INTERNATIONAL DOCTORATE SCHOOL IN INFORMATION AND COMMUNICATION TECHNOLOGIES

DISI - UNIVERSITY OF TRENTO

DEALING WITH SEMANTIC HETEROGENEITY IN CLASSIFICATIONS

Vincenzo Maltese

Advisor:
Prof. Fausto Giunchiglia
Università degli Studi di Trento

Abstract

Many projects have dealt with mappings between classifications both in computer science and digital library communities. The adopted solutions range from fully manual to fully automatic approaches. Manual approaches are very precise, but automation becomes unavoidable when classifications contain thousands of nodes with millions of candidate correspondences. As fundamental preliminary step towards automation, S-Match converts classifications into formal ontologies, i.e. *lightweight ontologies*. Despite many solutions to the problem have been offered, with S-Match representing a state of the art matcher with good accuracy and run-time performance, there are still several open problems. In particular, the problems addressed in this thesis include: (a) *Run-time performance.* Due to the high number of calls to the SAT reasoning engine, semantic matching may require exponential time; (b) *Maintenance*. Current matching tools offer poor support to users for the process of creation, validation and maintenance of the correspondences; (c) *Lack of background knowledge*. The lack of domain specific background knowledge is one important cause of low recall. As significant progress to (a) and (b), we describe MinS-Match, a semantic matching tool we developed evolving S-Match that computes the *minimal mapping* between two lightweight ontologies. The minimal mapping is that minimal subset of correspondences such that all the others can be efficiently computed from them and are therefore said to be redundant. We provide a formal definition of *minimal* and, dually, *redundant mappings*, evidence of the fact that the minimal mapping always exists and it is unique and a correct and complete algorithm for computing it. Our experiments demonstrate a substantial improvement in run-time. Based on this, we also developed a method to support users in the validation task that allows saving up to 99% of the time. We address problem (c) by creating and by making use of an extensible *diversity-aware knowledge base* providing a continuously growing quantity of properly organized knowledge. Our approach is centered on the fundamental notions of *domain* and *context*. Domains, developed by adapting the *faceted approach* from library science, are the main means by which diversity is captured and allow scaling as with them it is possible to add new knowledge as needed. Context allows a better disambiguation of the terms used and reducing the complexity of reasoning at run-time. As proof of the applicability of the approach, we developed the *Space* domain and applied it in the Semantic Geo-Catalogue (SGC) project.

Keywords: Semantic matching; minimal mappings; mapping validation; diversity-aware knowledge base; domains; context;

Acknowledgements

The research leading to these results has received funding from:

- The European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 231126, LivingKnowledge: LivingKnowledge - Facts, Opinions and Bias in Time;
- The European Social Fund under the act n° 1637 (30.06.2008) of the Autonomous Province of Trento (the Semantic Geo-Catalogue project);
- Live Memories - Active Digital Memories of Collective Life" funded by the Autonomous Province of Trento;
- The Interconcept project between the University of Maryland, the University of Trento and the U.S. National Agricultural Library (NAL).

I would like to thank all my colleagues of the KnowDive group and in particular my advisor Fausto Giunchiglia for his constant guidance in these years of intense work. A special thank goes to Devika Madalli, Dagobert Soergel and Ilya Zaihrayeu for their kind support and the fruitful discussions, especially during the early years of my PhD.

Previously published material

The work at the basis of this thesis has been exploited as follows:

Computation of the minimal mapping

- Giunchiglia, F., Maltese, V., Autayeu A. (2012). Computing minimal mappings between lightweight ontologies. International Journal on Digital Libraries. DOI: 10.1007/s00799-012-0083-2. http://www.springerlink.com/content/aq381263656v6755/
- Giunchiglia, F., Maltese, V., Autayeu, A. (2009). Computing minimal mappings. 4th Ontology Matching Workshop. http://eprints.biblio.unitn.it/archive/00001525/01/078.pdf
- Maltese, V., Autayeu A. (2009). Computing minimal and redundant mappings between lightweight ontologies. First AISB Workshop on Matching and Meaning. http://eprints.biblio.unitn.it/archive/00001526/01/079.pdf

Mapping validation

- Giunchiglia, F., Soergel, D., Maltese, V., Bertacco, A. (2009). Mapping large-scale Knowledge Organization Systems. 2nd International Conference on the Semantic Web and Digital Libraries (ICSD). http://eprints.biblio.unitn.it/archive/00001616/01/029.pdf
- Maltese, V., Giunchiglia, F., Autayeu, A. (2010). Save up to 99% of your time in mapping validation. 9th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE). http://eprints.biblio.unitn.it/archive/00001883/01/046.pdf

Mapping evaluation

- Autayeu, A., Maltese, V., Andrews, P. (2010). Recommendations for Better Quality Ontology Matching Evaluations. 2nd AISB Workshop on Matching and Meaning. http://eprints.biblio.unitn.it/archive/00001834/01/024.pdf
- Autayeu, A., Maltese, V., Andrews, P. (2009). Best practices for ontology matching tools evaluation. Poster at the 4th Ontology Matching Workshop. http://eprints.biblio.unitn.it/archive/00001655/01/046.pdf

Domain knowledge

- Giunchiglia, F., Maltese, V., Dutta, B. (2012). Domains and context: first steps towards managing diversity in knowledge. Journal of Web Semantics, special issue on Reasoning with Context in the Semantic Web. DOI: 10.1016/j.websem.2011.11.007 http://dx.doi.org/10.1016/j.websem.2011.11.007
- Maltese, V., Farazi, F. (2011). Towards the Integration of Knowledge Organization Systems with the Linked Data Cloud. UDC Seminar. http://eprints.biblio.unitn.it/archive/00002214/01/techRep466.pdf
- Maltese, V., Giunchiglia, F., Denecke, K., Lewis, P., Wallner, C., Baldry, A., Madalli, D. (2009). On the interdisciplinary foundations of diversity. First Living Web Workshop at ISWC. http://eprints.biblio.unitn.it/archive/00001651/01/040.pdf
- Giunchiglia, F., Maltese, V. (2010). Ontologie leggere a faccette. AIDA Informazioni. Rivista di Scienze dell'Informazione, n. 3-4, 87-106. http://eprints.biblio.unitn.it/archive/00001798/01/005.pdf
- Giunchiglia, F. Dutta, B. Maltese, V. (2009). Faceted lightweight ontologies. In "Conceptual Modeling: Foundations and Applications", A. Borgida, V. Chaudhri, P. Giorgini, Eric Yu (Eds.) LNCS Springer. http://eprints.biblio.unitn.it/archive/00001610/01/022.pdf

The Space domain

- Giunchiglia, F., Dutta, B., Maltese, V., Farazi, F. (2012). A facet-based methodology for the construction of a large-scale geo-spatial ontology. Journal of Data Semantics. DOI: 10.1007/s13740-012-0005-x http://eprints.biblio.unitn.it/archive/00002271/01/techRep479.pdf
- Dutta, B., Giunchiglia, F., Maltese, V. (2011). A facet-based methodology for geo-spatial modelling. Geospatial semantics conference (GEOS), 6631, 133–150. http://eprints.biblio.unitn.it/archive/00001928/01/062.pdf
- Giunchiglia, F., Maltese, V., Farazi, F., Dutta, B. (2010). GeoWordNet: a resource for geo-spatial applications. 7th Extended Semantic Web Conference (ESWC). http://eprints.biblio.unitn.it/archive/00001777/01/071.pdf

The semantic Geo-Catalogue of the Autonomous Province of Trento

- Farazi, F., Maltese, V., Dutta, B., Ivanyukovich, A. (submitted). A semantic geo-catalogue for a local administration. AI Review Journal. http://eprints.biblio.unitn.it/archive/00002276/01/techRep483.pdf

- Farazi, F., Maltese, V., Giunchiglia, F., Ivanyukovich, A. (2011). A faceted ontology for a semantic geo-catalogue. 8th Extended Semantic Web Conference (ESWC). http://eprints.biblio.unitn.it/archive/00001927/01/061.pdf
- Farazi, F., Maltese, V., Dutta, B., Ivanyukovich, A. (2011). Extending a geo-catalogue with matching capabilities. LHD Workshop on Discovering Meaning On the Go in Large Heterogeneous Data. http://eprints.biblio.unitn.it/archive/00002211/01/techRep464.pdf
- Shvaiko, P., Ivanyukovich, A., Vaccari, L., Maltese, V., Farazi, F. (2010). A semantic geo-catalogue implementation for a regional SDI. INSPIRE conference. http://eprints.biblio.unitn.it/archive/00001850/01/033.pdf

Contents

Chapter 1

1. Introduction

1.1. The Context

Semantic heterogeneity, in its broader sense, can be seen as the difficulty of establishing a certain level of connectivity between people, software agents or IT systems [Uschold and Gruninger, 2004] at the purpose of enabling each of the parties to appropriately *understand* the exchanged information [Pollock, 2002].

Early connectivity has focused on physical and syntactic layers only. Physical connectivity relies on the presence of a stable communication channel between the parties, for instance ODBC data gateways and software adapters. Syntactic connectivity is established by instituting a common vocabulary of terms to be used by the parties or by point-to-point bridges that translate messages written in one vocabulary in messages in the other vocabulary. This rigidity and lack of explicit meaning causes very high maintenance costs (up to 95% of the overall ownership costs) as well as integration failure (up to 88% of the projects) [Pollock, 2002]. Standard vocabularies, by fixing the terminology to be used in a broad area, mitigate the problem but they are difficult to develop and maintain. In fact, they imply accomplishing a very high level of agreement between the parties. Ultimately, these solutions all aim at *integration*, i.e. at a very tight and rigid connection between the parties.

An alternative solution, that is at the basis of the Semantic Web vision [Berners-Lee et al., 2001], is represented by the establishment of some form of *semantic interoperability* between the parties, i.e. the possibility to exchange information by reaching a certain degree of agreement about the content meaning, still maintaining local autonomy in the maintained data, in the way the terminology is used and in the way the computation is performed locally. In turn, by not hard coding all knowledge in proprietary code and scripts, interoperability solutions allow reducing operational and maintenance costs. In the past recent years, many techniques to achieve semantic interoperability have been proposed. These techniques tend to follow a similar pattern and are characterized by the following key points [Pollock, 2002]:

- ***Semantic mediation***: the usage of an ontology [Studer et al.,1998], providing a shared vocabulary of terms with explicit meaning.
- ***Semantic mapping***: using the ontology, the establishment of a mapping constituted by a set of correspondences between semantically similar data elements independently maintained by the parties.
- ***Context sensitivity***: the mapping has contextual validity, i.e. it has to be used by taking into account the conditions and the purposes for which it was generated.

1.2. The Problem

In this thesis, the work focuses on the sub-problem of establishing the semantic mapping between classifications, i.e. a set of correspondences between the nodes of tree-like hierarchical structures where node labels are associated with natural language terms. Classifications are typically used to index and search bibliographic and digital material. This is a hard problem since they may differ in structure, reflect different visions of the world, contain different terminology and polysemous terms, have different degrees of specificity, scope and coverage, can be expressed in different languages [Giunchiglia et al., 2009b].

Many projects have dealt with mappings between classifications, for example the German CARMEN[1], the EU Project Renardus [Koch et al., 2003], and OCLC initiatives [Vizine-Goetz et al., 2004]. One possible approach is to exploit mappings from a reference scheme, or spine, to search and navigate across a set of satellite vocabularies. For instance, Renardus and HILT [Nicholson et al., 2006] use the Dewey Decimal Classification[2] (DDC). Some others prefer the Library of Congress Subject Headings[3] (LCSH) [Whitehead, 1990], [O'Neill and Chan, 2003]. Some of them are based on fully manual approaches, while others rely on automatic tools for the identification of an initial set of correspondences to be manually validated and augmented. See for instance [Falconer and Storey, 2007], which also describes a tool that supports this task. A quite recent paper [Lauser et al., 2008] focusing on the agricultural domain compares the two approaches and concludes that automatic procedures can be very effective but tend to fail when background knowledge is needed. [Shvaiko and Euzenat, 2007] and [Noy, 2004] represent two

[1] http://www.bibliothek.uni-regensburg.de/projects/carmen12/index.html
[2] http://www.oclc.org/dewey/
[3] http://www.loc.gov/aba/cataloging/subject/

good surveys of the state of the art on mapping computation and integration. The OAEI[4] initiative is a large-scale challenge aiming at providing an evaluation of such kind of tools.

Even if classifications turn out to be very effective in manual tasks, the ambiguity of the labels represents a serious barrier towards the automation of such processes. This problem motivated Giunchiglia et al. (see for instance [Giunchiglia et al., 2007a] to develop a series of techniques to formalize a classification into a *lightweight ontology*. With the conversion procedure they established, each node in the classification is associated a formula in a formal language codifying the meaning of the node given the context in which it appears (i.e. the path from the root to the node). Lightweight ontologies are taken in input by the S-Match tool [Giunchiglia et al., 2007c], in order to compute semantic relations between the nodes in the two ontologies. Possible semantic relations include disjointness ($\perp$), equivalence ($\equiv$), more specific ($\sqsubseteq$) and less specific ($\sqsupseteq$).

Despite many solutions to the problem of computing the semantic mapping between classifications have been offered, with S-Match representing a state of the art matcher with good accuracy and run-time performance, there are still several open problems (Shvaiko, and Euzenat, 2008). In particular, the problems addressed in this thesis include:

a. ***Run-time performance.*** Due to the high number of calls to the SAT reasoning engine and that SAT may require exponential time, computing the mapping is a very time consuming task [Giunchiglia et al., 2007c];
b. ***Maintenance***. Current matching tools offer poor support to users for the process of creation, validation and maintenance of the correspondences [Falconer and Storey, 2007]. Moreover they suffer of scalability problems as the number of nodes and correspondences grows [Robertson et al., 2005]. As a consequence, handling them turns out to be a very complex, slow and error prone task;
c. ***Lack of background knowledge***. As underlined by several studies, for instance in [Lauser et al., 2008] and in [Giunchiglia et al., 2006], the lack of domain specific background knowledge is one important cause of low recall.

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

1.3. The Solution

As significant progress in the area for the problems (a) and (b) presented in the previous section, in this thesis we describe MinSMatch, a semantic matching tool we developed evolving S-Match that takes two classifications, preliminary translated into lightweight ontologies, and computes the *minimal mapping* between them. The minimal mapping is that minimal subset of correspondences such that all the others can be efficiently computed from them, and are therefore said to be redundant. We provide a formal definition of *minimal* and, dually, *redundant mappings*, evidence of the fact that the minimal mapping always exists and it is unique and a correct and complete algorithm for computing it.

Our experiments demonstrate a substantial improvement in run-time, given a significant saving in the number of calls to SAT, and some improvement in the number of correspondences found w.r.t. S-Match. They also show that the number of correspondences in the minimal mapping is typically a very small portion of the overall set of correspondences between the two ontologies, up to 99% smaller. Therefore, minimal mappings have clear advantages in maintenance, visualization and user interaction. As we explain in this thesis, this is particularly important to reduce the effort in mapping validation. Being aware that the matching process cannot be completely automated and leveraging on the properties of minimal mappings, we propose the specifications for a new tool to interactively assist the user in the process of mapping creation and validation. Furthermore, the maintenance of smaller sets makes the work of the user much easier, faster and less error prone [Meilicke et al, 2008]. At the best of our knowledge no other tools directly compute minimal mappings.

Nevertheless, one of the main barriers towards the use of semantics is the lack of background knowledge (problem (c)). Dealing with this problem has turned out to be a very difficult task because on the one hand the background knowledge should be very large and virtually unbound and, on the other hand, it should be context sensitive and able to capture the diversity of the world, for instance in terms of language and knowledge. We address the problem by creating and by making use of an extensible diversity-aware knowledge base providing a continuously growing quantity of properly organized knowledge. This is done by adapting the *faceted approach*, a well-established methodology used in library science for the organization of knowledge in libraries [Ranganathan, 1967]. Our approach is centered on the fundamental notions of *domain* and *context*.

By domain we mean *any area of knowledge or field of study that we are interested in or that we are communicating about*. For instance they may include conventional fields of study (e.g., library science, mathematics, physics), applications of pure disciplines (e.g., engineering, agriculture), any aggregate of such fields (e.g., physical sciences, social sciences), and they may also capture knowledge about our everyday lives (e.g., Space, Time, music, movie, sport, recipes, tourism). Domains have two important properties. They are the main means by which diversity is captured, in terms of language, knowledge and personal experience. For instance, according to the personal perception and purpose, the *Space* domain may or may not include buildings and man-made structures; the food domain may or may not include dogs according to the local customs. Moreover, domains allow scaling as with them it is possible to add new knowledge at any time as needed. For instance, while initially local applications may require only knowledge of the *Space* domain, due to new scenarios, the food domain might be needed and added.

The notion of context, described for instance in [Giunchiglia, 2006], allows on the one hand a better disambiguation of the terms used (i.e. by making explicit some of the assumptions left implicit) and on the other hand, by selecting from the domains the language and the knowledge which are strictly necessary to solve the problem, it allows reducing the complexity of reasoning at run-time.

With the thesis we introduce the overall approach and describe our first steps towards the construction of a large scale diversity-aware knowledge base. This is done by vertically applying the methodology on the *Space* domain, a rather important domain given its pervasiveness, and by showing how we use it in practice in a concrete scenario, i.e. for the semantic extension of the Semantic Geo-Catalogue of the Autonomous Province of Trento (PAT) in Italy.

1.4. Structure of the Thesis

The thesis is organized as follows. Chapter 2 introduces the state of the art notions of ontology, lightweight ontology and semantic matching. Chapter 3 provides the definition of minimal and redundant mapping and the algorithms we propose for their computation. Chapter 4 focuses on the problem of mapping validation, describes user interaction issues and the experiments conducted with large-scale library classifications that prove the effectiveness of the proposed approach. Chapter 5 explores some important issues concerning gold standards and the evaluation

of different semantic matching tools. Chapter 6 focuses on the problem of building domain specific background knowledge. Starting from the analysis of the state of the art, it describes the faceted approach, the notion of diversity and our approach to domain representation and construction. The chapter concludes by showing how the matching problem benefits from the new approach. Chapter 7 explains the steps followed for the creation of the Space domain. Chapter 8 describes the work done with the Semantic Geo-Catalogue of the PAT. Finally Chapter 9 concludes the thesis by summarizing the work done and outlying the future work.

Chapter 1. **Introduction**

Chapter 2

2. State of the Art

In this section we first present the general notion of ontology (Section 2.1). We mainly distinguish between *descriptive ontologies,* i.e. ontologies built to describe a domain, and *classification ontologies*, i.e. ontologies built to classify documents. We will emphasize that the difference in the purpose is reflected in the different semantics of nodes and links.

We then present *lightweight ontologies* (Section 2.2) as formal classification ontologies. As it will be explained, the translation of classifications into lightweight ontologies is an essential preliminary step towards the automation of task such as semantic matching.

Finally, we present the state of the art matcher S-Match (Section 2.3), by focusing on the algorithm and its evaluation. Evaluation results show that S-Match on average outperforms other systems in terms of precision/recall and that it is significantly faster than the others. Nevertheless, several challenges have been identified. We conclude the section by summarizing those addressed with this thesis.

2.1. Ontologies

Ontologies have been used for centuries in different communities, for different purposes and with different modalities [Giunchiglia et al., 2009b]. The concept originated more than two thousand years ago from philosophy and more specifically from Aristotle's theory of categories[5]. The original purpose was to provide a categorization of all existing things in the world. Ontologies have been lately adopted in several other fields, such as Library and Information Science (LIS), Artificial Intelligence (AI), and more recently in Computer Science (CS), as the main means for describing how classes of objects are correlated, or for categorizing what archivists generically call documents (it can be any physical or digital material).

Many definitions have been provided. Studer et al. [1998], by extending the famous definition by Gruber [1993], define an ontology as follows:

> **Definition 1: (Ontology).** An ontology is a formal, explicit specification of a shared conceptualization.

The notion of *conceptualization* refers to an abstract model of how people theorize (part of) the world in terms of basic cognitive units called concepts. Concepts represent the intention, i. e. the set of properties that distinguish the concept from others, and summarize the extension, i.e. the set of objects having such properties. Concepts basically denote classes of objects. For instance, the medicine domain can be theorized in terms of doctors, patients, body parts, diseases, their symptoms and treatments used to cure or prevent diseases. *Explicit specification* means that the abstract model is made explicit by providing names and definitions for the concepts. In other words the name and the definition of the concept provide a specification of its meaning in relation with other concepts. The specification is said to be *formal* when it is written in a language with formal syntax and formal semantics, i.e. in a logic-based language. Given their ambiguity, this excludes the use of natural languages. The conceptualization is *shared* in the sense that it captures knowledge which is common to a community of people and therefore represents concretely the level of agreement reached in that community. An ontology, by providing a common formal terminology and understanding of a given domain of interest, allows for automation (logical inference), supports reuse and favor interoperability across applications and people. When

5 http://plato.stanford.edu/entries/aristotle-categories/

an ontology is populated with the instances of the classes, i.e. the individuals, it is called a *knowledge base*. In literature the terms *TBox* and *ABox* are often used to denote what is known about the classes and about the individuals, respectively.

Given that the common core is represented by a vocabulary of terms and corresponding specification of their meaning, there are however different kinds of ontologies, according to the degree of formality and expressivity of the language used to describe them [Uschold and Gruninger, 2004]. As depicted in Fig. 1, this generates a continuum of kinds of ontologies ranging from informal lists of terms, informal schemes like user classifications (e.g. the structure of folders in a file system) and Web directories (e.g. DMOZ, Yahoo! and Google[6]), to progressively more formal schemes like enumerative classification schemes (e.g. the DDC and the Library of Congress Classification[7] (LCC)), thesauri (e.g. AGROVOC[8] and NALT[9]), and faceted classification schemes (e.g., the Colon Classification (CC)), and, ultimately, formal ontologies which are expressed in a logic formal language such as DL or OWL.

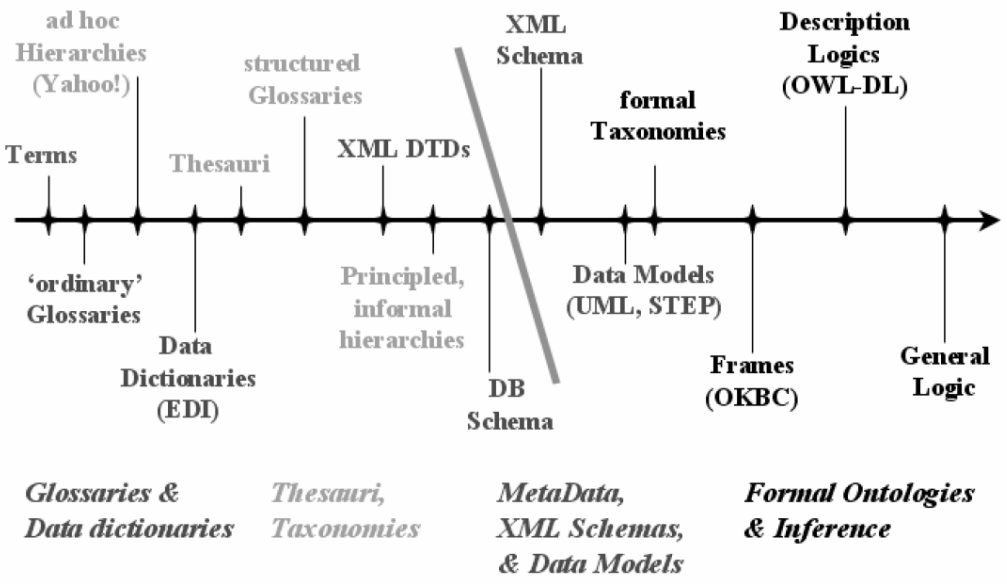


Fig. 1. Kinds of ontologies, taken from [Uschold and Gruninger, 2004].

The difference in the level of formality and expressivity is typically a function of the intended purpose. As a matter of fact, it is a well-known result in formal languages that the more expressive is the language the less efficient reasoning engines are. It is therefore fundamental to find the right balance in the level of formality and expressivity according to the problem to solve. For

[6] http://dmoz.org/; http://dir.yahoo.com/; http://directory.google.com/
[7] http://www.loc.gov
[8] http://aims.fao.org/website/AGROVOC-Thesaurus/sub
[9] http://agclass.nal.usda.gov/

the purposes of this work, and following the terminology provided in [Giunchiglia and Zaihrayeu, 2008], we mainly distinguish between:

- ***Descriptive ontologies***: ontologies which are mainly used to *describe* objects
- ***Classification ontologies***: ontologies which are mainly used to *categorize* objects

This distinction is reflected into the underlying semantics taken as reference, namely the *classification semantics* and the *real world semantics* described below.

2.1.1 Descriptive ontologies

Schemes built to describe a domain, called descriptive ontologies, are in *real world semantics* [Giunchiglia et al., 2009a] where terms at nodes represent either individuals or classes of real world objects.

Consider the example in Fig. 2, taken from [Maltese and Farazi, 2011]. It shows a scheme to describe some organizations and where they are located. White nodes represent classes while black nodes represent individuals. The first label at the nodes is the preferred term. Additional synonymous terms are eventually provided separated by semicolon. Arrows represent relations and the direction of the arrows indicates the direction of the relation. For instance, the term *country* (in the sense of the territory occupied by a nation) denotes all the real world countries, while the term *Italy* denotes Italy the country. Under this semantics, there is an *instance-of* relation between *country* (the class) and *Italy* (the individual). Other typical relations include *is-a* between classes (connecting a subclass to a class) and *part-of* between classes or between instances.

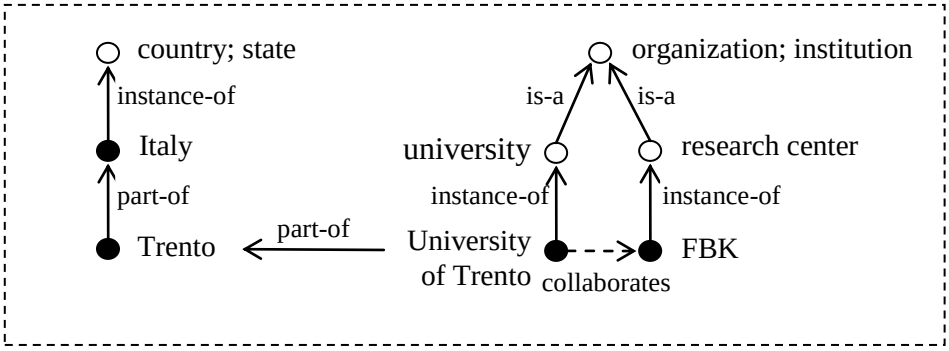


Fig. 2. An example of descriptive ontology

These schemes represent what we know about the domain and can be used to reason about it. Typical queries can include for instance:

1. Give me all the countries
2. Give me all the organizations
3. Give me all the organizations located in Italy

By exploiting the instances of the class *country*, the output of the first query is clearly {*Italy*}. The output of the second one - by leveraging on logical inference - should consider also the classes that are more specific than *organization* (*university* and *research center*) and therefore is {*University of Trento, FBK*}. To respond to the third query, one should also exploit the *part-of* relations between the entities and therefore, by assuming the *part-of* as transitive, it should return {*University of Trento*}.

In order to automate tasks, one should convert these schemes into formal (descriptive) ontologies. By using Description Logics (DL) [Baader et al., 2002], with the conversion:

- classes correspond to concepts
- instances correspond to individuals in the domain of interpretation
- *is-a* relations are translated into logical subsumption ($\sqsubseteq$)
- other relations correspond to DL roles

Specifically, the scheme in Fig. 2 can be codified with the following TBox and ABox:

TBox	**ABox**
university $\sqsubseteq$ organization	country(Italy)
research-center $\sqsubseteq$ organization	university(University of Trento)
	research-center(FBK)
	part-of(Trento, Italy)
	part-of(University of Trento, Trento)
	collaborates(University of Trento, FBK)

Deciding on the transitivity of the relations is an important choice in modelling. In DL there are ways to enforce transitivity of roles [Horrocks and Sattler, 1999], e.g. for the *part-of* relation above (subsumption itself is assumed to be transitive). However, this might be problematic.

There are several works about the transitivity of *part-of* relations. As described in [Varzi, 2006], the generic *part-of* relation is always transitive. However, if we start distinguishing about the different kinds of *part-of* then they might lose the transitivity property, in particular when we try to combine them together. The typical example is the handle that is part of the door that is part of the house that after a chain of other *part-of* relations ends to be part of the universe.

In our example, we may say that the *part-of* relation between *Italy* and *Trento* is an administrative *part-of* relation, while the one between *Trento* and *University of Trento* can be characterized as being a topological *part-of* or even just a generic associative relation. In fact, it is actually the building hosting the university as institution that is located in Trento, not the institution as such. The response to the query (3) will highly depend on whether or not we consider the composition of these relations to be transitive.

To publish the ontology, for instance as linked data [Bizer et al., 2009], one may encode it using the RDF[10] Web language. A fragment of a possible translation would look as follows:

```
<!--Classes-->
<rdfs:Class rdf:about="#research_center">
   <rdfs:subClassOf rdf:resource="#organization"/>
</rdfs:Class>
<rdfs:Class rdf:about="#university">
   <rdfs:subClassOf rdf:resource="#organization"/>
</rdfs:Class>
<!--Properties-->
<rdf:Property rdf:about="#collaborates"/>
<rdf:Property rdf:about="#part_of"/>
<!--Individuals-->
<administrative_division rdf:about="#Trento">
   <part_of rdf:resource="#Italy"/>
</administrative_division>
<university rdf:about="#University_of_Trento">
   <collaborates rdf:resource="#FBK"/>
```

[10] http://www.w3.org/RDF/

```
    <part_of rdf:resource="#Trento"/>
  </university>
```

In this representation the constructs *rdfs:Class* and *rdf:Property* are used to encode classes and properties, respectively. By linking the RDF code to a standard vocabulary such as WordNet[11] [Miller, 1998], we can disambiguate the meaning of the classes university and organization to university sense #3 (*a large and diverse institution of higher learning created to educate for life and for a profession and to grant degrees*) and organization sense #1 (*a group of people who work together*), respectively. This assignment is consistent with the code above since in WordNet university sense #3 *is-a* organization sense #1.

Nevertheless by construction RDF cannot prevent the modeller to add a new relation between the class university with a new class artefact (*a man-made object taken as a whole*) to enforce that a university as artefact can be *part-of* a location, e.g. that the University of Trento is *part-of* Trento. This makes the meaning of the class university ambiguous. In fact, university as artefact would rather match with university sense #2 (*establishment where a seat of higher learning is housed, including administrative and living quarters as well as facilities for research and teaching*). This could be prevented by making artefact and organization disjoint, but RDF does not support the use of disjointness. An immediate consequence is that if we have two RDF ontologies, the first codifying university as organization and the other codifying university as artefact, nothing prevents to merge them into one single class when integrating them.
Another well-known limitation of RDF is that, even if it distinguishes between classes and instances, a class can be treated as an instance [Brickley and Guha, 2004]. Moreover, in RDF transitivity cannot be enforced at the level of instances. In the example, this pertains in particular the transitivity of the *part-of* between University of Trento and Trento.

2.1.2 Classification ontologies

Schemes built to classify documents, called classification ontologies, are in *classification semantics* [Giunchiglia et al., 2009a] where terms at nodes always represent classes of documents. In this respect in these schemes the instances are the documents themselves.
Consider the example in Fig. 3, taken from [Maltese and Farazi, 2011]. It shows a thesaurus built with the purpose of classifying documents by country and by organization. Similarly to Fig. 2,

[11] http://wordnet.princeton.edu/

labels at the nodes denote the preferred term, optionally followed by synonymous terms separated by semicolon, while arrows represent relations. Documents at nodes are denoted with the letter *d* followed by an index. In these schemes NT/BT (narrower term/broader term) relations (where the direction of each arrow goes from the narrower to the broader term) - being hierarchical - mainly serve the purpose of facilitating the indexing and search tasks, while the RT (related term) relations - being associative - are mostly used for navigational purposes or for query expansion (to increase recall). In particular, following NT relations will allow identifying progressively more specific concepts (thus decreasing the extension, i.e. the set of documents about the concept) while following the inverse direction using the BT relation will allow identifying progressively more general concepts (thus increasing the extension).

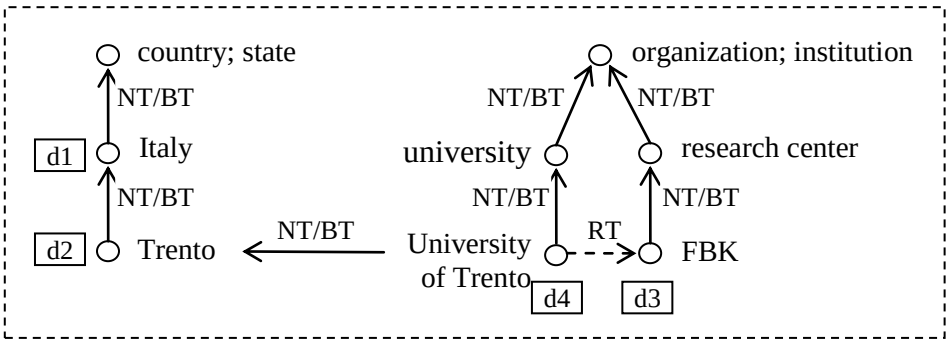


Fig. 3. An example of classification ontology

In the example, the term *country* denotes all documents about countries. Under this semantics NT/BT relations represent subset/superset relations (where NT and BT are one the inverse of the other). For instance, if the node *Italy* is connected to *country* through a BT relation, then the semantics of the node *Italy* is the set of documents about Italy the country.
We can use the scheme to classify documents and to search or browse a document collection. Typical queries can include for instance:

1. Give me all documents about Italy
2. Give me all documents about countries

What is the output of the first query? Actually this is a bit tricky. Assume we always apply query expansion, but without using RT relations. Somebody may argue that it should correspond to the set {d1, d2, d4}, while some others may rather propose {d1, d2}. This depends on the nature of the NT relation between *Trento* and *University of Trento*. If transitive, then the output should be

the former, otherwise the latter. This is even more evident by looking at the second query. One may expect as output the set {d1}, but actually according to the transitivity or not of the NT relations below the node *country*, one may have {d1, d2, d4} (if all the relations are transitive), {d1, d2} (if the relation between *Trento* and *University of Trento* is not transitive) or {d1} (if none of them is transitive). For what said in the previous section, if not transitive they should not even be encoded as NT relations, but rather as RT relations.

To make explicit the intended semantics and automate tasks one should provide a formal representation of the schema. Once again, by using DL the schema can be converted into the corresponding formal (classification) ontology. With the conversion:

- classes correspond to concepts
- documents correspond to individuals in the domain of interpretation
- transitive NT/BT relations are translated into logical subsumption ($\sqsubseteq$)
- RT and non-transitive NT/BT relations correspond to DL roles

Specifically, assuming all NT/BT to be transitive, the scheme in Fig. 3 can be codified with the following TBox and ABox:

TBox	**ABox**
university $\sqsubseteq$ organization	italy(d1)
research-center $\sqsubseteq$ organization	trento(d2)
university-of-trento $\sqsubseteq$ university	fbk(d3)
fbk $\sqsubseteq$ research-center	university-of-trento(d4)
italy $\sqsubseteq$ country	
trento $\sqsubseteq$ italy	
university-of-trento $\sqsubseteq$ trento	
university-of-trento $\sqsubseteq \exists$RT.fbk	

As it can be noticed, those elements of the scheme that in a formal descriptive ontology would be codified as individuals (e.g. Trento, see Fig. 2) in formal classification ontologies correspond to concepts (denoting the set of documents about Trento, see Fig. 3).

To publish the classification ontology above we may use SKOS. If we want to publish the scheme only (without the documents) we can use in particular the RDF exchange syntax, for instance as in the fragment below:

```
<skos:Concept rdf:about="#research_center">
   <skos:broaderTransitive rdf:resource="#organization"/>
<skos:Concept>
<skos:Concept rdf:about="#university">
   <skos:broaderTransitive rdf:resource="#organization"/>
</skos:Concept>
<skos:Concept rdf:about="#Trento">
   <skos:broader rdf:resource="#administrative_division"/>
   <skos:broaderTransitive rdf:resource="#Italy"/>
</skos:Concept>
<skos:Concept rdf:about="#FBK"/>
<skos:Concept rdf:about="#University_of_Trento">
   <skos:broaderTransitive rdf:resource="#Trento"/>
   <skos:related rdf:resource="#FBK"/>
</skos:Concept>
```

As it can be noticed, consistently with the TBox above, in SKOS both real world classes and individuals are codified as concepts, or better as instances of *skos:Concept*. However, since there is no distinction between concepts and instances, we cannot represent corresponding documents in SKOS.

Similarly to RDF, there is no support for disjointness in SKOS [Miles and Bechhofer, 2009]. On the other hand, differently from RDF, transitivity can be enforced using *skos:broaderTransitive* or *skos:narrowerTransitive* properties, while for non-transitive *part-of* (e.g. membership or containment) *skos:broader* or *skos:narrower* can be used.

2.1.3 From descriptive to classification ontologies and vice versa

From what discussed in the previous two sections, it should now be clear how the difference in the purpose is reflected in two totally different semantics - in terms of individuals, classes and

relations - and therefore it is obviously not appropriate for instance to integrate a classification ontology with a descriptive ontology (exactly because the semantics is different). However, this does not mean that it cannot be done, but that we rather need to preliminary convert them such that they have the same semantics. If the purpose is to classify, one should codify both schemes into classification ontologies. Conversely, if the goal is to describe a domain, one should codify them into descriptive ontologies.

The conversion from a descriptive to the corresponding classification ontology can be done as follows:

- convert instances into classes
- convert *instance-of*, *is-a* and transitive *part-of* into NT/BT relations
- convert other relations into RT relations

Note that this is in line with what was postulated by Ranganathan [1967] when he says that hierarchies are constructed on the basis of *genus-species* (*is-a* and *instance-of*) and *whole-part* (*part-of*) relations. As it can be noticed, modulo the indexed documents, the classification ontology in Fig. 3 corresponds to the conversion of the descriptive ontology in Fig. 2.

The translation process can be easily automated. However, with the translation we have a clear loss of information. Real world classes and instances collapse into document classes. Similarly, *instance-of*, *is-a* and transitive *part-of* relations become undifferentiated hierarchical relations, while all other relations become associative relations. For this reason it is clear that the opposite conversion cannot be automated, but it strictly requires manual intervention:

- each class has to be mapped to either a real world class or instance
- each NT/BT relation (assuming all of them to be transitive) has to be converted to an *instance-of*, *is-a* or transitive *part-of*
- each RT relation has to be codified into an appropriate real world associative relation

Distributing schemes as descriptive ontologies ensures maximum reusability. In fact, this would directly serve those applications that need to reason on a domain and at the same time it would require a minimum effort to convert them into classification ontologies when needed. When a scheme is available as classification ontology, a significant human effort will be necessary in-

stead to reconstruct its real world version. For instance, a system working with descriptive ontologies is able to respond to the following questions at the same time:

1. Give me all the organizations located in Italy
2. Give me all the lakes in Trento with an altitude greater than 500 m
3. Give me all documents about Italy
4. Give me all documents about countries
5. Give me all documents about the University of Trento

It is clear that in order to serve these diverse applications both descriptive and classification ontologies are needed. However, to minimize maintenance costs it is possible to codify only the real world version of the semantics and efficiently compute at run time, whenever needed, the corresponding classification semantics. For instance, this can be done by computing the corresponding transitive closure (i.e. the set of all possible relations derived because of the transitivity). According to the application, it is possible to specify if *part-of* relations between entities have to be included or not in the closure (i.e. depending on whether we want to enforce their transitivity or not). Of course the time necessary for the computation of the closure heavily depends on the amount of available relations between entities.

2.2. Lightweight ontologies

Classifications are being traditionally used as indexing and browsing structures for books and other bibliographic material in libraries. Nowadays they are used pervasively. On-line business catalogues, Web directories, folders on our personal computers are all examples of classifications. Classifications (in their core part) are tree-like hierarchical structures where the content is described by attaching natural language labels to nodes and where the links between nodes implicitly represent subset relations. For instance, the fact that a node labeled *milk* is put under *cow* typically means that it is meant to contain documents about milk produced by cows, and that this set of documents is a subset of the documents about cows. Though, depending on their target application, a different interpretation can be given to the nodes and links of the classifications [Giunchiglia et al., 2007a].

Giunchiglia et al. [2007a] define a classification as follows:

> **Definition 2: (Classification).** A classification is a rooted tree C = <N, E, L> where N is a finite set of nodes, E is a set of edges on N, and L is a finite set of labels expressed in natural language, such that for any node $n_i \in N$ these is one and only one label $l_i \in L$.

Consider the example in Fig. 4, adopted from [Giunchiglia et al., 2012]. It represents two very simple classifications. White nodes represent categories while black nodes exemplify annotated documents. Solid arrows between nodes represent sub-category relations while dashed arrows denote the fact that a document is categorized into a certain category. Corresponding labels are also given attached to nodes. Initially, we do not know the circumstance in which they were created nor their precise purpose. As humans, we may understand that they were both built to categorize documents about places and, by tagging them, to eventually provide some opinions about those places.

