An incremental method for meaning elicitation of a domain ontology

Sonia Bergamaschi
and Laura Po
and Maurizio Vincini
DII-Università di Modena e Reggio Emilia
via Vignolese 905, 41100 Modena
Italy
Email: firstname.lastname@unimore.it

Paolo Bouquet
and Daniel Giacomuzzi
DIT - Università di Trento
Via Sommarive 14, 38050 Trento
Italy
Email: firstname.lastname@unitn.it

Francesco Guerra
DEA-Università di Modena e Reggio Emilia
v.le Berengario 51, 41100 Modena
Italy
Email: firstname.lastname@unimore.it

***Abstract*— Internet has opened the access to an overwhelming amount of data, requiring the development of new applications to automatically recognize, process and manage information available in web sites or web-based applications. The standard Semantic Web architecture exploits ontologies to give a shared (and known) meaning to each web source elements.**

In this context, we developed MELIS (Meaning Elicitation and Lexical Integration System). MELIS couples the lexical annotation module of the MOMIS system with some components from CTXMATCH2.0, a tool for eliciting meaning from several types of schemas and match them. MELIS uses the MOMIS' *WNEditor* and CTXMATCH2.0 to support two main tasks in the MOMIS ontology generation methodology: the source annotation process, i.e. the operation of associating an element of a lexical database to each source element, and the extraction of lexical relationships among elements of different data sources.

I. Introduction

The growth of information available on the Internet has required the development of new methods and tools to automatically recognize, process and manage information available in web sites or web-based applications. The aim of the semantic web is to build a web of data by providing a common framework which enables data sharing and reuse across application, enterprise, and community boundaries. In this way, semantic web applications may automatically discover, exchange and use data without any human intervention. Different W3C recommendations contribute to the implementation of the Semantic Web; here we only mention the Resource Description Framework (RDF), RDF-Schema and the Web Ontology Language (OWL) . These languages can then be used to describe knowledge about resources[1], according to a schema (RDF-Schema) or to a more sophisticated web ontology. The use of shared schemas and ontology should provide a well-defined basis of shared meanings for data integration and reuse.

[1]A resource can be anything that has identity. [. . .] A resource is not necessarily accessible via the Internet; e.g., human beings, corporations, and bound books in a library can also be resources. Likewise, abstract concepts can be resources, such as the operators and operands of a mathematical equation, the types of a relationship (e.g., "parent" or "employee"), or numeric values (e.g., zero, one, and infinity). [RFC3986], `http://www.gbiv.com/protocols/uri/rfc/rfc3986.html`.

However, practical experience in developing semantic-enabled web sites and information systems shows that the simple and intriguing vision sketched above is not a solution to all problems. In particular, we stress the following issues:

1) selecting an appropriate ontology for describing an application's data may be very difficult. Indeed, engineering a new ontology from scratch can be extremely time consuming, and requires appropriate skills; and finding a pre-existing ontology which perfectly fits local data is very unlikely, as most available ontologies are either too generic (and therefore semantically poor) or too specific (and therefore not suited for data different from those of the original application);
2) because of the intrinsically distributed nature of knowledge on the web, different applications may refer to different ontologies to specify the meaning of their data. A lot of effort has been put in developing techniques for ontology matching and reconciliation, but we are still quite far from a general and robust method, and the precision of existing tools tend to be quite low for business applications;
3) finally, there is no standard recommendation or specification for referencing ontologies in information sources. Several proposals and tools have been developed for including references to ontologies in HTML pages. However, such operation is typically executed off-line by adding "annotations" to the content sources.

In this paper, we present MELIS (**M**eaning **E**licitation and **L**exical **I**ntegration **S**ystem), a tool for supporting the source annotation process, i.e. the operation of associating an element of a lexical database to each source element, and the extraction of lexical relationships among elements of different data sources. MELIS has been coupled with the MOMIS system (**M**ediator envir**O**nment for **M**ultiple **I**nformation **S**ystems) [5] providing a (partial) solution to the issues listed above. To achieve this goal, MELIS adapts and integrates components from CTXMATCH2.0 [6], a tool for eliciting meaning from several types of schemas and match them.

Works related to the issues discussed in this paper are in the area of languages and tools for annotations ([2], [14] and [10] where an approach similar to our is adopted), techniques for extending WordNet ([9], [12] and [13] where a system coupled with Protègè[2] for enriching and annotating sources is proposed), and systems for ontology management (see the the Ontoweb[3] and the Knowledgeweb Network of Excellence[4] technical reports for complete surveys).

