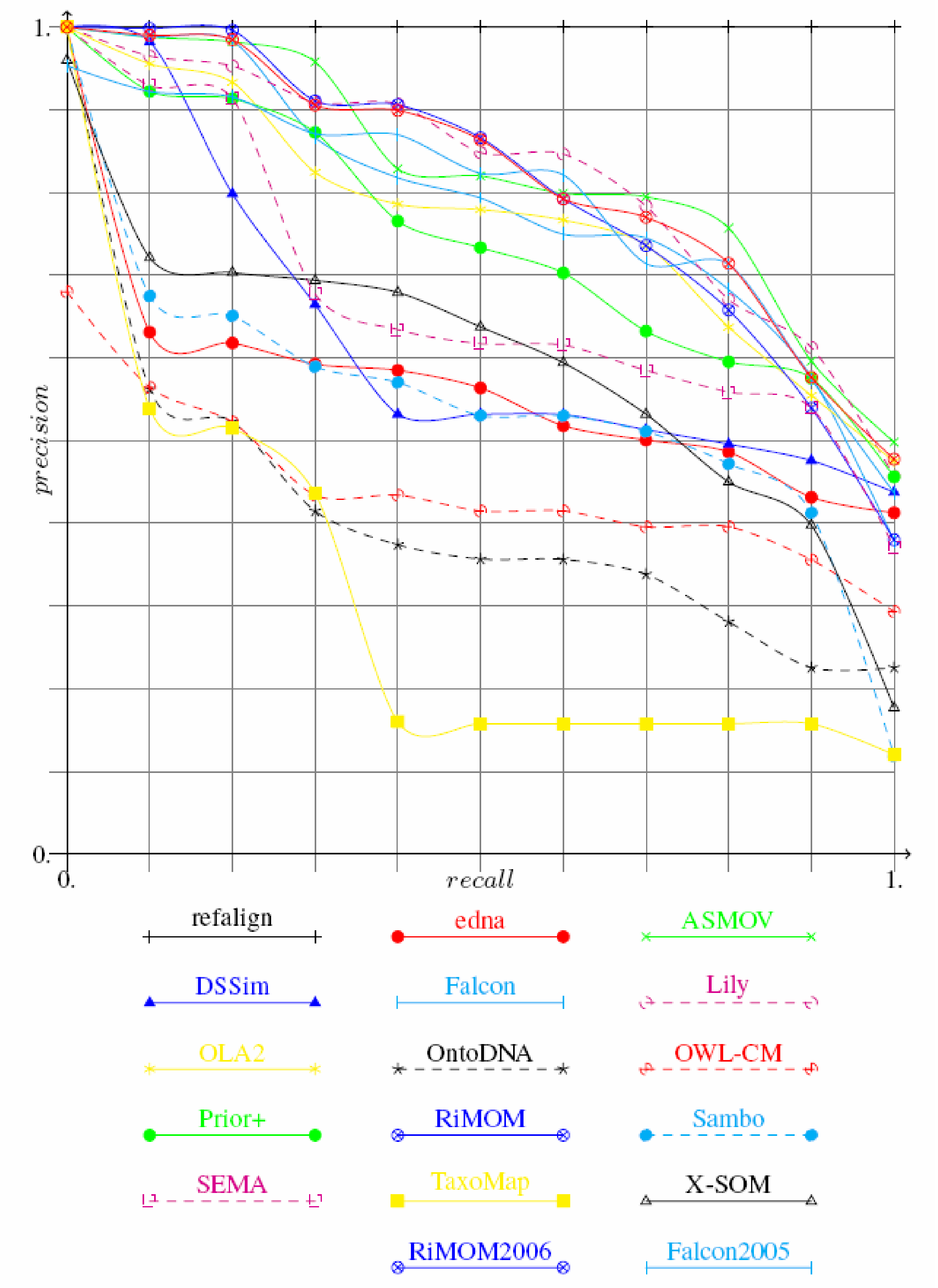

Figure 4. Precision/recall graphs. Reproduced from [10]

5.2. Characteristics of surveyed systems

5.2.1. ASMOV

ASMOV [17], [18] (Automated Semantic Mapping of Ontologies with Validation) is an automatic ontology matching tool which has been designed in order to facilitate the integration of heterogeneous systems, using their data source ontologies.[3]

The ASMOV matching process features pre-processing, a fix-point iterative similarity computation algorithm and rule-based mapping validation.