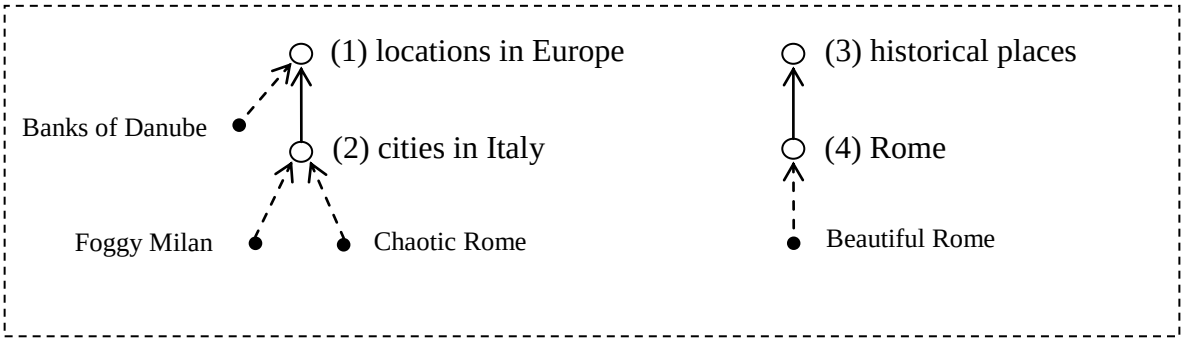


Fig. 4. Example of classifications.

Even if classifications turn out to be very effective in manual tasks, the ambiguity of the labels represents a serious barrier towards the automation of such processes. As a preliminary step towards the automation of these processes, it is therefore fundamental to convert them into formal classification ontologies. For this purpose, Giunchiglia et al. (see for instance [Giunchiglia et al., 2007a], [Giunchiglia and Zaihrayeu, 2008], [Giunchiglia et al, 2009c], [Autayeu, et al., 2010]) in the past recent years developed a series of techniques to formalize the meaning of labels and links in a classification. This conversion procedure associates to each node in the classification a formula in a formal language codifying the meaning of the node in terms of classification semantics. This conversion procedure generates what they called *lightweight ontology* defined as:

> **Definition 3: (Lightweight ontology).** A (formal) lightweight ontology is a triple O = <N, E, C> where N is a finite set of nodes, E is a set of edges on N, such that <N, E> is a rooted tree, and C is a finite set of *concepts* expressed in a formal language F, such that for any node $n_i \in N$, there is one and only one concept $c_i \in C$, and, if n_i is the parent node for n_j, then $c_j \sqsubseteq c_i$.

The formal language F used to encode concepts in C belongs to the family of DL languages and it may differ in its expressive power and reasoning capabilities. However, decades of work in library science and several studies conducted in the context of library classifications show that the expressive power necessary is very low. In fact node labels tend to be noun phrases and it is therefore sufficient to describe them in terms of conjunctions of atomic concepts [Autayeu, et al., 2010] representing intersections of sets of documents. Furthermore, in a recent experiment [Giunchiglia et al, 2009c] the labels of the classifications considered turn out to have a simple translation into propositional DL (DL without roles) with a few *local* disjunctions (around 1% of the overall number of logical connectives) and no negations. The set of concepts C are taken

from some form of background knowledge, for instance from WordNet. In fact, WordNet synsets, grouping words with same meaning, can be approximated to concepts, hypernym and part-meronym relations between synsets can be approximated to subsumption between concepts and where the semantics is the classification semantics.

The conversion of a classification into a lightweight ontology is performed in two steps:

1. For all the labels in the classification compute the *concept at label*
2. For all the nodes in the classification compute the *concepts at node*

With the first step the labels of the nodes are taken in isolation. Using NLP techniques tuned for short noun phrases, such as those in [Zaihrayeu et al., 2007], their meaning is determined by constructing a corresponding formula, called the *concept at label*. However, since the label alone does not provide enough clues for the disambiguation, all possible senses of the words have to be kept. For instance, the concept at label of node 2 in Fig. 4 is (city#1 ⊔ city#2 ⊔ city#3) ⊓ Italy#1, where:

- **city#1**: city, metropolis, urban center -- (a large and densely populated urban area; may include several independent administrative districts; "Ancient Troy was a great city")
- **city#2**: city -- (an incorporated administrative district established by state charter; "the city raised the tax rate")
- **city#3**: city, metropolis -- (people living in a large densely populated municipality; "the city voted for Republicans in 1994")
- **Italy#1**: Italy, Italian Republic, Italia -- (a republic in southern Europe on the Italian Peninsula; was the core of the Roman Republic and the Roman Empire between the 4th century BC and the 5th century AD)

With the second step, each formula is completed by taking into account the relative position of each node in the classification. This is done by taking the conjunction (⊓) of all the formulas along the path from the root to the node and by filtering out the senses which are not compatible each other, i.e. not related by relations in WordNet. This formula is called the *concept at node*. For instance, to determine the concept at node for node 2 in Fig. 4 we need to consider that for the words *location* and *Europe* the following meanings are provided in WordNet:

- **location#1:** location -- (a point or extent in space)
- **location#2:** placement, location, locating, position, positioning, emplacement -- (the act of putting something in a certain place)
- **location#3:** localization, localisation, location, locating, fix -- (a determination of the place where something is; "he got a good fix on the target")
- **location#4:** location -- (a workplace away from a studio at which some or all of a movie may be made; "they shot the film on location in Nevada")

- **Europe#1:** 1. (28) Europe -- (the 2nd smallest continent (actually a vast peninsula of Eurasia); the British use `Europe' to refer to all of the continent except the British Isles)
- **Europe#2:** European Union, EU, European Community, EC, European Economic Community, EEC, Common Market, Europe -- (an international organization of European countries formed after World War II to reduce trade barriers and increase cooperation among its members; "he took Britain into Europe")
- **Europe#3:** Europe -- (the nations of the European continent collectively; "the Marshall Plan helped Europe recover from World War II")

By further observing that in WordNet only the first and second meaning of *city* are related (through a chain of hypernym relations) to the first meaning of *location*, and that the first meaning of *Europe* is related (through part-meronym) to the only sense available for *Italy*, while all the other senses are unrelated, after the sense filtering the concept at node of node 2 is computed as (location#1 ⊓ Europe#1) ⊓ ((city#1 ⊔ city#2) ⊓ Italy#1). The lightweight ontologies generated from the classifications in Fig. 4 are provided in Fig. 5.

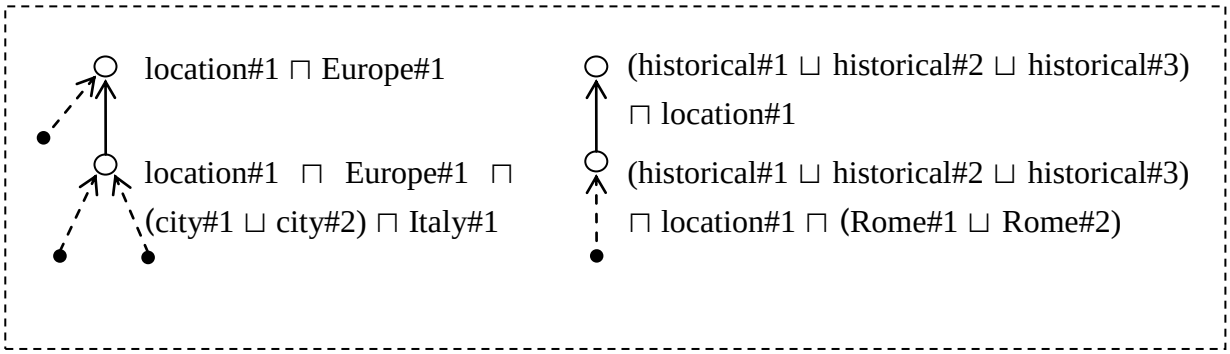


Fig. 5. The classifications in Fig. 4 converted into lightweight ontologies.

The level of accuracy in the translation process highly depends on the accuracy of the NLP techniques used for the translation of the node labels into formal formulas. For instance, in [Autayeu, et al., 2010] the grammars developed to parse labels, constituted by a few rules (from 9 to 17 according to the classification) are able to cover up to 99.81% of the labels under examination and reaches an accuracy of 84.39%.

As described in [Giunchiglia and Zaihrayeu, 2008], lightweight ontologies can be used in many applications including document classification, semantic search, and matching of classifications, for instance for data integration. In all these applications classifications are preliminary translated into lightweight ontologies:

- **Document classification.** Document classification is the problem of assigning a document to one or more nodes in the classification based on the subject of a document, i.e. what the document is about. The approach presented in [Giunchiglia et al., 2007b], is based on the *get-specific* principle according to which a document is classified as deep as possible in the classification. The basic idea is that each document is assigned a formula in the formal language and is automatically classified by reasoning about subsumption on the nodes of the lightweight ontology. Note that this approach does not require the creation of a training dataset which would normally be required in machine learning approaches.
- **Semantic search.** Semantic search, applied to classifications, is the problem of finding those documents in the classification which correspond to a natural language query given in input. In brief, this problem can be approached by determining the concept corresponding to the query and by identifying, as answer to the query, those documents whose concept is more specific or equivalent to the concept of the query. In the approach reported in [Giunchiglia et. al, 2009d], the computational complexity is reduced by running the query on the nodes of the lightweight ontology, thus reducing the search space. The set of documents classified in those nodes are given in output.
- **Semantic matching.** As a preliminary step towards integration and data coordination (interoperability in the broader sense) of heterogeneous repositories, semantic matching between classifications consists in identifying semantic relations between the nodes in the two schemas. In the approach proposed in [Giunchiglia et al., 2007c], and described in detail in the next section, possible semantic relations include disjointness ($\perp$), equivalence ($\equiv$), more specific ($\sqsubseteq$) and less specific ($\sqsupseteq$).

2.3. Semantic matching with S-Match

Ontology matching is a fundamental procedure which aims at establishing a set of correspondences, called *mapping* or *alignment*, between two ontologies in input. Different solutions are offered. A good survey is provided in [Shvaiko and Euzenat, 2007]. In this thesis we focus on schema-based solutions, i.e. those techniques that - to determine the nodes in two ontologies which semantically correspond to each other - exploit schema information only and do not consider instance information. Traditional approaches (e.g. Cupid [Madhavan et al, 2001]) typically compute a coefficient in the [0,1] range for each pair of nodes in the two ontologies. This gives a measure of the matching confidence. Semantic matching tools calculate semantic relations between the nodes instead, thus capturing the meaning codified in their natural language labels.

In this section, we present the main features of the S-Match tool as well as its reached accuracy and run-time performance (in comparison with similar tools) as they are described for instance in [Giunchiglia et al., 2007c]. We conclude the section by outlining open problems with particular emphasis to those addressed in this thesis.

2.3.1 The S-Match algorithm

S-Match has been designed to work on tree-like hierarchical structures, i.e. those schemas that can be translated, exactly or with a certain degree of approximation, to lightweight ontologies. By relying on some form of background knowledge (WordNet in the early versions) it computes a set of correspondences, also called *mapping elements*, between the nodes in the two schemas in input where each correspondence is a tuple <id, n_i, n_j, R> where id is a unique identifier, n_i and n_j are two nodes in the first and second schema, and R is a semantic relation in the set {⊥, ≡, ⊑, ⊒}. The algorithm is organized into four macro steps as follows:

1. For all the labels in the two schemas compute the *concept at label*
2. For all the nodes in the two schemas compute the *concepts at node*
3. For all pairs of labels in the two schemas compute the semantic relations between the concepts at labels
4. For all pairs of nodes in the two schemas compute the relations between the concepts at node

With the first two steps S-Match converts the two schemas into lightweight ontologies. For instance, the two classifications in Fig. 4 are translated in the two lightweight ontologies reported in Fig. 5.

With the third step, the background knowledge is used to determine the semantic relations holding between all the atomic concepts appearing in the concepts at label in the two lightweight ontologies. For, instance, it may contain the fact that city#1 $\sqsubseteq$ location#1. In other words, it allows constructing the local theory T used to draw final conclusions. This step is fundamental to reduce the number of axioms to reason about when computing the semantic relations between nodes in the last step.

During the last step, the problem of matching the two schemas is decomposed into *n* x *m* node to node matching problems, where *n* and *m* are the sizes of the two schemas. For each pair of nodes, the problem of determining the semantic relation holding between them is reduced to an unsatisfiability problem using the local theory T determined at the previous step. More in detail, for each pair of nodes n_i and n_j with corresponding concepts at node c_i and c_j the theory T is used to determine the strongest semantic relation in the partially ordered set $\{\bot, \equiv, \sqsubseteq, \sqsupseteq\}$ such that:

$$T \models c_i \; R \; c_j$$

This is done by constructing the formula $T \rightarrow c_1 \; R \; c_2$ and proving its validity by running a propositional SAT solver on the negation of the formula and by checking for its unsatisfiability. The strongest semantic relation holding between the two nodes is finally returned. This process is optimized by prioritizing the order in which the relations are checked and by taking into account that $\bot$ is stronger than $\equiv$, $\sqsubseteq$ are $\sqsupseteq$ unordered, and that $\equiv$ holds when both $\sqsubseteq$ and $\sqsupseteq$ hold.

Fig. 6 shows the alignment resulting from the application of S-Match on the classifications provided in Fig. 4.

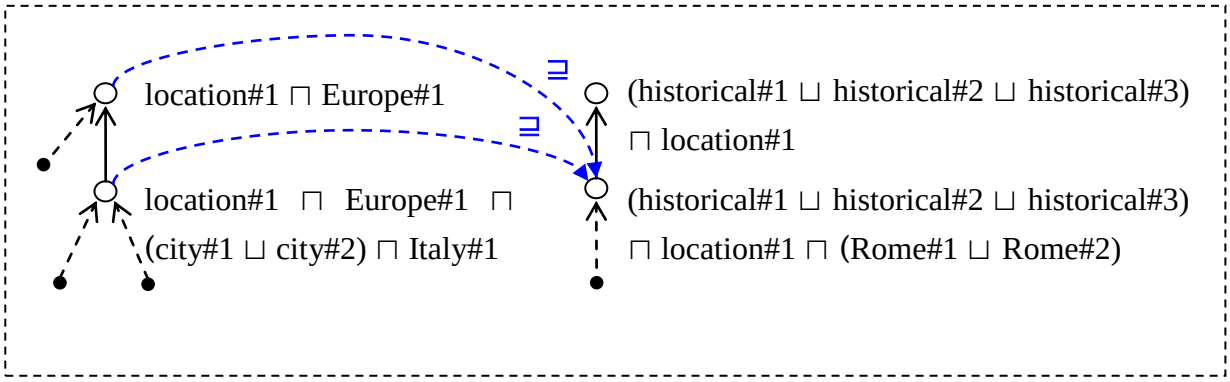


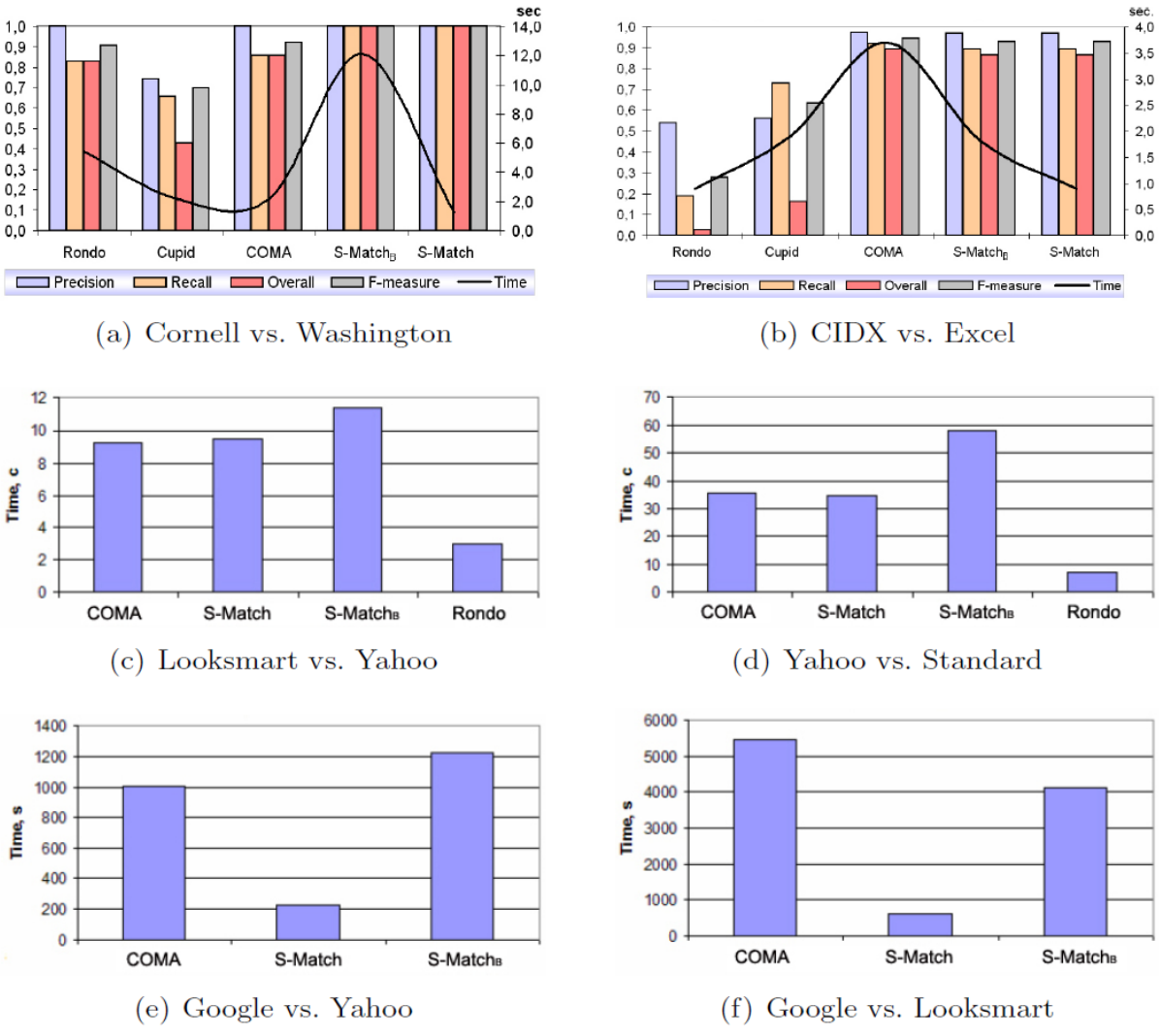
Fig. 6. The alignment between the classifications in Fig. 4 as computed by S-Match.

2.3.2 S-Match evaluation

Among the myriad of approaches to matching that have been proposed, the closest to S-Match include:

- **Rondo** [Melnik et al., 2002] [Melnik et al., 2003] implements the Similarity Flooding (SF) approach based on similarity propagation. The algorithm exploits syntactic techniques at the element and structure level of the schemas, represented as directed labeled graphs. An initial alignment, obtained by a string-based comparison of the labels of the nodes in the two schemas, is progressively refined.
- **Cupid** [Madhavan et al, 2001] implements a hybrid matching algorithm comprising different syntactic techniques at the element (for instance the presence of a common prefix or suffixes) and structure levels (for instance tree matching weighted by leaves). To determine the similarity measures it also exploits external lexical resources.
- **COMA** [Do and Rahm, 2002] implements a composite matching approach which exploits syntactic and external techniques. It provides an extensible library of matching algorithms, a framework which allows combining the results obtained with them and a platform for their evaluation. Most of the available algorithms rely on string-based techniques, such as n-gram and edit distance; others share techniques with Cupid, such as tree matching weighted by leaves; a reuse-oriented matcher tries to reuse previously obtained results for entire new ontologies or for their fragments. However, w.r.t. Cupid, COMA provides a more flexible architecture and allows performing iterations in the matching process.

Fig. 7 (taken from [Shvaiko et al., 2010b]) provides some figures about the evaluation of S-Match, described in detail in [Giunchiglia et al., 2007c]. It provides quality and performance measures of S-Match in comparison with Rondo, Cupid and Coma. In the picture, S-Match$_B$ denotes the basic version of S-Match, while S-Match denotes its optimized version, described in [Giunchiglia et al., 2005]. Quality is measured using the standard precision, recall and F-measure. Performance is measured in the time (milliseconds) required by the algorithm. Because of the difficulty in establishing a gold standard for big datasets, quality is evaluated only on small and medium size schemas (cases (a) and (b)).



(a) Cornell vs. Washington

(b) CIDX vs. Excel

(c) Looksmart vs. Yahoo

(d) Yahoo vs. Standard

(e) Google vs. Yahoo

(f) Google vs. Looksmart

Fig. 7. Evaluation results.

Evaluation results show that S-Match on average is as good as COMA and outperforms other systems in terms of quality indicators and that it is significantly faster than the others, up to 9 times, particularly on big schemas having hundreds of nodes.

2.3.3 Challenges in semantic matching

In a fairly cited paper entitled "Ten challenges in ontology matching", Shvaiko and Euzenat [Shvaiko, and Euzenat, 2008] underline that despite the progress made in the area, a lot of work still has to be done. This difficulty is not unexpected because a solution to the matching problem would amount to solving the semantic heterogeneity problem at the level of schematic metadata (e.g., ontologies). As suggested by the title, the paper focuses on ten challenges in the field and for each of them it provides the corresponding recent advances. Given the purpose of this thesis, we focus on (a) run-time performance, (b) maintenance (referred to as user involvement in the paper) and (c) lack of background knowledge:

a. ***Run-time performance.*** Performance is of prime importance in many dynamic applications, for instance when the output has to be given to a user issuing a query in an interactive environment. Execution time is a clear indicator of system scalability. The fact that during the evaluation of S-Match some systems went out of memory for big tasks, suggests that their performance time is achieved at the price of using a large amount of main memory. The optimized version of S-Match - by leveraging on the characteristic of formulas, and in particular on the fact that for certain and quite frequent cases, e.g. when the formula is in Horn form, satisfiability can be resolved in linear time (while SAT in general may require exponential time) - is able to save a significant amount of time w.r.t. its basic version. However, the number of calls to SAT necessary to compute the mapping remains pretty high. In general, the results of the OAEI initiative [Euzenat et al, 2011] show that some systems still may take hours or even days.

b. ***Maintenance***. It has been already underlined, that automatic tools cannot deliver perfect results. It is therefore essential to rely on semi-automatic approaches where for instance the automatic phase is followed by a manual validation or where the matching process is iterated given some user intervention at the beginning (e.g. by providing an initial set of correspondences) or at regular intermediate steps (e.g. by fixing some of the mistakes in the correspondences returned by the tool). However, current matching tools offer poor support to users for the process of creation, validation and maintenance of the correspondences. Moreover, interactive approaches face soon problems of scalability as the

number of nodes and correspondences grows. As a consequence, handling them turns out to be a very complex, slow and error prone task. This research area is still largely unexplored.

a. ***Lack of background knowledge***. One of the main sources of difficulty for the matching tasks is that ontologies have often a limited coverage. Moreover they are always designed by making some assumptions concerning their applicability that, being implicit, remain unknown to the matching tools. This clearly limits the quality of the results. In fact, as underlined by several studies, for instance in [Lauser et al., 2008] and in [Giunchiglia et al., 2006], the lack of background knowledge is one important cause of low recall. Various strategies have been proposed to attack the problem of the lack of background knowledge. These strategies typically look at some supplementary knowledge for instance by providing it manually in form of additional axioms, by reusing previous match results, by querying the web, by using some domain specific corpus, by using domain specific ontologies or taken from the semantic web. Despite the progress made, these techniques still have to be systematically investigated, combined in a complementary fashion and improved. Nevertheless, recent experiments such as the one described in [Shamdasani et al., 2009], where S-Match is used to align two different vocabularies in the medicine domain using UMLS[12] as background knowledge, prove that - when appropriate domain knowledge is used - precision and recall can be very high.

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

Chapter 3

3. Computing minimal mappings

The work described in this chapter represents an important step ahead to the run-time performance problem of S-Match where a high number of calls to SAT are necessary to compute the mapping between two schemas. Given two classifications, preliminary converted into lightweight ontologies, we compute the *minimal mapping*, namely that subset of all possible correspondences between them such that i) all the others can be computed from them - and are therefore said to be *redundant* - in time linear in the size of the input ontologies, and ii) none of them can be dropped without losing property i).

We provide a formal definition of *minimal* and *redundant mappings*, evidence of the fact that the minimal mapping always exists and it is unique and define a time efficient correct and complete algorithm for their computation which minimizes the number of comparisons between the nodes of the two input ontologies. The experimental results show a substantial improvement both in the computation time, number of calls to SAT saved and in the number of correspondences which need to be handled.

The main advantage of minimal mappings is that they are the minimal amount of information that needs to be dealt with. Notice that this is a rather important feature as the number of possible correspondences can grow up to n x m with n and m being the size of the two input ontologies. In particular, minimal mappings become crucial with large ontologies, e.g., DMOZ, with 10^5 - 10^6 nodes, where even relatively small subsets of the number of possible correspondences (10^{12}) are unmanageable. As proven by work like [Falconer and Storey, 2007] and [Robertson et al., 2005], many systems and corresponding interfaces, mostly graphical, have been provided for the management of mappings but all of them hardly scale with the increasing number of nodes, and the resulting visualizations are rather messy. In this respect, minimal mappings provide clear usability advantages. Among other things, the maintenance of smaller sets makes the work of the user much easier, faster and less error prone [Meilicke et al, 2008].

As far as we know very little work has been done on the problem of computing minimal mappings. In general the computation of minimal mappings can be seen as a specific instance of the mapping inference problem [Madhavan et al, 2002]. Closer to our work, in [Stuckenschmidt et al, 2006] [Meilicke et al, 2006] [Meilicke et al, 2008] the authors use Distributed Description Logics (DDL) [Borgida and Serafini, 2003] to represent and reason about existing ontology mappings. They introduce a few debugging heuristics which remove correspondences which are redundant or generate inconsistencies from a given set [Meilicke et al, 2006]. The main problem of this approach, as also recognized by the authors, is the complexity of DDL reasoning [Meilicke et al, 2008]. In our approach, by restricting to lightweight ontologies and instead of pruning redundant correspondences, we directly compute the minimal mapping. Among other things, our approach allows us to minimize the number of calls to the node matching functions, i.e. those functions that decide about the existence of a semantic relation between two nodes, thus minimizing the number of calls to SAT.

The rest of the chapter is organized as follows. Section 3.1 provides a motivating example. Section 3.2 provides the definition for redundant and minimal mappings, and it shows that the minimal mapping always exists and it is unique. Section 3.3 describes the algorithms for the computation of the minimal and the mapping of maximum size, i.e. the mapping containing the maximum number of minimal and redundant correspondences. Section 3.4 provides the results of the evaluation. Section 3.5 describes a variant of the algorithm where nodes are associated attribute name-value pairs.

3.1. Motivating example

Consider the two fragments of classifications depicted in Fig. 8. They are designed to arrange more or less the same content, but from different perspectives. The second is a fragment taken from the Yahoo web directory[13] (category Computers and Internet).

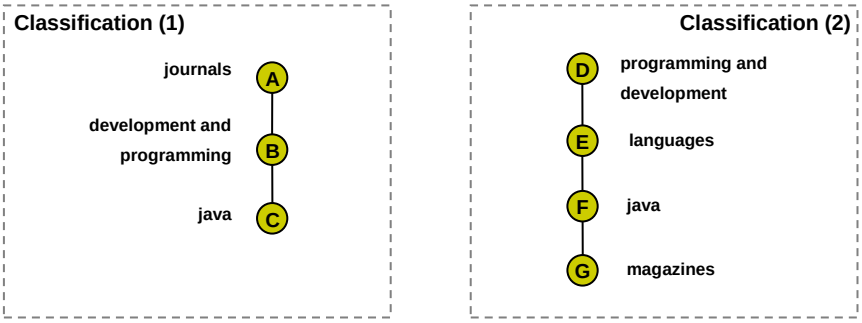


Fig. 8. Two classifications

With the conversion of the classifications into lightweight ontologies, each node label can be translated in an unambiguous, propositional DL formula. The resulting formulas are reported in Fig. 9. Here each string denotes a concept (e.g., journals#1) and the numbers at the end of the strings denote a specific concept constructed from a WordNet sense. Notice that the formula associated to each node contains the formula of the node above to capture the fact that the meaning of each node is contextualized by the meaning of its ancestor nodes. As a consequence, the backbone structure of the resulting lightweight ontologies is represented by subsumption relations between nodes.

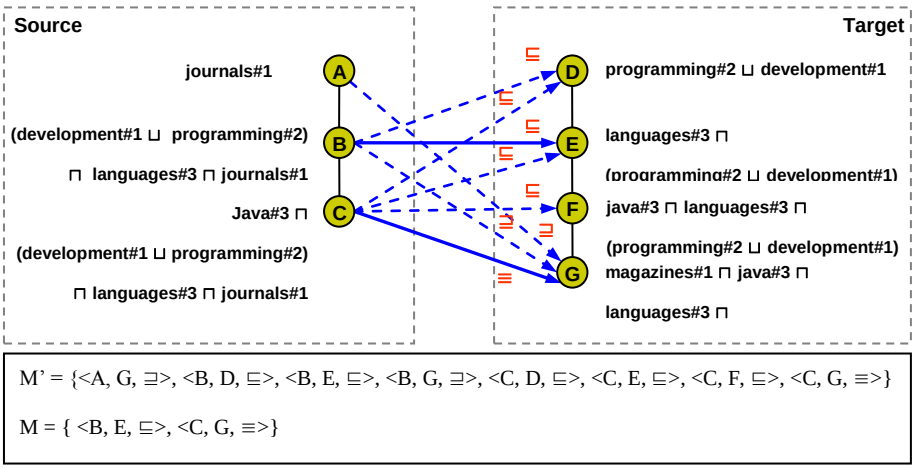


Fig. 9. The minimal and redundant mapping between two lightweight ontologies

[13]http://dir.yahoo.com/

Fig. 9 also reports the correspondences computed by S-Match. Notice however that not all the correspondences have the same semantic valence. For instance, B⊑D is a trivial consequence of B⊑E and E⊑D, and similarly C⊑F is a consequence of C≡G. We represent redundant correspondences using dashed lines; the other correspondences, represented with solid lines, constitute the minimal mapping. We denote with M' the set of maximum size, i.e. the set including the maximum number of minimal and redundant correspondences, and with M the minimal mapping. The problem is how to compute the minimal set in the most efficent way.

3.2. Redundant and minimal mappings

The notions of minimal and redundant mappings are based on the notion of correspondence between two lightweight ontologies. Following the terminology used for S-Match, correspondences are called *mapping elements* as in the definition above:

Definition 4 (Mapping element). Given two lightweight ontologies O_1 and O_2, a mapping element m between them is a triple $<n_1, n_2, R>$, where:

a) $n_1 \in N_1$ is a node in O_1, called the source node;
b) $n_2 \in N_2$ is a node in O_2, called the target node;
c) $R \in \{\perp, \equiv, \sqsubseteq, \sqsupseteq\}$ is the strongest semantic relation holding between n_1 and n_2.

Relations are given in a partial order. The partial order is such that disjointness is stronger than equivalence which, in turn, is stronger than subsumption (in both directions), and such that the two subsumption symbols are unordered. This in order to return subsumption only when equivalence does not hold or one of the two nodes being inconsistent (this latter case generating at the same time both a disjointness and a subsumption relation), and similarly for the order between disjointness and equivalence. Notice that, under this ordering, there can be at most one mapping element between two nodes.

The next step is to define the notion of redundancy. The key idea is that, given a mapping element $m = <n_1, n_2, R>$, a new mapping element $m' = <n_1', n_2', R'>$ is redundant with respect to the first if the existence of the second can be asserted simply by looking at the relative positions of n_1 with n_1', and n_2 with n_2'. In algorithmic terms, this means that the second can be computed without running the time expensive node matching functions. We have identified four basic redundancy patterns as follows:

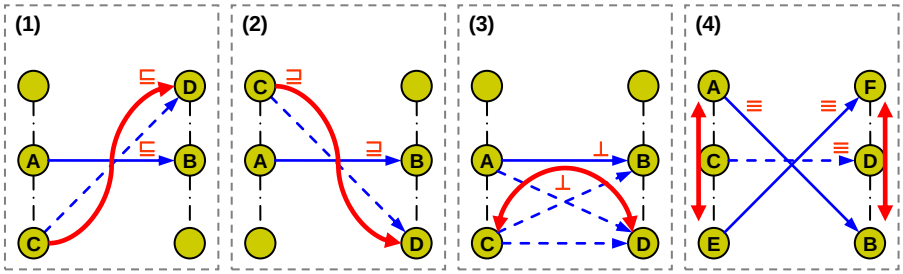


Fig. 10. Redundancy detection patterns

In Fig. 10, the blue dashed mapping elements are redundant w.r.t. the solid blue ones. The bold red solid lines show how a semantic relation propagates. Let us discuss the rationale for each of the patterns:

- **Pattern (1)**: each mapping element of the kind <C, D, ⊑> is redundant w.r.t. <A, B, ⊑>. In fact, C is more specific than A which is more specific than B which is more specific than D. As a consequence, by transitivity C is more specific than D.
- **Pattern (2)**: dual argument as in pattern (1).
- **Pattern (3)**: each mapping element of the kind <C, D, ⊥> is redundant w.r.t. <A, B, ⊥>. In fact, we know that A and B are disjoint, that C is more specific than A and that D is more specific than B. This implies that C and D are also disjoint.
- **Pattern (4)**: Pattern 4 is the combinations of patterns (1) and (2).

Notice that patterns (1) and (2) are still valid in case we substitute subsumption with equivalence. However, in this case we cannot exclude the possibility that a stronger relation holds between C and D. A trivial example of where this is not the case is provided in Fig. 11.

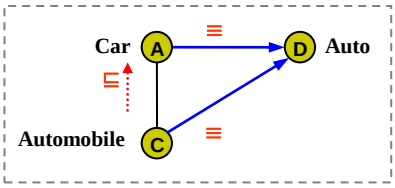


Fig. 11. Two non redundant mappings

On the basis of the patterns and the considerations above we can define redundant elements as follows. Here path(n) is the path from the root to the node n.

Definition 5 (Redundant mapping element). Given two lightweight ontologies O_1 and O_2, a mapping M and a mapping element m'∈M with m' = <C, D, R'> between them, we say that m' is redundant in M iff the following holds:

(1) If R' is ⊑, ∃m∈M with m = <A, B, R> and m ≠ m' such that R ∈ {⊑, ≡}, A ∈ path(C) and D ∈ path(B);
(2) If R' is ⊒, ∃m∈M with m = <A, B, R> and m ≠ m' such that R ∈ {⊒, ≡}, C ∈ path(A) and B ∈ path(D);
(3) If R' is ⊥, ∃m∈M with m = <A, B, ⊥> and m ≠ m' such that A ∈ path(C) and B ∈ path(D);
(4) If R' is ≡, conditions (1) and (2) must be satisfied.

See how Definition 5 maps to the four patterns in Fig. 10. Fig. 9 given in the previous section provides examples of redundant elements. Definition 5 can be proved to capture all and only the cases of logical redundancy of a mapping element w.r.t. another one.

Theorem 1 (Redundancy, soundness and completeness). Given a mapping M between two lightweight ontologies O_1 and O_2, a mapping element m' ∈ M is logically redundant w.r.t. another mapping element m if and only if it satisfies one of the conditions of Definition 5.

The soundness argument is the rationale described for the patterns above. Completeness can be shown by constructing the counterargument that we cannot have logical redundancy in the remaining cases. We can proceed by enumeration, negating each of the patterns, encoded one by one in the conditions appearing in the Definition 5. The complete proof is given in Appendix.

Fig. 12 provides some examples of non redundancy. The first, based on pattern (1), tells us that the existence of a correspondence between two nodes does not necessarily propagate to the two nodes below. For example we cannot derive that Canine ⊑ Dog from the set of axioms {Canine ⊑ Mammal, Mammal ⊑ Animal, Dog ⊑ Animal}, and it would be wrong to do so. The second, based on pattern (3), shows that disjointness cannot propagate to the target (or to the source) one level up. For example we cannot derive that Dog ⊥ Animal only from {Dog ⊑ Canine, Cat ⊑ Animal, Canine ⊥ Cat}. The third example, based on pattern (4), tells us that we cannot derive equivalence if the source node C or target D is not between the source and target nodes of the

two equivalence mapping elements A ≡ B and E ≡ F. Notice that, by chance, the other equivalence mapping holds.

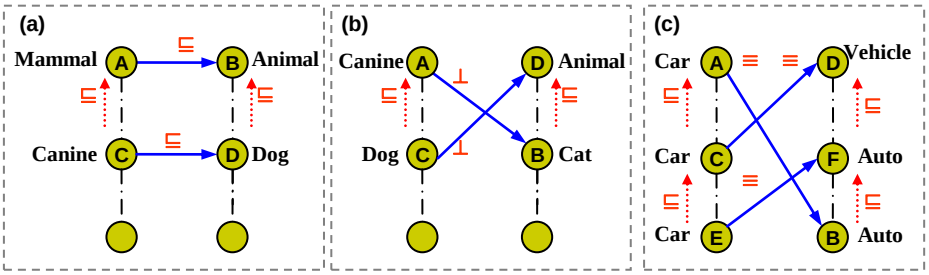


Fig. 12. Some examples of non redundant mapping elements

The notion of redundancy allows us to formalize the notion of minimal mapping as follows:

Definition 6 (Minimal mapping). Given two lightweight ontologies O_1 and O_2, we say that a mapping M between them is minimal iff:

1) $\nexists m \in M$ such that m is redundant (minimality condition);
2) $\nexists M' \supset M$ satisfying condition a) above (maximality condition).

A mapping element is said to be *minimal* if it belongs to the minimal mapping.

Note that conditions (a) and (b) ensure that the minimal mapping is the set of mapping elements of maximum size among those with no redundant elements. As an example, the set M in Fig. 9 is minimal. Comparing this mapping with M' we can observe that all elements in the complement set M' - M are redundant and that, therefore, there are no other supersets of M with the same properties. In effect, <A, G, ⊒> and <B, G, ⊒> are redundant w.r.t. <C, G, ≡> for pattern (2); the mapping elements <C, D, ⊑>, <C, E, ⊑> and <C, F, ⊑> are redundant w.r.t. <C, G, ≡> for pattern (1); <B, D, ⊑> is redundant w.r.t. <B, E, ⊑> for pattern (1). Note that M contains far less mapping elements w.r.t. M'.

As last observation, for any two given lightweight ontologies, the minimal mapping always exists and it is unique. This is stated by the theorem below. A proof is given in the Appendix.

Theorem 2 (Minimal mapping, existence and uniqueness). Given two lightweight ontologies O_1 and O_2, there is always one and only one minimal mapping between them.

3.3. Computing the minimal and redundant mappings

The patterns described in the previous section suggest how to significantly reduce the amount of calls to the node matching functions. By looking for instance at pattern (2) in Fig. 10, given a mapping element m = <A, B, ⊒> we know that it is not necessary to compute the semantic relation holding between A and any descendant C in the sub-tree of B since we know in advance that it is ⊒. At the top level the algorithm is organized as follows:

- **Step 1, computing the minimal mapping modulo equivalence**: compute the set of disjointness and subsumption mapping elements which are minimal modulo equivalence. By this we mean that they are minimal modulo collapsing, whenever possible, two subsumption relations of opposite direction into a single equivalence mapping element;
- **Step 2, computing the minimal mapping**: eliminate the redundant subsumption mapping elements. In particular, collapse all the pairs of subsumption elements (of opposite direction) between the same two nodes into a single equivalence element. This will result into the minimal mapping;
- **Step 3, computing the mapping of maximum size**: Compute the mapping of maximum size (including the maximum amount of minimal and redundant mapping elements). During this step the existence of a (redundant) element is computed as the result of the propagation of the elements in the minimal mapping.

The first two steps are performed at matching time, while the third is activated whenever the user wants to exploit the pre-computed mapping elements for instance for their visualization or data integration. The following three subsections analyze the three steps above in detail.