II. The MOMIS architecture coupled with MELIS

MOMIS starts from a collection of data sources and provides a collection of tools for:

1) semi-automatically building a customized ontology which represents the information sources;
2) annotating each source according to the resulting ontology;
3) mapping the created ontology to a lexical database (WordNet[5]) to support interoperability with other applications.

MELIS has been experimented in MOMIS in order to improve the MOMIS methodology in two main directions: it supports the semi-automatic annotation of the original data sources, and provides methods for extracting rich relationships across terms by exploiting lexical and domain knowledge.

The key idea of MELIS is that Semantic Web information systems require a double level of annotation: *conceptual annotations* and *lexical annotations*. Conceptual annotations provide a specification of how some terminology is used to describe some domain (the standard role of OWL ontologies); lexical annotations provide a natural and rich connection between formal objects (e.g. OWL classes and properties) and their *intended* meaning. The intuition is that grasping the intended interpretation of an OWL ontology requires both un understanding of the formal properties of the conceptual schema, but also knowledge about the meaning of labels used for the ontology elements. In other words, an OWL ontology can be viewed as a collection of *formal* constraints between terms, whose meaning depends also on lexical knowledge. As we will argue, annotating ontology elements with lexical information is crucial, as it allows us to exploit in a much deeper way the original data sources for synthesizing a suitable ontology, and increases interoperability across ontologies.

MOMIS already provides this double level of annotation for data sources and the resulting ontology: for each source, conceptual annotations map the original structure into a formalized ontology and lexical annotations assign a reference to a WordNet element for each source term. Moreover, the ontology structure is formalized by means of a standard model and each concept is annotated according to a lexical database. MELIS allows a greater automation in the process of source annotation, and provides a way for automatically extracting knowledge about the relationships across lexical elements, which means a much richer semantics which can be exploited in integration and interoperability.

[2]http://protege.stanford.edu/
[3]http://www.ontoweb.org, in particular deliverable 1.4
[4]http://knowledgeweb.semanticweb.org/
[5]http://wordnet.princeton.edu

Figure 1 shows the MOMIS architecture coupled with the MELIS component, where the process of creating the ontology and defining the mappings is organized in five step: (1) local source schema extraction, (2) lexical knowledge extraction performed with MELIS, (3) common thesaurus generation, (4) GVV generation, and (5) GVV and local sources annotation. The following sections describe the details of these steps.

A. Local source schema extraction

To enable MOMIS to manage web pages and data sources, we need specialized software (wrappers) for the construction of a semantically rich representations of the information sources by means of a common data model. Wrappers in MOMIS logically converts the data source structure into the internal object language ODL$_{I^3}$.

B. Lexical knowledge extraction

The extraction of lexical knowledge from data sources is typically based on an annotation process aiming at defining a fixed meaning to each source element according to WordNet.

MELIS supports the user in this task by providing two tools for extending the lexical database and for executing the task incrementally, i.e. by iteratively exploiting the annotation made to a subset of the source elements to infer the correct meaning to the whole source (see section III).

C. Common thesaurus generation

The common thesaurus is a set of relationships describing inter- and intra-schema knowledge about the source schemas. The following ODL$_{I^3}$relationships may be specified:

- SYN (Synonym-of), defined between two terms that are considered synonyms in every considered source.
- BT (Broader Terms), defined between two terms such as the first one has a broader, more general meaning than the second one. The opposite of BT is NT (Narrower Terms).
- RT (Related Terms) defined between two terms that are generally used together in the same context.

The common thesaurus is constructed through a process that incrementally adds four types of relationships: schema-derived relationships, lexicon derived relationships, designer-supplied relationships and inferred relationships.

- **Schema-derived relationships.** The system automatically extracts these relationships by analyzing each schema separately and applying a heuristic defined for the specific kind of source managed. For example, when analyzing XML data files, MOMIS generates BT and NT relationships from pairs of IDs and IDREFs.
- **Lexicon-derived relationships.** These relationships are generated by MELIS exploiting a component of CTX-MATCH2.0 to extract complex relationships between meanings of terms annotated with lexical senses. These relationships may be inferred not only from lexical knowledge (e.g. by querying WordNet for relationships across senses), but also from background knowledge (e.g.

