Schema Normalization for Improving Schema Matching

Serena Sorrentino, Sonia Bergamaschi, Maciej Gawinecki, Laura Po

Dipartimento di Ingegneria dell'Informazione
University of Modena and Reggio Emilia
via Vignolese 905, 41125 Modena, Italy

Abstract

Schema matching is the problem of finding relationships among concepts across heterogeneous data sources (heterogeneous in format and in structure). Starting from the "hidden meaning" associated to schema labels (i.e. class/attribute names) it is possible to discover relationships among the elements of different schemata. Lexical annotation (i.e. annotation w.r.t. a thesaurus/lexical resource) helps in associating a "meaning" to schema labels. However, performance of semi-automatic lexical annotation methods on real-world schemata suffers from the abundance of non-dictionary words such as compound nouns, abbreviations and acronyms. We address this problem by proposing a method to perform schema label *normalization* which increases the number of comparable labels. The method semi-automatically expands abbreviations/acronyms and annotates compound nouns, with a minimal manual effort. We empirically prove that our normalization method helps in the identification of similarities among schema elements of different data sources, thus improving schema matching results.

Keywords: schema matching, normalization, natural language for DKE, lexical annotation, interoperability and heterogeneity

1. Introduction

Schema matching is a critical step in many applications such as: data integration, data warehousing, e-business, semantic query processing, peer

Email address: `firstname.lastname@unimore.it` (Serena Sorrentino, Sonia Bergamaschi, Maciej Gawinecki, Laura Po)

data management and semantic web applications [22]. In this work, we focus on schema matching in the context of data integration [3], where the goal is the creation of mappings between heterogeneous data sources (heterogeneous in format and in structure). Mappings are obtained by a schema matching system by using a set of semantic matches (e.g. location = area) between different schemata. A powerful mean to discover matches is the understanding of the "meaning" behind the names denoting schemata elements, i.e. labels in the following [25]. In this context, lexical annotation, i.e. the explicit association of the "meaning"/"sense" to a label w.r.t. a thesaurus (WordNet [16] in our case) is a key tool.

The strength of a thesaurus, like WordNet(WN), is the presence of a wide network of semantic relationships among word meanings, thus providing a corresponding inferred semantic network of lexical relationships among the labels of different schemata. Its weakness, is that it does not cover, with the same detail, different domains of knowledge and that many domain dependent words, as *non-dictionary words*, may not be present in it. Non-dictionary words include Compound Nouns (CNs), abbreviations and acronyms.

The result of automatic lexical annotation techniques is strongly affected by the presence of these non-dictionary words in schemata. For this reason, a method to expand abbreviations and to semantically "interpret" CNs is required. In the following, we will refer to this method as schema label *normalization*. Schema label normalization helps in the identification of similarities between labels coming from different data sources, thus improving schema mapping accuracy.

A manual process of label normalization is laborious, time consuming and itself prone to errors. Starting from our previous works on semi-automatic lexical annotation of structured and semi-structured data sources [4], we propose a semi-automatic method for the normalization of schema labels able to expand abbreviations and acronyms and to annotate CNs w.r.t. WN.

Our method is implemented in the MOMIS (Mediator envirOnment for Multiple Information Sources) system [9, 3]. However, it may be applied in general in the context of schema mapping discovery, ontology merging, data integration systems and web interface integration. Moreover, it might be effective for reverse engineering tasks, when we need to extract an ER schema from a legacy database.

The rest of the paper is organized as follows. In Section 2, we define the problem in the context of schema matching; in Section 3 a brief overview of

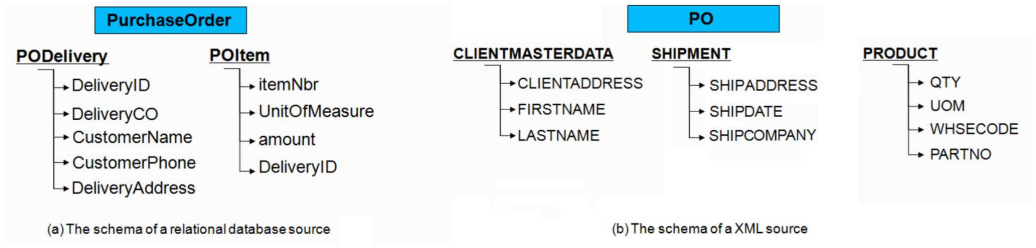

Figure 1: Graph representation of two schemata with elements containing abbreviations and CNs: (a) relational database schema, (b) XML schema.

the method is given; in Sections 4, 5 and 6 we describe subsequent phases of our method: label preprocessing, abbreviation expansion and CN interpretation. Section 7 describes related work; in Section 8 we demonstrate the effectiveness of the method with extensive experiments on real-world data sets; finally, Section 9 is devoted to sketch conclusion and future work.

2. Problem definition

Element names represent an important source for assessing similarity between schema elements. This can be done semantically by comparing their meanings.

Definition 1 *Lexical annotation of a schema label is the explicit assignment of its meaning w.r.t. a thesaurus.*

Starting from the lexical annotation of schema labels we can derive lexical relationships among them on the basis of the semantic relationships defined in WN among their meanings.

Definition 2 *Let S and T be two heterogeneous schemata, and $E_S = \{s_1, ..., s_n\}$ and $E_T = \{t_1, ..., t_k\}$, respectively, the set of labels of S and T. A lexical relationship is defined as the triple $< s_i, t_j, R >$ where $s_i \in E_S$, $t_j \in E_T$ and R specifies a lexical relationship between s_i and t_j. The lexical relationships are:*

- *SYN: (Synonym-of), defined between two labels that are synonymous (it corresponds to a WN synonym relationship);*
- *BT: (Broader Term), defined between two labels where the first is more general then the second (the opposite of BT is NT, Narrower Term) (it corresponds to a WN hypernym/hyponym relationship);*

- *RT: (Related Term) defined between two labels that are related in a meronymy hierarchy (it corresponds to a WN meronym relationship).*

Definition 3 *A compound noun (CN) is a word composed of more than one word called CN constituents. It is used to denote a concept, and can be interpreted by exploiting the meanings of its constituents.*
Definition 4 *An abbreviation/acronym is a shortened form of a word or phrase, that consists of one or more letters taken from the word or phrase.*
In the following we will refer to both abbreviations and acronyms by the term *abbreviations*.

Figure 1 shows two schemata to be integrated, containing many labels with non-dictionary CNs (e.g. "CustomerName"), acronyms (e.g. "PO") and abbreviations (e.g. "QTY"). These labels cannot be directly annotated, because they do not have an entry in WN. Schema label normalization (also called *linguistic normalization* in [22]) is the reduction of the form of each label to some standardized form that can be easily recognized by a schema designer. In our case, with label normalization we intend the process of abbreviation expansion, and CN interpretation.
Definition 5 *The interpretation of a CN is the task of determining the semantic relationships holding among the constituents of a CN.*
Definition 6 *Abbreviation expansion is the task of finding a relevant expansion (long form) for a given abbreviation (short form).*

Schema label normalization improves the schema matching process by reducing the number of discovered *false positive/false negative relationships*.
Definition 7 *Let $< s_i, t_j, R >$ be a lexical relationship. Then it is a false positive relationship, if the concept denoted by the label s_i is not related by R to the concept denoted by the label t_j.*

For example, let us consider the two schema labels "CustomerName" and "CLIENTADDRESS", respectively in the source "PurchaseOrder" and "PO" (Figure 1). If we annotate separately the terms "Customer" and "Name", and "CLIENT" and "ADDRESS", then we will discover a SYN relationship between them, because the terms "Customer" and "CLIENT" share the same WN meaning. In this way, a false positive relationship is discovered because these two CNs represent "semantically distant" schema elements.
Definition 8 *Let $< s_i, t_j, R >$ be a lexical relationship R is a false negative relationship if the concept denoted by the label s_i is related by R to the concept denoted by the label t_j, but the schema matching process does not return this relationship.*

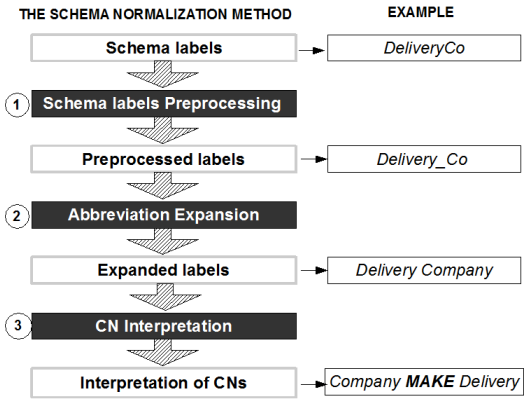

Figure 2: Overview of the schema normalization method showing input and output of each phase together with an example of schema label.