3.3.1 Step 1: Computing the minimal mapping modulo equivalence

The minimal mapping is computed by a function **TreeMatch** whose pseudo-code is provided in Fig. 13. M is the minimal set of correspondences while T1 and T2 are the input lightweight ontologies. **TreeMatch** is called on the root nodes of T1 and T2. It is crucially dependent on the node matching functions **NodeDisjoint** (Fig. 14) and **NodeSubsumedBy** (Fig. 15) which take two nodes n1 and n2 and return a positive answer in case of disjointness or subsumption between the corresponding formulas, or a negative answer if it is not the case or they are not able to establish it. Notice that these two functions hide the heaviest computational costs; in particular their

computation time is exponential when the relation holds and, exponential in the worst case, but possibly much faster, when the relation does not hold. The main motivation for this is that the node matching problem, in the general case, should be translated into disjointness or subsumption problem in propositional DL and thus resolved through a call to a SAT solver.

```
10  node: struct of {cnode: wff; children: node[];}
20  T1,T2: tree of (node);
30  relation in {⊑, ⊒, ≡, ⊥};
40  element: struct of {source: node; target: node; rel: relation;};
50  M: list of (element);
60  boolean direction;

70  function TreeMatch(tree T1, tree T2)
80   {TreeDisjoint(root(T1),root(T2));
90    direction := true;
100   TreeSubsumedBy(root(T1),root(T2));
110   direction := false;
120   TreeSubsumedBy(root(T2),root(T1));
130   TreeEquiv();
140  };
```

Fig. 13. Pseudo-code for the tree matching function

The goal, therefore, is to compute the minimal mapping by minimizing the calls to the node matching functions and, in particular minimizing the calls where the relation will turn out to hold. We achieve this purpose by processing both trees top down. To maximize the performance of the system, **TreeMatch** has therefore been built as the sequence of three function calls: the first call to **TreeDisjoint** (line 80) computes the minimal set of disjointness mapping elements, while the second and the third call to **TreeSubsumedBy** compute the minimal set of subsumption mapping elements in the two directions modulo equivalence (lines 90-120). Notice that in the second call, **TreeSubsumedBy** is called with the input ontologies with swapped roles. These three calls correspond to Step 1 above. Line 130 in the pseudo code of **TreeMatch** implements Step 2 and it will be described in the next subsection.

```
10  function TreeDisjoint(node n1, node n2)
20   {c1: node;
30    NodeTreeDisjoint(n1, n2);
40    foreach c1 in GetChildren(n1) do TreeDisjoint(c1,n2);
50   };

60  function NodeTreeDisjoint(node n1, node n2)
70   {n,c2: node;
80    foreach n in Path(Parent(n1)) do if (<n,n2,⊥> ∈ M) then return;
90    if (NodeDisjoint(n1, n2)) then
100     {AddMappingElement(<n1,n2,⊥>);
110      return;
120     };
130   foreach c2 in GetChildren(n2) do NodeTreeDisjoint(n1,c2);
140  };

150 function boolean NodeDisjoint(node n1, node n2)
160  {if (Unsatisfiable(mkConjunction(n1.cnode,n2.cnode))) then
        return true;
170   else return false; };
```

Fig. 14. Pseudo-code for the **TreeDisjoint** function

TreeDisjoint (in Fig. 14) is a recursive function which finds all disjointness minimal elements between the two sub-trees rooted in n1 and n2. Following the definition of redundancy, it basically searches for the first disjointness element along any pair of paths in the two input trees. Exploiting the nested recursion of **NodeTreeDisjoint** inside **TreeDisjoint**, for any node n1 in T1 (traversed top down, depth first) **NodeTreeDisjoint** visits all of T2, again top down, depth first. **NodeTreeDisjoint** (called at line 30, starting at line 60) keeps fixed the source node n1 and iterates on the whole target sub-tree below n2 till, for each path, the highest disjointness element, if any, is found. Any such disjoint element is added only if minimal (lines 90-120). The condition at line 80 is necessary and sufficient for redundancy. The idea here is to exploit the fact that any two nodes below two nodes involved in a disjointness mapping element are part of a redundant element and, therefore, it is appropriate to stop the recursion thus saving a lot of time expensive calls (*n***m* calls with *n* and *m* the number of the nodes in the two trees). Notice that this check needs to be performed on the full path. **NodeDisjoint** checks whether the formula obtained by

the conjunction of the formulas associated to the nodes n1 and n2 is unsatisfiable (lines 150-170).

```
10  function boolean TreeSubsumedBy(node n1, node n2)
20   {c1,c2: node; LastNodeFound: boolean;
30    if (<n1,n2,⊥> ∈ M) then return false;
40    if (!NodeSubsumedBy(n1, n2)) then
50      foreach c1 in GetChildren(n1) do TreeSubsumedBy(c1,n2);
60    else
70      {LastNodeFound := false;
80       foreach c2 in GetChildren(n2) do
90         if (TreeSubsumedBy(n1,c2)) then LastNodeFound := true;
100      if (!LastNodeFound) then AddSubsumptionMappingElement(n1,n2);
120      return true;
140     };
150   return false;
160  };

170 function boolean NodeSubsumedBy(node n1, node n2)
180  {if (Unsatisfiable(mkConjunction(n1.cnode,negate(n2.cnode)))) then
        return true;
190   else return false; };

200 function AddSubsumptionMappingElement(node n1, node n2)
210  {if (direction) then AddMappingElement(<n1,n2,⊑>);
220   else AddMappingElement(<n2,n1,⊒>); };
```

Fig. 15. Pseudo-code for the **TreeSubsumedBy** function

TreeSubsumedBy (in Fig. 15) recursively finds all minimal mapping elements where the strongest relation between the nodes is ⊑ (or dually, ⊒ in the second call; in the following we will concentrate only on the first call). Notice that **TreeSubsumedBy** assumes that the minimal disjointness elements are already computed. As a consequence, at line 30 it checks whether the mapping element between the nodes n1 and n2 is already in the minimal set. If this is the case it stops the recursion. This allows computing the stronger disjointness relation rather than subsumption when both hold (namely in presence of an inconsistent node). Given n2, lines 40-50

implement a depth first recursion in the first tree till a subsumption is found. The test for subsumption is performed by function **NodeSubsumedBy** that checks whether the formula obtained by the conjunction of the formulas associated to the node n1 and the negation of the formula for n2 is unsatisfiable (lines 170-190). Lines 60-140 implement what happens after the first subsumption is found. The key idea is that, after finding the first subsumption, **TreeSubsumedBy** keeps recursing down the second tree till it finds the last subsumption. When this happens, the mapping element is added to the minimal mapping (line 100). Notice that both **NodeDisjoint** and **NodeSubsumedBy** call the function **Unsatisfiable** which embeds a call to a SAT solver.

To fully understand **TreeSubsumedBy,** the reader should check what happens in the four situations in Fig. 16. In case (a) the first iteration of the **TreeSubsumedBy** finds a subsumption between A and C. Since C has no children, it skips lines 80-90 and directly adds the mapping element <A, C, ⊑> to the minimal set (line 100). In case (b), since there is a child D of C the algorithm iterates on the pair A-D (lines 80-90) finding a subsumption between them. Since there are no other nodes under D, it adds the mapping element <A, D, ⊑> to the minimal set and returns true. Therefore LastNodeFound is set to true (line 90) and the mapping element between the pair A-C is recognized as redundant. Case (c) is similar. The difference is that **TreeSubsumedBy** will return false when checking the pair A-D (line 30), thanks to previous computation of minimal disjointness mapping elements, and therefore the mapping element <A, C, ⊑> is recognized as minimal. In case (d) the algorithm iterates after the second subsumption mapping element is identified. It first checks the pair A-C and iterates on A-D concluding that subsumption does not hold between them (line 40). Therefore, it recursively calls **TreeSubsumedBy** between B and D. In fact, since the mapping element <A, C, ⊑> will be recognized as minimal, it is not worth checking <B, C, ⊑> for pattern (1). As a consequence <B, D, ⊑> is recognized as minimal together with <A, C, ⊑>.

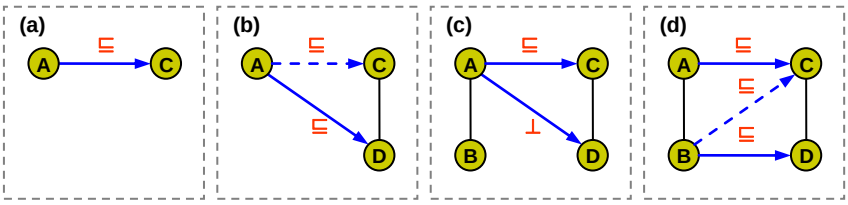


Fig. 16. Examples of applications of the **TreeSubsumedBy**

Five observations. The first is that, even if, overall, **TreeMatch** implements three loops instead of one, the wasted (linear) time is largely counterbalanced by the exponential time saved by avoiding a lot of useless calls to the SAT solver. The second is that, when the input trees T1 and T2 are two nodes, **TreeMatch** behaves as a node matching function which returns the semantic relation holding between the input nodes. The third is that the call to **TreeDisjoint** before the two calls to **TreeSubsumedBy** allows us to implement the partial order on relations defined in the previous section. In particular it allows returning only a disjointness mapping element when both disjointness and subsumption hold. The fourth is that, in the body of **TreeDisjoint**, the fact that the two sub-trees where disjointness holds are skipped is what allows not only implementing the partial order (see the previous observation) but also saving a lot of useless calls to the node matching functions (line 2). The fifth and last observation is that the implementation of **TreeMatch** crucially depends on the fact that the minimal elements of the two directions of subsumption and disjointness can be computed independently (modulo inconsistencies).

3.3.2 Step 2: Computing the minimal mapping

The output of Step 1 is the set of all disjointness and subsumption mapping elements which are minimal modulo equivalence. The final step towards computing the minimal mapping is that of collapsing any two subsumption relations, in the two directions, holding between the same two nodes into a single equivalence relation. The tricky part here is that equivalence is in the minimal set only if both subsumptions are in the minimal set. We have three possible situations:

1) None of the two subsumptions is minimal (in the sense that it has not been computed as minimal in Step 1): nothing changes and neither subsumption nor equivalence is memorized as minimal;
2) Only one of the two subsumptions is minimal while the other is not minimal (again according to Step 1): this case is solved by keeping only the subsumption mapping as minimal. Of course, during Step 3 (see below) the necessary computations will have to be done in order to show to the user the existence of an equivalence relation between the two nodes;
3) Both subsumptions are minimal (according to Step 1): in this case the two subsumptions can be deleted and substituted with a single equivalence element.

Notice that Step 3 can be computed very easily in time linear with the number of mapping elements output of Step 1: it is sufficient to check for all the subsumption elements of opposite direction between the same two nodes and to substitute them with an equivalence element. This is performed by function **TreeEquiv** in Fig. 13.

3.3.3 Step 3: Computing the mapping of maximum size

For brevity we concentrate on the following problem: given two lightweight ontologies T1 and T2 and the of minimal mapping M compute the mapping element between two nodes n1 in T1 and n2 in T2 or the fact that no element can be computed given the current available background knowledge. Corresponding pseudo-code is given in Fig. 17. **ComputeMappingElement** is structurally very similar to the NodeMatch function used by S-Match and described in [Giunchiglia et al., 2007c], modulo the key difference that no calls to SAT are needed. **ComputeMappingElement** always returns the mapping element with strongest relation between the two nodes. The test for redundancy performed by **IsRedundant** reflects the definition of redundancy (Definition 5). For sake of simplicity, we provide below only the code which does the check for the first pattern; the others are analogous. Given for example a mapping element <n1, n2, ⊑>, condition 1 is verified by checking whether in M there is an element <c1, c2, ⊑> or <c1, c2, ≡> with c1 ancestor of n1 and c2 descendant of n2. Notice that **ComputeMappingElement** calls **IsRedundant** at most three times and, therefore, its computation time is linear with the number of mapping elements in M.

```
10  function mapping ComputeMappingElement(node n1, node n2)
20   {isLG, isMG: boolean;
30    if ((<n1,n2,⊥> ∈ M) || IsRedundant(<n1,n2,⊥>)) then
        return <n1,n2,⊥>;
40    if (<n1,n2,≡> ∈ M) then return <n1,n2,≡>;
50    if ((<n1,n2,⊑> ∈ M) || IsRedundant(<n1,n2,⊑>)) then isLG := true;
60    if ((<n1,n2,⊒> ∈ M) || IsRedundant(<n1,n2,⊒>)) then isMG := true;
70    if (isLG && isMG) then return <n1,n2,≡>;
80    if (isLG) then return <n1,n2,⊑>;
90    if (isMG) then return <n1,n2,⊒>;
100   return NULL;
110  };

120 function boolean IsRedundant(mapping <n1,n2,R>)
130  {switch (R)
140    {case ⊑: if (VerifyCondition1(n1,n2)) then return true; break;
150     case ⊒: if (VerifyCondition2(n1,n2)) then return true; break;
160     case ⊥: if (VerifyCondition3(n1,n2)) then return true; break;
170     case ≡: if (VerifyCondition1(n1,n2) &&
                    VerifyCondition2(n1,n2)) then return true;
180    };
190   return false;
200  };

210 function boolean VerifyCondition1(node n1, node n2)
220  {c1,c2: node;
230   foreach c1 in Path(n1) do
240     foreach c2 in SubTree(n2) do
250       if ((<c1,c2,⊑> ∈ M) || (<c1,c2,≡> ∈ M)) then return true;
260   return false;
270  };
```

Fig. 17. Pseudo-code to compute a mapping element

3.4. Evaluation

The algorithm presented in the Section 3.3, that we called MinSMatch, has been implemented by taking the node matching routines of S-Match and by changing the way the tree structure is matched. The evaluation has been performed by directly comparing the results of MinSMatch and S-Match on several real-world datasets. All tests have been performed on a Pentium D 3.40GHz with 2G of RAM running Windows XP SP3 operating system with no additional applications running except the matching system. Both systems were limited to allocating no more than 1GB of RAM. The tuning parameters of the matchers were set to the default values. The selected datasets had been already used in [Avesani et al., 2005]. Some of these datasets can be also found at the OAEI website. The first two datasets describe courses and will be called **C**ornell and **W**ashington, respectively. The second two come from the arts domain and will be referred to as **T**opia and **I**con, respectively. The third two datasets have been extracted from the Looksmart, Google and Yahoo! directories and will be referred to as **S**ource and **T**arget. The fourth two datasets contain portions of the two business directories eCl@ss[14] and UNSPSC[15] and will be referred to as **E**class and **U**nspsc.

Table 1 describes some indicators of the complexity of these datasets.

#	Dataset pair	Node count	Max depth	Average branching factor
1	Cornell/Washington	34/39	3/3	5.50/4.75
2	Topia/Icon	542/999	2/9	8.19/3.66
3	Source/Target	2857/6628	11/15	2.04/1.94
4	Eclass/Unspsc	3358/5293	4/4	3.18/9.09

Table 1. Complexity of the datasets

Consider Table 2. The reduction in the last column is calculated as `(1-m/t)`, where `m` is the number of elements in the minimal set and `t` is the total number of elements in the mapping of maximum size, as computed by MinSMatch. As it can be easily noticed, we have a significant reduction, in the range 68-96%.

[14] http://www.eclass-online.com/
[15] http://www.unspsc.org/

	S-Match	MinSMatch		
#	Total mapping elements (t)	Total mapping elements (t)	Minimal mapping elements (m)	Reduction, %
1	223	223	36	83.86
2	5491	5491	243	95.57
3	282638	282648	30956	89.05
4	39590	39818	12754	67.97

Table 2. Mapping sizes.

The second interesting observation is that in Table 2, in the last two experiments, the number of total mapping elements computed by MinSMatch is slightly higher (compare the second and the third column). This is due to the fact that in the presence of one of the patterns, MinSMatch directly infers the existence of a mapping element without testing it. This allows MinSMacth, differently from S-Match, to avoid missing elements because of failures of the node matching functions because of lack of background knowledge [Giunchiglia et al., 2006]. One such example from our experiments is reported below (directories Source and Target):

\Top\Computers\Internet\Broadcasting\Video Shows
\Top\Computing\Internet\Fun & Games\Audio & Video\Movies

We have a minimal mapping element which states that Video Shows ⊒ Movies. The element generated by this minimal one, which is captured by MinSMatch and missed by S-Match (because of the lack of background knowledge about the relation between 'Broadcasting' and 'Movies') states that Broadcasting ⊒ Movies.

To conclude our analysis, Table 3 shows the reduction in computation time (computed by running S-Match and MinSMatch on the same machine with same settings) and calls to SAT. As it can be noticed the time reductions are substantial, in the range 16-59%, but where the smallest savings are for very small ontologies.

	Run Time, ms			SAT calls		
#	S-Match	MinSMatch	Reduction,%	S-Match	MinSMatch	Reduction,%
1	472	397	15.88	3978	2273	42.86
2	141040	67125	52.40	1624374	616371	62.05
3	3593058	1847252	48.58	56808588	19246095	66.12
4	6440952	2642064	58.98	53321682	17961866	66.31

Table 3. Run time and SAT problems

MinSMatch, together with S-Match are part of an open source framework that can be freely downloaded from http://semanticmatching.org/. At present, the tools have been downloaded more than 3000 times.

3.5. Attribute-based minimal mapping

In this section we present a variant of the minimal mapping algorithm which works on light-weight ontologies where each node is associated a set of attribute name-value pairs.

Each attribute name and value is taken from a formal language. As from [Giunchiglia et al, 2012], the expressive power of the language is that of propositional DL with only conjunctions, no negations and no disjunctions. The expressive power we exploit is very low. Still, decades of work in library science and several studies conducted in the context of library classifications show that it is sufficient to describe their labels in terms of conjunctions of atomic concepts [Autayeu et al., 2010] representing intersections of sets of documents (classification semantics). Furthermore, in an experiment [Giunchiglia et al., 2009c] the labels of the classifications considered turn out to have a simple translation into propositional DL with a few "local" disjunctions (around 1% of the overall number of logical connectives) and no negations. In order to further simplify reasoning we also decided to drop the computation of disjointness and therefore compute more specific, less specific and equivalence (when both less and more specific hold) only. As described in this section, this allows computing the semantic relations between nodes without running the time expensive SAT.

3.5.1 Preliminary notions

For the attribute-based algorithm, we need these additional notions to be defined:

> **Definition 7 (attribute).** An attribute *a* is a pair *<an, av>* where *an* is the attribute name and *av* is the attribute value. an is an atomic formula and av is a arbitrary well-formed formula in a propositional DL language with only conjunctions;

An example of attribute is therefore <Location, Italy ⊓ France>.

> **Definition 8 (subsumption between attributes).** An attribute $a_1 = <an_1, av_1>$ is more specific that $a_2 = <an_2, av_2>$ (in symbols $a_1 \sqsubseteq a_2$) iff $an_1 \sqsubseteq an_2$ and $av_1 \sqsubseteq av_2$.

For example, given $\mathbf{a_1}$ = <City, Rome> and $\mathbf{a_2}$ = <Location, Italy>, then we have $\mathbf{a_1} \sqsubseteq \mathbf{a_2}$ (by assuming City more specific than Location and Rome more specific than Italy).

Definition 9 (subsumption between sets of attributes). A set of attributes $\{a\}_i$ is more specific than a set of attributes $\{a\}_j$, in symbols $\{a\}_i \sqsubseteq \{a\}_j$, iff $\forall\ a_2 \in \{a\}_j\ \exists\ a_1 \in \{a\}_i$ such that $a_1 \sqsubseteq a_2$;

For example, given **a** = {<City, Rome>, <Topic, Dog>} and **b** = {<Location, Italy >}, **a** ⊑ **b**. In fact, <City, Rome> ⊑ <Location, Italy>}.

Notice that the following principle holds:

Principle 1: $\mathbf{av_1} \sqsubseteq \mathbf{av_2}$ iff ∀ ***c*** atomic concept in $\mathbf{av_1}$ ∃ ***d*** atomic concept in $\mathbf{av_2}$ such that ***c*** ⊑ ***d***.

This principle is important since it allows determining the subsumption between two attributes (sets of attributes) simply by checking pairwise subsumptions between concepts. By pre-computing the semantic closure of the knowledge base it is therefore conclude subsumption *without running the time expensive SAT*, but by simply verifying whether the relation is in the closure[16].

We then define an attribute-based lightweight ontology as follows:

Definition 10 (attribute-based lightweight ontology). An attribute-based lightweight ontology O is a rooted tree <N, E, A> where N is a finite set of nodes, E is a set of edges on N, A is a set of attributes and where each node n ∈ N is associated a set of attributes $\{a\}_i \subseteq A$.

Since nodes in the ontologies are associated sets of attributes, and to still take the context of each node into account, to determine if a certain semantic relation holds between two nodes, we collect attributes along the paths of the nodes in the ontologies. More specifically, given two lightweight ontologies O_1 and O_2, a source node n_i in O_1 and a target node n_j in O_2:

[16] Details about the procedure are given in the DISI technical reports: (1) ***SAT-less Formula Matching*** by I. Zaihrayeu and (2) ***Attribute-Based Node Matching*** by F. Giunchiglia, U. Kharkevich, and I. Zaihrayeu.

- n_i is more specific than n_j iff $\{a\}_i \sqsubseteq \{a\}_j$;
- n_i is less specific than n_j iff $\{a\}_j \sqsubseteq \{a\}_i$;
- n_i is equivalent to n_j iff $\{a\}_i \sqsubseteq \{a\}_j$ and $\{a\}_j \sqsubseteq \{a\}_i$;

where $\{a\}_i$ and $\{a\}_j$ are the set of attributes collected along the paths of n_i and n_j respectively. In analogy with the basic algorithm, we say that $<n_i, n_j, R>$, with $R \in \{\sqsubseteq, \sqsupseteq, \equiv\}$, is a *mapping element* of a *mapping* M.

Notice that as a consequence of the fact that we dropped disjointness the definition of redundancy given in Section 3.2 is adapted by removing the disjointness condition. The definitions of *minimal* and *redundant mapping* remain the same instead.

3.5.2 The attribute-based algorithm

We use the same intuitions which are at the basis of the basic algorithm described in Section 3.3, with two main differences: (a) we do not compute disjointness and (b) we compute subsumption using Definition 9. The first one leads to a simplification of the pseudo-code, while the second one implies that for each node in the two trees we need to collect the attributes along the path. Similarly to the basic algorithm, the attribute-based algorithm is organized as follows:

- **Step 1:** compute the set of subsumption mapping elements which are minimal. Notice that we collapse, whenever possible, two subsumption relations of opposite direction into a single equivalence mapping element;
- **Step 2:** Compute on user request the mapping of maximum size (including minimal and redundant mapping elements) by propagating the elements in the minimal set.

3.5.3 Step 1: computing the minimal mapping

Exactly as in the basic algorithm, the minimal mapping is computed by a function **TreeMatch** whose pseudo-code is given in Fig. 18. M is the minimal set. T1 and T2 are the input attribute-based lightweight ontologies where each node is associated a set of attributes and a set of children. Each attribute is a pair <*an*, *av*> as from Definition 7.

```
10  attribute: struct of {an: concept; av:formula;};
20  node: struct of {attributelist: attribute[]; children: node[];}
30  T1,T2: tree of (node);
40  relation in {⊑, ⊒, ≡};
50  element: struct of {source: node; target: node; rel: relation;};
60  M: list of (element);
70  L1, L2: hashtable {key: node; attributelist: attribute[];};

80  function TreeMatch(tree T1, tree T2)
90   {CollectAttributesOnPath(root(T1), {}, L1);
100   CollectAttributesOnPath(root(T2), {}, L2);
110   TreeSubsumedBy(root(T1),root(T2), true);
120   TreeSubsumedBy(root(T2),root(T1), false);
130  };

140 function CollectAttributesOnPath(node n, attribute[] A, hashtable L)
150  {c: node;
160   A := A ∪ GetAttributes(n);
170   add(L, n, A);
180   foreach c in GetChildren(n) do CollectAttributesOnPath(c, A, L);
190  }
```

Fig. 18. Pseudo-code for the tree matching function

The first step consists in collecting the attributes along the path for all the nodes in the two trees (the two call at lines 90-100)[17]. This is achieved by the function **CollectAttributesOnPath**. To store collected attributes, two hash tables L1 and L2, in which nodes are used as keys, are provided (line 70). Functions **Add** and **Get** are used to add a new entry to and get an existing one (returning the set of attributes associated to a node) from a hash table respectively. Function **CollectAttributesOnPath** (lines 140-190) recursively computes the set of attributes as the union of the original set of attributes of the node n (which is returned by the **GetAttributes** function) with the attributes A of the parent (which are passed as a parameter to the function and are NULL for the root node). Notice that here we do not group attributes having same name.

[17] This corresponds to the step 2 of the original S-Match algorithm, namely computing the concept at node.

Hence, we process both trees top down. This is in order to minimize the number of calls to the node matching function described below. To maximize the performance of the system, **Tree-Match** has therefore been built as the sequence of two function calls to **TreeSubsumedBy** which compute the minimal set of subsumption mapping elements in the two directions (lines 80-90). Notice that in the second call, **TreeSubsumedBy** is called with the input ontologies with swapped roles (and with different boolean value codifying the direction). We also eventually collapse two subsumptions of opposite direction into equivalence when they hold on the same nodes (see Fig. 19, lines 180-210).

```
10  function boolean TreeSubsumedBy(node n1, node n2, boolean direction)
20   {c1,c2: node; LastNodeFound: boolean; F1, F2: attribute[];
30    if (direction) { A1 := get(n1,L1); A2 := get(n2,L2); }
40    else { A1 := get(n1,L2); A2 := get(n2,L1); }
50    if (!IsMoreSpecificThan(A1, A2)) then
60      foreach c1 in GetChildren(n1) do TreeSubsumedBy(c1,n2,direction);
70    else
80      {LastNodeFound := false;
90       foreach c2 in GetChildren(n2) do
100        if (TreeSubsumedBy(n1,c2,direction)) then LastNodeFound := true;
110      if (!LastNodeFound) then AddMappingElement(n1,n2,direction);
120      return true;
130     };
140   return false;
150  };

160 function AddMappingElement(node n1, node n2, boolean direction)
170  {if (direction) then M := M ∪ {<n1,n2,⊑>};
180   else if (<n1,n2,⊑> ∈ M) then {
190           M := M - {<n1,n2,⊑>};
200           M := M ∪ {<n1,n2,≡>};
210        }
220        else M := M ∪ {<n2,n1,⊒>};
230   };
```

Fig. 19. Pseudo-code for the **TreeSubsumedBy** function

TreeSubsumedBy (in Fig. 19) recursively finds all subsumption minimal mapping elements. Lines 30-60 implement a depth first recursion in the first tree till a subsumption is found. The matching task is crucially dependent on the node matching function **IsMoreSpecificThan** which takes the two sets of attributes associated to the two nodes n1 and n2 (collected along their whole path) and returns a positive answer in case of subsumption, or a negative answer if it is not the case or it is not able to establish it. By making use of the pre-computed semantic closure, the function **IsMoreSpecificThan** directly computes subsumption between set of attributes, as from Definition 9, without running SAT. Lines 70-140 implement what happens after the first subsumption is found. The key idea is that, after finding the first subsumption, **TreeSubsumedBy** keeps recursing down the second tree till it finds the last subsumption. When this happens, the resulting mapping element is added to the minimal mapping (line 90) using the function **AddMappingElement** (lines 160-230). See how we put equivalence instead (removing the previous subsumption) in the set when both subsumption directions hold (lines 180-210).

3.5.4 Step 2: Computing the mapping of maximum size

Given two attribute-based lightweight ontologies T1 and T2 and the minimal mapping M between them, computing the mapping of maximum size can be seen as the result of the propagation of the elements in M according to the given redundancy condition for subsumptions. Note that for each pair of nodes in M this can be done very efficiently since no calls to SAT are needed for that. See the pseudo-code for the **Maximize** function above. The mapping M' is initialized with the minimal set M (line 20) and then for each element in it we apply propagation as from patters 1 and 2, implemented by functions **PropagateP1** and **PropagateP2** respectively. They take care of substituting a subsumption element with equivalence when both directions hold. As it should be intuitive, **SubTree** and **Path** functions compute the set of nodes in the sub-tree and the set of nodes along the path of a given node (including the node itself) respectively.

```
10  function list of (element) Maximize(list of (element) M)
20   {list of (element) M' := M;
30    m: element;
40    foreach m in M do
50       if (GetRelation(m) ∈ {⊑, ≡}) then PropagateP1(node n1, node n2, M');
60       if (GetRelation(m) ∈ {⊒, ≡}) then PropagateP2(node n1, node n2, M');
70    return M'; };

80  function PropagateP1(node n1, node n2, list of {element} M')
90   {foreach c1 in SubTree(n1) do
100     foreach c2 in Path(n2) do {
110        if (!((c1 = n1) && (c2 = n2))) then
120          if (<c1,c2, ⊒> ∈ M') then {
130             M' := M' - {<c1,c2, ⊒>};
140             M' := M' ∪ {<c1,c2,≡>};
150          }
160          else if (<c1,c2, ≡> ∉ M') then M' := M' ∪ {<c1,c2, ⊑>};
170     }
180  };

190 function PropagateP2(node n1, node n2, list of {element} M')
200  {foreach c1 in Path(n1) do
210     foreach c2 in SubTree(n2) do {
220        if (!((c1 = n1) && (c2 = n2))) then
230          if (<c1,c2, ⊑> ∈ M') then {
240             M' := M' - {<c1,c2, ⊑>};
250             M' := M' ∪ {<c1,c2,≡>};
260          }
270          else if (<c1,c2, ≡> ∉ M') then M' := M' ∪ {<c1,c2, ⊒>};
280     }
290  };
```

Fig. 20. Pseudo-code for the computation of the mapping of maximum size

3.5.5 Implementation and evaluation

The attribute-based version of the minimal mapping algorithm has been implemented as part of the SWeb platform under development in our research group which already contained an implementation of S-Match (in its SAT-less version). We have compared the performances of the two tools on a toy example constituted by two classifications of 4 nodes each. In order to do that, the nodes in the classifications were assumed to have exactly one attribute called *label* whose value is the label of the node itself. The experiment was conducted on an Inter P4 2.6 Ghz, 1 GB RAM. The S-Match algorithm took 515 ms to compute the mapping composed of 9 correspondences (4 less general, 4 more general and 1 equivalence). The new implementation took 320 ms to compute the minimal mapping composed of 3 correspondences (1 per kind) and less than 1 ms to maximize it, with a saving of around 40% of the time. The mapping obtained coincides with the one computed by S-Match.

Chapter 4

4. Mapping validation

Fully manual approaches to the identification of semantic correspondences between ontologies are very precise, but they are extremely costly and hardly scale in case of very large ontologies. Conversely, automatic approaches can be very effective, but tend to fail when domain specific background knowledge is needed (see for instance [Lauser et al., 2008]). As a tradeoff between the two approaches, people can use automatic tools for the identification of an initial mapping that can be later refined by manually validating it. Yet, when the number of computed correspondences is very large, the validation phase can be very expensive.

In the previous chapter we have already mentioned the work by [Falconer and Storey, 2007] that underline how current matching tools offer poor support to users for the process of creation, validation and maintenance of the correspondences. In fact, given two schemas in input, most of the tools limit their support to the suggestion of an initial set of correspondences which is automatically computed by the system. In addition, even when a graphical interface is provided, they suffer of scalability problems as the number of nodes and correspondences grows [Robertson et al., 2005]. It is rather difficult to visualize even a single ontology. Current visualization tools do not scale to more than 10,000 nodes, and only a few systems support more than 1,000 nodes [Halevy, 2005]. The problem becomes even more challenging with matching, because it is necessary to visualize two ontologies, the source and target ontologies, and the potentially very big set of correspondences between them that grows quadratically in the size of the ontologies. As a consequence, handling them turns out to be a very complex, slow and error prone task.

In order to reduce the problems above, we propose to compute and validate the minimal mapping. With this purpose, in this chapter (Section 4.1) we provide the specifications for a new tool to interactively assist the user in the process of mapping creation and validation. A graphical user interface, still not fully implementing these specifications, is provided with the open source S-Match suite.

We also present the results of a matching experiment conducted as part of the *Interconcept* project (Section 4.2), a collaboration between the University of Trento, the University of Maryland

in the person of Prof. Dagobert Soergel[18] (currently at the University of Buffalo), and the U.S. National Agricultural Library (NAL). The main goal of the project was to test the MinSMatch tool on large-scale knowledge organization systems (KOS), i.e. NALT and LCSH in the specific case. With the experiment:

- we show that the automatic parsing of their structures can identify problems and imprecisions which are really difficult or nearly impossible to identify by manual inspection, such as duplicated entries, cycles, and redundant relations.
- we show that current matching results can be significantly improved by enhancing the NLP pipeline and by improving the quality and coverage of the available background knowledge
- we show that, since the number of correspondences in the minimal mapping is typically a very small portion of the overall set of correspondences between the two ontologies, by concentrating on the minimal mapping it is possible to save up to 99% of the manual checks required for validation.

[18] http://www.dsoergel.com/

4.1. User interaction during validation

Validating a mapping means taking a decision about the correctness of the correspondences suggested by the system. We say that the user *positively validates* a correspondence if the user accepts it as correct, while we say that the user *negatively validates* a correspondence if the user rejects it, i.e. the user does not accept it as correct. Both rejected and accepted correspondences have to be marked to record the decision taken. We use MinSMatch to compute the initial minimal mapping. Focusing on the elements in this set minimizes the work load of the user. In fact, they represent the minimum amount of information which has to be validated as it consequently results in the validation of the rest of the (redundant) mapping elements.

To do that, the system has to suggest step by step the order of correspondences to be validated. In particular, this order must follow the partial order over the mapping elements defined in Section 3. The intuition is that if an element *m* is judged as correct during validation, all mapping elements which are redundant w.r.t. *m* are consequently correct. Conversely, if *m* is judged as incorrect we need to include in the minimal set the *maximal elements* (as they are defined in partial ordered sets, see for instance [Davey and Priestley, 2002]) from the set of mapping elements which are redundant w.r.t. *m*, that we call the *sub-minimal elements* of *m*, and ask the user to validate them.

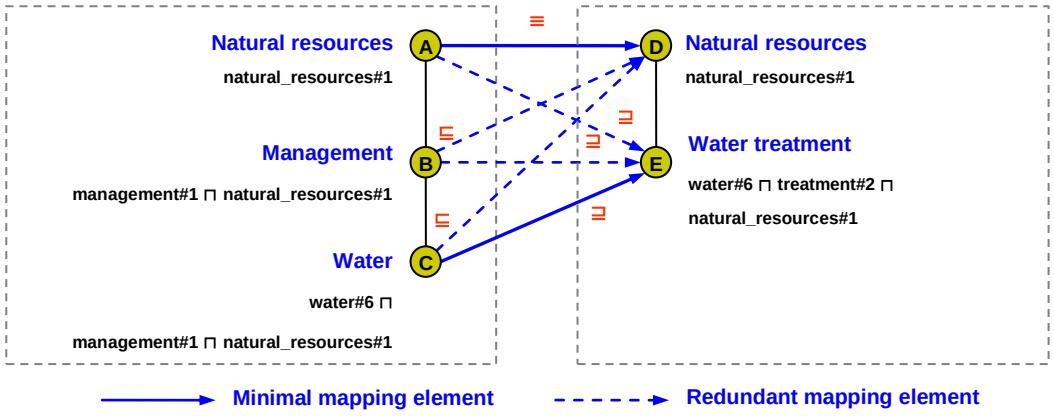


Fig. 21. The minimal and redundant mapping between two lightweight ontologies

As an example, consider Fig. 21 that shows the minimal mapping (the solid arrows) and the mapping of maximum size computed between two lightweight ontologies in the agricultural domain. If <A, D, ≡> is rejected, we need to extend the validation to the elements in the set of

mapping elements {<A, E, ⊒>, <B, D, ⊑>, <C, D, ⊑>} which are redundant w.r.t. *m*. <A, E, ⊒> and <B, D, ⊑> are the maximal elements in the set. Notice in fact that the element <C, D, ⊑> needs to be validated only if <B, D, ⊑> is further rejected. In fact, <C, D, ⊑> is in turn redundant w.r.t. <B, D, ⊑>. As described below, sub-minimal elements can be efficiently computed.

Note that, for a better understanding of the correspondences, it is important to show to the user the strongest semantic relation holding between the nodes, even if it is not in the minimal set. For example, showing equivalence where only a direction of the subsumption is minimal.

The validation process is illustrated in Fig. 22. The minimal mapping M between the two light-weight ontologies T1 and T2 is computed by the **TreeMatch** (line 10) and validated by the function **Validate** (line 20). Both M and the **TreeMatch** function are described in the previous chapter (Fig. 13). At the end of the process, M will contain only the mapping elements accepted by the user. The **Validate** function is given at lines 30-90. The validation process is carried out in a top-down fashion (lines 40-50). This is to evaluate in sequence the elements that share as much contextual information as possible. This in turn reduces the cognitive load requested to the user to take individual decisions [Falconer and Storey, 2007]. The presence of an element *m* between two nodes n1 and n2 in M is tested by the function **GetElement** (line 60). In positive case the function returns it, otherwise NULL is returned. Each element is then validated using the function **ValidateElement** (line 70), whose pseudo-code is given in Fig. 23. The process ends when all the nodes in the two trees have been processed. A possible optimization consists in stopping the process when all the elements in M have been processed.

```
10  M := TreeMatch(T1, T2);
20  Validate(M);

30  function void Validate(list of (element) M)
40  { foreach n1 in T1 do
50       foreach n2 in T2 do {
60          m := GetElement(M, n1, n2);
70          if (m != NULL) ValidateElement(m);
80       }
90  };
```

Fig. 22. The validation process of the minimal mapping M

```
10  function void ValidateElement(element m)
20   { S: list of (element);
30     if IsValid(m) AddElement(m, M);
40     else {
50        RemoveElement(m, M);
60        S := GetSubminimals(m);
70        foreach m in S do { if (!IsRedundant(m)) ValidateElement(m); }
80     }
90  };
```

Fig. 23. The validation process of a single element m

The validation of a single element *m* is embedded in the **ValidateElement** function. The correctness of *m* is established through a call to the function **IsValid** (line 30), that takes care of the communication with the user. The user can accept or reject *m*. If *m* is accepted (**IsValid** returns true), *m* is added to the set M using the function **AddElement** (line 30). Note that this is strictly necessary when the **ValidateElement** is called on a sub-minimal element at line 70. Otherwise, if *m* is rejected (**IsValid** returns false), it is removed from M using the function **RemoveElement** (line 50) and its sub-minimal elements, computed by the function **GetSubminimals** (line 60), are recursively validated (line 70). The pseudo-code for the **GetSubminimals** function is given in Fig. 24. It encodes the rules for propagation suggested in Section 3.3.3 (based on the structure of the two ontologies) to identify the elements that follow an element *m* in the partial order.

Two observations are needed.

The first is that a sub-minimal element can be redundant w.r.t. more than one element in M. In these cases we postpone their validation to the validation of the elements for which they are redundant. For instance, in Fig. 21 <A, E, ⊒> is redundant w.r.t. both <A, D, ≡> and <C, E, ⊒>. Therefore, the validation of <A, E, ⊒> is postponed to the validation of <C, E, ⊒>. In other words, if <C, E, ⊒> is positively validated, then it will be superfluous asking the user to validate <A, E, ⊒>. We use the function **IsRedundant**, given in Fig. 17 (line 70), for this. This also avoids validating the same element more than once.

The second is that, in order to keep the strongest semantic relation holding between two nodes, the following rules are enforced:

(a) if we add to M two subsumptions of opposite directions for the same pair of nodes, we collapse them into equivalence;
(b) if we add an equivalence between two nodes, it substitutes any subsumption previously inserted between the same nodes, but it is ignored if we already have in M a disjointness between these nodes;
(c) if we add a disjointness between two nodes, it substitutes any other relation previously inserted in M between the same nodes.

```
10  function list of (element) GetSubminimals(element <n1,n2,R>)
20   { S: list of (element);
30     if (R == ⊑ || R == ≡) {
40       c2 := GetParent(n2);
50       if (c2 != NULL) AddElement(S, <n1,c2,⊑>);
60       else foreach c1 in GetChildren(n1) do AddElement(S, <c1,n2,⊑>);
70     }
80     if (R == ⊒ || R == ≡) {
90       c1 := GetParent(n1);
100      if (c1 != NULL) AddElement(S, <c1,n2,⊒>);
110      else foreach c2 in GetChildren(n2) do AddElement(S, <n1,c2,⊒>);
120    }
130    if (R == ⊥) {
140      foreach c2 in GetChildren(n2) do AddElement(S, <n1,c2,⊥>);
150      foreach c1 in GetChildren(n1) do AddElement(S, <c1,n2,⊥>);
160    }
170    return S;
180  };
```

Fig. 24. The function for the identification of the sub-minimal elements

4.2. The LCSH vs. NALT experiment

Rather than evaluating the mapping found, i.e. in terms of precision and recall, the main goal of the Interconcept project was to test the MinSMatch tool on large-scale library KOS to understand what are the typically issues that have to be faced (described in Section 4.2.2) and what is the percentage of elements in the minimal mapping w.r.t. the overall number of correspondences between the two KOS (Section 4.2.3). The experiment was conducted on NALT and LCSH:

- **NALT** (US National Agriculture Library Thesaurus) 2008 version contains 43,037 subjects, mainly about agriculture, which are divided in 17 subject categories including for instance *Taxonomic Classification of Organisms*, *Chemistry and Physics* and *Biological Sciences*. NALT was available as a text file formatted to make relations between subjects recognizable.
- **LCSH** (US Library of Congress Subject Headings) 2007 version contains 339,976 subjects in many fields. LCSH was available in the MARC 21 format encoded in XML.

In both of them the records are unsorted and the information about the hierarchical structure is implicitly codified in the relations between the preferred terms of the subjects.

4.2.1 Phases of the experiment

The matching experiment has been organized in a sequence of 5 subsequent steps, reported in Fig. 25, which can be iterated to progressively improve the quantity and the quality of the mapping elements found. From the analysis of the node formulas and the mapping elements computed by the MinSMatch algorithm, at each iteration problems and mistakes can be identified and fixed. In the following we provide additional details about single steps performed.

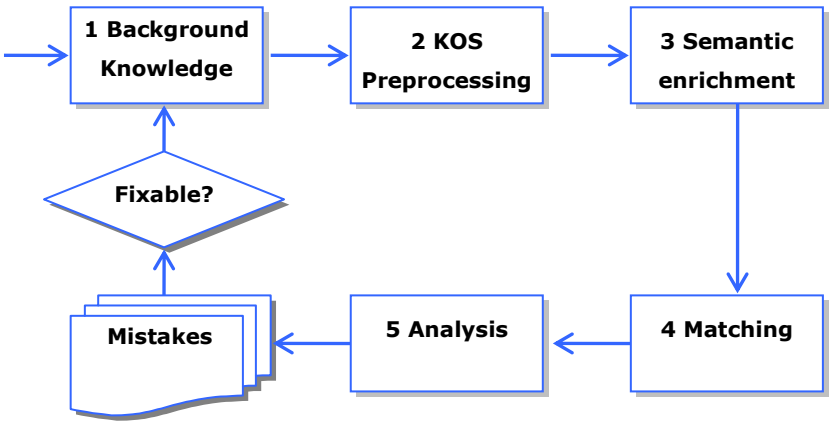


Fig. 25. A global view of the phases of the experiment

Step 1. Background Knowledge. The availability of an appropriate amount of background knowledge is clearly fundamental for any application which deals with semantics. By default, MinSMatch uses WordNet.

Step 2. Preprocessing: from KOS to classifications. During this step, by using only preferred terms and BT/NT (Broader Term/Narrower Term) relations, the KOS are parsed and approximated to classifications. This step is automatic.

Step 3. Semantic enrichment: from classifications to lightweight ontologies. The goal of this step is to encode the classifications, output of the previous step, into lightweight ontologies. With the translation, natural language labels are translated into propositional DL formulas. This process is also called semantic enrichment. We used the standard NLP pipeline presented in [Zaihrayeu et al., 2007], consisting of tokenization, part-of-speech (POS) tagging and word sense disambiguation (WSD). It applies a finite set of BNF[19] rules (derivation rules) which cover a finite set of patterns obtained by training the pipeline on the DMoz Web directory.