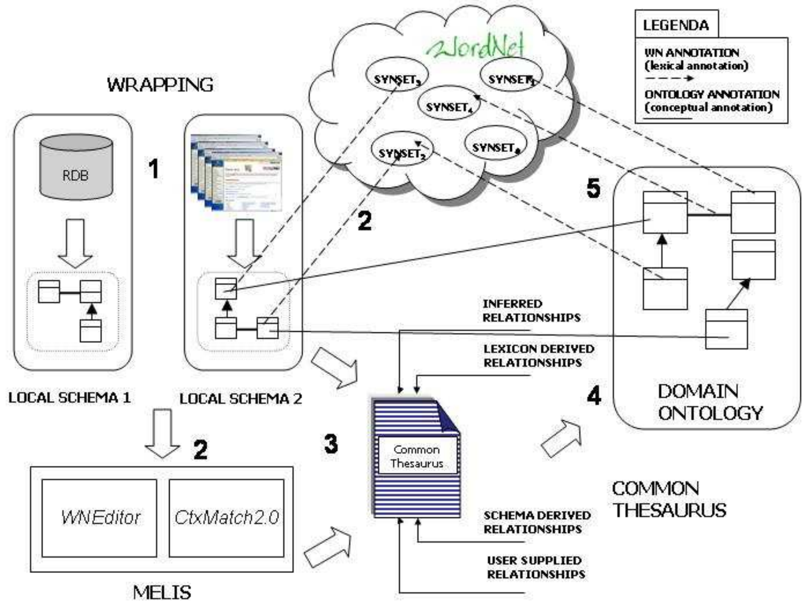


Fig. 1. Functional representation of MOMIS and MELIS

domain ontologies) which are available at the time of the annotation. As we will say later (section III), at any step MELIS can (re)use any piece of ontology generated by the current extraction process as a source of domain knowledge to incrementally refine the extraction of new relationships.

- **Designer-supplied relationships.** To capture specific domain knowledge, designers can supply new relationships directly.
- **Inferred relationships.** MOMIS exploits description logics techniques from ODB-Tools [4] to infer new relationships by applying subsumption computation to virtual schemas obtained by interpreting BT and NT as subclass relationships and RT as domain attributes.

D. GVV generation

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

Going into details, the GVV generation is a process where ODL$_{I^3}$ classes describing the same or semantically related concepts in different sources are identified and clusterized in the same global class based on the affinity among classes. The Ontology Designer may interactively refine and complete the proposed integration results; in particular, the mappings which have been automatically created by the system can be finely tuned.

E. GVV and local sources annotation

Exploiting the local schemas according to WordNet and the mappings between local and global schemas, the system assigns name and meaning to each element of the global schema. Such name and meaning has to be confirmed by the designer.

III. MELIS: THE LEXICAL KNOWLEDGE COMPONENT

In general, annotating a source according to a lexical database requires a heavy user involvement. Turning the process to fully automatic is very hard, as:

- a complete lexical database including all possible terms does not exist. WordNet, for example, contains general terms, other specialized lexical databases are specific domain lexicons;
- there are several polysemous terms: the choice of the specific meaning associated to the term is context dependent;
- it is difficult to associate meaning to the relationship between term in a compound term. For example, what are the meaning of the relationship in “University of Modena”, “table leg”, “football team”?
- a standard model/language for describing lexical databases does not exist. Consequently, it is difficult to integrate different lexical resources.

Several tools which were developed for annotating sources only provide a GUI helping the user in the manual execution of the task (see the introduction for some references). The lexical knowledge module in MELIS adds two jointly working tools to a standard GUI: *WNEditor* [3] and CTXMATCH2.0 [6], [7]. *WNEditor* supports the user in extending the available lexical database, CTXMATCH2.0 supports the process of incremental annotations and the extraction of lexical relationships.

A. The WNEditor

During the annotation of source schemas, the ontology designer is asked to select WordNet meanings for each element of a schema. More precisely, the WordNet-designer firstly chooses the *word form*: the WordNet morphological processor stems each name and eliminates suffixes due to declination/conjugation. Then, given a word form, the designer has to manually solve possible ambiguities by mapping the given name on zero, one or more WordNet senses. When WordNet does not contain satisfactory word forms and/or meanings (or does not contain word forms and/or meanings at all), the item name is considered unknown and the semantic richness of data sources is lost. *WNEditor* aids the designer during the creation of (additional) specific-domain lexicon addressing two main issues: the reference ontology has to be physically extended and a way for sharing such extensions has to be developed.

