DOTTORATO DI RICERCA IN
Computer Engineering and Science
SCUOLA DI DOTTORATO IN INFORMATION AND COMMUNICATION TECHNOLOGIES
XXI CICLO

Università degli Studi di MODENA e REGGIO EMILIA

TESI PER IL CONSEGUIMENTO DEL TITOLO DI DOTTORE DI RICERCA

Automatic Lexical Annotation: an effective technique for dynamic data integration

Candidato	Laura Po
Tutor	Prof. Sonia Bergamaschi
Co-Tutor	Prof. Paolo Bouquet
Il Direttore	Prof. Sonia Bergamaschi

To Franci and Lella

Keywords:

Data integration
Dynamic data integration
Word Sense Disambiguation
Probabilistic annotation
Probabilistic relationship

Abstract

In this thesis I affirm and prove how lexical annotation is helpful in data integration. Data integration is the problem of combining data residing at different sources, and providing the user with a unified view of these data. Lexical Annotation is a piece of information added in a document (book, online record, video, or other data), that refers to a semantic resource. Each annotation has the property to own one or more lexical descriptions. After the lexical annotation of a source, new lexical relationships between the elements of a schema or among elements of different schemata can be discovered. Several methods to accomplish the automatic annotation of data sources will be described and a set of evaluations on different scenarios will be shown. It will be demonstrated that lexical annotation can refine ontology matching techniques. To this purpose, a set of experiments applied on the results of a matcher will be displayed. Moreover, a probabilistic annotation approach and its role in a dynamic integration processes will be explained.

Sommario

In questa tesi viene dimostrato come l'annotazione lessicale sia un elemento cruciale in ambito di integrazione dati. Grazie all'annotazione lessicale, vengono scoperte nuove relazioni tra gli elementi di uno schema o tra elementi di schemi diversi. Saranno descritti diversi metodi per eseguire automaticamente l'annotazione delle sorgenti dati, inoltre sar*à* mostrata una serie di valutazioni dei metodi su vari scenari. L'annotazione lessicale pu*ò* perfezionare anche sistemi per la scoperta di matching tra ontologie. Alcuni esperimenti applicati ai risultati di un matcher saranno presentati. Inoltre, sar*à* introdotto l'approccio all'annotazione probabilistica e verr*à* spiegato come tale approccio costistuisca uno strumento chiave nei processi di integrazione dinamici.

Acknowledgments

The work on this thesis was partially supported by MUR FIRB Network Peer for Business project (http://www.dbgroup.unimo.it/nep4b) and by the IST FP6 STREP project 2006 STASIS (http://www.dbgroup.unimo.it/stasis).

I would like to thank the supervisor of my thesis, Professor Sonia Bergamaschi for her support in my research work, the co-tutor of this thesis, Prof. Paolo Bouquet, and one of my colleague at University of Modena and Reggio Emilia, co-writer of many of my papers, Serena Sorrentino, who has supported me during the last one year and a half of research.

As member of the DBGROUP, I wish to thank all the collaborators and in particular the colleagues that shared the life in the laboratory with me: Mirko Orsini, Antonio Sala, Francesco Guerra, and Matteo Di Gioia.

A part of the experimental evaluation of the methodologies was carried out during a 3 months stay at KMI[1]. Except from the English weather (that it is very hard to deal with for an Italian, especially during the summer), I was really stunned by the harmonious environment I found in KMI: the cooperation, the help, the professionalism that all the people has shown is something that I really appreciated and helped me during all my experience. In particular, I would like to thank Professor Enrico Motta and Marta Sabou who were my primary source of help during my stay; they opened me the door the first day and supported me all the time.

I would like to thank Andriy Nikolov of the KMI for introducing me to the Dempster-Shafer Theory, which has influenced this thesis and Jorge Garcia who helped me in improving my WSD algorithms and finding the best way to deal with the huge data sets, sources of my tests.

I also want to thank everyone that make my days in England so glad, putting a ray of sun even in the worst raining days, in particular, Carlo, Claudia, Annalisa. A special thank goes to Claudio B. whom I never met in KMI but taught me everything I need to know before arriving in Milton Keynes (this really helped me a lot). I wish to thank all the others that shared

[1] http://kmi.open.ac.uk/

with me some nice evenings or nice trips: Silvio, Jozef, Carlos, Vanessa, Davide, Ainhoa, Sofia, Neil and all the others.

From other points of view, the first persons I want to thank are my parents, who have never stopped supporting me, even during the bad times and sometimes remember me that I am “grande” (it is difficult to express this Italian word, because it covers a wide range of meanings: adult, big, great ...). A special thank goes to my brother, my sister and my grandmother, they all supported me a lot even when I was far from them.

And, finally, warm thanks go to all of my friends. In a random order I start thinking at Lella, Francy, my cousins Silvia and Cecilia, Sandro, Cicci, Marti, Ade, Truz, Bitta, Cristian, Orto, Santolin, Arghin, Maddy, Sara, Lori, Giuly, Bene, Silvia and many many others. Last but not least, I would like to thanks Ricky who gave me a huge support in the last days of my thesis, the best to back me up after whole days spent in writing. Thanks to all!

Contents

List of Figures

List of Tables

Introduction

Modern enterprises are often organized as "virtual networks", where they operate through inter-enterprise cooperative processes. To manage inter-enterprise and data exchange processes a key issue is to mediate among the heterogeneity of different information systems. Data Integration is a technological solution to build a shared and integrated knowledge base.

Data integration is the problem of combining data residing at different sources, and providing the user with a unified view of these data [67]. The problem of designing data integration systems is important in current real world applications, and is characterized by a number of issues that are interesting from a theoretical point of view. Ongoing research activity proves that many questions in data integration remain unanswered.

The core of data integration is solving the correspondences (find the right matches) among elements from different data sources. Usually, data sources are organized by many developers, according to different categorization (e.g. different collections of photos might be organised in different ways: classified according to years and then place, or, as an alternative, to people and date). Therefore, it is necessary, for the discovery of mappings, to understand the modelling logic behind structuring information (i.e the *structural relationships* among schema elements). Moreover, it is often difficult to understand the meaning behind the names denoting schemata elements (i.e. recognize how the data are "labelled"). Annotation becomes, thus, crucial to understand the meaning of schemata.

In general, annotation is the inclusion of extra information on a data source. Lexical Annotation is a particular kind of annotation that refers to a semantic resource (i.e. thesaurus, semantic network, semantic lexicon...). Each lexical annotation has the property to own a lexical description. Annotation of data sources associates meanings to each schema element.

Lexical annotation can be accomplished by making use of disambiguation methods in the case we want to decrease human intervention. Word Sense Disambiguation (WSD), in computational linguistics, is the process of identifying which sense of a word is used in any given sentence, when the word has a number of distinct senses. The WSD area has been typically focused on disambiguation of natural language texts. In order to apply WSD methods in the field of Data Integration we need to adapt these methods in order to cope with structured data sources.

As it is described in [61] the WSD task involves two steps: (1) the determination of all the different senses for every word under consideration; and (2) a mean to assign to each occurrence of a word its appropriate senses. The most recent works on WSD rely on predefined senses for step (1), including: a list of senses such as those found in dictionaries or thesauri.

In this thesis, all the developed WSD algorithms make use of a well-known and shared thesaurus (in our case WordNet). WordNet provides a reliable set of meanings and allows to share with others the result of the annotation process. Moreover, the fundamental peculiarity of a thesaurus is the presence of a wide network of relationships between words and meanings. The disadvantage in using a thesaurus is that it does not cover, with the same detail, different domains of knowledge. Some terms may not be present or, conversely, other terms may have many associated and related meanings. These considerations and the first tests executed led to the need of expanding the thesaurus with more specific terms (this can be easily done using the MOMIS component, called *WNEditor*, which allows adding new terms and linking them within WordNet [6]). On the other hand, when a term have many associated and related meanings, we need to overcome the usual disambiguation approach and relate the term to multiple meanings: i.e. to union of the meanings associated to it. For these reason, all WSD algorithms developed to accomplish the task of lexical annotation may associate more than one meaning to a term and is, thus, differs from the traditional approaches.

Traditional data integration systems interconnect a limited number of data sources, which are relatively stable in time. On the other hand, data applications broaden more and more and ask for flexibility and handling of uncertainty. For example, applications like Google Base or involving a large number of sources as in the deep web or tool dealing with biological data [68], require semantic mappings between the mediated schema and the data sources, to be approximate as they need to be automatically extracted. The future of data integration technologies will be in dynamic data integration systems where semantic mappings among schemata of different sources have to be discovered on the fly with a minimal human intervention.

Also in a traditional data integration context it is essential to make the lexical annotation process automatic when you are dealing with many sources, with sources with many elements and/or many ramifications. Greater size and more complicated structure bring to difficult understanding of the schemata even for a domain expert. Thus, I tried to pursue the goal of automatic annotation which lead to the need of "live with" uncertainty and probabilistic annotations. In order to handle uncertainty and to combine more WSD algorithms, I developed a probabilistic annotation approach. Starting from probabilistic annotations, a set of probabilistic lexical relationships can be derived.

During my PhD period I developed different methods to disambiguate structured and semi-structured data sources. I started my research work on WSD taking part to the development of a semi-automatic annotation method, called MELIS [13], then I focused my attention to WSD algorithms that can exploit the structural information of the data sources, and I contributed to the development of CWSD [17], a combined approach that makes use of the structural knowledge and domain knowledge of a source.

The evaluation of the results of these first methods, make me understand that approaches that combine several WSD algorithms are more efficient. The problem to be face become how the different methods have to be combined. To this purpose, I began to investigate on an approximate approach that can combine a flexible number of WSD algorithms [84]. Trying to realize the idea of a flexible combined approach I began studying probabilistic theory and finally, I found in the Dempster-Shafer theory the approach I was looking for to enhance the responses from different algorithms. Exploiting this theory I designed and implemented (in collaboration) PWSD (see section 2.3 for details). Instead of forcing to determine a unique best meaning for a term, PWSD associates to a term a set of related meanings each with its own reliability degree. My approach is supported by Renisk and Yarowsky that have introduced [87] the problem of disambiguation is not confined to search for the best meaning. They thought it is significant that a method reduces all possible meanings associate with a term, and that within this set cheques accurate probability to the correct meanings. Choosing more meanings for a term means that the number of discovered lexical relationships connecting a term to other meanings increase.

Proceeding with the study and test of combination methodologies, I participated to the development of a GUI that can support the user in finding the best combination of WSD algorithms for a specific scenario (see section 2.4 for details).

In the first chapter of this thesis, the background information about data integration and annotation are presented, and the MOMIS system, the data integration system where each of the methods have been integrated, is presented. All the methods are described in details in chapter 2. In chapter 3 I show the evaluation of the methods and the improvements achieved. Lexical annotation can be applied directly even on a matcher, not just in the context of data integration system: an evaluation of this scenario is shown in chapter 4.

Chapter 1

Data Integration Systems and Annotation

Data integration is a pervasive challenge faced in applications that need to query across multiple autonomous and heterogeneous data sources. Data integration is crucial in large enterprises that own a multitude of data sources, for progress in large-scale scientific projects, where data sets are being produced independently by multiple researchers, for better cooperation among government agencies, each with their own data sources, and in offering good search quality across the millions of structured data sources on the World-Wide Web.

Data integration Systems supply a transparent access to a collection of data stored in multiple, autonomous, and heterogeneous data sources [67]. Data integration is a very large fraction of the work in information technology (IT) departments in large enterprises. In the following, we report some of the main data integration activities [19]:

- Data warehousing: developing extraction, transformation and loading (ETL) programs to load a data warehouse.

- Lineage tracing: integrating ETL tools and data transformation applications to show the sequence of transformations used to populate a field in a database or data warehouse.

- Data integration: developing mappings from a mediated schema to data source schemas.

- Enterprise application integration: wrapping applications with message interfaces and developing mappings between the input and output messages of different applications.

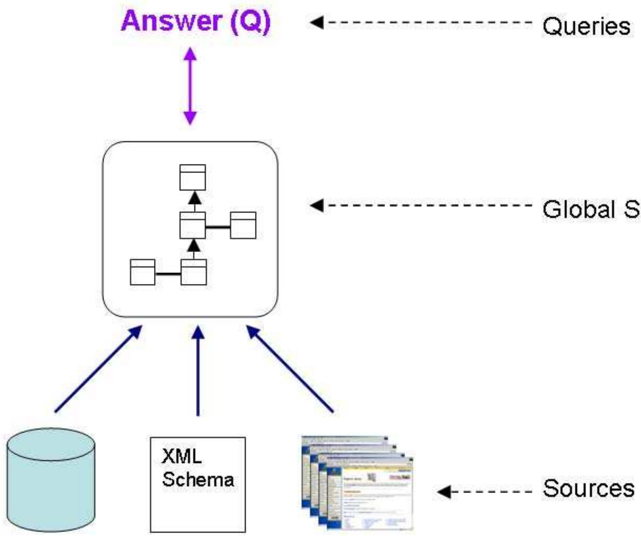


Figure 1.1: A data integration system

- Business-to-Business E-commerce: developing software to translate messages to and from business partners format.

- Information resource management: maintaining an inventory of information assets, such as databases, applications, forms, and message types.

- Document management: Attaching meta data to documents, placing them in a searchable store, and integrating them with workflow processes.

In this chapter, it will be analyzed in detail the data integration models, some data integration systems and, in particular, MOMIS (Mediator EnvirOment for Multiple Information Sources), the data integration system developed at the University of Modena and Reggio Emilia by the DBGROUP[1]. The lexical annotation methods (which are focus of this thesis) were developed in the last three years as MOMIS components. Moreover, it will be introduced the concept of *Lexical Annotation* and how this task is crucial in data integration systems.

[1] See http://www.dbgroup.unimore.it for references about the MOMIS project.

1.1 The modeling problem

Starting from a collection of data sources (different in format and structure) a data integration system focuses on providing a global schema that represent a unified view of the sources. The user can query the global schema to retrieve the information from the collection of sources (see figure 1.1).

In literature, we find different approaches to data integration:

- Mediator-based data integration
- Data exchange [44, 66]: where the global view is materialized and the query answering is done without accessing the sources
- P2P data integration [27, 55]: where the integration is done among several peers, each peer with local and external sources and the query answering is done over one peer

We focus our analysis on Mediator-based data integration, these systems aim at combining the data residing at different sources, and providing the user with a unified view of these data. Such a unified view is represented by the global schema, and provides a reconciled view of all data, which can be queried by the user. In this approach the data are stored in a collection of sources (the different sources from which the global schema is built); each query is expressed over the global schema (a.k.a. mediated schema, enterprise model, global view, ...). Moreover, there are specialized softwares, called wrappers, that access the sources and provide a view in a uniform data model of the data stored in the sources. The mediator elaborates the queries over the global schema and forwards them to local sources through the wrappers, in the end, it combines answers coming from wrappers to answer a query. To better understand the integration process, an example will be introduced in Section 1.1.3.

One of the main task in the design of a data integration system is to establish the mapping between the sources and the global schema, and such a mapping should be suitably taken into account in formalizing a data integration system. The main components of a data integration system are the global schema, the sources, and the mappings. Thus, we formalize a *data integration system I.*

Definition 1.1 ***Data Integration System***
A data integration system I is defined in terms of a triple $\langle G, S, M\rangle$*, where*

- *G is the global schema, expressed in a language* L_G *over an alphabet* A_G*. The alphabet comprises a symbol for each element of G (i.e., relation if G is relational, class if G is object-oriented, etc.).*

- *S is the source schema, expressed in a language L_S over an alphabet A_S. The alphabet A_S includes a symbol for each element of the sources.*

- *M is the mapping between G and S, constituted by a set of assertions of the forms*

$$q_S \rightsquigarrow q_G,$$
$$q_G \rightsquigarrow q_S$$

where q_S and q_G are two queries of the same arity, respectively over the source schema S, and over the global schema G. Queries q_S are expressed in a query language $L_{M,S}$ over the alphabet A_S, and queries q_G are expressed in a query language $L_{M,G}$ over the alphabet A_G. Intuitively, an assertion $q_S \rightsquigarrow q_G$, specifies that the concept represented by the query q_S over the sources corresponds to the concept in the global schema represented by the query q_G (similarly for an assertion of type $q_G \rightsquigarrow q_S$).

One of the most important aspects in the design of a data integration system is the specification of the mappings between the data at the sources and those in the global schema. They are exactly these mappings that will determine how the queries posed to the system are answered. Two basic approaches for specifying the mapping in a data integration system have been proposed in the literature, called global-as-view (GAV) [9], the approach that maps the concepts in the global schema to views over the sources, and local-as-view (LAV) [26], the approach that maps the sources into views over the global schema. Later on the advantages of LAV and GAV approaches were combined in a mediation language called GLAV [47].

1.1.1 LAV (Local As View) approach

In a data integration system $I = \langle G, S, M \rangle$, based on the LAV approach, the mapping M associates to each element s of the source schema S a query q_G over G.

In other words, the query language $L_{M,S}$ allows only expressions constituted by one symbol of the alphabet A_S. Therefore, a LAV mapping is a set of assertions, one for each element s of S, of the form

$$s \rightsquigarrow q_G,$$

From the modeling point of view, the LAV approach is based on the idea that the content of each source s should be characterized in terms of a view q_G

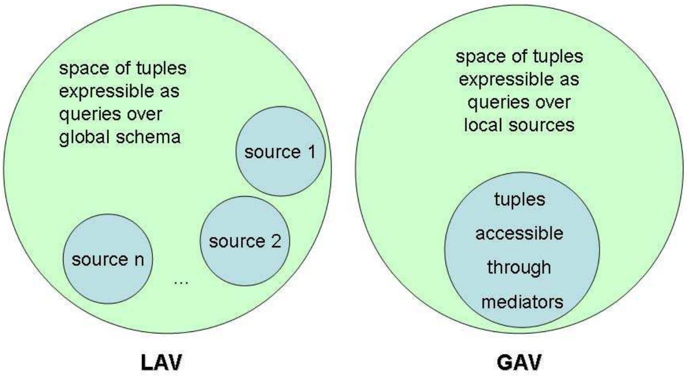

space of tuples expressible as queries over global schema
source 1
source 2
source n
...
LAV
space of tuples expressible as queries over local sources
tuples accessible through mediators
GAV

Figure 1.2: Illustration of tuple space of the GAV and LAV mappings

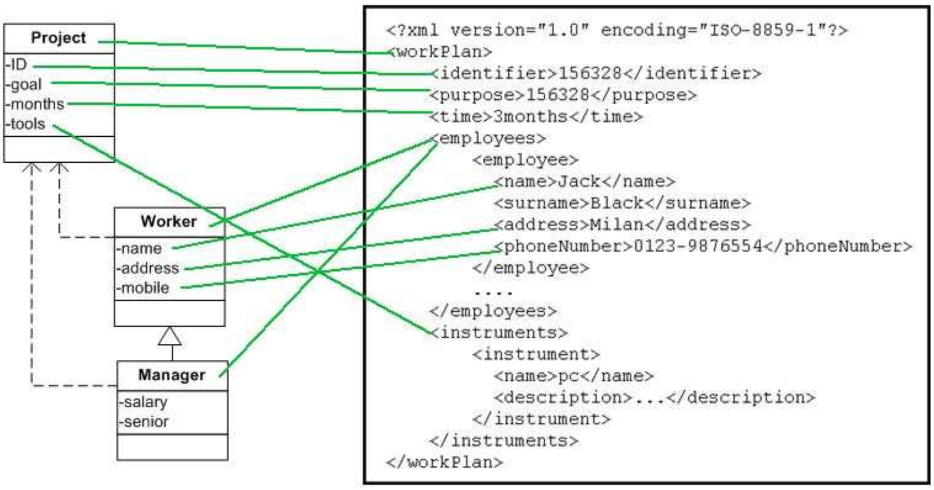
over the global schema (see figure 1.2). A notable case of this type is when the data integration system is based on an enterprise model, or an ontology [58]. This idea is effective whenever the data integration system is based on a global schema that is stable and well-established in the organization. Note that the LAV approach favors the extensibility of the system: adding a new source simply means enriching the mapping with a new assertion, without other changes. On the other hand, a pre-existing stable ontology is not very often available.

1.1.2 GAV (Global As View) approach

In the GAV approach, the mapping M associates to each element g in G a query q_S over S. In other words, the query language $L_{M,G}$ allows only expressions constituted by one symbol of the alphabet A_G. Therefore, a GAV mapping is a set of assertions, one for each element g of G, of the form

$$g \leadsto q_S,$$

From the modeling point of view, the GAV approach is based on the idea that the content of each element g of the global schema should be characterized in terms of a view q_S over the sources (see figure 1.2). In some sense, the mapping explicitly tells the system how to retrieve the data when one wants to evaluate the various elements of the global schema. This idea is effective whenever the data integration system is based on a set of sources that is stable. Note that, in principle, the GAV approach favors the system in carrying out query processing, because it tells the system how to use the sources to retrieve data. However, extending the system with a new source is