Step 4. Matching. MinSMatch is executed on the lightweight ontologies output of the previous step to compute the minimal mapping and the mapping of maximum size between them.

Step 5. Analysis of mistakes. The analysis identifies problems in each single step which are fixed if possible. The process can be iterated to further improve the results. This step is manual.

[19] http://www.garshol.priv.no/download/text/bnf.html

4.2.2 Critical issues found

The whole process was iterated only once. In this section we summarize and comment on the main results of the experiment in terms of difficulties encountered and their quantitative analysis. In particular, we discuss problems of the sources, loss of information, problems due to the NLP pipeline and missing background knowledge.

Problems with the sources. We have identified the following problems/imprecisions in the KOS structures:

- **Ambiguous terms.** Both in NALT and LCSH, preferred terms are directly used as indexes to define relations between entries (e.g. *Geodesy* BT *Geophysics*). However, lexically equivalent terms might represent a potential source of ambiguity. In LCSH there are 575 cases where the same preferred term is used in different records. For instance they include *Computers*, *Film trailers*, *Periodicals*, *Christmas*, *Cricket*.
- **Cycles.** In LCSH we have found 6 chains of terms forming cycles. For instance: *#a Franco-Provencal dialects* BT *#a Provencal language #x Dialects* BT *#a Provencal language* BT *#a Franco-Provencal dialects*.
- **Redundant relations.** We discovered several redundant BTs, namely distinct chains of BTs (explicitly or implicitly declared) with same source and target. For instance, in NALT the following chains were identified:

  life history BT *biology* BT *Biological Sciences*
  life history BT *Biological Sciences*

  sprouts (food) BT *vegetables* BT *plant products*
  sprouts (food) BT *plant products*

Table 4 provides some statistics about the overall amount of BTs and redundant BTs in NALT and LCSH. It also provides information about the number of parsed terms and the number of cases in which we found multiple non redundant BTs (i.e., a polyhierarchy) for a given node. These results show that automatic parsing provides clear added value with respect to manual inspection. In fact, these problems (identified during the parsing phase) are really difficult or nearly impossible to identify manually. They also give some clue about the quality of the sources. In

NALT almost 2% of the BTs are redundant, while in LCSH this quantity reaches 3%. However, they still play a useful role in navigation.

	NALT	LCSH
Preferred terms imported	43,038	335,701
Total number of BTs	46,400	344,796
Multiple non redundant BTs	2,821	87,395
Redundant BTs	807	9,256

Table 4. Statistics about preferred terms and BT relations

Loss of information. The output of the parsing phase is given as directed acyclic graph (DAG) structures in which node labels are the preferred terms appearing in the original sources. Table 4 provides the number of preferred terms, and therefore of nodes, in the two graphs. In order to work with MinSMatch, they have to be approximated to classifications. With this goal, we preliminary remove redundant BTs. The remaining BTs are analyzed to identify cases of multiple BTs with same source. For each of them we keep only one BT (for instance, giving priority to those which lead to main headings) and remove all the others. After removing redundant BTs and selecting one of the multiple BTs (for ease of processing), both NALT and LCSH appear as a forest of trees where node labels are the preferred terms of the original NALT/LCSH records. In detail:

- LCSH is decomposed into 65,744 trees where the 25 most populated trees include 196,723 nodes (58%) and 59,105 trees are constituted by only one node (18% of the overall number of nodes);
- NALT is decomposed in all nodes 17 trees, each of them linked to a Subject Category[20].

For each KOS we introduced a dummy root node (called TRUE) to create a single large tree. During this phase we have a clear loss of information, in particular in the kind of relations selected (we keep only BTs and NTs), in the terms selected (we keep only preferred terms) and structural information (we remove multiple BTs).

[20] http://agclass.nal.usda.gov/dne/search_sc.shtml

NLP problems. Table 5 summarizes some statistics about the quantity (#) and percentage (%) of labels which can be processed by the NLP pipeline used.

Table **6** provides some examples of node labels and corresponding formulas (taking into account the full path). Concepts without the number (for instance anti_corrosives#, which is also recognized as a multiword) are evidence of lack of background knowledge. Notice that for the properties of the lightweight ontologies, a failure in the enrichment of a node propagates to the whole sub-tree rooted in it. Label of nodes in such sub-trees are what we call *affected labels* in Table **6**. Nodes which cannot be parsed are clearly skipped during the matching phase.

	NALT		**LCSH**	
	#	%	#	%
Number of imported nodes	43,037	---	196,723	---
Nodes parsed with success	27,782	65%	83,576	42%
Nodes failed during parsing	15,255	35%	113,147	58%
Rejected labels	1,175	3%	25,618	13%
Affected labels	14,078	33%	87,529	45%

Table 5. Statistics about the semantic enrichment

From Table 5 it is easy to note that nodes that cannot be parsed are a significant portion of the total number in the two KOS. This is particularly evident in LCSH where more than half of the labels cannot be parsed.

By analyzing the labels which are not supported by the NLP pipeline used, we identified some recurrent patters. Specifically, labels including round parenthesis - such as *Life (Biology)* - and labels including 'as' - such as *brain as food* - cannot be parsed. These kinds of labels are very frequent in thesauri. The term in parenthesis, or after the 'as', is used to better describe, disambiguate or contextualize terms. In particular, in NALT and LCSH, labels of the first kind are mainly used:

- to provide the acronym of a term - "Full term (Acronym)" - or to provide the description of an acronym - "Acronym (Full term)". For instance, *nitrate reductase (NADH)*;
- to disambiguate terms. For instance, *mercury (planet)* and *mercury (material)*;
- to represent a compound concept. For instance, *growth (economics) (*which could also be represented as *economic growth)*.

Notice that 83% of label rejections in LCSH and 30% in NALT are due to the missing parenthesis pattern. The pattern with 'as' is less frequent and represents around 1% of the rejection cases, both in NALT and LCSH. The pipeline could be, therefore, significantly improved by including new rules for these patters. As a matter of fact, this use of parenthesis is pretty frequent in thesauri but it is not in Web directories (e.g. DMoz), that were used to train the NLP pipeline used. It is clear that a rule based pipeline cannot cover all the cases and work uniformly when dealing with different kinds of sources. An extended NLP pipeline which gets around all these problems is described in [Autayeu et al., 2010]. However, this was not developed yet at the time of the experiment.

LCSH label (and path)	**DL formula**
Water repellents *Path: Chemicals/Repellents/Water repellents*	chemicals#34600 ⊓ repellents#1626 ⊓ (water#75538 ⊓ repellents#1626)
Neutron absorbers *Path: Chemicals/Bioactive compounds/Poisons/Neutron absorbers*	chemicals#34600 ⊓ (bioactive# ⊓ compounds#84901) ⊓ poisons#23087 ⊓ (neutron#27237 ⊓ absorbers#95684)
Stress corrosion *Path: Chemicals/Chemical inhibitors/Corrosion and anti-corrosives/Stress corrosion*	chemicals#34600 ⊓ (chemical#21081 ⊓ inhibitors#93475) ⊓ (corrosion#67669 ⊔ anti_corrosives#) ⊓ (Stress#66019 ⊓ corrosion#67669)

Table 6. Some examples of labels from LCSH which can be successfully parsed

Missing background knowledge. Our experiment confirms that the quality and the quantity of the correspondences identified by the algorithm directly depend on the quality and the coverage of available knowledge. In fact, we found that 30% of the logic formulas computed for LCSH and 72% for NALT contain at least one concept which is not present in our background knowledge. The fact that the phenomenon is more evident in NALT is most likely because NALT contains a higher number of domain specific terms which are therefore not present in WordNet. To increase the quantity of knowledge we could import it from a selection of knowledge sources. We analyzed two possible candidates, the Alcohol and Other Drugs Thesau-

rus[21] (AOD) and the Harvard Business School Thesaurus[22] (HBS) which were made available as material for the Interconcept project. However, we found that the increment of the pure syntactic (surface) overlap of the new terms (including preferred and non-preferred terms) with NALT and LCSH would be less than 0.5%. This is something not unexpected, since the reason of this discouraging result is probably due to the different focus of the thesauri: NALT is mainly about agriculture, while AOD is about drugs and HBS is about business. This is also confirmed by a very low syntactic overlap between NALT and AOD (7%) and between NALT and HBS (4%). However, AOD and HBS are partially faceted and contain many general conceptual primitives that would be useful in a deeper semantic analysis but that would not be detected as matches at the surface level. Domain thesauri, like AGROVOC, would be more appropriate.

4.2.3 Matching results

We ran MinSMatch on a selection of NALT/LCSH branches which turned out to have a high percentage of labels that could be successfully parsed. See Table 7 for details.
Table 8 shows details about conducted experiments in terms of the branches which are matched, the number of elements in the mapping of maximum size (obtained by propagation from the elements in the minimal mapping), the number of elements in the minimal mapping and the percentage of reduction in the size of the minimal set w.r.t. the size of the mapping of maximum size.

Id	Source	Branch	Number of nodes	Enriched nodes
A	NALT	Chemistry and Physics	3944	97%
B	NALT	Natural Resources, Earth and Environmental Sciences	1546	96%
C	LCSH	Chemical Elements	1161	97%
D	LCSH	Chemicals	1372	93%
E	LCSH	Management	1137	91%
F	LCSH	Natural resources	1775	74%

Table 7 - NALT and LCSH branches used

[21] http://etoh.niaaa.nih.gov/aodvol1/aodthome.htm
[22] http://hul.harvard.edu/ois/ldi/

We executed MinSMatch both between branches with an evident overlap in the topic (A vs. C and D, B vs. F) and between clearly unrelated branches (A vs. E). As expected, in the latter case we obtained only disjointness relations. This demonstrates that the tool is able to provide clear hints of places in which it is not worth to look at in case of search and navigation. All the experiments show that the minimal mapping contains significantly less elements w.r.t. the mapping of maximum size (from 57.4% to 99.3%). Among other things, this can incredibly speed-up the validation phase. It also shows that exact equivalence is quite rare. We found just 24 equivalences, and only one in a minimal mapping. This phenomenon has been observed also in other projects, for instance in Renardus [Koch et al., 2003] and CARMEN.

Matching experiment		Mapping of maximum size	Minimal mapping	Reduction
A vs. C	**Mapping elements found**	**17716**	**7541**	**57,43%**
	Disjointness	8367	692	91,73%
	Equivalence	0	0	---
	more general	0	0	---
	more specific	9349	6849	26,74%
A vs. D	**Mapping elements found**	**139121**	**994**	**99,29%**
	Disjointness	121511	754	99,38%
	Equivalence	0	0	---
	more general	0	0	---
	more specific	17610	240	98,64%
A vs. E	**Mapping elements found**	**9579**	**1254**	**86,91%**
	Disjointness	9579	1254	86,91%
	Equivalence	0	0	---
	more general	0	0	---
	more specific	0	0	---
B vs. F	**Mapping elements found**	**27191**	**1232**	**95,47%**
	Disjointness	21352	1141	94,66%
	Equivalence	24	1	95,83%
	more general	2808	30	98,93%
	more specific	3007	60	98,00%

Table 8. Results of matching experiments

Chapter 5

5. Mapping evaluation

Despite this has not been claimed as one of the main objectives of the thesis, some work has also been done on the issue of evaluating and comparing different matching techniques. This has been recognized as one of the important challenges in [Shvaiko and Euzenat, 2008]. In fact, the rapid growth of various matching approaches makes the issues of their evaluation and comparison more and more severe. The OAEI initiative started in 2005 precisely with this aim. Among the issues to be addressed in ontology matching evaluation in order to empirically prove the matching technology to be mature and reliable, Shvaiko and Euzenat include the need for:

- Large datasets and corresponding gold standards for their evaluation
- Methods for the comparison of different ontology matching techniques
- More accurate evaluation methods and quality measures

However, with matching techniques being the main focus of the ontology matching field, a few initiatives pay attention to evaluation. On the one hand, general [Noy and Musen, 2002] [Do et al., 2002] and domain-specific [Kaza and Chen, 2008] [Isaac et al, 2009] evaluation experiments are reported, without discussing the evaluation methodology followed. On the other hand, considerable attention has been paid to appropriateness and quality of the measures [Ehrig and Euzenat, 2005] [Euzenat, 2007][David and Euzenat, 2008]. Attention has also been brought to the mapping itself. In [Meilicke and Stuckenschmidt, 2008] the authors propose to complement the precision and recall with new measures to take into account possible mapping incoherence, thus addressing the issues of internal logical problems of the mapping and the lack of reference mappings. In [van Hage et al., 2007] two evaluation techniques are proposed. The first is practice-oriented and evaluates the behaviour of the mapping in use. The second focuses on the manual evaluation of a mapping sample and the generalization of the results. [Sabou and Gracia, 2008] raises the issue of evaluating non-equivalence correspondences, pointing out that several systems also produce subsumption and disjointness correspondences. In particular, they discuss the issue of evaluating a mapping that contains redundant correspondences, that is, correspondences that can be logically derived from the others in the mapping. They compute precision both for the original set and the set from which the redundant correspondences are removed.

In this chapter, with focus on schema-based semantic matching techniques, we address the three issues and show that, by following certain rules, the quality of the evaluations can be significantly improved, particularly in regard to the accuracy of precision and recall measures obtained.

5.1. Coverage of a gold standard

Gold standards, also called reference mappings or reference alignments, are of fundamental importance for computing the well-known precision and recall measures at the purpose of evaluating a matching tool. Typically, hand-made positive (GS+) and negative (GS-) gold standards contain correspondences considered correct and incorrect, respectively. Ideally, the GS- complements the GS+, leading to a precise evaluation. Yet, annotating all correspondences in big datasets (with thousands or millions of correspondences between nodes) is impractical and therefore the gold standard is often composed of three sets:

- **GS+**: the set of correspondences considered correct;
- **GS-**: the set of correspondences considered incorrect;
- **Unk**: the pairs of nodes for which the semantic relation is unknown.

If we denote the result of the matcher (the mapping) with *Res*, precision and recall can be computed in standard ways as follows [Giunchiglia et al., 2008]:

(1) **Precision** = TP / (TP + FP) = |Res ∩ GS+| / (|Res ∩ GS+| + |Res ∩ GS-|)

(2) **Recall** = TP / (TP + FN) = |Res ∩ GS+| / |GS+|

where:

- **TP** (True Positives) is the set of correspondences found by the algorithm that hold;
- **FP** (False Positives) is the set of correspondences found by the algorithm that do not hold;
- **FN** (False Negatives) is the set of correspondences that hold, but which were not found by the algorithm.

For the cases in which GS- is not available, precision can be approximated as follows:

(3) **Precision** = |Res ∩ GS+| / |Res|

These sets are illustrated in Fig. 26. The precision gives an indication of the amount of noise that is retrieved by the matching algorithm (how many correct correspondences it returns) while the recall is a measure of the coverage of the algorithm (how many correspondences the algorithm found and missed).

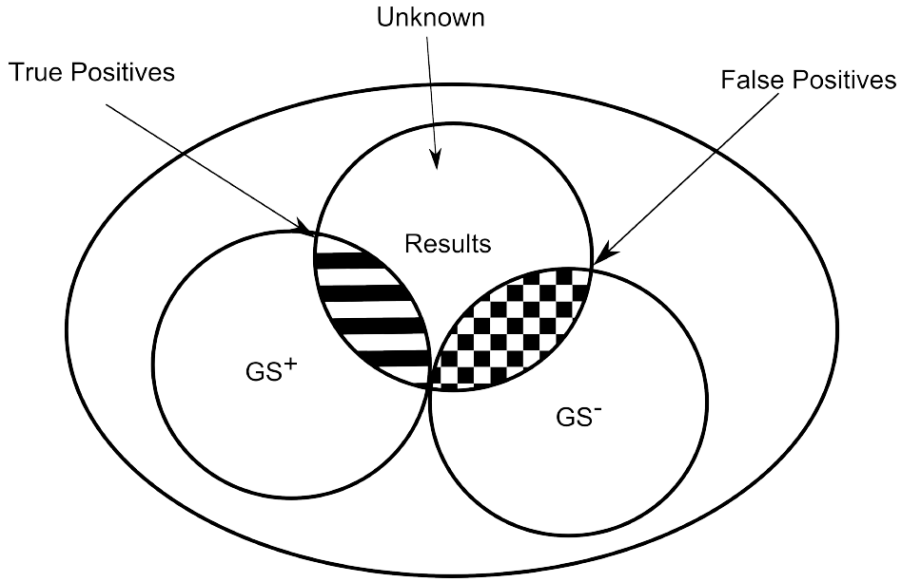


Fig. 26. Evaluating a mapping given a positive and negative gold standard

For example, if for sake of simplicity we use numbers to indicate correspondences, we could have:

Res = {1, 2, 3, 4}
GS+ = {1, 2, 5, 7, 9, 10}
GS- = {3, 4, 6, 8}
Unk = {}

(4) Precision = 2 / (2 + 2) = 0.5 Recall = 2 / 6 = 0.33

Given two ontologies of size *n* and *m*, the size of a mapping and the gold standards clearly range in [0, *n* x *m*]. To enable precise computation of precision and recall, one should inspect all *n* x *m*

combinations of nodes and consider all possible semantic relations that can hold between them. For large ontologies this is practically impossible. The huge effort required for their construction is the main reason why only a few gold standards are available and evaluation campaigns tend to use very small ontologies, risking a loss of statistical significance of the results and biasing towards one algorithm or the other.

When setting up exhaustive GS+ and GS- is not possible, the common practice is to inspect only a subset of the *n* x *m* node pairs [Avesani et al., 2005] [Giunchiglia et al., 2008]. Partial coverage clearly leads to an approximated evaluation. In particular, we cannot say anything about the subset Res ∩ Unk of the correspondences. However, if GS+ and GS- are sampled properly, the precision and recall can be still evaluated in a statistically significant manner. Suppose we could have reduced coverage compared to the previous example, as follows:

Res = {1, 2, 3, 4}
GS+ = {1, 2, 7}
GS- = {3, 6, 8}
Unk = {4, 5, 9, 10}

(5) Precision = 2 / (2 + 1) = 0.66 Recall = 2 / 3 = 0.66

As it turns out by comparing the measures in (5) with those in (4), such evaluations may be very different from the real values. From this simple analysis it should be clear that:

- There is a need for large gold standards;
- It is important to provide a negative gold standard for a good approximation of the precision and recall measures;
- To be statistically significant, an adequate portion of the correspondences must be covered by the positive and negative gold standards;
- In a sampled gold standard, reliability of results depends on the portion of the pairs covered by the positive and negative gold standard and dually those unknown.

5.2. Comparing different matchers

Current state of the art matchers output different kinds of relations. While most of the matching tools, such as the already presented Rondo (Similarity Flooding), Cupid and Coma only produce equivalence, some tools - such as Aroma [David et al., 2007] - also produce less general and more general relations. At the best of our knowledge, apart from S-Match and MinSMatch, only ctxMatch [Bouquet et al., 2003a] and Spider [Sabou and Gracia, 2008] also produce explicit disjointness.

To cope with this diversity, different matchers are usually compared without distinguishing among the different semantic relations produced and only the presence or absence of a relation between a pair of nodes is evaluated. This means, for instance, that subsumption and equivalence are considered as the same. This approach can be used to compare heterogeneous correspondences, but clearly leads to imprecise results.

A particular discourse has to be made for disjointness relations. Typically disjointness correspondences are seen as negative results. They provide a clear indication of two completely unrelated nodes. As such they play an important role in cutting the search space. Nevertheless, since the majority of matching tools do not consider them interesting to the users, they do not compute them at all, but corresponding node pairs are rather put in the GS-. Notice that for instance this means that inconsistent nodes are reduced to the subsumption case (see Section 3). This penalizes the evaluation of tools returning disjointness. In fact, they would be considered as false positives in the case.

It is important to do not confuse disjointness with overlap. Consider the example in Fig. 27 where two classifications have been translated into lightweight ontologies with formulas given under the labels. In particular, notice how the meaning of the node C includes the meaning of the node B above it. The correspondence <A, C, ⊒> is a correct matching result and as such should be part of the GS+. What about the relation between A and B? They are not disjoint as they share C. The relation is rather an overlap, namely $A \cap B \neq \varnothing$. Discriminating the two cases above is fundamental both to conclude the right relations between the nodes and to correctly evaluate precision and recall of disjointness relations when they are explicitly computed by the matching tool. In fact, the main problem is that negative gold standards (when available) typically contain undifferentiated correspondences. For instance, the authors of [Giunchiglia et al., 2008] make no

difference between disjointness and overlap relations. To the best of our knowledge, no evaluations take disjointness and overlap relations into account when measuring precision and recall.

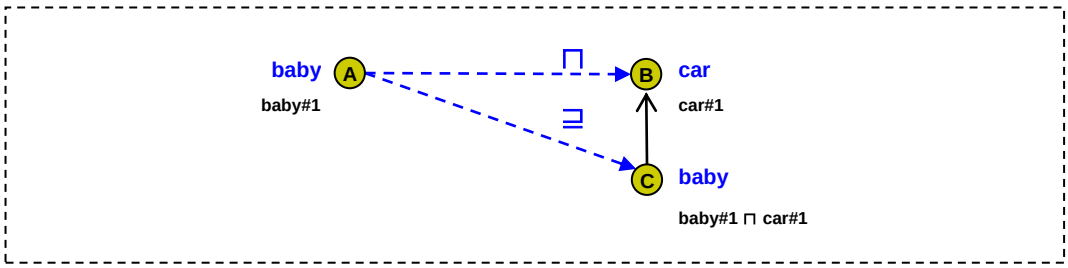


Fig. 27. Overlap between nodes A and B

5.3. Maximized and minimized gold standard

The notion of minimal mapping can be used to judge about the quality of a gold standard. For this purpose, we define two new functions: the **Min**(mapping) function to remove the redundant correspondences from a mapping - producing the minimized mapping - and the **Max**(mapping) function to add all the redundant correspondences to a mapping, producing what we call here the maximized mapping.

In the following we provide three observations.

The first observation is that under our settings and staying within lightweight ontologies guarantees that the maximized mapping is always finite and thus corresponding precision and recall can always be computed.

The second observation is that, in contrast with [Sabou and Gracia, 2008], we argue (and show with an example) that comparing the minimized versions of the mapping and the gold standards is not informative. The reason is that the minimization process can significantly reduce the amount of correspondences in their intersection. In other words, they can share a few non-redundant correspondences still generating a significant amount of redundant correspondences in common. Notice that different non-redundant correspondences can generate the same redundant correspondences.

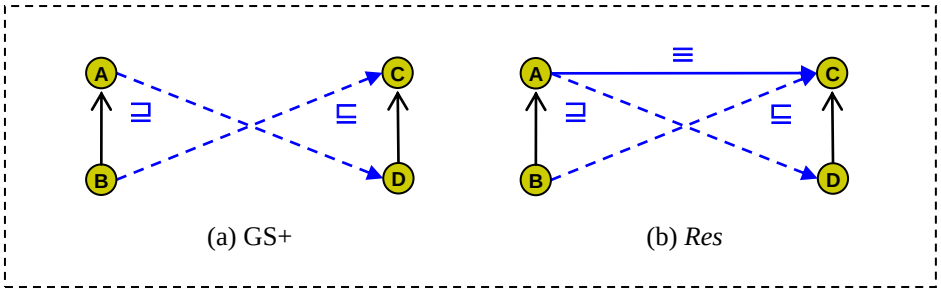


Fig. 28. Example of minimization affecting precision and recall

Consider the examples in the Fig. 28. Suppose that all the displayed correspondences are correct. Notice that in (b) <A, D, ⊒> and <B, C, ⊑> follow from <A, C, ≡>. Suppose that our gold standard, given in (a), as it often happens with large datasets, is incomplete (it contains only the <A, D, ⊒> and <B, C, ⊑>, while <A, C, ≡> is unknown) and to use formula (3) to compute precision which returns an approximated value. Suppose that the matcher, being good enough, finds all the correspondences displayed in (b). By computing the precision and recall figures first on the original and then on the minimized versions of the mapping and the gold standard we would obtain:

(6) Res = {1, 2, 3}	GS+ = {1, 2, 7}	Precision = 0.66	Recall = 1
(7) Min(Res) = {1}	Min(GS+) = {2, 3}	Precision = 0	Recall = 0

Compare the normal situation (6) with (7) that shows the situation when minimized sets are used to calculate precision and recall figures. From this simple example we see that precision and recall figures computed on the minimized versions are far from the real values and are actually unreliable.

Our last observation is that using maximized sets gives no preference to redundant or non-redundant correspondences and leads to more accurate results. In particular, recall figure better shows the amount of information actually found by the system. In fact, by maximizing the sets we also decrease the number of unknown correspondences. Consider the example in Fig. 29. The precision and recall figures are given in (8) for the original sets and in (9) for the maximized ones.

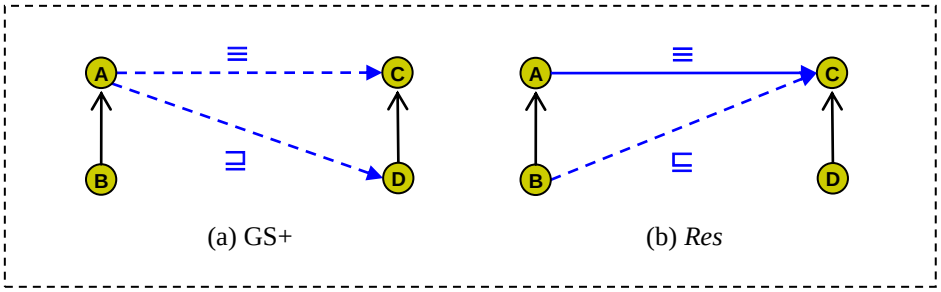


Fig. 29. Example of maximization affecting precision and recall

(8) Res = {1, 3}	GS+ = {1, 2}	Precision = 0.5	Recall = 0.5
(9) Max(Res) = Max(GS+) = {1, 2, 3}		Precision = 1	Recall = 1

Maximizing a gold standard can also reveal some unexpected problems and inconsistencies. For instance, we can discover that even if GS+ and GS- are disjoint, Max(GS+) and Max(GS-) are not, namely, Max(GS+) ∩ Max(GS-) ≠ ∅. During our experiments with the TaxME2 gold standard [Giunchiglia et al., 2008], we discovered that there are two correspondences in the intersection of GS+ and GS- and 2187 in the intersection of their maximized versions.

We conducted several experiments to study the differences between precision and recall measures when comparing the minimized and maximized versions of the gold standards with the minimized and maximized versions of the mapping returned by S-Match. We used three gold standards. The first two datasets (103/304) come from OAEI; they describe publications, contain few nodes and corresponding gold standard is exhaustive. It only contains equivalence correspondences. The second and third are described in described in Section 3.4. In particular, the second two (the Topia/Icon) come from the arts domain and the gold standard is crafted by experts manually. The third two (the Source/Target) have been extracted from the Looksmart, Google and Yahoo! web directories and the gold standard is part of the TaxME2, described in [Giunchiglia et al., 2008].

Unfortunately, all these gold standards suffer to a certain degree from the problems described in the previous sections, thus the measures obtained must be considered as indicative. Table 9 contains precision and recall figures calculated using standard precision and recall formulas (1) and (2). For the cases where no GS- is provided, (3) is used instead of (1). In particular, these figures are the result of the comparison of the minimized mapping with the minimized gold standards (min), the original mapping with the original gold standards (res) and the maximized mapping

with the maximized gold standards (max), respectively. As it can be noted from the measures obtained comparing the maximized versions with the original versions, the performance of the algorithm is on average better than expected. We can then conclude that to obtain accurate measures it is fundamental to maximize both the gold standard and the matching result.

		Precision, %			Recall, %		
#	Dataset pair	min	res	max	min	res	max
1	103/304	32.47	9.75	69.67	86.21	93.10	92.79
2	Topia/Icon	16.87	4.86	45.42	10.73	20.00	42.11
3	Source/Target	74.88	52.03	48.40	10.35	40.74	53.30

Table 9. Precision and recall for the minimized, original and maximized sets

Chapter 6

6. Building domain knowledge

In the last forty years many projects have aimed at constructing knowledge bases. Both manual and automatic approaches are followed. On the one hand, hand-crafted resources are surely more accurate but difficult to construct and maintain. On the other hand, automatically built resources typically suffer of poor quality and often of a not clear semantics.

In Library and Information Science (LIS), the usage of methodologies to domain knowledge construction at the purpose of organizing books on the shelves, with the *faceted approach* representing a peak of excellence, has reached a high level of maturity in the last century.

In this chapter we describe our approach to domain knowledge construction and usage that - by embracing and adapting the faceted approach - is centered on the fundamental notions of *domain* (as originated in LIS) and *context* (as originated in AI). Domains allow capturing the different aspects of knowledge and allow scaling as with them it is possible to add new knowledge at any time as needed. At run-time, context allows a better disambiguation of the terms used and reducing the complexity of reasoning. In our diversity-aware knowledge base, built by following a precise methodology and principles, we represent classes, entities, attributes and relations between them at three different levels, i.e. the natural language (the way in which they are lexicalized), the formal language (the way in which they are formalized) and the knowledge level (codifying what is known about them).

The rest of the chapter is organized as follows. Section 6.1 provides the state of the art in knowledge bases and methods followed for their construction. Section 6.2 describes the faceted approach since its origins. Section 6.3 describes the fundamental notions of diversity and context and the solution articulated in three steps that we propose to support semantic tasks. Section 6.4 provides our definition of domain and corresponding data model. Section 6.5 describes the main steps and the guiding principles that we follow for the construction of domain knowledge. Section 6.6 shows how the proposed solution applies to semantic matching. Finally, Section 6.7 describes our first steps towards the creation of the diversity-aware knowledge base.

6.1. Knowledge bases and approaches followed for their construction

In the last forty years many projects have aimed at constructing knowledge bases. DENDRAL [Buchanan and Lederberg, 1971] is widely considered the first expert system ever created embedding a knowledge base with domain specific knowledge (organic chemistry). We can divide knowledge bases into two main broad categories: (a) automatically built and (b) hand-crafted.

Among the projects aiming at automatic extraction of knowledge (mainly unary and binary predicates) from free-text we can mention for instance KnowItAll [Etzioni et al., 2004] and TextRunner [Banko et al, 2007]. However, since working in open scenarios is extremely difficult, these techniques typically achieve limited accuracy. For this reason, projects like DBPedia [Auer et al., 2007] and YAGO [Suchanek et al., 2011] that extract information from semi-structured knowledge sources (mainly Wikipedia infoboxes and categories) obtain more accurate results. In particular, while in general these systems lack explicit quality control systems and semantics, in YAGO this is achieved through an explicit quality control mechanism mainly based on a unique entity reference system (there cannot be two entities with the same name) and type checking routines on the domain and range of the predefined binary predicates. Moreover, in YAGO there is a precise knowledge representation model based on RDFS[23]. In its 2009 version[24], it contains around 2.5 million entities and 20 million facts.

Among hand-crafted resources it is worth mentioning CYC [Matuszek et al., 2006] that is a general-purpose common sense knowledge base containing around 2.2 million assertions and more than 250,000 terms about the real world. Its open source version OpenCYC contains 306,000 assertions and 47,000 terms. Organized according to the generality principle [McCarthy, 1987], the content of CYC is distributed along three levels from broader and abstract knowledge (the upper ontology) and widely used knowledge (the middle ontology) to domain specific knowledge (the lower ontology). Similarly to CYC, SUMO (Suggested Upper Merged Ontology) [Pease et al., 2010] is a free formal ontology of about 1,000 terms and 4,000 definitional statements. Its extension, called MILO (MId-Level Ontology), covers individual domains, comprising overall 21,000 terms mapped with WordNet and 73,000 axioms. Both SUMO and MILO are therefore quite small.

[23] http://www.w3.org/RDF/
[24] http://www.mpi-inf.mpg.de/yago-naga/yago/downloads_yago.html

Neither in DBPedia nor in YAGO there is an explicit notion of domain. Everything is codified in terms of generic facts between entities (triples of the form source-relation-target). Notice that both in DBPedia and YAGO the entities can be anything, i.e. they include individuals, classes and attribute values. Moreover, both have the disadvantage that their different released versions are not aligned, i.e. there is no direct way to map the same fact or entity in different versions. In CYC there is a notion of domain, but it is used only to partition knowledge into easier to manage components. Moreover, in CYC, too, there is a generic notion of entity.

Even if not specifically developed for supporting reasoning tasks, WordNet [Fellbaum, 1998] - as demonstrated by the thousands of citations - is the most widely used linguistic resource nowadays. This is mainly due to the fact that it is manually constructed and exhibits a significant quality and size. For this reason it is also frequently adapted for semantic applications. However, one of its main drawbacks is that it is not tailored for any particular domain. Moreover, it is often considered too fine grained to be really useful in practice [Mihalcea and Moldovan, 2001]. Multilingual extensions of WordNet include MultiWordNet[25] and EuroWordNet[26].

Other valuable resources can be found in digital library communities, especially as regards domain specific knowledge encoded in informal or semi-formal knowledge organization systems such as subject headings and thesauri. For instance, about agriculture we can mention AGROVOC[27] and NALT[28]; about medicine the most widely known is UMLS. In general their main drawback is the lack of an explicit semantics (see for instance [Soergel et al., 2004]).

Hand-crafted resources are surely more accurate but difficult to construct and maintain. To alleviate this problem, some recent projects like Freebase [Bollacker et al., 2008] follow a collaborative approach by leveraging on volunteers to fill the knowledge base. Here the main focus is on named entities. Freebase however, does not make any effort to guarantee consistency in the use of the terminology and leaves its users free to independently define their axioms without enforcing effective mechanisms for duplicate detection or quality control.

[25] http://multiwordnet.fbk.eu
[26] http://www.illc.uva.nl/EuroWordNet
[27] http://aims.fao.org/website/AGROVOC-Thesaurus/sub
[28] http://agclass.nal.usda.gov/

6.2. The faceted approach to domain construction

LIS provides a solid theoretical and historicized background in domain knowledge construction and information categorization. In particular, we focus on *category based subject indexing systems*, i.e. systems which allow indexing documents by *subjects* (short strings following a simple syntax that denote what the documents are about [Battacharyya, 1975]) in a classification scheme of a few fundamental categories.

Even if the idea is probably even older, Kaiser (as it is reported for instance in [Dousa, 2007]) is most likely the first who formulated a category based subject indexing system in its *Systematic Indexing* [Kaiser, 1911]. He is widely considered the precursor of the key principles of *faceted classifications*, i.e. categorization systems where the classifications - grouped into fundamental categories - encode different aspects or *facets* of the domain knowledge. Three fundamental categories to group terms are proposed: *Concrete*, *Process* and *Country*. Since not all combinations of the terms are meaningful, he also provided a simple syntax constituted by three kinds of statements: *Concrete-Process*, *Country-Process* and *Concrete-Country-Process*. During the categorization, basic constituents of each subject were identified by using a sort of semantic factoring of complex subjects into basic classes. Later, this process has been recognized to be important for the detection of structural relationships between concepts [Soergel, 1972].

A whole range of generally applicable tables, for categories like *Place*, *Time*, *Form*, *Language* and *Point of view*, were also introduced by Otlet and La Fontaine in 1905 within the first edition of the Universal Decimal Classification[29] (UDC) [Broughton, 2006].

The Indian librarian Ranganathan was the first who proposed and formalized a theory of *facet analysis*, which is widely recognized as the fundamental methodology that guides in the creation of a faceted classification for a domain (see for instance [Broughton, 2006], [Broughton, 2008]). He developed his first faceted classification scheme (the Colon Classification) in the late 1930's. He proposed five main fundamental categories *Personality*, *Matter*, *Energy*, *Space* and *Time*, plus facets of general applicability called *common isolates* or *modifiers* (e.g. Language and document Form). Table 10 provides a small example for the medicine domain.

[29] http://www.udcc.org/

Entity

- Body and its organs
  - Cell
  - Tissue
  - Lower extremity
    - Toe
    - Foot
    - Leg
  - Head
- Digestive system
- Circulatory system
- Nervous system
- Respiratory system
  - Nose
    - Outer nose
    - nasal
  - Larynx
  - Trachea
  - Bronchi
  - Lung
  - Pleural sac
  - Mediastinum

Property

- Obstetrics
- Disease
  - General
  - Infection
    - Tuberculosis
    - Virus
    - Bacteria
  - Parasite
  - Poison
- Functional disorder
- Nutrition

Disease modifier

- Infectious
- Viral
- Bacterial
- Fungal

Action

- Nursing
- Symptom and diagnosis
  - Clinical
  - Physical
  - Microscope
  - X-ray
  - Chemical
- Pathology
- Therapeutics
- Surgery

Table 10. A small example of the medicine domain and its modifiers taken from the Colon Classification

According to the *analytico-synthetic* approach [Ranganathan, 1965], [Ranganathan, 1967], facets for a given domain are defined following two steps:

- **Analysis**. Relevant terms of the domain are identified. They can be gained by consulting domain experts and all sorts of information sources about the domain. This process starts in the so called *idea plane*, the language independent conceptual level, where atomic concepts are identified. Each identified concept, in turn, is expressed in the *verbal plane* in a given language, for example in English, trying to articulate the idea *coextensively*, namely identifying a term which exactly and unambiguously expresses the concept;

- **Synthesis.** Identified terms (also called *isolate ideas*) are progressively categorized into *facets* according to their distinguishing *characteristics*. Terms sharing the same charac-

teristic are put at the same level in the hierarchy and form what is called an *array* of homogeneous terms. The arrangement must follow a meaningful and helpful sequence, i.e. in a way to make easier the identification of the right piece of information, for instance as regards the identification of books on the shelves.

It is the collection of these facets that constitutes the faceted classification. For example, in the medicine domain, the terms *Nose, Larynx, Trachea, Bronchi, Lung, Pleural sac, Mediastinum* form a facet called *Respiratory system* (these entities are in the *part-of* relation with *Respiratory system*). The terms *Outer nose* and *Nasal*, which are *part-of* Nose, can form a facet called *Nose* which will be treated as *sub-facet* of the facet *Respiratory system*.

As described in [Giunchiglia et al., 2009a], facets possess the essential properties listed below:

- **Hospitability**. They are easily extensible. New terms representing new knowledge can be accommodated without difficulty in the hierarchical structure. Terms in the hierarchies are clearly defined, mutually exclusive and collectively exhaustive;
- **Compactness**. Facet-based systems need less space to classify the universe of knowledge with respect to the other hierarchical knowledge organization systems. There is no explosion of the possible combinations as the basic elements (facets) are taken in isolation; they allow what in libraries is called *post-coordination* of the subjects, i.e. the possibility to build subjects by using the concepts in the facets as building blocks (what Ranganathan calls the *meccano* property). This is in contrast with *pre-coordination* where all the subjects have to be listed exhaustively;
- **Flexibility.** Hierarchical knowledge organization systems are mostly rigid in their structure, whereas facet based systems are flexible in nature;
- **Reusability**. A facet-based ontology developed for a particular domain could be partially usable into another related domain;
- **Clear, but rigorous, structure**. The faceted approach aims at the identification of the logical relations between concepts and concepts groups. Sibling concepts must share a common characteristic;
- **The methodology**. A strong methodology for the analysis and categorization of concepts along with the existence of reliable rules for synthesis is provided;

- **Homogeneity.** A facet represents a homogeneous group of concepts, according to the specified common characteristic(s).

When a generic collection of documents has to be organized, facets of the corresponding domain are used as building blocks for the construction of the most suitable indexing classification for them. This classification can be used both for shelf arrangement and digital indexing purposes. When used to search, each path in the indexing classification corresponds to an entry in the so called associative or chain index. This is at the basis of systems like Ranganathan's *Chain Indexing*, Bhattacharyya's *Postulate-based permuted Subject Indexing* (POPSI) [Battacharyya, 1975] and Devadason's *Deep Structure Indexing System* [Devadason, 2002].

Underestimated for years, facet analysis is at the basis of modern classification systems such as the second edition of the Bliss Bibliographic Classification (BC2)[30] and projects like FAKTS [Broughton and Slavic, 2007], a project which attempts to provide facets useful in online environments by reorganizing BC2 and UDC auxiliary tables. It is more and more used in several other traditional classification systems for the definition of facets of general use as an add-on to the standard classification schemes. It is also used as a guideline for the generation of thesauri since it helps in the identification of terms and relationships between them [Broughton, 2008]. In effect, some researchers argue that faceted classifications are not a particular kind of library classification, but rather the only viable form enabling the locating and relating of information to be optimally predictable [Mills, 2004]. Therefore, the principles at the basis of the faceted approach can be applied to different systems.

As described for instance by Soergel in [Soergel, 1972], an analogous technique, but from opposite perspective, called *semantic factoring* consists in decomposing complex subjects into elementary ones, as a way to identify and consequently generate auxiliary multi-hierarchical indexing structures useful to index existing *highly compound* subject heading catalogues. Using semantic factors makes easier the access to information otherwise difficult to locate through the direct use of complex subjects. He suggests the generation of these hierarchies also as a way to promote interoperability between different schemes. These ideas have been resumed more recently in the FAST project [Dean, 2003] which defines a faceted vocabulary for LCSH.