Let us consider two corresponding schema labels: "amount" of the "PurchaseOrder" source and "QTY" (abbreviation for "quantity") of the "PO" source (Figure 1). Without abbreviation expansion we cannot discover that there exists a SYN relationship between the elements "amount" and "QTY".

3. Overview of the schema normalization method

The schema label normalization method consists of three phases: (1) schema label preprocessing, (2) abbreviation expansion and (3) CN interpretation.

To give an intuitive idea of the whole method, Figure 2 shows a simple example of application of the normalization method on the schema element "DeliveryCO" that belongs to the "PurchaseOrder" schema in Figure 1. The method receives as input the schema label "DeliveryCO"; then it performs automatic label preprocessing which returns the label tokenized in two words, "Delivery" and "CO", where the last is recognized as an abbreviation. The abbreviation "CO" is expanded as "Company" by using an automatic algorithm for abbreviation expansion; finally the expanded label "Delivery Company" is recognized as a CN and a semi-automatic method for its interpretation is applied. Moreover, as it will be shown in the following, a new CN for the label "Delivery Company" is inserted in the WN lexical database. The output of the normalization method is the normalized label "Delivery Company" with its interpretation ("Company MAKE Delivery"). In the following sections, we will describe each phase in details.

4. Schema label preprocessing

To perform schema label normalization, schema labels need to be preprocessed. Schema label preprocessing is divided in three main sub-steps: (1) identification, (2) tokenization, (3) classification.

Step 1. Identification
The goal is to identify those schema labels that do not have an entry in WN and thus need to be normalized.

CNs (e.g. "company name") and abbreviations (e.g. "GDP" standing for "Gross Domestic Product") having an entry in WN need no normalization. Moreover, we identify a set of exceptions (*standard schema abbreviations*) that, although they have an entry in WN, in the context of schemata are mostly used as abbreviations (e.g. "id", which in WN is a concept in psychology, in schemata is often use as a short form of "identifier"). All such abbreviations are put into a "user-defined dictionary" and automatically identified for normalization.

Definition 9 *A label has to be normalized, if it occurs on the list of standard schema abbreviations or if it does not have an entry in WN.*

Step 2. Tokenization
This step tokenizes the previously identified labels by using one of the preexisting approaches described in [17]: *simple* (ST) – based on camel case and punctuation, and *greedy* – handling also multi-word labels without clearly defined word boundaries (e.g. "WHSECODE"). The latter uses simple tokenization to split the label around explicit word boundaries into single words and then for each non-dictionary word iteratively looks for the biggest prefixing/suffixing dictionary word or standard schema abbreviation. We consider two alternative variants of greedy tokenization: GT/WN that makes use of WN to identify dictionary words, and GT/Ispell that makes use of Ispell English words list[1].

Step 3. Classification
This step classifies tokenized labels into three groups: dictionary words that exist in WN, abbreviations that need expansion and CNs that need inter-

[1]Ispell is a popular tool for spelling errors correction: url-http://wordlist.sourceforge.net/

pretation. The same heuristic rules as during identification step are applied here.

For instance, let us assume we are preprocessing "DeliveryCO" label (shown in Figure 1). This label is neither a dictionary word nor a standard schema abbreviation, thus it is given as input to the method and processed as in Figure 2.

5. Automatic abbreviation expansion

Automatic abbreviation expansion of words classified as abbreviations requires the execution of the following steps: (1) searching for potential long forms for the given short form; and (2) selecting the most appropriate long form from the set of long form candidates.

A schema can contain both *standard* and *ad hoc* abbreviations. Standard abbreviations either denote important and repeating domain concepts (*domain standard abbreviations*), e.g. "Co" (Company), or are commonly used by schema designers but do not belong to any specific domain (*standard schema abbreviations*), e.g. "Nbr" (Number). For instance, the OTA standard[2] contains a list of recommended standard schema abbreviations. On the contrary, ad hoc abbreviations are mainly created by a schema designer to save space, starting from phrases that would not be abbreviated in a normal context [34, 19].

5.1. Expansion resources

To observe how different types of abbreviations can be handled automatically, we analyzed short forms and their corresponding long forms in several open-source schemata (see Table 1 for a list of them). Based on our manual inspection, we found four *expansion resources* relevant for finding possible long form candidates: (1) *local context* (LC), (2) *complementary schemata* (CS), (3) *online abbreviation dictionary* (OD), and (4) *user-defined dictionary* (UD). To define the *local context* and the complementary schemata resources, let us suppose sf to be a short form identified in a schema label l. The label l is either an attribute name of a class c or a class name belonging to a schema s. Then, the local context of sf is the class c or the schema

[2]OpenTravel Alliance Xml schema for travel industry. Available online at `http://www.opentravel.org/`.

Schema from	Type	URL
Freeway	relational	http://freeway.sourceforge.net
Zen Cart E-Commerce	relational	http://zencart.sourceforge.net
phpMyAdmin	relational	http://phpmyadmin.net
MediaWiki	relational	http://mediawiki.org
eCanteen	relational	http://ecanteen.sourceforge.net
Freeside	relational	http://freshmeat.net/projects/freeside
ImpressCMS	relational	http://impresscms.sourceforge.net
Open Travel Alliance	XSD	http://opentravel.org
Geography Markup Language	XSD	http://www.opengeospatial.org/standards/gml
XML Common Business Language	XSD	http://xcbl.org
XWebTD Web service	XSD	http://ws.xwebservices.com/XWebTD/V1/Order_Types.xsd
Extended camera ontology	OWL	http://hnsp.inf-bb.uni-jena.de/opossum/

Table 1: Schemata analyzed for manual abbreviation expansion.

s. The complementary schemata are the other schema that have to be integrated with the schema s. Local context and complementary schemata are particularly relevant for expanding ad hoc abbreviations. It is common practice to abbreviate class name in its attribute name (for instance, "SHIPMENT" table in Figure 1 contains "SHIPADDRESS", "SHIPDATE" and "SHIPCOMPANY" attributes, where "SHIP" is an abbreviation for "SHIPMENT"). For the complementary schemata we observed, for instance, that the short form "UOM" in the XML schema (Figure 1b) can be expanded with long form "Unit Of Measure" from the relational database schema (Figure 1a). An online abbreviation dictionary (in our case Abbreviations.com) is particularly useful for expanding domain standard abbreviations.

Finally, the user-defined dictionary is created by extracting standard schema abbreviations from the mentioned OTA standard. Moreover, the designer can enrich this user-defined dictionary by inserting new standard abbreviations.

5.2. Abbreviation expansion algorithm

To handle different types of abbreviations the algorithm uses the four aforementioned resources of long forms. However, the syntax of a short form itself does not provide any mean for distinguishing between ad hoc and standard abbreviations and thus we are not able to choose in advance the

Pattern	**Regular expression**	**Short form**	**Long form**
Acronym	c_0`[a-z]+`c_1`[a-z]+...[a-z]+`c_n	*mfag*	***m****edical* ***f****irst* ***a****id* ***g****uide*
Prefix	sf`[a-z]+`	*dep*	***dep****artment*
Dropped Letter	c_0`[a-z]*`c_1`[a-z]*...[a-z]*`c_n	*dept*	***dep****artment*
Combination Word	c_0`[a-z]*?`c_1`[a-z]*?...[a-z]*?`c_n	*pdef*	***p****eriod* ***def****ined* *first*

Table 2: List of abbreviation patterns given in the order in which they are evaluated for a short form [15]. A pattern is a regular expression created from the characters of a short form: $sf = c_o c_1 \dots c_n$.

relevant resource to expand a given short form. Nevertheless, we can consider the local context and complementary schemata as the most relevant resources in general, because they closely reflect the intention of a schema designer.

For each abbreviation the algorithm (1) queries all four resources for long form candidates, (2) scores the relevance of the long form candidates (potential expansions), (3) combines the scores and chooses the top-scored one. In the following, we describe in details each of these steps.

Step 1. Obtaining long forms candidates from expansion resources We look for possible long form candidates in the local context and in the complementary schemata using the four abbreviation patterns proposed in [15] and listed in Table 2. These abbreviations patterns cover all possible patterns between a short form and a long form candidate. Only the first matching candidate is considered. Moreover, the algorithm tries to find an entry for the target sf into the online and user-defined dictionaries; it returns all the long forms returned by these two expansion resources.