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<workPlan>
    <identifier>156328</identifier>
    <purpose>156328</purpose>
    <time>3months</time>
    <employees>
        <employee>
          <name>Jack</name>
          <surname>Black</surname>
          <address>Milan</address>
          <phoneNumber>0123-9876554</phoneNumber>
        </employee>
        ....
    </employees>
    <instruments>
        <instrument>
          <name>pc</name>
          <description>...</description>
        </instrument>
    </instruments>
</workPlan>
```

Figure 1.3: Example of two schema representations and their correspondences

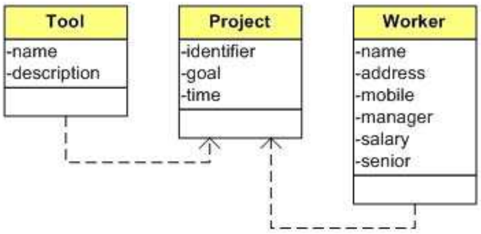


Figure 1.4: Example of integrated schema

now a problem: the new source may indeed have an impact on the definition of various elements of the global schema, whose associated views need to be redefined.

1.1.3 Example of integration

Let us suppose that we need to create a unique global view of two different sources: one is a relational data base and the second is a XML file, both describe how project management. The two representation of the schemas and their correspondences are shown in figure 1.3. The arrows, in the source on the left, represent referential constraints: a manager has references to a project, as well as, a worker. In SourceB, `workPlan` is the correspondent of `project` on SourceA. There is no distinction between workers and managers, but they are grouped and represented as `employees`. Differently from

SourceA, the `instruments` in a `workPlan` are represented by structured information such as `name` and `description`. Furthermore, SourceB includes additional information than SourceA (employees have surname).

Given such source schemas, the preliminary step towards the computation of an integrated schema is to provide a set of correspondences between the schemas. These correspondences are bi-directional and signify potentially equivalent elements in the two schemas. Figure 1.3 depicts the correspondences as the green lines across the two schemas. Not all attributes need to be an end-point of a correspondence. Also, in general, an attribute can be the end-point of multiple correspondences.

Starting from the correspondences an integration system can derive an integrated schema such as the one proposed in figure 1.4. According to the rules of the data integration system the matches are established and the integrated schema is generated. Let us suppose, that two concepts A and B match (i.e., they can be merged) if and only if there is a correspondence between an attribute specific to A and an attribute specific to B. For example, `project` matches with `workPlan` and hence they may be merged. The integrated schema will be the focus of querying in order to extract data from both the sources.

1.2 Ontologies and the problem of ontology mapping

Definition 1.2 ***Ontology***
An ontology is an explicit specification of a conceptualization.

This definition was proposed initial by Gruber in [53]. An ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. An ontology provides a shared vocabulary, which can be used to model a domain that is, the type of objects and/or concepts that exist, and their properties and relations. Ontologies are used in artificial intelligence, the Semantic Web, software engineering, biomedical informatics, library science, and information architecture as a form of knowledge representation about the world or some part of it. Ontologies are used to reason about the properties of that domain, and may be used to define the domain.

Definition 1.3 ***Ontology mapping***
The ontology mapping procedure for two separate and autonomous ontologies, O_1 and O_2, consists of the following steps:

- *Step 1: Finding corresponding entities in ontologies O_1 and O_2;*

- *Step 2: Representing the found correspondences and using it to achieve some goal.*

For Step 1, the main ontology entities that can be considered, when finding correspondences between ontologies O_1 and O_2, are: classes (concepts), individuals (instances), and properties (relations). For Step 2, for using the found correspondences, they need to be represented in a suitable format.

Notice that the goals (i.e. usages) of ontology mapping determine, what candidates to consider, when we are finding the correspondences. The goals also determine how to represent the correspondences. This definition of ontology mapping is quite precise, but also overarching, such that it encompasses the different goals of the problem.

There can be identified two quite distinct goals for ontology mapping, based on real world use cases. One possible goal of mapping is *ontology development*; that is when an ontology is being designed or engineered by an organization. The other possible goal of mapping is *interoperability*; that is when there are various parties, which are using different ontologies, and the users need a mechanism to be able to query the information.

Within the second goal (that can be considered as an ontology reuse process), some authors [31] differentiate three mapping categories: ontology merging, integration, and alignment. *Ontology merging* is the process of generating a single, coherent ontology from two or more existing and different ontologies related to the same subject. *Ontology alignment* is the task of creating links between two original ontologies. *Ontology integration* is the process of generating a single ontology in one subject from two or more existing and different ontologies in different subjects.

Definition 1.4 ***Ontology matching***
Ontology Alignment, or ontology matching, is the process of determining correspondences between concepts. A set of correspondences is also called an alignment.

The matching operation determines the alignment A_0 for a pair of ontologies O_1 and O_2, each of which consisting of a set of discrete entities, such as classes, properties or individuals.

There are some other parameters that can extend the definition of the matching process, namely: (i) the use of an input alignment A, which is to be completed by the process; (ii) the matching parameters, for instance, weights, thresholds; and (iii) external resources used by the matching process, for instance, common knowledge and domain specific thesauri.

Definition 1.5 ***Ontology alignment***
Given two ontologies, a correspondence is a 5-uple: $< id, e_1, e_2, n, r >$*, where: id is a unique identifier of the given correspondence;* e_1 *and* e_2 *are entities (e.g., tables, XML elements, properties, classes) of the first and the second ontology, respectively; n is a confidence measure (typically in the* $[0; 1]$ *range) holding for the correspondence between* e_1 *and* e_2*; r is a relation (e.g., equivalence (=), more general (*$\sqsupseteq$*), disjointness (*$\perp$*), overlapping (*$\sqcap$*)) holding between* e_1 *and* e_2*.*

The correspondence $< id, e_1, e_2, n, r >$ *asserts that the relation r holds between the ontology entities* e_1 *and* e_2 *with confidence n. The higher the confidence, the higher the likelihood that the relation holds.*

1.3 Metadata and Lexical Annotation

An *Annotation* is a piece of information added in a book, document, online record, video, or other data (an intuitive example of annotation are the notes about the quality of a document written on the sheets of a draft document by a reader).

In the field of Semantic Web, annotation becomes a set of instantiations related to an ontology and referring to an HTML document [57]. This annotation is called *Metadata Annotation*.

Metadata is structured data which describes the characteristics of a resource. Literally, metadata is “data about data”. It shares many similar characteristics to the cataloguing that takes place in libraries, museums and archives. The term “meta” derives from the Greek word denoting a nature of a higher order or more fundamental kind. A metadata record consists of a number of predefined elements representing specific attributes of a resource, and each element can have one or more values. A definition of metadata supplied by Tim Berners-Lee, with reference to the Web, is a “Machine-understandable information about Web resources or other things”[2].

Below is an example of a simple metadata record:

Each metadata schema will usually exhibit a limited number of elements, the name and the meaning of each element.

The *Semantic Web* supports its users to find accurate information, to combine easily related pieces of information into an overarching picture and to compose new applications without programming knowledge. To achieve these objectives not only human readers have to understand what is offered on a web page, software agents also must be able to interpret existing information. This is only possible when the relevant information is represented in

[2] Tim Berners-Lee, W3C (1997)

Element name	Value
Title	Web catalogue
Creator	Dagnija McAuliffe
Publisher	University of Queensland Library
Identifier	http://www.library.uq.edu.au/iad/mainmenu.html
Format	Text/html
Relation	Library Web site

a declarative and semantically precise way and when it is thus understandable for the computer. This need creates the necessity to generate semantically accurate, ontology-based metadata. Handschuh and others distinguish an *Ontology-based Metadata Annotation*, an annotation in which metadata refer to an ontology, as (i) instantiations of ontology classes (class instance), (ii) instantiated properties from one class instance to a datatype instance (attribute instance), and (iii) instantiated properties from one class instance to another class instance (relationship instance).

On the Web we encounter dynamic pages, so it is necessary to introduce the concept of *Deep Annotation*, the annotation of dynamic Web documents: i.e. a semantic mapping to the underlying database if the database owner cooperates in the Semantic Web, that allows for direct access to the database [57].

Semantic annotation has been developed within language technology in recent years in connection with more integrated tasks like information extraction. Natural language applications, such as information extraction and machine translation, require a certain level of semantic analysis. An important part of this process is semantic tagging [24]: the annotation of each content word with a semantic category. The activity of semantic tagging refers to the activity of annotating text documents (written in plain ASCII or HTML format) with a tags set defined on the ontology. Semantic categories are assigned on the basis of a semantic lexicon, like WordNet for English [75] or EuroWordNet, which links words between many European languages through a common inter-lingua of concepts, in a larger multilingual context.

There are different semantic resources that are available to be used in semantic tagging. Special emphasis is given also to the important subtask of sense disambiguation, which is needed if a word or term corresponds to more than one possible semantic class (polysemy of words).

Starting from metadata annotation and sense tagging we have derived and defined the concept of lexical annotation.

Definition 1.6 ***Lexical Annotation***
Lexical Annotation is a particular kind of Metadata Annotation that refers to a semantic resource. Each annotation has the property to own one or more lexical descriptions.

Lexical Annotation differs from the *Ontology-based Metadata Annotation* where we annotate w.r.t. an ontology and it is not mandatory that an ontology class has a lexical description.

1.3.1 Semantic Resources

Semantic knowledge is captured in resources like dictionaries, thesauri, and semantic networks, all of which express, either implicitly or explicitly, a general ontology of the world or of more specific domains, such as medicine. They can be roughly distinguished into the following three groups:

- Thesauri: Semantic resources that group together similar words or terms according to a standard set of relations, including broader term, narrower term, sibling, etc.

- Semantic Lexicons: Semantic resources that group together words (or more complex lexical items) according to lexical semantic relations like synonymy, hyponymy, meronymy, and antonymy

- Semantic Networks: Semantic resources that group together objects denoted by natural language expressions (terms) according to a set of relations that originate in the nature of the domain of application.

Thesauri Roget is a thesaurus of English words and phrases. It groups words in synonym categories or concepts. Besides synonyms also antonyms are covered.

MeSH (Medical Subject Headings) is a thesaurus for indexing articles and books in the medical domain, which may then be used for searching MeSH-indexed databases[3]. MeSH provides for each term a number of term variants that refer to the same concept. It currently includes a vocabulary of over 250,000 terms.

Semantic Lexicons As Semantic Lexicons we find WordNet [75]. WordNet has primarily been designed as a computational account of the human capacity of linguistic categorization. It therefore covers a rather extensive

[3] http://www.nlm.nih.gov/mesh/meshhome.html

set of semantic classes (called synsets), over 110.000 in the current version (WordNet 3.0). Synsets are collections of synonyms, grouping together lexical items according to meaning similarity. For instance, similar lexical items can be grouped together in one synset. At the same time, however, one item can also refers to different synsets.

So, in fact synsets are actually not made up of lexical items, but rather of lexical meanings (i.e. senses). Observe, that synsets define lexical meaning in an implicit way, contrary to using explicit definitions.

Synsets range from the very specific to the very general. Very specific synsets typically cover only a small number of lexical items, while very general ones tend to cover many.

EuroWordNet [97] is a multilingual semantic lexicon for several European languages and is structured in a way similar to WordNet. Each specific language (Euro)WordNet is linked to all others through the Inter-Lingual-Index (ILI), which is based on WordNet 1.5. Via this index the languages are interconnected, so that it is possible to move from a word in one language to similar words in any of the other languages in the EuroWordNet semantic lexicon.

Semantic Networks UMLS (Unified Medical Language System)[4] is one of the most extensive semantic resources available. It is based in part on the MeSH thesaurus and is specific to the medical domain. UMLS integrates linguistic, terminological and semantic information in three corresponding parts: the Specialist Lexicon, the Metathesaurus and the Semantic Network. The Metathesaurus is a multilingual thesaurus that groups term variants together that correspond to the same concept, for instance terms variants in several languages.

The Semantic Network organises all concepts in the Metathesaurus into 134 semantic types and 54 relations between semantic types. Relations between semantic types are represented in the form of triplets, with two semantic types linked by one or more relations.

CYC[5] is a semantic network of over 1,000,000 manually defined rules that cover a large part of common sense knowledge about the world. Each concept in this semantic network is defined as a constant, which can represent a collection (e.g. the set of all people), an individual object (e.g. a particular person), a word (e.g. the English word person), a quantifier (e.g. there exist), or a relation (e.g. a predicate, function, slot, attribute).

[4] http://www.nlm.nih.gov/research/umls/

[5] http://www.cyc.com

1.3.2 Languages and Tools for Annotation

A lot of researches have dealt with the annotation problem and have proposed different languages and tools to facilitate the user to annotate contents or to extract information from contents and semi-automatically annotate.

In the research community, there is a substantial agreement about the "general" meaning of annotation, as the operation of inserting "extra information associated with a particular point in a document or other piece of information"[6]. On the other hand, different points of view are expressed regarding the features and the information provided with such annotations and the languages for expressing them.

In the Semantic Web, these differences are stressed because of the massive use of these statements. In this community, annotating generally means the operation of associating metadata with web resources.

Different formalism have been proposed to represent information provided with annotations and the languages for expressing them. For example in [71], annotations are classified on the basis of several dimensions: formal vs. informal, explicit vs. tacit, permanent vs. transient , published vs. private, and so on. The authors in [5] point out a problem regarding the lack of a common understanding of the annotation process: for this reason they refer to "the semantics of semantic annotation" as the provision of a consistent interpretation of assumptions that we make and the context within which such annotation should be interpreted.

In the Semantic Web, the annotation task has been transformed into the problem of associating metadata with web resources. In particular, [5] qualifies such metadata as semantic metadata since they provide some indications about the contents of a resource. The proposed COHSE system includes three kinds of annotation: textual annotation, i.e. the process primarly targeted at human readers whereby notes or commentaries are added to the resources; link annotation, i.e. the content of the annotation is given by a link destination; semantic annotation, where the content of the annotation consists of some semantic information accessible to machine-processing. This last idea of semantic annotation has been pursued in Ontobroker [34], SHOE [59] and in the KIM Platform [85]. In [94] some further features of semantic annotation are introduced: in particular they are semantically interlinked, and they need to be congruent with ontology definition. This fact generates new issues, related to the manual execution of the annotation process and the management of changes in the ontologies that compels to have an annotation framework able to handle ontology creation, mapping and versioning. In [5] several types of annotation uses are specified: as dec-

[6] http://en.wikipedia.org/wiki/Annotation.

oration, i.e. for providing commentaries, as link, as instance identification, as instance reference, for specifying aboutness and pertinence.

In [57], a parallelism between annotation and mapping is highlighted. By means of the "deep annotation", the authors define an annotation process that utilizes information properties, information structures and information context in order to derive mappings between information structures. In a similar way, our framework builds annotations for mapping local source contents into a domain ontology and a lexical reference.

Several tools supporting users in annotation have been developed[7]. Here below we describe some of the principal tools.

Annotea [65] is a tool that shares the idea of creating a kind of user comment about Web pages. The term "annotation" in this framework is understood as a remark to an existing document. Annotea allows reliance on an RDF schema as a kind of template that is filled in by the annotator. The annotation metadata can be stored locally or in one or more annotation servers and presented to the user by a client capable of understanding this metadata and capable of interacting with an annotation server with the HTTP service protocol. Annotea enhances collaboration via shared metadata based Web annotations, bookmarks, and their combinations. When a user asks for a document, he or she can also load the annotations attached to it, from an annotation server, and see what his peer group thinks. Similarly shared bookmarks can be attached to Web documents to help organize them under different topics, to easily find them later, to help find related material and to collaboratively filter bookmarked material. Annotea is open source; it uses and helps to advance W3C standards[8] when possible and it is part of the Semantic Web efforts.

SHOE [58] is a small extension to HTML which allows web page authors to annotate their web documents with machine-readable knowledge. HTML was never meant for computer consumption; its function is for displaying data for humans to read. The "knowledge" on a web page is in a human-readable language (usually English). Unfortunately, even with state-of-the-art natural language technology, getting a computer able to read and understand web documents is very difficult. This makes it very difficult to create an intelligent agent that can wander the web on its own, reading and comprehending web pages as it goes. SHOE eliminates this problem by making it possible for web pages to include knowledge that intelligent agents can actually read. The SHOE Knowledge Annotator is a Java program that allows users to mark-up web pages with SHOE knowledge without having to worry about

[7] http://annotation.semanticweb.org/tools/

[8] http://www.w3.org/2001/Annotea/

the HTML-like codes.

WebKB [72, 43] uses conceptual graphs for representing the semantic content of Web documents. It embeds conceptual graph statements into HTML pages. Essentially they offer a Web-based template like interface. The first version, WebKB-1 [72], was a private annotation tool, it permits Web users to store, organize and retrieve knowledge or document elements within Web-accessible files. With the second version, WebKB-2 [43], a shared annotation tool has been proposed. Simple interfaces exploiting the content of the knowledge base are provided to permit easy accesses and updates to the knowledge base. Updates are regulated by permissions and protocols to ensure harmonious collaboration between the users.

1.4 Improving the data integration process through lexical annotation

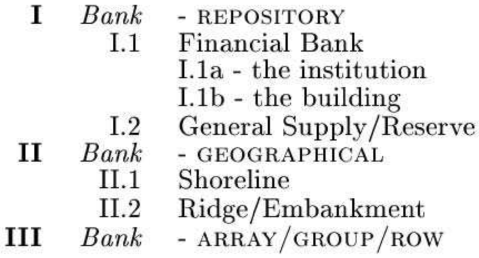
The lexical annotation of the sources (textual, structured or semi-structured) aims to solve the semantic differences among different data representations. In this way, starting from the semantic associated to the schema elements of distributed sources, it is possible to discover conceptual mappings among the elements themselves. Lexical annotation seems to be a critical task to develop smart methods for ontology learning and matching, therefore, it should not come as a surprise that a large number of tools includes some lexical resource (mainly WordNet[9]) as a component, and uses it in some intermediate step to annotate schema elements and ontology classes/properties with lexical knowledge.

In these context, a batch of methods have been developed to support the source annotation process, i.e. the operation of associating an element of a lexical reference database (WordNet in our implementation, but the method is independent from this choice) to all source elements.

To perform lexical annotation of the sources, we need to explore the area of Word Sense Disambiguation (WSD). As it is described in [61] the WSD task involves two steps: (1) the determination of all the different senses for every word under consideration; and (2) a mean to assign to each occurrence of a word its appropriate senses. The most recent works on WSD rely on predefined senses for step (1), including: a list of senses such as those found in dictionaries or thesauri.

The use of a well-known and shared thesaurus (in our case WordNet) provides a reliable set of meanings and allows to share with others the result

[9] See `http://wordnet.princeton.edu` for more information on WordNet.



I	*Bank*	- REPOSITORY
	I.1	Financial Bank
		I.1a - the institution
		I.1b - the building
	I.2	General Supply/Reserve
II	*Bank*	- GEOGRAPHICAL
	II.1	Shoreline
	II.2	Ridge/Embankment
III	*Bank*	- ARRAY/GROUP/ROW

Figure 1.5: The grouping of the WordNet synsets of “Bank”.

of the disambiguation process. Moreover, the fundamental peculiarity of a thesaurus is the presence of a wide network of relationships between words and meanings.

The disadvantage in using a thesaurus is that it does not cover with the same detail different domains of knowledge. Some terms may not be present or, conversely, other terms may have many associated and related meanings. These considerations and the first tests made led to the need of expanding the thesaurus with more specific terms (this can be easily done using the MOMIS component, called *WNEditor*, which allows adding new terms and linking them within WordNet [6]). On the other hand, when a term have many associated and related meanings, we need to overcome the usual disambiguation approach and relate the term to multiple meanings: i.e. to union of the meanings associated to it. Even Resnik and Yarowsky [87] ratify that there are common case where several fine-grained senses may be correct.

Let us examine a representative example; figure 1.5 shows the noun `bank` in WordNet, this term has ten possible meanings (synsets in WordNet). At least some meanings are obviously different, in other cases, however, the different meanings can be closely related (one meaning being a metaphorical or metonymic extesion of another). In general, the division of words into meanings is a very difficult issue, this is proved by the fact that different dictionaries will provide different divisions of words into meanings.

As shown in figure 1.5, some meanings may be grouped in different categories and sub-categories. One solution some researchers, like , have suggested in order to depict a word with its sense, is to choose not just a meaning but a group of meaning that represent the sense of the word.

In this case, for example, to disambiguate the noun “bank” with the meaning of “repository” it is correct to choose the first four synsets (this happen because these meanings are not all mutually exclusive).

For these reason, all the WSD methods, we have proposed, may associate more than one meaning to a term and is, thus, differs from the traditional approaches.

Once we have performed the annotation task, the elements of the sources have been enriched of new information: their meanings (synset in WordNet). These information are not unrelated, but they are connected in a lexical network (the relationships among WordNet synsets). After the annotation task, we found new lexical relationships across elements of different data sources.

In the database world, schema integration is based upon schema matching: since the schemata are independently developed, they often have different structure and terminology, even if they model the same real world domain. Schema matching can be greatly improved if the lexical relationships are considered alongside of structural relationships defined in the schemata.

1.5 An overview of the data integration systems

1.5.1 Data Integration Tools

Several surveys about the approaches in the data integration area have been published [56, 39, 62, 32, 86, 78, 93, 63]. This topic is generally divided into three categories: ontology development, ontology and schema matching and ontology alignment.

Concerning the ontology development, the ONTOWEB project published a complete technical report[10] where tools are classified on the basis of the implemented methodologies (from scratch, reengineering ontologies, based on a cooperative construction, and managing the evolution). Several researches address topics in the ontology matching area, i.e. the techniques for identifying similar concepts in different ontologies: in [86] several systems are evaluated on the basis of the generated mappings (five kinds of criteria are identified), [78] focuses on mapping discovery, reasoning and representation. The ontology alignment, i.e. the automated resolution of semantic correspondences between the elements of heterogeneous ontologies, is one of the new challenge in the ontology management and it includes ontology mapping and schema mapping. The Knowledgeweb Network of Excellence[11] has largely investigated about this issue publishing several reports.

[10] http://www.ontoweb.org deliverable 1.4

[11] http://knowledgeweb.semanticweb.org/

Ontology and schema matching and ontology alignment tool are deeply analyzed in [62], where four phases for semantic matching are individuated:

1. Pre-integration preparation (normalization, lifting)
   Pre-integration preparation, includes abstract model construction, syntax unification, etc. Source models are normalised and lifted to a uniform representation so that they remove conflicts caused by syntactic heterogeneity.

2. Similarity discovery
   The similarity discovery phase takes as input the uniformly represented source models and identifies the correspondences. Other functionalities included in this phase are matching identification, ranking, and confirmation (evaluation). While some pure mapping and matching systems end at this phase providing us with a set of correspondences, others proceed to the next phase of integration, similarity representation.

3. Similarity representation (also includes reasoning)
   The choice of representation formalisms for correspondences is critical in automated integration as well as supervised approaches. In early days, correspondences are enumerated in plain text or represented in pairwise fashion in tables for human experts to examine, which is not practical in a large, distributed environment, like the Semantic Web. One of the purposes of formally representing correspondences is to facilitate similarity execution.

4. Similarity execution (post-process)
   Depending on the strategies defined in the pre-integration phase, the output of similarity execution might be a concrete global semantic model representing the merge of source models; a virtual global view of source models; a set of articulation rules; and/or a query rewriting formula.

For each stage, the metodologies adopted by the 38 analyzed tools/systems are compared against each other. The results offering a complete vision of the state of the art. We report here some of the main tools with which to compare MOMIS and we represent in table 1.1, a comparison with MOMIS (no information about the pre-integration phase is given because it is a task typically executed by wrappers).

- **Clio**
  Clio [98] is a research prototype of a schema mapping creation tool.

The authors proposed a local query rewriting algorithm without constructing the unified integrated schema. Clio takes user defined value correspondences as inputs and tries to discover the mapping between more than one source schema on one hand and the target schema on the other. When multiple mappings are obtained, a ranking mechanism is applied. Queries are generated as results of Clio mapping to facilitate further direct access to source data. There are many differences between Clio and MOMIS: first, in the Clio framework the focus is on schema mapping issues, while the focus of our proposal is the semi-automatic generation of a "target" schema common to each source (the Global Virtual View). Moreover, our proposal relies on structural and lexical relationships among the sources.

- **COMA**
  COMA [37] (and COMA++ [2]) is a composite matcher, that provides an extensible library of different matchers and supports various aggregating and selecting strategies. In a single iteration of matching, based on the characteristics of the input schemata, the system invokes several matchers from a Matcher Library. Matching scores between elements from the two schemata, e.g. S1 and S2, are aggregated as a similarity cube. Matching scores produced by different matchers are combined into a similarity matrix between S1 and S2. A filter strategy is then applied to determine the most plausible ones as the resultant scores of the combined matching process. Matchers exploit linguistic, data-type, and structural information, plus previous matches, to produce similarity matrix. Then particular similarity values are selected as good match candidates, which are combined to a single value. This process is executed for whole schemas or for two schema elements, and is repeated after the user provides feedback. COMA supports a reuse approach focusing on existing mappings, which can be generalized for different reuse granularities, or fragment- and schema-level match results. The starting mappings (or similarity) are user-defined, unlike our approach that is mainly focused on the use of lexical dictionaries (like WordNet) for semantic relationships discovering.

- **CUPID**
  The CUPID [69] approach adopts the following strategy: compute the similarity coefficients between the two schemata and then deduce the mappings using those coefficients. The first phase is based on linguistic matching. The individual schema elements are matched based on their names, data types, domains, etc. CUPID makes use of a the-

saurus to help match names by identifying commonly used abbreviations, acronyms and synonyms. The result is a linguistic similarity coefficient, this matching includes: normalization, categorization, and comparison; each of these employ a number of popular natural language processing techniques from text engineering such as tokenization, stemming, tagging, elimination, expansion, etc. The second phase is based on structural matching. The schema elements are matched based on the similarity of their contexts or vicinities. The structural match depends partly on linguistic matches calculated previously. The result is a structural similarity coefficient whose computation algorithm checks first for atomic elements (leaves) that are similar (individually similar - linguistic and Data type) and if their respective vicinities (children, parents) are similar. Then it checks for non-leaf elements (if their subtrees are similar), and non-leaf elements that their immediate children do not match but are still highly similar (leaves represent the atomic data that the schema ultimately describes). The last phase is the creation of mappings. A mapping is created by choosing pairs of schema elements with maximal weighted similarity. The authors claim that once mappings are generated then they can be enriched by regarding sub-elements of mapped elements as mapped, e.g, two mapped XML elements will have their attributes (sub-elements) mapped as well. This is similar to the treatment of OWL SameAs but there are some practical considerations here with respect to the type of mapping extensions to CUPID: it generates generic schemata, hence, not apply the tree-based algorithms as real world schemas are rarely trees, since they share sub-structure and have referential constraints. They use XSD of which elements are regarded as XML elements, and they define and use three types of relationships: containment, aggregation and isDerivedFrom (generalization of IS-A and isTypeOf). They also match referential constraints in the sense that they interpret them as potential join views. Some engineering tuning of algorithm includes: (i) optional vs. required elements (helps with choosing the right nodes for the coecients' calculations) (ii) initial-mapping (similar idea to kick-o mappings used in IF-Map) used to reduce computation time and complexity (iii) views (where views are defined as referential constraints) (iv) lazy expansions, and (v) pruning leaves, if root and immediate children are the same, the rest is skipped.

- **FCA-Merge**
  Stumme and Maedche [95] presented the FCA-Merge method for ontology merging. It is based on Ganter and Wille's work on Formal Concept Analysis [49] and lattice exploration. The authors incorpo-

rate natural language techniques in FCA-Merge to derive a lattice of concepts. The lattice is then explored manually by a knowledge engineer who builds the merged ontology with semi-automatic guidance from FCA-Merge. In particular, FCA-Merge works as follows: the input to the method is a set of documents from which concepts and the ontologies to be merged are extracted. These documents should be representative of the domain at question and should be related to the ontologies. They also have to cover all concepts from both ontologies as well as separating them well enough. These strong assumptions have to be met in order to obtain good results from FCA-Merge. As this method relies heavily on the availability of classified instances in the ontologies to be merged, the authors argue that this will not be the case in most ontologies, the authors opt to extract instances from documents. The first step of FCA-Merge could be viewed as an ontology population mechanism. This initial step could be skipped, though, if there are shared classified instances in both ontologies. Once the instances are extracted, and the concept lattice is derived, Stumme and Maedche use Formal Concept Analysis techniques to generate the formal context for each ontology. They use lexical analysis to perform, among other things, retrieval of domain-specific information. Using this lexical analysis the authors associate complex expressions, like Hotel Schwarzer Adler with concept Hotel. Next, the two formal contexts are merged to generate a pruned concept lattice. This step involves disambiguation (since the two contexts may contain the same concepts) by means of indexing. The computation of the pruned concept lattice is done by an algorithm, TITANIC, which computes formal contexts via their key sets (or minimal generators). In terms of Formal Concept Analysis, the extents of concepts are not computed (these are the documents that they originate from, and are not needed for generating the merged ontology, the authors say), only the intents are taken into account (sets of concepts from the source ontologies). Finally, Stumme and Maedche do not compute the whole concept lattice, as it would provide too many too specific concepts. They restrict the computation to those formal concepts which are above at least one formal concept generated by an (ontology) concept of the source ontologies. Having the pruned concept lattice generated, FCA-Merge enters its last phase, the non-automatic construction of the merged ontology, with human interaction. This construction is semi-automatic as it requires background knowledge about the domain. The engineer has to resolve possible conflicts and duplicates, but there is automatic support from FCA-Merge in terms of a query/answering mechanism, which aims to guide and focus the engineer's attention on specific parts of the construction process. A number of heuristics are incorporated in this phase (like using the key sets of concepts for evidence of class membership), and the is a lattice is derived automatically.

- **GARLIC**
  Garlic [28] builds a wrapper-based architecture to describe the local source data using an object oriented language. The authors propose an approach where data and schema transformations are handled in a uniform fashion. Garlic is in principle a database middleware system, not a dedicated schema matching system. Its main component is a query processor which optimises and executes queries over diverse data sources posed in an object-extended SQL. The system interacts with wrappers which do the data transformations from source data to the middleware's model either directly or by constructing views over the middleware's schema. Garlic wrappers use an abstraction, Garlic objects, and the Garlic Definition Language to describe data from diverse resources into a uniform format, the wrapper's format. This is then used to produce the global schema. These objects are also used when they construct views as they encapsulated multiple data sources. Although they acknowledge the problem of schematic heterogeneity (data under one schema are represented as metadata in another), they don't tackle it directly. Instead, they have built a semi-automatic tool, Clio, which helps to integrate data and schema transformation processes.

- **GLUE and iMAP**
  GLUE [25] is an extension of LSD system [24]. GLUE and iMAP [40] are an extension of LSD system [38] whose goal is to semi-automatically find schema mappings for data integration. Like its ancestor LSD, Glue use machine learning techniques to find mappings[41]. GLUE is declared to be able to discover semantic matching by leveraging the instances associated with each schema. It is different from the previous LSD system which produces mappings between a local schema and a predefined global one. GLUE can work directly on local source schema. iMAP [22] is the latest development along this direction that has the capability to discover the so-called "complex matching", i.e. 1 : n matching. Given two ontologies, for each concept in one ontology, GLUE finds the most similar concept in the other ontology by using probabilistic definitions of several practical similarity measures. The authors claim that this is the difference when comparing their work with other machine learning approaches, where only a single similar-

ity measure is used. The similarity measures they employ is the joint probability distribution of the concepts involved, so instead of committing to a particular definition of similarity, GLUE calculates the joint distribution of the concepts, and lets the application use the joint distribution to compute any suitable similarity measure. This approach relies on data instances techniques. On the other hand, the MOMIS methodology is based on schema analysis (we are experimenting the introduction of instance based components, see [7] for some preliminary results).

- **OBSERVER**
  Mena and colleagues developed the Ontology Based System Enhanced with Relationships for Vocabulary hEterogeneity Resolution (OBSERVER) [74] in order to access heterogeneous, distributed, and independently developed data repositories. Their aim was to tackle the problem of semantic information integration by leveraging the relationships between domain-specific ontologies. They use interontology relationships such as synonyms, hyponyms and hypernyms defined between terms in different ontologies to assist the brokering of information across domain-specific ontologies. Their system is based on a query- expansion strategy where the user poses queries in one ontology's terms and the system tries to expand the query to other ontologies' terms. This is supported by algorithms to manage the relevance of information returned. As far as the mappings are concerned, they use the data structures underlying the domain-specific ontologies and the synonymy, hyponymy and hypernymy relations to inform linguistic matches between concepts.

- **PROMPT and PROMPTDIFF**
  Noy and Musen have developed a series of tools for performing ontology mapping, alignment and versioning [79, 80]. The tools use linguistic similarity matches between concepts for initiating the merging or alignment process, and then use the underlying ontological structures of *Protegè* to inform a set of heuristics for identifying further matches between the ontologies. The authors distinguish in their work between the notions of merging and alignment, where merging is defined as the creation of a single coherent ontology and alignment as establishing links between [ontologies] and allowing the aligned ontologies to reuse information from one another.

  PROMPT is a (semi-)automatic tool and provides guidance for the engineer throughout the steps performed during merging or alignment.

Tools/ Systems	Major objectives	Similarity Discovery	Similarity Representation	Similarity Execution
Clio	Rewriting	x		x
GARLIC	Information Integration	x		x
COMA	Matching	x		
CUPID	Matching	x		
FCA–MERGE	Mapping	x		
GLUE and iMAP	Matching	x		
OBSERVER	Rewriting	x		x
PROMPT and PROMPTDIFF	Ontology Mapping & Integration	x		
MOMIS	Information Integration	x	x	x

Table 1.1: Classification of semantic integration systems

PROMPT analyses the ontologies to be merged, and if linguistic matches are found, the merge is done automatically. Otherwise the user is prompted for further action. Their latest tool, PROMPTDIFF, is an algorithm which integrates different heuristic matchers for comparing ontology versions. The authors combine these matchers in a fixed-point manner, using the results of one matcher as input for others until the matcher produces no more changes. PROMPTDIFF addresses structure-based comparison of ontologies as its comparisons are based on the ontology structure and not their text serialisation, the authors argue.

1.5.2 MOMIS

The growth of information available on the Internet has required the development of new methods and tools to automatically recognize, process and manage information available in web sites or web-based applications. On the other side, one of the key issues faced in data integration projects is understanding the data to be integrated [54].