[30] http://www.blissclassification.org.uk/

In the recent years, in the attempt of enhancing user experience, faceted classifications are becoming very popular, especially in the Web environment. Systems making use of faceted classifications are known as *faceted systems*. However, in these systems rather than classifying documents by their subject, they classify objects (e.g. products in e-commerce websites) by their metadata (e.g. by prize, by dimension). In this respect, the main contribution to faceted systems is definitely represented by the Flamenco project (see for instance [Yee et al., 2003]). They offer a suite for the creation of Web interfaces providing classical keyword search and faceted navigation over a collection of objects. Facets are typically exposed on the Web page. Users can combine the two approaches by choosing at each step alternatively between navigation and search. Studies, such as [Marchionini and Shneiderman, 1988], [Bates, 1990], [Marchionini, 2006], [White et al., 2006], [White et al., 2007], demonstrate that combining search and browsing capabilities in an explorative approach can greatly help users in finding relevant information without forcing them to follow a unique searching paradigm. Selected choices remain visible to be able to discard/expand them at any moment as desired. This implements a sort of query refinement and query expansion device. This mechanism is at the basis of *exploratory search* techniques.

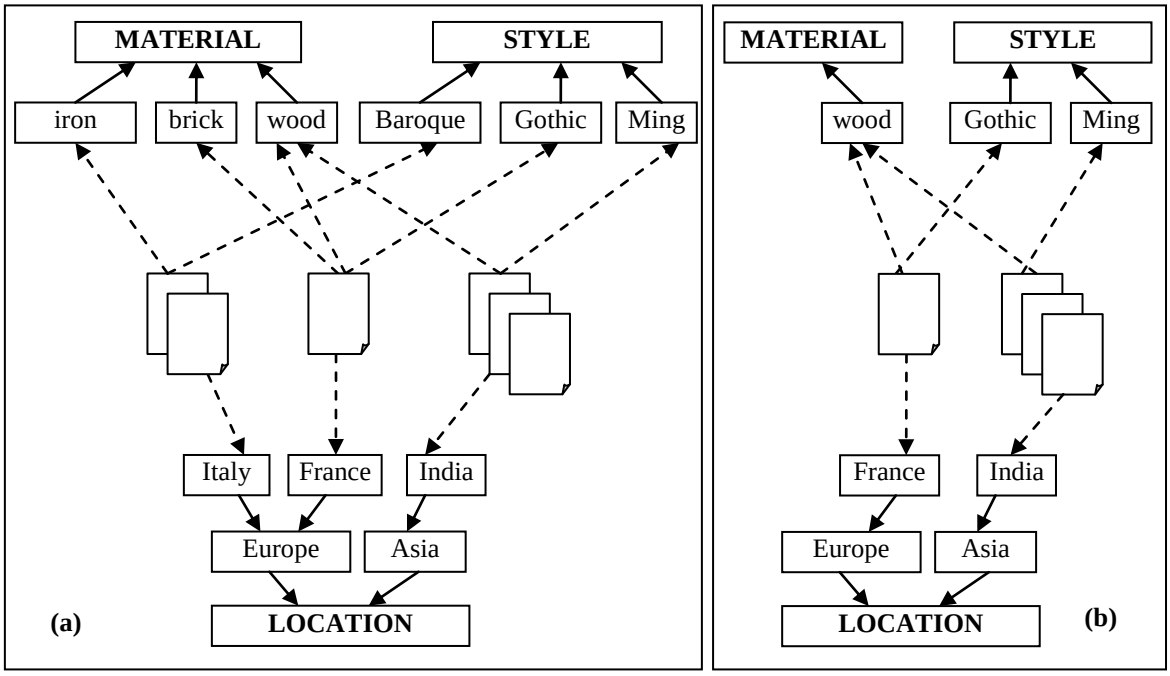


Fig. 30. (a) An example of faceted classification before the selection of nodes; (b) The faceted classification is modified after the selection of the value *wood* from the *Material* facet.

Fig. 30 shows an example, adapted from [English et al., 2003], where a collection of documents about pieces of art is indexed according to the facets Material, Style and Location. They can be

seen as distinctive properties of pieces of art. Each facet can be seen as an alternative way to access and navigate the same data. At seeking time, the selection of a node along a path from a facet permits to progressively reduce the amount of selected objects to only those sharing the corresponding attribute value. Consequently, the aspect of the facet hierarchies is modified by *pruning* all paths that do not index selected objects.

Usability tests [Kules et al., 2009] show that facets play an important role in supporting exploratory search processes. Nevertheless, some critiques have been moved against these systems. For instance, La Barre [La Barre, 2004] believes that current faceted systems realize only a very small portion of the ideas provided by LIS. She also emphasizes how there is often confusion in the way the original notions are interpreted and applied.

6.3. Diversity in knowledge

Semantics is core in many knowledge management applications, such as natural language data and metadata understanding [Schwitter and Tilbrook, 2006], [Fuchs et al., 2006], [Zaihrayeu et al., 2007], [Autayeu et al., 2010], natural language driven image generation [Adorni et al., 1984], abstract reasoning [Giunchiglia and Walsh, 1989] [Giunchiglia et al., 1997], converting classifications into formal ontologies [Bouquet et al., 2003a], [Magnini et al., 2003], [Giunchiglia et al., 2007a], automatic classification [Giunchiglia et al., 2007b], ontology matching [Giunchiglia et al., 2005] [Giunchiglia et al., 2007c] and semantic search [Giunchiglia et al., 2009d]. However, despite the progress made, one of the main barriers towards the success of these applications is the lack of background knowledge. In fact, as underlined by several studies - see for instance [Magnini et al., 2004] [Giunchiglia et al., 2006] [Lauser et al., 2008] [Shvaiko and Euzenat, 2008], [Aleksovski et al., 2008] - without high quality and contextually relevant background knowledge it is impossible to achieve accurate enough results.

Dealing with this problem has turned out to be a very difficult task. In fact, on the one hand, in order to provide all the possible meanings of the words and how they are related to each other, the background knowledge should be very large and virtually unbound. On the other hand, the background knowledge should be context sensitive and able to capture the diversity of the world. The world is in fact extremely diverse and diversity is visibly manifested in language, data and knowledge. The same real world object can be referred to with many different words in different communities and in different languages. For instance, it is widely known that in some Nordic circumpolar groups of people the notion of *snow* is denoted with hundreds of different words in the local language carrying very fine grained distinctions [Norsk Polarinstitutt, 2005]. This phenomenon is often a function of the role and importance of the real world object in the life of a community. Conversely, the same word may denote different notions in different domains; for instance, *bug* as insect in entomology and *bug* as a failure or defect in a computer program in computer science. Space, time, individual goals, needs, competences, beliefs, culture, opinions and personal experience also play an important role in characterizing the meaning of a word. Diversity is an unavoidable and intrinsic property of the world and as such it cannot be avoided. At the same time, diversity is a local maximum since it aims at minimizing the effort and maximizing the gain [Giunchiglia, 2006].

Consider for instance the example of classifications given in Section 2.2, Fig. 4. In approaching semantic tasks, we should consider that diversity emerges at least along three main dimensions:

- ***Diversity in natural language***: terms may denote classes (common nouns), entities (proper nouns), relations, attributes and other modifiers (adjectives and adverbs); different terms can be used to denote the same notion (synonymy), e.g. the term *location* in the first classification and the term *place* in the second; the same term may denote different things (polysemy), e.g. the term *bank* in the first classification may mean a sloping land or a financial institution. At the entity level, *Rome* the capital of Italy is also known as the *Eternal City*; there might be different places in the world (and in general different entities) called *Rome*;
- ***Diversity in formal language***: when disambiguated, each term corresponds to a concept written in some formal language. Different classifications, according to their specific scope and purpose, may use different formal languages. For instance, while for somebody it might be enough to distinguish between mountains and rivers, some others may need to further distinguish between mountains and hills, rivers, creeks and rivulets or even between oversea and undersea mountains.
- ***Diversity in knowledge***: at this level the relations between concepts are recognized. The amount of knowledge, in terms of axioms, necessary for a certain task is also a function of the local goals, culture, opinions and personal experience. For instance, while dogs are mainly perceived as pets, they are regularly served as food in China (culture); while someone may consider beautiful the city of Rome in Italy, somebody else may consider it too chaotic (opinion); somebody may consider climate change an urgent problem to be solved, while somebody else may even negate its existence (school of thought).

The ambiguity of natural language is a critical issue in the conversion of classifications into lightweight ontologies. In this respect, it is fundamental to identify resources providing the background knowledge relevant for the disambiguation. However, the meaning of the words and the context of use is almost always left implicit. This implicit knowledge, or *implicit assumptions* (as they have been called in [Giunchiglia, 2006]), is what allows their meaning to be determined. In other words, implicit assumptions constitute what is relevant and necessary to disambiguate and understand the labels. It is also quite intuitive and important to note that the amount of implicit knowledge is potentially infinite. As a consequence, it is quite never possible or desirable to completely determine them. A considerable portion of knowledge remains in the human minds

[Prusak, 1997]. Our approach is to take into account this diversity and exploit it to *make explicit the local semantics*, i.e. the meaning of words in a certain context, such that information becomes unambiguous to humans as well as to machines.

The second problem we should consider is that by increasing the size of the background knowledge, reasoning can become arbitrarily complex. It is therefore fundamental to reduce the number of axioms that we use to reason about to only those relevant to solve the problem.

As described in the introduction to this chapter, our approach is centered on the fundamental notions of *domain* (as originated in LIS) and *context* (as originated in AI). Domains are the main means by which diversity is captured (in terms of language, knowledge and personal experience) and allow scaling as with them it is possible to add new knowledge at any time as needed. Context allows on the one hand a better disambiguation of the terms used (i.e. by making explicit some of the assumptions left implicit) and on the other hand, by selecting from the domains the language and the knowledge which are strictly necessary to solve the problem, it allows reducing the complexity of reasoning at run-time. More in detail, our proposed solution consists in addressing the problem in three steps:

1. Develop an extensible diversity-aware knowledge base explicitly codifying the differences in (natural and formal) language and knowledge in multiple domains;
2. Given the specific problem, build the corresponding *context* as a formal local theory by (2.1) determining from the knowledge base the implicit assumptions which are relevant to understand it and (2.2) building the corresponding context as a logical theory;
3. Solve the problem in context.

Developing a diversity-aware knowledge base requires appropriate methodologies for its representation, construction and maintenance. At this purpose we propose and adapt the *faceted approach*. In the rest of the section we describe how we have been doing that in terms of data model (Section 6.4), methodology and principles to domain construction (Section 6.5).

Concerning the notion of context, the first formal theories were independently proposed by McCarthy [McCarthy, 1993] and Giunchiglia [Giunchiglia, 1993].

According to McCarthy, contexts are a way to *partition* knowledge into a limited set of locally true axioms with common assumptions. This set of axioms should be at the right level of abstraction thus excluding irrelevant details in order to simplify local reasoning as much as possible. This is known as the generality principle [McCarthy, 1987]. In this setting, it is always possible to *lift* from the local context to a more general one by progressively making explicit the assumptions. This allows, among other things, *integrating* two or more contexts under the umbrella of a more general theory, thus assuming that a unique global schema can be always reconstructed. This process is called *relative decontextualization*. CYC is an example of knowledge base following this approach (see for instance [Guha and Lenat, 1993]) as a way to partition huge quantities of common sense knowledge into smaller, easier to manage, sets of axioms.

According to Giunchiglia, context is a tool to specifically *localize* reasoning to a subset of facts known by an agent [Bouquet et al., 2003b]. This is motivated by the intuition that reasoning is always local and always represents a partial approximate theory and subjective view of the world. Unlike McCarthy, in this view each context typically has its own language, grammar and theory, thus leading to the maximum level of local autonomy. Moreover, the existence of a common global schema is not guaranteed. However, taking into account implicit assumptions, it might be possible to (partially) *relate compatible* axioms in distinct contexts [Ghidini and Giunchiglia, 2001]. These relations are the basis for interoperability.

By extending [Giunchiglia, 2006], we define a context as follows.

Definition 11 (Context). A context *ctx* is a 4-tuple $<id, L_c, K_c, IA>$, where:

- *id* is an identifier for the context
- L_c is the local (formal) language
- K_c is the local knowledge
- *IA* is a set of implicit assumptions

6.4. A domain-centric data model

The first step towards the creation of the diversity-aware knowledge base is the definition of the corresponding data model. With this purpose, we follow and adapt the *faceted approach*.

In the original LIS approach, since the purpose is to classify bibliographic material, facets are classification ontologies, i.e. each concept in the ontology denotes a set of documents while links between concepts denote subset relations [Giunchiglia et al., 2009a] [Maltese and Farazi, 2011]. As we emphasize in [Giunchiglia et al, 2012], the major drawback of the original approach stands in the fact that it fails in making explicit the way the meaning (semantics) of subjects is built starting from the semantics of their constituents. In fact, they only consider the syntactic form by which subjects are described in natural language (syntax). Consequently, they do not allow for a direct translation of their elements - terms and arcs in the facets - into a formal language, e.g. in form of DL axioms. They do not explicitly specify the taxonomical *is-a* and *instance-of* (genus/species) and mereological *part-of* (whole/part) relations between the classes, thus limiting their applicability. In particular, making them explicit is a fundamental step towards automation and interoperability.

To overcome these limitations, in our approach we define facets as descriptive ontologies. As described in Section 2.1.3, this also allows minimizing maintenance costs, serving different kinds of applications, and ensures maximum reusability since it allows efficiently computing both the real world and classification semantics version of the transitive closure as needed by the application.

We define a *domain* as follows.

Definition 12 (Domain). A domain *D* is a 4-tuple <C, E, R, A>, where

- C is a set of classes
- E is a set of entities
- R is a set of binary relations
- A is a set of attributes.

These sets correspond to what in the faceted approach are called *fundamental categories*. More in detail:

- **C**: Elements in C denote classes of real world objects
- **E**: Elements in E represent the instances of the classes in C
- **R**: The set R provides structure to the domain by relating entities and classes. It includes the canonical *is-a* (between classes in C), *instance-of* (associating instances in E to classes in C) and *part-of* (between classes in C or between entities in E) relations and is extended with additional relations according to the purpose, scope and subject of the ontology. We assume *is-a* and *part-of* to be transitive. Since they constitute the backbone of the facet hierarchies, *is-a* and *part-of* relations are said to be *hierarchical*. Other relations are said to be *associative*. Among other things, they allow elements from different facets to be connected.
- **A**: Elements in A denote qualitative/quantitative and descriptive attributes of the entities. We further differentiate between attribute names and attribute values. Each attribute name in A denotes a relation associating each entity to corresponding attribute values. With this purpose, we also define a *value-of* relation that associates each attribute name to the corresponding set of possible values (the range of the relation).

Within each fundamental category, we organize each domain (e.g. *Space*) in three levels:

- ***Formal language level***: it provides the terms used to denote the elements in C/E/R/A. We call them *formal terms* to indicate the fact that they are language independent and that they have a precise meaning and role in (logical) semantics. Each term in C denotes a class (e.g. *lake, river* and *city*). Each term in E denotes an entity (e.g. *Garda lake*). Each term in R represents the name of a relation (e.g. *direction*). Each term in A denotes either an attribute name (e.g. *depth*) or an attribute value (e.g. *deep*). Elements in C, R and A are arranged into facets using *is-a*, *part-of* and *value-of* relations.
- ***Knowledge level***: it codifies what is known about the entities in E in terms of attributes (e.g. *Garda lake* is *deep*), the relations between them (e.g. *Tiber* is part of *Rome*) and with corresponding classes (e.g. *Tiber* is an instance of *river*). Terms in E are at the leaves of the facets and populate them. The knowledge level is codified using the formal language described in the item above and is, therefore, also language independent;
- ***Natural language level***: we define a natural language as a set of words (i.e. strings), that we also call *natural language terms*, such that words with same meaning within each

natural language are grouped together and mapped to the same formal term. This level can be instantiated to multiple languages (at the moment only to English and Italian);

Similarly to WordNet and following same terminology, words are disambiguated by providing their meaning, also called *sense*. The meaning of each word can be partially described by associating it a natural language description. For instance, *stream* can be defined as *"a natural body of running water flowing on or under the earth"*. Within a language, words with same meaning (synonymy) are grouped into a *synset*. For instance, since *stream* and *watercourse* have the same meaning in English, they are part of the same synset. Given that a word can have multiple meanings (homonymy), the same word can correspond to different senses and therefore belong to different synsets. For instance, the word *bank* may mean *"sloping land (especially the slope beside a body of water)"*, *"a building in which the business of banking transacted"* or *"a financial institution that accepts deposits and channels the money into lending activities"*. In our data model, within a language each synset is associated a set of words (the synonyms), a natural language description, a part of speech (noun, adjective, verb or adverb) and a corresponding formal term.

In each domain we clearly separate the elements of C/R/A that provide the basic terminology, from those in E that provide the instantiation of the domain. The data model we propose has a direct formalization in DL. In fact, classes correspond to concepts, entities to instances, relations and attributes to roles. The formal language level provides the TBox, while the knowledge level provides the ABox for the domain. They correspond to what people call the *background knowledge* [Giunchiglia et al., 2006], i.e. the a-priori knowledge which must exist to make semantics effective. Each facet corresponds to what in logics is called *logical theory* [Giunchiglia et al., 1997] and to what in computer science is called *ontology*, or more precisely *lightweight ontology*, and plays a fundamental role in task automation (formal reasoning). The natural language level provides instead an interface to humans and can be exploited for instance in Natural Language Processing (NLP).

Below we provide the corresponding formalization into DL in the real world semantics.

Domain of interpretation

The domain of interpretation $D = F \cup G$ where:

- F is a set of individuals
- G is a set of attribute values (that can be further partitioned in different data types)

Entities

For all $e \in E$, $I(e) = e^I \in F$

Classes

For all $c \in C$, $I(c) = c^I \subseteq F$

Relations

Relations are formalized as follows:

- is-a corresponds to subsumption, i.e. given $c, d \in C$ such that is-a(c,d) we add $c \sqsubseteq d$ to the TBOX that means $c^I \subseteq d^I$;
- instance-of corresponds to concept assertions, i.e. given $e \in E$, $c \in C$ such that instance-of(e,c) we add c(e) to the ABOX that means $e^I \in c^I$;
- Other relations (but the value-of, see below) correspond to DL roles and corresponding role assertions.

In general for all $r \in R$, $I(r) = r^I \subseteq F \times F$. When defined between classes and given $c, d \in C$ such that r(c,d) we add $c \sqsubseteq \exists r.d$ to the TBOX with the usual semantics. When defined between entities and given $e, f \in E$ such that r(e,f) we add r(e,f) to the ABOX that means $(e^I, f^I) \in r^I$

Attributes

- Attribute values correspond to values in the domain of interpretation, i.e. for all $av \in AV$, $I(av) = av^I \in G$;
- Attribute names correspond to DL roles and corresponding role assertions. In general, for all $an \in AN$, $I(an) \subseteq F \times G$.

Given e ∈ E and av ∈ AV such that an(e, av) we add an(e, av) to the ABOX that means $(e^I, av^I) \in an^I$. Intuitively, this means that the individual e has attribute an with value av.

value-of

For all an ∈ AN, value-of is a special function that restricts the set of possible values of an to a subset V of AV. Therefore if value-of: an → V, for all c ∈ C we add c ⊑ ∀an.V to the TBOX with the usual semantics.

By default we assume c ⊑ ∃an.V for all c ∈ C and an ∈ AN (with constraints imposed by the value-of on an and corresponding V). However, we are working on an entity type theory which allows posing restrictions on the kinds of attributes that can be assigned to entities of a certain class.

As an example, Fig. 31 provides a small fragment of the *Space* domain following the proposed data model, where classes are represented with circles, entities with squares, relation names with hexagons, attribute names with trapezoids and attribute values with stars. Letters inside the nodes (capital letters for entities and small letters for classes, relations and attributes) denote formal terms, while corresponding natural language terms are provided as labels of the nodes. For sake of simplicity, synonyms are not given. Arrows denote relations between the elements in C/E/R/A; solid arrows represent those relations constituting the facets (*is-a*, *part-of* and *value-of* relations) and which are part of the formal language level; dashed arrows represent *instance-of*, *part-of* and the other relations (*depth* in this case) which are part of the knowledge level. Here the hierarchies rooted in *body of water*, *populated place* and *landmass* are facets of entity classes and are subdivisions of *location*, the one rooted in *direction* is a facet of relations and the one rooted in *depth* is a facet of attributes.

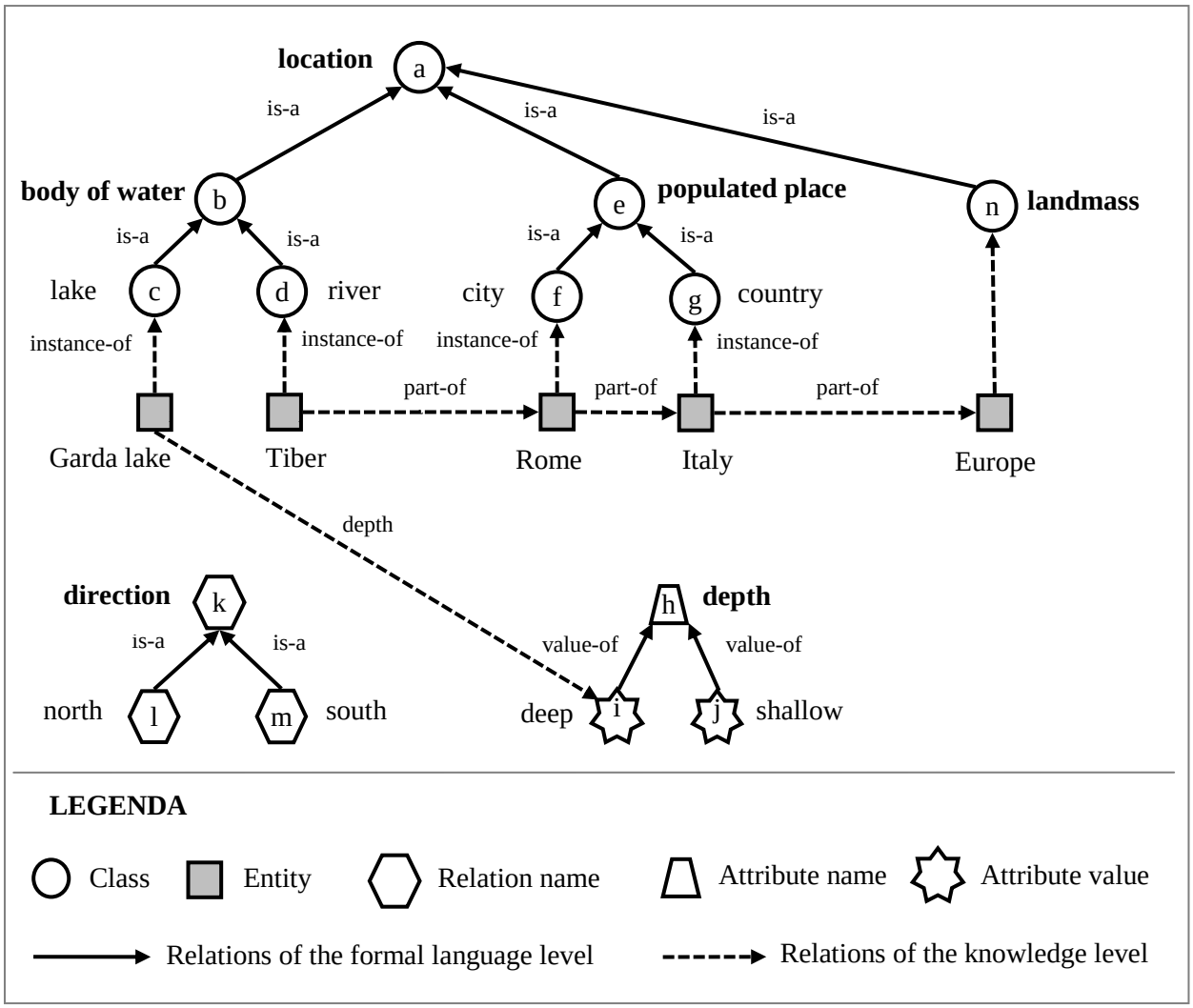


Fig. 31 - A small fragment of the *Space* domain following the proposed data model

6.5. A facet-based methodology to domain construction

6.5.1 Methodology

The process to build a faceted ontology is organized in five subsequent phases: Identification of the terminology, Analysis, Synthesis, Standardization and Ordering. Let us describe them in turn.

Step 1: Identification of the terminology. It consists in collecting and classifying the natural language terms. In general, in the faceted approach this is mainly done by interviewing domain experts and by reading available literature about the domain under examination including *inter-alia* indexes, abstracts, glossaries, reference works. Analysis of query logs, when available, can be extremely valuable to determine user's interests. In our approach, each natural language term is analyzed and disambiguated by reconstructing the corresponding sense, by grouping those with same meaning into synsets, and by associating each synset to a formal term. Each formal term is then classified as a class, entity, relation or attribute (name or value).

Step 2: Analysis. The formal terms collected during the previous phase are analyzed per *genus et differentia*, i.e. in order to identify their commonalities and their differences. The main goal of the analysis is to identify as many characteristics as possible of the real world entities represented by each of the terms. This allows being as fine grained as wanted in differentiating among them. For instance, for the term *river*, defined as *"a large natural stream of water (larger than a brook)"*, we can identify the following characteristics: a body of water; a flowing body of water; no fixed boundary; confined within a bed and stream banks; larger than a brook.

Step 3: Synthesis. With the synthesis, formal terms are arranged into facets. This is done by referring to their lexicalization in a language, e.g. to the corresponding English or Italian synsets, and according to the characteristics identified with the previous phase. Following the principles described in the next section, the levels of the facet hierarchies are progressively formed by grouping terms into arrays by a common characteristic.

Step 4: Standardization. For each formal term in a facet, a standard (or preferred) term should be selected among the natural language terms associated to the corresponding synset. In the faceted approach this is usually done by identifying the term which is most commonly used in the

domain and which minimizes the ambiguity. This is similar to the WordNet approach where words are ranked in the synset. The first word is the preferred one. For instance, the term *building* (defined as *"a structure that has a roof and walls and stands more or less"*) is more commonly used than the term *edifice*.

Step 5: Ordering. Formal terms in each array are ordered. There are many criteria one may follow, e.g., by chronological order, by spatial order, by increasing and decreasing quantity (for instance by size), by increasing complexity, by canonical order, by literary warrant and by alphabetical order. The criteria should be based upon the purpose, scope and subject of the ontology.

6.5.2 Guiding principles

Ranganathan provided a huge amount of principles and canons to be used to build facets. Inspired by them, we propose a minimal set of guiding principles:

1. ***Relevance***. The selection of the characteristics that are used to form the facets should reflect the purpose, scope and subject of the ontology. For example, while in the context of the *Space* domain the characteristic *by populated cluster group* is appropriate to group villages, cities and towns, it is instead not suitable to classify state capitals, provincial capitals and national capitals. In fact, in the latter case the characteristic *by seat of government of a political entity* would be more realistic and appropriate. It is worthwhile also noting that the selection of the characteristics should be done carefully, as they cannot be changed unless there is a change in the purpose, scope and subject of the ontology.

2. ***Ascertainability***. Characteristics must be definite and verifiable. For example, the characteristic *flowing body of water* for rivers can be ascertained easily from the scientific literature and from the geo-scientists.

3. ***Permanence***. Each characteristic should reflect a permanent quality of an entity. For example, a *spring* (*"a natural flow of ground water"*) is always a flowing body of water, thus the facet *flowing body of water* represents a permanent characteristic of *spring*.

4. ***Exhaustiveness***. Terms in each array should be totally exhaustive w.r.t. their respective common parent term in the facet hierarchy. For example, to classify the bodies of water based on the *water movement*, we need both *flowing body of water* and *stagnant body of water*. If we miss any of these two, the classification becomes incomplete.

5. ***Exclusiveness***. All the characteristics used to classify a term must be *mutually exclusive*, i.e. no two facets can overlap in content. For example, the bodies of water cannot be classified by both the characteristics *inland body of water* and *water movement*, as they would produce the same division for bodies of water such as lakes, rivers and ponds.

6. ***Context***. The position of a formal term in the ontology is a function of its meaning. This principle is particularly helpful to distinguish among homonyms. For instance, in order to distinguish between the following two meanings of bank:
   - bank, sloping land (especially the slope beside a body of water)) "they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"
   - bank - a building in which the business of banking transacted; "the bank is on the corner of Nassau and Witherspoon"

   We can position them in two different facets of the ontology as follows:
   - Landform > Natural elevation > Continental elevation > Slope > Bank
   - Facility > Business establishment > Bank

7. ***Currency***. The words chosen to denote formal terms should be those of current usage in the subject field. For example, in the context of transportation systems, *metro station* is more commonly used than *subway station*.

8. ***Reticence***. The words chosen to denote formal terms should not reflect any bias or prejudice (e.g. of gender, cultural, religious) or express any personal opinion of the person who develops the ontology. For example, it is not appropriate to use words like *devils places, criminal houses* to mean the jailhouses or any other type of correctional places.

9. ***Ordering***. The order of the facets and of the terms within each facet should reflect the purpose, scope and subject of the ontology. It should be applied consistently and should not be changed unless there is a change in the purpose, scope or subject of the ontology. Ordering carries semantics as it provides implicit relations between terms within an array. For example, the facet *populated place* may include *hamlet*, *village*, *town* and *city*. They are in ascending order according to population. This ordering clearly reflects that a *hamlet* is less populated than a *village*, that a *village* is less populated than a *town*, and so forth.

6.6. Diversity-aware semantic matching

To understand the role of domains and context in semantic applications, let us revisit the problem of matching the two classifications given in Section 2.2, Fig. 4. The three steps that we suggested to address generic semantic tasks can be mapped into the four steps of the semantic matching - explained in Section 2.3.1 - as shown in Table 11.

Steps for a generic semantic task	**Steps in semantic matching**
(1) create a diversity-aware knowledge base	
(2) given the problem, build the context	
(2.1) determine the implicit assumptions	1. For all the labels in the two classifications compute the *concept at label* 2. For all the nodes in the two classifications compute the *concepts at node*
(2.2) build the context	3. For all pairs of labels in the two classifications compute the semantic relations between the concept at labels
(3) use the context to solve the problem	4. For all pairs of nodes in the two classifications compute the relations between the concepts at node

Table 11. Mapping the semantic matching problem in the general three steps

The construction of the diversity-aware knowledge base is preliminary to any semantic application. In the case of semantic matching, implicit assumptions consist of a selection of the domains from the knowledge base which are relevant to understanding the meaning of the words in a certain framework. This can be done (but this is still an open research problem) by parsing node labels and documents in classifications, linking them to the diversity-aware knowledge base and identifying the smallest set of domains in which words take a precise meaning. For instance, the analysis of the words appearing in the labels of the two classifications in Fig. 4 might reveal that the words *location*, *place*, *city* and *bank* (the root form of the words appearing in the labels) denote geographical classes, and that *Europe*, *Italy*, *Rome*, *Milan* and *Danube* are location names in the *Space* domain. Since, most of the words assume a precise meaning if interpreted in the *Space* domain we can assume that it can provide most of the implicit assumptions. The lightweight ontologies that we obtain are therefore more accurate. For instance, we might obtain those depicted in Fig. 32. Since each concept occurring in node labels correspond to a concept in the faceted

knowledge base, they correspond to what in [Giunchiglia et al., 2009a] we called *faceted lightweight ontologies*.

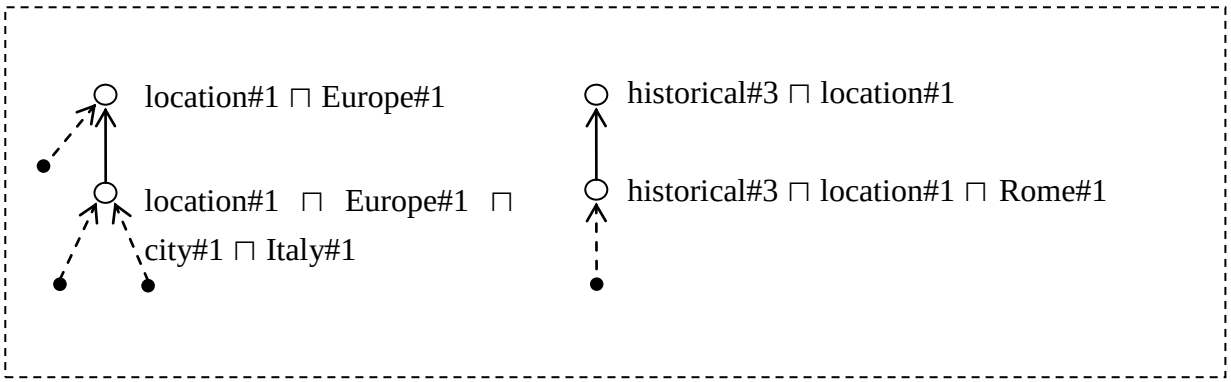


Fig. 32. The faceted lightweight ontologies constructed by using the domain knowledge

The local context *ctx* is built by selecting from the domains the language and the knowledge which are strictly necessary to solve the problem. This corresponds to the third step in the matching and it is done on the basis of the concepts that were used in the formulas at labels. L_c is the set of all atomic concepts in the formulas at labels, while K_c is built by computing the strongest semantic relation holding between each of the concepts in L_c. Our approach is similar to the work described in [Hoder and Voronkov, 2011] where the relevant knowledge is constructed by progressively expanding the set of axioms in the premises on the basis of the symbols occurring in the formula. Nevertheless, here the problem is easier given the lower complexity of reasoning (propositional). Moreover, the use of domains further mitigates the problem.

A context is therefore a logical theory over a certain language and domain of interpretation. More precisely, for the problem of matching classifications, the theory is a propositional DL theory. The FL and K of the selected domains are used as follows:

- Classes, entities, attributes and values from FL which are used in the formulas are codified as atomic concepts of the formal language L_c
- All the relations in K correspond to subsumption[31] [Giunchiglia et al., 2009a]. For all the concepts in L_c the semantic relations holding between them are selected or computed from K and added to K_c

[31] Note that for the matching problem the value-of relations (described in Section 6.4) are not used, but they play an important role in maintenance.

Finally, the problem is solved in context. For instance, to understand the meaning of the second node in the first classification (cities in Italy) and the second node in the second classification (Rome) and compute the strongest semantic relation holding between the two, reported in Fig. 33

- L_c must include a concept for each of the following terms: location, city, Europe, Italy, Rome and historical
- K_c must include the following axioms, contained (the first three) or inferred (the second three) from the background knowledge:

  Rome ⊑ city (Rome is a city)

  Rome ⊑ Italy (Rome is part of Italy)

  Italy ⊑ Europe (Italy is part of Europe)

  city ⊑ location (cities are locations)

  Rome ⊑ location (Rome is a location)

  Rome ⊑ Europe (Rome is part of Europe).

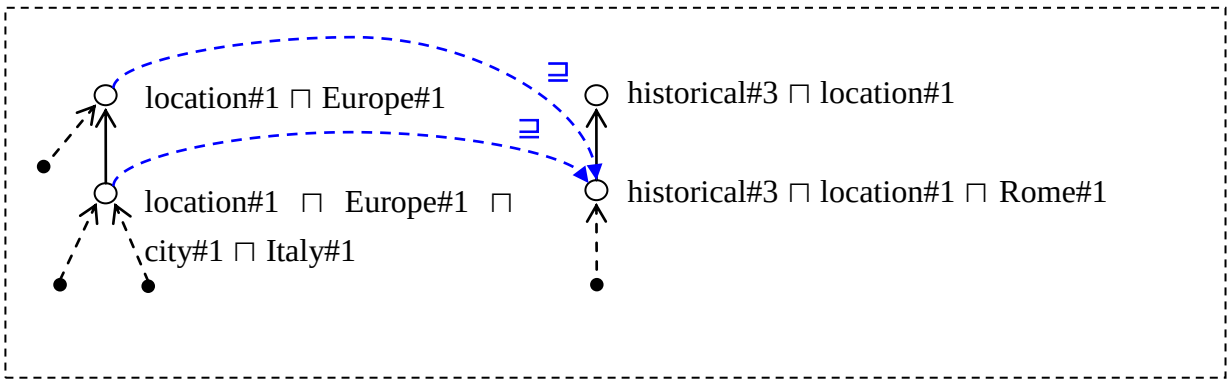


Fig. 33. The alignment between two faceted lightweight ontologies

For the matching task, the semantics associated with the formal language is the classification semantics, therefore an interpretation function I: $L_c \rightarrow P(D)$ assigns each atomic concept in the formal language to a set of documents in D. For instance, the extension of the concept *city* will be the set of documents about real world cities, while the extension of the concept *beautiful* will be the set of documents about real world beautiful objects.

6.7. Entitypedia: our diversity-aware knowledge base

Following the data model presented in Section 6.4, we have been developing a framework and a diversity-aware knowledge base with an initial set of domains necessary for the kinds of scenarios we need to serve, but - in the spirit of the proposed approach - extensible according to the local scope, purpose, language and personal experience. We called it *Entitypedia*[32]. The general strategy to incrementally populate it is described in the following. In the rest of the thesis we focus instead on the work done for the *Space* domain and one of its applications, the semantic geocatalogue of the Autonomous Province of Trento in Italy.

Phase I: bootstrapping the knowledge base. We initially populated Entitypedia with general terminology imported from WordNet 2.1 and the Italian section of MultiWordNet. This essentially provided what is needed to bootstrap the natural language level, in English and Italian, respectively[33]. The work was done in collaboration with my colleagues Ilya Zaihrayeu and Marco Marasca who worked on the definition of the data structures and Feroz Farazi and Abdelhakim Freihat who worked on the importing of WordNet and MultiWordNet.
We imported words, synsets and lexical relations between them from WordNet and MultiWordNet to the natural language part of our knowledge base, instantiated for the English and Italian language, respectively. We decided to do not import WordNet instances/entities for two main reasons. First, they are not a significant number and no attributes are provided for them. Second, we plan to import huge quantities of entities and corresponding metadata from other resources. Note that the official number of entities in WordNet is 7671 [Miller and Hristea, 2006], while we found out that 683 of them are common nouns instead. We identified the wrong ones by manually verifying those with no uppercased lemma. The wrong ones were converted into noun synsets, while the other 6988 were considered still entities. Figures are provided in Table 12. Excluding the 6988 entities and corresponding relations, WordNet was completely imported. MultiWordNet, mainly due to the heuristics used to reconstruct the mapping with WordNet 2.1, was only partially imported. In particular, we imported 92.47% of the words, 94.28% of the senses and

[32] http://entitypedia.org/
[33] These two languages were selected because of the importance that the English and Italian languages have respectively in the context of the Living Knowledge (http://livingknowledge-project.eu) and the Live Memories (http://www.livememories.org) projects we are involved in.

94.30% of the synsets. We did not import the 318 (Italian) lexical and semantic relations provided.

WordNet 2.1	
Object	**Instances**
Synset	110,609
Relation	204,481
Word	147,252
Sense	192,620
Word exceptional form	4,728

MultiWordNet	
Object	**Instances**
Synset	36,448
Relation	-
Word	41,705
Sense	63,595
Word exceptional form	-

Table 12. Data imported from WordNet 2.1 and MultiWordNet

For each synset in the two languages, a language-independent concept was created at formal language level. If the same notion can be expressed in the two languages then corresponding synsets are linked to the same concept. Since MultiWordNet is aligned with the older WordNet 1.6 version, the mapping between the two languages was reconstructed by combining the existing mapping[34] between WordNet 1.6 and 2.0 with another one we created expressly between WordNet 2.0 and 2.1 using some heuristics. Notice that for adjectives and adverbs we had to directly compute the mapping between WordNet 1.6 and 2.1 since not available elsewhere. Notice also that due to the partial coverage of the language in MultiWordNet and the well-known problem of gaps in languages (i.e. given a lexical unit in a language, it is not always possible to identify an equivalent lexical unit in another language) not all concepts have a corresponding synset in Italian. *Hypernym* (is-a) and transitive *part meronym* (part-of) relations were elected as semantic hierarchical relations (corresponding to subsumption under classification semantics). All the other relations were defined as associative relations. We plan to significantly reorganize them in the future.

Phase II: building the Space and Time domains. Given their pervasiveness and the specific scenarios we need to serve, we started populating the knowledge base with the *Space* and *Time* domains. As described more in detail in the rest of the thesis, to construct the *Space* domain we followed a semi-automatic approach. Domain specific terms were extracted mainly from GeoNames[35] and TGN[36]. Following the methodology presented in Section 6.5, these terms were

[34] http://www.cse.unt.edu/~rada/downloads.html#wordnet
[35] http://www.geonames.org/

analyzed, organized into facets and mapped with the concepts created with the previous phase. This process led to the creation of a set of facets containing overall more than 1000 concepts (still increasing in size). A significant amount of manual work was done in this phase to guarantee high quality of the data. Similarly to *Space*, the *Time* domain was built by using WordNet and Wikipedia[37]. For instance, *holidays* are grouped *by religion*. *Christian holydays* include *Easter* and *Christmas*; *Islamic holidays* include *Eid Al-Fitr* and *Eid Al-Adha*.

Phase III: populate the knowledge base with entities. 7 million entities from GeoNames were automatically imported at knowledge level in our knowledge base (see Chapter 7). As part of the S-Match open source framework, we released a significant part of this data as an open source geo-spatial ontology, that we called *GeoWordNet*[38] [Giunchiglia et al, 2010]. Notice that, in GeoWordNet we did not explicitly provide the facets. GeoWordNet - distributed in WordNet format - can be used by S-Match in alternative to WordNet as background knowledge. With an experiment, still not completed, that I am conducting with YAGO, around 600,000 additional locations as well as 700,000 persons and 150,000 organizations will be also imported.

Phase IV, next steps: building the Internet domains. Our long term goal is not to build the world knowledge (this would be too ambitious and the state of the art shows that this would be actually impossible), but to identify those domains which are more likely to play a role in everyday life and in particular on the Web. In the context of the Living Knowledge EU project[39], this has been identified as strategic towards enabling diversity-aware applications for the Web. From a preliminary analysis on the query logs of the AOL search engine[40] conducted by our partners at the Indian Statistical Institute[41] in Bangalore, a prioritized list of around 350 domains was formed. On the very top of this list we find domains such as *Space*, *Time*, *food*, *sports*, *tourism*, *music*, *movie* and *software*. We refer to them as *Internet domains*.