Let us focus on the expansion of the "CO" abbreviation contained in "DeliveryCO" label. The local context of "DeliveryCO", in this case, is its schema, while the schema "PO" is the complementary schema. The abbreviation expansion algorithm receives the following expansions: (a) from online dictionary {"Company", "Colorado", and "Check Out"} (b) from the local context no results, (c) from the complementary schemata {"Company"}. Next, the algorithm merges lists of long form candidates into a single one: {"Company", "Colorado", "Check Out"}.

Step 2. Scoring expansions
For the user-defined dictionary, the local context and the complementary schemata the score of lf_i is 1, if lf_i is found in the given resource, or 0, otherwise.

The online dictionary may suggest more then one long form for a given short form. For this purpose we propose a disambiguation technique based on two factors: (a) the number of domains a given long form shares with the schemata to be integrated and (b) its popularity in these domains. The intuition is that the relevant expansions are those that are the most popular in the domains described by the schemata to be integrated. We found that information about the domain of a long form and its popularity can be found in online dictionaries like Abbreviations.com.

The entries in the online dictionary (OD) for a short form sf can be modeled as a combination of a long form lf_i and a domain d in which it appears with the associated popularity $p(< lf_i; d >)$. To compute a score for each online dictionary expansion we need to identify the main domains of the schemata to be integrated. We use WordNet Domains (WN Domains)[3] which is an extension of WN that assigns to each WN synset one or more domains. Using the algorithm proposed in [4], we compute the prevalent domains for the schemata. The algorithm examines all possible WN synsets connected to all labels of schemata and extract all domains associated to those synsets. Next, it returns the top m prevalent domains[4].

We define the score of a long form candidate, $sc_{OD}(lf_i)$, as follows:

$$sc_{OD}(lf_i) = \sum_{d \in D(lf_i) \cap D(schemata)} \frac{p(< lf_i; d >)}{P_{schemata}}$$

where $D(schemata)$ is the list of prevalent domains (of WN Domains) associated with the schemata to integrate; $D(lf_i)$ is the list of domains associated by the of online dictionary to the long form lf_i. $P_{schemata}$ is used as a normalizing constant, and is defined as the sum of the popularity of all the long form candidates for the short form sf:

$$P_{schemata} = \sum_{j} \sum_{d \in D(lf_i) \cap D(schemata)} p(< lf_j; d >)$$

[3] `http://wndomains.itc.it/`
[4] We use $m := 3$.

The domain taxonomy used by the Abbreviation.com is different than the one of WN Domains. To translate the dictionary domains of a long form into WN Domains we have manually defined mappings between the two taxonomies.

For example, "Commerce", "Sociology" and "Metrology" are the prevalent domains for the schemata in Figure 1. For the abbreviation "CO" the online dictionary returns the following expansions: "Company" (Business), "Colorado" (Regional), "Check Out" (Medical). However, among these three entries only the first one is relevant, because its category is mapped to "Commerce" domain of WN Domains (one of the schemata domains); we obtain $sc_{OD}(Company) = 1$.

Step 3. Combining expansion resources
During this step, for each previously identified long form lf_i the algorithm computes a combined score $sc(lf_i) \in [0, 1]$. Then, the algorithm selects the top-scored long form candidate. If the list of long form candidates is empty, the original short form is preserved. The score $sc(lf_i)$ is computed by combining scores from the single resources:

$$sc(lf_i) = \alpha_{UD} \cdot sc_{UD}(lf_i) + \alpha_{CS} \cdot sc_{CS}(lf_i) + \alpha_{LC} \cdot sc_{LC}(lf_i) + \alpha_{OD} \cdot sc_{OD}(lf_i)$$

where $\alpha_{UD} + \alpha_{CS} + \alpha_{LC} + \alpha_{OD} = 1$ are weights of resources relevance. These weights can be configured by the designer. However, we select as default weights $\alpha_{UD} = 0.4$, $\alpha_{LC} = 0.3$, $\alpha_{CS} = 0.2$ and $\alpha_{OD} = 0.1$. We choose these default weights as they reflect the relevance of the resources during the abbreviation expansion process.

For example, for the long form candidate "Company" of the abbreviation "CO" the score becomes:

$$sc(Company) = 0.3 * 1 + 0.2 * 1 = 0.5$$

6. Compound noun interpretation

In order to perform semi-automatic CN annotation, a method for their interpretation needs to be devised. In the natural language disambiguation literature different CN classifications have been proposed [33, 29]. In this work we use the classification introduced in [33], where CNs are classified in four distinct categories: *endocentric*, *exocentric*, *copulative* and *appositional*; we consider only endocentric CNs.

Definition 10 *An Endocentric CN consists of a head (i.e. the categorical part that contains the basic meaning of the whole CN) and modifiers, which restrict this meaning. A CN exhibits a modifier-head structure with a sequence of nouns composed of a* head *noun and one or more modifiers where the head noun occurs always after the modifiers.*

The constituents of endocentric compounds are noun-noun or adjective-noun, where the adjective derives from a noun (e.g. "dark room", where the adjective "dark" derives from the noun "darkness"). Our restriction on endocentric CNs is motivated by the following observations: (1) the vast majority of CNs of schemata fall in endocentric category; (2) endocentric CNs are the most common type of CNs in English; (3) exocentric and copulative CNs, which are represented by a unique word, are often present in a dictionary; (4) appositional CNs are not very common in English and less likely used as elements of a schema. Endocentric CNs are usually not included in a dictionary, but can be interpreted by using the knowledge of the constituents as well as knowledge about constituents combination. For this property, an endocentric CNs can be also defined as *transparent* [2]. We consider endocentric CNs composed of only two constituents, because CNs consisting of more than two words need to be constructed recursively by *bracketing* them into pairs of words and then interpreting each pair. In the following, we will refer to endocentric CNs simply as CNs.

Our method can be summed up into four main steps: (1) CN constituent disambiguation; (2) redundant constituent identification; (3) CN interpretation via semantic relationships; (4) creation of a new WN meaning for a CN.

Step 1. CN constituent disambiguation
In this step the correct WN synset of each constituent is chosen in two steps:

1. *Compound Noun part of speech tagging*: this step performs the part of speech analysis of CN constituents, in order to identify the syntactic category of its head and modifier. To do that we use the Stanford part of speech tagger [39] [5]. If the CN does not fall under the endocentric syntactic structure (noun-noun or adjective-noun where the adjective derives from a noun), then it is ignored. For example the constituents

[5]The Stanford part of speech tagger is freely available at `http://nlp.stanford.edu/software/tagger.shtml#Download`

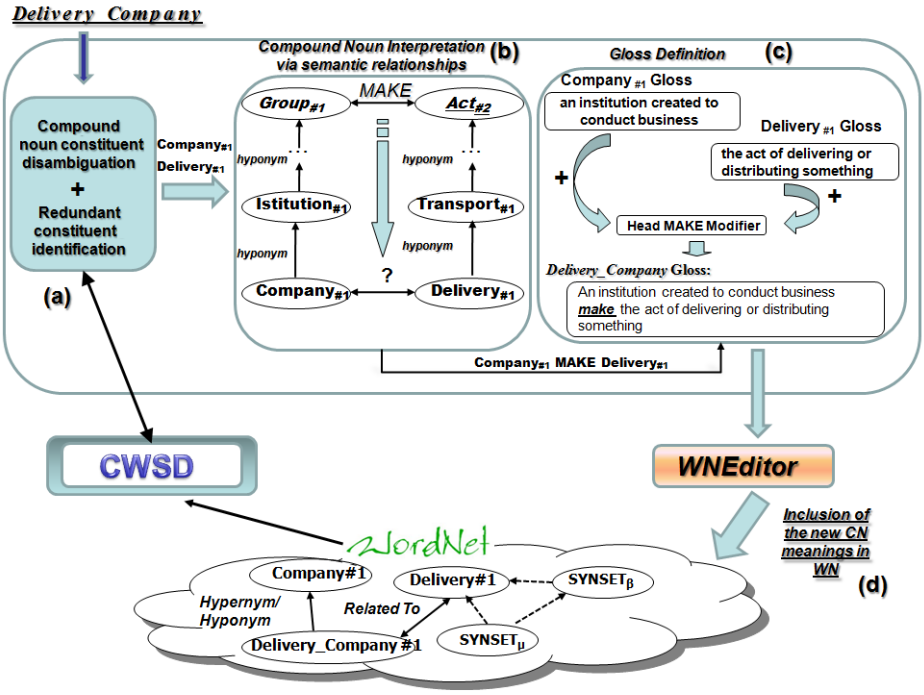

Figure 3: The CN interpretation process.

of the CN "Delivery Company" belong both to the noun syntactic category;

2. *Disambiguating head and modifier*: this step is part of the general lexical disambiguation problem. By applying our CWSD (Combined Word Sense Disambiguation) algorithm [4], each word is automatically mapped onto its corresponding WN 2.0 synsets.