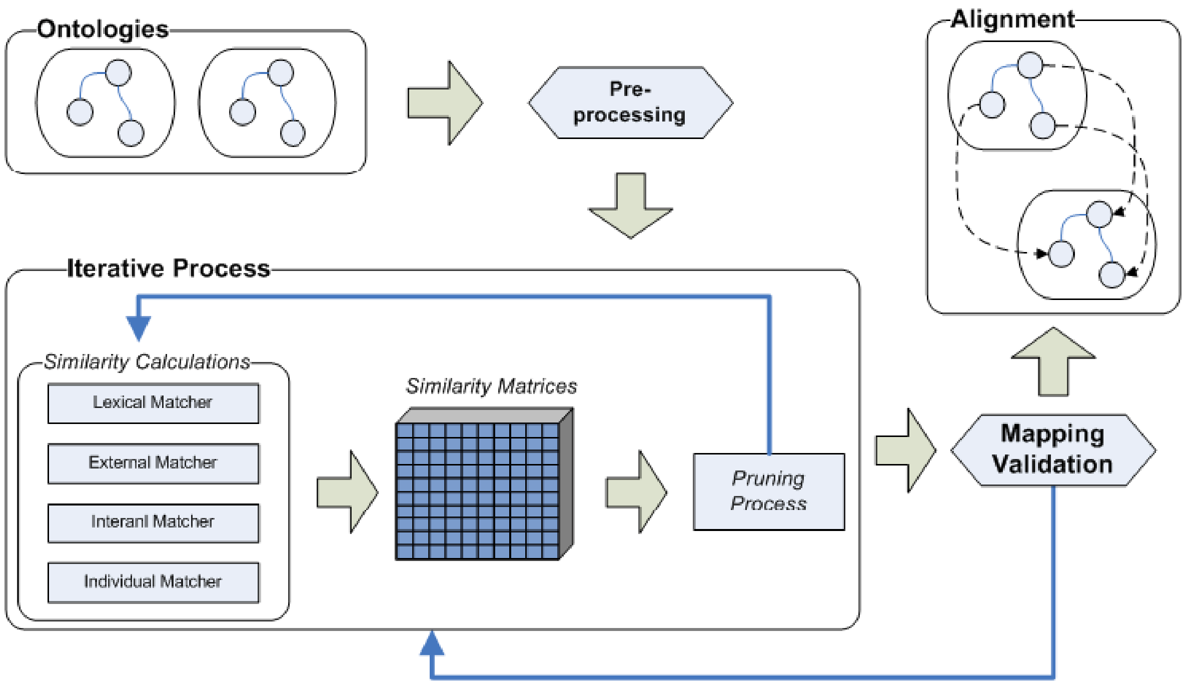

Figure 5. The matching process of ASMOV. Reproduced from [17].

In the pre-processing phase, classes and properties are tagged with the meaning of their ids and labels using a thesaurus. In addition, composition weights are adjusted in order to reflect the distribution of information between various aspects of the ontologies.

In the iterative process, four matchers are utilized. Each of these matchers exploits a class of ontology features: textual description (id, label, and comment), external structure (parents and children), internal structure (property

[3] http://support.infotechsoft.com/integration/ASMOV/index.html

restrictions for classes; types, domains, and ranges for properties), and individual similarity.

The partial similarity values are combined using a weighted sum and stored in a 2-dimensional matrix. On this matrix, pruning is executed after each iteration, which checks compliance to structure-based rules. Namely, mappings must not violate subsumption relations and one-to-many mappings are only approved in a special configuration of relationships.