In both the direction, significant effort is needed in order to understand the semantic relationships between sources and convey those to the integration system. One of the most promising ideas of the Semantic Web is that the use of standard formats and shared vocabularies and ontologies will

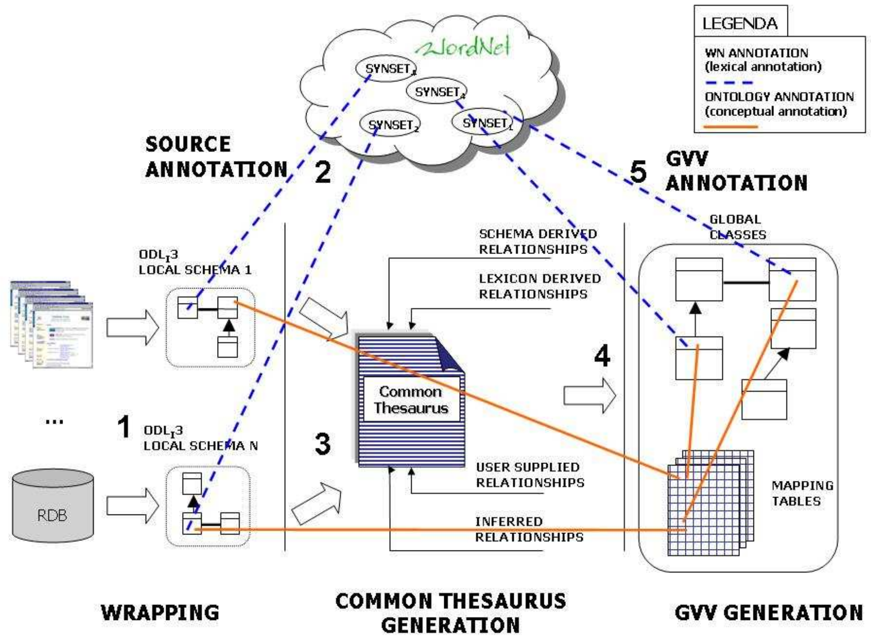


Figure 1.6: Functional representation of MOMIS

provide a well-defined basis for automated data integration and reuse. However, practical experience in developing semantic-enabled web applications and information systems shows that this idea is not so easy to implement.

In particular, we stress two critical issues: on the one hand, building an ontology for a domain is a very time consuming task, which requires skills and competencies which are not always available in enterprises; and, on the other hand, there is an irrefutable level of semantic heterogeneity, which has to do with the fact that different people/organizations tend to use "local" schemas for structuring their data, and ontologies if available at all are often designed to fit locally available data rather than aiming at being general specifications of domain knowledge. The consequence is a situation where data are organized to comply to some local schema (e.g. a relational schema, or a directory tree) and no explicit (formal) ontology is available; or if an ontology is available it is tailored on local data/schemas and therefore of little use for data integration.

The two issues above led the Semantic Web and Database communities to address two very hard problems: *ontology learning* (inducing ontologies from data/schemas) and *ontology matching/integration* (bridging different ontologies). For our argument, we only need to observe that several methods

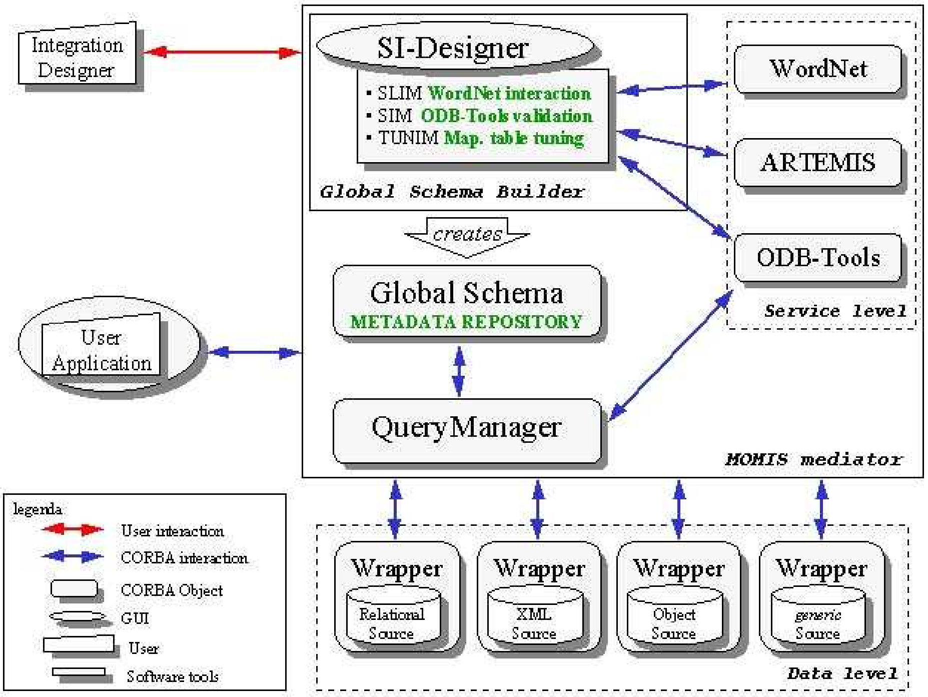


Figure 1.7: Architecture of the MOMIS system

and tools developed to address the two problems rely in different ways on the use of lexical information. The reason is simple: beyond the syntactic and semantic heterogeneity of schemas and ontologies, it is a fact that their elements and properties are named using natural language expressions, and that this is done precisely because they bring in useful (but often implicit) information on the intended meaning and use of the schema/ontology under construction.

Therefore, it should not come as a surprise that a large number of tools for ontology learning and schema/ontology matching includes some lexical resource (mainly WordNet[12]) as a component, and uses it in some intermediate step to annotate schema elements and ontology classes/properties with lexical knowledge. To sum up, lexical annotation seems to be a critical task to develop smart methods for ontology learning and matching.

The MOMIS[13] system is an I^3 framework designed for the integration of heterogeneous data sources. MOMIS starts from a collection of data sources and provides a collection of tools for:

1. semi-automatically building a customized ontology which represents

[12] See `http://wordnet.princeton.edu` for more information on WordNet.

[13] See http://www.dbgroup.unimore.it for references about the MOMIS project.

the information sources;

2. annotating each source according to the resulting ontology;
3. mapping the created ontology and the original sources into a lexical database (WordNet) to support interoperability with other applications.

MOMIS provides a double level of annotation for data sources and the resulting ontology: for each source, conceptual annotations map the original structure into a formalized ontology and lexical annotations assign a reference to a set of WordNet elements for each source term. Moreover, the ontology structure is formalized by means of a standard model and each concept is annotated according to a lexical reference.

The process of creating the ontology and defining the mappings is organized in five step (each task number is correspondingly represented in figure 1.6) : (1) local source schema extraction, (2) lexical annotation of local sources,(3) common thesaurus generation, (4) Global Virtual View (GVV) generation, and (5) GVV lexical annotation. The following paragraphs describe the details of these steps.

Local source schema extraction To enable MOMIS to manage web pages and data sources, we need specialized software (wrappers) for the construction of a semantically rich representations of the information sources by means of a common data model (see figure 1.7). The wrappers in MOMIS are the access point for the data sources. Each data source (relational, object, XML, ...) must be presented to MOMIS in a standard way, and this is what the wrapper does . A wrapper is a CORBA object that is connected to a source and is able to describe the source structure using the ODL$_{I^3}$ language and supplies a way to query to source using the OQL$_{I^3}$ language.

The wrapper architecture and interfaces are crucial, because wrappers are the focal point for managing the diversity of data sources. For conventional structured information sources (e.g. relational databases), schema description is always available and can be directly translated. For semistructured information sources, a schema description is in general not directly available at the sources. A basic characteristic of semistructured data is that they are “self-describing” hence information associated with the schema is specified within data. Thus, a wrapper has to implement a methodology to extract and explicitly represent the conceptual schema of a semi-structured source. We developed a wrapper for XML/DTDs files. By using that wrapper, DTD elements are translated into semi-structured objects, according to different proposed methods [1].

Lexical annotation of local sources For each element of the local schema, the integration designer has to choose the appropriate meanings w.r.t. WordNet lexical database. This task was completely performed manually until three years ago, then new annotation tools has been introduced in the MOMIS system, and now the user can choose to perform manual o automatic annotation.

The manual annotation is composed of two different steps: in the Word Form choice step, the WordNet morphologic processor aids the designer by suggesting a word form corresponding to the given term; in the Meaning choice step the designer can choose to map an element on zero, one or more senses. During the lexical annotation phase, one or more names (lemmas) are assigned to a source term. These names can be the original ones, extrapolated by the stemming techniques, or word forms chosen by the designer. Moreover the lexical annotation phase select a set of meanings (each one with a probability value in the probabilistic annotation), to each local class and attribute of the local schema.

A semi-automatic annotation tool has been introduced in MOMIS with the development of MELIS (see section 2.1 for a detail description). A user can partially annotate data sources and then MELIS, with a incremental process, deduces new lexical annotations.

Later on, a combined approach of two WSD algorithms has been suggested, it is called CWSD (see section 2.2 for a detail description). By CWSD it is possible to and performs an automatic annotation.

Afterwards, PWSD, a probabilistic lexical annotation tool, has been proposed (see section 2.3 for a detail description).

In the end, a GUI, addressed to both skilled and inexpert user, has been integrated in MOMIS (ALA, Automatic Lexical Annotator).The annotation panel of ALA enables the user to chose a set of WSD algorithms, to select how to execute the WSD algorithms (configuring the reliability of each algorithm), and to collect the outputs of the algorithms choosing a particular operator. ALA is independent of the algorithms and the operators implemented, and allows the programmer to add new algorithms or operators without recompile the source code. Now, in ALA are integrated five WSD algorithms: SD (Structural Disambiguation) [17], WND (WordNet Domains Disambiguation) [17], WordNet first sense heuristic, Gloss Similarity [11] and Iterative Gloss Similarity [11]. The GUI allows the user to select among three execution modalities (that make use of three operators): Pipe (Default), Parallel, Formula (see section 2.4 for details). Moreover, the ALA panel displays the statistical results after an execution, this allows expert users to re-configure the algorithms or the modalities and improve the automatic annotation.

By the manual annotation or by the use of MELIS and CWSD, the annotation are ordinary. These means that more than one meaning can be associate to a term (for a formal description see the definition 2.2 in chapter 2). Using PWSD or ALA the annotation become probabilistic. These methods provide for each term a set of lexical annotations where each annotation is associate to a probability value (for a formal description see the definition 2.1 in chapter 2).

Common thesaurus generation The common thesaurus is a set of relationships describing inter- and intra-schema knowledge about the source schemas. The common thesaurus is constructed through a process that incrementally adds four types of relationships: schema-derived relationships, lexicon derived relationships, designer-supplied relationships and inferred relationships.

- **Schema-derived relationships.** The system automatically extracts these relationships by analyzing each schema separately and applying a heuristic defined for the specific kind of source managed.
- **Lexicon-derived relationships.** These relationships, generated by lexical annotations, represent complex relationships between meanings of terms annotated with lexical senses. These relationships may be inferred from lexical knowledge (e.g. by querying WordNet for relationships across senses).
- **Designer-supplied relationships.** To capture specific domain knowledge, designers can supply new relationships directly.
- **Inferred relationships.** MOMIS exploits description logic techniques from ODB-Tools [10] to infer new relationships.

Here are defined the structural and lexical ODL$_{I^3}$ relationships.

Definition 1.7 ***The structural ODL$_{I^3}$ relationship***
The structural ODL$_{I^3}$ relationships are:

- *BT_{EXT}: `t1` subsumes `t2` iff t2 ISA t1 (the opposite of BT_{EXT} is NT_{EXT});*
- *RT_{EXT}: `t1` is related to `t2` iff `t1` is a property of `t2`.*

Structural relationships are automatically extracted by the MOMIS wrapper and ODB-Tools [9].

Definition 1.8 ***The lexical ODL$_{I^3}$ relationship***
The lexical ODL$_{I^3}$ relationships are defined on the basis of thesaurus relationships:

- *SYN: (Synonym-of), defined between two terms that are synonymous (i.e. a synonym relationship holds between the terms in the thesaurus);*
- *BT: (Broader Term), defined between two terms where the first generalizes the second (i.e. a hypernym relationship holds between the terms in the thesaurus), the opposite of BT is NT, Narrower Term (i.e. a hyponym relationship holds between the terms in the thesaurus);*
- *RT: (Related Term) defined between two terms that are related(i.e. a holonym relationship or a meronym relationships holds between the terms in the thesaurus).*

Probabilistic lexical annotations (supplied by PWSD and ALA) lead to a probabilistic evaluation of the lexical relationships (each lexical relationship has a probability value that describe its reliability). A probabilistic relationship holds between two terms, if it exists a lexical relationships between their meanings in the lexical database WordNet .

With the introduction of probabilistic relationship the Common Thesaurus (CT) has been transformed in a Probabilistic Common Thesaurus (PCT) (see section 2.3.2 for details).

GVV generation The MOMIS methodology allows us to identify similar ODL$_{I^3}$ classes (i.e. classes that describe the same or semantically related concept in different sources). To this end, affinity coefficients are evaluated for all possible pairs of ODL$_{I^3}$ classes, based on the relationships in the Common Thesaurus properly strengthened. Affinity coefficients determine the degree of matching of two classes based on their names (*Name Affinity coefficient*) and their attributes (*Structural Affinity coefficient*) and are fused into the *Global Affinity coefficient*, calculated by means of the linear combination of the two coefficients [29]. Global affinity coefficients are then used by a hierarchical clustering algorithm, to classify ODL$_{I^3}$ classes according to their degree of affinity. For each cluster Cl, a Global Class GC, with a set of Global Attributes $GA_1, , GA_N$, and a Mapping Table MT, expressing mappings between local and global attributes, are defined. The Mapping Table is a table whose columns represent the local classes, which belong to the Global Class and whose rows represent the global attributes. An element $MT[GA][LC]$ is a function which represents how local attributes of LC are mapped into the global attribute GA:

$MT[GA][LC] = f(LAS)$
where LAS is a subset of the local attributes of LC.

The Global Virtual View (GVV) consists of a set of classes (called Global Classes), plus mappings to connect the global attributes of each global class and the local sources attributes. Such a view conceptualizes the underlying domain; you can think of it as an ontology describing the sources involved.

GVV lexical annotation To annotate a GVV means to assign a name and a set (eventually empty) of meanings to each global element (class or attribute).

MOMIS automatically proposes a name and meanings for each global class of a GVV [8] (considering the set of all its ”broadest” local classes, w.r.t. the relationships included in the Common Thesaurus). Names and meanings have to be confirmed by the ontology designer.

1.6 Towards a dynamic data integration system

Data integration systems offer a single-point interface to a set of data sources. A data integration application is typically built by creating a mediated schema for the domain at hand, and creating semantic mappings between the schemas of the data sources and the mediated schema. The user (or application) poses queries using the terminology of the mediated schema, and the query is reformulated onto the sources using the semantic mappings. Despite recent progress in the field, setting up and maintaining a data integration application still requires significant upfront and ongoing effort. Hence, reducing the effort required to set up a data integration application, often referred to as on-the-fly integration, has been a recurring challenge for the field. In fact, as pointed out in [46], many application contexts (e.g., the web, personal information management, enterprise intranets) do not require full integration in order to provide useful services. This observation led to proposing a pay-as-you-go approach to integration, where the system starts with very few (or inaccurate) semantic mappings and these mappings are improved over time as deemed necessary [91].

Chapter 2

Towards an automatic probabilistic annotator

In most real world applications, ontology elements are labeled by natural language expressions. In our opinion, the crucial reason for this aspect of ontology engineering is the following: while conceptual annotations provide a specification of how some terminology is used to describe some domain (the standard role of OWL ontologies), natural language labels (lexical annotations) provide a natural and rich connection between formal objects (e.g. OWL classes and properties) and their *intended* meaning. The intuition is that grasping the intended interpretation of an ontology requires not only an understanding of the formal properties of the conceptual schema, but also knowledge about the meaning of labels used for the ontology elements. In other words, an OWL ontology can be viewed as a collection of *formal* constraints between terms, whose intended meaning also depends on lexical knowledge.

Lexical annotation is a difficult task, and making it accurate may require a heavy user involvement. Here are some of the reasons:

- **coverage**: a complete lexical database including all possible terms does not exist. WordNet, for example, contains a very large number of general terms, but does not cover specialized domains, whereas specialized lexical databases tend to disregard general terms;

- **polysemy**: in natural language, many terms are polysemous, namely may have many possible meanings. The choice of the specific meaning associated to the term is context dependent, and therefore this choice (called word sense disambiguation in NLP) is very difficult to automate;

- **compound terms**: schemas and ontologies are often labeled with compound nominal expressions, like “full professor”, “table leg”, “football team”. Compound terms do not appear in any lexical database, unless they form a stable compound (e.g. “station wagon”). Their annotation is therefore more difficult, as the choice of the right lexical meaning often depends on determine, it is difficult to associate meaning to the relationship between term in a compound term;

- **acronyms and abbreviations**: schemas and ontologies are often labeled with words whose meaning is not always obvious and can be misinterpreted; in these cases it is necessary to collect the words in a reference table in which they explained their meanings;

- **integration** a standard model/language for describing lexical databases does not exist. Consequently, it is difficult to integrate different lexical resources.

That is why several tools which were developed for annotating sources only provide a GUI for supporting the user in the manual execution of the task. However, this manual work can be highly time consuming, and very tedious for humans.

The research activities, during my PhD, has been focus on the study and development of several methods to perform the lexical annotation of structured and semi-structured data sources. All these methods accomplish two important tasks: (1) the source lexical annotation process, i.e. the operation of associating an element of a lexical reference database (WordNet in our case) to all source elements, (2) the discovery of mappings among concepts of distributed data sources/ontologies.

After the lexical annotation process, exploiting the network of lexical relationships among the meanings associated to terms, it is possible to derive lexical relationships between concepts. This lead to the discovery of mappings among concepts.

One key aspect is that it has been more and more crucial that annotation process is performed automatically. The manual annotation process is a boring and time-consuming task, moreover it is boring and difficult for a user to have a overall view of the sources and understand the context in which a term has been posed. While for small sources a user can be able to perform a good annotation, this task is harder for large sources. Large sources can not be explored by a user in an overall view, on these sources it become more efficient to use automatic methods. Automatic techniques do not have any difficulty in applying the rules on a large scale and exploring all in the branch in deep (for example exploring the complete hierarchical chain of a concept in an ontology).

Different methods to disambiguate structured and semi-structured data sources have been developed and tested within the MOMIS system: two in CWSD [16], one in MELIS [13], and others based on the gloss similarity shown in [11]. The results of the cited methods are good, even if not totally satisfying as they do not disambiguate all the terms in a data integration scenario. In [16] we discovered that the combination of methods is an effective way of improving the WSD process performance, then, we focused on how the different method have to be combined.

Instead of forcing to determine a unique best meaning of a term, we proposed the PWSD method (see section 2.3) that automatically annotates source terms and associates to any annotation a probability value that indicates the reliability level of the annotation. Our idea is supported by Renisk and Yarowsky. In [87], they argue the problem of disambiguation is not confined to search for the best meaning, instead, it is significant that a method reduces all possible meanings associate to a term and cheques, within this set, accurate probability to the correct meanings.

The PWSD method is based on a probabilistic combination of different WSD algorithms. The use of different WSD algorithms leads to an epistemic uncertainty, i.e. the type of uncertainty which results from the lack of knowledge about a system; for this reason, we studied which probabilistic theory best deals with this uncertainty and we focus on the Dempster-Shafer theory.

After the annotation task, it is possible to extract lexical relationships across elements of different data sources. Moreover, we discovered that with combined approaches, like PWSD, we can get more lexical relationships among terms than what can be achieved with a single WSD algorithms.

In this chapter the different annotation methods where I have taken part will described. In section 2.1 the MELIS method is presented. MELIS is a method and software tool for incrementally annotating data sources according to a lexical database (WordNet in our approach). MELIS has been developed in conjunction with the University of Trento. MELIS exploits the annotation of a subset of source elements to infer annotations for the remaining source elements, so that improving the activity of manual annotation.

In section 2.2, the software tool based on the CWSD (Combined Word Sense Disambiguation) algorithm, will be explored. CWSD is a combined method based on the exploitation of WordNet Domains, on the utilization of the structural knowledge of a data source and on the employment of the extension of the lexical annotation module of the MOMIS data integration system (*WNEditor*). The distinguishing feature of the algorithm is its low dependence on human intervention. CWSD consists of two algorithms: SD (Structural Disambiguation) which tries to disambiguate terms by using semantic relationships inferred from the structure of data sources; WND

(WordNet Domains Disambiguation) tries to disambiguate terms exploiting domains information supplied by WordNet Domains.

Starting from the evaluation of CWSD, it has been detected that combining a variety of WSD algorithms maximize the performance of the annotation tool. The use of different WSD algorithms leads to an epistemic uncertainty; for this reason, I started looking for probabilistic techniques to combines more WSD algorithms [84]. The study has led me to the discovery of a proper theory for handling the uncertainty caused by the combination of WSD algorithms: the Dempster-Shafer theory [92, 82]. Thanks to this theory, PWSD (Probabilistic Word Sense Disambiguation) has been developed (that is described in section 2.3). PWSD is a probabilistic method to combines the results of more WSD algorithms; it automatically annotates terms of data sources and associates to any annotation a probability value that indicates the reliability level of the annotation. In practice, it extends CWSD with uncertainty in annotation (i.e. probabilistic annotation). The relevance of these senses is probabilistically determined through the application of the Dempster-Shafer theory.

Studying the behaviour of PWSD, we affirm that uncertainty in data integration is best coped with by using a probabilistic view and we present in section2.4 ALA (Automatic Lexical Annotation), a tool for automatic lexical annotation, that permit to discover probabilistic relationships between heterogeneous data sources and to collect them in a Probabilistic Common Thesaurus (PCT). ALA focused on the creation of an annotation GUI that integrates all the WSD algorithms previously developed and combines them with different operators. This tool is addressed to both skilled and inexpert user.

The annotation panel of ALA enables the user to chose a set of WSD algorithms, to select how to execute the WSD algorithms (configuring the reliability of each algorithm), and to collect the outputs of the algorithms choosing a particular operator. ALA is independent of the algorithms and the operators implemented, and allows the programmer to add new algorithms or operators without recompile the source code.

2.1 MELIS: the lexical knowledge component

MELIS (**M**eaning **E**licitation and **L**exical **I**ntegration **S**ystem) supports the annotation process by automatically providing a candidate lexical annotation of the source terms as the combination of lexical knowledge (from WordNet) and domain knowledge (if available). In addition, MELIS uses the *WNEditor* [6] to support customized extensions of WordNet with missing words and senses.

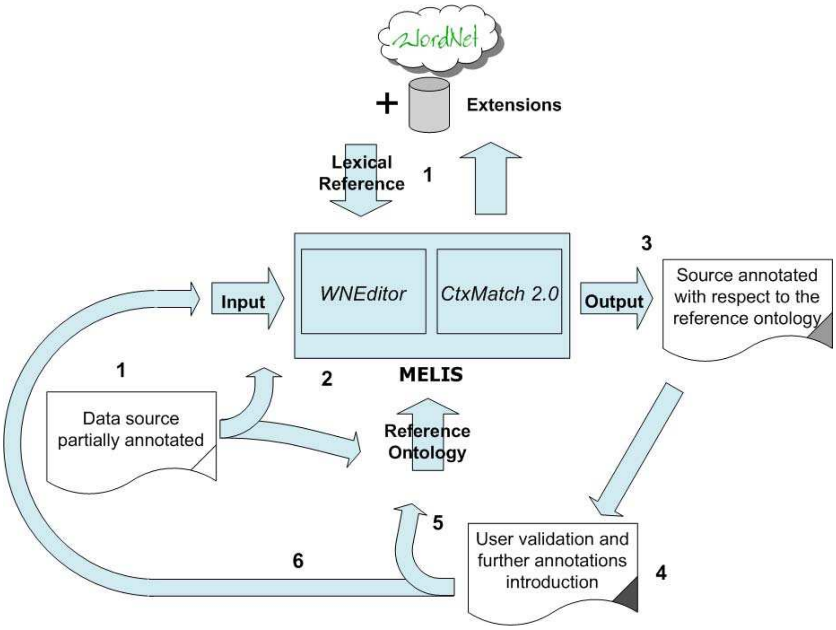

WordNet
+
Extensions
Lexical Reference
1
WNEditor
CtxMatch 2.0
MELIS
Input
Output
3
Source annotated with respect to the reference ontology
1
Data source partially annotated
2
Reference Ontology
5
6
User validation and further annotations introduction
4

Figure 2.1: Functional representation of MELIS

MELIS has been experimented in MOMIS to show that it can improve the MOMIS methodology in two main directions: by supporting the semi-automatic annotation of the original data sources (currently the process is manually executed), and by providing methods for extracting rich relationships across terms by exploiting lexical and domain knowledge. MELIS inside MOMIS allows a greater automation in the process of source annotation, and provides a way for discovering relationships among sources elements.

In the following we describe the MELIS method, its heuristic rules and the main features of *WNEditor*.

2.1.1 The MELIS method

The way MELIS works is depicted in Figure 2.1. We start from a collection of data sources which cover related domains, e.g. hotels and restaurants. In general we do not assume that a domain ontology is initially available, though this may be the case. The process is a cycle which goes as follows:

1. a schema, which can be already partially annotated with lexical information, is given as input to MELIS, together with a (possibly empty) domain ontology (considered as a reference ontology for the system).

Lexical information is extracted from WordNet which may be extended with words/senses which are not available by interacting with *WNEditor*;

2. the automatic lexical annotation process starts; its output is a partial annotation of schema elements, together with a list of discovered relations across different elements. This annotation, whose main rules are described below, is obtained by using two main knowledge sources: WordNet (for lexical senses and relations across them), and the reference ontology, if not empty (it provides non-lexical – domain dependent – relations across senses, e.g. between "hotel" and "price"). Pre-existing lexical annotations are not modified, as they may come either from manual annotation or from a previous annotation round;

3. the resulting annotated schema is passed to a user, who may validate/complete the annotation produced by MELIS;

4. the relations discovered across terms of the schema are added to the reference ontology (which means that an extended – and lexically annotated – version of the domain ontology is produced, even if initially it was empty);

5. the process restarts with the following schema, if any; otherwise it stops.

The process is incremental, as at any round the lexical database and the reference ontology may be extended and refined. As we said, the process might even start with an empty reference ontology, and the ontology is then constructed incrementally from scratch.

2.1.2 The rules for generating new annotations

A crucial part of the process has to do with the rules which are used to produce the MELIS lexical annotations. The core rules are derived from CtxMatch2.0 (the details are shown in [20, 21]). However, to improve the precision and recall of MELIS, we added a few specialized heuristic rules.

The annotation process takes as input a schema O and works in two main steps: first, for every label in O, the method extracts from WordNet all possible senses for the words composing the label; then, it filters out unlikely senses through some heuristic rules. The remaining senses are added as lexical annotation. Below is a general description of the heuristic rules used by MELIS. See appendix A for a graphical representation and an example of application of each rule.

To illustrate the MELIS heuristic rules, we will use the following notational conventions:

- Letters: capital letters (`A`, `B`, `C`, ...) stand for class labels, low case letters (`a`, `b`, `c`, ...) stand for datatype property labels, letters followed by "`#n`" (where `n` is a natural number) refer to the `n`-th sense of the label for which the letter stands in WordNet(e.g. $b_{\#2}$ is the synset associated to the second sense of the word occurring in the label "b").
- Arrows: arrows denote a subclass relation when link two classes, arrows linking a class and a property denote datatype properties, dashed arrows denote object properties.
- Ontologies: O is used for the ontology to be annotated, DO_i for the i-th domain ontology available for the current elicitation process.

The elicitation process takes as input an ontology O and works in two main steps:

1. first, for each class and property label in O, the method extracts all candidate senses from WordNet, i.e. the candidate synsets consisting in sets of synonym words or collocations;
2. second, it filters out candidate senses following some heuristic rules, i.e. it selects the appropriate synset(s).

Below is a detailed description of the heuristic rules used by MELIS in the second step of the elicitation process. In Section 3.1.1 we provide a running example of their application on a specific domain (tourism), whereas in Appendix A, we provide a graphical representation of each rule, together with an intuitive example of its application.

Rule 1 *If in O we find a class labeled* `A` *with a datatype property* `b`*, and in some DO_i we find a class annotated as $A_{\#i}$ with a datatype property annotated as $b_{\#j}$, then we conclude that the annotations $A_{\#i}$ and $b_{\#j}$ are acceptable candidate annotations for* `A` *and* `b` *in O (see figure A.1).*

Rule 2 *If in O we find a class labeled* `A` *with a datatype property* `b`*, and in some DO_i we find a class annotated as $B_{\#j}$, with a datatype property annotated as $b_{\#k}$ and a subclass $A_{\#i}$, then we conclude that the annotations $A_{\#i}$ and $b_{\#k}$ are acceptable candidate annotations for* `A` *and* `b` *in O (see figure A.2).*

Rule 3 *If in O we find a class labeled* `A` *with a datatype property* `b`*, and in some DO_i we find a class annotated as $A_{\#i}$, with a subclass $B_{\#j}$, and the latter has associated a datatype property annotated as $b_{\#k}$, then we conclude that the annotations $A_{\#i}$ and $b_{\#k}$ are acceptable candidate annotations for* `A` *and* `b` *in O (see figure A.3)*[1].

Rule 4 *If in O we find a class labeled* `A` *with a datatype property* `b`*, and in some DO_i we find a class annotated as $C_{\#k}$ with two subclasses annotated as $A_{\#i}$ and $B_{\#j}$, and there is a datatype property annotated $b_{\#h}$ associated to $B_{\#j}$, then we conclude that the annotations $A_{\#i}$ and $b_{\#h}$ are acceptable candidate annotations for* `A` *and* `b` *in O(see figure A.4).*

Rule 5 *If in O we find a pair of classes labeled* `A` *and* `B`*, connected through any object property, and in DO_i we find a pair of classes annotated as $A_{\#i}$ and $C_{\#k}$, and $C_{\#k}$ has a subclass $B_{\#j}$, then we conclude that the annotations $A_{\#i}$ and $B_{\#j}$ are acceptable candidate annotations for* `A` *and* `B` *in O(see figure A.5).*

Rule 6 *If in O we find a pair of classes labeled* `A` *and* `B` *(with* `B` *subclass of* `A`*), and in DO_i we find a pair of classes annotated as $A_{\#i}$ and $B_{\#j}$ (with $B_{\#j}$ subclass of $A_{\#i}$), then we conclude that the annotations $A_{\#i}$ and $B_{\#j}$ are acceptable candidate annotations for* `A` *and* `B` *in O(see figure A.6).*

Rule 7 *If in O we find a pair of classes labeled* `A` *and* `B` *(with* `B` *subclass of* `A`*), and in some DO_i we find a subclass hierarchy in which two classes are annotated as $A_{\#i}, \ldots, B_{\#j}$ (with at least one intermediate class in between), then we conclude that the annotations $A_{\#i}$ and $B_{\#j}$ are acceptable candidate annotations for* `A` *and* `B` *in O(see figure A.7).*

When all heuristic rules are applied, we discharge any candidate pair of annotations which is not supported by any of the rules above.

2.1.3 The *WNEditor*

WNEditor aids the designer during the creation of (additional) specific-domain lexicon addressing the issue of consistent extension of WordNet with specific domain knowledge.

[1] Despite the first impression, this rule does not correspond to a form of inverse inheritance from child to parent nodes. The rule covers the case when in a domain ontology we find a property attached to the parent node and in the ontology to be annotated we find a pattern which corresponds to the property attached to the child node. This situation is very frequent. An intuitive example is given in the Appendix A. In a sense, this rule covers many situations in which the domain ontology and the ontology to be annotated are specified at a different level of granularity.

Extending WordNet WordNet is distributed *as-it-is* and external applications are not allowed to directly modify its data files. Therefore, *WNEditor* addresses two important issues:

1. providing a physical structure where WordNet and all its possible extensions are stored and efficiently retrieved;
2. developing a general technique which can support users in consistently extending WordNet.

The first issue is technically solved by storing the original WordNet (and all its possible extensions) in a relational database. The second issue is addressed by giving ontology designers the possibility to perform the following basic operations:

1. **Inserting new synsets.** To insert a new synset (i.e. meaning) for a term, the designer has to preliminary check whether such a synset already exists in the database. The *WNEditor* provides an *approximate matching technique* that computes *syntactic* and *semantic* similarity between the definitions associated to two synsets. The syntactic similarity function performs an approximate text match based on the edit distance or the name match[60]. The semantic similarity function exploits the heuristic known in literature as *definition match* approach. In particular, two different well-known IR techniques are implemented: *vector space model match*[3] and *latent semantic indexing match*[35].
2. **Inserting new lemmas.** We developed an *approximate string match* algorithm to perform the similarity search on the whole synset network based on the edit distance and on a *reverse index*, representing, at any time, the set of terms used within the reference ontology to build sense definitions.
3. **Inserting new relations.** *WNEditor* supports the designer in the definition of new relationships between synsets: given a source synset, the designer is assisted in searching for the most appropriate target synset. The implemented algorithm exploits synset definitions and the definition match heuristic for providing a list of candidate synsets the user has to confirm (see [6] for details).

Exporting WordNet extensions WordNet extensions may be exported and then reused in other applications. For this purpose, a basic technique supporting the sharing of different extension was developed:

- WordNet extensions are marked with the name of the data sources/-ontology and the user inserting them; when user2 imports WordNet extended by user1, user1 extensions are temporary and user2 may decide what extensions have to be added to his local WordNet version.

- all users are allowed to include new lemmas, synsets and relations in their local WordNet version;

- the system includes in the exported version of an annotated source/-ontology, both a code identifying the original WordNet version and a minimal subset of the extended annotations (i.e. it has to contain all and only the elements needed for rebuilding the new annotations starting from scratch);

2.2 The CWSD method

CWSD is a method for the automatic annotation of structured and semi-structured data sources. Rather than being mainly targeted to textual data sources like most of the traditional WSD algorithms found in the literature, this algorithm can exploit information coming from the structure of the sources together with the lexical knowledge associated with the term itself.

This approach exploits the lexical database WordNet integrated with the domain knowledge WordNet Domains. WordNet Domains is an extension of WordNet where each synset is labelled with one or more domain labels. The use of WordNet Domains ([50] [18]) allow to overcome one of the main issues of WordNet: its excessive granularity in distinguishing the different synsets makes the WSD process nontrivial for many real applications.

The structural knowledge, as we will see, is automatically extracted from the source using the ODB-Tools [] component and then coded in order to be used in the disambiguation process. The CWSD shows how the structural information of structural and semi-structural sources is a meaningful aspect to take advantage of during the annotation phase. In fact structural relationships can be used to infer new semantic relationships useful for the WSD process.

We have integrated CWSD in the MOMIS system in order to decrease the human intervention in the hard task of normal annotation of distributed data sources, but this approach may be applied in general P2P data sources.

In order to disambiguate the sense of an ambiguous word, any WSD algorithm receives as input (and works in) a *context*. Many algorithms in literature represent the context as a “bag-of-words” that must be disambiguated, and sometime the information of the word positions in the text [81]. Other approaches [4] consider a “window-of-context” around every target word and submit all the words in this window as input to the disambiguation algorithm.

In CWSD we define the context composed of: a set of terms (classes and attributes names) to be disambiguated, and a set of structural relationships among these terms included in a Common Thesaurus (CT) (as shown in figure 2.2).

Terms pertaining to the same context are disambiguated at the same time. Consequently, the wrong inclusion of some terms within the context, represents noise that can lead to a wrong disambiguation. For this reasons, the choice of the context represents a strategic issue of our approach. The determination of the context, must take into consideration two main factors:

- The more is the number of the considered terms, the greater will be the probability to introduce noise in the process;
- The less is the number of considered terms, the smaller will be the probability to find relationships among the considered terms.

The ideal context would have to include all and only the terms, whose related information concur to determine in the exact way their meanings. This presupposes the existence of relations between the terms that allow to recognize them like pertaining to the same context. At the moment we did not explore how to cluster terms in different context on the basis of the relationships among them. The *default context* is given by all the terms in the data source to be integrated and all the structural ODL$_{I^3}$ relationships among these terms.

CWSD is composed of two algorithms: SD (Structural Disambiguation) and WND (WordNet Domains Disambiguation). SD tries to disambiguate terms by using structural ODL$_{I^3}$ relationships and WND tries to disambiguate terms exploiting domains information supplied by WordNet Domains. Both the proposed algorithms, may associate more than one meaning to a term. To disambiguate a set of data sources, CWSD applies (separately) to each source SD and, after that, WND to disambiguate the remaining terms using domains information supplied by WordNet Domains.

After the annotation perform by CWSD on a context, we have source terms annotated and we can derived from these annotations new relationships among terms. All the relationships in MOMIS are collected in the CT and are encoded in ODL$_{I^3}$ language.

CT stores a set of ODL$_{I^3}$ relationships describing inter- and intra-schema relationships among a set of data source schemas. The ODL$_{I^3}$ (Object Definition Language with extensions for information integration) relationships can be structural or lexical.

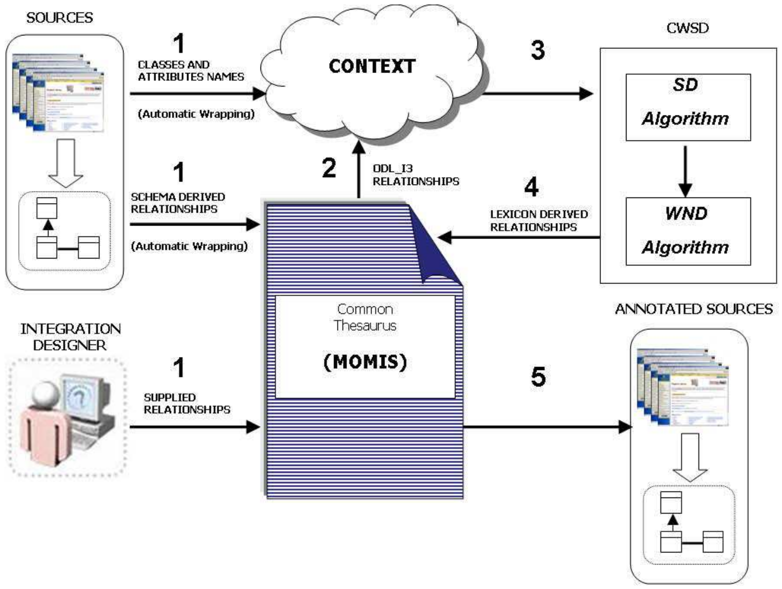


Figure 2.2: Automatic annotation of local data sources with CWSD

The lexical annotation we performed with CWSD is executed w.r.t. WordNet. The use of a well-known and shared lexical database provides a reliable set of meanings and allows the result of the disambiguation process to be shared with others, especially if the lexical resource is freely and publicly available (as WordNet is). Moreover, the fundamental peculiarity of a lexical database like WordNet is the presence of a wide network of semantic relationships between words and meanings (SYN,BT,RT).

The disadvantage in using a lexical database is that it does not cover with the same detail different domains of knowledge. Some terms may not be present in the thesaurus or, conversely, other terms may have many associated meanings. The first tests led to the need of expanding the lexical database with more specific terms (in this case, the MOMIS system already includes a component, *WNEditor*, which allows adding new terms and linking them within WordNet, see section 2.1.3). On the other hand, when a term have many associated and related meanings, we need to overcome the usual disambiguation approach and relate it to multiple meanings: i.e. union of its associated meanings.

2.2.1 The Structural Disambiguation algorithm (SD)

The SD algorithm exploits the structural ODL$_{I^3}$ relationships of a data source to annotate source terms and to infer lexical ODL$_{I^3}$ relationships on the basis of WordNet.

As described in [14], ODB-Tools extracts BT/NT relationships between the classes of one source, directly from the generalization hierarchies, and RT relationships, from the aggregation ones. Other relationships are obtained from foreign keys extracted by a relational schema; in the case in which a foreign key is applied on a primary key of both the tables, a relationship of BT (and the specular NT) is extracted, on the contrary only an RT relation is extracted.

By exploiting ODB-Tools capabilities and semantically rich schema descriptions, an initial set of BT, NT, RT can be automatically extracted. In particular, by translating ODLI3 into OLCD descriptions, ODB-Tools extracts BT/NT relationships among classes directly from generalization hierarchies, and RT relationships from aggregation hierarchies, respectively. Other RT relationships are extracted from the specification of foreign keys in relational source schemas. When a foreign key is also a primary key both in the original and in the referenced relation, a BT/NT relationship is extracted. Moreover, the attributes on which is defined the foreign key, on the referencing table and on the referenced table, are connected by a SYN relationships. In case of semi-structured sources, ODB-Tools extracts RT relationships, due to the nature of relationships defined in the semi-structured data model.

These structural ODL$_{I^3}$ relationships of BT, NT, SYN, RT, extracted by ODB-Tools, can be used in the lexical annotation process according to a lexical database.

SD tries to find a corresponding lexical relationship when a structural relationship holds among two terms. In practice, if we have a direct/chain of relationship between two terms, we try to find the semantically related meanings and annotate the terms with these meanings. A chain of relationships is obtained navigating through the lexical database relationships.

For all the NT_{EXT} relationships, SD finds the corresponding NT relationships in WordNet. The relation of equivalence (SYN_{EXT}) is used to find the corresponding SYN relationship in the lexical database. The RT_{EXT} relationship is used to find holonym or meronym relationships (RT) in the lexical database. When we do not find a direct lexical relationship between two meanings, we navigate through the lexical database relationships in order to find a path of lexical relationships that connect the two meanings (see Algorithm 1).

Algorithm 1 Structural disambiguation algorithm