Some of these domains are currently under development. In particular, there are two collaborations in place with industrial partners. The first involves the SORA Institute[42], a company based in Austria specialized in statistical surveys conducted using media content analysis techniques.

[36] http://www.getty.edu/research/tools/vocabularies/index.html
[37] http://www.wikipedia.org/
[38] http://geowordnet.semanticmatching.org/
[39] http://livingknowledge-project.eu/
[40] http://search.aol.com/aol/webhome
[41] http://drtc.isibang.ac.in/DRTC/
[42] http://www.sora.at/

With them we have been developing the *political science* domain [Madalli and Prasad, 2011]. The second involves Telecom Italia[43], a well-known telecommunication company based in Italy. The purpose of the collaboration is to develop the *food* domain following our methodology, integrate it in Entitypedia and experiment it in some specific Web applications. As preliminary result we delivered a technical report in which we describe the outcomes of an analysis conducted on a small portion of the ontology describing wines. With the analysis, that follows the principles at the basis of the methodology, we identified typical pitfalls and mistakes and gave some concrete suggestions about how to improve the ontology. A collaboration with the Province of Trento, described in Chapter 8, aims at customizing the *Space* domain for local needs instead.

[43] http://www.telecomitalia.it/

Chapter 7

7. The Space domain

As our first step towards the population of the diversity-aware knowledge base with domain specific knowledge and by applying the methodology and principles described in the previous chapter, we developed the *Space* domain. Given its pervasiveness, *Space* is a rather important domain in a large spectrum of applications. One of them is described in the next chapter.

Taking into account the different aspects of *Space*, we built the domain as a descriptive ontology that was fully integrated in Entitypedia. Obtained from the refinement and extension of some existing geographical resources, mainly TGN and GeoNames, it provides knowledge about places of the world, their classes, their attributes and the spatial relations between them. The construction procedure was largely automatic, with manual intervention for the critical parts. This allowed us to obtain a very satisfactory quantitative and qualitative result.

The rest of the chapter is organized as follows. Section 7.1 introduces the problem and provides relevant state of the art. Sections 7.2 to 7.5 illustrate and provide examples concerning the application of the single steps of the methodology. Section 7.6 summarizes some of the difficulties that we had to deal with during these phases. Finally, Section 7.8 provides some details about the *Space* ontology we created.

7.1. Building the Space ontology

As an essential support to geo-spatial applications, there is a pressing need and growing interest in geo-spatial ontologies [Egenhofer, 2002][Kolas et al., 2005]. We consider *Space* in accordance with what people commonly understand by this term, which includes the surface of the earth, the space inside it and the space outside it. It comprises the usual geographical classes, often known as features, like land formations (continents, islands, countries), water formations (oceans, seas, streams) and physiographical classes (desert, prairie, mountain). It also comprises the areas occupied by a population cluster (city, town, village) and buildings or other man-made structures (school, bank, mine). Thus, for geo-spatial ontology we mean an ontology including geo-spatial entities, their classes, their attributes and relations (such as *part-of*, *overlaps*, *near-to*) between them. For instance, a geo-spatial ontology can provide the information that *Florence* (the entity) is a *city* (its class) in *Italy* (its ancestor in the *part-of* hierarchy) and, among its attributes, the corresponding latitude and longitude coordinates. In some contexts, tools which maintain this kind of information are also called semantic gazetteers [Keßler et al., 2009] or semantic geo-catalogues [Shvaiko et al., 2010a].

Applications requiring the use of geo-spatial ontologies include semantic Geographic Information Systems [Shvaiko et al., 2010a][Abdelmoty et al., 2007], semantic annotation (but also matching and discovery) of geo-spatial Web services [Roman et al., 2006][Janowicz et al., 2009], geographic semantics-aware web mining [Chaves et al., 2005] and Geographical Information Retrieval (GIR) [Jones et al., 2003][Buscardi and Rosso, 2008]. In particular, restricted to GIR, there are various competitions, for instance GeoCLEF[44], specifically for the evaluation of geographic search engines. In all such applications, ontologies are mainly used for word sense disambiguation [Vorz et al., 2007], semantic (faceted) navigation [Auer et al., 2009], document indexing and query expansion [Jones et al., 2003][Buscardi and Rosso, 2008], but in general they can be used in all the contexts where ontologies are needed to foster interoperability.

Unfortunately, the current geographical standards, for instance the specifications provided by the Open Geospatial Consortium (OGC)[45], do not represent an effective solution to the interoperability problem. In fact, they specifically aim at syntactic agreement [Kuhn, 2005]. For example, if it

[44] http://ir.shef.ac.uk/geoclef/
[45] http://www.opengeospatial.org/

is decided that the standard term to denote a harbour (defined as "*a sheltered port where ships can take on or discharge cargo*") is *harbour*, they will fail in applications where the same concept is denoted with a different term, e.g. with *seaport.* Similarly, gazetteers do not represent a satisfactory solution. In fact, they are no more than yellow pages for place names and, consisting of ambiguous plain descriptions, they do not support logical inference [Keßler et al., 2009]. As a response to this problem, some frameworks have been recently proposed to build and maintain geo-spatial ontologies (see for instance [Auer et al., 2009][Chaves et al., 2005][Abdelmoty et al., 2007]), but to the best of our knowledge no comprehensive, sufficiently accurate and large enough ontologies are currently available.

WordNet, even if not specifically designed for this, is *de facto* used as knowledge base in many semantic applications. Unfortunately, its coverage in terms of geographic information is very limited [Buscardi and Rosso, 2008], especially if compared to geographic gazetteers that usually contain millions of place names as well as fine-grained distinctions between classes, such as GeoNames. In addition, WordNet does not provide latitude and longitude coordinates as well as other relevant information which is of fundamental importance in geo-spatial applications. To overcome these limitations, some recent attempts have been developed with the goal to integrate WordNet with geographical resources. [Angioni et al., 2007] proposed a semi-automatic technique to integrate terms (classes and instances) from GEMET. [Vorz et al., 2007] created a new ontology from the integration of WordNet with a limited set of classes and corresponding instances from GNS[46] and GNIS[47]. [Buscardi and Rosso, 2008] used the same resources to enrich 2,012 WordNet synsets with latitude and longitude coordinates. Unfortunately, all the above mentioned approaches are very limited in the number of terms covered and accuracy.

Our main contribution to this problem is a very large and accurate geo-spatial faceted ontology that we call *Space* obtained from the refinement and extension of GeoNames, WordNet and the Italian part of MultiWordNet. Following the data model and the methodology presented in Chapter 6, *Space* accounts for the relevant classes, entities, their relations and attributes and, because constructed following the principles at the basis of the faceted approach, it is of very high quality in terms of robustness, extensibility, reusability, compactness and flexibility [Spiteri, 1998] [Broughton, 2006]. *Space* is the first domain that we created in Entitypedia.

[46] http://earth-info.nga.mil/gns/html/index.html
[47] http://geonames.usgs.gov

7.2. Identification of the terminology

The first step in the methodology to domain construction consists in the selection of the resources that allow identifying the natural language terms representing the geo-spatial classes, the entities, the relations, the attributes and their disambiguation into formal terms. In the construction of *Space* this was done in four steps as follows:

- ***Step 2.1: Selection of the information sources.*** Possible sources of terminology were collected, evaluated in terms of quality and quantity of the information provided and the best candidates were selected. This step was manual.

- ***Step 2.2: Resource pre-processing.*** It consisted in (a) the extraction of the relevant natural language terms from each selected source, (b) the analysis and categorization of the terms into classes, entities, relations and attributes, (c) the disambiguation of the terms into senses, thus making explicit the meaning of each term and, in case of multiple terms with same meaning, grouping them into synsets. This step was manual, but in general it can be partially automated if the sources are sufficiently structured.

- ***Step 2.3: Mapping the resources.*** As preliminary step towards the integration, synsets identified with the previous step were mapped across sources. Among other things, this allowed duplicates to be identified. The mapping was manually produced and validated.

- ***Step 2.4: Integration of the resources.*** It consisted in using the mapping produced with the previous step to integrate the synsets extracted from the different sources. This step was fully automatic.

These steps are extensively described in the next four sections.

7.2.1 Selection of the information sources

Among the various sources of *Space* specific terminology, we particularly concentrated on geo-spatial gazetteers. In fact, these gazetteers contain huge quantities of locations and corresponding classes. They are sometimes organized in hierarchies, thus providing also relations between them, and offer attributes such as latitude and longitude coordinates. On the basis of quantity and

quality criteria, we evaluated several candidates including Wikipedia[48], DBPedia[49], YAGO, GEMET[50] and the ADL gazetteer[51], but they are all limited in classes, entities, relations or attributes. GeoNames and TGN, instead, both met our requirements:

- ***Thesaurus of Geographical Names (TGN)***[52]. TGN is a poly-hierarchical (i.e. multiple parents are allowed) structured vocabulary containing 688 classes and around 1.1 million place names.

- ***GeoNames***. GeoNames provides 8 million place names in various languages amounting to 7 million unique places and corresponding attributes such as latitude, longitude, altitude and population. At the top level, the places are categorized into 9 broader categories, called feature classes, further divided into 663 classes, most of them associated with a natural language description. A special *null* class contains unclassified entities. In Table 13 they are given in detail.

We used GeoNames as the main source. Being a thesaurus, TGN was instead used for consultation in order to better disambiguate GeoNames classes and relations. Nevertheless, both TGN and GeoNames are pretty poor in relations. Since, understanding spatial relations is one of the fundamental features of Geographic Information Systems (GIS), we looked elsewhere for their identification. In particular, in producing our set of relations, we mainly followed the work by Arpinar et al. [2004], Egenhofer and Herring [1991], Egenhofer and Dupe [2009] and Pullar and Egenhofer [1988]. According to Egenhofer and Herring, spatial regions form a relational system comprising the relations between interiors, exteriors, and boundaries of two objects. Arpinar et al. suggest three major types of spatial relations: topological relations, cardinal direction and proximity relations. Egenhofer and Dupe propose topological and directional relations. According to them, topological relations have a leading role in qualitative spatial reasoning. Pullar and Egenhofer group spatial relations into direction relations (e.g. *north*, *northeast*), topological relations (e.g. *disjoint*), comparative or ordinal relations (e.g. *in*, *at*), distance relations (e.g. *far from*, *near to*) and fuzzy relations (e.g. *next to*, *close*). The spatial relations we propose include all these relations and some additional relations such as relative level (e.g. *above*, *below*), longitudinal

[48] http://www.wikipedia.org/
[49] http://dbpedia.org/About
[50] http://www.eionet.europa.eu/gemet/about
[51] http://www.alexandria.ucsb.edu/gazetteer/
[52] http://www.getty.edu/research/conducting_research/vocabularies/tgn

(e.g. *in front*, *behind*), side-wise (e.g. *right*, *left*) and position in relation to border or frontier (e.g. *adjacent*, *overlap*). In addition to spatial relations, we also consider some other kinds of relations, which can be treated as functional. For example, in the context of lakes, *primary inflow* and *primary outflow* are two important functional relations.

Feature Class	**Description**	**Number of classes**
A	Administrative divisions of a country. It also represents states, regions, political entities and zones	16
H	Water bodies, e.g., ocean, sea, river, lake, stream, etc.	137
L	Parks, areas, etc.	49
P	Populated places, e.g., capitals, cities, towns, small towns, villages, etc.	11
R	Roads and railroads	23
S	Spots, buildings and farms	242
T	Mountains, hills, rocks, valleys, deserts, etc.	97
U	Undersea areas	71
V	Forests, heaths, vineyards, groves, etc.	17

Table 13. Classes in GeoNames (version downloaded on March 2009)

7.2.2 Resource pre-processing

With this step we extracted from GeoNames the natural language terms denoting the names of the classes, the names of the entities and the names of the attributes. Attribute values, being mostly quantitative, do not provide additional terminology. A part from the basic ones, the only relation explicitly provided in GeoNames is *neighbour* connecting each country with those of neighboring ones.

With the analysis, we mainly focused on the relations. In fact, since in GeoNames entities are neatly separated from classes with attributes directly associated to each entity, they could be easily identified. Conversely, with the only exception of *neighbour*, the kind of the relations is in general not explicitly provided. Relations between instances can be mapped to a generic *part-of* relation, including administrative and physical containment. The former connects administrative divisions, i.e. entities of classes such as country, province and district. The latter connects enti-

ties of classes such as lake, river and mountain to the corresponding administrative division. Relations between entities and classes correspond to *instance-of*. Since in GeoNames classes are provided in a flat list, no relations between classes are available.

With the disambiguation we created the senses by associating a natural language description (in English and Italian) to each natural language term found. Since we did not find cases of synonymy, each sense coincided with the synset.

Concerning the disambiguation of the classes, we found that out of the 663 classes in GeoNames, in 57 cases no definition is provided at all. For these names we tried to understand the exact intended meaning, most of the time by considering the context of the term used, i.e. the corresponding feature class, and the instances associated to it. It was also observed that, even though definitions are provided for the remaining terms, in some cases they are either ambiguous or not clear enough. Consider for instance the class *astronomical station*. GeoNames defines it as "*a point on the earth whose position has been determined by observations of celestial bodies*". Conversely, we decided that a more appropriate definition is "*a station from which celestial bodies and events can be observed*" and therefore we substituted it.

Concerning the disambiguation of the entities, the names were directly extracted from the *name* and *alternative name* attributes in GeoNames, while the descriptions, in English and Italian, were automatically generated starting from the information provided by the *is-a* and *instance-of* relations. Several rules were used. For instance, one that we used for English is:

entity_name + " is " + *article* + " " + *class_name* + " in " + *parent_name* + "(" + *parent_class* + " in " + *country_name* + ")";

This allows for instance describing the *Garda Lake* as *"Garda Lake is a lake in Trento (Administrative division in Trentino Alto-Adige)"*.

The only relation found, *neighbour* was disambiguated as *"a nearby object of the same kind"*.

The disambiguation of the attributes led to the identification of 13 distinct attributes including *name, latitude, longitude* and *altitude*. Notice that we defined one single attribute representing

name and *alternative name*, the latter codifying secondary names for the locations. In fact, we considered the value of the *name* attribute as standard term. These attributes are those provided for all the entities, while the other attributes are mainly provided for populated places and administrative divisions. The attributes extracted from GeoNames, with corresponding natural language description, are provided in Section 7.7.

7.2.3 Mapping the resources

As preliminary step towards the integration with Entitypedia, synsets created from GeoNames were mapped with WordNet. However, this was only done for the synsets of the classes, the attributes and the *neighbour* relation. In fact, the other relations in GeoNames correspond to the basic ones, while for the reasons already stressed in the previous chapter we ignored the synsets representing entities in WordNet. We distinguished the following cases:

- ***Case 1: there is an equivalent synset in WordNet.*** Two synsets were marked as equivalent if they denote the same meaning. We say that we have an *exact match* if the word in the GeoNames synset is also present in the WordNet synset. We say that there is a *partial match* if there is a corresponding synset in WordNet but the word in the GeoNames synset is not present in the WordNet synset. It is clear that the latter case is very difficult to detect with automatic tools. An example of the first case is *river*. An example of the second case is *leprosarium.* This term is not available in WordNet, but there is a synset for the equivalent term *lazaret*.

- ***Case 2: there is a more general synset in WordNet***. In case of mismatch, we looked for a more general synset according to the *is-a* relation. In this case the GeoNames synset was marked as more specific than the WordNet synset. Consider for instance the class *palm grove*, defined in GeoNames as "*a planting of palm trees*". There is no equivalent synset for it in WordNet, but the more general synset for *grove*, defined as "*garden consisting of a small cultivated wood without undergrowth*", is available in WordNet. In this case *palm grove* in GeoNames is marked as more specific than *grove* in WordNet.

- ***Case 3: there is a synset in WordNet that can be linked using part-of.*** We occasionally considered appropriate to associate synsets using the *part-of* relation instead of the *is-a* relation. In these cases, we explicitly marked the GeoNames synset as *part-of* the Word-

Net synset. For instance, an *icecap depression*, defined in GeoNames as "*a comparatively depressed area on an icecap*", is a part of an *icecap*, defined in GeoNames as "*a dome-shaped mass of glacial ice covering an area of mountain summits or other high lands; smaller than an ice street*", and not something more specific. A similar discourse can be done for *canal bend* and *section of canal* which are both parts of *canal*.

To assess the quality of the mapping produced, a validation work was carried out by some experts in library science, particularly skilled in knowledge organization. The experts were different from those who were involved in the first phase of our work. This was done in order to assure that the validation work was not influenced by any unexpected external factor or bias. In order to carry out the validation work, the validators had to look at factors like the soundness of the natural language description for the senses determined during the first phase, suitability of the selected synsets in WordNet and suitability of assigned names for the plural forms. Section 7.6 provides a list and corresponding description of the most interesting issues. In case of disagreement we iterated on the previous steps till all the conflicting cases were solved. The result of our analysis is summarized in Table 14.

GeoNames Classes	**Instances**	**%**
Which have a description in GeoNames	606	91.40
Which have no description in GeoNames	57	8.60
For which we provided or changed the description	92	13.88
For which we found a corresponding synset in WordNet	306	46.15
For which only one noun synset is available in WordNet	160	24.13
For which multiple noun synsets are available in WordNet	242	36.50
For which one part of the description matches with one synset and another part of the description matches with another synset	15	2.26
For which there is no equivalent synset in WordNet	357	53.84

Table 14. Outcomes of the GeoNames class analysis and their mapping to WordNet

7.2.4 Integration of the resources

Once the mapping was produced and validated, the next phase consisted in the integration of the resources. This phase was fully automatic and consisted of the following steps[53]:

- ***Concept Integration***. By using the mapping between GeoNames and WordNet, we integrated GeoNames synsets with those in Entitypedia. Here, by integration we mean the importing in Entitypedia of the GeoNames synsets which do not have an exact or partial match with WordNet and are therefore not already present in Entitypedia. For each missing synset, this was done by creating a corresponding English and Italian synset in the natural language level of Entitypedia by specifying the word, the natural language description and the part of speech. We also created a corresponding formal term and the *is-a* or *part-of* relation necessary to connect it to the parent term. For the cases of partial match, we just added the missing word to the corresponding synset in Entitypedia. For the cases of exact match, we just saved a reference to the synset in Entitypedia for future use (see next step).

- ***Instance importing***. This step consisted in importing the entities contained in GeoNames into Entitypedia. For each of the entities in GeoNames we created a new formal term denoting an entity in the knowledge part of Entitypedia and, by means of instance-of relations, we related each of them to the formal term of the corresponding class previously created or identified as equivalent to an existing one. We also created *part-of* relations between such entities, according to the information provided in GeoNames. For instance, we codify the information that *Florence* is an instance of *city* and is part of the *Tuscany* region in *Italy*. Note that, the entities of the special *null* class were treated as instances of the generic class *location*.

- ***Attribute importing***. The attributes associated to each entity in GeoNames were imported as attributes and corresponding values (focusing on English and Italian names for the moment) in the knowledge part of Entitypedia. This generated around 70 million attributes and corresponding values.

[53] GeoWordNet corresponds to the knowledge base obtained after these phases.

Table 15 shows the amount and kind of new relations that we created with the integration of GeoNames. Notice that for each relation we also created the corresponding inverse relations. Therefore, the actual number of relations is double the number shown in the table.

Objects involved	**Kind of relation**	**Quantity**
Relations between classes	is-a	327
	part-of	36
Relations between entities and classes	instance-of	6,907,417
Relations between entities	part-of	2,265,283

Table 15. Statistics about the number of relations created

7.3. Analysis

With the analysis, the terms collected and disambiguated during the previous phase were used as building blocks for the construction of the facets that constitute the *Space* ontology. For sake of simplicity, for the rest of the steps we focus only on the terms denoting classes.

The integration of the resources also helped us identifying the main sub-trees in Entitypedia containing the necessary synsets representing geographical classes. In fact, with the integration, each of the synsets coming from GeoNames was hooked to one of the sub-trees rooted in:

- **location** - a point or extent in space
- **artifact, artefact** - a man-made object taken as a whole
- **body of water, water** - the part of the earth's surface covered with water (such as a river or lake or ocean); "they invaded our territorial waters"; "they were sitting by the water's edge"
- **geological formation, formation** - the geological features of the earth
- **land, ground, soil** - material in the top layer of the surface of the earth in which plants can grow (especially with reference to its quality or use); "the land had never been plowed"; "good agricultural soil"
- **land, dry land, earth, ground, solid ground, terra firma** - the solid part of the earth's surface; "the plane turned away from the sea and moved back over land"; "the earth shook for several minutes"; "he dropped the logs on the ground"

It is worthwhile to underline that not all the nodes in these sub-trees necessarily need to be part of *Space*. As a matter of fact, many of the descendants of *location* and *artifact* cannot be classified in our fundamental categories and therefore they were not included in *Space*. For instance, the following terms were discarded:

(Descendants of location)

- **there** - a location other than here; that place; "you can take it from there"
- **somewhere** - an indefinite or unknown location; "they moved to somewhere in Spain"
- **seat** - the location (metaphorically speaking) where something is based; "the brain is said to be the seat of reason"

(Descendants of artifact)

- **article** - one of a class of artifacts; "an article of clothing"
- **anachronism** - an artifact that belongs to another time
- **block** - a solid piece of something (usually having flat rectangular sides); "the pyramids were built with large stone blocks"

Terms denoting classes of real world entities were analyzed using their topological, geometric or geographical characteristics. We tried to be exhaustive in their determination. This leaves open the possibility to form a huge number of very fine grained groups. In order to illustrate the analysis process, consider the following list:

- **Mountain** - a land mass that projects well above its surroundings; higher than a hill
- **Hill** - a local and well-defined elevation of the land; "they loved to roam the hills of West Virginia"
- **Stream** - a natural body of running water flowing on or under the earth
- **River** - a large natural stream of water (larger than a brook); "the river was navigable for 50 miles"

Following the principles provided in Section 6.5.2, and in particular the *principle of relevance* and the *principle of ascertainability*, we can derive the following characteristics:

Mountain characteristics:

- the well-defined elevated land
- formed by the geological formation (where geological formation is a natural phenomenon)
- altitude in general >500m

Hill characteristics:

- the well-defined elevated land
- formed by the geological formation, where geological formation is a natural phenomenon
- altitude in general <500m

Stream characteristics:

- a body of water
- a flowing body of water
- no fixed boundary
- confined within a bed and stream banks

River characteristics:

- a body of water
- a flowing body of water
- no fixed boundary
- confined within a bed and stream banks
- larger than a brook

7.4. Synthesis

Consider the list of characteristics selected with the analysis. The first characteristic of each of the terms above clearly suggests the distinction between two basic categories, the first consisting of *mountain* and *hill* and the second consisting of *stream* and *river*. Based upon those characteristics, two facets can be formed. They can be named *natural elevation* and *flowing body of water*, respectively. A further analysis of the characteristics suggested the creation of the more general facets *landform* and *body of water*, respectively.

The terms *mountain* and *hill* can be further differentiated *by size*. Note that, according to the *principle of relevance* and the *principle of permanence*, in this case size is a good distinguishing characteristic. In fact, it can be considered (almost) permanent in nature. Note that this is not true in general. For instance, it is not appropriate to distinguish animals by size because in this respect size is transitional in nature, i.e. their size rapidly changes over time. This is an example of what Aristotle called *accidental predicates* [Smith and Mark, 1998].

Note that *river* is a natural stream, and therefore a special kind of *stream*. In particular, this means that all the properties of stream are inherited by river (but not the vice versa). This is reflected in the facet hierarchy by putting *river* under *stream*. Based upon the observations above we can build the following two facets, *body of water* and *landform*:

Body of water	**Landform**
Flowing body of water	Natural elevation
Stream	Mountain
River	Hill

An important property of facets is that they are *hospitable* (see Section 6.2), i.e. they can be easily extended to accommodate additional terms as needed. Assume for instance that the new term *lake*, defined as *"a body of (usually fresh) water surrounded by land"*, is identified. By analyzing it, we can derive the following characteristics:

Lake characteristics:

- a body of fresh water
- fixed geographical boundary
- a stagnant body of water

Going through the characteristics above, it should be quite easy to understand that *lake* cannot be put under the *flowing body of water*, even though it is a *body of water*. This implies that our classification is not good enough to classify all sorts of body of water, i.e. it is not exhaustive (*principle of exhaustiveness*). In order to include lakes, we need to extend the body of water facet with *stagnant body of water* in the same array of *flowing body of water*. This solves our problem.

In order to understand the importance of the *principle of exclusiveness*, assume to create in our classification the sub-classes *inland body of water*, *marine body of water*, *flowing body of water* and *stagnant body of water* and to put them in the same array under the main class *body of water*. Such categorization brings to confusion. In fact, lake can be now classified as both *inland body of water* and *stagnant body of water*. To avoid this confusion, the *principle of exclusiveness* plays an important role. According to this principle, all the characteristics used to classify a term must be mutually exclusive. So, we should not include all those four classes in the same array.

Similarly to lakes, we can extend the *natural elevation* facet in order to accommodate the term *valley* (defined as "*a long depression in the surface of the land that usually contains a river*"). Valley is a natural depression. So, in order to assign a place for *valley* inside this scheme, we have to create another sub-facet, namely, *natural depression*. Consider also that valleys are seen in both the oceanic areas (called *oceanic valleys*) and continental areas (called *valleys*). There is in general symmetry of real world entities in the continental and oceanic areas. For most of the continental entity classes there is a corresponding oceanic entity class with similar features but different name. So, in order to correctly classify the entities based upon the characteristic of their location, i.e. oceanic or continental, we should create the sub-facets *oceanic* and *continental* under the *natural elevation* and *natural depression* respectively as shown below. These additional facets make the classification of *landforms* exhaustive. See the appendix for an extended version of the *body of water* facet.

Body of water

- Flowing body of water
  - Stream
    - Brook
    - River
- Stagnant body of water
  - Pond
  - Lake

Landform

- Natural depression
  - Oceanic depression
    - Oceanic valley
    - Oceanic trough
  - Continental depression
    - Trough
    - Valley
- Natural elevation
  - Oceanic elevation
    - Seamount
    - Submarine hill
  - Continental elevation
    - Hill
    - Mountain

7.5. Standardization and ordering

Specifying different words for the same notion allows supporting semantic interoperability between systems using different terminology. Nevertheless, within each synset we selected a standard term among the synonyms. Following the *principle of currency*, for the synsets extracted from WordNet, we followed the order of the words in the corresponding synsets. Analogously, for the synsets created or enriched with the words from GeoNames we either kept the original terms - if found appropriate - or we changed them based on the study of some relevant scientific publications or standard vocabularies. For instance, we substituted *mountains* (from the feature class T, including land formations) with *mountain range* (as from Geology terminology), and *hill* (from the feature class U, including undersea entities) with *submarine hill* (as from Oceanography terminology).

In general it is good practice to avoid choosing the same standard term to denote two totally different concepts. However, in one case - for the word *bank* - we had to allow an exception:

- **bank** - sloping land (especially the slope beside a body of water)) *"they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"*
- **bank** - a building in which the business of banking transacted; *"the bank is on the corner of Nassau and Witherspoon"*

In these extreme cases, it is the context that disambiguates their meaning (*principle of context*). The two meanings of bank were disambiguated as follows:

- **Landform >** Natural elevation > Continental elevation > Slope > Bank
- **Facility >** Business establishment > Bank

Given our purpose and scope, following the *principle of ordering* we ordered the classes based upon the *decreasing quantity* of the entities instantiating the class. Within each chain of terms, from the root to the leaves, we followed the same ordering preference. However, it is not always possible or appropriate to establish this order, especially when the classes do not share any characteristic. For example, we could not establish any order between *body of water* and *landform*. In

such cases we preferred the *canonical order*, i.e. the order traditionally followed in library science. The final result, after ordering, was as follows:

Landform

- Natural elevation
  - Continental elevation
    - Mountain
    - Hill
  - Oceanic elevation
    - Seamount
    - Submarine hill
- Natural depression
  - Continental depression
    - Valley
    - Trough
  - Oceanic depression
    - Oceanic valley
    - Oceanic trough

Body of water

- Flowing body of water
  - Stream
    - River
    - Brook
- Stagnant body of water
  - Lake
  - Pond

7.6. Critical issues faced

The main difficulties we faced in the process described in the previous sections were mainly due to the different conceptualization in GeoNames and WordNet. Here we briefly describe them.

Facility: the service vs. function approach. The term *facility* is a key term in GeoNames. Being generic, a quite considerable amount of more specific classes are present in GeoNames. A mistake in the analysis of this term would have major consequences. In WordNet there are 5 different noun senses for the term, most of them focusing more on the notion of "service", rather than on the notion of "function":

- **facility**, installation (a building or place that provides a particular service or is used for a particular industry) *"the assembly plant is an enormous facility"*
- adeptness, adroitness, deftness, **facility**, quickness (skillful performance or ability without difficulty) *"his quick adeptness was a product of good design"; "he was famous for his facility as an archer"*
- **facility**, readiness (a natural effortlessness) *"they conversed with great facility"; "a happy readiness of conversation"--Jane Austen*
- **facility** (something designed and created to serve a particular function and to afford a particular convenience or service) *"catering facilities"; "toilet facilities"; "educational facilities"*
- **facility** (a service that an organization or a piece of equipment offers you) *"a cell phone with internet facility"*

On the other hand, the description of the term provided in GeoNames ("*a building or buildings housing a center, institute, foundation, hospital, prison, mission, courthouse, etc.*") is rather generic and incomplete as includes only a building or a group of buildings. There are classes which are not buildings but that can be still treated as facilities, e.g., farms and parks. This is in line with the first sense in WordNet, where a facility can be a building or a place. On the one hand many buildings provide services. Buildings housing banks usually provide transaction services; buildings housing hospitals usually provide health care services; buildings housing libraries usually provide access to the catalogue and book consultation. On the other hand, there are also buildings (or generic constructions) that do not provide any service, but are rather intended to

have a function. For instance, houses are used for living purposes, while roads, streets and bridges have a transportation function (but no specific service is provided).

We decided to adhere to the WordNet vision and clearly distinguish between buildings and places providing a service (placed under the first sense) and those having just a (specific or generic) function (placed under the forth sense).

Plurals and Parenthesis. 92 class names in GeoNames are given in singular form, e.g., *populated place* and *vineyard*, as well as in plural form, e.g., *populated places* and *vineyards*. In addition, 99 class names are given as a mixed singular-plural form, e.g., *arbour(s), marsh(es)* and *distributary(-ies),* sometimes in conjunction with the singular or plural form also. From our analysis, singular forms are used to denote single entities; plural forms indicate groups of entities; mixed forms are preferred when it is not easy to discriminate between the two previous cases.
The approach we followed was to avoid plurals, thus identifying for each plural or mixed form a more appropriate name. For instance, we substituted *lakes* with *lake chain* and *mountains* with *mountain range*.

Dealing with polysemy. 242 class names in GeoNames are polysemous, namely they have two or more similar or related meanings in WordNet. It is not always easy to understand the correct meaning meant, especially in the cases in which no description is provided. To find out the right concept, we compared the description of each class, if available, to each of the meanings of that class in WordNet. In 15 cases, we found out that a part of the description matches with one sense and another part of the description matches with another sense. Examples of such classes are *university*, *library* and *market*. During disambiguation such situations were overcome by comparing related terms in WordNet, for instance the ancestors, with the GeoNames feature class. To be more concrete consider the following example for the term *university*, defined in GeoNames as: "*an institution for higher learning with teaching and research facilities constituting a graduate school and professional schools that award master's degrees and doctorates and an undergraduate division that awards bachelor's degrees*". It can be then summarized to be an institution for higher learning including teaching and research facilities that award degrees. The term university has three meanings in WordNet:

- **university** (the body of faculty and students at a university)
- **university** (establishment where a seat of higher learning is housed, including administrative and living quarters as well as facilities for research and teaching)
- **university** (a large and diverse institution of higher learning created to educate for life and for a profession and to grant degrees)

The first meaning has little connection with the description given in GeoNames and is therefore excluded. The second meaning is relevant as it describes a university as an establishment for higher learning which also facilitates research and teaching. The third meaning is also relevant as it describes that it is a large institution of higher learning to educate for life and to grant degrees. To better disambiguate between the two remaining candidate meanings we then compared the hypernym hierarchy of the two synsets with the feature class provided for the term in GeoNames. The third meaning is a descendant of *social group*. The second meaning is a descendant of *construction*, which is closer to the feature class S (spots, building and farms). As a consequence, we finally selected the second meaning.

When such kind of analysis was not enough to disambiguate, we analyzed the instances from all close matched senses of WordNet and looked for their co-occurrence with the instances in GeoNames. In case of a match at instance level, we chose the corresponding sense. For example, consider the candidate term *palace*. GeoNames defines it as "*a large stately house, often a royal or presidential residence*". The first ("*a large and stately mansion*") and forth ("*official residence of an exalted person (as a sovereign) correspond to it*") senses for the term in WordNet look like possible candidates. Following the proposed approach, we found that *Buckingham Palace* is the only instance in common with the first sense whereas there are no instances in common with the fourth sense. Therefore, we chose the first sense.

Unique name provision. In GeoNames, the same name is occasionally used to denote different concepts in different feature classes. This is particularly frequent for the classes under the feature class T, which denotes mountains, hills, rocks, and U, which denotes undersea entities. Some examples are *hill, mountain, levee* and *bench*. Conversely, we provided distinct names for them. For the above examples, we distinguished between *hill* and *submarine hill*, between *mountain* and *seamount*, between *levee* and *submarine levee*, and between *bench* and *oceanic bench*.

Clearly, these terms were not just arbitrarily assigned. They were in fact collected from authentic literature on Geography, Oceanography and Geology (e.g., Encyclopaedia Britannica[54]).

Physical vs Abstract entities. It is important to note that, since GeoNames always provides latitude and longitude coordinates for the entities, all of them must be seen as physical entities, i.e. having physical existence. However, when mapping the classes from GeoNames to WordNet, we observed that for 27 of them, WordNet only provides abstract senses, namely they are categorized as descendant of *abstract entity*. For example, for the concept *political entity* ("*a unit with political responsibilities*") WordNet provides a single synset at distance 6 from *abstract entity*. It is clear that, it would be incorrect to associate a geo-political entity, say *India*, under the abstract concept provided by WordNet. In these cases we rather preferred to create a new synset in WordNet somewhere under *physical entity*. In the specific case, we created a new synset with the term *geo-political entity* defined as "*the geographical area controlled or managed by a political entity*" as more specific than *physical object*.

[54] http://www.britannica.com/

7.7. Objects in the Space ontology

Table 16 provides the total number of objects we identified for each C/E/R/A in the *Space* ontology. Note that for the relations we do not include the basic *is-a*, *part-of*, *instance-of* and *value-of* relations. Similarly, for the attributes we do not include the attribute values, but only the attribute names.

Objects	Quantity
Classes (C)	845
Entities (E)	6,907,417
Relations (R)	70
Attributes (A)	31

Table 16. Overall statistics of the *Space* ontology

The facets of entity classes we created are:

- **Region** – "a large indefinite location on the surface of the Earth"
- **Administrative division** – "a district defined for administrative purposes"
- **Populated place** – "a city, town, village, or other agglomeration of buildings where people live and work"
- **Facility** – "a building or any other man-made permanent structure that provides a particular service or is used for a particular industry"
- **Abandoned facility** – "abandoned or ruined building and other permanent man made structure which are no more functional"
- **Land** – "the solid part of the earth's surface"
- **Landform** – "the geological features of the earth"
- **Body of water** – "the part of the earth's surface covered with water (such as a river or lake or ocean)"
- **Agricultural land** – "a land relating to or used in or promoting agriculture or farming"
- **Wetland** – "a low area where the land is saturated with water"

Each of these top-level facets is further sub-divided into several sub-facets. For example, *facility* is sub-divided into *living accommodation*, *religious facility*, *education facility*, *research facility*, *education research facility*, *medical facility*, *transportation facility*, and so on. Similarly, *body of water* is further sub-divided primarily into the two sub-facets *flowing body of water* and *stagnant body of water*. In a similar way, *landform* is further subdivided into the two sub-facets *natural elevation* and *natural depression*. At lower levels all of them are further sub-divided into sub-sub-facets and so on. For example, *natural elevation* consists of *continental elevation* and *oceanic elevation*, while *natural depression* consists of *continental depression* and *oceanic depression*. Some examples of facets of relations are reported in Table 17.

Direction	East South-east South South-west …
External spatial relation	Alongside Adjacent Near Neighbourhood ...
Sideways spatial relation	Right (right side) Centre-line Left Alongside …
Relative level	Above Below Up ...

Table 17. Examples of spatial relations

The attributes extracted from GeoNames are the following:

- **Name** - "a language unit by which a person or thing is known"
- **Latitude** - "the angular distance between an imaginary line around a heavenly body parallel to its equator and the equator itself"
- **Longitude** - "the angular distance between a point on any meridian and the prime meridian at Greenwich"
- **Altitude** - "elevation especially above sea level or above the earth's surface"
- **Total area** - "the sum of all land and water areas delimited by international boundaries and/or coastlines"
- **Population** - "the number of inhabitants (either the total number or the number of a particular race or class) in a given place (country or city etc.)"
- **Top level domain** - "one of the domains at the highest level in the hierarchical Domain Name System (DNS) of the Internet"
- **Domain name** - "strings of letters and numbers (separated by periods) that are used to name organizations and computers and addresses on the internet"
- **Natural language** - "a human written or spoken language used by a community"
- **Calling code** - "a number usually of 3 digits assigned to a telephone area as in the United States and Canada"
- **Country code** - "short alphabetic geographical codes developed to represent countries and dependent areas"
- **Code** - "a coding system used for transmitting messages requiring brevity or secrecy"
- **Time zone** - "any of the 24 regions of the globe (loosely divided by longitude) throughout which the same standard time is used"

We extended this set by defining some additional attributes, including for instance *depth* (e.g. of a lake), *climate* and *temperature*.

The ontology allows the 6,907,417 entities extracted from GeoNames to be indexed, browsed and exploited. Table 18 provides a fragment of the populated ontology.

Objects	**Quantity**
Mountain	279,573
Hill	158,072
Mountain range	19,578
Chain of hills	11,731
Submarine hills	78
Chain of submarine hills	12
Oceanic mountain	5
Oceanic mountain range	0

Table 18. A fragment of the populated scheme

In comparing it to the existing geo-spatial ontologies, our *Space* ontology turns out to be much richer in all its aspects. Just to provide a small glimpse, GeoNames and TGN count 663 and 688 classes respectively; while in our ontology we already have, at this stage, 845 classes. In fact, it is worthwhile to underline that, since hospitality is one of the significant features of facets, maintenance costs are kept low as it is always possible to extend it at the desired level of granularity. In this respect, we have been already working to further extend it. For instance, this is what has been done by importing classes and locations from the dataset of the Autonomous Province of Trento in Italy (see next chapter). This allows a more and more accurate annotation, disambiguation, indexing and search on geographical resources.

Chapter 8

8. The semantic geo-catalogue

To be effective, geo-spatial applications need to provide powerful and flexible search capabilities to support their users. This is specifically underlined by the INSPIRE[55] directive and regulations [European Commission, 2009] [European Parliament, 2009] that establish minimum criteria for the *discovery services* to support search within the INSPIRE metadata elements. However, discovery services are often limited by only syntactically matching user terminology to metadata describing geographical resources [Shvaiko et al., 2010a]. In fact, the way in which this is often achieved is by following current geographical standards that tend to fix the terminology to be used. Though, this introduces a high level of rigidity in the way users and applications interoperate. This weakness has been identified as one of the key issues for the future of the INSPIRE implementation [Crompvoets et al., 2004] [Smits and Friis-Christensen, 2007] [Lutz et al., 2009] [Vaccari et al., 2009]. As part of the solution, geo-spatial ontologies by providing domain specific terminology represent an essential support [Egenhofer, 2002] [Kolas et al., 2005].

A *Spatial Data Infrastructure* (SDI) is a framework and a set of tools that allow managing spatial data and metadata in an efficient and flexible way. With the Semantic Geo-Catalogue (SGC) project, promoted by the Autonomous Province of Trento (PAT) in Italy with the collaboration of Informatica Trentina, Trient Consulting Group and the University of Trento, the geo-portal within the SDI of the PAT was extended by providing *semantic query processing* support. The main requirement was to allow users to submit queries such as *bodies of water in Trento*, run them on top of the available metadata files and - by semantically expanding the terms in the query - get results also for more specific classes such as *rivers* and *lakes*. Technological requirements coming from the INSPIRE directive included (a) *performance*: send one metadata record within 3s (this includes, in our case, the time required for the semantic expansion of the query); (b) *availability*: service up by 99% of the time; (c) *capacity*: 30 simultaneous service requests within 1s.

[55] INSPIRE is the EU initiative aiming at establishing an infrastructure in Europe to make geo-spatial information more accessible and interoperable: http://inspire.jrc.ec.europa.eu/

Chapter 8. **The** semantic geo-catalogue

In this chapter we report our work on the implementation of the semantic geographical catalogue of the PAT by focusing in particular on the semantic extension of its discovery service which provides the necessary support for query expansion. The key points to meet the goal were:

- the adoption of the S-Match open source semantic matching tool;
- the development and use of a faceted geo-spatial ontology codifying the domain knowledge about the local geography of the PAT;

The faceted ontology was built, in English and Italian, following the methodology and the principles presented in Chapter 6. In our case, each node in the ontology represents either a geographical class or a location (our entities); we use *is-a* and *part-of* relations to connect classes, *part-of* relations to connect locations and *instance-of* relations to connect locations to corresponding geographical classes. The faceted ontology includes *inter-alia* the *administrative divisions* (e.g., municipalities, villages), the *bodies of water* (e.g., lakes, rivers) and the *land formations* (e.g., mountains, valleys) of the PAT. Therefore it can be seen as a sort of customized version of the *Space* domain. Before querying the metadata files, terms in user queries are expanded by S-Match with domain specific terms taken from the faceted ontology.