1) Extending WordNet: WordNet is distributed *as-it-is* and external applications are not allowed to directly modify its data files. *WNEditor* for extending WordNet has to face two main issues:

1) developing a physical structure (i.e. a relational database) where storing the elements of the original WordNet and the new extensions;
2) developing a technique to support the user in extending WordNet.

The second issue is concerned to the criticalness of the extension process due to the complexity of the lexical database. Thus, the designer should have the possibility to perform step-by-step operations, such as (1) providing definitions for new synsets (glosses or meanings), (2) providing new lemmas (word forms) and (3) building relationships between added synsets and the pre-existing ones.

1) **Inserting new synsets.** In order to insert new synsets without introducing any redundancy, the designer has to pre-emptively check if a similar synset already belongs to the database. It is a typical problem of "ontology alignment", where the goal is to define semantic correspondences between elements of heterogeneous systems. Under the assumption that two definitions of the same concept may share at least one significant word, we propose an *approximate matching technique* that computes a *syntactic* and *semantic* similarity measure between the definitions associated to two synsets.
   The semantic similarity function exploits the heuristic known in literature as *definition match* approach, based on the edit distance or the name match[11] function. In particular, two different well-known IR techniques are implemented: *vector space model match*[1] and *Latent Semantic Indexing match*[8].
2) **Inserting new lemmas.** We developed an *approximate string match* algorithm to perform the similarity search on the whole synset network based on the edit distance and on a *reverse index*, representing, at every moment, the set of terms used within the reference ontology to build senses' definitions.
3) **Inserting new relationships.** given a new concept as source synset, the designer should be helped in searching for the most appropriate target synset. The implemented algorithm exploits synset definitions and the definition match heuristic for providing a list of candidate synsets the user has to confirm (see [3] for details).

2) Exporting new annotations: WordNet extensions may be exported in order to be shared by other users/applications. The approach is based on these fundamentals:

- Each user is able to include new lemmas, synsets and relationships in his local WordNet version.
- For exporting an annotated source, the system includes in the exported version both a code identifying the WordNet version and the set of the extended annotations used.
- To allow a user to understand the annotations made by another user, the system temporary loads external WordNet's extensions into a standard WordNet edition. The extensions are not persistently stored in the target system: the user may in case decide if enriching his own WordNet version, performing the inserting of a new synset operation. In this case ontology alignment techniques have to be exploited in order to avoid an incoherent or redundant result.

B. *Incremental annotation using* CTXMATCH2.0

CTXMATCH2.0 is a tool for matching schemas in a Semantic Web environment. The method for matching schemas is based on two main functional steps: *meaning elicitation* and *meaning comparison*. The first, which is thoroughly described in [7], takes as input a bare schema (like a web directory, e.g. the Google directory) and returns an extended schema, where each node is associated with a formal representation (in a Description Logic – DL – language) of its meaning in the context of the given schema. The second takes as input two extended schemas and returns a collection of mappings across pairs of nodes (essentially, a mapping is a triple $\langle n_1, n_2, R \rangle$, where n_1 and n_2 are nodes belonging to different schemas, and R is a logical relationship between the meaning associated to the two nodes, e.g. equivalence or subsumption); mappings are computed through a standard DL theorem prover. For both steps, CTXMATCH2.0 can jointly use lexical databases (like WordNet) and domain ontologies, which provide further background knowledge for selecting the right meaning of words, for computing relationships across different words, and as a source of "axioms" which can be used to support the inference of mappings across nodes of different schemas.

MELIS uses the elicitation component of CTXMATCH2.0 to support the process of annotation and extraction of lexical relationships between elements of a source schema (see Figure 2 for a functional representation of the process). Imagine, for example, a collection of information sources about overlapping domains, e.g. hotels and restaurants. In general we do not assume that a domain ontology is initially available, though this may be the case. The process goes as follows:

1) we start with the first schema (converted in OWL), which can be already partially annotated with lexical information;