CWSD is an algorithm and a tool for the automatic annotation of structured and semi-structured data sources. Instead of being targeted to textual data sources like most of the traditional WSD (Word Sense Disambiguation) algorithms, CWSD exploit the structure of data sources together with the lexical knowledge associated with schema elements to discover the right meaning to be associated to each word. CWSD is composed of two algorithms: SD (Structural Disambiguation) and WND (WordNet Domains Disambiguation). SD tries to disambiguate source terms by using semantic relationships inferred from the structure of data sources (intra-schema relationships) and WND tries to disambiguate the terms using domains information supplied by

WN Domains. Thanks to CWSD each word is semi-automatically mapped into its corresponding WN 2.0 synsets.

We agree with [26] that WSD can significantly improve the accuracy of CN interpretation.

For example, as shown in Figure 3a, for the schema elements "DeliveryCO", previously expanded as "Delivery Company", we obtain the two constituents annotated with the correspondent WN meanings (i.e. "$Company_{\#1}$" and "$Delivery_{\#1}$").

Step 2. Redundant constituent identification and pruning
During this step we control whether a CN constituent is a *redundant word*.
Definition 11. *A redundant word is a word that do not contribute new information as its semantics contribution can be derived from the schema or from the lexical resource.*

The typical situation in a schema is when the name of a class is a part of its attribute name, see for instance the "SHIPADDRESS" attribute of the "SHIPMENT" class (Figure 1). The "SHIPADDRESS" attribute is expanded in the abbreviation expansion phase as "SHIPMENT ADDRESS". As a result, the constituent class name is not considered, because the relationship holding among a class and its attributes can be derived from the schema. Moreover, a redundant word exists when one of the constituents is an hypernym/hyponym of the other, e.g. the CN "mammal animal" where the meaning associated by CWSD to the head "animal" is an hyponym of the meaning associated to the modifier "mammal". The information that "a mammal is a kind of animal" is redundant because can be directly derived from the WN hierarchy.

Step 3. CN interpretation via semantic relationships
This step concerns selecting from a set of predefined relationships the one that in the best way captures the semantic relation between the meanings of a head and a modifier. The problem of devising a set of semantic relationships to be considered for the CN interpretation has been widely discussed in the natural language disambiguation literature [20]. In [29] Levi defines a set of nine possible semantic relationships to interpret CNs: CAUSE ("flu virus"), HAVE ("college town"), MAKE ("honey bee"), USE ("water wheel"), BE ("chocolate bar"), IN ("mountain lodge"), FOR ("headache pills"), FROM ("bacon grease") and ABOUT ("adventure story"). On the contrary, Finin in [24] claims an unlimited number of semantic relationships. In [33] the

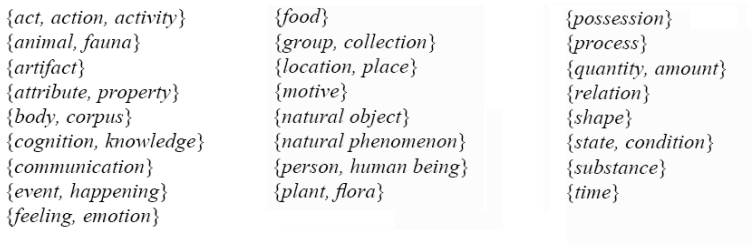

Figure 4: The 25 unique beginners for WN nouns.

problems of relationships set is sidestep: the semantics of a CN is then simply the assertion of an unspecified relation between its constituents. Other set of semantic relationships to interpret CNs are proposed in [26, 32, 35]. We choose the Levi semantic relationships set, as it is the best choice in the simplified context of a data integration scenario. According to [23], our method is based on the following assumption:

Definition 12 *The semantic relationship between a head and its modifier of a CN is derived from the one holding between their top level WN nouns in the WN nouns hierarchy.*

Top levels of a lexical resource include concepts that make important ontological distinctions, and although they contain relatively few concepts, these concepts are important for the task of CN interpretation and whole they cover all different conceptual and lexical domains present in the lexical resource. In particular, the WN nouns hierarchy has been proven to be very useful in the CN interpretation task [20]. The top level concepts of the WN hierarchy are the 25 *unique beginners* (e.g. act, animal, artifact etc.) for WN English nouns defined by Miller in [16] (see Figure 4). These unique beginners were selected after considering all the possible adjective-noun or noun-noun combinations that could be expected to occur and are suitable to interpret noun-noun or adjective-noun CNs as in our case.

For each possible couple of the unique beginners we manually associate the relationship from the Levi's set that best describes the meaning of this couple. For example, for the unique beginner pair "group and act" we choose the Levi's relationship MAKE (e.g. "group MAKE act"), that can be expressed as: a group performs an act. In this way, as shown in Figure 3b, we are able to interpret the label "Delivery Company" with the MAKE relationship, because "Company" is a hyponym of "group" and "Delivery" is a hyponym of "act".

Our method requires an initial human intervention to associate the right relationship to each pair of unique beginners. However, it may be considered acceptable, when compared with the much greater effort required for other approaches based on pre-tagged corpus where the number of CNs to be annotated is much higher [20, 27]. Moreover, the method is independent from the domain under consideration and can be applied to any thesaurus providing a wide network of hyponym/hypernym relationships between defined meanings.

Step 4. Creation of a new WN meaning for a CN
During this step, we automatically create a new WN meaning for a CN starting from the meanings of its constituents and using the discovered relationship. We distinguish the following two steps:

1. *Gloss definition*: a WN *gloss* is the definition and explanation in natural language of a meaning for a term. Starting from the relationship associated to a CN and exploiting the glosses of the CN constituents, we create the gloss to be associated to a CN. To create a new gloss for the CN, we have the need to express by natural language the meanings of a semantic relationship. As described previously, we chose to interpret CNs by Levi's relationships, which can be used directly in the gloss definition. Figure 3c shows an example of this step. The glosses of the constituents "Company" and "Delivery", are joined according to the Levi's relationship MAKE, as consequence, the new gloss for the CN "Delivery Company" will be "An institution created to conduct business MAKE the act of delivering or distributing something"
2. *Inclusion of a new CN meaning in WN*: the insertion of a new CN meaning into the WN hierarchy implies the definition of its relationships with the other WN meanings. As the concept denoted by a CN is a subset of the concept denoted by the head, we assume that a CN inherits most of its semantics from its head [33]. Starting from this consideration, we can infer that the CN is related, in the WN hierarchy, to its head by an hyponym relationship. Moreover, we represent the CN semantics related to its modifier by inserting a generic relationship RT (*Related term*), corresponding to WN relationships as *member meronym*, *part meronym* etc. During this step, we automatically control also if other new CNs with the same head have been previously inserted in WN. For example, if we have to insert in WN a new mean-

ing for the CN "student name" and we have previously inserted the CN "person name", we control if there exists a *hyponym/hypernym* relationship between the modifiers "person" and "student". In this case, we insert the new meaning for the CN "student name" as a hyponym of the already inserted CN "person name". However, the insertion of these two relationships is not sufficient; it is necessary to discover also the relationships of the new inserted meaning w.r.t. the other WN meanings. For this purpose, we use the WNEditor tool to create/manage the new meaning and to set relationships between it and the WN ones [9]. The WNEditor automatically retrieves a list of candidate WN meanings sharing similarities with the new meaning. Then, the user is asked to explicitly declare the type of relationship (hyponymy, meronymy etc.) to relate the new meaning to another, if any. Figure 3d shows an example of this step.

7. Related work

Works related to the issues discussed in this paper are in the area of linguistic normalization, normalization techniques in schema matching and finally the use of WN in schema matching.

7.1. Linguistic Normalization

The problem of linguistic normalization has received much attention in different areas such as: machine translation, information extraction, information retrieval.