```
Input: WordNet lexical Database and its extensions if any
$T = [t_i \;\; i = 1..Ncont]$ the set of the terms to be disambiguated;
$R = [r_k \;\; k = 1..Nrel]$ the set of the structural relationships linking two different terms $t_i$ and $t_j$;
Variables:
$S_i = [s_{iy} \;\; y = 1..Nsyn_i]$ the set of all possible synsets related to the term $t_i$;
$F_{il} = [f_{yw} \;\; y = 1..Nsyn_i, w = 1..Nf_il]$ the set of synsets linking by a chain of hypernym relationships of length $l$ to one of the synsets $\in S_i$;
$ANNOT_i = [syn_{iz} \;\; z = 1..Ncsyn_i]$ the set of synsets chosen by the algorithm to disambiguate the term $t_i$;
for all $r_{ij} \in R$ that link two terms $t_i, t_j \in T$ do
    if $r_ij$ is a BT relationship then
        determine $S_i$ and $S_j$;
        initialize $l = 1$ and $annotation = false$;

        repeat
            determine $F_{il}$;
            if $F_{il}$ not empty then
                for all $s_{jz}$ in $S_j$ do
                    if exist a $f_{yw} = s_{jz}$ where $s_{jz} \in S_j$ and $f_{yw} \in F_{il}$ then
                        insert the synset $s_{iy}$ in $ANNOT_i$;
                        insert the synset $s_{jz}$ in $ANNOT_j$;
                        set $annotation$=true;
                    end if
                end for
            end if
            set $l = l + 1$;
        until ($annotation = true$) or (no more hypernyms)
    end if
    if $r_ij$ is a SYN relationship then
        if $t_i \neq t_j$ then
            set $ANNOT_i = ANNOT_j = S_i \cap S_j$;
        end if
    end if
end for
Output: different set $ANNOT_i$ for each term $t_i$ that have a structural relation in R.
```

2.2.2 The WordNet Domains algorithm (WND)

WordNet Domains [50]can be considered an extended version of WordNet, (or a lexical resource) in which meanings (synsets of WordNet) have been annotated with one or more domain labels. The information brought by domains is complementary with the one already present in WordNet. Besides, domains may group set of synsets of the same word into a thematic cluster which has the important side effect of reducing the level of ambiguity of polysemic words. WordNet Domains organises about two hundred domains in a hierarchy, where each level is made up of codes of the same degree of specificity, as described in [18]. Some synset does not belong to a specific domain, but rather can appear in almost all of them. For this reason, a factotum label has been created which basically includes this types of synsets: generic synset (e.g. Man #1 - an adult male person), stop sense synsets (e.g. colours, numbers, ecc).

Algorithm 2 WordNet Domains disambiguation algorithm