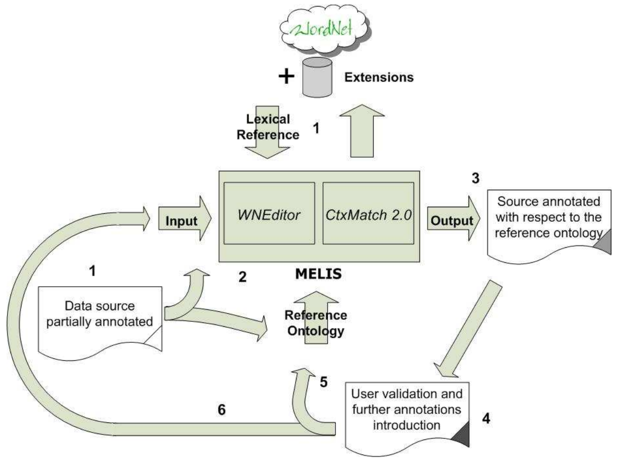

WordNet
+
Extensions
Lexical Reference
1
3
WNEditor
CtxMatch 2.0
Input
Output
Source annotated with respect to the reference ontology
MELIS
1
2
Data source partially annotated
Reference Ontology
5
6
User validation and further annotations introduction
4

Fig. 2. Functional representation of incremental annotation

2) this schema is given as input to CtxMatch2.0, together with a (possibly empty) domain ontology (called reference ontology in Figure 2);
3) CtxMatch2.0 starts the meaning elicitation process, whose output is a complete lexical annotation of schema elements. This annotation is built by using two main knowledge sources: WordNet (for candidate word senses), and the reference ontology (if not empty). [7] describes how WordNet and the reference ontology are used in CtxMatch2.0 for disambiguating labels in the context of the current schema (for example, the label "bank" is very likely to refer to a financial institution – and not to the slope beside a body of water – if its parent node is labeled "organizations"). Notice that pre-existing annotations are not modified, as presumably they come from manual annotation;
4) the resulting annotated schema is passed to a user, who may validate and extend the annotation produced by CtxMatch2.0;
5) the relationships discovered across terms of the schema are added to the reference ontology (which means that an extended version of the ontology is produced);
6) the process restarts with the following schema, if any; otherwise it stops.

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

In the development of the MELIS module, a few specialized heuristics were added to CtxMatch2.0 to improve the precision of results. In what follows, we use the following notational conventions:

- Letters: capital letters (A, B, C,) stand for class labels, low case letters (a, b, c,) stand for datatype property labels, letters followed by "#n" (where n is a natural number) refer to the n-th sense of the label for which the letter stands (e.g. b#2 is the second sense of the word occurring in the label "b").
- Ontologies: O is used for the ontology to be annotated, DO_i for the i-th domain ontology available for the current elicitation process.

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

first, for each (class and property) label in O, the method extracts all candidate senses from WordNet; second, it filters out candidate senses by using a collection of heuristic rules. Below is a detailed description of the heuristic rules used in the second phase of the elicitation process:

Rule 1: If in O we find a class labeled A with a datatype property b, and in some DO_i we find a class annotated as `A#i` with a datatype property annotated as `b#j`, then we conclude that the annotations `A#i` and `b#j` are acceptable candidate annotations for A and b in O.

For example, let us consider a class labeled person with a datatype property address in O, and a DO_i where a class `person` is annotated as `person#1` with a datatype property address annotated as `address#2`. The application of this rule generates the annotation `person#1` for the class person and `address#2` for its datatype property address.

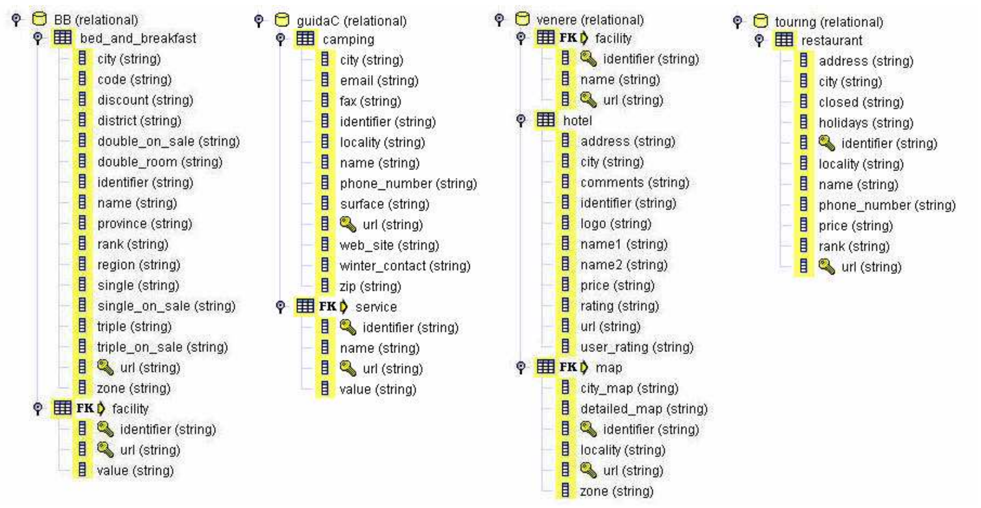
Rule 2: if in O we find a class labeled A with a datatype property b, and in some DO_i we find a class annotated as `B#j` , with a datatype property annotated as `b#k` and a subclass[6] `A#i`, then we conclude that the annotations `A#i` and `b#k` are acceptable candidate annotations for A and b in O.