In many works on NLP (Natural Language Processing) it is often assumed that abbreviations are words built up of a specific syntax, for instance Taghva and Gilbreth [38] propose to consider as acronyms only upper-case words of three to ten characters. However, this makes sense only for the labels with mixed case (e.g. "buildingVAT", but not for "BUILDING_VAT"). Moreover, when dealing with abbreviations in texts many approaches consider the problem of abbreviation recognition as a problem of finding a pair of abbreviation and expansion, based on the observation that in documents the expansion and introduced abbreviation often occur together in explicit position patterns e.g. *"long form (short form)"* [41, 8]. However, this is difficult to achieve in the context of structured or semi-structured data sources, where schema labels do not manifest such clear patterns.

In [21] a mechanism of Integrated Scoring for Spelling error correction, Abbreviation expansion and Case restoration (ISSAC) of textual sources is presented. It uses several sources of expansion including online abbreviation dictionaries, generic and domain specific corpora. Each candidate expansion is ranked on the basis of a combination of weights and the top-scored expansion is selected. However, unlike our approach, ISSAC uses different weights such as Edit Distance or the "general significance" weight (based on the frequency of the candidate expansion in a general collection). Moreover, it does not give different relevance to expansions coming from different sources.

Many works in the literature for interpreting CNs involve costly pre-tagged corpus and heavy manual intervention [20, 27]. These approaches are based on a statistic co-occurrence of a relationship between two words on corpus that contain different CNs manually labeled with the right semantic relationship. Moreover, there are other two main problems with corpus-based methods: (1) in these approaches, there has been some underlying assumption in terms of domain or range of interpretations; this leads to problems in scalability and portability to novel domains; (2) there is a trade-off between how much training data (pre-tagged corpus) are used and the performance of the method. According to [23], we claim that the cost of acquiring knowledge from manually tagged corpus for different domains may overshadow the benefit of interpreting the CNs.

7.2. Normalization techniques in schema matching systems

As observed, the presence of non-dictionary words in schema elements labels (including CNs and abbreviations) may affect the quality of *schema elements matching* and requires additional techniques to deal with [10].

Surprisingly, current schema integration systems either do not consider the problem of abbreviation expansion at all or solve it in a non-scalable way by including of a user-defined abbreviation dictionary or by using only simple string comparison techniques.

Both the well known CUPID [30] and COMA [1] schema matching systems rely on the availability of a complete user-dictionary or a tool for abbreviation expansion. Dealing with abbreviations using an abbreviations dictionary suffers from the lack of scalability. This comes from the fact that: (a) the vocabulary evolves over the time and it is necessary to maintain the table of abbreviations and (b) the same abbreviations can have different expansions depending on the domain, thus an intervention of a schema/domain expert is still required.

The Similarity Flooding [31] approach utilizes a hybrid matching algorithm; it starts from an *initial mapping* which is obtained using a simple string matcher that compares common prefixes and suffixes of literals, thus, it is able to detect some matches among elements labeled with simple abbreviations and the corresponding long forms (e.g. Dept with Department).

In [7] a method for analyzing and revising data integration schemata to improve their matchability is proposed. They propose an approach to automatically identify potential matching mistakes and to suggest to the designer a set of revision for the schemata. To do that, they use a set of predefined domain-independent and domain-dependent rules. In particular, they recognize that some common matching mistakes deriving from the presence of abbreviations and acronyms in the schema labels. However, to expand abbreviations (e.g. "gName" in "gene Name") domain-depended transformation rules have to be manually provided by the designer.

From the best of our knowledge, the only paper that examines the problem of abbreviation expansion in schema matching in a comprehensive way is [34]. To discover expansions, this approach makes use of the Brown corpus, a corpus containing million of words from American English texts printed in 1961. Two techniques have been used for the finding candidate expansions in the corpus: (a) hard-coded heuristics with predefined patterns and (b) machine learning for discovering new patterns (i.e. patterns between abbreviations in the schema labels and candidates in the corpus). There are several limitations in the proposed approach: first, the machine-learned patterns algorithm requires a training set; second, the approach returns a plain list of a large number of expansion candidates, without ranking them; finally, this approach considers only ad hoc abbreviations and not standard domain abbreviations.

Similarly to the abbreviation expansion problem, only a few papers address the problem of CN interpretation.

In [37] a preliminary CNs comparison for ontology mapping is proposed. This approach suffers from two main problems: first, it starts from the assumption that the ontology entities are accompanied with comments that contain words expressing the relationship between the constituents of a CN; second, it is based on a set of rules manually created.

Xu and Embley proposed a parallel composition approach to discover correspondences between graph-like structures (e.g., XML schemas, classifications) [14, 13]. They performed some automatic linguistic normalization, such as stemming and removing stop words, while abbreviation expansion

and CN interpretation are manually executed.

Other schema and ontology matching tools employing *syntactical matching* techniques do not interpret nor normalize CNs but they treat words in CNs separately [28]. This oversimplification leads to the discovery of false positive relationships, so it worsens the matching results.

The S-Match algorithm [25, 36] implements the idea of semantic matching by analyzing the meaning which is codified in the entities and the structures of ontologies. In the preprocessing phase, they compute the meaning of a label at a node (in isolation) by analyzing its real-world semantics. However, only CNs contained in WN are treated as CNs, while non-dictionary CNs are split in single words and treated separately.

7.3. The use of WN in schema matching systems

Semantic taxonomies and thesauri such as WN [16] are a key source of knowledge for natural language processing applications, and provide structured information about semantic relations between concepts.

This is the reason why WN is used in several methods dealing with the linguistic aspect of information integration.

Potentially, all matchers that exploit WN or other thesaurus to discover semantic relationships can integrate the techniques of abbreviation expansion and interpretation of CNs we propose, and thus refine the relationships involving non-dictionary words; some of these tools are: Ctx-Match [5], H-Match [6] and S-Match [25].

8. Experimental evaluation

Our evaluation goals were: (1) measuring and explaining the performance of our method, (2) checking whether our method improves the *lexical annotation* process and finally (3) estimating the effect of schema label normalization on the *lexical relationships discovery* process. To address those goals we conducted detailed experiments. The method was integrated within the MOMIS system [3]. Schema label normalization is performed during the lexical annotation phase of MOMIS: during this phase, each schema element of a local source is semi-automatically annotated by the CWSD algorithm.

8.1. Experimental setup

We tested the effectiveness of our method in several real integration scenarios.

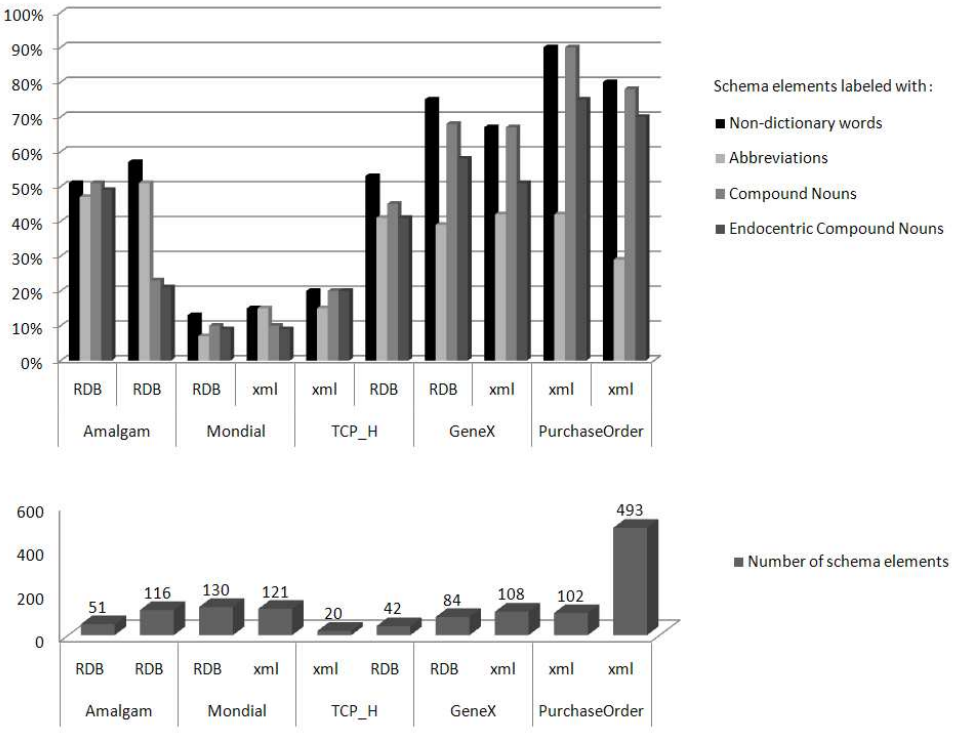

Figure 5: Feature summary of the data sets.