Mapping validation combines information from ontologies and the mapping and checks for the integrity of classes and properties. If a violation is detected, iteration is restarted and creation of the violating mapping is prohibited.

5.2.2. Falcon-AO

Falcon-AO [16] is a key component of the Falcon open-source ontology processing environment.[4] Unlike many prototypes, Falcon-AO provides a graphical user interface.

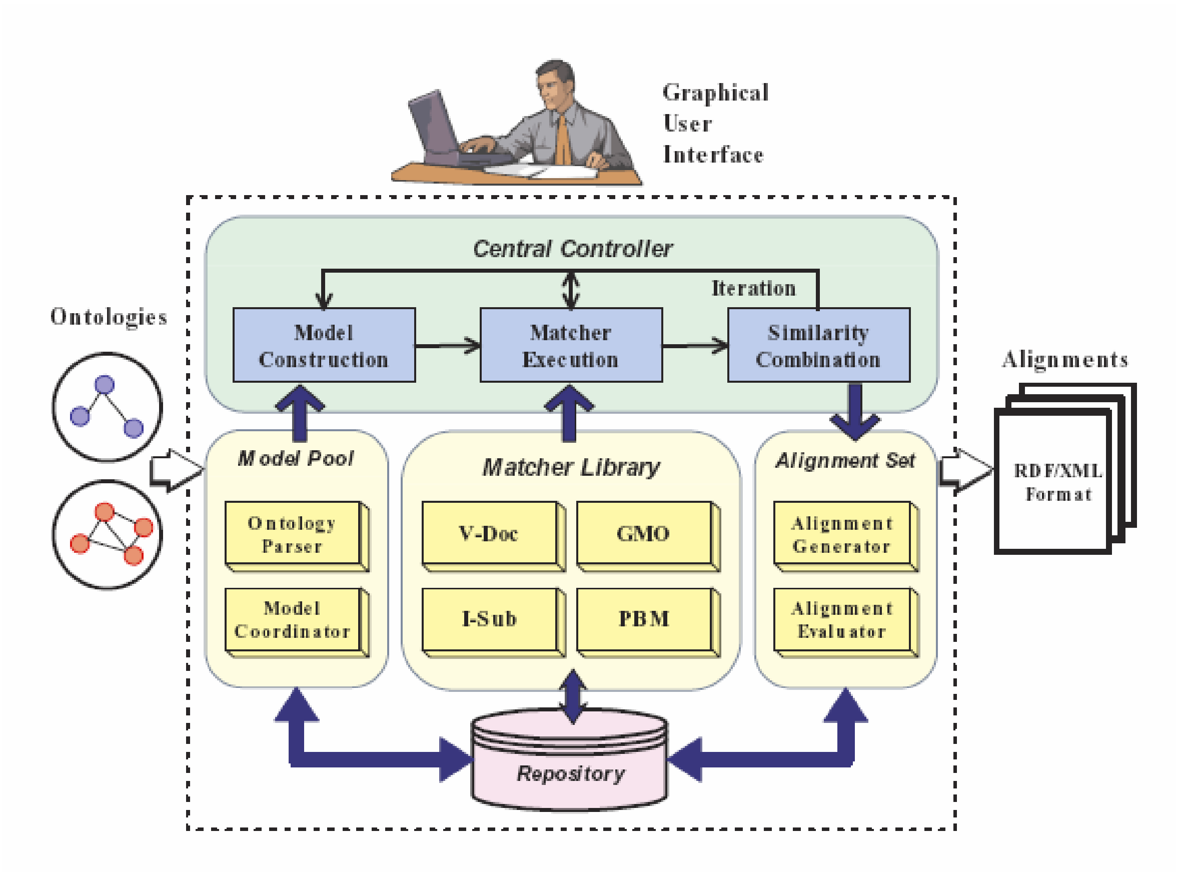

Figure 6. Falcon-AO architecture. Reproduced from [16].