The rest of the chapter is organized as follows. Section 8.1 describes the overall system architecture with particular emphasis on the semantic extension. Section 8.2 describes the local dataset of the PAT and how we pre-processed it. This corresponds to the analysis phase of our methodology to domain construction. Section 8.3 provides details about the construction and population of the faceted ontology. This corresponds to the synthesis phase of our methodology to domain construction. Section 8.4 provides the evaluation of the semantic extension. Section 8.5 explains how we integrated the faceted ontology with Entitypedia. Following common practices, we also complied with the Open Government Data[56] (OGD) initiative. This was done by publishing in RDF (Resource Description Framework) [Brickley and Guha, 2004] useful data and metadata taken from the local repository of the PAT and by linking them to relevant vocabularies. Finally, Section 8.6 presents the OGD initiative, the released data and a mashup application we developed using them.

[56] http://opendefinition.org/government/

8.1. The architecture

The overall architecture of the SDI of the PAT, exemplified in Fig. 34, is constituted by the front-end, business logic and back-end layers as from the standard three-tier paradigm. The geo-catalogue is one of the services of the existing geo-cartographic portal[57] of the PAT. It has been implemented by adapting GeoNetwork[58], that is conform to the INSPIRE directive, and by taking into account the rules enforced at the national (Italian) level. Following the best practices for the integration of the third-party software into the BEA ALUI framework[59] (the current engine of the geo-portal), external services are brought together using a portlet[60]-based scheme, where GeoNetwork is used as a back-end.

At the front-end, the functionalities are realized as three portlets for:

1. ***Metadata management***, including harvesting, search and catalogue navigation functionalities;
2. ***User/group management***, to administer access control on the geo-portal;
3. ***System configuration***, which corresponds to the functionalities of the GeoNetwork's Administrator Survival Tool (GAST) tool of GeoNetwork.

These functionalities are mapped *1-to-1* to the back-end services of GeoNetwork. Notice that external applications, such as ESRI ArcCatalog, can also access the back-end services of GeoNetwork.

The discovery service of GeoNetwork was extended by providing *semantic query processing* support. This was achieved by using S-Match. Initially designed as a standalone application, S-Match was integrated with the SDI through a wrapper that provides web services to be invoked by GeoNetwork. This approach mitigates risks of failure in experimental code while still following strict uptime requirements of the production system. Another advantage of this approach is the possibility to reuse this service in other applications with similar needs.

[57] http://www.territorio.provincia.tn.it/
[58] http://geonetwork-opensource.org
[59] http://download.oracle.com/docs/cd/E13174_01/alui/
[60] http://jcp.org/en/jsr/detail?id=168

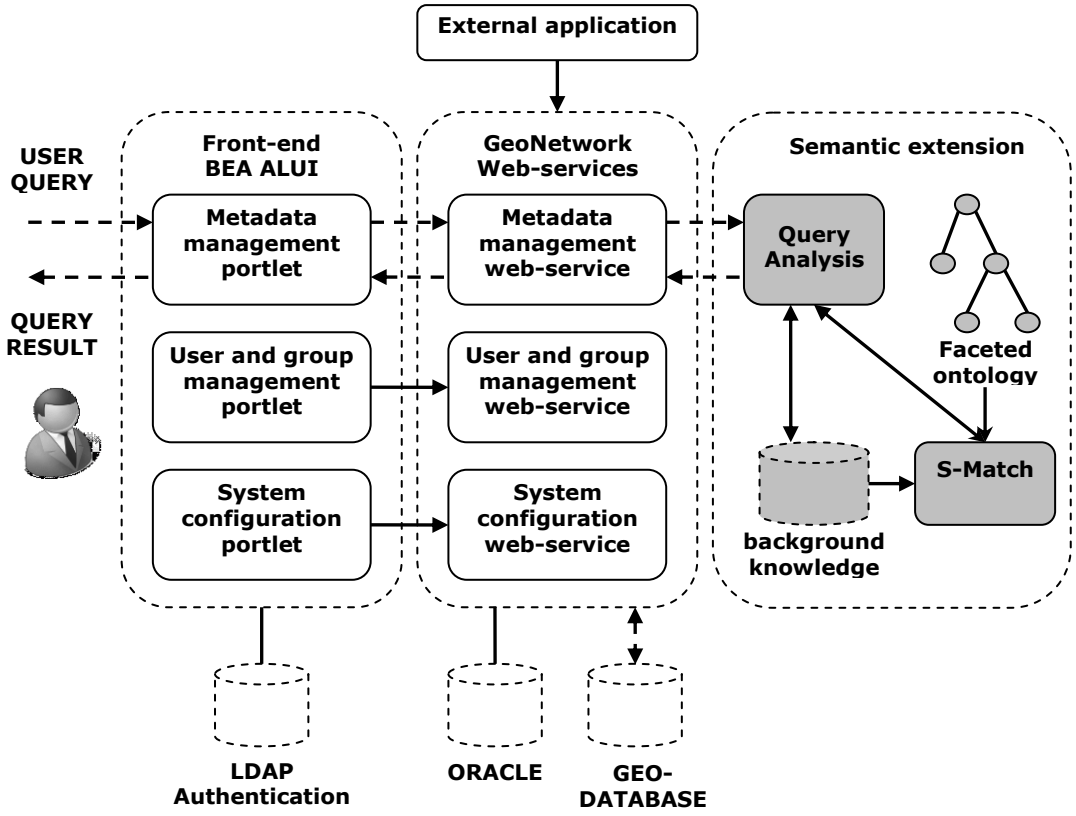


Fig. 34. The overall system architecture

The flow of information, starting from the user query to the query result, is represented with arrows in Fig. 34. Once the user enters a natural language query (which can be seen as a classification composed by a single node), the system translates it into a formal language according to the knowledge codified in the background knowledge. The formal representation of the query is then given as input to S-Match that matches it against the faceted ontology. All the labels in the subtrees of the matched nodes are returned, thus expanding the query with domain specific terms. The expanded query is then used by the metadata management component of the SDI to search into metadata files and finally access the corresponding maps in the geo-database. For instance, the query *watercourse* is expanded as *watercourse stream river rivulet*. In fact, *watercourse* and *stream* are synonyms in WordNet while *river* and *rivulet* are more specific terms in the *body of water* facet of the faceted ontology.

8.2. Dataset pre-processing

The first step towards the construction and population of the faceted ontology was to analyze the data provided by the PAT, extract the classes, the locations, their attributes and relations between them and filter out noisy data. The data were put in a temporary database. Fig. 35 summarizes the main phases of the pre-processing. They are described in detail in the rest of the section.

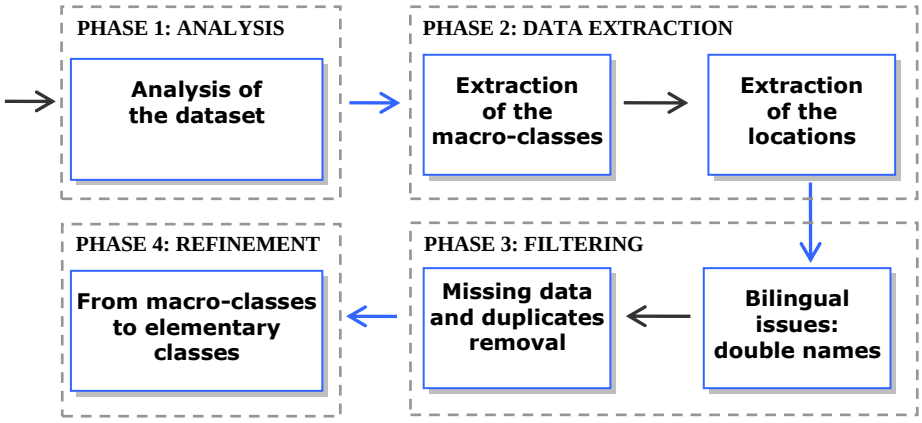


Fig. 35. A global view of the phases for the pre-processing

Analysis of the dataset. The data were directly gathered from the PAT administration in form of MS Excel files, described in Table 19. With the analysis of these files we discovered that the *features* file contains information about 45 classes; the *ammcom* file contains 256 municipalities; the *localita* file contains 1,507 wards and ward parts, that we generically call populated places; the *toponimi* file contains 18,480 generic locations (including *inter-alia* villages, mountains, lakes and rivers).

File name	Description
features.xls	It provides the name and the identifier of the classes.
ammcom.xls	It provides the name, the identifier, latitude and longitude of the municipalities.
localita.xls	It provides the name, the identifier, latitude and longitude of the wards and ward parts (that we map to populated places). It also provides the identifier of the municipality a given ward or ward part belongs to.
toponimi.xls	It provides the name, the identifier, class, latitude and longitude of the locations. It also provides the identifiers of the ward or ward part and municipality a given generic location belongs to.

Table 19. The name and description of the files containing PAT data

Extraction of the macro-classes. The classes extracted from the *features* file are very generic and represent heterogeneous kinds of locations grouped together. For this reason we call them *macro-classes*. This is mainly due to the criteria used by the PAT during categorization that were based not only on location type but also on importance and population criteria. In this file each macro-class is associated an identifier (e.g., P110) and an Italian name (e.g., *Monti principali*). We did not process the macro-class with identifier P310 (*Regioni limitrofe*) as it represents locations in the neighbouring of Trento (out of the scope of our interest) and P472 (*Indicatori geografici*) as it represents geographic codes. Since there are no macro-classes explicitly defined for provinces, municipalities, wards and populated places (they are directly encoded as fields in the files) we created 4 additional classes for them as reported in Table 20. Overall 47 macro-classes were imported in the database.

Identifier	**English name**	**Italian name**
E000	province	provincia
E010	municipality	comune
E020	ward	frazione
E021	populated place	località popolata

Table 20. Names of the administrative classes

Extraction of the locations. We imported all the locations into the database by organizing them in a *part-of* hierarchy[61] *province* > *municipality* > *ward* > *populated place* (and other location kinds) as follows:

- ***The province level***. We created an entry representing the Province of Trento. This location is not explicitly defined in the dataset but it is clearly the root of the hierarchy. We assigned the following names to it: *Provincia Autonoma di Trento*, *Provincia di Trento* and *Trento*. It was assigned to the *province* class.
- ***The municipality level***. Municipalities were extracted from the *ammcom* file. We created an entry for each municipality and a *part-of* relation between each municipality and the province. They were assigned to the *municipality* class.
- ***The ward and populated place level***. Wards and populated places (sections of wards) were extracted from the *localita* file. Here each ward is connected to the corresponding municipality and each populated place to the corresponding ward by specific internal

[61] By *part-of* we mean a generic containment relation between locations. It can be administrative or topological containment.

codes. For each ward and populated place we created a corresponding entry. Using the internal codes, each ward was connected to the corresponding municipality and each populated place to the corresponding ward. They were assigned to the class *ward* or *populated place* accordingly.

- ***Other locations***. All other (non-administrative) locations were extracted from the *toponimi* file. Here each of them is connected to a municipality, a ward or a populated place by specific internal codes. Using the internal codes, we connected them accordingly. A few of them are not connected to any place and therefore we directly connected them to the province. Each location in the database was temporarly assigned to the corresponding macro-class.
- Locations are provided with latitude and longitude coordinates in Cartesian WGS84 (World Geodetic System 1984) format, a standard coordinate reference system mainly used in cartography, geodesy and navigation to represent geographical coordinates on the Earth[62]. Since in GeoWordNet coordinates are stored in WGS84 decimal format, for compatibility we converted them accordingly.

Double names: bilingual issues. Locations are always provided with a name and in some case with alternative names. A few names are double names, e.g., *Cresta di Siusi Cresta de Sousc*. The first (*Cresta di Siusi*) is in Italian and the second (*Cresta de Sousc*) is in Ladin. Ladin is a language spoken in a small part of Trentino and other Alpine regions. The combination of the two names is the official name of the location in the PAT. In order to identify these cases, the PAT provided an extra text file for each municipality containing the individual Italian and Ladin version of the names. In the temporary database, we put the Italian and Ladin names as alternative names. These extra files also contain additional name variants, which are also treated as alternative names. In the end, we found 53 additional Italian names, 53 Ladin names and 8 name variants. For instance, for the location *Monzoni*, the Ladin name *Monciogn* and the name variant *Munciogn (poza)* are provided.

Missing data and duplicates removal. While importing the locations in the temporary database, we found that 8 municipalities and 39 wards were missing in the *ammcom* and *localita* files respectively, and 35 municipalities were duplicated in the *ammcom* file. The missing municipalities are due to the fact that they were merged with other municipalities on 1st January 2010,

[62] https://www1.nga.mil/ProductsServices/GeodesyGeophysics/WorldGeodeticSystem/

while the duplicates are related to administrative islands (regions which are not geometrically connected to the main area of each municipality). We automatically created the missing locations and eliminated the duplicates.

From macro-classes to elementary classes. In order to create the faceted ontology, we had to refine the macro-classes into elementary classes. In [Giunchiglia et al., 2009a] each of these classes is called an atomic concept. Since they are not accompanied by any description, it is by analyzing the locations contained in them that each macro-class was disambiguated and refined, i.e. split, merged or renamed. This was done through a statistical analysis. For each macro-class, corresponding locations were searched in GeoWordNet. We looked at all the locations in the *part-of* hierarchy rooted in the Province of Trento having same name and collected corresponding classes. Only a little portion of the locations were found, but they were used to understand the classes corresponding to each macro-class. By using this heuristic we found one-to-one, one-to-many, many-to-one and many-to-many correspondences between macro-classes and GeoWordNet classes. We decided to cluster them in groups accordingly. For instance, all macro-classes related to bodies of water were put in the same cluster. The classes were manually refined and some of them required a deeper analysis (with open discussions). New classes also emerged. With the refinement, we generated 39 elementary classes. In Table 21 we report an example for each kind of correspondence. The first row shows a one-to-one correspondence; the second a one-to-many; the third a many-to-one; the forth a many-to-many correspondence.

Macro-classes	**classes**
P410 Capoluogo di Provincia	Province
P465 Malghe e rifugi	Shelter Farm Hut
P510 Antichita importanti P520 Antichita di importanza minore	Antiquity
P210 Corsi dacqua/laghi (1 ord.) P220 Corsi dacqua/laghi (2 ord.) P230 Corsi dacqua/Canali/Fosse/Cond. forz./Laghi (3 ord.) P240 Corsi dacqua/Canali/Fosse/Cond. forz./Laghi (>3 ord.-25.000) P241 Corsi dacqua/Canali/Fosse/Cond. forz./Laghi (>3 ord.)	Lake Group of lakes Stream River Rivulet Canal

Table 21. Examples of correspondences between macro-categories and elementary classes

8.3. Building the faceted ontology

To build the faceted ontology, the elementary classes and the locations determined with the previous step were arranged into facets. This was done in two steps:

- ***Step 1: Arranging elementary classes into facets***
- ***Step 2: Populating the facets with locations***

***Step 1:* Arranging elementary classes into facets.** With this step, elementary classes identified with the previous step were arranged into facets. Similarly to the work done with the *Space* domain, this was done by following the principles at the basis of the faceted approach, i.e. by progressively grouping the elementary classes according to their differences and commonalities. This lead to the creation of a faceted ontology constituted by five distinct facets:

- *Antiquity*
- *Geological formation* (divided into *natural elevation* and *natural depression*)
- *Body of water*
- *Facility*
- *Administrative division*

As an example, below we provide the *body of water* and *geological formation* facets.

Body of water (Idrografia)

Lake (Lago)
Group of lakes (Gruppo di laghi)
Stream (Corso d'acqua)
River (Fiume)
Rivulet (Torrente)
Spring (Sorgente)
Waterfall (Cascata)
Cascade (Cascatina)
Canal (Canale)

Geological formation (Formazione geologica)

Natural elevation (Rilievo naturale)

Highland (Altopiano)

Hill (Collina, Colle)

Mountain (Montagna, Monte)

Mountain range (Catena montuosa)

Peak (Cima)

Chain of peaks (Catena di picchi)

Glacier (Ghiacciaio, Vedretta)

Natural depression (Depressione naturale)

Valley (Valle)

Mountain pass (Passo)

Step 2: **Populating the facets with locations.** Each location in the temporary database was initially associated a macro-class. The facets were instead built by using the elementary classes generated from their refinement. In order to populate the facets, we assigned each location in the database to the corresponding elementary class by applying some heuristics based on the location names.

First of all, each facet was associated one or more of the groups of macro-classes identified with the refinement. Macro-classes corresponding to the same facet constitute what we call a block of classes. For instance, the 11 macro-classes from P110 (*Monti principali*) to P142 (*Catene-Vedrette-Altipiani di piccola estensione*) correspond to the *natural elevation* block, including *inter-alia* mountains, peaks, passes and glaciers.

Secondly, different heuristics were applied to each block. For instance, within the *natural elevation* block, locations with name starting with *Monte* were considered as instances of the class *montagna* in Italian (*mountain* in English), while locations with name starting with *Passo* were mapped to the class *passo* in Italian (*pass* in English). The general criterion we used is that if we could successfully apply a heuristic we classified the location in the corresponding elementary class otherwise we choose a more generic class, which is the root of a facet (same as the block name) in the worst case. In some cases, for instance for farms and huts, we reached a success rate of 98%. On average, about 50% of the locations were put in a leaf class thanks to the heuristics.

Finally, we applied the heuristics beyond the boundary of the blocks for further refinement. The idea was to understand whether, by mistake, locations were classified in the wrong macro-class. For instance, in the 5 macro-classes from P320 (*Grandi regioni geografiche 1*) to P350 (*Aree di piccola estensione*) that correspond to the *natural depression* block (including locations such as highlands and valleys) we found that 6 locations have name starting with *Monte* and therefore they should be mountains instead. The right place for them would be therefore the *natural elevation* facet. In total we found 48 potentially misclassified locations, which were checked manually. In 41.67% of the cases it revealed that the heuristics were valid, in only 8.33% of the cases the heuristics were invalid, while we could not provide any answer for the remaining 50% of the cases since no information about them could be found in the Web. We moved those considered valid in the right classes.

Some figures about the faceted ontology we developed are reported in Table 22.

Kinds of objects	**Quantity of the objects**
facets	5
classes	39
locations	20,162
part-of relations between locations	20,161
alternative names of locations	7,929

Table 22. Objects identified with the pre-processing

8.4. Evaluation of the semantic extension

We evaluated the discovery service on a dataset of around 800 metadata files used within the platform to index geographical maps. The evaluation was conducted using standard information retrieval metrics and in particular in terms of:

- precision (the number of true positive (relevant) documents found divided by the total number of documents found) of the discovery service without the semantic extension, that we call the *baseline*
- precision of the discovery service with the semantic extension, that we call *semantic search*
- rate of increment (the number of documents found with the semantic expansion divided by the number of document found without it) in the number of documents found
- rate of increment (The number of true positive (relevant) documents found with the semantic expansion divided by the number of true positive (relevant) documents found without it) in the number of true positive documents found with the semantic extension

A log of the queries performed by the users was available. The evaluation was carried out on around 33% of these queries randomly selected. To simplify the evaluation, it was restricted to only those queries returning less than 30 documents.

Some examples of queries are given in Table 23. The first and second columns provide the query in Italian and corresponding translation in English (as it is returned by the platform), respectively. The third column provides the list of those terms that, expanded by the semantic extension, provide non empty results. The fourth and fifth columns provide the results of the baseline and semantic search facilities in terms of total number of documents found (tot), true positive (TP) and false positive (FP) documents, respectively.

Italian term	**English term**	**Expanded terms**	**baseline search**			**semantic search**		
			tot	**TP**	**FP**	**tot**	**TP**	**FP**
fiume	river	alveo (8)	9	8	1	17	16	1
strade	roads	localita' (7), strada provinciale (3), strada (7)	7	4	3	14	4	10
città	city	localita' (7)	2	1	1	9	6	3
bosco	forest	foresta (16)	7	7	0	22	14	8
malè	male		2	1	1	9	2	7
fuoco	fire	incendio (2)	0	0	0	2	2	0
bovini	cattle		1	1	0	1	1	0
alberi	trees		0	0	0	0	0	0
vegetazioni	vegetations		1	0	1	23	14	9
ferrovia	rail		2	2	0	6	6	0
torrente	torrent		2	1	1	2	1	1
treno	train	ferroviario (6)	0	0	0	6	6	0

Table 23 - Examples of queries

The outcome of the evaluation is summarized in Table 24. As it can be seen from the table, the results are pretty satisfactory. In fact, at the price of a drop in precision of 0.16% and by inspecting 2.64 times the initial set of documents we can get around the double of positive documents.

baseline search precision	**semantic search precision**	**Rate of increment of documents found**	**Rate of increment in true positive documents found**
0.81	0.65	2.64	2.12

Table 24. Summary of the evaluation of the discovery service

We also found out that:

- Around 18% of the queries correspond to place names. See for instance *Malè* (a municipality in Trento) in Table 23. At the moment they cannot be expanded by the search facility.

- In 8% of the cases, thanks to the expansion, the semantic search could actually return results when no results where instead found by the baseline. This is for instance the case for the queries *fuoco* and *treno* reported in Table 23.
- When a query term is very generic huge quantities of documents might be given in output as result of the query expansion. However, since results - as shown in the third column in Table 23 - are displayed grouped by expanded term with preview of the number of corresponding documents given in round parenthesis, the user can easily filter the results by selecting only those terms which are considered more relevant.
- In some rare cases the translation facility does not produce a good enough translation. For instance, a better English translation for the Italian term *torrente* is *rivulet*.

8.5. Integration of the faceted ontology with Entitypedia

With the integration of the *Space* domain, Entitypedia contains around 7 million locations from all over the world, but only a few locations for the Province of Trento. This was the motivation that led us to the decision of integrating the faceted ontology with Entitypedia. The integration was done in two sequential phases, the first focusing on classes and the second on locations:

- ***Step 1: Class matching and integration***
- ***Step 2: Location matching and integration***

They are described in detail below.

Step 1: Class matching and integration. This step consisted in mapping elementary classes from the faceted ontology to Entitypedia synsets. This has been done similarly to the way in which GeoNames was mapped with WordNet (see Section 7.2.3). However, while in that case the mapping was conducted fully manually, in this work we partially automated the matching task. More in detail, we used the name of each class to identify in Entitypedia a corresponding synset or, if not available, a more general synset that we call the parent synset. The matching procedure is as follows:

1. ***Identification of the facet synset.*** For each facet, the class at the root is manually mapped with Entitypedia. We call the corresponding synset the *facet synset*. For instance, *body of water* was mapped with the synset "*body of water, water -- (the part of the earth's surface covered with water (such as a river or lake or ocean))*";
2. ***Class Identification.*** Within each facet, for each class we check if there exist in Entitypedia any synset such that it contains the name of the class among the synonyms, it has noun as part of speech and it is more specific than the corresponding facet synset. In affirmative case, we select it otherwise we move to the next step. For instance, *lake* was mapped with the synset "*lake -- (a body of (usually fresh) water surrounded by land)*";
3. ***Parent Identification.*** Here we distinguish two sub-cases. (case a) If the class name starts with either *group of* or *chain of*, we remove this part from the name and lemmatize the remaining part. Similarly to the previous step we then check if there exist in Entitypedia any synset corresponding to the obtained lemma. In affirmative case, all the parent

synsets of the identified synset are selected as parent for the class[63]. For instance, *group of lakes* was converted to *lake* and matched with the synset "*lake -- (a body of (usually fresh) water surrounded by land)*" and then its parent synset "*body of water, water -- (the part of the earth's surface covered with water (such as a river or lake or ocean))*" was selected as parent for *group of lakes*. (case b) If the class name consists of two or more words, we select the last word and we look for a corresponding synset. The corresponding synset, if any, is selected as parent for the class. For instance, *provincial road* was converted into *road* and the synset "*road, route -- (an open way (generally public) for travel or transportation)*", more specific than the facet synset *"facility, installation -- (a building or place that provides a particular service or is used for a particular industry)*" was selected as parent for it. If none of the two approaches can be followed or both fail the parent is manually assigned[64].

With the integration, missing synsets and corresponding description, manually provided in English and Italian, were created in Entitypedia. They were also linked to the corresponding parent through *is-a* relation.

Step 2: Location matching and integration. This step consisted in mapping the locations in the faceted ontology with Entitypedia locations. Several heuristics were experimented and tuned to obtain the highest precision possible. The entity matching task was accomplished within and across the two datasets. The following rules led to the best results, i.e. two entities match if:

- **Rule 1**: name, class and coordinates are the same
- **Rule 2**: name, class, coordinates and parent are the same
- **Rule 3**: name, class, coordinates, parent, children and alternative names are the same

Here by parent and children we mean the parent and children locations according to the *part-of* hierarchy. For example, since *Povo* is (administrative) part of *Trento* then *Trento* is the parent of *Povo*. As it can be noticed, Rule 2 is an extension of Rule 1 and Rule 3 is an extension of Rule 2.

[63] In our case we always found exactly one parent
[64] In our case parent identification never failed

We obtained the following results:

- ***Within Entitypedia.*** By matching Entitypedia with itself, we found 15,665 matches with Rule 1, 12,112 matches with Rule 2 and 12,058 matches (involving 22,641 entities) with Rule 3. By deleting duplicates these entities were reduced to 10,583. In fact, if two or more entities match by Rule 3 we can safely reduce them by keeping one of them and deleting the others. Matching entities are clearly undistinguishable.
- ***Within the faceted ontology.*** By matching the faceted ontology with itself, we found 12 matches with Rule 1 and 11 matches (involving 22 entities) with Rule 2. The result did not change by applying Rule 3 as all of the matched entities are leaves and they have either the same or no alternative name. By deleting duplicates these entities were reduced to 11.
- ***Across the two datasets.*** By applying Rule 1 we found only 2 exact matches between the faceted ontology and Entitypedia, which is far smaller than the number we expected. For this reason, we decided to allow a tolerance while matching coordinates. The results obtained with different offsets are reported in Table 25. At the end the last one was applied, leading to 174 matches. It corresponds to Rule 3 with a tolerance of +/- 5.5 Km. We checked most of them manually and they are undistinguishable. Note that while matching classes across datasets, we took into account the *is-a* hierarchy. For example, Trento as *municipality* in the faceted ontology is matched with Trento as *administrative division* in Entitypedia. In fact, the former is more specific than the latter. Note also that the heuristic above aims only at minimizing the number of duplicated entities but it cannot prevent the possibility of still having some duplicates. However, further relaxing it would generate false positives. For instance, by dropping the condition of having same children we found 5% (1 over 20) of false matches.

Same name	Same class	Same coordinates	Same parent	Same Children
1385	1160	2 (exact match)	0	0
		11 (using the offset +/-0.0001)	0	0
		341 (using the offset +/-0.001)	13	12
		712 (using the offset +/-0.01)	65	60
		891 (using the offset +/-0.05)	194	174

Table 25. Matching coordinates with tolerance

With the integration, we imported all but the overlapping locations from the faceted ontology to Entitypedia. For each location, we created the following attributes:

- **Name**: it codifies English and Italian names and alternative names
- **Description**: a natural language description of the location in English and Italian
- **Latitude**: it codifies the latitude coordinate
- **Longitude**: it codifies the longitude coordinate

We also created an *instance-of* relation between each location and the corresponding class and *part-of* relations between the locations according to the information stored in the temporary database. Note that natural language descriptions were automatically generated following the same rules used when importing GeoNames entities.

8.6. Open Government Data

The open definition[65] states that *"a piece of content or data is open if anyone is free to use, re-use, and redistribute it - subject only, at most, to the requirement to attribute and share-alike"*. Therefore, by open government data we mean those content, data or information produced, or commissioned, by government that are made available following the open definition. To be compliant with this definition it is important to appropriately choose the format and the license used to publish the data.

Tim Berners-Lee designed the 5-stars rating system, in which he identifies 5 levels that can be used to evaluate the openness of data made available on the Web [Bizer et al., 2008]. It is quite easy to reach the 3 stars, since it is enough to publish data in a structured and non-proprietary format (for instance as CSV instead of MS Excel files), and under an open license. More challenging is to achieve 4 or 5 stars. In fact, to reach the 4 stars data has to be encoded using an open standard such as RDF and the most important data elements have to be identified by a URI, while to reach the 5 starts they have to be linked to other relevant datasets in the Web. Data achieving 4 or 5 stars is easier to find and recombine with data coming from other sources.

The Open Knowledge Foundation (OKF) community provides a list[66] with data and content licenses that are conformant with the open definition. In this paper we are interested only to data licenses. In this respect, two families are declared compliant: Open Data Commons[67] (ODC) and Creative Commons[68] (CC). ODC licenses were designed explicitly to support use, reuse and redistribution of open data, while CC licenses were studied to support use, reuse and redistribution of creative works. Creative Commons Attribution (CC-BY) and Creative Commons Attribution Share-Alike (CC-BY-SA) licenses are not included in this list. We think that this happens because usually in Europe data is not considered a creative work.

In Europe there is a growing interest on this topic. In Italy, for example, many communities have been recently created to promote OGD activities. Among others it is worth mentioning Data-

[65] http://www.opendefinition.org
[66] http://opendefinition.org/licenses
[67] http://www.opendatacommons.org/
[68] http://creativecommons.org/

Gove.it[69], wishing to promote an open and transparent government in Italy, and Trentino Open Data[70], aiming at sensitize public awareness of open data issues stating from the Trentino region. Moreover, in Italy many public administrations are working to publish their datasets following the principles stated by the open definition. Piemonte region, for example, has already published a dataset in the Web[71] achieving 3 stars of the 5 stars rating system.

The UK follows the INSPIRE Directive[72], which requires to have, among others, a resolvable unique identifier for each spatial object [European Commission, 2007]. To meet these requirements UK publishes their data as Linked Data [Berners-Lee, 2006] where objects are named using resolvable HTTP URIs [Sheridan and Tennison, 2010]. Moreover, the UK government has decided to publish their data using open standards, e.g., RDF for representation [Brickley and Guha, 2004], SPARQL Endpoint for exposing [Berners-Lee, 2006] [W3C, 2008], DCMI (Dublin Core Metadata Initiative) vocabulary for annotation [Dublin Core Metadata Initiative, 2010] and GML (Geography Markup Language) for representing geographic features [Cox et al., 2004]. Essentially, the use of a SPARQL Endpoint for exposing data allows the Semantic Web search engines (for instance Sindice[73], Swoogle[74] and Watson[75]) to discover, crawl and index the RDF data which in turn helps increasing the visibility of the data. Ordnance Survey[76], the national mapping agency in the UK, spearheaded the publishing of geospatial information as part of the Linked Data [Goodwin et al., 2009].

In Portugal, some progress has been recently made towards publishing some datasets as Linked Data. The *Geo-Net-PT 02* [Lopez-Pellicer et al., 2010] dataset was created at the University of Lisbon to support applications requiring geographic information about Portugal. This dataset is published in RDF and linked to Yahoo!GeoPlanet[77]. Standard vocabularies were used including, for instance, DCMI for metadata and WGS84[78] vocabulary for geographical coordinates. This dataset is also used as a geo-spatial ontology and a SPARQL Endpoint is provided for querying it.

---

[69] http://www.datagov.it/
[70] http://www.trentinoopendata.eu/
[71] http://dati.piemonte.it/
[72] http://www.ec-gis.org/inspire/
[73] http://www.sindice.com/
[74] http://swoogle.umbc.edu/
[75] http://watson.kmi.open.ac.uk/
[76] http://www.ordnancesurvey.co.uk
[77] http://developer.yahoo.com/geo/geoplanet/
[78] http://www.w3.org/2003/01/geo/wgs84_pos

Substantial work in the arena of Linked Data has been done in Spain. The GeoLinked Data initiative [Blázquez et al., 2010] at the University Politecnica de Madrid has contributed bringing Spanish geographic and statistical information to the Linked Data. They have dealt with the data sources owned by the Spanish National Geographic Institute (IGN-E)[79] and Spanish National Statistical Institute (INE)[80]. The produced dataset, called *GeoLinked Data*, is linked to GeoNames and DBPedia[81]. For the representation of the statistical (e.g., unemployment rate), geometrical (e.g., shape) and geo-positioning (e.g., geographical coordinates) information, Statistical Core Vocabulary (SCOVO)[82], GML and WGS84 vocabularies were used, respectively.

In the context of the SGC project we have dealt with the authoritative geographic data of the Trentino region managed by the PAT. Among the existing datasets, the PAT decided to disclose those concerning the streams and bicycle tracks. Data was available as shape files, while metadata was available as XML files. To make them part of the Linked Data cloud [Bizer et al., 2009], we converted both data and metadata in RDF. For the representation of the metadata in RDF we used the DCMI and DCMI-BOX[83] standard vocabularies, while for the data we used WGS84. Following best practices and standards [Bizer et al., 2008], the RDF we produced was linked to the most relevant vocabularies, i.e. DBPedia, Freebase[84], GeoNames and Open Street Map[85].

To demonstrate the effectiveness of the data published and the potential of the Linked Data in linking and using different datasets, we built a mashup application. The application was built to support the following scenario:

> *John is visiting Trento during summer and he is cycling in the bicycle path between Trento and Riva del Garda. While he is cycling in the lakefront region of the Mori-Torbole bicycle track, he is fascinated by the beauty of the lake. Hence, he is willing to know more about the panoramic views of the other parts of the bicycle track and the surrounding hotels to stay there for few days. While cycling in the summer noon he becomes thirsty and therefore he wants to know the position of the drinking water fountains in the neighborhood of the bicycle track.*

[79] http://www.ign.es/
[80] http://www.ine.es/
[81] http://dbpedia.org/
[82] http://vocab.deri.ie/scovo/
[83] http://dublincore.org/documents/dcmi-box/
[84] http://www.freebase.com/
[85] http://www.openstreetmap.org/

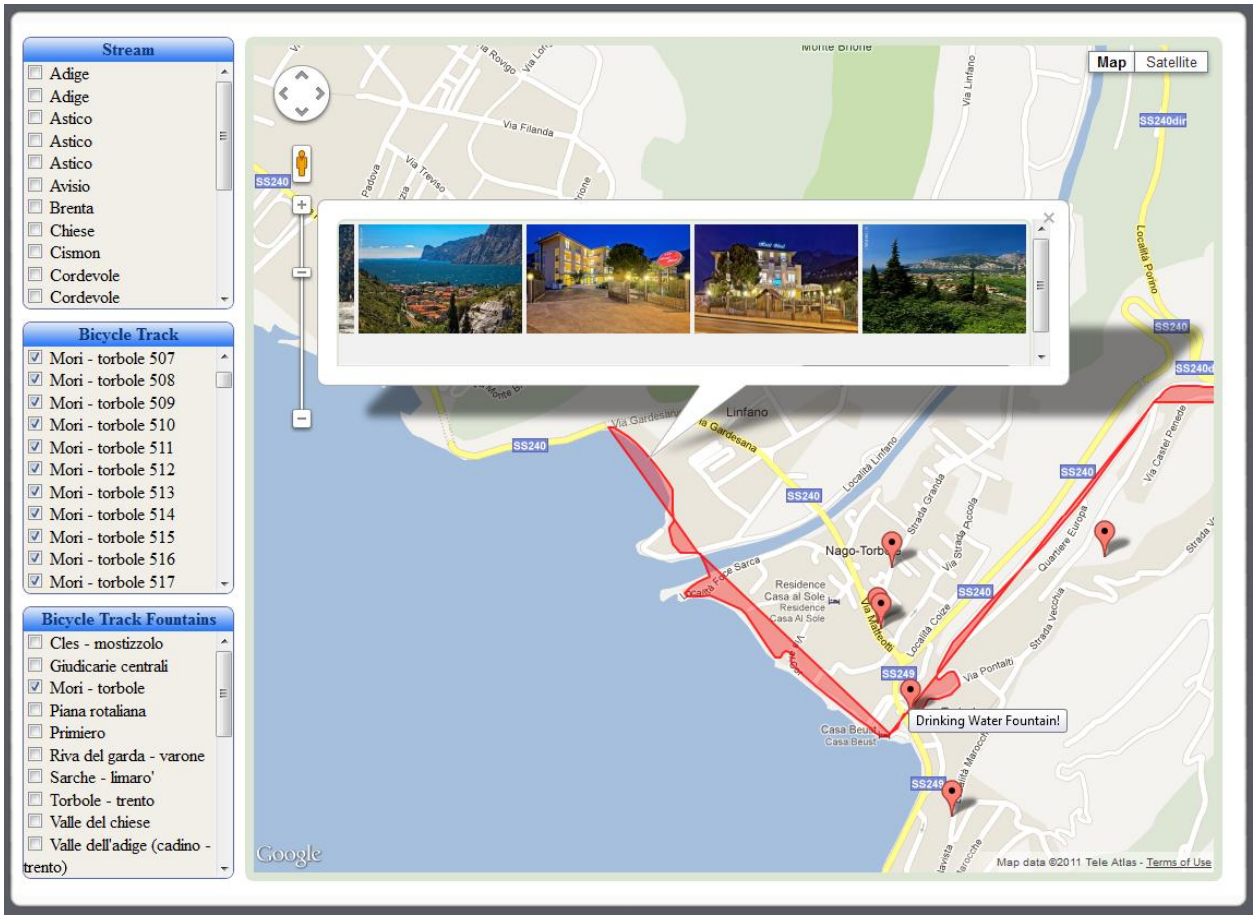


Fig. 36. The mashup developed to support the cyclist-tourist scenario

A snapshot of the mashup supporting this scenario is shown in Fig. 36. Streams (e.g., Adige), bicycle tracks (e.g., Mori - Torbole 507) and bicycle track fountains are shown on the left as a list of check boxes, where the numbers to the right of the tracks represent the identifiers of the track parts which constitute the whole track. Selected streams, bicycle tracks and fountains are displayed using Google Maps[86] as polygons, polylines and markers, respectively. By clicking on a bicycle track it is possible to visualize a set of images of the nearby hotels and panoramic views. We collected images from Flickr[87] and we gathered information about drinking water fountains from Open Street Map through LinkedGeoData[88]. For gathering, combining and merging information from different sources published in RDF, we used DERI Pipes[89], a state-of-the-art Semantic Web mashup tool [Le-Phuoc et al., 2009].

[86] http://maps.google.com/
[87] http://www.flickr.com/
[88] http://linkedgeodata.org/
[89] http://pipes.deri.org/

8.7. Future improvements

Many improvements can be done on the semantic geo-catalogue. One immediate and obvious improvement would consist in substituting WordNet with GeoWordNet (recently made available in WordNet format[90]) as background knowledge, thus having a higher coverage in terms of domain specific terminology. This can be easily done through the latest version of the S-Match libraries (not available when we started the SGC project).

At the moment, since S-Match by default works on English, the system supports queries in Italian through their translation in English. We are currently extending the libraries of S-Match to directly work in Italian. This can be done with the adoption of the Italian part of MultiWordNet and appropriately extending it with domain specific terminology. Among other things, this will avoid possible mistakes due to wrong translation.

S-Match is currently used to expand terms denoting classes only. For instance, in the query *Bodies of water in Trento* only *Bodies of water* is expanded. In the future, it could be extended to allow the expansion of terms denoting locations. For instance, *Trento* could be expanded with its administrative sub-divisions.

[90] Freely available at http://geowordnet.semanticmatching.org/

Chapter 9

9. Conclusions and future work

As significant progress to the problem of matching classifications, with this thesis we presented MinSMatch, a semantic matching tool we developed evolving S-Match that translates the classifications into lightweight ontologies and computes the *minimal mapping* between them. We provided a formal definition of *minimal* and *redundant mappings*, evidence of the fact that the minimal mapping always exists and it is unique and a correct and complete algorithm for computing it. Our experiments demonstrate a substantial improvement in run-time, given a significant saving in the number of calls to SAT, and a slight improvement in recall. Based on this, we also developed a method to support users in the validation task that allows saving up to 99% of the time. Therefore, minimal mappings have clear advantages in maintenance, visualization and user interaction.

Despite the progress made, one of the main barriers towards the use of semantics is the lack of background knowledge. The solution we proposed is to create a very large and virtually unbound knowledge base, that we called *Entitypedia*, able to capture the diversity of the world in language, knowledge and personal experience. The approach is centered on the two fundamental notions of *domain* (from library science) and *context* (from artificial intelligence). Domains are developed using a general semantic-aware methodology and technique for structuring the background knowledge originated from the *faceted approach*, a well-established methodology used with profit for decades in library science for the organization of knowledge. Using standard techniques, context is built at run-time as a fundamental tool to reduce reasoning at run-time.

By comparing Entitypedia with respect to pre-existing systems, our knowledge base has at least the following distinctive features:

- There is a clear split between natural language, formal language and knowledge
- There is an explicit definition of domain as a way to codify knowledge which is local to a community thus reflecting their specific purpose, needs, competences, beliefs and personal experience
- There is an explicit distinction between classes, entities, relations and attributes

- It is completely modular, in the sense that it can be continuously extended with knowledge about new domains and new vocabularies
- Domain knowledge is created following a precise methodology and principles inspired by well-established library science methodologies and practices
- Domain knowledge is used to construct the context formalized (given the specific tasks we want to serve) as a propositional DL theory and therefore the complexity of reasoning is limited to propositional reasoning
- It does not only consist of a data repository, but it comes with a framework[91] to support a precise set of basic semantic tasks including natural language understanding, automatic classification, semantic matching and semantic search by encoding knowledge in the most appropriate semantics according to the task at hand

These features are summarized in the table below. As we can see from it, we can consider the combination of SUMO plus MILO as the closest in spirit to our approach.