For example, let us introduce a class labeled student with a datatype property address, and let DO_i contain a class `person` annotated as `person#1` with a datatype property address annotated as `address#2`, and a subclass `student` annotated as `student#1` (notice that the class student is hyponym of enrollee, which is hyponym of person). The application of this rule generates the annotation `student#1` for the class labeled student, and `address#2` for its datatype property.

Rule 3: if in O we find a class labeled A with a datatype property b, and in some DO_i we find a class annotated as `A#i` , with a subclass `B#j`, and the latter has associated a datatype property annotated as `b#k`, then we conclude that the annotations `A#i` and `b#k` are acceptable candidate annotations for A and b in O.

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

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



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

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

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

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

Fig. 3. Sources used for evaluating MELIS

Rule 4: if in O we find a class labeled A with a datatype property b, and in some DO_i we find a class annotated as `C#k` with two subclasses annotated as `A#i` and `B#j`, and there is a datatype property annotated `b#h` associated to `B#j`, then we conclude that the annotations `A#i` and `b#h` are acceptable candidate annotations for A and b in O.

For example, let us introduce a class labeled hotel with a datatype property name, and let DO_i contain a class `building` annotated as `building#1` and a subclass `restaurant` annotated as `restaurant#1` with a datatype property name annotated as `name#1`. As also `hotel#1` is hyponom of `building#1`, the application of this rule generates the annotation `hotel#1` for the class labeled hotel, and `name#1` for its datatype property.

Rule 5: If in O we find a pair of classes labeled A and B, connected through any object property, and in DO_i we find a pair of classes annotated as `A#i` and `C#k` connected through any object property, and a generic relationship links `C#k` and `B#j`, then we conclude that the annotations `A#i` and `B#j` are acceptable candidate annotations for A and B in O.

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

Rule 6: if in O we find a pair of classes labeled A and B (with B subclass of A), and in some DO_i we find a subclass hierarchy in which two classes are annotated as `A#i`, . . . , `B#j` (with none, one or more intermediate classes in between), then we conclude that the annotations `A#i` and `B#j` are acceptable candidate annotations for A and B in O.

For example, let us introduce a pair of classes labeled animal and dog (where dog is a subclass of animal), and let DO_i contain this subclass hierarchy: `animal#1`, `vertebrate#1`, `carnivore#1` and `dog#1`. The application of this rule generates the annotations `animal#1` and `dog#1` for the classes animal and dog.

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

IV. Evaluation on a real domain

We test MELIS by building an ontology of a set of data-intensive websites[7] containing data related to the touristic domain (see figure 3), which have been wrapped by inferring a structured schema for each website and storing the data into four relational databases off-line available. The main classes of these sources are: hotel (of the "venere" database), restaurant ("touring" database), camping (guidaC database) and bedandbreakfast (BB database).

As discussed before, the incremental annotation process starts with the annotation of parts of the data sources, i.e. for each source element the ontology designer selects one or more corresponding Wordnet synsets. Figure 4 shows some WordNet synsets related to the sources domain and the lexical relationships among them. In particular:

- "hotel" and "restaurant" are "brother" terms, i.e. they have a common direct hypernym;
- "hotel" , "house", "restaurant" are direct hyponyms of "building": such relationship may seem misleading: typically restaurants are not considered as buildings but places where a service is provided;
- "bed and breakfast" is an hyponym of "building";
- the closest hypernym that "campsite" and "building" share is "physical object", a top level synset in WordNet. This relationship does not allow to find lexical connections between "camping" and the other classes. Consequently, by means of the MELIS component WNEditor, a

[7]http://www.bbitalia.it, http://www.guidacampeggi.com, http://www.venere.com, http://www.touringclub.com

direct relationship between “campsite” and the hierarchy of “building” is introduced.