```
Input: WordNet lexical Database and its extensions if any
$T = [ti \quad i = 1..Ncont]$ the set of the terms to be disambiguated;
$NMaxDOM$ the maximum number of domains we want to use in the algorithm.
Variables:
$S_i = [s_{iy} \quad y = 1..Nsyn_i]$ the set of all possible synsets related to the term $t_i$;
$D_j = [d_k \quad k = 1..Ndom_j]$ the set of the possible domains related to the synset $s_j$;
$DOM = [dom_x \quad x = 1..Ndom]$ an ordered set of the domains related to the set of the terms $T$;
$FreqDOM = [f_x \quad x = 1..Ndom]$ the corresponding set of the frequency of the domains related to the set of the terms $T$;
$ANNOT_i = [syn_{iz} \quad z = 1..Ncsyn_i]$ the set of synsets chosen by the algorithm to disambiguate the term $t_i$;
for all $t_i$ in $T$ do
    for all $s_{ij}$ in $S_i$ do
        for all $d_{jk}$ in $D_j$ do
            if $d_{jk} \in DOM$ then
                increase the $FreqDOM_k$;
            else
                insert the domain $d_jk$ in $DOM$ and set $FreqDOM_k = 1$;
            end if
        end for
    end for
end for
for all $t_i$ in $T$ do
    if $t_i$ is a monosemic term then
        $ANNOT_i = syn_{ij}$;
    else
        set $annotation = false$;
        for $x = 1$ to $NMaxDOM$ do
            for all $s_{ij}$ in $S_i$ do
                if $d_x$ is contained in $D_j$ then
                    insert the synset $syn_{ij}$ in the $ANNOT_i$;
                    set $annotation = true$;
                end if
            end for
            if $annotation = true$ then
                BREAK the cycle FOR;
            end if
        end for
    end if
end for
for all $t_i$ in $T$ do
    if $ANNOT_i$ is empty then
        for all $s_{ij}$ in $S_i$ do
            if $factotum$ is contained in $D_j$ then
                insert the synset $syn_{ij}$ in the $ANNOT_i$;
            end if
        end for
    end if
end for
Output: a list $DOM$ of the more frequent domains, and a set $ANNOT_i$ of sysnset that disambiguate each terms in $T$.
```

The availability of WordNet Domains[2] has allowed us to undertake a domain-oriented analysis of the structural or semi-structural data sources and to implement an effective WSD algorithm based on domain information.

[2] See http://wndomains.itc.it

The WND algorithm (see Algorithm 2) takes inspiration from the one proposed in [70]. First, we examine all the possible synsets connected to a term and extract the domains associated to these synsets, with this information we calculate a list of the *prevalent domains* in the chosen context. Then, we compare this list of domains with the ones associated to each term. For a term we choose as the correct synsets all the synsets associated to the prevalent domains.

In WordNet Domains there is a particular domain called "factotum" which is the domain associated to synsets that do not belong to a specific domain and in general it is the most frequent domain in a context. In accordance with [25], we do not use the "factotum" domain. We calculate the most frequent domains in a context and, if a term does not have any synset related to one of these domains, we choose the first WordNet sense.

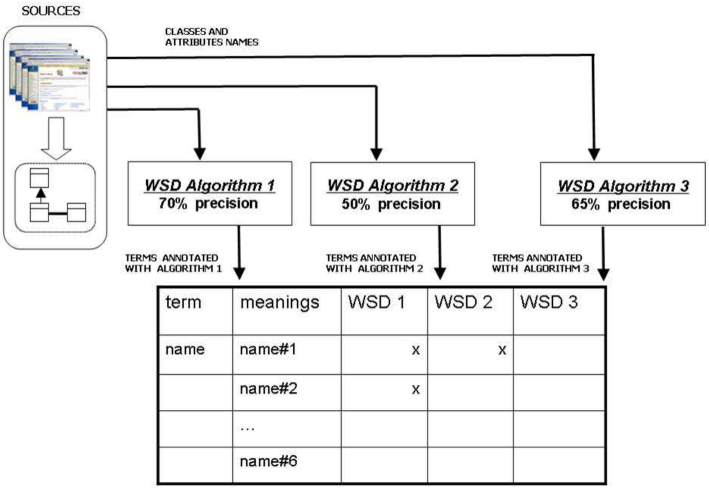
WND results depends on the context and on the *configuration* chosen. The configuration is the maximum number of domains we select for the disambiguation. The choice of the configuration and of the context (if not default) are delegated to the user. These are the only user interventions required.

2.3 The PWSD method

PWSD is based on a probabilistic combination of different WSD algorithms. The method is completely independent from the set of WSD algorithms chosen. All these algorithms need to be configured about their reliability. The default reliability for each WSD algorithm is based on its precision evaluated on a benchmark.

PWSD method allows to associate more than one meaning to a term. When a term has many associated and related meanings, we overcome the usual disambiguation approach and relate the term to the union of the meanings associated to it. In PWSD we still collect ordinary annotation.

Definition 2.1 ***Probabilistic Annotation***
Let T be a schema and t be a term (class or attribute name) $\in T$. We define $S_t = \{t_{\#1}, .., t_{\#n}\}$ as the set of all synsets for a term t w.r.t. a lexical resource (as WordNet). The probabilistic annotation of the term t is the triple (T, t, A_t), where $A_t = \{a_1, ..., a_k\}$ is the set of annotation associated to t. In particular, a_i is defined as the couple $(t_{\#i}, P(t_{\#i}))$, where $t_{\#i} \in S_t$ is a meaning for the term t, and $P(t_{\#i})$ is the probability value assigned (this probability indicates how well the meaning $t_{\#i}$ represented the term t).


SOURCES
CLASSES AND ATTRIBUTES NAMES
WSD Algorithm 1
70% precision
WSD Algorithm 2
50% precision
WSD Algorithm 3
65% precision
TERMS ANNOTATED WITH ALGORITHM 1
TERMS ANNOTATED WITH ALGORITHM 2
TERMS ANNOTATED WITH ALGORITHM 3

term	meanings	WSD 1	WSD 2	WSD 3
name	name#1	x	x	
	name#2	x		
	...			
	name#6			

Figure 2.3: WSD algorithms outputs

Definition 2.2 ***Ordinary Annotation***
Let T be a schema and t be a term (class or attribute name) $\in T$. An ordinary annotation for a term t is a probabilistic annotation where the probability value assigned to a set of meanings for a term t is equal to "1".

An example of an ordinary annotation might be a manual annotation (a user manually chooses one or more meanings to disambiguate a term), or an annotation assigned to a monosemous term. A monosemous term is described by a unique WordNet synset, therefore the disambiguation is certain for this term, and the probability value associated to the synset is equal to "1".

As we have shown in the previous sections, we have developed different types of WSD algorithms, which constitute an evolution of the ones proposed in the area of Natural Language Processing to disambiguate text, adapted to the case of structured and semi-structured data sources. The terms of these sources, used to label classes and attributes, are linked to each other differently from text data sources. For this reason, the set of WSD algorithms chosen for the application of PWSD, must include also algorithms able to exploit the structural information deriving by structured and semi-structured data sources. For example, our SD algorithm described in section 2.2.1, tries to disambiguate source terms exploiting the structural ODL$_{I^3}$ relationships extracted from the sources.

At the moment, we have developed five different WSD algorithms that we can combine using PWSD (SD, WND, WordNet first sense heuristic, Gloss similarity [11], Iterative gloss similarity [11]).

The use of different disambiguation algorithms leads to an epistemic uncertainty, i.e. the type of uncertainty which results from the lack of knowledge about a system. As a matter of fact, not every algorithm finds a meaning to be assigned to each term. In addition, each algorithm may be appropriate to certain situations, so its behavior is not 100% trustworthy. To maximize our system accuracy we employ a variety of WSD algorithms. Our point of view focuses on finding a flexible method that can combine a variable number of algorithms, thus obtaining results different from ones provided by each algorithm.

As a case in point, let us consider the term "name". In WordNet we found six different meanings for "name" ($name_{\#1}$, $name_{\#2}$, .., $name_{\#6}$). Suppose we have to combine three algorithms that give different outputs (like the case show in figure 2.3): WSD1 that chooses a set of meanings formed by $name_{\#1}$, $name_{\#2}$, WSD2 that provides $name_{\#1}$ as the correct meaning and WSD3 that does not give any result. What we want to obtain is a rate of confidence to be assigned to each possible meaning of the term under consideration.

In a probabilistic approach, different methods are applied, and, eventually, is not only evaluated the union or intersection of the possible meanings of a term, but the list of all its associated meanings with their probabilities. The probabilities will be associated to the methods on the basis of their reliability. So if a method is trusted, the method will have a high probability. Moreover, if a WSD method is iterative, it could obtain different meanings with different probabilities in each iteration.

At the beginning, the probability associated with a method was defined by the user who can interpret the experimental results provided by methods. For example, as we will show in section 3.2, the SD method has a higher precision then WND, then the user can associate to SD method a higher probability. Now, after the evaluation of our WSD algorithms on different test cases (see chapter 3), it is available a default configuration that assigns to each WSD algorithm a reliability based on its average precision evaluated on those scenarios.

What are the reasons to switch to an approximate approach? The combination of multiple disambiguation algorithms can be easily handled and the final results are more accurate than the results of a single method. Moreover, choosing more meanings for a term means that the number of discovered lexical relationships connecting a term to other meanings in the thesaurus increase. From our point of view, the PWSD is the first step to obtain probabilistic mapping. The set of lexical relationships together with the set of structural relationships in a dynamic integration environment are the input of the mapping process. Enriching the input of the mapping tool means to improve the discover mappings and so to refine the global schema. Moreover widening the use of probability to the discover relationships it is possible to compute probabilistic mappings. Finding probabilistic mapping is the basis of managing uncertainty in data integration system. And this lead to new method of query answering like suggesting in [42].

2.3.1 Uncertainty in disambiguation - The use of the Dempster-Shafer theory

The set of WSD algorithms defines a type of evidence that can be consistent or arbitrary. Because these types of evidence cannot be handled by the traditional probability theory without resorting to further assumptions of the probability distributions within a set, we decided to support the use of the Dempster-Shafer theory [92, 82]. This theory allows us to model ignorance through lack of knowledge.

The theory deals with the so-called *frame of discernment*, the set of base elements θ in which we are interested (in our case, the set of all possible meaning for the term under consideration), and it power set 2^{θ}, which is the set of all subsets of the interesting elements (all the possible subsets of the set of possible meanings). The basic of the measure of uncertainty is a *probability mass function m(·)* that assigns zero mass to empty set and a value [0,1] to each element of 2^{θ}, the total mass distributed being 1 so that:

$$\sum_{A \subseteq \theta} m(A) = 1$$

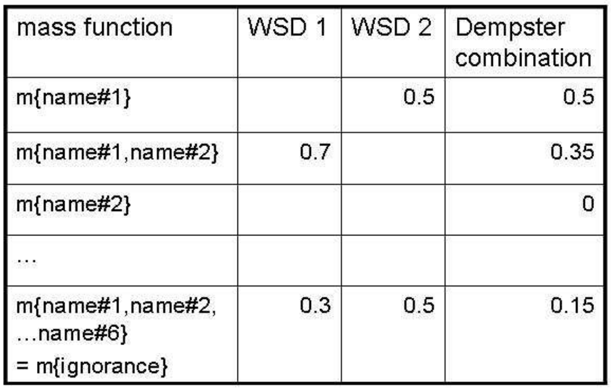
We can apportion the probability mass exactly as we wish, ignoring assignment to those levels of detail, that we know nothing about. We derive the belief mass function from the output and the precision of the WSD algorithms. To combine the output of the WSD algorithms we use the Dempster's rule of combination:

$$m(a) = K \sum_{\bigcap A_i = a} \prod_{1 \leq i \leq n} m_i(A_i)$$

$$K^{-1} = 1 - \sum_{\bigcap A_i = \emptyset} \prod_{1 \leq i \leq n} m_i(A_i) = \sum_{\bigcap A_i \neq \emptyset} \prod_{1 \leq i \leq n} m_i(A_i)$$

where n is the number of the WSD algorithms that supplied a disambiguation output for the term under analysis.

In the end, to obtain the probability assigned to each meaning we split the belief mass function concerning a set of meanings.



mass function	WSD 1	WSD 2	Dempster combination
m{name#1}		0.5	0.5
m{name#1,name#2}	0.7		0.35
m{name#2}			0
...			

m{name#1,name#2,
...name#6}
= m{ignorance}
	0.3	0.5	0.15

Figure 2.4: The mass functions assigned

Formula 2.1 ***Probability value associated to a probabilistic annotation***

$$P(syn) = \sum_{syn \in A} \frac{m(A)}{\|A\|}$$

Let us see an example of application of the PWSD method.

As shown in figure 2.3, through MOMIS, the sources terms are automatically extracted and automatically annotated by the application of the WSD algorithms. We might have to combine different disambiguation output. The PWSD method will not consider the algorithms that do not supply any annotation for the term. So, in this case, the evaluation will be executed only using the output from WSD1 and WSD2. Let us suppose that we configure the reliability of each algorithm as the precision of the algorithm evaluated on a benchmark; the complementary value will be the ignorance of the algorithm, i.e. the mass function assign to the entire set of meanings. If a WSD algorithm has a 70% of reliability this means that the algorithm has a 30% of ignorance.

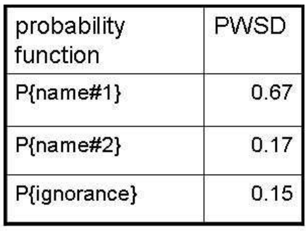
The application of the Dempster's rule of combination is shown in figure 2.4. As WSD1 supplies a set composed of two meanings, the probability will be assigned to this set.

So far, the mass function assign by the WSD1 algorithm is: $m_1\{name_{\#1}$, $name_{\#2}\}$=0.7 and $m_1\{$ignorance$\}$=0.3. While the WSD2 algorithm assigns: $m_2\{name_{\#1}\}$=0.5 and $m_2\{$ignorance$\}$=0.5.

Using the Dempster's rule of combination we obtain:
$m\{name_{\#1}\}= m_1\{name_{\#1}, name_{\#2}\}* m_2\{name_{\#1}\} +$
$+ m_1\{\text{ignorance}\} * m_2\{name_{\#1}\}= 0.35 + 0.15 = 0.5$

$m\{name_{\#2}\}= m_1\{name_{\#1},name_{\#2}\}^* \ m_2\{name_{\#2}\}= 0$
$m\{name_{\#1},name_{\#2}\}= m_1\{name_{\#1},name_{\#2}\}^* \ m_2\{\text{ignorance}\}= 0.35$
$m\{\text{ignorance}\}= m_1\{\text{ignorance}\}^* \ m_2\{\text{ignorance}\}= 0.15$

The results obtained after the application of the Dempster's rule of combination show the probability assigned to different set of meanings. To use this result for computing lexical relationships we have to bring back to the case of probabilities assigned to individual meanings. As shown in figure 2.5, the probability assigned to the set of meanings $\{name_{\#1},name_{\#2}\}$ will be split in the single probability assigned to $name_{\#1}$ and $name_{\#2}$.



probability function	PWSD
P{name#1}	0.67
P{name#2}	0.17
P{ignorance}	0.15

Figure 2.5: Generation of the probabilistic annotation

2.3.2 The Probabilistic Common Thesaurus (PCT)

The Probabilistic Common Thesaurus (PCT) is an extension of the Common Thesaurus (CT). It contains a set of ODL$_{I^3}$ relationships describing inter- and intra-schema knowledge among the source schemas. The ODL$_{I^3}$ relationships can be structural or lexicon derived and ordinary or probabilistic. The definition of structural and lexical ODL$_{I^3}$ relationships is the same than in the CT (see definitions 1.7 and 1.8 in section 1.5.2).

Definition 2.3 ***Probabilistic ODL$_{I^3}$ relationships***
A probabilistic ODL$_{I^3}$ relationship is a pair $(Rel_{ODL_{I^3}}, P(Rel_{ODL_{I^3}}))$*, where* $Rel_{ODL_{I^3}}$ *is a ODL$_{I^3}$ relationship and* $P(Rel_{ODL_{I^3}})$ *is a probability value, in the interval* $[0-1]$.

Definition 2.4 ***Ordinary ODL$_{I^3}$ relationships***
An ordinary ODL$_{I^3}$ relationship is a probabilistic ODL$_{I^3}$ relationship with probability value equal to "1".

Lexical ODL$_{I^3}$ relationships can be both probabilistic and ordinary; structural ODL$_{I^3}$ relationship are only ordinary, because deriving directly by the local source structure. In addition to this relationships, other ordinary ODL$_{I^3}$

relationships can be supplied directly by the designer, interacting with the MOMIS Ontology Builder .

MOMIS exploits description logic techniques (ODB-Tools) [9] to infer new relationships by applying subsumption computation to “virtual schemas” obtained by interpreting BT and NT as subclass relationships and RT as domain attributes.

2.3.3 From the probabilistic annotations to the discovery of probabilistic relationships

As previously described in section 1.5.2, MOMIS uses the annotation output to compute the lexicon relationships to be included in the PCT (Probabilistic Common Thesaurus). The application of the PWSD method associates a term in a source to a set of probabilistic meanings. Therefore, a term `t` is described by the meaning $t_{\#i}$ characterized with a certain probability. Because all the provided meanings are included in the lexical resource WordNet, each of them is located within a network of lexical relationships. When we assign the meaning $t_{\#i}$ to the term `t`, `t` will inherit the same lexical relationships that occur for the synset $t_{\#i}$ within the WordNet relationships network.

In a data integration scenario, we restrict to the sub-network of relationships that branch off from $t_{\#i}$, in the context of analysis of the sources to be integrated. From this sub-net of lexical relationships between meanings, lexical ODL$_{I^3}$ relationships are derived. MOMIS derives lexical ODL$_{I^3}$ relationships between local sources terms from the semantic relationships defined in WordNet between meanings. It generates lexical ODL$_{I^3}$ relationships by using the following WordNet constructors:

- SYNONYMY (similar relation) corresponds to a SYN relationship;
- HYPONYMY (sub-name relation) corresponds to an NT relationship;
- HYPERNYMY (super-name relation) corresponds to a BT relationship;
- HOLONYMY (whole-name relation) corresponds to an RT relationship;
- MERONYMY (part-name relation) corresponds to an RT relationship in ODL$_{I^3}$.
- CORRELATION (two terms share the same hypernym) corresponds to a RT relationship in ODL$_{I^3}$.

A probabilistic relationship holds between two terms, if exists a lexical relationships between their meanings in the lexical database WordNet. The probability assigned to the lexical relationships depends on the probability value of the meaning under consideration for a term.

Thank to the formula of the *join probability*, the probability value associated to a probabilistic ODL$_{I^3}$ relationship (see definition 2.3) holding among $t_{\#i}$ and $s_{\#j}$ can be defined as:

Formula 2.2 ***Probability value associated to a probabilistic ODL$_{I^3}$ relationship***

$$P(Rel_{ODL_{I^3}}(t_i, s_j)) = P(t_i) * P(s_j)$$

Collecting these probabilistic ODL$_{I^3}$ lexical relationships we populate the PCT that already contains the structural ODL$_{I^3}$ relationships. PCT is organized in a structure similar to an *Associative Network*, where nodes (classes or attributes names), are connected through bidirectional ODL$_{I^3}$ relationships. Eventually, to decrease the introduction of errors, probabilistic relationships with a probability value under a certain threshold can be filtered.

2.4 ALA Overview

After the development of different WSD algorithms and after the study of a probabilistic combination (PWSD), we started thinking about the fact that not just one o two combinations of algorithm are possible, but that an annotation designer could choose the most bizarre combination that he/she likes. This idea was suggested by the knowledge that the best combination of algorithms is not fixed, but depends on the context and on the kind of data sources we want to annotate. An expert annotation designer might know which kind of algorithms are better than others on a particular set of sources and what is the best combination of the WSD algorithms. Moreover, it could be useful for the user to be able to tune the annotation process (trying different executions of automatic annotation with specific configuration and comparing the results). Therefore, we tried to answer all these needs with a GUI to run the automatic annotation, configure the parameters and compare the results.

This tool is called ALA (Automatic Lexical Annotation). ALA is a flexible method that may combine a variable number of WSD algorithms and their outputs.

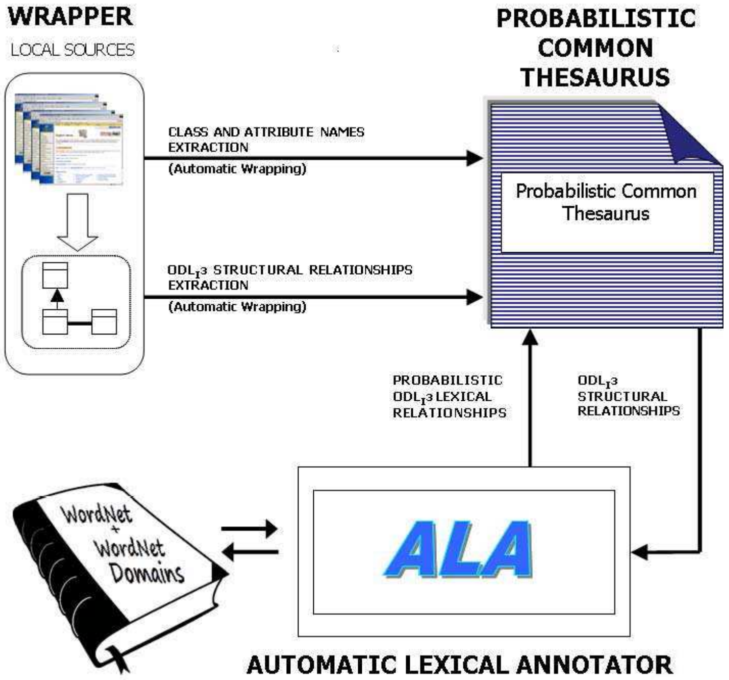

WRAPPER
LOCAL SOURCES
PROBABILISTIC COMMON THESAURUS
CLASS AND ATTRIBUTE NAMES EXTRACTION
(Automatic Wrapping)
Probabilistic Common Thesaurus
ODL$_{I^3}$ STRUCTURAL RELATIONSHIPS EXTRACTION
(Automatic Wrapping)
PROBABILISTIC ODL$_{I^3}$ LEXICAL RELATIONSHIPS
ODL$_{I^3}$ STRUCTURAL RELATIONSHIPS
WordNet + WordNet Domains
ALA
AUTOMATIC LEXICAL ANNOTATOR

Figure 2.6: Building the PCT

In contrast with most previous word sense disambiguation tools (SenseLearner[3], GWSD[4], SSI[5]), ALA includes the following innovative features:

- it improves the annotation process, exploiting both structural and lexical knowledge of a set of data sources (the structural knowledge we exploit is derived from the intensional knowledge of each source, thus our tool differs from SSI and GWSD that make use of the structural knowledge extracted from thesauri);
- a term is associated to a set of meanings which are not necessarily orthogonal or mutually exclusive;
- it suggests a rank of meanings (WordNet synsets) for each term, according to their probabilities;
- it supports an annotation GUI addressed to skilled and inexpert user.

The tool is integrated in the MOMIS system [8, 15] (but might be coupled with any data integration system).In particular, the tool uses special-

[3] http://lit.csci.unt.edu/ senselearner/
[4] http://www.cse.unt.edu/ rada/downloads.html#gwsd
[5] http://lcl.uniroma1.it/ssi/

ized software (wrappers) for logically converting the format of the data source schemata to the internal object language ODL$_{I^3}$, and to build the Probabilistic Common Thesaurus (PCT) which collects the discovered relationships among data sources (see figure 2.6).

The output of ALA is a set of probabilistic lexical ODL$_{I^3}$ annotations, from these annotations a set of probabilistic relationships is calculated. A probabilistic relationship holds between two terms, if it exists a lexical relationships between their meanings in the lexical database WordNet. The probability assigned to the lexical relationships depends on the probability value of the meaning under consideration for a term.

ALA permits to automatically annotate the schemata of a given set of data sources and to discover lexical relationships among schema elements. Thank to the formula 2.2, we can calculate the probability value associated to an ODL$_{I^3}$ relationship holding among two synsets. These probabilistic lexical relationships are collected in the PCT that already contains the structural ODL$_{I^3}$ relationships.

The annotation panel of ALA enables the user to choose a set of WSD algorithms, to select how to execute the different WSD algorithms (configuring the reliability of each algorithm), and to collect the outputs of the algorithms. Moreover, the tool supports an expert user in finding the best combination of WSD algorithms: with the formula panel a user can combine algorithms and operator as he wishes, using the GUI or directly writing the formula (according to the syntactic rules of the operators).

ALA is independent of the algorithms and the operators implemented, and allows the programmer to add new algorithms or operators without recompile the source code of ALA. At the moment in ALA we can combine five different WSD algorithms (the same algorithms we have combined in PWSD): SD, WND, WordNet first sense heuristic, Gloss similarity [11], Iterative gloss similarity [11].

The user can choose all or a subset of these algorithms and combine the annotation results using different operators:

- **Pipe operator**: it combines the annotation outputs of different WSD algorithms provided in a given order (the default order is established by the algorithm reliability value or may be chosen by the user). The default reliability of each algorithm derives from the precision of the algorithm evaluated on a benchmark. The pipe operator uses the output of the first algorithm and for the terms where the first algorithm does not supply an annotation executes the second algorithm and so on. Each term can be disambiguates at most by one WSD algorithm.

- **Parallel operator**: it combines the annotation results from different

WSD algorithms on the basis of the Dempster-Shafer theory as PWSD (see section 2.3). Using the Dempster's rule of combination we determine with which probability each source element can be associated to its meanings [92, 82]. The theory deals with the so-called *frame of discernment*, in our case, the set of all possible meanings for the term under consideration, and its power set, which is the set of all the possible subsets of the set of possible meanings. We derive the belief mass function from the output and the reliability of the WSD algorithms. To combine the output of the WSD algorithms we use the Dempster's rule of combination [92, 82]. With the paralles operator each term is disambiguated with the contribution of all the selected WSD algorithms.

- **Threshold operator**: it filters out the annotations under a given threshold, this operator can be applied on the annotation output of a combination of WSD algorithms.

2.4.1 ALA - an example of use

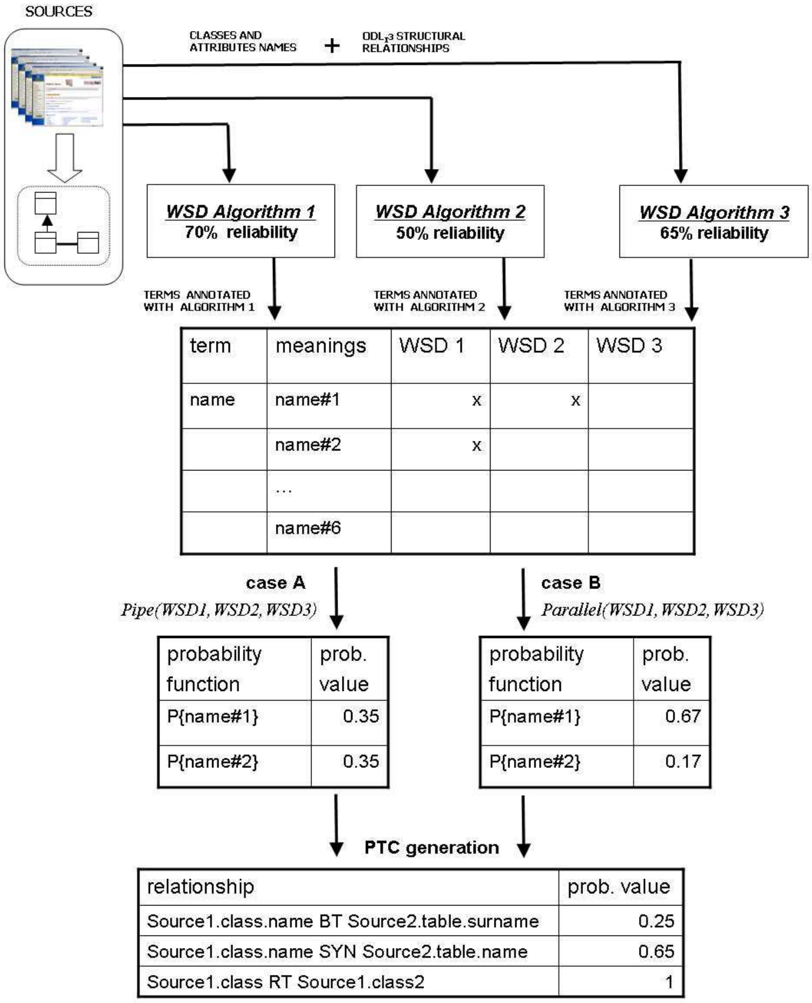
As a case in point, let us consider the term "name". In WordNet we find six different meanings for "name" ($name_{\#1}$, $name_{\#2}$, .., $name_{\#6}$). Suppose we have to combine three WSD algorithms that give different outputs (like the case show in figure 2.7): WSD1 that chooses the set of meanings $\{name_{\#1}$, $name_{\#2}\}$, WSD2 that provides $name_{\#1}$ as the correct meaning and WSD3 that does not give any result. What we want to obtain is a rate of probability to be assigned to each possible meaning of the term under consideration.

As shown in figure 2.7 the sources terms are automatically extracted and annotated by the application of the WSD algorithms.

In case A, the pipe operator is applied following a manual order (WSD1, WSD2, WSD3). For the term "name" both the meanings $name_{\#1}$ and $name_{\#2}$ are selected. The probability associated to the annotation $name_{\#1}$ and $name_{\#2}$ is equal to the reliability of the WSD1 algorithm split on each meaning.

The parallel operator, case B, does not consider the algorithms that do not supply any annotation for a term. So, in this case, the evaluation is executed only combining the probability mass function of WSD1 and WSD2. Let us point out that the parallel operator increases the probability associated to the meaning $name_{\#1}$ because both the WSD algorithms select this meaning.

After the annotation phase, the probabilistic lexical relationships are extracted. The lexical probabilistic relationships are inserted in the PCT (the first and the second relationships in figure 2.7) and are added together to the structural relationships (the third relationship in figure 2.7).


SOURCES
CLASSES AND ATTRIBUTES NAMES
+
ODL$_{I}$3 STRUCTURAL RELATIONSHIPS
WSD Algorithm 1
70% reliability
WSD Algorithm 2
50% reliability
WSD Algorithm 3
65% reliability
TERMS ANNOTATED WITH ALGORITHM 1
TERMS ANNOTATED WITH ALGORITHM 2
TERMS ANNOTATED WITH ALGORITHM 3
case A
Pipe(WSD1, WSD2, WSD3)
case B
Parallel(WSD1, WSD2, WSD3)
PTC generation

term	meanings	WSD 1	WSD 2	WSD 3
name	name#1	x	x	
	name#2	x		
	...			
	name#6			

probability function	prob. value
P{name#1}	0.35
P{name#2}	0.35

probability function	prob. value
P{name#1}	0.67
P{name#2}	0.17

relationship	prob. value
Source1.class.name BT Source2.table.surname	0.25
Source1.class.name SYN Source2.table.name	0.65
Source1.class RT Source1.class2	1

Figure 2.7: Automatic annotation process with ALA: an example

2.4.2 Handling annotations

Developing ALA, we encountered the problem of how manage the annotations that refer to a source element. We thought to keep the semantically rich representation of the lexical annotation by means of the use of *annotation groups*. Each annotation group has a probability value and is composed of a set of annotations. Each annotation is a reference to a WordNet synset and include a probability value.

The WSD algorithms can choose more meanings to annotate a term; to manage the lexical annotations derived from different WSD it is helpful to consider the annotations supplied by each method in a group of annotation

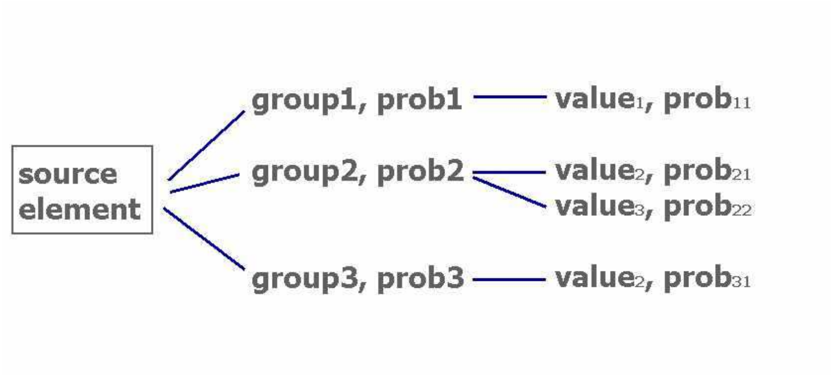


Figure 2.8: Automatic annotation process: an example of annotation groups

that is characterized by the probability of the WSD algorithm.

The annotation groups are the easiest way to represent the lexical annotations supply by ALA.

Let us see an example, in figure 2.8 a source element is associated to three annotation groups. Each group is characterized by a probability assigned to the group and a list of annotations, where each annotation has a value (a synset identifier or a triple composed by lemma, syntactic category and sense number) that can identify a meaning, and a probability assigned to this value. Note that, the value can be repeated in different groups, this is because each group defines a specific lexical annotation.

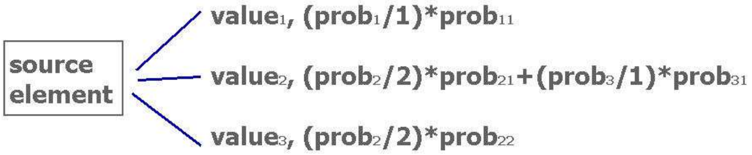


Figure 2.9: Automatic annotation process: an example of “flattered” annotation groups

While the WSD algorithms and ALA deal with annotation groups, the user is not able to have a overall view of the *annotation groups*. In fact, although the number of possible synsets associated to a source element is limited (we can not have more than all the possible synsets for each lemma of which the source element is composed), there is no limitation for the number of groups a source element can have. For this reason, we implemented a procedure to "flatter" the annotations.

This procedure aims at directly associate annotations to a source element (as it was before the development of ALA in the MOMIS system). Each annotation will be then enriched of a probability value that is derived from the "votes" that annotation has received in the annotation groups.

Formula 2.3 ***Probability value associated to an annotation after the "flatter" operation***

$$prob(value_i) = \sum_{j \in Groups} P_{group_j}(value_i) * \frac{P_{group_j}}{\|group_j\|}$$

Export and Import of Annotations Once we have executed the annotation of a data source (the annotation can be partial), the annotation can be save in an xml file. The annotations may be exported and then reused in other applications or shared by different users. In the file, the lexical annotations are saved as annotation groups (without the "flatter" operation).

Importing annotations from a file permits the annotator to exploit them for a next run.

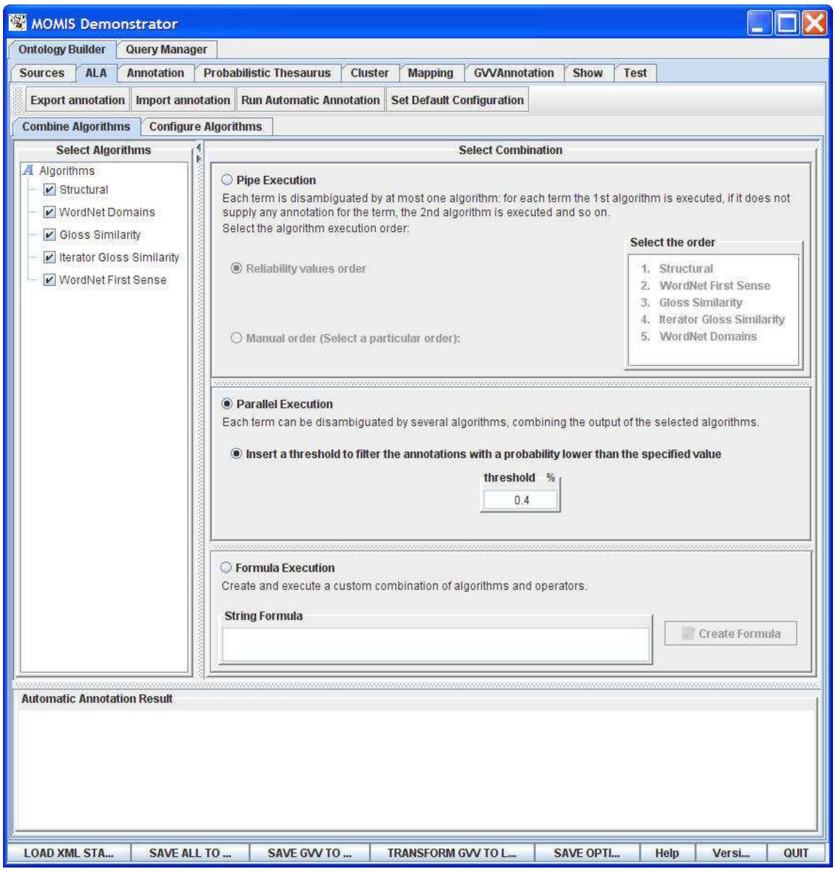
2.4.3 ALA Panel

As shown in figure 2.10, ALA permits to combine different WSD algorithms (the ones shown on the left side). The user may select the algorithms on the left side of the panel and choose one of the execution modalities:

- Pipe (Default): the inexpert user does not set any parameter. In the default execution modality, the WSD algorithms are executed in pipe following the reliability order.

- Parallel: the user may select the WSD algorithms to be applied. A parallel execution can be performed with or without a threshold filtering.

- Formula: the skilled user may express complex combination of WSD algorithms directly writing the formula or through the GUI, defining a "tree" of combinations (see figure 2.11 where the formula panel is shown).



Algorithms
☑ Structural
☑ WordNet Domains
☑ Gloss Similarity
☑ Iterator Gloss Similarity
☑ WordNet First Sense

○ Pipe Execution

Each term is disambiguated by at most one algorithm: for each term the 1st algorithm is executed, if it does not supply any annotation for the term, the 2nd algorithm is executed and so on.
Select the algorithm execution order:

● Reliability values order

○ Manual order (Select a particular order):

Select the order
1. Structural
2. WordNet First Sense
3. Gloss Similarity
4. Iterator Gloss Similarity
5. WordNet Domains

● Parallel Execution

Each term can be disambiguated by several algorithms, combining the output of the selected algorithms.

● Insert a threshold to filter the annotations with a probability lower than the specified value

threshold % 0.4

○ Formula Execution

Create and execute a custom combination of algorithms and operators.

String Formula

Figure 2.10: Screenshot of the ALA panel

2.5 Related work

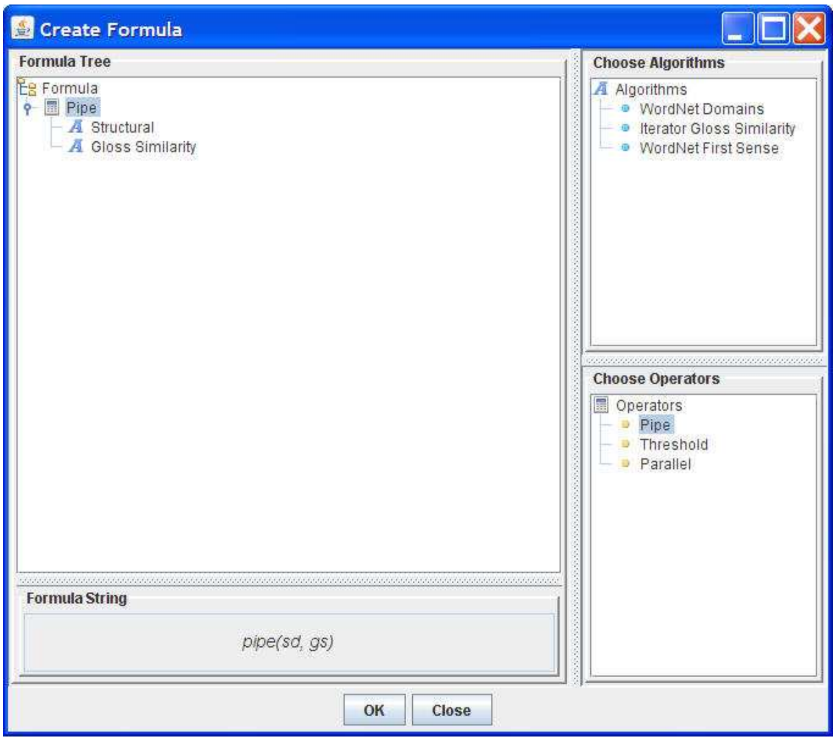
2.5.1 Techniques for extending WordNet

Different extension of WordNet are proposed by many researchers.

Some researches aims at producing a formal specification of WordNet as an axiomatic theory. Among them, the OntoWordNet project [48] derives an ontology from WordNet. WordNet synset taxonomies and relations are reorganized and enriched, by extracting, interpreting and axiomatizing semantic relations implicitly encoded.

An other approach tries to classify WordNet synsets [76] using a method to create set of semantically similar WordNet synset and organizing them in categories. Such categories, as the context in MELIS, are then used for assigning the correct meaning to elements.

The Ontoling system, that jointly works with the Protégé editing tool,



Formula String

pipe(sd, gs)

Figure 2.11: Screenshot of the formula panel of ALA

has been proposed [83]. The system goals are very similar to our purposes, as it provides a graphical interface for browsing linguistic resources (thesauri, dictionaries, wordnets...), linguistically enriching ontologies with elements from these linguistic resources, building new ontologies, starting from existing linguistic resources. Ontoling does not permit to add new relationships into the lexical database whereas we allow the user to extend the WordNet relationships.

2.5.2 Probabilistic mapping

In the recent years there has been an increasing interest in the research on database system that handle uncertainty, in particular in the area of data integration system. Our method draws inspiration from [42], where is defined the concept of probabilistic schema mapping, together with an algorithm for answering queries in the presence of approximate schema mappings. This paper starts from a initial probabilistic schema mappings, and without dealing with the generation of probabilistic mappings, proposes a probabilistic query answering method. Our goal is the generation of a set of probabilistic (and ordinary) relationships that represent the first step in calculating of a set of probabilistic mappings that represent the input of the probabilistic querying answering proposed in [42].

In literature many tools for automatic ontology mapping are offered, but only a few use a probabilistic approach.

Some authors, dealing with ontology matching, have proposed a method to resolve semantic ambiguity in order to filtering the appropriate mappings between different ontologies [51]. The paper shows the application of two similarity measures: one based on the ontological context of the terms, and another based on WordNet. Using the semantic similarity measures, the mappings found by an ontology matching tool can be filtered, so the precision of the system improves. The limit of this method is that it does not disambiguate the label of the ontology classes, but only evaluates the possible meanings.

In [30] a method for discovering schema mappings, based on the lexical relationships extracted from WordNet, is proposed. However, considering all the synsets associated to a term by WordNet, this approach does not realize any sort of disambiguation. The main disadvantage is the inclusion of wrong synsets and therefore the extraction of lexical relationships that can define erroneous mappings. For these reason, we propose a probabilistic WSD algorithm that ensures to find more accurate relationships.

Our PWSD is based on a combined WSD approach. Combination methods are an effective way of improving the WSD process performance. The idea of combining the results of different WSD methods was used in most approaches to WSD in literature. In [23], is presented an evaluation study on different combination of different WSD algorithms, and i is showed that combinations system outperform the behavior of the single algorithms of which it is composed. In [77], is described a combined WSD method, based on various sources of knowledge, that combines two WSD methods: a knowledge-based method and a corpus-based method. Most accurate combination approaches have been studied previously in [45]. These WSD systems (called supervised WSD system), depend on the availability of training data, i.e. corpus occurrences of ambiguous words marked up with labels indicating the appropriate sense given the context. However, the acquisition of sufficient labeled data is very expensive and limits the use in new domains and languages [22].

All these approaches, like most of the traditional WSD algorithms, are applied on textual data sources, and performed an exact annotation, assigning to a term only one meaning. An innovative aspect of the PWSD method is the possibility to associate more than one synset to each term. The exact annotation, in fact, is simple but suffers from some limitations, in particular when we deal with disambiguation of structured or semi-structured data sources. On structured and semi-structured data source there is not as a wide context as in a text source; in addition, there are less words that concur to the definition of a concept (i.e. only classes and attributes). On these kind of data sources, it is difficult also for a domain expert to select only one sense as the correct one for each element of the source (an element in a scheme, encloses a wider meaning than term in a sentence). For these reason, we propose a new approximate annotation that, differently from the traditional approaches, permitted to trait the intrinsic uncertainty deriving from an automatic WSD method.

Chapter 3

Test cases

We experimented MELIS, CWSD and PWSD in conjunction with the MOMIS system for improving the lexical annotation phase.Moreover, with PWSD, we concentrated to discover new probabilistic relationships among terms.

We evaluated the algorithms in different scenarios. The automatic annotation results have been statistically evaluated w.r.t. the golden standard, in terms of precision and recall. The golden standard for the benchmark is the annotation selected by an expert user. The expert may select more than one meaning for each term and the evaluation we have done is based on the each possible meaning selected for a term.

This chapter is organized in a first section that shows some examples of the application of the WSD algorithms to a test case, then in section 3.2 and 3.3 we show some experimental results on real data sets.

3.1 Running examples

3.1.1 MELIS

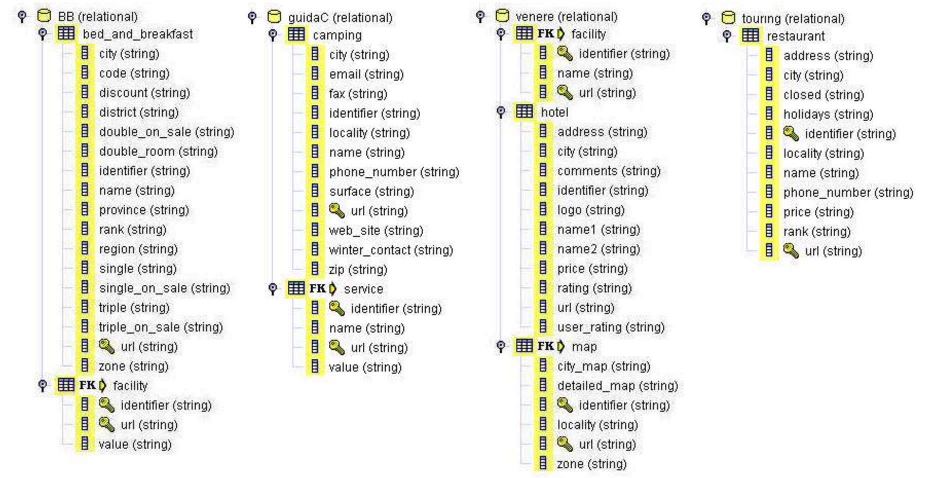
We tested MELIS coupled with MOMIS by building an ontology for a set of data-intensive web sites containing data related to the tourism domain (see figure 3.1). The web sites were wrapped, and the corresponding data were structured and stored into four relational databases. The main classes extracted from the four sources are: `hotel` (from the "venere" database[1]), `restaurant` (from the "touring" database[2]), `camping` (from the "guidaC" database[3]) and `bed and breakfast` (the "BB" database[4]).

[1] `http://www.venere.com`.

[2] `http://www.touringclub.com`.

[3] `http://www.guidacampeggi.com`.

[4] `http://www.bbitalia.it`.