Data Sets. To evaluate our method, we used the following five data sets: (1) GeneX, (2) Mondial, (3) Amalgam (from integration benchmark for bibliographic data [18]), (4) TCP-H and (5) PurchaseOrder (composed by Paragon schema and OpenTrans e-business standard schema). Each data set consists of two schemata that need to be integrated. These data sets[6] have been used in several schema matching experiments [1, 40]. Figure 5 summarizes the features of the schemata. We choose these data sets for the following reasons: they are particularly suitable to evaluate schema normalization as they contain several non-dictionary words; they represent different application domains; finally, they contain both relational (RDB) and XML schemata (XML in different formats: XML schema, DTD, XDR).

[6]All the data sets are publicly available at `http://queens.db.toronto.edu/project/clio/index.php\#testschemas` and `http://dbs.uni-leipzig.de/Research/coma_index.html`

Experimental methodology. To assess the quality of our method gold standards were created for each normalization phase and also for the lexical annotation and the lexical relationships discovery process. The gold standards were manually generated by an human expert. The results obtained in each experimentation were compared w.r.t. the corresponding gold standard.

External resources. The experiments were carried out by using as external sources the lexical database WN 2.0, its extension WN Domains 3.2 and the Abbreviations.com online abbreviation dictionary.

Experimental Measures. To evaluate the performance of our method we used the quality measures defined in [11]. We compared the gold standards with the automatically results obtained by using our method. For each experimental phase we determined: the true positives, i.e. correct results (TP), as well as the false positives (FP) and the false negatives (FN). Based on the cardinalities of these sets, the following quality measures were computed:

- *Precision*= $\frac{|TP|}{|TP|+|FP|}$
- *Recall*= $\frac{|TP|}{|FN|+|TP|}$
- *F-Measure*= $2 * \frac{Precision*Recall}{Precision+Recall}$

Precision and Recall originate from the information retrieval area but have been commonly used also in schema matching and natural language processing studies.

8.2. Evaluating normalization

The normalization method consists of different phases. Since the errors of each phase can be cumulated in the next phases, we evaluated the performance of each phase separately and then as a whole.

8.2.1. Schema label preprocessing evaluation

In order to perform a complete evaluation of this phase, we evaluated tokenization separately and then identification and classification together as they are based on the same heuristics (see Section 4).

Evaluating tokenization. We evaluated three tokenization methods: (1) *ST*—simple, (2) *GT/WN*—greedy with WN and (3) *GT/Ispell*—greedy with Ispell English words list as a dictionary (see Section 4). We evaluated tokenization only for labels identified for normalization in the gold standard.

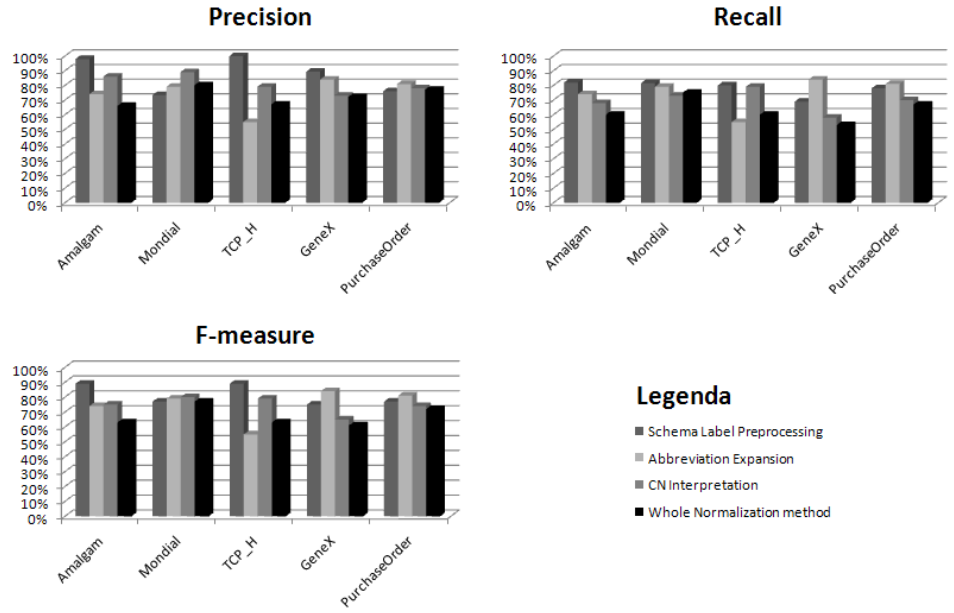

Figure 6: Performance of schema normalization for the different phases (schema label preprocessing (with GT/Ispell tokenization), abbreviation expansion, CN interpretation) and for the whole normalization method.

The GT/WN shows the worst F-Measure (on average 64%), because WN contains many short abbreviations (e.g. "auth" is tokenized to: "au", "th"). For the remaining two tokenization methods the F-Measure was affected by the nature of the schema labels, but the GT/Ispell (F-Measure 86%) was on average 7% more accurate and more stable (6% of standard deviation in F-Measure in contrast to 26% for the ST).

Evaluating identification and classification. In the case that a label is not identified for normalization the whole normalization method might return incorrect expansions or unnecessary interpreted CNs. To estimate how relevant is the problem of identification and classification we experimented those steps separately. Identification did not work well only on average 4% of labels to be normalized (96% of recall). The errors were caused by false negative identifications: labels with abbreviations which are dictionary words in WN. Amalgam and TCP-H schemata contain such difficult abbreviations, e.g. "RID" standing for "Record Identifier" is also a synonym of the verb "free" in WN. The same reason caused more serious drop in recall (on aver-

age 78%) for classification of manually tokenized labels, especially for GeneX (54%). Finally, a number of errors was caused by the presence of stop words (e.g. "to") in schema labels that have not an entry in WN.

8.3. Evaluating abbreviation expansion

We evaluated automatic abbreviation expansion starting from the manually preprocessed labels (gold standard). We used the default relevance weights for expansion resources described in Section 5.2 ($\alpha_{UD} = 0.4$, $\alpha_{LC} = 0.3$, $\alpha_{CS} = 0.2$, $\alpha_{OD} = 0.1$). During the evaluation, an expanded abbreviation was considered as a TP (i.e. *correctly expanded*) if the automatic expansion was the same as the one returned by the gold standard; if not it was considered an FP expansion. FN expansions were all the expansions rendered by the gold standard but not returned by the algorithm. The results of the algorithm are presented in Figure 6. The algorithm provided correct expansions on average for the 74% of abbreviations. Evaluating the output of the algorithm, we found that the 99% of errors were caused by the lack of correct expansions from expansion resources, while only the 1% of errors were caused by incorrect selection (among the long form candidates). This indicates that the default relevance weights lead to the selection of a relevant expansion in most cases.

Since the most errors were caused by the deficiencies in the quality of expansion resources, we investigated on the contribution of each expansion resource to the final performance of the algorithm. To estimate, we evaluated F-Measure for the single expansion resources and for only internal resources (LC plus CS) and for only external resources (OD plus UD). Results are presented in Table 3. If we treat user-defined dictionary as a baseline for our test, we observe that other resources are less correct in providing expansions, but their combination is a good strategy: it leads to significant improvement over the baseline.

It makes thus sense to focus on the quality of each single expansion resource. TCP-H data sets with an F-Measure of 55% revealed the poor quality of the chosen online abbreviation dictionary. For instance, the short form "mfgr" does not have any expansion in Abbreviations.com online dictionary, but we found the correct expansion "manufacturer group" in the AcronymFinder[7], alternative one. When we used the local context or the

[7] `http://www.acronymfinder.com/`

UD	OD	LC	CS	external	internal	all
0.41	0.13	0.09	0.22	0.46	0.27	0.71

Table 3: F-Measure of the use of particular expansion resources in the abbreviation expansion algorithm (the value shown is the average value over all schemata).

complementary schemata, the algorithm suggested as candidate expansions words that are also abbreviations in the schemata. This requires the improvement of the algorithm for the extraction of long form candidates. However, there are deficiencies on which a user integrating schemata may not have influence. For instance, PurchaseOrder data sets benefits from the complementary schemata resource much less (4%) then all other data sets (on average 22%). The results in Figure 6 also shown that F-Measure of a particular expansion resource ranges widely among schemata, thus the only general strategy to provide relevant expansions for a variety of schemata is to combine a diversity of expansion resources.