[4] http://iws.seu.edu.cn/projects/matching/

This system consists of five components: the Repository to temporarily store the data during the matching process; the Model Pool to manipulate ontologies and to construct different models for different matchers; the Alignment Set to generate and to evaluate exported alignments; the Matcher Library to manage a set of elementary matchers; the Central Controller to configure matching strategies and to execute matching operations.

The matcher library contains four matchers. V-Doc exploits entity context, I-Sub is a string-based matcher, GMO computes relation similarity, and PBM performs fragmentation of large ontologies.

Model coordinator is a part of the model pool. It uses coordination rules to transform or reduce ontologies before they are sent to matchers for processing.

The alignment generator computes the linguistic and structural comparability of the whole matched ontologies. On basis of this comparability, it sets up thresholds for the individual matchers to restrict candidate mappings identified by them. For example, if structural comparability is higher then linguistic comparability, mappings generated by the GMO matcher are prioritized.

5.2.3. LILY

LILY [37] is an ontology matching system based on semantic subgraph matching. The alignment creation process of LILY consists of three steps.

In the first step, semantic subgraphs are constructed for ontology entities. Semantic subgraph is the context of the entity in the ontology it belongs to. In the approach of this tool, semantic subgraph captures the meaning of the entity itself.

Over the extracted semantic subgraphs, alignment similarity is computed. For entity identifiers, labels and descriptions this computation employs string-based methods; specialization, relation range and domain and related instance data are exploited for the entity context similarity. Resulting partial similarities are then combined using weighted sum with weights gained through experimentation.

Third, optional step is similarity propagation. Similarity computed in the previous phase is distributed to neighbors of the entity in the semantic subgraph. According to the authors, this operation does not improve precision. Therefore, it is only introduced when the previous steps did not produce enough mappings.

The resulting similarity matrix, computed by the described steps is then searched for values of similarity that exceed a threshold. This threshold is established depending on reached similarity values. For this purpose, the similarity matrix is treated as an image, and an image threshold is computed. To acquire one-to-one mappings, the stable marriage strategy is followed.

From the experience acquired in the evaluation campaign, the authors state that the matching algorithm used in LILY is not efficient enough. Therefore, it is not suitable for matching large ontologies.

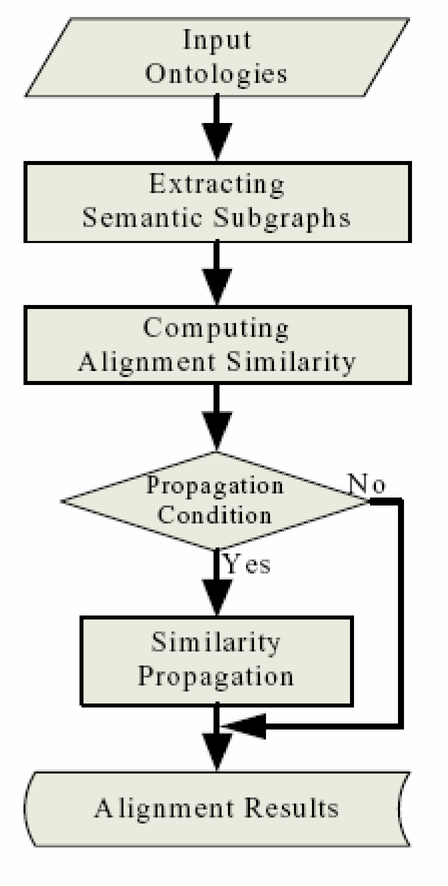

Figure 7. Matching process of Lily. Reproduced from [37].

5.2.4. OLA2

OLA2 [21] is a matching system following the similarity paradigm. OLA2 is a successor of the OLA (OWL Lite Alignment) system and applies the same basic principles as described in [11].

OLA2 addresses the OWL DL dialect. It introduces an integrated approach towards similarity, which is computed for all entity types in the same manner. For the purpose of matching, ontologies are translated to so-called match graphs, representing ontology entities and relationships holding between them. OLA2 defines similarity as a linear combination of surrounding ontology entities. This results into an equation system that is solved by an approximate fix-point algorithm.