```
BB (relational)
  bed_and_breakfast
    city (string)
    code (string)
    discount (string)
    district (string)
    double_on_sale (string)
    double_room (string)
    identifier (string)
    name (string)
    province (string)
    rank (string)
    region (string)
    single (string)
    single_on_sale (string)
    triple (string)
    triple_on_sale (string)
    url (string)
    zone (string)
  FK facility
    identifier (string)
    url (string)
    value (string)

guidaC (relational)
  camping
    city (string)
    email (string)
    fax (string)
    identifier (string)
    locality (string)
    name (string)
    phone_number (string)
    surface (string)
    url (string)
    web_site (string)
    winter_contact (string)
    zip (string)
  FK service
    identifier (string)
    name (string)
    url (string)
    value (string)

venere (relational)
  FK facility
    identifier (string)
    name (string)
    url (string)
  hotel
    address (string)
    city (string)
    comments (string)
    identifier (string)
    logo (string)
    name1 (string)
    name2 (string)
    price (string)
    rating (string)
    url (string)
    user_rating (string)
  FK map
    city_map (string)
    detailed_map (string)
    identifier (string)
    locality (string)
    url (string)
    zone (string)

touring (relational)
  restaurant
    address (string)
    city (string)
    closed (string)
    holidays (string)
    identifier (string)
    locality (string)
    name (string)
    phone_number (string)
    price (string)
    rank (string)
    url (string)
```

Figure 3.1: Data sources of the tourism domain

The incremental annotation process starts with the partial annotation of the data sources (notice that, in principle, this step can be skipped, which means that the entire work is delegated to automatic annotation); for some source elements, the ontology designer selects one or more corresponding WordNet meanings. Figure 3.2 shows some WordNet meanings and the lexical relationships among some data sources elements. In particular:

- $hotel_{\#1}$ and $restaurant_{\#1}$ are siblings, i.e. they have a common direct hypernym;
- $hotel_{\#1}$, $house_{\#3}$, $restaurant_{\#1}$ are direct hyponyms of $building_{\#1}$ (though we observe that this relationship may appear a bit misleading: typically restaurants are viewed as buildings, but rather as places where a service is provided);
- $bed\_and\_breakfast_{\#1}$ is an hyponym of $building_{\#1}$;
- the closest hypernym that $campsite_{\#1}$ and $building_{\#1}$ share is $physical\_object_{\#1}$, a top level synset in WordNet. This relationship does not allow the system to find lexical connections between the class "camping" and the other classes. Consequently, by means of the MELIS component *WNEditor*, a direct relationships between $campsite_{\#1}$ and the hierarchy of $building_{\#1}$ is introduced.

Notice that the choice of annotations can be tricky, even for simple structures as the ones we selected. For example, if we annotated the source element

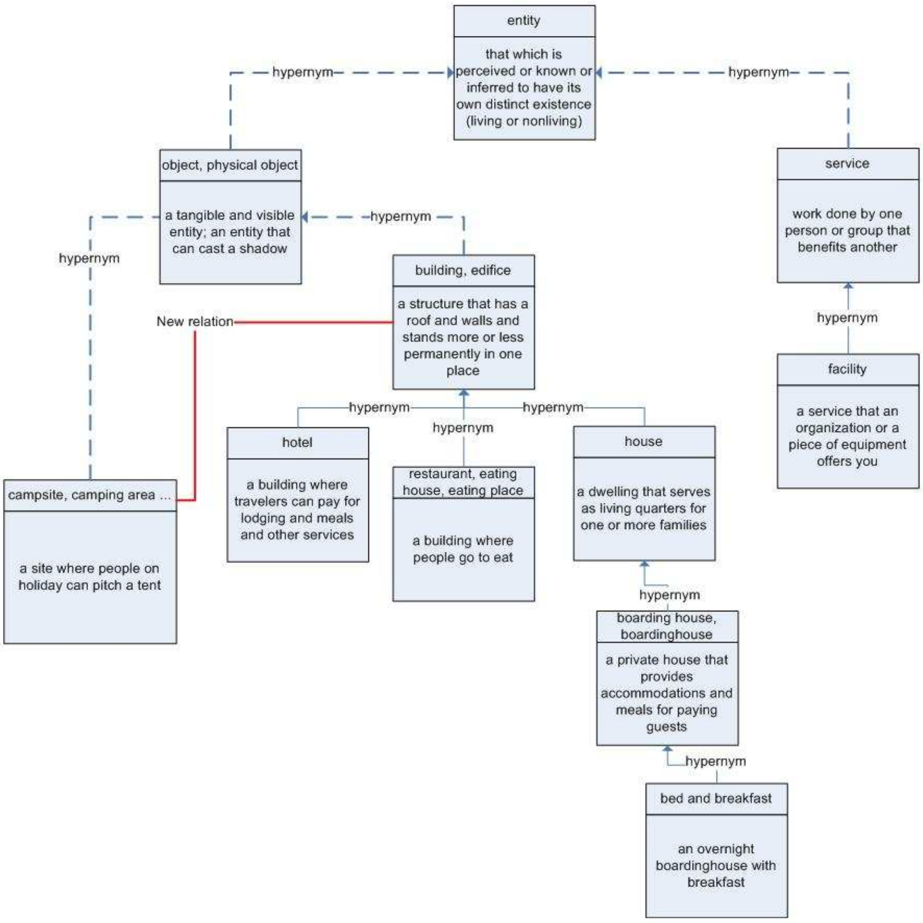

entity
that which is perceived or known or inferred to have its own distinct existence (living or nonliving)
hypernym
hypernym
object, physical object
a tangible and visible entity; an entity that can cast a shadow
service
work done by one person or group that benefits another
hypernym
hypernym
building, edifice
a structure that has a roof and walls and stands more or less permanently in one place
New relation
hypernym
facility
a service that an organization or a piece of equipment offers you
hypernym
hypernym
hypernym
hotel
a building where travelers can pay for lodging and meals and other services
restaurant, eating house, eating place
a building where people go to eat
house
a dwelling that serves as living quarters for one or more families
campsite, camping area ...
a site where people on holiday can pitch a tent
hypernym
boarding house, boardinghouse
a private house that provides accommodations and meals for paying guests
hypernym
bed and breakfast
an overnight boardinghouse with breakfast

Figure 3.2: Lexical relationships among data sources elements

"camping" with the only WordNet meaning associated to the word "camping" (i.e. $camping_{\#1}$), we get a wrong meaning (in this context), as its gloss is "the act of encamping and living in tents in a camp"; whereas the intuitively correct synset in our context is $campsite_{\#1}$, defined as "the site where people can pitch a tent". Finally, to test a larger number of implemented heuristic rules, $hotel_{\#1}$ has been annotated as its hypernym: "building" through *WNEditor*.

The first annotated schema is then passed to MELIS both as an input and as a domain ontology. The tool starts the meaning elicitation process (see section 2.1.2 for datails) and produces a set of inferred lexical annotations of the schema elements. Figure 3.3 illustrates the results of a sample test of incremental annotation on one of our schemas. It shows the annotations manually provided by the ontology designer, a fraction of the new annota-

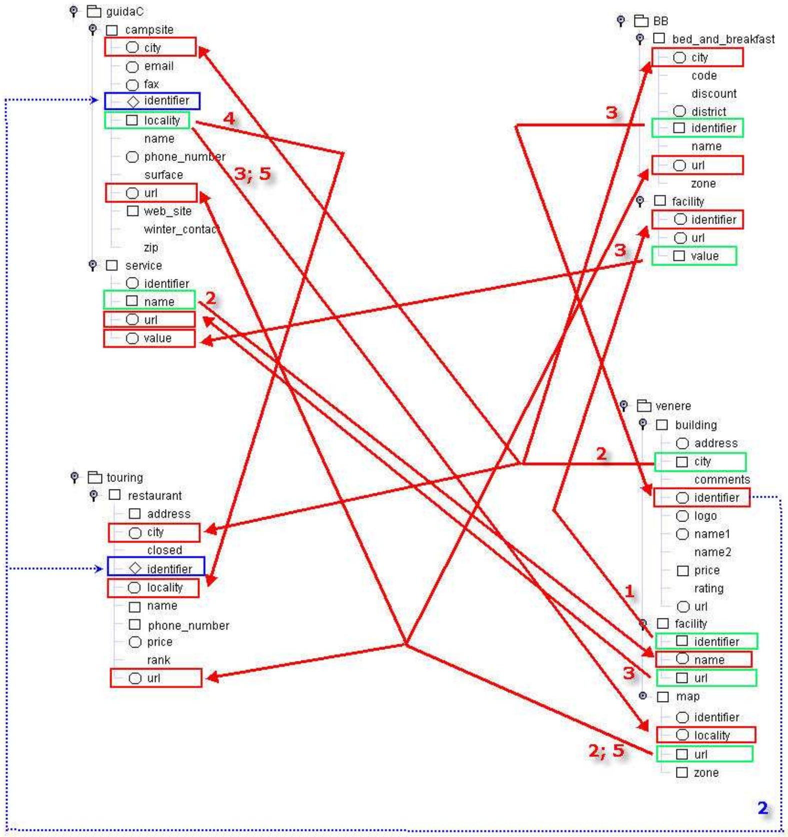


Figure 3.3: Annotations generated by MELIS

tions generated after a first run of MELIS, and the additional annotations generated after a second run, when the outcome of the first run was provided as additional background knowledge in input; the numbers on the arrows refer to the heuristic rule which was used to generate the annotation. Notationally, a square near a class/attribute means that the element was manually annotated, a circle means that the element was automatically annotated after the first run, and a rhombus that it was incrementally annotated after the second run.

For illustration purposes, for every heuristic rule, we explain one of the generated annotations.

- Rule 1: the attribute “identifier” of the class “facility” in the source “VENERE” is annotated as $identifier_{\#1}$ of the class “facility” in the source “BB”, since both the classes are annotated with the same synset.
- Rule 2: because of the hyponym relationships generated by the an-

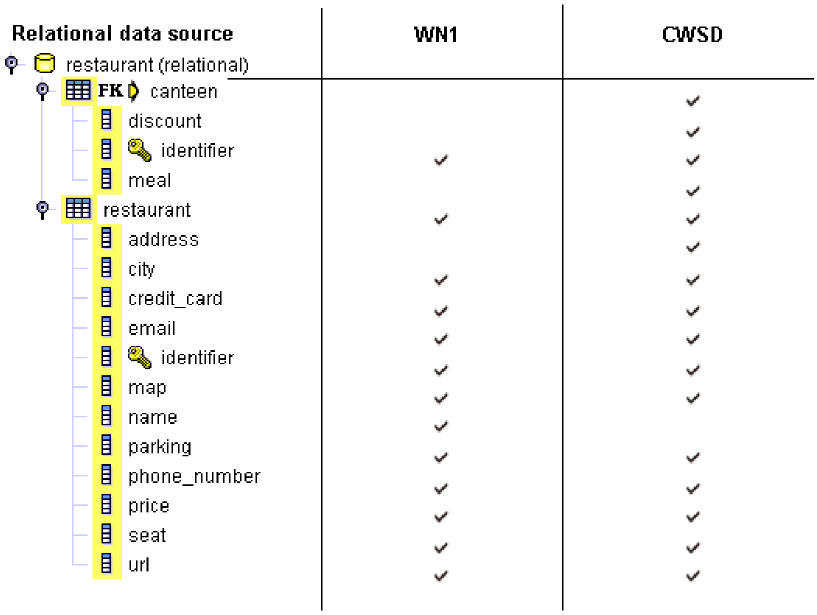
notations of the classes "hotel", "campsite", "bed and breakfast" and "building", the attribute "city" of the class "building" in the source "VENERE" produces the annotation of the same attribute in the sources "BB", "touring" and "guidaC".

- Rule 3: because of the hypernym relation generated by the annotations of "building" and "bed and breakfast", the attribute "identifier" of the class "bed and breakfast" in the source "BB" generates the annotation of the same attribute in the source "VENERE". By executing a second run of the MELIS process, the attribute "identifier" on the class "building" generates the annotation of the same attribute on the classes "campsite" and "restaurant" of the sources "guidaC" and "touring" (application of Rule 2).
- Rule 4: because of the new relationship introduced in WordNet, $campsite_{\#1}$ is a sibling of $restaurant_{\#1}$. Consequently, the attribute "locality" is annotated in the same way in the sources "guidaC" and "touring".
- Rule 5: in the relational database there are foreign keys that represent a connection between two classes. In the source "VENERE", the class "map" has a foreign key: the attribute "url" that refers to the class "hotel". As this relationship joins with hierarchical relationships $hotel_{\#1}$, $campsite_{\#1}$, $bed\_and\_breakfast_{\#1}$ and $building_{\#1}$, the attribute "url" of the class "map" in the source "VENERE" generates the annotation of the same attribute in the classes "campsite", "bed and breakfast" and "restaurant" of the other sources.

The heuristic rules 6 and 7 are not exploited in our test. In fact, to fire these rules, we would need hierarchical structures.

3.1.2 Two examples of the application of the CWSD method

Figure 3.4 shows a simple example of the application of the CWSD algorithms. We chose a relational source composed of two different tables (canteen, restaurant) connected by a structural relationship (foreign key). In figure we evaluated the right senses supplied by different disambiguation approaches. The WordNet first sense heuristic (labeled as WN1 in the figure) was already used in MOMIS and chooses the more frequent WordNet sense (the first one) as the correct meaning for a term. CWSD overcomes the WordNet first sense heuristic as it is able to disambiguate more terms. In particular, as shown in Figure 3.5, first SD exploits the structural relationship

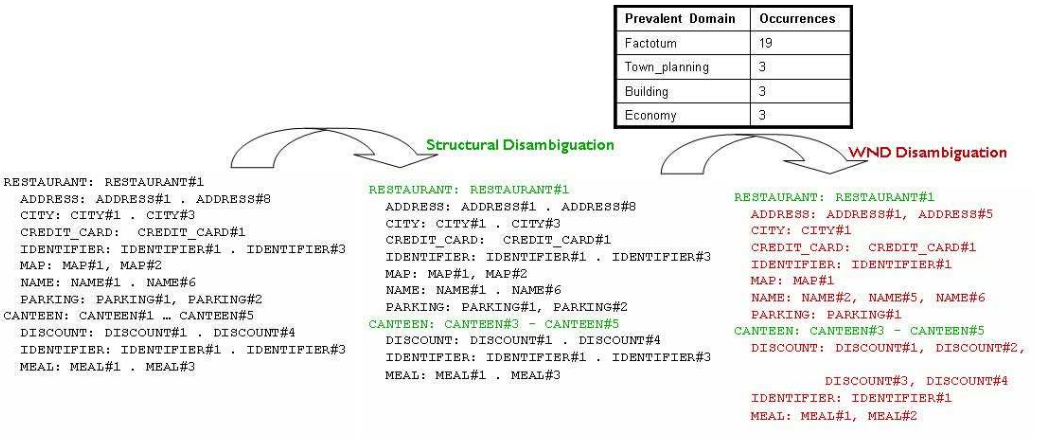


Relational data source	WN1	CWSD
restaurant (relational)		
FK canteen		✓
discount		✓
identifier	✓	✓
meal		✓
restaurant	✓	✓
address		✓
city	✓	✓
credit_card	✓	✓
email	✓	✓
identifier	✓	✓
map	✓	✓
name	✓	
parking	✓	✓
phone_number	✓	✓
price	✓	✓
seat	✓	✓
url	✓	✓

Figure 3.4: Evaluation of CWSD on a relational data source

”foreign key” to assign the correct meaning to the terms: `restaurant` and `canteen`; then WND works on terms ignored by SD, calculates the prevalent domains over the entire set of terms and compares these domains with the ones associated to each term to determine the correct meaning. The unique term annotated in a wrong way is `name` because for the annotation of that term it is chosen a synset associated to the “factotum” domain, while the correct sense is associated to “linguistics” domain (that is not present in the *prevalent domains*).

Another example, is the evaluation of CWSD on a hierarchical data source (see figure 3.7). The hierarchical data source is a portion of the “society” subtree in the Google directory. All the ISA relationships in the schemata are inserted in the Common Thesaurus (CT) as NT relationships. In Figure 3.7 is shown a comparison between the annotation results achieved by CWSD and the ones obtained by WordNet first sense heuristic on the hierarchical data source. If we disambiguate by using the WordNet first sense heuristic, we obtain only one sense for each term. Despite these results are fairly good, they are not complete. With the CWSD algorithm we improved the results in two directions:(1) the disambiguation of the terms is more accurate; in fact, we are able to assign to term more than one meaning;(2) moreover, as shown in figure 3.6, the application of CWSD enriches the CT of new relationships. These relationships will be very useful for the integration task. When we



Prevalent Domain	Occurrences
Factotum	19
Town_planning	3
Building	3
Economy	3

```
RESTAURANT: RESTAURANT#1
  ADDRESS: ADDRESS#1 . ADDRESS#8
  CITY: CITY#1 . CITY#3
  CREDIT_CARD:  CREDIT_CARD#1
  IDENTIFIER: IDENTIFIER#1 . IDENTIFIER#3
  MAP: MAP#1, MAP#2
  NAME: NAME#1 . NAME#6
  PARKING: PARKING#1, PARKING#2
CANTEEN: CANTEEN#1 … CANTEEN#5
  DISCOUNT: DISCOUNT#1 . DISCOUNT#4
  IDENTIFIER: IDENTIFIER#1 . IDENTIFIER#3
  MEAL: MEAL#1 . MEAL#3
```

```
RESTAURANT: RESTAURANT#1
  ADDRESS: ADDRESS#1 . ADDRESS#8
  CITY: CITY#1 . CITY#3
  CREDIT_CARD:  CREDIT_CARD#1
  IDENTIFIER: IDENTIFIER#1 . IDENTIFIER#3
  MAP: MAP#1, MAP#2
  NAME: NAME#1 . NAME#6
  PARKING: PARKING#1, PARKING#2
CANTEEN: CANTEEN#3 - CANTEEN#5
  DISCOUNT: DISCOUNT#1 . DISCOUNT#4
  IDENTIFIER: IDENTIFIER#1 . IDENTIFIER#3
  MEAL: MEAL#1 . MEAL#3
```

```
RESTAURANT: RESTAURANT#1
  ADDRESS: ADDRESS#1, ADDRESS#5
  CITY: CITY#1
  CREDIT_CARD:  CREDIT_CARD#1
  IDENTIFIER: IDENTIFIER#1
  MAP: MAP#1
  NAME: NAME#2, NAME#5, NAME#6
  PARKING: PARKING#1
CANTEEN: CANTEEN#3 - CANTEEN#5
  DISCOUNT: DISCOUNT#1, DISCOUNT#2,