8.3.1. Evaluating CN interpretation

In this phase the gold standard is represented by the manual interpretation of all CNs contained in the data sets. During the evaluation, a CN was considered as a TP (i.e. *correctly interpreted*) if the Levi's relationship automatically selected was the same as the one returned by the gold standard, if not it was considered a FP interpretation. FN interpretation were obtained for all the the interpretation contained in the gold standard but not returned by our method.

As shown in Figure 6, the CN interpretation method obtained good results both for precision (on average 81%) and recall (on average 70%) and thus for F-Measure (on average 75%). In all data sets the recall value was affected by the presence in the schemata of non-endocentric CNs such as "ManualPublished", "isMember" or "InProceedings" that our method is not able to interpret. Moreover, GeneX, PurchaseOrder and Mondial data sets contain also schema elements labeled with digits (e.g. "sea 2" or "treatment list sequence 1"). As digits are dictionary words in WN, these CNs were automatically considered as endocentric and interpreted in a wrong way by our method. This wrong interpretations mainly stems from the fact that the problem of the presence of digits in schema labels need to be treated in a different way.

The poorest performance was obtained for the GeneX data set. There

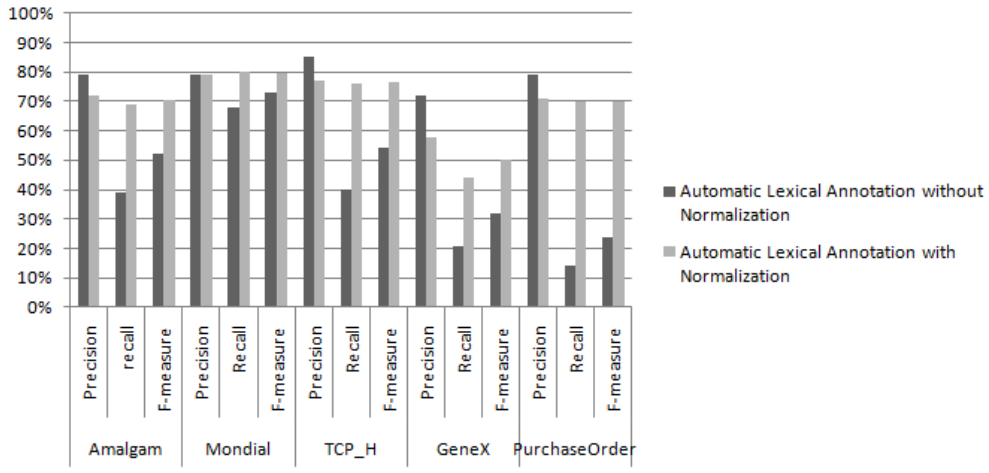

Figure 7: Lexical Annotation Evaluation.

are two main reasons: first GeneX contains several complex CNs composed by three or four constituents (e.g. "GEML" expanded as "gene expression markup language" or "AM_FACTORVALUE" expanded as "array measurement factor value") which are difficult to be interpreted even for a human expert; second, in this source the number of non-endocentric CNs is greater w.r.t. the other data sets (20% of the total number of CNs in GeneX). On the other side, for the PurchaseOrder data set, even if it is characterized by several quite complex CNs, we obtained good results for both precision and recall; in fact, the pruning step (see Section 6 - Step 2) significantly helps in reducing the complexity of CNs (e.g. the attribute label "ORDERCHANGE_ITEM_LIST" of the class "ORDERCHANGE" in the Paragon schema is reduced to the CN "ITEM_LIST").

8.3.2. Evaluating the whole schema normalization method

The input of the whole schema normalization method is the set of the original schema labels and the output the set of normalized schema labels. The method has been evaluated with the GT/Ispell tokenization method that achieved the best results for the considered schemata. Figure 6 shows the result of the whole method. We obtained good results for both precision and recall (the average precision is 63% and the average recall is 72%).

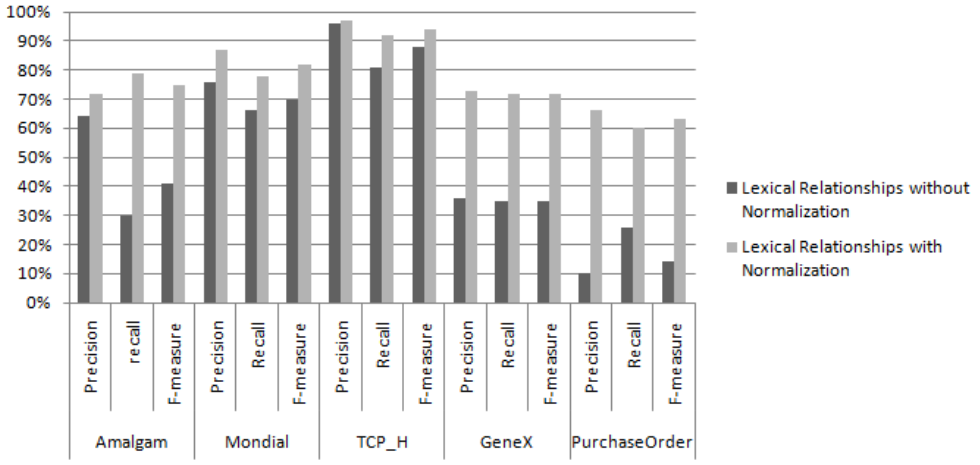

Figure 8: Lexical Relationships Evaluation.

8.4. Evaluating the impact of normalization on lexical annotation

The evaluation of the lexical annotation process was carried out by comparing the annotations returned by CWSD (starting from automatically normalized schemata) w.r.t. the gold standard. The gold standard was created by manually annotating each schema elements w.r.t WN starting from manually normalized schemata. During the evaluation, a schema element annotation was considered as a TP (i.e. *correctly annotated*) if the WN meaning selected by CWSD iwas the same as the one returned by the gold standard; otherwise it was considered an FP annotation. The FN annotations were originated obtained for schema element for not annotated or incorrectly annotated schema elements.

Figure 7 shows the result of lexical annotation performed by CWSD *without and with* our normalization method. Also in this experiment the poorest performance was obtained on the GeneX data set. However, the results shown that for each data sets, by using our normalization method, we are able to improve significantly the F-Measure. In particular, the improvement is more evident when schemata contain several non-dictionary words (e.g. Amalgam and PurchaseOrder data sets). Without schema normalization, for each data sets, CWSD obtains a low recall value, because a lot of CNs and abbreviations are present in the schemata. The application of our method permits to increase the recall while preserving a good precision.

8.5. Evaluating the impact of normalization on lexical relationships

To create the gold standard for the lexical relationships discovery process, we manually mapped the schema elements with the appropriate lexical relationships. During the evaluation, a lexical relationship was considered as a TP (i.e. a *correct lexical relationship*) if it was present in the set of manually determined lexical relationships (gold standard), if not it was considered a FP relationship. FN relationships included all the relationships that were not returned by the automatic lexical relationships discovery process. During this evaluation, we decided to consider only "synonymy" (SYN) and "hypernymy/hyponymy"(BT/NT) relationships and not the "related term" (RT) relationships. This choice is supported by two main observations: RT relationships have a minor relevance w.r.t. BT and SYN relationships; moreover, when the number of schema elements to be mapped become very large the creation of the gold standard with also RT relationships become difficult and error-prone even for a human designer.

Figure 8 shows the result of the lexical relationships discovery process without and with normalization. In the first case, the lexical relationships discovery process was performed without abbreviation expansion and by considering the constituents of a CN as single words with an associated WN meaning. Without schema label normalization we discovered few lexical relationships; the low value of precision was due to the presence of a lot of false positive relationships. Moreover, the recall was very low as a lot of lexical relationships between schema elements labeled with abbreviation were not discovered. Hence, in general, the lexical relationships discovery process without normalization establishes wrong lexical relationships between the schema labels that share some words. Instead, with our method we are able to improve recall and precision (thus also F-Measure) significantly.