This similarity measure employs the use of both string-based methods and most types of structure-based methods, as the integrated similarity definition exploits all kinds of ontology entities.

For future versions, the authors contemplate the utilization of WordNet and using a kind of adaptive weights, i.e. to not consider the kinds of entities missing in the ontologies for use in the similarity measure.

5.2.5. Prior+

The Prior+ [25] ontology matching system takes advantage of propagation theory, information retrieval technique and artificial intelligence model to solve the ontology mapping problem. It measures the profile similarity and structure similarity of different elements of ontologies in a vector space model and dynamically aggregates different similarities according to the harmoniousness of ontologies and deal with ontology constraints using interactive activation network.

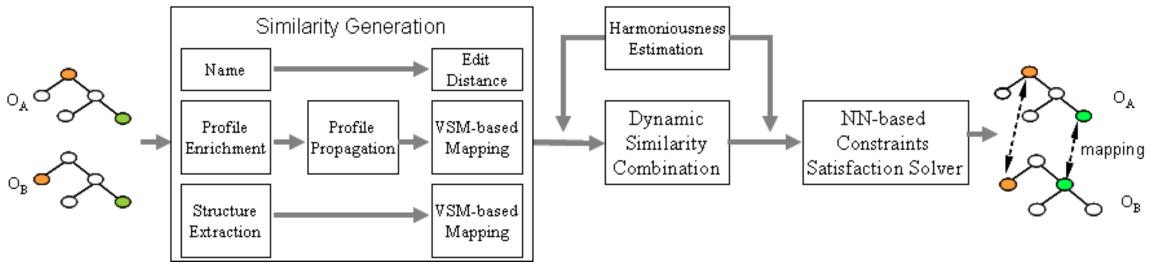

Figure 8. PRIOR+ architecture. Reproduced from [25].

The similarity generation model tries to detect the similarity of both linguistic and structural information of ontologies. Edit distance matching is used for matching concept names, profile enrichment and propagation collects lexical information stored in the concept and combines it with the profiles of related concepts through the utilization of specialization relations. Resulting profiles are considered elements in a vector space and compared using the cosine similarity. In the same manner, structural information of the concept is encoded and compared.

Harmoniousness is the level of similarity between ontologies. From the results of similarity generation, PRIOR+ computes name, profile and structure harmoniousness, in order to determine an appropriate matching strategy. Harmoniousness values are then used as weights for the combination of name, profile and structure similarity.

Then again, harmoniousness of the combined similarity is computed. If the ontologies are found similar enough, candidate mappings are sent to a NN-based

Constraints Satisfaction Solver as refined hypotheses. In the final phase, the solver approaches the problem as a constraint satisfaction problem, where the ontology structure presents a set of constraints. The interactive activation network method is then used to find the solution.

5.2.6. RiMOM

RiMOM [23] is a matching system that provides multiple matching strategies and a mechanism for selection of strategy (or strategies) appropriate for a given matching task.[5]

First, RiMOM estimates two ontology similarity factors, the structure similarity and the label similarity of the two ontologies. Following a set of rules, these similarities determine which strategies should be used, and what weights will they be assigned.

Possible matching strategies include edit distance matching, vector similarity (computing distance between the vector of concept's name, comment and instances), path-based matching or knowledge base (namely Wikipedia) utilization.

These strategies are performed independently and their results are composed with a combination measure that employs the sigmoid function and weights determined in the ontology similarity detection phase.

Next phase performs similarity propagation. Again, there are multiple strategies and ontology similarity is used to choose which ones to run. Available strategies exploit specialization, composition and general relations.

In the end, a set of heuristic rules is used to refine the resulting alignment.

5.3. Synthesis

Not surprisingly, all of the described tools use a number of elementary matching techniques to achieve the goal of high quality ontology matching.