Knowledge base	**#entities**	**#facts**	**Domains**	**Distinction concepts instances**	**Distinction NL/FL**	**Manually built**	**Framework included**
YAGO	2.5 M	20 M	No	No	No	No	No
CYC	250,000	2.2 M	Yes	No	No	Yes	No
OpenCYC	47,000	306,000	Yes	No	No	Yes	No
SUMO	1,000	4,000	No	Yes	Yes	Yes	No
MILO	21,000	74,000	Yes	Yes	Yes	Yes	No
DBPedia	3.5 M	500 M	No	No	No	No	No
Freebase	22 M	unknown	No	Yes	No	Yes	No
Entitypedia	10 M	80 M	Yes	Yes	Yes	Yes	Yes

Table 26. Comparison of existing knowledge bases in terms of support to diversity

Nevertheless, as a drawback of the proposed approach, in order to guarantee the high quality of the knowledge, its construction and maintenance requires a significant amount of manual work. In fact, building a domain may take several weeks of work by an expert familiar with the classical faceted approach and the novelties introduced by our methodology. For instance, bootstrap-

[91] The framework has been developed in the KnowDive group. My contribution was in the semantic matching component.

ping the *Space* domain - that, given its pervasiveness, is among the biggest ones - took around 6 man months. Other domains should take much less. We plan to overcome this issue by adopting crowdsourcing techniques integrated with a certification pipeline based on ideas already exploited on ESP games [Von Ahn, 2006]. Given the precise split that we enforce between concepts and instances, we plan to establish two pipelines: the first for experts at the purpose of defining the basic terminology of domains, in terms of classes, relations and attributes (the TBox); the second for generic users at the purpose of providing actual data for the entities (the ABox). The main reason for this distinction is that the first requires a higher level of expertise. At this purpose, in the context of the Living Knowledge project we already conducted some training activities with our partners at the Indian Statistical Institute where some library science students were asked to use our methodology for the construction of sample domains. Notice how the second pipeline will have to be able to manage a quantity of knowledge which is several orders of magnitude bigger that the first.

When possible, given format and quality of the data, ready-made entities can be directly imported from existing sources. This is for instance what we did for the population of the *Space* domain from GeoNames and we are currently experimenting with YAGO. At this purpose and as core of Entitypedia, we are already working on a *theory of entity types* which allows a more rigorous definition and support to entity management and corresponding applications. We strongly believe into Entitypedia as future provider of high quality up-to-date semantics on large scale.

Bibliography

[Abdelmoty et al., 2007] Abdelmoty, A. I., Smart, P., Jones C. B. (2007). Building Place Ontologies for the Semantic Web: issues and approaches. 4th ACM workshop on GIR.

[Adorni et al., 1984] Adorni, G., Di Manzo, M., Giunchiglia, F. (1984). Natural Language Driven Image Generation. 10th International Conference on Computational Linguistics COLING, 495-500.

[Aleksovski et al., 2008] Aleksovski, Z., ten Kate, W., van Harmelen, F. (2008). Using multiple ontologies as background knowledge in ontology matching. ESWC workshop on collective semantics.

[Angioni et al., 2007] Angioni, M., Demontis, R., Tuveri, F. (2006). Enriching WordNet to Index and Retrieve Semantic Information. 2nd Int. Conf. on Metadata and Semantics Research.

[Arpinar et al., 2004] Arpinar, I. B., Sheth, A., Ramakrishnan, C. (2004). Geospatial ontology development and semantic analytics. Handbook of Geographic Information Science, J. P. Wilson, A. S. Fotheringham (Eds.), Blackwell Pub.

[Auer et al., 2007] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, C., Ives, Z. (2007). DBpedia: A Nucleus for a Web of Open Data, 6th International Semantic Web Conference ISWC.

[Auer et al., 2009] Auer, S., Lehmann, J., Hellman, S. (2009). LinkedGeoData - Adding a Spatial Dimension to the Web of Data. 8th World Int. Semantic Web Conference (ISWC).

[Autayeu et al., 2010] Autayeu, A., Giunchiglia, F., Andrews, P. (2010). Lightweight parsing of classifications into lightweight ontologies. European Conference on Research and Advanced Technology for Digital Libraries (ECDL), 327-339.

[Avesani et al., 2005] Avesani, P., Giunchiglia, F., Yatskevich, M. (2005). A Large Scale Taxonomy Mapping Evaluation. International Semantic Web Conference (ISWC), 67-81.

[Baader et al., 2002] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. F. (2002). The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press.

[Banko et al, 2007] Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., Etzioni, O. (2007). Open information extraction from the web. IJCAI conference.

[Bates, 1990] Bates, M. J. (1990). Where should the person stop and the information search interface start? Inf. Process. Manage, 26 (5), 575-591.

[Battacharyya, 1975] Battacharyya, G. (1975). POPSI: its fundamentals and procedure based on a general theory of Subject Indexing Languages. Library Science with a slant to doc. 16 (1), 1-34.

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web. Scientific America, 34-43.

[Berners-Lee, 2006] Berners-Lee, T. (2006). Linked Data. http://www.w3.org/DesignIssues/LinkedData.html

[Bizer et al., 2008] Bizer, C., Cyganiak, R., Heath, T. (2008). How to publish Linked Data on the Web. http://www4.wiwiss.fu-berlin.de/bizer/pub/linkeddatatutorial/

[Bizer et al., 2009] Bizer, C., Heath, T., Lee, T. B. (2009). Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems, 5(3), 1-22.

[Blázquez et al., 2010] Blázquez, L. M. V., Villazón-Terrazas, B., Saquicela, V., de León, A., Corcho, Ó., Gómez-Pérez, A. (2010). GeoLinked data and INSPIRE through an application case. In GIS, 446–449.

[Brickley and Guha, 2004] Brickley, D., Guha, R. V. (2004). RDF Vocabulary Description Language 1.0: RDF Schema, (Editors), W3C Recommendation.

[Borgida and Serafini, 2003] Borgida, A., Serafini, L. (2003). Distributed Description Logics: Assimilating Information from Peer Sources. Journal on Data Semantics, 153-184.

[Bollacker et al., 2008] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge, ACM SIGMOD international conference on Management of data, 1247-1250.

[Bouquet et al., 2003a] Bouquet, P., Serafini, L., Zanobini, S. (2003). Semantic coordination: A new approach and an application. International Semantic Web Conference (ISWC), 130–145.

[Bouquet et al., 2003b] Bouquet, P., Ghidini, C., Giunchiglia, F., Blanzieri, E. (2003). Theories and uses of context in knowledge representation and reasoning. Journal of Pragmatics, 35(3).

[Broughton, 2006] Broughton, V. (2006). The need for a faceted classification as the basis of all methods of information retrieval. Aslib Proceedings, 58(1/2), 49-72.

[Broughton, 2008] Broughton, V. (2008). A Faceted Classification as the Basis of a Faceted Terminology: Conversion of a Classified Structure to Thesaurus Format in the Bliss Bibliographic Classification, 2nd Edition. Axiomathes Journal, Springer Online Issue, 18 (2), 193-210.

[Broughton and Slavic, 2007] Broughton, V., Slavic, A. (2007). Building a faceted classification for the humanities: principles and procedures. J Doc 63(5), 727–754.

[Buchanan and Lederberg, 1971] Buchanan, B. G., Lederberg, J. (1971). The Heuristic DENDRAL program for explaining empirical data, Stanford University, technical report.

[Buscardi and Rosso, 2008] Buscardi, D., Rosso, P. (2008). Geo-wordnet: Automatic Georeferencing of wordnet. 5th Int. Conference on Language Resources and Evaluation (LREC).

[Chaves et al., 2005] Chaves, M. S., Silva, M. J., Martins, B. (2005). A Geographic Knowledge Base for Semantic Web Applications. 20th Brazilian Symposium on Databases (SBBD).

[Cox et al., 2004] Cox, S., Daisey, P., Lake, R., Portele, C., Whiteside, A. (2004). OpenGIS Geography Markup Language (GML) Implementation Specification, version 3.1.0.

[Crompvoets et al., 2004] Crompvoets, J., Wachowicz, M., de Bree, F., Bregt, A. (2004). Impact assessment of the INSPIRE geo-portal. 10th EC GI&GIS workshop.

[Davey and Priestley, 2002] Davey, B. A., Priestley, H. A. (2002). Introduction to Lattices and Order (2nd edition).

[David et al., 2007] David, J., Guillet, F., Briand, H. (2007). Association rule ontology matching approach. International Journal on Semantic Web and Information Systems, 3(2), 27–49.

[David and Euzenat, 2008] David, J., Euzenat, J. (2008). On fixing semantic alignment evaluation measures. Third International Workshop on Ontology Matching (OM).

[Dean, 2003] Dean, R. J. (2003). FAST: Development of Simplified Headings for Metadata. In Authority Control: Definition and International Experiences conference.

[Devadason, 2002] Devadason, F. J. (2002). Faceted Indexing Based System for Organizing and Accessing Internet Resources. Knowledge Organization, 29 (2), 65-77.

[Do and Rahm, 2002] Do, H., Rahm, E. (2002). COMA - a system for flexible combination of schema matching approaches. VLDB.

[Do et al., 2002] Do, H., Melnik, S., Rahm, E. (2002). Comparison of schema matching evaluations. Second International Workshop on Web Databases, 221–237.

[Dousa, 2007] Dousa, T. (2007). Everything Old is New Again: Perspectivism and Polyhierarchy in Julius O. Kaiser's Theory of Systematic Indexing. In Lussky, Joan, Eds. 18th Workshop of the American Society for Information Science and Technology Special Interest Group in Classification Research.

[Dublin Core Metadata Initiative, 2010] Dublin Core Metadata Initiative (2010). Dublin Core Metadata Element Set, Version 1.1: Reference Description, http://dublincore.org-/documents/dces/

[Egenhofer, 2002] Egenhofer, M. J. (2002). Toward the Semantic GeoSpatial Web. In the 10th ACM International Symposium on Advances in Geographic Information Systems (ACM-GIS), 1-4.

[Egenhofer and Dupe, 2009] Egenhofer, M. J., Dube, M. P. (2009. Topological Relations from Metric Refinements. 17th ACM SIGSPATIAL Int. Conference on Advances in GIS.

[Egenhofer and Herring, 1991] Egenhofer, M. J., Herring, J. (1991). Categorization binary topological relationships between regions, lines, and points in geographic databases. In "A Framework for the Definition of Topological Relationships and an Approach to Spatial Reasoning within this Framework", M. Egenhofer and J. Herring (Eds.), Santa Barbara, CA.

[Ehrig and Euzenat, 2005] Ehrig, M., Euzenat, J. (2005). Relaxed precision and recall for ontology matching. Integrating Ontologies Workshop.

[English et al., 2003] English, J., Hearst, M., Sinha, R., Swearingen, K., Lee, K. P. (2003). Flexible search and navigation using faceted metadata. Technical report, University of Berkeley, School of Information Management and Systems.

[European Commission, 2007] European Commission (2007) INSPIRE Technical Architecture – Overview.

[European Commission, 2009] European Commission, 2009. "COMMISSION REGULATION (EC) No 976/2009 implementing Directive 2007/2/EC as regards the Network Services".

[European Parliament, 2009] European Parliament, 2009. "Directive 2007/2/EC establishing an Infrastructure for Spatial Information in the European Community (INSPIRE)".

[Euzenat, 2007] Euzenat, J. (2007). Semantic precision and recall for ontology alignment evaluation. International Joint Conference of Artificial Intelligence (IJCAI), 348–353.

[Euzenat et al, 2011] Euzenat, J., Ferrara, A., van Hage, W. R., Hollink, L. Meilicke, C., Nikolov, A., Ritze, D. Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Šváb-Zamazal, O., Trojahn, C. (2011). First results of the Ontology Alignment Evaluation Initiative. Ontology Matching Workshop.

[Etzioni et al., 2004] Etzioni, O., Cafarella, M. J., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D. S., Yates, A. (2004). Web-scale information extraction in KnowItAll, WWW conference.

[Falconer and Storey, 2007] Falconer, S., Storey, M. (2007). A cognitive support framework for ontology mapping. International Semantic Web Conference (ISWC).

[Fellbaum, 1998] Fellbaum, C. (1998). WordNet: An Electronic Lexical Database, MIT Press.

[Fuchs et al., 2006] Fuchs, N. E., Kaljurand, K., Schneider, G. (2006). Attempto controlled english meets the challenges of knowledge representation, reasoning, interoperability and user interfaces. FLAIRS Conference, 664-669

[Giunchiglia and Walsh, 1989] Giunchiglia, F., Walsh, T. (1989). Abstract Theorem Proving, 11th International Joint Conference on Artificial Intelligence IJCAI, 372-377.

[Giunchiglia, 1993] Giunchiglia, F. (1993). Contextual reasoning, Epistemologica - Special Issue on I Linguaggi e le Macchine, 16, 345–364.

[Giunchiglia et al., 1997] Giunchiglia, F., Villafiorita. A., Walsh, T. (1997). Theories of Abstraction. AI Communications 10 (3/4), 167-176.

[Giunchiglia et al., 2005] Giunchiglia, F., Yatskevich, M., Giunchiglia, E. (2005). Efficient semantic matching. European semantic Web Conference (ESWC), 272-289.

[Giunchiglia, 2006] Giunchiglia, F. (2006). Managing Diversity in Knowledge. Invited Talk at the European Conference on Artificial Intelligence ECAI, Lecture Notes in Artificial Intelligence.

[Giunchiglia et al., 2006] Giunchiglia, F., Shvaiko, P., Yatskevich, M. (2006). Discovering missing background knowledge in ontology matching. ECAI conference, 382–386.

[Giunchiglia et al., 2007a] Giunchiglia, F., Marchese, M., Zaihrayeu, I. (2007). Encoding Classifications into Lightweight Ontologies. Journal of Data Semantics, 8, 57-81.

[Giunchiglia et al., 2007b] Giunchiglia, F., Zaihrayeu, I., Kharkevich U. (2007). Formalizing the get-specific document classification algorithm. European Conference on Research and Advanced Technology for Digital Libraries.

[Giunchiglia et al., 2007c] Giunchiglia, F., Yatskevich, M., Shvaiko, P. (2007). Semantic matching: Algorithms and implementation. Journal on Data Semantics, 9.

[Giunchiglia and Zaihrayeu, 2008] Giunchiglia, F., Zaihrayeu, I. (2008). Lightweight ontologies. Encyclopedia of Database Systems.

[Giunchiglia et al., 2008] Giunchiglia, F., Yatskevich, M., Avesani, P., Shvaiko, P. (2008). A large dataset for the evaluation of ontology matching systems. The Knowledge Engineering Review Journal, 24, 137–157.

[Giunchiglia et al., 2009a] Giunchiglia, F., Dutta, B., Maltese, V. (2009). Faceted Lightweight Ontologies. In: Conceptual Modeling: Foundations and Applications, A. Borgida, V. Chaudhri, P. Giorgini, Eric Yu (Eds.) LNCS 5600 Springer.

[Giunchiglia et al., 2009b] Giunchiglia, F., Soergel, D., Maltese, V., Bertacco, A. (2009). Mapping large-scale Knowledge Organization Systems. 2nd International Conference on the Semantic Web and Digital Libraries (ICSD).

[Giunchiglia et al., 2009c] Giunchiglia, F., Zaihrayeu, I., Farazi, F. (2009). Converting classifications into owl ontologies. Artificial Intelligence and Simulation of Behaviour Convention Workshop on Matching and Meaning.

[Giunchiglia et al., 2009d] Giunchiglia, F., Kharkevich, U., Zaihrayeu, I. (2009). Concept search. European Semantic Web Conference (ESWC).

[Giunchiglia et al, 2010] Giunchiglia, F., Maltese, V., Farazi, F., Dutta, B. (2010). GeoWordNet: a resource for geo-spatial applications. 7th Extended Semantic Web Conference (ESWC).

[Giunchiglia et al, 2012] Giunchiglia, F., Maltese, V., Dutta, B. (2012). Domains and context: first steps towards managing diversity in knowledge. Journal of Web Semantics, special issue on Reasoning with Context in the Semantic Web.

[Ghidini and Giunchiglia, 2001] Ghidini, C., Giunchiglia, F. (2001). Local Model Semantics, or Contextual Reasoning = Locality + Compatibility, Artificial Intelligence, 127 (2), 221–259.

[Goodwin et al., 2009] Goodwin, J., Dolbear, C., Hart, G. (2009). Geographical linked data: The administrative geography of Great Britain on the semantic web. Transaction in GIS 12(1).

[Gruber, 1993] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge Aquisition, 5 (2), 199–220.

[Guha and Lenat, 1993] Guha, R., Lenat, D. (1993). Context dependence of representations in cyc. Colloque ICO.

[Halevy, 2005] Halevy, A. (2005). Why your data won't mix. ACM Queue, 3(8), 50–58.

[Hoder and Voronkov, 2011] Hoder, K., Voronkov, A. (2011). Sine qua non for large theory reasoning. International Conference on Automated Deduction.

[Horrocks and Sattler, 1999] Horrocks, I., Sattler, U. (1999). A description logic with transitive and inverse roles and role hierarchies. Journal of Logic and Computation, 9(3), 385–410.

[Isaac et al, 2009] Isaac, A., Wang, S., Zinn, C., Matthezing, H., van der Meij, L., Schlobach, S. (2009). Evaluating thesaurus alignments for semantic interoperability in the library domain. IEEE Intelligent Systems, 24(2), 76–86.

[Janowicz et al., 2009] Janowicz, K., Schade, S., Bröring, A., Keßler, C., Stasch, C., Maue', P., Diekhof, Y. (2009). A transparent Semantic Enablement Layer for the Geospatial Web. Terra Cognita Workshop.

[Jones et al., 2003] Jones, C. B., Adbelmoty, A. I., Fu, G. (2003). Maintaining Ontologies for Geographical Information Retrieval on the Web. On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE, Lecture Notes in Computer Science.

[Kaiser, 1911] Kaiser, J. (1911). Systematic indexing. London: Isaac Pitman & Sons

[Kaza and Chen, 2008] Kaza, S., Chen, H. (2008). Evaluating ontology mapping techniques: An experiment in public safety information sharing. Decision Support Systems, 45(4), 714–728.

[Keßler et al., 2009] Keßler, C., Janowicz, K., Bishr, M. (2009). An agenda for the Next Generation Gazetteer: Geographic Information Contribution and Retrieval. Int. Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS).

[Koch et al., 2003] Koch, T., Neuroth, H., Day, M. (2003). Renardus: Cross-browsing European subject gateways via a common classification system (DDC). I.C. McIlwaine (Ed.), Subject retrieval in a networked environment. Proceedings of the IFLA satellite meeting held in Dublin - IFLA Information Technology Section and OCLC, 25–33.

[Kolas et al., 2005] Kolas, D., Dean, M., Hebeler, J. (2005). Geospatial Semantic Web: architecture of ontologies. First Int. Conference on GeoSpatial Semantics (GeoS), 183-194.

[Kules et al., 2009] Kules, B., Capra, R., Banta, M., Sierra, T. (2009). What do Exploratory Searchers Look at in Faceted Search Interface? In JCDL, 313-322.

[Kuhn, 2005] Kuhn, W. (2005). Geospatial semantics: Why, of What, and How? Journal of Data Semantics (JoDS), 3, 1–24.

[La Barre, 2004] La Barre, K. (2004). Adventures in faceted classification: A brave new world or a world of confusion? Knowledge organization and the global information society, 8th ISKO conference.

[Lauser et al., 2008] Lauser, B., Johannsen, G., Caracciolo, C., Keizer, J., van Hage, W. R., Mayr, P. (2008). Comparing human and automatic thesaurus mapping approaches in the agricultural domain. International Conference on Dublin Core and Metadata Applications.

[Le-Phuoc et al., 2009] Le-Phuoc, D., Polleres, A., Hauswirth, M., Tummarello, G., Morbidoni, C. (2009). Rapid Prototyping of Semantic Mash-ups through Semantic Web Pipes. WWW Conference.

[Lopez-Pellicer et al., 2010] Lopez-Pellicer, F., Silva, M., Chaves, M., Javier Zarazaga-Soria, F., Muro-Medrano, P. (2010). Geo linked data. 21st International Conference on Database and expert systems applications (DEXA'10), Vol. 6261, Springer-Verlag, 495-502.

[Lutz et al., 2009] Lutz, M., Ostlander, N., Kechagioglou, X., Cao, H. (2009). Challenges for Metadata Creation and Discovery in a multilingual SDI - Facing INSPIRE. ISRSE conference.

[Madalli and Prasad, 2011] Madalli, D. P., Prasad, A. R. D. (2011). Analytico synthetic approach for handling knowledge diversity in media content analysis. UDC seminar.

[Madhavan et al, 2001] Madhavan, J., Bernstein, P., Rahm, E. (2001). Generic schema matching with Cupid. Very large Databases (VLDB).

[Madhavan et al, 2002] Madhavan, J., Bernstein, P., Domingos, P., Halevy, A. Y. (2002). Representing and Reasoning about Mappings between Domain Models. 18th National Conference on Artificial Intelligence (AAAI).

[Magnini et al., 2003] Magnini, B., Serafini, L., Speranza, M. (2003). Making explicit the semantics hidden in schema models. Workshop on Human Language Technology for the Semantic Web and Web Services held at ISWC.

[Magnini et al., 2004] Magnini, B., Speranza, M., Girardi, C. (2004). A semantic-based approach to interoperability of classification hierarchies. Evaluation of linguistic techniques, COLING.

[Maltese and Farazi, 2011] Maltese, V., Farazi, F. (2011). Towards the Integration of Knowledge Organization Systems with the Linked Data Cloud, UDC seminar.

[Marchionini and Shneiderman, 1988] Marchionini, G., Shneiderman, B. (1988). Finding Facts v.s. Browsing Knowledge in Hypertext Systems. Computer, 21 (1), 70–80.

[Marchionini, 2006] Marchionini, G. (2006). Exploratory search: from finding to understanding. Communications of the ACM, 49(4), 41-46.

[Matuszek et al., 2006] Matuszek, C., Cabral, J., Witbrock, M., DeOliveira, J. (2006). An introduction to the syntax and content of Cyc, AAAI Spring Symposium.

[McCarthy, 1987] McCarthy, J. (1987). Generality in artificial intelligence, Communications of ACM, 30, 1030–1035.

[McCarthy, 1993] McCarthy, J. (1993). Notes on formalizing context. Bajcsy, R. (Ed.), Thirteenth International Joint Conference on Artificial Intelligence, IJCAI, 555-560.

[Meilicke et al, 2006] Meilicke, C., Stuckenschmidt, H., Tamilin, A. (2006). Improving automatically created mappings using logical reasoning. 1st International Workshop on Ontology Matching (OM), CEUR Workshop Proceedings Vol. 225.

[Meilicke and Stuckenschmidt, 2008] Meilicke, C., Stuckenschmidt, Heiner, (2008). Incoherence as a basis for measuring the quality of ontology mappings. Third International Workshop on Ontology Matching (OM).

[Meilicke et al, 2008] Meilicke, C., Stuckenschmidt, H., Tamilin, A. (2008). Reasoning support for mapping revision. Journal of Logic and Computation.

[Melnik et al., 2002] Melnik, S., Garcia-Molina, H., Rahm, E. (2002). Similarity flooding: a versatile graph matching algorithm. ICDE.

[Melnik et al., 2003] Melnik, S., Rahm, E., Bernstein, P. (2003). Developing metadata-intensive applications with Rondo. Journal of Web Semantics.

[Mihalcea and Moldovan, 2001] Mihalcea, R., Moldovan, D. I. (2001). Automatic generation of a coarse grained wordnet, NAACL Workshop on WordNet and Other Lexical Resources.

[Miles and Bechhofer, 2009] Miles, A., Bechhofer, S. (2009). SKOS Reference. W3C Recommendation.

[Miller, 1998] Miller, G. A. (1998). WordNet: An electronic Lexical Database. MIT Press.

[Miller and Hristea, 2006] Miller, G. A., Hristea, F. (2006). WordNet Nouns: classes and instances. Computational Linguistics, 32(1), 1 – 3.

[Mills, 2004] Mills, J. (2004). Faceted classification and logical division in information retrieval, Library trends, 52 (3), 541-570.

[Nicholson et al., 2006] Nicholson, D., Dawson, A., Shiri, A. (2006). HILT: A pilot terminology mapping service with a DDC spine. Cataloging & Classification Quarterly, 42 (3/4), 187-200.

[Norsk Polarinstitutt, 2005] Norsk Polarinstitutt (2005). Artic Climate Impact Assessment, Cambridge University Press, 973.

[Noy and Musen, 2002] Noy, N., Musen, M. A. (2002). Evaluating ontology mapping tools: Requirements and experience. OntoWebSIG3 Workshop, 1–14.

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

[O'Neill and Chan, 2003] O'Neill, E., Chan, L. (2003). FAST (Faceted Application for Subject Technology): A Simplified LCSH-based Vocabulary. World Library and Information Congress: 69th IFLA General Conference and Council, 1-9.

[Pease et al., 2010] Pease, A., Sutcliffe, G., Siegel, N., Trac, S. (2010). Large theory reasoning with SUMO at CASC. AI Communications, 23(2-3), 137–144.

[Pollock, 2002] Pollock, J. (2002). Integration's Dirty Little Secret: It's a Matter of Semantics. Whitepaper, The Interoperability Company.

[Prusak, 1997] Prusak, L. (1997). Knowledge in Organizations. Cap. 7: The tacit dimension by M. Polanyi.

[Pullar and Egenhofer, 1988] Pullar, D., Egenhofer, M. J. (1988). Toward formal definitions of topological relations among spatial objects. 3rd International Symposium on Spatial Data Handling, Sydney, Australia, 165–176.

[Ranganathan, 1967] Ranganathan, S. R. (1967). Prolegomena to library classification. Asia Publishing House.

[Ranganathan, 1965] Ranganathan, S. R. (1965). The Colon Classification. In S. Artandi, editor, Vol IV of the Rutgers Series on Systems for the Intellectual Organization of Information. New Brunswick, NJ: Graduate School of Library Science, Rutgers University.

[Robertson et al., 2005] Robertson, G. G., Czerwinski, M. P., Churchill, J. E., 2005. Visualization of mappings between schemas. SIGCHI Conference on Human Factors in Computing Systems.

[Roman et al., 2006] Roman, D., Klien, E., Skogan, D. (2006). SWING – A Semantic Web Service Framework for the Geospatial Domain. Terra Cognita Workshop.

[Sabou and Gracia, 2008] Sabou, M., Gracia, J. (2008) Spider: Bringing non-equivalence mappings to OAEI. Third International Workshop on Ontology Matching (OM).

[Schwitter and Tilbrook, 2006] Schwitter, R., Tilbrook, M. (2006). Lets talk in description logic via controlled natural language. LENLS.

[Shamdasani et al., 2009] Shamdasani, J., Hauer, T., Bloodsworth, P., Branson, A., Odeh, M., McClatchey, R. (2009). Semantic Matching Using the UMLS. European Semantic Web Conference (ESWC).

[Sheridan and Tennison, 2010] Sheridan, J., Tennison. J. (2010) Linking UK Government Data. WWW2010 workshop: Linked Data on the Web (LDOW), ACM.

[Shvaiko and Euzenat, 2007] Shvaiko, P., Euzenat, J., 2007. Ontology Matching. Springer-Verlag New York, Inc. Secaucus, NJ, USA.

[Shvaiko and Euzenat, 2008] Shvaiko, P., Euzenat, J., 2008. Ten Challenges for Ontology Matching. 7th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE).

[Shvaiko et al., 2010a] Shvaiko, P., Ivanyukovich, A., Vaccari, L., Maltese, V., Farazi, F. (2010). A semantic geo-catalogue implementation for a regional SDI. INPSIRE Conference.

[Shvaiko et al., 2010b] Shvaiko, P., Giunchiglia, F., Yatskevich, M. (2010). Semantic matching with S-Match. Semantic Web Information Management: a Model-Based Perspective, XX, 183–202.

[Smith and Mark, 1998] Smith, B., Mark, D. M. (1998). Ontology and geographic kinds. International Symposium on Spatial Data Handling, Vancouver, Canada.

[Smits and Friis-Christensen, 2007] Smits, P., Friis-Christensen, A. (2007). Resource discovery in a European Spatial Data Infrastructure. Transactions on Knowledge and Data Engineering, 19(1), 85–95.

[Soergel, 1972] Soergel, D. (1972). A general model for indexing languages: The basis for compatibility and integration. Subject Retrieval in the Seventies - New Directions New York: Greenwood; College Park, MD.: University of Maryland, 36-61.

[Soergel et al., 2004] Soergel, D., Lauser, B., Liang, A., Fisseha, F., Keizer, J., Katz, S. (2004). Reengineering thesauri for new applications: the agrovoc example, Journal of Digital Information, 4.

[Spiteri, 1998] Spiteri, L. (1998). A Simplified Model for Facet Analysis. Journal of Information and Library Science, 23, 1-30.

[Stuckenschmidt et al, 2006] Stuckenschmidt, H., Serafini, L., Wache, H. (2006). Reasoning about Ontology Mappings. ECAI-06 Workshop on Contextual Representation and Reasoning.

[Studer et al., 1998] Studer, R., Benjamins, V. R., Fensel, D. (1998). Knowledge engineering: principles and methods, Data and Knowledge Engineering, 25, 161–197.

[Suchanek et al., 2011] Suchanek, F. M., Kasneci, G., Weikum, G. (2011). YAGO: A Large Ontology from Wikipedia and WordNet, Journal of Web Semantics.

[Uschold and Gruninger, 2004] Uschold, M., Gruninger, M. (2004). Ontologies and semantics for seamless connectivity. SIGMOD Rec., 33(4), 58–64.

[Vaccari et al., 2009] Vaccari, L., Shvaiko, P., Marchese, M. (2009). A geo-service semantic integration in spatial data infrastructures. Journal of Spatial Data Infrastructures Research, 4, 24–51.

[van Hage et al., 2007] van Hage, W. R., Isaac, A., Aleksovski, Z. (2007). Sample evaluation of ontology-matching systems. EON, 41–50.

[Varzi, 2006] Varzi, A. C. (2006). A note on the transitivity of parthood. Applied Ontology, 1, 141-146.

[Vizine-Goetz et al., 2004] Vizine-Goetz, D., Hickey, C., Houghton, A., Thompson, R. (2004). Vocabulary Mapping for Terminology Services. Journal of Digital Information, 4(4), article n. 272.

[Von Ahn, 2006] Von Ahn, L. (2006). Games with a purpose, IEEE Computer Magazine, 96-98.

[Vorz et al., 2007] Vorz, R., Kleb, J., Mueller, W. (2007). Towards ontology-based disambiguation of geographical identifiers. 16th WWW Conference.

[Yee et al., 2003] Yee, K. P., Swearingen, K., Li, K., Hearst, M. (2003). Faceted metadata for image search and browsing. Conference on Human Factors in Computing Systems, 401-408.

[W3C, 2008] W3C (2008). SPARQL Query Language for RDF. W3C Recommendation, http://www.w3.org/TR/rdf-sparql-query/

[White et al., 2006] White, R. W., Kules, S. B., Drucker, M., Schraefel, M.C. (2006). Supporting exploratory search. Communications of the ACM, 49(4), 36-39.

[White et al., 2007] White, R. W., Drucker, M., Marchionini, G., Hearst, M., Schraefel, M.C. (2007). Exploratory search and HCI: designing and evaluating interfaces to support exploratory search interaction. In CHI2007 Extended Abstracts on Human Factors in Computing Systems, ACM Press.

[Whitehead, 1990] Whitehead, C. (1990). Mapping LCSH into Thesauri: the AAT Model. In Beyond the Book: Extending MARC for Subject Access, 81.

[Zaihrayeu et al., 2007] Zaihrayeu, I., Sun, L., Giunchiglia, F., Pan, W., Ju, Q., Chi, M., Huang, X. (2007). From web directories to ontologies: Natural language processing challenges. International Semantic Web Conference (ISWC).

Appendix A: Proofs of the theorems. **Conclusions** and future work

Appendix A: Proofs of the theorems

10. Soundness and completeness of the redundancy

Theorem 1 (Redundancy, soundness and completeness). Given a mapping M between two lightweight ontologies O_1 and O_2, a mapping element m' $\in$ M is logically redundant w.r.t. another mapping element m if and only if it satisfies one of the conditions of Definition 5.

Proof:

Soundness: The argumentation provided in Section 3.2 as a rationale for the patterns already provides a full demonstration for soundness.

Completeness: We can demonstrate the completeness by showing that we cannot have redundancy in the cases which do not fall in the conditions listed in Definition 5. We proceed by enumeration, negating each of the conditions. There are some trivial cases we can exclude in advance:

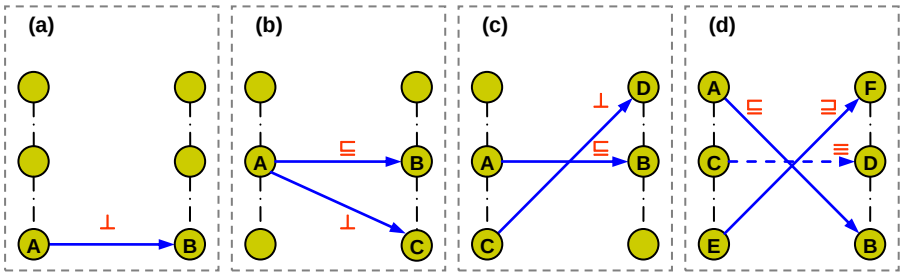


Fig. 37. Some trivial cases which do not fall in the redundancy patterns

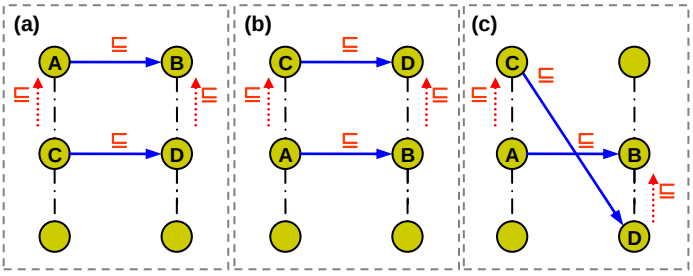
- **One element.** The trivial case in which m' is the only mapping element between the lightweight ontologies. See Fig. 37 (a);
- **Incomparable symbols.** The only cases of dependency across symbols are captured by conditions (1) and (2) in Definition 5, where equivalence can be used to derive the redundancy of a more or less specific mapping element. This is due to the fact that equivalence is exactly the combination of more and less specific. No other symbols can be expressed in terms of the others. This means for instance that we cannot establish implications between an element with more specific and one with disjointness. In Fig. 37 (b) the two elements do not influence each other;

Appendix A: Proofs of the theorems. **Soundness** and completeness of the redundancy

- **Inconsistent nodes.** See for instance Fig. 37 (c). If we assume the element <A, B, ⊑> to be correct, then according to pattern (1) the mapping element between C and D should be <C, D, ⊑>. However, in case of inconsistent nodes the stronger semantic relation ⊥ holds. The algorithm presented in section 4 correctly returns ⊥ in these cases;
- **Underestimated strength.** In includes those cases of underestimated strength not covered by the previous cases, namely the cases in which equivalence holds instead of the (weaker) subsumption. Look for instance at Fig. 37 (d). The two subsumptions in the mapping elements <A, B, ⊑> and <E, F, ⊑> must be equivalences. As a consequence, <C, D, ≡> is redundant for pattern (4). In fact, the chain of subsumptions E ⊑ … ⊑ C ⊑ … ⊑ A ⊑ B ⊑ … ⊑ D ⊑ … ⊑ F allows to conclude that E ⊑ F holds and therefore we can conclude that E ≡ F. Symmetrically, we can conclude that A ≡ B. Note that the mapping elements <A, B, ⊑> and <E, F, ⊒> are minimal. We identify the strongest relations by propagation (at step 3 of the proposed algorithm, as described at the beginning of Section 3.3).

We refer to all the other cases as the *meaningful cases*.

Condition (1): its negation is when R ≠ ⊑ or A ∉ path(C) or D ∉ path(B). The cases in which R = ⊑ are shown in Fig. 38. For each case, the provided rationale shows that available axioms cannot be used to derive C ⊑ D from A ⊑ B. The remaining meaningful cases, namely only when R = ≡, are similar.

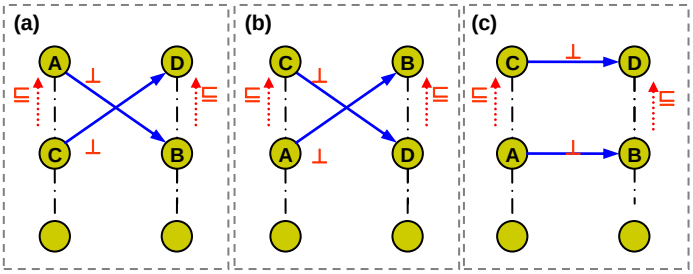


	A ∉ path(C)	D ∉ path(B)	Rationale
(a)	NO	YES	C ⊑ … ⊑ A, D ⊑ … ⊑ B, A ⊑ B cannot derive C ⊑ D
(b)	YES	NO	A ⊑ … ⊑ C, B ⊑ … ⊑ D, A ⊑ B cannot derive C ⊑ D
(c)	YES	YES	A ⊑ … ⊑ C, D ⊑ … ⊑ B, A ⊑ B cannot derive C ⊑ D

Fig. 38 - Completeness of condition (1)

Condition (2): it is the dual of condition (1).

Condition (3): its negation is when R $\neq \perp$ or A $\notin$ path(C) or B $\notin$ path(D). The cases in which R $= \perp$ are shown in Fig. 39. For each case, the provided rationale shows that available axioms cannot be used to derive C $\perp$ D from A $\perp$ B. There are no meaningful cases for R $\neq \perp$.

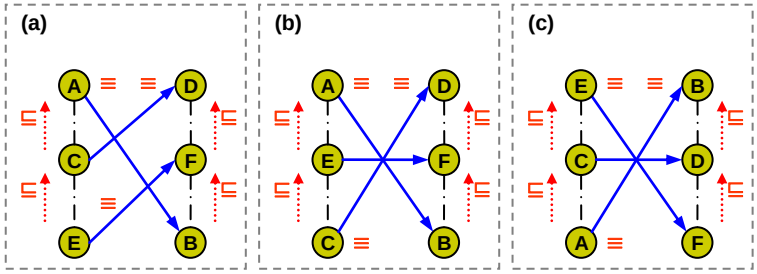


	A $\notin$ path(C)	B $\notin$ path(D)	Rationale
(a)	NO	YES	C $\sqsubseteq$ … $\sqsubseteq$ A, B $\sqsubseteq$ … $\sqsubseteq$ D, A $\perp$ B cannot derive C $\perp$ D
(b)	YES	NO	A $\sqsubseteq$ … $\sqsubseteq$ C, D $\sqsubseteq$ … $\sqsubseteq$ B, A $\perp$ B cannot derive C $\perp$ D
(c)	YES	YES	A $\sqsubseteq$ … $\sqsubseteq$ C, D $\sqsubseteq$ … $\sqsubseteq$ B, A $\perp$ B cannot derive C $\perp$ D

Fig. 39 - Completeness of condition (3)

Condition (4): it can be easily noted from Fig. 10 that the redundant elements identified by pattern (4) are exactly all the mapping elements m' = <C, D, $\equiv$> with source C and target D respectively between (or the same of) the source node and target node of two different mapping elements m = <A, B, $\equiv$> and m'' = <E, F, $\equiv$>. This configuration allows to derive from m and m'' the subsumptions in the two directions which amount to the equivalence. The negation of condition 4 is when R $\neq \equiv$ in m or m'' or A $\notin$ path(C) or D $\notin$ path(B) or C $\notin$ path(E) or F $\notin$ path(D). In almost all the cases (14 over 15) in which R $= \equiv$ we just move the source C or the target D outside these ranges. For brevity we show only some of such cases in Fig. 40. The rationale provided for cases (a) and (b) shows that we cannot derive C $\equiv$ D from A $\equiv$ B and E $\equiv$ F. The only exception (the remaining 1 case over 15), represented by case (c), is when A $\notin$ path(C) and D $\notin$ path(B) and C $\notin$ path(E) and F $\notin$ path(D). This case however is covered by condition 4 by in-

verting the role of m and m''. The remaining cases, namely when R ≠ ≡ in m or m'', are not meaningful.



	A ∉ path(C)	D ∉ path(B)	C ∉ path(E)	F ∉ path(D)	Rationale
(a)	NO	NO	NO	YES	E ⊑ … ⊑ C, C ⊑ … ⊑ A, B ⊑ … ⊑ F, F ⊑ … ⊑ D, A ≡ B and E ≡ F cannot derive C ≡ D (we can only derive C ⊑ D).
(b)	NO	NO	YES	YES	C ⊑ … ⊑ E, E ⊑ … ⊑ A, B ⊑ … ⊑ F, F ⊑ … ⊑ D, A ≡ B and E ≡ F cannot derive C ≡ D (we can only derive C ⊑ D).
…					
(c)	YES	YES	YES	YES	Covered by condition (4) inverting the roles of m and m''

Fig. 40 - Completeness of condition (4)

This completes the demonstration.□

11. Existence and uniqueness of the minimal mapping

Theorem 2 (Minimal mapping, existence and uniqueness). Given two lightweight ontologies O_1 and O_2, there is always one and only one minimal mapping between them.

Proof:

The proof is based on the observation that Definition 5 enforces a strict partial order over mapping elements, while they are not ordered otherwise. Given two mapping elements m, m' $\in$ M, we say that m' < m iff m' is redundant w.r.t. m. The fact that this ordering is partial is a direct consequence of the tree structure of lightweight ontologies.

Under the strict partial order above, if we "open" equivalence relations in the two subsumptions of opposite direction, the minimal mapping is the set of all the *maximal elements* of the partially ordered set, where subsumptions of opposite direction involving the same nodes are collapsed into a (minimal) equivalence mapping element. For the properties of partial orders, this set always exists and it is unique. □