                DISCOUNT#3, DISCOUNT#4
  IDENTIFIER: IDENTIFIER#1
  MEAL: MEAL#1, MEAL#2
```

Figure 3.5: Annotation results obtained by CWSD applied on a relational data source

apply SD, all the ISA relationships in the schemata are extracted from the source and inserted in the CT as NT_{EXT}/BT_{EXT} relationships. Then, SD searches hyponymy/hypernymy relationships, if exists, among terms related by NT_{EXT}/BT_{EXT} relationships. As you can see in the WordNet cloud, we find relationships between the synsets of Religion ($Religion_{\#1}$, $Religion_{\#2}$) and the ones of its child nodes ($Christianity_{\#1}$, $Buddhism_{\#2}$, $Taoism_{\#2}$, $Taoism_{\#1}$). Thus, we choose these synsets as the correct ones for the terms.

As we can observe in figure 3.7, the unique term annotated in a wrong way is `Society`, this is caused by the WND algorithm. WND chooses a synset associated to the "factotum" domain, while the correct sense is associated to "anthropology" domain (that is not present in the *prevalent domains*).

3.2 Test case 1: Google and Yahoo directories

In this scenario, we selected the first three levels of a subtree of the Yahoo and Google directories ("society and culture" and "society", respectively), which amounts to 327 categories for Yahoo and 408 for Google, arranged in two different subtrees.

Lexical annotation results have been evaluated in terms of recall (the number of correct annotations made by the algorithm divided by the total number of annotations, i.e. one for each category, as defined in a golden standard) and precision (the number of correct annotations retrieved divided by the total number of annotations retrieved). The recall and precision values

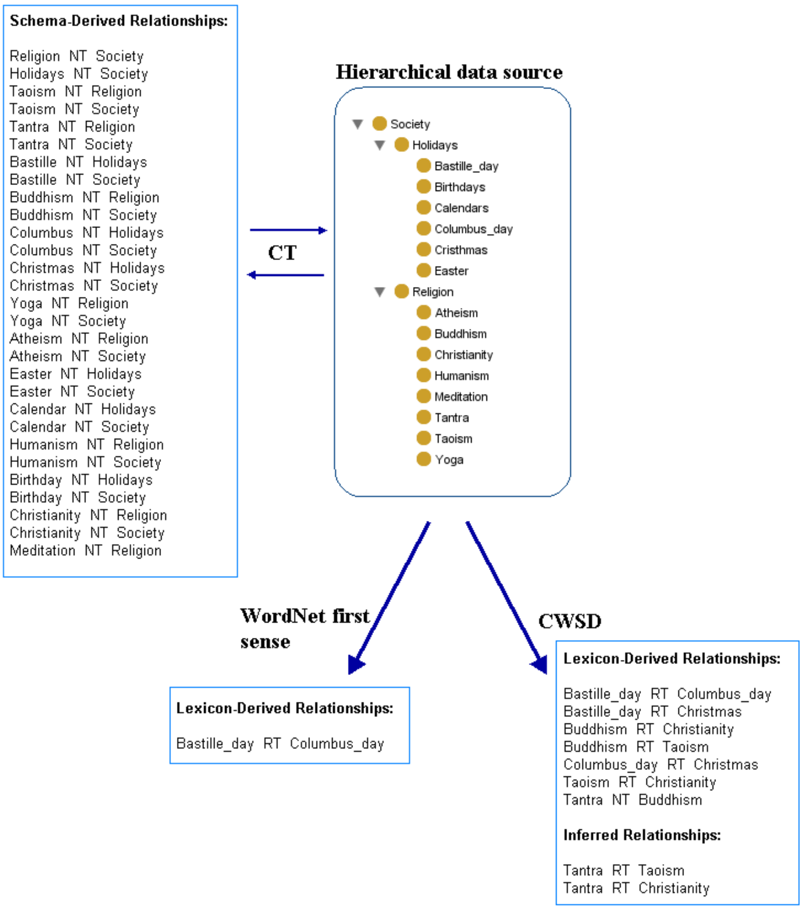
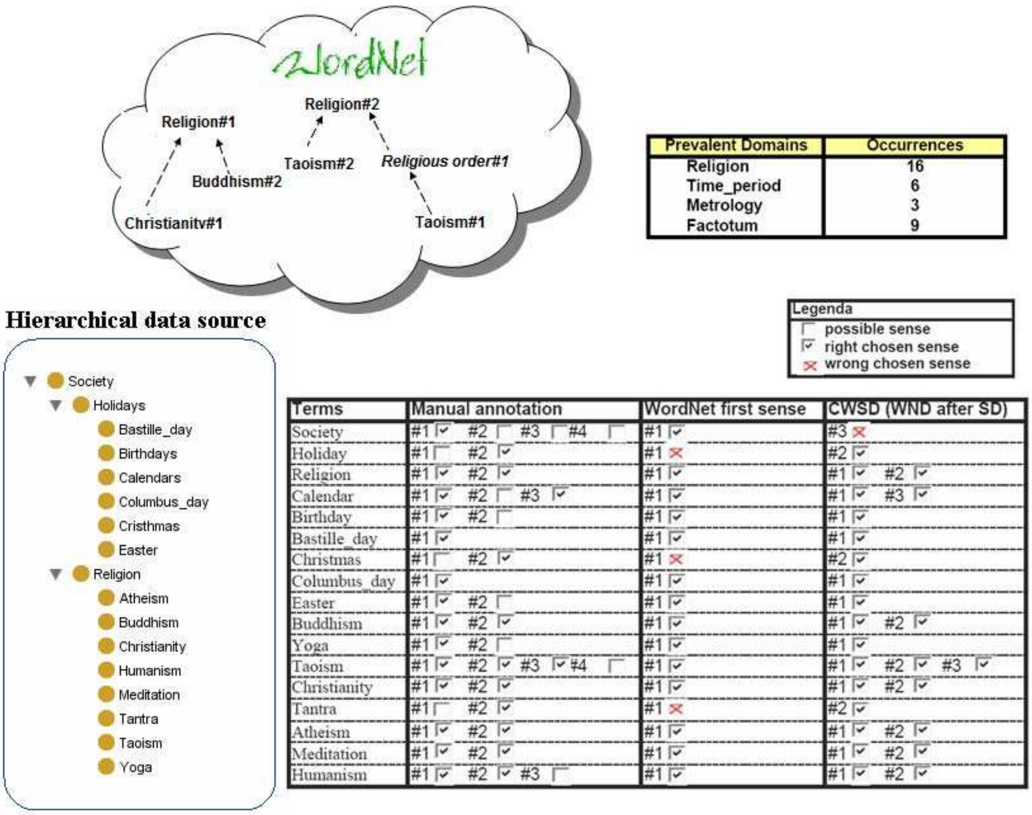


Figure 3.6: Enrichment of the CT with relationships extracted by CWSD on a hierarchical data source

are obtained by considering an element as correctly annotated if the annotation given by the user is included in the set of annotations calculated by the WSD algorithms.

3.2.1 Tuning of CWSD

We experimented the CWSD over Google and Yahoo directories to perform tests on different context. Over these sources we have evaluated the application of WordNet Domains disambiguation algorithm by itself (WND) and the Combined WSD algorithm (Structural+WND) on a global context (3.1), and on a structural context (3.2). In a global context we consider all the terms of a source; in a structural context we consider the terms of the classes that are correlated with an ISA relationship. Of course, these are not the



Prevalent Domains	Occurrences
Religion	16
Time_period	6
Metrology	3
Factotum	9

Legenda
□ possible sense
☑ right chosen sense
☒ wrong chosen sense

Terms	Manual annotation	WordNet first sense	CWSD (WND after SD)
Society	#1 ☑ #2 □ #3 □ #4 □	#1 ☑	#3 ☒
Holiday	#1 □ #2 ☑	#1 ☒	#2 ☑
Religion	#1 ☑ #2 ☑	#1 ☑	#1 ☑ #2 ☑
Calendar	#1 ☑ #2 □ #3 ☑	#1 ☑	#1 ☑ #3 ☑
Birthday	#1 ☑ #2 □	#1 ☑	#1 ☑
Bastille_day	#1 ☑	#1 ☑	#1 ☑
Christmas	#1 □ #2 ☑	#1 ☒	#2 ☑
Columbus_day	#1 ☑	#1 ☑	#1 ☑
Easter	#1 ☑ #2 □	#1 ☑	#1 ☑
Buddhism	#1 ☑ #2 ☑	#1 ☑	#1 ☑ #2 ☑
Yoga	#1 ☑ #2 □	#1 ☑	#1 ☑
Taoism	#1 ☑ #2 ☑ #3 ☑ #4 □	#1 ☑	#1 ☑ #2 ☑ #3 ☑
Christianity	#1 ☑ #2 ☑	#1 ☑	#1 ☑ #2 ☑
Tantra	#1 □ #2 ☑	#1 ☒	#2 ☑
Atheism	#1 ☑ #2 ☑	#1 ☑	#1 ☑ #2 ☑
Meditation	#1 ☑ #2 ☑	#1 ☑	#1 ☑ #2 ☑
Humanism	#1 ☑ #2 ☑ #3 □	#1 ☑	#1 ☑ #2 ☑

Figure 3.7: Application of the CWSD algorithm on a hierarchical data source

only possible contexts on which we can apply the algorithms, but we have focus our evaluation on these two contexts because they show very different environments on which we could study the behaviour of our algorithms.

The annotation process starts without any partial annotation of the data sources (notice that, this means that the entire work is delegated to automatic annotation). The schemas are passed to WND (see section 2.2.2 for details about the algorithm), and then, selecting an appropriate number of domains, we obtained the disambiguation of the source terms. Up to this point of our work, the only user involvement is in choosing the number of domains.

As shown in table 3.1 and 3.2 the precision and recall increase in the combined approach. Indeed, the application of SD (see section 2.2.1 for details about the algorithm) over the web directories exploits the 792 ISA relationships and allows to obtain 60 annotations of which 58 are correct annotations.

The only SD deduces an high precision(97%) but a very low recall (8%). For our experience this is caused by the incompleteness of the semantic Word-

	WND		Structural+WND	
number of domains	**Recall**	**Precision**	**Recall**	**Precision**
1 + factotum	58.56%	71.43%	66.12%	73.84%
2 + factotum	57.05%	68.53%	64.61%	71.15%
3 + factotum	56.30%	67.22%	63.85%	69.93%
4 + factotum	57.81%	67.11%	65.37%	69.76%
5 + factotum	58.69%	66.10%	66.25%	68.76%
6 + factotum	58.44%	65.63%	65.99%	68.32%
7 + factotum	60.83%	66.62%	68.39%	69.17%
8 + factotum	60.58%	66.25%	68.14%	68.83%
9 + factotum	61.96%	65.08%	69.52%	69.52%
10 + factotum	61.08%	64.15%	68.64%	68.64%
11 + factotum	61.46%	64.30%	69.02%	69.02%
12 + factotum	59.95%	62.71%	67.51%	67.51%

Table 3.1: Disambiguation on the global context of google and yahoo directories

Net relationships. The algorithm disambiguates a term with at most two synsets. We have checked that in the case of polysemy the chosen synsets are both correct.

We can compare these results to the ones we obtain with the MELIS approach. In a different way from MELIS, CWSD approach does not need initial annotations to disambiguate the source terms. Consequently, here we only compare CWSD tests with the ones in MELIS that start with no annotations at all. In that case CWSD outperforms MELIS in terms of precision and recall.

The experimental results show how our CWSD permit to obtain good results, moreover, structural knowledge of structured and semi-structured sources is shown to significantly improve the disambiguation results obtained by applying only WND algorithm.

After this experimentation, we investigated the role of the context choice in our CWSD and determine a criteria to choose the best number of domains during the configuration of the WND algorithm.

3.2.2 Evaluation of MELIS, SD, WND and CWSD

A MELIS evaluation was done in the context of web directories. In particular, we selected the first three levels of a subtree of the Yahoo and Google directories ("society and culture" and "society", respectively), which amounts

	WND		Structural+WND	
number of domains	**Recall**	**Precision**	**Recall**	**Precision**
1 + factotum	65.74%	74.47%	73.30%	76.48%
2 + factotum	65.62%	72.46%	73.17%	74.58%
3 + factotum	65.74%	71.12%	73.30%	73.30%
4 + factotum	65.87%	70.30%	73.43%	73.43%
5 + factotum	65.87%	69.83%	73.43%	73.43%
6 + factotum	66.62%	69.97%	74.18%	74.18%
7 + factotum	66.62%	69.33%	74.18%	74.18%

Table 3.2: Disambiguation on the ISA context of google and yahoo directories

to 327 categories for Yahoo and 408 for Google, arranged in two different subtrees.

MELIS results have been evaluated in terms of recall (the number of correct annotations made by MELIS divided by the total number of annotations, i.e. one for each category, as defined in a golden standard) and precision (the number of correct annotations retrieved divided by the total number of annotations retrieved). Moreover, since the algorithm is incremental, the evaluation is done after a first run and after a number of runs until a fixed point has been reached.

The process exploits knowledge provided by the initial annotation of the sources to generate the remaining annotations. Consequently, the initial set of annotations may highly affect the result. For this reason we considered eight different starting points:

1. **No Annotation**: the two subtrees are given to MELIS with no annotations at all.

2. **Y1-G0**: only the first level of the Yahoo subtree has been manually annotated.

3. **Y(1&2)-G0**: the Yahoo subtree have been manually annotated.

4. **Y0-G1**: only the first level of the Google subtree has been manually annotated.

5. **Y0-G(1&2)**: the Google subtree have been manually annotated.

6. **Y1-G1**: the first level of the Yahoo and Google subtrees has been manually annotated.

7. **Y1-G0: WN enriched**: only the first level of the Yahoo hierarchy has been annotated. The annotator extended WordNet with 6 new terms and synsets to properly represent the subtree elements.

8. **Y(1&2)-G0: WN enriched**: the first two levels of the Yahoo hierarchy has been annotated. The annotator extended WordNet with 18 new terms and synsets to properly represent the subtree elements.

The MELIS method is supervised, i.e. the user may check the results calculated after each run and eventually correct the imprecise annotations Such operation surely improves the result quality, but it is dependent on the user knowledge about the source domains and WordNet. Table 3.3 shows the evaluation results when MELIS is executed without any user intervention. As a consequence, the results we present are a kind of "worst case" scenario for MELIS.

	1st run		**Fix point**	
	Recall	**Precision**	**Recall**	**Precision**
1. No Annotation	22.90%	82.01%	24.08%	79.51%
2. Y1-G0	24.82%	85.28%	26.88%	85.85%
3. Y(1&2)-G0	74.45%	98.82%	76.96%	98.49%
4. Y0-G1	23.78%	78.54%	25.85%	77.78%
5. Y0-G(1&2)	73.41%	98.81%	74.89%	98.45%
6. Y1-G1	29.69%	85.17%	29.69%	85.17%
7. Y1-G0:WN enriched	26.29%	85.99%	28.36%	86.49%
8. Y(1&2)-G0:WN enriched	78.88%	98.89%	81.39%	98.57%

Table 3.3: MELIS evaluation

As described in section 2.1, MELIS associates a set of senses to each element and then on the basis of some rules, one or more appropriate senses are selected. Consequently, in Table 3.3 only the annotations perfectly fitting with the reference provided by the user are evaluated as correct annotations (when MELIS returns more than one annotation – possibly including the correct one – for a label, such result is taken as wrong). By analyzing Table 3.3, we observe that:

- the automatic execution of MELIS without any supervision generates at the fix point results in terms of precision not always better than the ones obtained after the first run. This is because some incorrect annotations, generated after the first run, propagate wrong knowledge in the following runs;

- the results are highly dependent on the input annotations (see for example case 3 and the symmetric case 5) and on the lexical database reference: by using a lexical database where specific terms and relationships are included improves the results (see the measures of cases 7 and 8).

In order to evaluate the results in case of a user assisted process, next table shows the recall and precision values obtained by considering an element as properly annotated if the annotation given by the user is included in the set of annotations calculated by MELIS.

	1st run		**Fix point**	
	Recall	**Precision**	**Recall**	**Precision**
1. No Annotation	50.22%	60.39%	53.03%	58.85%
2. Y1-G0	50.52%	61.73%	55.83%	62.38%
3. Y(1&2)-G0	88.48%	92.30%	93.80%	90.84%
4. Y0-G1	53.32%	64.35%	56.72%	63.79%
5. Y0-G(1&2)	79.76%	94.08%	82.72%	92.72%
6. Y1-G1	61.15%	66.45%	61.15%	66.45%
7. Y1-G0:WN enriched	52.29%	62.54%	57.61%	63.11%
8. Y(1&2)-G0:WN enriched	92.91%	92.64%	98.23%	91.22%

Table 3.4: MELIS evaluation - second test

By analyzing Table 3.4, we observe that:

- the results are very interesting and they allow us to hypothesize that a supervised MELIS use may provide very valuable support to the annotation task;

- the results allow us to show another way of using our tool: MELIS may suggest to the user a set of candidate annotations and among them it may indicate the most promising one. The user then can confirm such annotation.

Our experience shows that the annotation process supported by MELIS is less time-consuming and, in general, it converges to the final result after three runs.

In table 3.5, we compare the disambiguation of the subtree of the Google and Yahoo directories obtained with different algorithms: only SD, only WND, CWSD and MELIS.

We compared CWSD results with the ones in MELIS configured with no annotations at start.

WSD approach	Recall	Precision
SD	8.00%	97.00%
WND	66.62%	69.97%
CWSD	74.18%	74.18%
MELIS	53.03%	58.85%

Table 3.5: Comparing different WSD algorithms on the Google and Yahoo directories

3.3 Test case 2: OAEI benchmark

This scenario makes use of ontologies from the benchmark 2008 of the OAEI project[5] to evaluate the automatic annotation. For sake of simplicity, we considered only three ontologies, but the process is scalable and applicable to a large set of data sources. The domain of the test is bibliographic references. In particular we have selected the complete ontology (`onto101`) and an ontology where labels are replaced by synonyms (`onto205`), and another bibliographic ontology (`onto209`).

The golden standard for the benchmark is the annotation selected by an expert user. The expert may select more than one meaning for each term and the evaluation we have done is based on the each possible meaning selected for a term.

3.3.1 Evaluation Measures

The automatic annotation results have been statistically evaluated w.r.t. the golden standard. All of the statistics are calculated based on the `contingency table`, which looks like this:

	Reference=Y	Reference=N
Assigned=Y	a	b
Assigned=N	c	d

We calculate the following statistics:

- **accuracy** measures the portion of all decisions that were correct decisions. It is defined as $(a + d)/(a + b + c + d)$.

- **error** measures the portion of all decisions that were incorrect decisions. It is defined as $(b + c)/(a + b + c + d)$.

[5] http://oaei.ontologymatching.org/2008/

- **precision** measures the portion of the assigned categories that were correct. It is defined as $a/(a+b)$.
- **recall** measures the portion of the correct categories that were assigned. It is defined as $a/(a+c)$.
- **F1** measures an even combination of precision and recall. It is defined as $2a/(2a+b+c)$.

All these measures fall in the range from 0 to 1, with 1 being the best score, except of error measure, where 0 is the best score.

3.3.2 Evaluation of CWSD and PWSD

We compared CWSD and PWSD over the OAEI benchmark.

	Accuracy	**Error**	**Precision**	**Recall**	**Fmeasure**
CWSD	0.78%	0.22%	0.66%	0.55%	0.60%
PWSD	0.75%	0.25%	0.56%	0.76%	0.65%
PWSD Threshold=0.2	0.84%	0.16%	0.80%	0.70%	0.75%
WN1	0.83%	0.17%	0.81%	0.53%	0.64%

Table 3.6: PWSD comparison with respect to other WSD methods

We compared the results of PWSD with the WordNet first sense heuristic (labeled as WN1 in table 3.6), and with CWSD.

The WordNet first sense heuristic, is often used as baseline for WSD systems, and often outperforms many of these system which take surrounding context into account [73].

As table 3.6 shows, the precision and recall of PWSD does not increase with respect to the CWSD approach, this is due to a high number on annotations with a very low probability value.

Filtering the annotation results with a threshold refine the annotation results of PWSD. The threshold chosen is quite low (the average probability value of PWSD was 0.34), this permits to filter out only the annotations that are not supported by a lot of WSD algorithms (the annotations that can introduce noise) without decrease the recall.

As shown in figure 3.6, the perfomence of PWSD used with a threshold are really good. in this case the precision is high, (83, and quite similar to the WN1 heuristic, but hte value of recall is really promising, These evaluation is the only one that shown how many meaning can be lost with a WSD algorithm that allow only one meaning for each term (as WN1 is).

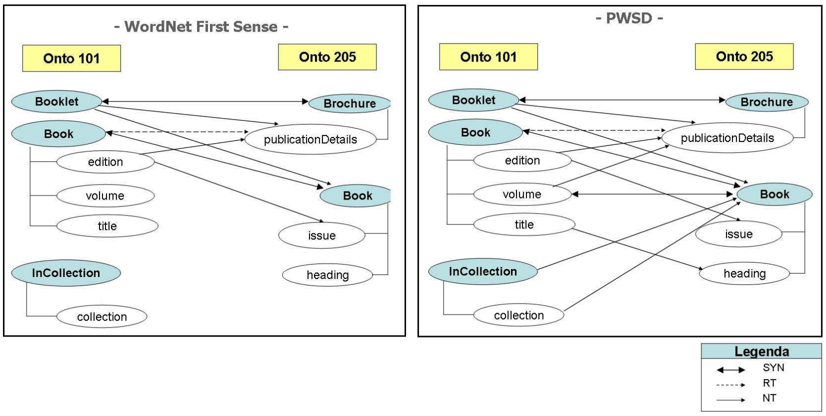


Figure 3.8: An example of the lexical relationships obtained by WordNet first sense heuristic and by PWSD

3.3.3 Discovering relationships

With PWSD we can derived more lexical relationships among terms than with other WSD algorithms. An example is showed in figure 3.8 where a subset of the OAEI benchmark has been selected. While the property `volume` of the class `book` is annotated with a wrong meaning with the use of WordNet first sense heuristic, is instead correctly annotated with the support of PWSD. Therefore only with PWSD we can discover the right relationships among `volume`, `book` and `publicationDetails`.

Other examples of lexical relationships extracted by PWSD are the ones between `Collection` and `book`, and `title` and `heading`.

The example in figure 3.8 is only a small portion of the selected ontologies, but it shows how the amount of lexical relationships extracted after the use of a complete and correct disambiguation method increase.

Chapter 4

Automatic lexical annotation applied to an Ontology Matcher

A new generation of semantic applications are emerging in the area of Ontology Matching, focused on exploiting the increasing amount of online semantic data available on the Web. These applications handle the high semantic heterogeneity introduced by the increasing number of available online ontologies (different domains, different points of view, different conceptualisations). These matching algorithms exhibit very good performance, but they rely on merely syntactical techniques to anchor the terms to be matched to those found on the Semantic Web. As a result, their precision can be affected by ambiguous terms. A critical issue is to solve these ambiguity problems by introducing techniques from Word Sense Disambiguation, which validate the mappings by exploring the semantics of the terms involved in the matching process.

In these context, I explored a way to apply the WSD algorithms (shown in chapter 2) to the SCARLET matcher[1]. This evaluation has been done in collaboration with the Knowledge Media Institute (KMi)[2] where SCARLET has been developed.

The WSD algorithms have been evaluated on the results supply by the SCARLET matcher in order to improved the precision of the matcher. SCARLET is a technique for discovering relations between two concepts by making use of online available ontologies. The matcher can discover semantic relations by reusing knowledge declared within a single ontology or by combining knowledge contained in several ontologies. To discover a relation holding between two concepts, is necessary to find similar concepts in an online on-

[1] http://scarlet.open.ac.uk/

[2] http://kmi.open.ac.uk/

tology. My study has been focused on the lexical annotation of the concepts involved in the matching process, in order to identify which concepts has similar meaning and which not. The evaluation has been done on two different test cases. Moreover, in the second scenario, it has been compared with other WSD techniques introduced in SCARLET [51].

This chapter is organized as follow: Section 4.1 describes the SCARLET matcher, in section 4.2 are shown two different evaluation scenarios, in the final section some conclusion are sketched.

4.1 SCARLET matcher

SCARLET [88, 90] is a technique for discovering relations between two concepts by making use of online available ontologies. Developed in the context of the NeOn[3] and OpenKnowledge[4] projects, SCARLET has been primarily used to support tasks such as ontology matching and enrichment. SCARLET is available online[5], moreover, the SCARLET API is available for free download.

SCARLET discovers semantic relations between concepts by using the entire Semantic Web as a source of background knowledge: it automatically identifies and explores multiple and heterogeneous online ontologies to derive relations. Its strategy consists in using semantic search engines such as Swoogle [36] and WATSON [89] to find ontologies containing concepts with the same names as the candidate concepts and to derive mappings from their relationship in the selected ontologies.

The hypothesis is that ontology mapping, while trying to cope with the heterogeneity of the Semantic Web, could actually exploit it. In other words, online available ontologies could provide the background knowledge sources, which are needed to support ontology mapping. On the one hand, they can be selected dynamically, thus circumventing the need for an a priori, manual ontology selection. On the other hand, by relying on semantic sources, SCARLET avoids the inherent noise caused by information extraction based methods.

Two basic strategies are devised (see Figure 4.1):

- First, SCARLET can discover semantic relations by reusing knowledge declared within a single ontology;

[3] http://www.neon-project.org/web-content/

[4] http://www.openk.org/

[5] http://scarlet.open.ac.uk/

- Second, the matcher can also combine knowledge contained in several ontologies, thus discovering relations across ontologies.

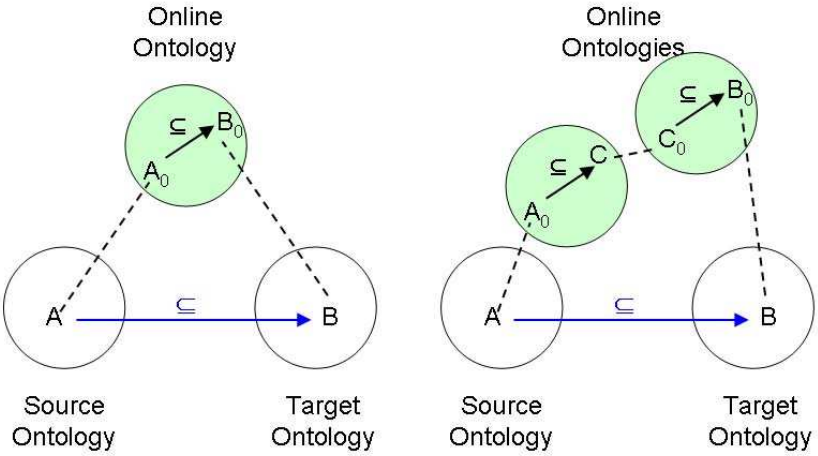


Figure 4.1: SCARLET strategies to infer relationships among concepts by harvesting the Web.

Figure 4.1 illustrates the basic idea of Ontology Matching by harvesting the Semantic Web. A and B are the concepts to relate, and the first step is to find online ontologies containing concepts A_0 and B_0 equivalent to A and B. This process is called *anchoring* and A_0 and B_0 are called the *anchor terms* (or *anchor concepts*). Based on the relations that link A_0 and B_0 in the retrieved ontologies, a mapping is then derived between A and B.

This strategy assumes that a semantic relation between the candidate concepts can be discovered in a single ontology. However, some relations could be distributed over several ontologies. Therefore, if no ontology is found that relates both candidate concepts, then the mappings should be derived from two (or more) ontologies. In this case, mapping is a recursive task where two concepts can be mapped because the concepts they relate in some ontologies are themselves mapped. In the draw on the right in Figure 4.1, A and B are equivalent to A_0 and B_0 that belong to different online ontologies. Reasoning on the online ontologies permits to find a path from A_0 to B_0, the discovered relationship on this path is inferred among the concepts A and B.

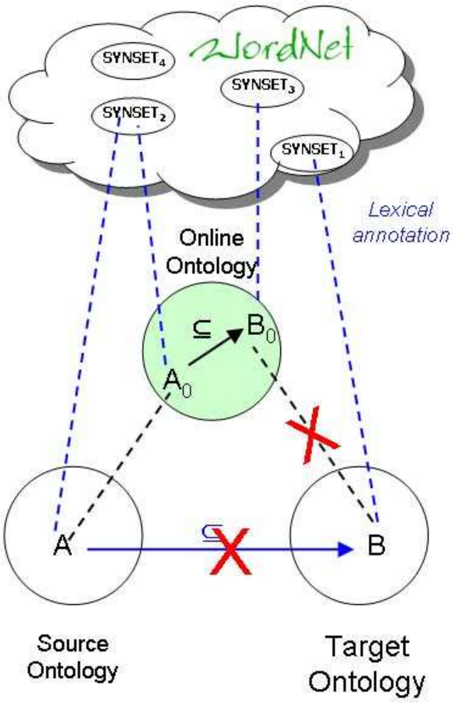


Figure 4.2: Lexical annotation of concepts involved in the anchoring, elimination of anchoring and discharging of SCARLET matches.

4.2 Evaluation

My evaluation has been focused on the lexical annotation of the anchoring terms in order to filter out the anchoring between concepts with different meanings and discharge the discovered relation. The idea was to apply the a combination of WSD algorithms to the source ontologies and the background ontologies involved in the matching process. Once we obtain a lexical annotation of these ontologies, we examined the concepts involved in the anchoring. If a concept and its anchoring concept have disregarding meanings (i.e. if they do not have the same list of meanings), the anchoring is discharge. The evaluations shown that lexical annotation can filter out wrong anchoring (with a good precision) and so, it can improved the efficiency of the matcher. Figure 4.2 shows how the lexical annotation influences the anchoring. After the annotation of all the concepts involved in the anchoring (A, B, A_0, B_0), it is possible to compare the meanings of a concept with the meanings of its anchoring concept. In the figure, the anchoring between A and A_0 is preserved because the concepts have the same meanings. Instead, the anchoring between B and B_0 is discharged because the concepts have different meanings. As a consequence, the mapping among A and B is no longer valid.

Below are shown two evaluation scenarios. In the first one we only examined the wrong anchoring and evaluated if these mappings can be discharge with the use of WSD techniques. In the second scenario, instead, we tested the WSD algorithms over both correct and incorrect anchoring to evaluate not only which wrong mappings are discharge after lexical annotation, but even, which good mappings are lost due to lexical annotation.

4.2.1 NALT and AGROVOC wrong mappings evaluation

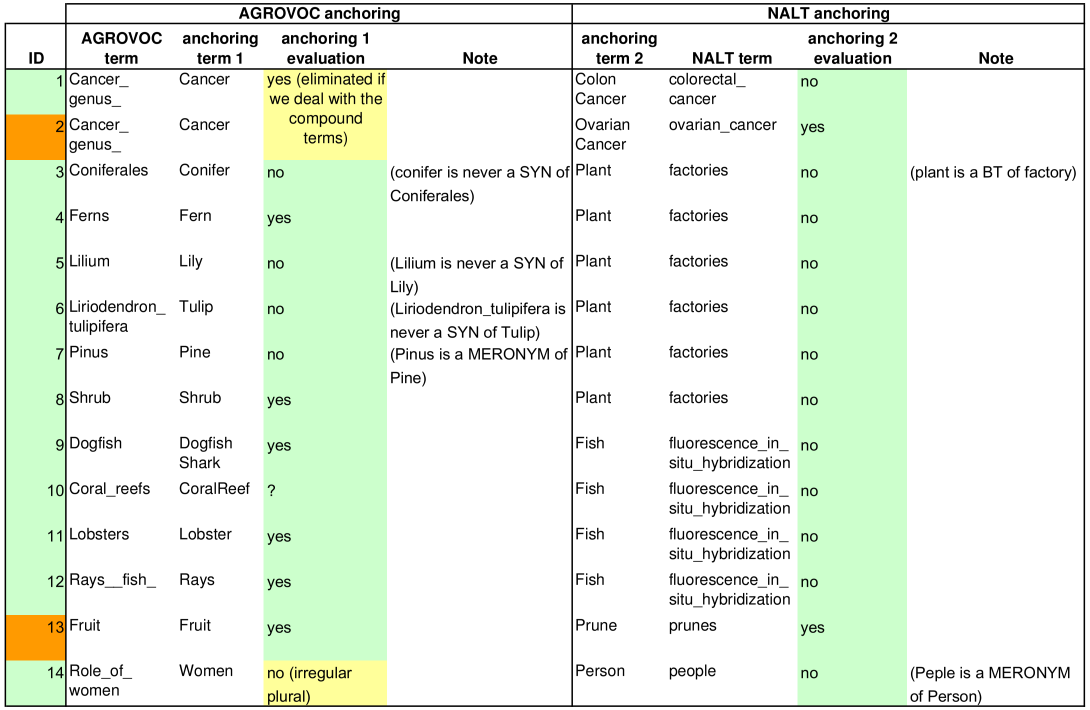
A baseline implementation of SCARLET has been evaluated using two very large, real life thesauri[6] [90] . The United Nations Food and Agriculture Organization (FAO)s **AGROVOC** thesaurus, version May 2006, consists of 28.174 descriptor terms (i.e., preferred terms) and 10.028 non-descriptor terms (i.e., alternative terms). The United States National Agricultural Library (NAL) Agricultural thesaurus **NALT**, version 2006, consists of 41.577 descriptor terms and 24.525 non-descriptor terms. Both alternative and preferred terms are used in the SCARLET experiment.

Based on results shown in [90], it is concluded that online ontologies are useful to solve real life matching tasks. Indeed, if combined appropriately, they can provide a large amount of mappings between the matched ontologies.

While the use of online ontologies generally leads to correct mappings there are also cases when false mappings are derived. On this scenario, a sample of 1000 mappings obtained by SCARLET has been manually validated, resulting in a promising 70% precision. Moreover, an analysis of the causes of false mappings provided interesting results. A manual inspection of the 217 false mappings on which both groups agreed revealed that 114 (i.e., 53%) are due to SCARLET's simplistic anchoring mechanism (i.e., finding concepts in online ontologies that correspond to the matched concepts), while the 47% of false mappings are causes by subsumption used to model generic relations, subsumption used to model part-whole relations, subsumption used to model roles, inaccurate labelling or different views.

More than half of wrong mappings were due to an incorrect anchoring: because of ambiguities, elements of the source ontology have been anchored to online ontologies using the considered terms with different senses. The employed naive anchoring mechanism is thus clearly insufficient, as it fails to distinguish words having several different senses and so, to handle ambiguity.

[6] This data set was used in the "OAEI06 food Thesaurus Mapping Task", http://www.few.vu.nl/∼wrvhage/oaei2006/



	AGROVOC anchoring				NALT anchoring			
ID	AGROVOC term	anchoring term 1	anchoring 1 evaluation	Note	anchoring term 2	NALT term	anchoring 2 evaluation	Note
1	Cancer_ genus_	Cancer	yes (eliminated if we deal with the compound terms)		Colon Cancer	colorectal_ cancer	no	
2	Cancer_ genus_	Cancer			Ovarian Cancer	ovarian_cancer	yes	
3	Coniferales	Conifer	no	(conifer is never a SYN of Coniferales)	Plant	factories	no	(plant is a BT of factory)
4	Ferns	Fern	yes		Plant	factories	no	
5	Lilium	Lily	no	(Lilium is never a SYN of Lily)	Plant	factories	no	
6	Liriodendron_ tulipifera	Tulip	no	(Liriodendron_tulipifera is never a SYN of Tulip)	Plant	factories	no	
7	Pinus	Pine	no	(Pinus is a MERONYM of Pine)	Plant	factories	no	
8	Shrub	Shrub	yes		Plant	factories	no	
9	Dogfish	Dogfish Shark	yes		Fish	fluorescence_in_ situ_hybridization	no	
10	Coral_reefs	CoralReef	?		Fish	fluorescence_in_ situ_hybridization	no	
11	Lobsters	Lobster	yes		Fish	fluorescence_in_ situ_hybridization	no	
12	Rays__fish_	Rays	yes		Fish	fluorescence_in_ situ_hybridization	no	
13	Fruit	Fruit	yes		Prune	prunes	yes	
14	Role_of_ women	Women	no (irregular plural)		Person	people	no	(Peple is a MERONYM of Person)

Figure 4.3: Lexical annotation on the NALT and AGROVOC scenario.

Discovering a solution to the anchoring problem can potentially halve the total number of errors.

Our hypothesis is that integrating techniques from WSD to complement the anchoring mechanism would lead to an important increase in precision. After the lexical annotation of concepts and anchoring concepts, we can discover if the meanings are consistently linked or not.

To analyze the 114 wrong anchoring on AGROVOC and NALT, these steps have been followed:

1. Using *Protegè*[7] I tried to convert the background ontologies available in DAML format to an OWL format (because MOMIS system can accept only ontologies in OWL or RDF format) .

2. I filtered out the mappings that can not be analyzed (those for which the background ontologies are not available or are not convertible in the OWL format). Only 30 out mappings can be analysed.

3. I created a MOMIS schema for each processable mapping. Because usually the ontologies are quite large, it was difficult to analyze and annotate them, therefore I decided to extract a sub-part of the ontology around the main concept used to define the anchoring. A MOMIS schema (created for a discovered mapping) contains different sources:

    - a sub-part of the NALT ontology around the concept;
    - a sub-part of the AGROVOC ontology around the concept;
    - sub-part of background ontologies used in the anchoring, around the anchoring concepts.

    Unfortunately this phase highlighted that not all the background ontologies are correctly written, so, some of the mappings are discharged from the process because it was impossible to find the anchoring concepts in their background ontologies. At the end, only 14 MOMIS schemata were generated.

4. I applied the WSD algorithms that we have implemented at On MOMIS schemas are applied five WSD algorithms: SD (see section 2.2.1), WND (see section 2.2.2), WordNet first sense heuristic, Gloss similarity [11] and Iterative gloss similarity [11]. A pipe execution based on the default reliability order of the algorithms has been chosen(see section 2.4 for details about the execution modalities), i.e. the algorithms are applied from the one that has shown a better precision in the previous evaluation to the worst one; after the execution of each algorithm only

[7] http://protege.stanford.edu/

the terms that are not yet disambiguated are passed to the next algorithm. Each source, in the MOMIS schemata, has been individually annotated.

5. I evaluated the results of the lexical annotation on the concepts involved in the anchoring. As you can see in figure 4.3, if only one of the anchoring is made void, the mapping is no longer valid. As a result, it is sufficient that the lexical annotation reveal that a concept has a meaning different from its anchoring, so that the anchoring is discharged and the mapping is revealed not valid. Thanks to the lexical annotation of the concepts, 12 out of 14 mappings have been invalidated (looking at figure 4.3, the green ID cells are the anchoring that we are able to discharged).

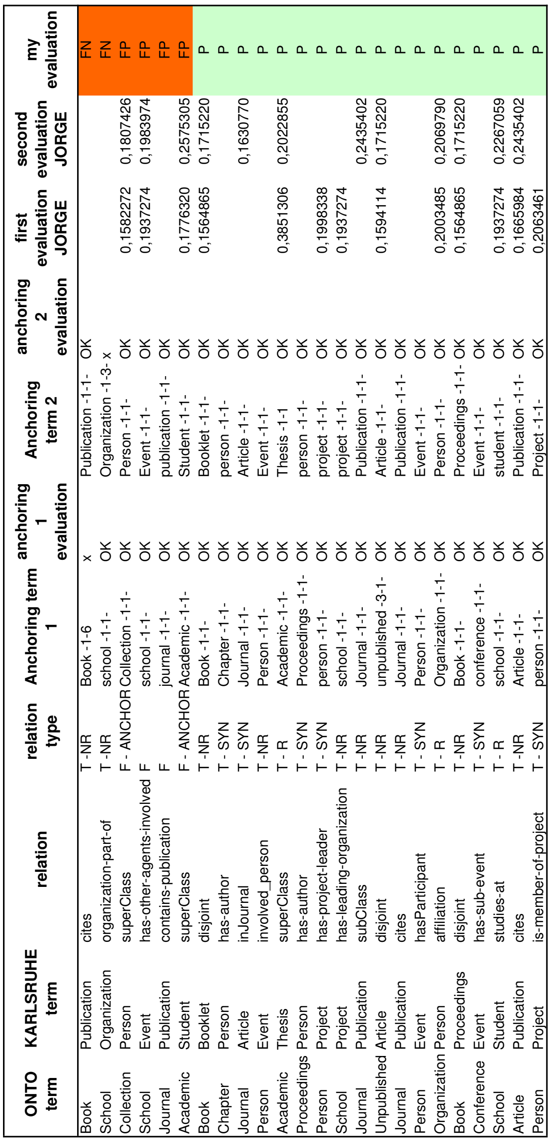
4.2.2 OAEI evaluation