All of the surveyed systems performed impressively, demonstrating that fully automated ontology matching is achievable.

There is a clear trend of using adaptive techniques to adjust the matching process to specific properties of ontologies given for alignment.

- The ASMOV and Prior+ tools use alignment similarity (or harmoniousness) to set weights for similarity combination;

[5] http://keg.cs.tsinghua.edu.cn/project/RiMOM/

- Falcon-AO computes ontology comparability to set thresholds to individual matchers;
- RiMOM system uses similarity to decide which strategies should be used and what weights are they assigned.

The limited utilization of lexicons in the leading systems is surprising. Possible conclusions to this fact are

(a) String-based matching techniques are efficient enough and the use of lexicons does not present an advantage.

(b) Available lexicons are not sufficient due to incomplete data.

The similarity paradigm is prevailing, but some system present a kind of reasoning to validate the alignments.

6. Conclusions

The goal of this thesis was to discover the most successful approaches to ontology matching. To this end, a classification and listing of matching techniques and was presented, along with discussion of possible approaches of their combination into ontology matching systems.

State of the art matching systems were chosen according to their performance in the 2007 OAEI campaign. Architecture of these systems and techniques they use for ontology matching were inspected. As the main distinctive feature common to these leading systems, the use of adaptive techniques for matching process adjustment was recognized.

Thus, the goal of this thesis was fulfilled.

6.1. Related work

The survey of [34] gives an extensive survey of schema matching methods.

The survey of [19] presents an overview of state of the art in ontology alignment techniques and systems.

The survey of [36] focuses on a classification of schema-based matching approaches and provides an overview of matching systems.

The very brief survey of [30] concentrates on the identification of methods specific for ontology matching

With the intensity of research in the field of ontology matching, the state of the art in ontology matching changes rapidly. For illustration, of 17 participants, only one is mentioned in [36], none are present in other here mentioned surveys.

This thesis is unique in the utilization of ontology matching system evaluation and presents the present state of the art in ontology matching.

References

[1] D. Aumüller, H. H. Do, S. Massmann, and E. Rahm. Schema and ontology matching with COMA++. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, Software Demonstration, 2005.

[2] B. Ben Yaghlane, N. Laamari. OWL-CM: OWL combining matcher based on belief functions theory. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, 2001.

[4] P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: A new approach and an application. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 130–145, 2003.

[5] P. Bouquet, L. Serafini, S. Zanobini, and S. Sceffer. Bootstrapping semantics on the web: meaning elicitation from schemas. In *Proc. 15th International World Wide Web Conference (WWW)*, 505–512, Edinburgh (UK), 2006.

[6] S. Castano, A. Ferrara, and S. Montanelli. Matching ontologies in open networked systems: Techniques and applications. *Journal on Data Semantics*, pages 25–63, 2006.

[7] H. H. Do, S. Melnik, and E. Rahm. Comparison of schema matching evaluations. In *Proceedings of the workshop on Web and Databases*, 2002

[8] H. H. Do and E. Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proceedings of the Very Large Data Bases Conference (VLDB)*, pages 610–621, 2001.

[9] J. Euzenat. An API for ontology alignment. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 698–712, 2004.

[10] J. Euzenat, A. Isaac, C. Meilicke, P. Shvaiko, H. Stuckenschmidt, O. Šváb, V. Svátek, W. R. van Hage, M. Yatskevich. First results of the Ontology Alignment Evaluation Initiative 2007**.** Paper presented at *The Second*

International Workshop on Ontology Matching, Busan, 11 November 2007.

[11] J. Euzenat and P.Valtchev. Similarity-based ontology alignment in OWL-lite. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 333–337, 2004.

[12] A. Gangemi, N. Guarino, C. Masolo, and A. Oltramari. Sweetening WordNet with DOLCE. *AI Magazine*, (24(3)), 13–24, 2003.

[13] F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-Match: an algorithm and an implementation of semantic matching. In *Proceedings of the European Semantic Web Symposium (ESWS)*, pages 61–75, 2004.