Notice that the annotation process is a critical process: by annotating the source element ”camping” as the WordNet synset “camping” a mistake would be generated because it means ”the act of encamping”. The correct synset for camping is “campsite”, i.e. ”the site where people can pitch a tent”. Moreover, in order to test all the implemented heuristics,“hotel” has been annotated as its hypernym: “building”.

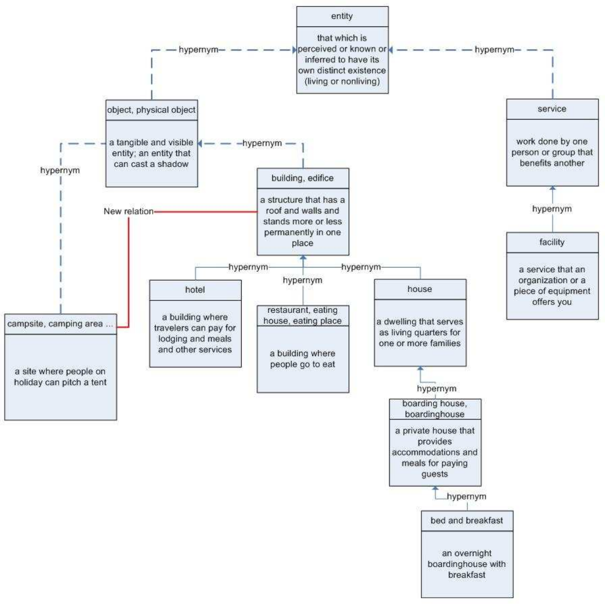

entity
that which is perceived or known or inferred to have its own distinct existence (living or nonliving)
hypernym
hypernym
object, physical object
a tangible and visible entity; an entity that can cast a shadow
service
work done by one person or group that benefits another
hypernym
hypernym
building,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

Fig. 4. Annotations through WNEditor

The annotated schema is then given both as input and as the reference ontology to CtxMatch2.0. The tool starts the meaning elicitation process and produces a set of inferred lexical annotations of the schema elements. The resulting annotated schema is shown to the designer, who may validate and extend the annotation produced by CtxMatch2.0 and, eventually, restart the process using the updated annotated schema as reference ontology.

Figure 5 illustrates the results of a sample test of incremental annotation on one of our schemas. It shows the annotations manually provided by the ontology designer, a fraction of the new annotations generated after a first run of MELIS, and the additional annotations generated after a second run, when the outcome of the first run was provided as additional background knowledge in input; the numbers on the arrows refer to the heuristic rule which was used to generate the annotation. Notationally, a square near a class/attribute means that the element was manually annotated, a circle means that the element was automatically annotated after the first run, and a rhombus that it was incrementally annotated after the second run. In the following, for each heuristic rule, we explain one of the generated annotations.

- Rule 1: the attribute “identifier” of the class “facility” in the source “VENERE” is annotated as “identifier” of the class “facility” in the source “BB” since both the classes are annotated with the same synset.
- Rule 2: because of the hyponym relationships generated by the annotations of the classes “hotel”, “campsite”, “bed and breakfast” and “building”, the attribute “city” of the class “building” in the source “VENERE” produces the annotation of the same attribute in the sources “BB”, “touring”, “guidaC”.
- Rule 3: because of the hypernym relationships generated by the annotations of “building” and “bed and breakfast”, the attribute “identifier” of the class “bed_and_breakfast” in the source “BB” generates the annotation of the same attribute in the source “VENERE”. By executing a second run of the MELIS process, the attribute “identifier” on the class “building” generates the annotation of the same attribute on the classes “campsite” and ”restaurant” of the sources “guidaC” and “touring” (application of the heuristic rule 2).
- Rule 4: because of the new relationship introduced in WordNet, “campsite” is a sister term of “restaurant”. Consequently, the attribute “locality” is annotated in the same way in the sources “guidaC” and ”touring”.
- Rule 5: In the source “VENERE” the class “map” has a foreign key: the attribute “url” that references to the class “hotel”. Because of this relationship joins with hierarchical relationships “hotel”, “campsite”, “bed and breakfast” and “building”, the annotation of attribute “url” of the class “map”, applying Rule 3 and Rule 5, generates the same annotation for “url” in the classes “campsite”, “bed_and_breakfast” and “restaurant” of the other sources.