Differently from the case of NALT and AGROVOC ontologies, the source ontologies from OAEI and the background ontologies were not so big, therefore was not necessary to extract sub-ontologies and the lexical annotation has been executed on the entire sources.

For each matching found by SCARLET, we compared the meanings of the terms on the source ontologies with the meaning of the correspondent anchoring terms in the background ontologies. If both of the couples have concord meanings, the anchoring is kept. If just one couple has disregarding meanings, the anchoring is discharged. After the lexical annotation the obtained results have been compared with the manual evaluation done by expert on the entire set of matching.

I evaluated a set of 109 cases. It was impossible to disambiguate 9 terms on the entire set. The evaluation found agreement with the manual evaluation of the anchoring results in 65 cases (62 good anchoring and 3 wrong anchoring). The disagreement with the manual evaluation has been found in 34 cases (in these cases our algorithm retrieved 25 false positive and 9 false positive). These experiments have been conducted to verify the feasibility of the lexical annotation to improve the Semantic Web based Ontology Matching method. The results confirm our initial hypothesis (the precision is increased by solving ambiguity problems) thus proving the value of the approach.

Moreover, I compared my evaluation with the Jorge's evaluation. The Jorge's disambiguation techniques are shown in [96, 52], these techniques have been applied on the SCARLET matcher [51] and they have been eval-



ONTO term	KARLSRUHE term	relation	relation type	Anchoring term 1	anchoring 1 evaluation	Anchoring term 2	anchoring 2 evaluation	first evaluation JORGE	second evaluation JORGE	my evaluation
Book	Publication	cites	T -NR	Book -1-6	x	Publication -1-1-	OK			FN
School	Organization	organization-part-of	T -NR	school -1-1-	OK	Organization -1-3-	x			FN
Collection	Person	superClass	F - ANCHOR	Collection -1-1-	OK	Person -1-1-	OK	0,1582272	0,1807426	FP
School	Event	has-other-agents-involved	F	school -1-1-	OK	Event -1-1-	OK	0,1937274	0,1983974	FP
Journal	Publication	contains-publication	F	journal -1-1-	OK	publication -1-1-	OK			FP
Academic	Student	superClass	F - ANCHOR	Academic -1-1-	OK	Student -1-1-	OK	0,1776320	0,2575305	FP
Book	Booklet	disjoint	T -NR	Book -1-1-	OK	Booklet -1-1-	OK	0,1564865	0,1715220	P
Chapter	Person	has-author	T - SYN	Chapter -1-1-	OK	person -1-1-	OK			P
Journal	Article	inJournal	T - SYN	Journal -1-1-	OK	Article -1-1-	OK		0,1630770	P
Person	Event	involved_person	T -NR	Person -1-1-	OK	Event -1-1-	OK			P
Academic	Thesis	superClass	T - R	Academic -1-1-	OK	Thesis -1-1	OK	0,3851306	0,2022855	P
Proceedings	Person	has-author	T - SYN	Proceedings -1-1-	OK	person -1-1-	OK			P
Person	Project	has-project-leader	T - SYN	person -1-1-	OK	project -1-1-	OK	0,1998338		P
School	Project	has-leading-organization	T -NR	school -1-1-	OK	project -1-1-	OK	0,1937274		P
Journal	Publication	subClass	T -NR	Journal -1-1-	OK	Publication -1-1-	OK		0,2435402	P
Unpublished	Article	disjoint	T -NR	unpublished -3-1-	OK	Article -1-1-	OK	0,1594114	0,1715220	P
Journal	Publication	cites	T -NR	Journal -1-1-	OK	Publication -1-1-	OK			P
Person	Event	hasParticipant	T - SYN	Person -1-1-	OK	Event -1-1-	OK			P
Organization	Person	affiliation	T - R	Organization -1-1-	OK	Person -1-1-	OK	0,2003485	0,2069790	P
Book	Proceedings	disjoint	T -NR	Book -1-1-	OK	Proceedings -1-1-	OK	0,1564865	0,1715220	P
Conference	Event	has-sub-event	T - SYN	conference -1-1-	OK	Event -1-1-	OK			P
School	Student	studies-at	T - R	school -1-1-	OK	student -1-1-	OK	0,1937274	0,2267059	P
Article	Publication	cites	T -NR	Article -1-1-	OK	Publication -1-1-	OK	0,1665984	0,2435402	P
Person	Project	is-member-of-project	T - SYN	person -1-1-	OK	Project -1-1-	OK	0,2063461		P

Figure 4.4: Lexical annotation on the OAEI scenario.

uated on the OAEI test case in [90]. The comparison of the these two disambiguation methods permitted to evaluate some possible threshold on the Jorge's similarity measures.

We report some of the analyzed cases in figure 4.4, my evaluation has been label with FP (false positive), P (positive), FN (false nagative), N (negative). Only for the positive cases it makes sense to calculate a average similarity, that can be useful in the tuning of the Jorge's similarity threshold, because it was evaluated on 36 items, instead of 3 items for the negative cases. The average similarity value on the Jorge's positive cases is 0.184, for the Jorge's first evaluation, and 0.207, for the Jorge's second evaluation.

4.3 Lexical Annotation and Ontology matcher improvements

Analysing the tests and the results I obtained, it was possible to outline different improvements, on one hand, improvements for the lexical annotation and on the other hand, improvements for the SCARLET matcher. The disambiguation algorithms could be improved with an appropriate method to treat compound terms. A lot of cases in the OAEI evaluation and in the AGROVC-NALT evaluation suggested rules to deal with compound terms. For example, the analysis of the super-classes and sub-classes of a compound term could suggest the disambiguation of part of the term. The SCARLET matcher could be improved in case that is possible to annotate the source ontologies before the anchoring phase. In these cases, we can improve the anchoring, looking for synonymous and hyper-classes of the term.

Chapter 5

Conclusions and suggestions for future work

The core problems in data integration are: schema matching, i.e. the identification of correspondences, or mappings, between schema objects, and schema merging, i.e. the creation of a unified schema based on the identified mappings. This thesis demonstrated how lexical annotation could enhance data integration as well as ontology mapping. In fact, annotating data sources allow to uncover "meaning" of data schemata and to perform the discovery of relationships among terms.

Manual annotation is usually time consuming, and it may be unfeasible, especially with large databases. On the other hand, lexical annotation is inherently uncertain because the semantics of schema elements is unsure (even expert user can disagree on the meanings of source elements).

This thesis illustrated several semi-automatic/automatic lexical annotation tools where I took part to the development. Starting from a semi-automatic annotation tool, methods with increased level of automation have been developed. Moreover, the methods evolved from single WSD algorithm to combined approaches. A probabilistic method to perform probabilistic annotation and to derive probabilistic relationships was introduced. In the end, a GUI to support user in finding the best combination of WSD algorithms to lexical annotate data sources has been illustrated.

A set of evaluations of the lexical annotation tools demonstrate that automatic methods can achieve good performance. Moreover, the results gained by the application of these tools on a matcher show how the ontology matching can be positively affected by lexical annotation. The probabilistic method has shown good performance in deriving lexical relationships, as it is able to discover more relationships between schema elements that the traditional approaches (like for example WordNet first sense heuristic).

The more the information to be matched increase the more difficult it becomes to determine only an exact match. In the area of data integration, it become to be crucial the necessity of flexible methods. To accomplish these requests, dynamic data integration systems have been proposed. In this systems, semantic mappings among schemata of different sources have to be discovered *on the fly* with a minimal human intervention.

Uncertainty in data integration is best coped with by using a probabilistic view. Using probabilistic information permits to insert potential matches and to assign to them a probability value. This significantly reduces the cost of integration by allowing it to be fully automated and thus scale to large number of data sources [33].

Further developments could focus on proposing a schema integration approach where uncertainty that is inherent in the schema matching process is explicitly represented. Uncertainty propagates to the schema merging process, and finally it could be depicted in the resulting integrated schema. Widening the use of probability to the discovered relationships it is possible to compute probabilistic mappings. Finding probabilistic mapping is the basis of managing uncertainty in data integration system. And this lead to new methods of query answering like suggesting in [42].

Appendix A

Appendix: heuristic rules description

In this appendix we provide a graphical representation of each rule we introduced in Section 2.1.2, together with an intuitive example of its application.

A.1 Rule 1

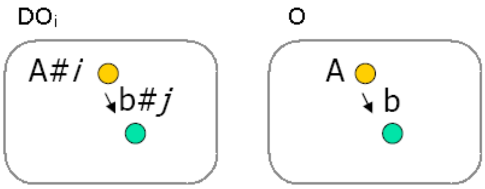


Figure A.1: Graphical representation of rule 1

For example, let us consider a class labeled individual with a datatype property address, and a DO_i where a class person is annotated as $person_{\#1}$ with a datatype property address annotated as $address_{\#2}$. The application of this rule generates the annotation $person_{\#1}$ for the class individual and $address_{\#2}$ for its datatype property address, since individual and person are part of the synonym words associated to $person_{\#1}$.

A.2 Rule 2

For example, let us introduce a class labeled student with a datatype property address, and let DO_i contain a class person annotated as $person_{\#1}$ with a

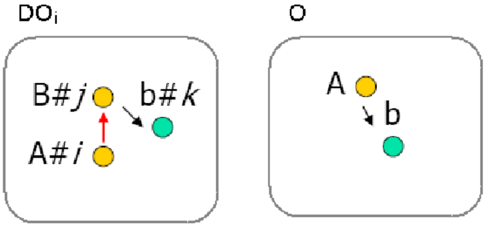


Figure A.2: Graphical representation of rule 2

datatype property address annotated as $address_{\#2}$, and a subclass student annotated as $student_{\#1}$ (notice that the class student is hyponym[1] of enrollee, which is hyponym of person). The application of this rule generates the annotation $student_{\#1}$ for the class labeled student, and $address_{\#2}$ for its datatype property.

A.3 Rule 3

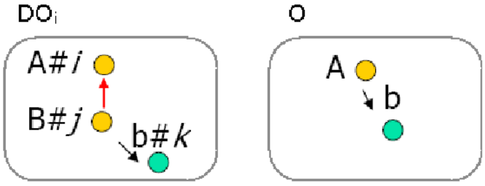


Figure A.3: Graphical representation of rule 3

For example, let us consider a class labeled prof with a datatype property email, and let us suppose that DO_i contains a class professor annotated as $professor_{\#1}$ and a subclass fullprofessor annotated as full$professor_{\#1}$ with a datatype property email annotated as $email_{\#1}$. The application of this rule generates an annotation $professor_{\#1}$ for the class labeled prof, and $email_{\#1}$ for its datatype property.

[1] In the WordNet terminology, we say that a noun X is an *hyponym* of a noun Y if X is less general than Y (X is a specialization of Y); conversely, we say that X is an *hypernym* of Y if X is more general than Y. Other relations across nouns are: X is a *holonym* of Y (Y is a part of X), and X is a *meronym* of Y (X is a part of Y). Different relations are used for other grammatical types, e.g. for verbs and adjectives. See `http://wordnet.princeton.edu` for more details.

A.4 Rule 4

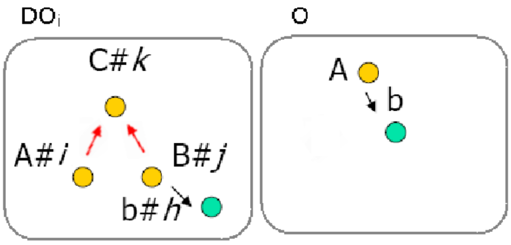


Figure A.4: Graphical representation of rule 4

For example, let us introduce a class labeled hotel with a datatype property name, and let DO_i contain a class building annotated as $building_{\#1}$ with two subclass, the first is a class annotated as $hotel_{\#1}$, the latter is class annotated as $restaurant_{\#1}$ with a datatype property name annotated as $name_{\#1}$. The application of this rule generates the annotation $hotel_{\#1}$ for the class labeled hotel, and $name_{\#1}$ for its datatype property.

A.5 Rule 5

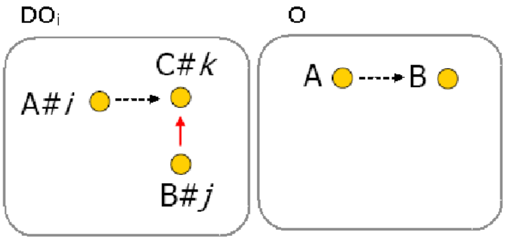


Figure A.5: Graphical representation of rule 5

For example, imagine in O we have a class labeled restaurants connected to a class named seafood by any object property (e.g. serves), and suppose DO_i contains a class restaurants annotated as $restaurant_{\#1}$ connected via some object property to a class food annotated as $food_{\#2}$, which in turn has a subclass seafood annotated as $seafood_{\#1}$. Then we conclude that $restaurant_{\#1}$ and $seafood_{\#1}$ are good candidates for the annotation of restaurants and seafood in O.

A.6 Rule 6

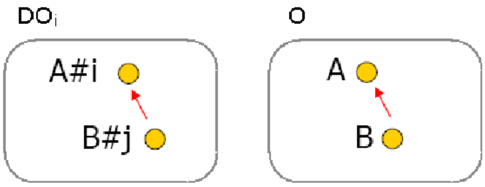


Figure A.6: Graphical representation of rule 6

For example, let us consider a pair of classes labeled person and client, where client is subclass of person, let DO_i contain a pair of classes annotated as $person_{\#1}$ and $client_{\#2}$, where $client_{\#2}$ is a subclass of $person_{\#1}$. The application of this rule generates the annotation $person_{\#1}$ and $client_{\#2}$ for the classes person and client.

A.7 Rule 7

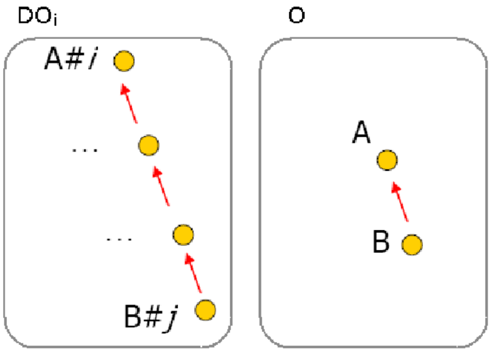


Figure A.7: Graphical representation of rule 7

For example, let us introduce a pair of classes labeled animal and dog (where dog is a subclass of animal), and let DO_i contain this subclass hierarchy: $animal_{\#1}$, $vertebrate_{\#1}$, $carnivore_{\#1}$ and $dog_{\#1}$. The application of this rule generates the annotations $animal_{\#1}$ and $dog_{\#1}$ for the classes animal and dog.

Bibliography

[1] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML.* Morgan Kaufmann, 1999.

[2] D. Aumueller, H. H. Do, S. Massmann, and E. Rahm. Schema and ontology matching with coma++. In *SIGMOD Conference*, pages 906–908, 2005.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley, 1999.

[4] S. Banerjee and T. Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *CICLing*, pages 136–145, 2002.

[5] S. Bechhofer, L. Carr, C. A. Goble, S. Kampa, and T. Miles-Board. The semantics of semantic annotation. In R. Meersman and Z. Tari, editors, *CoopIS/DOA/ODBASE*, volume 2519 of *Lecture Notes in Computer Science*, pages 1152–1167. Springer, 2002.

[6] R. Benassi, S. Bergamaschi, A. Fergnani, and D. Miselli. Extending a lexicon ontology for intelligent information integration. In R. L. de Mántaras and L. Saitta, editors, *ECAI*, pages 278–282. IOS Press, 2004.

[7] D. Beneventano, S. Bergamaschi, S. Bruschi, F. Guerra, M. Orsini, and M. Vincini. Instances navigation for querying integrated data from web-sites. In *WEBIST 2006, Proceedings of the Second International Conference on Web Information Systems and Technologies, Setubal, Portugal, April 11-13, 2006*, pages 46–53, 2005.

[8] D. Beneventano, S. Bergamaschi, F. Guerra, and M. Vincini. Synthesizing an integrated ontology. *IEEE Internet Computing*, pages 42–51, Sep-Oct 2003.

[9] D. Beneventano, S. Bergamaschi, and C. Sartori. Description logics for semantic query optimization in object-oriented database systems. *ACM Trans. Database Syst.*, 28:1–50, 2003.

[10] D. Beneventano, S. Bergamaschi, C. Sartori, and M. Vincini. ODB-QOPTIMIZER: A tool for semantic query optimization in oodb. In *Int. Conference on Data Engineering - ICDE97*, 1997. http://www.dbgroup.unimo.it.

[11] D. Beneventano, F. Guerra, M. Orsini, L. Po, A. Sala, M. D. Gioia, M. Comerio, F. de Paoli, A. Maurino, M. Palmonari, C. Gennaro, F. Sebastiani, A. Turati, D. Cerizza, I. Celino, and F. Corcoglioniti. Detailed design for building semantic peer. *Networked Peers for Business, Deliverable D.2.1, Final Version, available at http : //www.dbgroup.unimo.it/publication/d2_1.pdf*, pages 52–57, apr 2008.

[12] D. Beneventano, F. Guerra, M. Orsini, L. Po, A. Sala, M. D. Gioia, M. Comerio, F. de Paoli, A. Maurino, M. Palmonari, C. Gennaro, F. Sebastiani, A. Turati, D. Cerizza, I. Celino, and F. Corcoglioniti. Detailed design for building semantic peer. *Networked Peers for Business, Deliverable D.2.2, Draft Version*, pages 52–57, dec 2008.

[13] S. Bergamaschi, P. Bouquet, D. Giacomuzzi, F. Guerra, L. Po, and M. Vincini. An incremental method for the lexical annotation of domain ontologies. *Int. J. Semantic Web Inf. Syst.*, 3(3):57–80, 2007.

[14] S. Bergamaschi, S. Castano, D. Beneventano, and M. Vincini. Semantic integration of heterogeneous information sources. *Journal of Data and Knowledge Engineering*, 36(3):215–249, 2001.

[15] S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.

[16] S. Bergamaschi, L. Po, A. Sala, and S. Sorrentino. Data source annotation in data integration systems. In Koch et al. [64].

[17] S. Bergamaschi, L. Po, and S. Sorrentino. Automatic annotation for mapping discovery in data integration systems. In S. Gaglio, I. Infantino, and D. Saccà, editors, *SEBD*, pages 334–341, 2008.

[18] G. P. Bernardo Magnini, Carlo Strapparava and A. Gliozzo. The role of domain information in word sense disambiguation. In *Natural Language Engineering, special issue on Word Sense Disambiguation*, pages 359–373, 2002.

[19] P. A. Bernstein. The many roles of meta data in data integration. In F. Özcan, editor, *SIGMOD Conference*, page 792. ACM, 2005.

[20] P. Bouquet, L. Serafini, and S. Zanobini. Peer-to-peer semantic coordination. *Journal of Web Semantics*, 2(1), 2005.

[21] P. Bouquet, L. Serafini, S. Zanobini, and S. Sceffer. Bootstrapping semantics on the web: meaning elicitation from schemas. In *WWW2006 Conference Proceedings*. W3C, 2006.

[22] S. Brody and M. Lapata. Good neighbors make good senses: Exploiting distributional similarity for unsupervised WSD. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 65–72, Manchester, UK, August 2008. Coling 2008 Organizing Committee.

[23] S. Brody, R. Navigli, and M. Lapata. Ensemble methods for unsupervised wsd. In *ACL*. The Association for Computer Linguistics, 2006.

[24] P. Buitelaar and T. Declerck. Linguistic annotation for the semantic web. In *Annotation for the Semantic Web. IOS*. Press, 2003.

[25] D. Buscaldi, P. Rosso, and F. Masulli. Integrating conceptual density with wordnet domains and cald glosses for noun sense disambiguation. In J. L. V. González, P. Martínez-Barco, R. Muñoz, and M. Saiz-Noeda, editors, *EsTAL*, volume 3230 of *Lecture Notes in Computer Science*, pages 183–194. Springer, 2004.

[26] A. Cal, D. Calvanese, G. De Giacomo, and M. Lenzerini. On the expressive power of data integration systems. In *Proc. of the 21st Int. Conf. on Conceptual Modeling (ER 2002)*, volume 2503 of *Lecture Notes in Computer Science*, pages 338–350. Springer, 2002.

[27] Calvanese, D. Giacomo, Lenzerini, and Rosati. Logical foundations of peer-to-peer data integration. In *PODS: 23th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 2004.

[28] M. Carey, L. Haas, P. Schwarz, M. Arya, W. Cody, R. Fagin, M. Flickner, A. Luniewski, W. Niblack, D. Petkovic, J. T. II, J. Williams, and E. Wimmers. Towards heterogeneous multimedia information systems: The Garlic approach. *IBM Almaden Research Center, San Jose, CA 95120*, 1996.

[29] S. Castano, V. D. Antonellis, and S. D. C. di Vimercati. Global viewing of heterogeneous data sources. *IEEE Trans. Knowl. Data Eng.*, 13(2):277–297, 2001.

[30] S. Castano, A. Ferrara, D. Lorusso, T. H. Näth, and R. Möller. Mapping validation by probabilistic reasoning. In S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, editors, *ESWC*, volume 5021 of *Lecture Notes in Computer Science*, pages 170–184. Springer, 2008.

[31] N. Choi, I.-Y. Song, and H. Han. A survey on ontology mapping. *SIGMOD Record*, 35(3):34–41, 2006.

[32] K. Consortium. State of the art on ontology alignment. In *KnowledgeWeb Consortium*, 2004.

[33] N. N. Dalvi and D. Suciu. Management of probabilistic data: foundations and challenges. In L. Libkin, editor, *PODS*, pages 1–12. ACM, 2007.

[34] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology based access to distributed and semi-structured information. In R. Meersman, Z. Tari, and S. M. Stevens, editors, *DS-8*, volume 138 of *IFIP Conference Proceedings*, pages 351–369. Kluwer, 1999.

[35] S. Deerwester, S. Dumais, T. Landauer, G. W. Furnas, and R. H. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[36] L. Ding, R. Pan, T. W. Finin, A. Joshi, Y. Peng, and P. Kolari. Finding and ranking knowledge on the semantic web. In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 2005.

[37] H. H. Do and E. Rahm. Coma - a system for flexible combination of schema matching approaches. In *VLDB*, pages 610–621, 2002.

[38] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD Conference*, pages 509–520, 2001.

[39] A. Doan and A. Y. Halevy. Semantic integration research in the database community: A brief survey. *AI Magazine*, 26(1):83–94, 2005.

[40] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Learning to map between ontologies on the semantic web. In *WWW*, pages 662–673, 2002.

[41] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Ontology matching: A machine learning approach. In *Handbook on Ontologies*, pages 385–404. 2004.

[42] X. L. Dong, A. Y. Halevy, and C. Yu. Data integration with uncertainty. In Koch et al. [64], pages 687–698.

[43] P. Eklund. Large-scale cooperatively-built kbs. In *Conceptual Structures: Broadening the Base, number 2120 in Lecture Notes in Artificial Intelligence*, pages 231–244. SpringerVerlag, 2001.

[44] Fagin, Kolaitis, Miller, and Popa. Data exchange: Semantics and query answering. *TCS: Theoretical Computer Science*, 336, 2005.

[45] R. Florian, S. Cucerzan, and C. S. A. Yarowsky. Doi: 10.1017/s1351324902002978 printed in the united kingdom combining classifiers for word sense disambiguation, 2002.

[46] M. J. Franklin, A. Y. Halevy, and D. Maier. From databases to dataspaces: a new abstraction for information management. *SIGMOD Record*, 34(4):27–33, 2005.

[47] M. Friedman, A. Y. Levy, and T. D. Millstein. Navigational plans for data integration. In *AAAI/IAAI*, pages 67–73, 1999.

[48] A. Gangemi, R. Navigli, and P. Velardi. The ontowordnet project: Extension and axiomatization of conceptual relations in wordnet. In R. Meersman, Z. Tari, and D. C. Schmidt, editors, *CoopIS/DOA/ODBASE*, volume 2888 of *Lecture Notes in Computer Science*, pages 820–838. Springer, 2003.

[49] B. Ganter and R. Wille. *Formal Concept Analysis - Mathematical Foundations.* Springer, 1999.

[50] A. M. Gliozzo, C. Strapparava, and I. Dagan. Unsupervised and supervised exploitation of semantic domains in lexical disambiguation. *Computer Speech & Language*, 18(3):275–299, 2004.

[51] J. Gracia, V. Lopez, M. D'Aquin, M. Sabou, E. Motta, and E. Mena. Solving semantic ambiguity to improve semantic web based ontology matching. In P. Shvaiko, J. Euzenat, F. Giunchiglia, and B. He, editors, *Proceedings of the Workshop on Ontology Matching (OM2007) at ISWC/ASWC2007, Busan, South Korea*, November 2007.

[52] J. Gracia, R. Trillo, M. Espinoza, and E. Mena. Querying the web: a multiontology disambiguation method. In D. Wolber, N. Calder, C. H. Brooks, and A. Ginige, editors, *ICWE*, pages 241–248. ACM, 2006.

[53] T. R. Gruber and T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993.

[54] A. Y. Halevy, N. Ashish, D. Bitton, M. Carey, D. Draper, J. Pollock, A. Rosenthal, and V. Sikka. Enterprise information integration: successes, challenges and controversies. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 778–787, New York, NY, USA, 2005. ACM.

[55] A. Y. Halevy, Z. G. Ives, D. Suciu, and I. Tatarinov. Schema mediation in peer data management systems. In U. Dayal, K. Ramamritham, and T. M. Vijayaraman, editors, *ICDE*, page 505. IEEE Computer Society, 2003.

[56] A. Y. Halevy, A. Rajaraman, and J. J. Ordille. Data integration: The teenage years. In U. Dayal, K.-Y. Whang, D. B. Lomet, G. Alonso, G. M. Lohman, M. L. Kersten, S. K. Cha, and Y.-K. Kim, editors, *VLDB*, pages 9–16. ACM, 2006.

[57] S. Handschuh, S. Staab, and R. Volz. On deep annotation. In *Proceedings of the 12th International World Wide Web Conference*, pages 431–438, 2003.

[58] J. Heflin and J. Hendler. Searching the web with SHOE, May 04 2000.

[59] J. Heflin, J. A. Hendler, and S. Luke. Shoe: A blueprint for the semantic web. In D. Fensel, J. A. Hendler, H. Lieberman, and W. Wahlster, editors, *Spinning the Semantic Web*, pages 29–63. MIT Press, 2003.

[60] E. H. Hovy. Combining and standardizing large-scale, practical ontologies for machine translation and other uses. *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages Granada, Spain, May 28–30, AAAI Press, 1998.

[61] N. Ide and J. Véronis. Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

[62] Y. Kalfoglou, B. Hu, D. Reynolds, and N. Shadbolt. Semantic integration technonolgies survey. In *Technical Report, ECS, University of Southampton*, 2005.

[63] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: The state of the art. In Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.

[64] C. Koch, J. Gehrke, M. N. Garofalakis, D. Srivastava, K. Aberer, A. Deshpande, D. Florescu, C. Y. Chan, V. Ganti, C.-C. Kanne, W. Klas, and E. J. Neuhold, editors. *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*. ACM, 2007.

[65] M. R. Koivunen. Eswc 2005, usersweb workshop. 2005.

[66] P. G. Kolaitis. Schema mappings, data exchange, and metadata management. In ACM, editor, *Proceedings of the Twenty-Fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems: PODS 2005: Baltimore, Maryland, June 13-15, 2005*, pages 61–75, pub-ACM:adr, 2005. ACM Press.

[67] M. Lenzerini. Data integration: A theoretical perspective. In L. Popa, editor, *PODS*, pages 233–246. ACM, 2002.

[68] B. Louie, L. Detwiler, N. N. Dalvi, R. Shaker, P. Tarczy-Hornoch, and D. Suciu. Incorporating uncertainty metrics into a general-purpose data integration system. In *SSDBM*, page 19. IEEE Computer Society, 2007.

[69] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *VLDB*, pages 49–58, 2001.

[70] B. Magnini. Experiments in word domain disambiguation for parallel texts, Nov. 20 2000.

[71] C. C. Marshall. Toward an ecology of hypertext annotation. In *Hypertext*, pages 40–49. ACM, 1998.

[72] P. Martin and P. Eklund. Embedding knowledge in web documents. In *Proceedings of the Eighth International World Wide Web Conference*, pages 325–341, Toronto, Canada, May 1999. Elsevier.

[73] D. McCarthy and J. Carroll. Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences. *Computational Linguistics*, 29(4):639–654, 2003.

[74] E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. Domain specific ontologies for semantic information brokering on the global information infrastructure, 1998.

[75] A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[76] A. Montoyo, M. Palomar, and G. Rigau. Wordnet enrichment with classification systems. In *In Proc. of WordNet and Other Lexical Resources: Applications, Extensions and Customisations Workshop, NAACL-01*, pages 101–106, Carnegie Mellon Univ., Pittsburgh, USA, 2001.

[77] A. Montoyo, A. Suárez, G. Rigau, and M. Palomar. Combining knowledge- and corpus-based word-sense-disambiguation methods. *J. Artif. Intell. Res. (JAIR)*, 23:299–330, 2005.

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

[79] N. F. Noy and M. A. Musen. Prompt: Algorithm and tool for automated ontology merging and alignment. In *AAAI/IAAI*, pages 450–455, 2000.

[80] N. F. Noy and M. A. Musen. Promptdiff: A fixed-point algorithm for comparing ontology versions. In *AAAI/IAAI*, pages 744–750, 2002.

[81] T. Pahikkala, S. Pyysalo, F. Ginter, J. Boberg, J. Järvinen, and T. Salakoski. Kernels incorporating word positional information in natural language disambiguation tasks. In I. Russell and Z. Markov, editors, *FLAIRS Conference*, pages 442–448. AAAI Press, 2005.

[82] S. Parsons and A. Hunter. A review of uncertainty handling formalisms. In A. Hunter and S. Parsons, editors, *Applications of Uncertainty Formalisms*, volume 1455 of *Lecture Notes in Computer Science*, pages 8–37. Springer, 1998.

[83] M. T. Pazienza and A. Stellato. An open and scalable framework for enriching ontologies with natural language content. In M. Ali and R. Dapoigny, editors, *IEA/AIE*, volume 4031 of *Lecture Notes in Computer Science*, pages 990–999. Springer, 2006.

[84] L. Po. Improving data integration through disambiguation techniques. In E. Kapetanios, V. Sugumaran, and M. Spiliopoulou, editors, *NLDB*, volume 5039 of *Lecture Notes in Computer Science*, pages 372–375. Springer, 2008.

[85] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov. Kim - semantic annotation platform. In D. Fensel, K. P. Sycara, and J. Mylopoulos, editors, *International Semantic Web Conference*, volume 2870 of *Lecture Notes in Computer Science*, pages 834–849. Springer, 2003.

[86] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB J.*, 10(4):334–350, 2001.

[87] P. Resnik and D. Yarowsky. Distinguishing systems and distinguishing senses: new evaluation methods for word sense disambiguation. *Natural Language Engineering*, 5(2):113–133, 2000.

[88] M. Sabou, M. d'Aquin, and E. Motta. Using the semantic web as background knowledge for ontology mapping. In P. Shvaiko, J. Euzenat, N. F. Noy, H. Stuckenschmidt, V. R. Benjamins, and M. Uschold, editors, *Ontology Matching*, volume 225 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

[89] M. Sabou, M. Dzbor, C. Baldassarre, S. Angeletou, and E. Motta. Watson: A gateway for the semantic web. In *Poster session of the European Semantic Web Conference, ESWC*, 2007.

[90] M. Sabou, J. Gracia, S. Angeletou, M. d'Aquin, and E. Motta. Evaluating the semantic web: A task-based approach. In K. Aberer, K.-S. Choi, N. F. Noy, D. Allemang, K.-I. Lee, L. J. B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *ISWC/ASWC*, volume 4825 of *Lecture Notes in Computer Science*, pages 423–437. Springer, 2007.

[91] A. D. Sarma, X. Dong, and A. Y. Halevy. Bootstrapping pay-as-you-go data integration systems. In J. T.-L. Wang, editor, *SIGMOD Conference*, pages 861–874. ACM, 2008.

[92] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[93] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. pages 146–171, 2005.

[94] S. Staab, A. Maedche, and S. Handschuh. An annotation framework for the semantic web. In *S. Ishizaki (ed.), Proc. of The First International Workshop on MultiMedia Annotation. January. 30 - 31. Tokyo, Japan*, 2001.

[95] G. Stumme and A. Maedche. Fca-merge: Bottom-up merging of ontologies. In B. Nebel, editor, *IJCAI*, pages 225–234. Morgan Kaufmann, 2001.

[96] R. Trillo, J. Gracia, M. Espinoza, and E. Mena. Discovering the semantics of user keywords. *J. UCS*, 13(12):1908–1935, 2007.

[97] P. Vossen, editor. *EuroWordNet: a multilingual database with lexical semantic networks.* Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[98] L. L. .Yan, R. J. Miller, L. M. Haas, and R. Fagin. Data-driven understanding and refinement of schema mappings. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 30(2):485–496, 2001.