[14] M. Grüninger and J. Kopena. Semantic integration through invariants. In A. Doan, A. Halevy, and N. Noy, editors, *Workshop on Semantic Integration at ISWC-2003*, Sanibel Island, FL, 2003.

[15] I. Herman. State of the Semantic Web. Paper presented at *Semantic Days 2007*, Stavanger, Norway, 25 April 2007

[16] W. Hu, Y. Zhao, D. Li, G. Cheng, H. Wu, Y. Qu. Falcon-AO: results for OAEI 2007. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[17] Y. Jean-Mary, M. Kabuka. ASMOV results for OAEI 2007. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[18] Y. Jean-Mary, M. Kabuka. ASMOV: Ontology Alignment with Semantic Validation. *Joint SWDB-ODBIS Workshop on Semantics, Ontologies, Databases*, Vienna, Austria, 15–20, 2007.

[19] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review Journal (KER)*, (18(1)):1–31, 2003.

[20] J. Kang and J. F. Naughton. On schema matching with opaque column names and data values. In *Proceedings of the International Conference on Management of Data (SIGMOD),* 205–216, 2003.

[21] J. F. D. Kengue, J. Euzenat, P. Valtchev. OLA in the OAEI 2007 evaluation contest. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[22] L. Laera, V. Tamma, J. Euzenat, T. Bench-Capon and T. Payne. Reaching agreement over ontology alignments. In *5th International Semantic Web Conference (ISWC 2006)*, Athens, GA, 5 November 2006.

[23] Y. Li, Q. Zhong, J. Li, J. Tang. Result of ontology alignment with RiMOM at OAEI'07. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[24] J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching with Cupid. In *Proceedings of the Very Large Data Bases Conference (VLDB),* pages 49–58, 2001.

[25] M. Mao, Y. Peng. The PRIOR+: results for OAEI campaign 2007. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[26] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm. In *Proceedings of the International Conference on Data Engineering (ICDE)*, 117–128, 2002.

[27] A. G. Miller. WordNet: A lexical database for English. *Communications of the ACM*, (38(11)), 39–41, 1995.

[28] M. Nagy, M. Vargas-Vera, E. Motta. DSSim - managing uncertainty on the semantic web. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[29] I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS)*, 2–9, 2001.

[30] N.F. Noy. Semantic integration: a survey of ontology-based approaches. *ACM SIGMOD Record*, (33(4)), 65–70, 2004.

[31] N.F. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. Smi-2002-0926, University of Stanford, Stanford Medical Informatics, USA, 2002.

[32] N. Noy and M. Musen. Anchor-PROMPT: using non-local context for semantic matching. In *Proceedings of the workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI)*, 63–70, 2001.

[33] R. Ossewaarde. Simple library thesaurus alignment with SILAS. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[34] E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. *The International Journal on Very Large Data Bases (VLDB)*, (10(4)), 334–350, 2001.

[35] N. Shadbolt, T. Berners-Lee and W. Hall. The Semantic Web Revisited. *IEEE Intelligent Systems*, 21 (3), 96-101, 2006.

[36] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. *Journal on Data Semantics*, IV: 146–171, 2005.

[37] P. Wang, B. Xu. LILY: the results for the ontology alignment contest OAEI 2007. Paper presented at *The Second International Workshop on Ontology Matching*, Busan, 11 November 2007.

[38] The World Wide Web Consortium (2004). *OWL Web Ontology Language Guide: W3C Recommendation 10 February 2004* [online], available: http://www.w3.org/TR/owl-guide/ [accessed 2007-12-05].

[39] The World Wide Web Consortium (2004). *XML Schema Part 2: Datatypes Second Edition: W3C Recommendation 28 October 2004* [online], available: http://www.w3.org/TR/xmlschema-2/ [accessed 2007-12-05].

[40] N. Zhou. A study on automatic ontology mapping of categorical information. *ACM International Conference Proceeding Series*, Vol. 130, 1-4, 2003