Notice that heuristic 6 is not exploited in this example. Such rule may be exploited in nested structures as hierarchies, and they may not be applied in flat structures as relational databases.

The results are highly dependent on the annotation manually provided by the user as MELIS input. For this reason, it is not meaningful to give any evaluation in terms of number of new annotations discovered. Concerning the evaluation of the new annotations generated, our experience highlights that all of the new annotated elements have a correct meaning w.r.t. WordNet.

V. Conclusion and future work

In this paper we presented MELIS, an innovative system for incrementally and automatically annotating data sources according to a lexical database (WordNet in our approach), i.e. MELIS exploits the annotation of a subset of sources’ elements to infer annotations for the remaining source elements. Moreover, MELIS provides a component for enhancing WordNet with new terms and relationships and a tool for generating lexical relationships between annotated source elements.

We coupled MELIS with the MOMIS system in order to improve MOMIS methodology for creating an ontology from a set of data sources. The first results, within the WISDOM project, show that MELIS and MOMIS working in conjunction

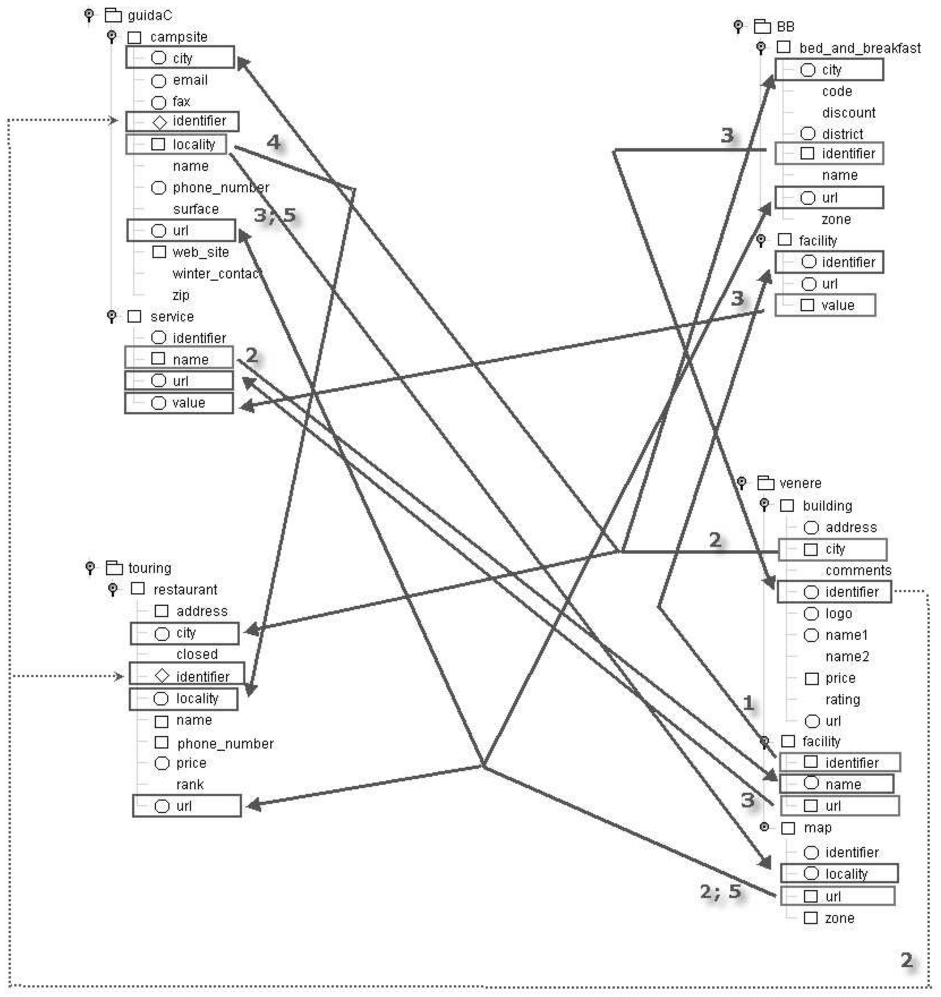


Fig. 5. Annotations generated with MELIS

are an effective and performative tool for creating a domain ontology.

References

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

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

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

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

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

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

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

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

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

[10] S. Handschuh, S. Staab, and R. Volz. On deep annotation. In *WWW*, pages 431–438, 2003.

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

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

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

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