Another observation drawn from the graph is that surprisingly, the performances of the lexical relationships discovery process outperform the performances of the lexical annotation process. This can be explained by different reasons: several wrong normalized schema labels (and consequently incorrectly annotated) are not related to any element in the other schema to be integrated (e.g. in the TCP_H data sets the labels "mfgr" and "ph" that are abbreviations, respectively, for "manufacturer group" and "phone" are normalized in a wrong way but they are not connected to any element in the complementary schemata, the same holds for the labels "language" and "update code" in Amalgam). Moreover, there are some lucky cases where, even if the schema elements are normalized and annotated in a wrong way, a correct

lexical relationship is discovered (e.g. in GeneX between the wrong normalized and annotated schema elements "Schema1.array.image_an_params" and "Schema2.ARRAYMEASUREMENT.IMAGE_AN_PARAMS" a correct SYN lexical relationships is discovered). Consequently, some errors in the normalization method did not affect the performance of the lexical relationships discovery process.

9. Conclusion & future work

In this paper we presented a method for the semi-automatic normalization of schema elements labeled with abbreviations and CNs in a data integration environment. However, our method can be applied to several other contexts: ontology merging, data-warehouses and web interface integration. The experimental results have shown the effectiveness of our method, which significantly improves the result of the automatic lexical annotation method, and, as consequence, improves the quality of the discovered inter-schema lexical relationships. Moreover, for larger schemata, the effectiveness of our method become even more evident. We showed that, due to the frequency of non-dictionary words in schemata, a schema matching system cannot ignore CNs and abbreviations without compromising recall.

Future work will be devoted to investigate on two main problems identified during the experimental evaluation: the presence of stop words (e.g. "to", "at", "and" etc.) and digits in schema labels [12]; the problem of false negative non-dictionary words during the identification step (e.g. "RID", "AID"). Moreover, we will investigate on other kind of non-dictionary words: words that are not present in the lexical resource as they belong to specific domains such as Medicine, Biology etc.

10. Vitae

Serena Sorrentino is a Ph.D. student (3th year) of the Doctorate School in "Information and Communication Technologies (ICT) - Computer Engineering and Science" at the Department of Information Engineering, University of Modena and Reggio-Emilia, Italy. Her areas of research are Intelligent Data Integration, Semantic Schema Matching and Natural Language Processing. She is a member of the "DBGROUP", led by professor Sonia Bergamaschi.

Sonia Bergamaschi is full professor of Computer Engineering in the Engineering Faculty, University of Modena and Reggio Emilia, leader of the

"DBGROUP" (www.dbgroup.unimo.it) and dean of the Doctorate School in ICT (www.ict.unimo.it) at the Department of Information Engineering, University of Modena and Reggio and Emilia, Italy. Her research activity covers knowledge representation and management in the context of very large databases facing both theoretical and implementation aspects. She has published several papers and conference contributions. Her researches have been founded by the Italian institutions and by European Community projects. She is a member of the IEEE Computer Society and of the ACM.

Maciej Gawinecki is a Ph.D. student in Computer Engineering and Science at the University of Modena and Reggio-Emilia, Italy. He graduated from the University of Adam Mickiewicz, Pozna?, Poland. He worked as a software engineer in several agent-oriented projects in Systems Research Institute of Polish Academy of Sciences. Currently, he is a member of AgentGroup of prof. Giacomo Cabri. His research interests include: reuse of software components, Web service discovery and semantic annotation of structured and semi-structured data.

Laura Po is a research fellow in Computer Engineering at the University of Modena and Reggio Emilia, Italy. She is a member of the DBGROUP, led by professor Sonia Bergamaschi. Her major research interests focus on analysis and comparison of Word Sense Disambiguation methods, the development of automatic techniques for extracting metadata and annotating data sources and the use of Description Logic in knowledge representation. She received a Ph.D. in Computer Engineering and Science from the University of Modena and Reggio Emilia in 2009. She has been involved in several research projects (National and European), and published several papers and conference contributions.

[1] D. Aumueller, H. H. Do, S. Massmann, and E. Rahm. Schema and ontology matching with COMA++. In *SIGMOD'05*, pages 906–908, New York, NY, USA, 2005. ACM.

[2] K. Barker and S. Szpakowicz. Semi-Automatic Recognition of Noun Modifier Relationships. In *COLING-ACL*, pages 96–102, 1998.

[3] S. Bergamaschi, S. Castano, and M. Vincini. Semantic Integration of Semistructured and Structured Data Sources. *SIGMOD Record*, 28(1):54–59, 1999.

[4] S. Bergamaschi, L. Po, and S. Sorrentino. Automatic annotation for

mapping discovery in data integration systems. In *SEBD 2008*, pages 334–341, 2008.

[5] P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: A new approach and an application. In D. Fensel, K. P. Sycara, and J. Mylopoulos, editors, *International Semantic Web Conference*, volume 2870 of *Lecture Notes in Computer Science*, pages 130–145. Springer, 2003.

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

[7] X. Chai, M. Sayyadian, A. Doan, A. Rosenthal, and L. Seligman. Analyzing and revising data integration schemas to improve their matchability. *PVLDB*, 1(1):773–784, 2008.

[8] J. T. Chang, H. Schtze, and R. B. Altman. Creating an online dictionary of abbreviations from MEDLINE. *Journal of the American Medical Information Association*, 9(6):612–620, 2002.

[9] D.Beneventano, S. Bergamaschi, F. Guerra, and M. Vincini. Synthesizing an Integrated Ontology. *IEEE Internet Computing*, 7(5):42–51, 2003.

[10] H. H. Do. *Schema Matching and Mapping-based Data Integration: Architecture, Approaches and Evaluation*. VDM Verlag, 2007.

[11] H. H. Do, S. Melnik, and E. Rahm. Comparison of Schema Matching Evaluations. In A. B. Chaudhri, M. Jeckle, E. Rahm, and R. Unland, editors, *Web, Web-Services, and Database Systems*, volume 2593 of *Lecture Notes in Computer Science*, pages 221–237. Springer, 2002.

[12] E. C. Dragut, F. Fang, A. P. Sistla, C. T. Yu, and W. Meng. Stop word and related problems in web interface integration. *PVLDB*, 2(1):349–360, 2009.

[13] D. W. Embley, D. Jackman, and L. Xu. Multifaceted Exploitation of Metadata for Attribute Match Discovery in Information Integration. In *Workshop on Information Integration on the Web*, pages 110–117, 2001.

[14] D. W. Embley, L. Xu, and Y. Ding. Automatic direct and indirect schema mapping: Experiences and lessons learned. *SIGMOD Record*, 33(4):14–19, 2004.

[15] E. H. et al. AMAP: automatically mining abbreviation expansions in programs to enhance software maintenance tools. In *MSR '08*, pages 79–88, New York, NY, USA, 2008. ACM.

[16] G. A. M. et al. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.

[17] H. F. et al. An Empirical Comparison of Techniques for Extracting Concept Abbreviations from Identifiers. In *SEA'06*, November 2006.

[18] R. J. M. et al. The Amalgam Schema and Data Integration Test Suite. www.cs.toronto.edu/ miller/amalgam, 2001.

[19] R. U. et al. Extracting Knowledge from Diagnostic Databases. *IEEE Expert: Intelligent Systems and Their Applications*, 8(6):27–38, 1993.

[20] V. N. et al. Learning Noun-Modifier Semantic Relations with Corpus-based and WordNet-based Features. In *AAAI*. AAAI Press, 2006.

[21] W. W. et al. Integrated scoring for spelling error correction, abbreviation expansion and case restoration in dirty text. In *AusDM '06*, pages 83–89, Darlinghurst, Australia, Australia, 2006. Australian Computer Society, Inc.

[22] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[23] J. Fan, K. Barker, and B. W. Porter. The Knowledge Required to Interpret Noun Compounds. In G. Gottlob and T. Walsh, editors, *IJCAI*, pages 1483–1485. Morgan Kaufmann, 2003.

[24] T. W. Finin. The Semantic Interpretation of Nominal Compounds. In *AAAI*, pages 310–312, 1980.

[25] F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: an algorithm and an implementation of semantic matching. In Y. Kalfoglou, W. M.

Schorlemmer, A. P. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, volume 04391 of *Dagstuhl Seminar Proceedings*. IBFI, Schloss Dagstuhl, Germany, 2005.

[26] S. N. Kim and T. Baldwin. Automatic interpretation of noun compounds using wordnet similarity. In R. Dale, K.-F. Wong, J. Su, and O. Y. Kwong, editors, *IJCNLP*, volume 3651 of *Lecture Notes in Computer Science*, pages 945–956. Springer, 2005.

[27] M. Lapata. The Disambiguation of Nominalizations. *Computational Linguistics*, 28(3):357–388, 2002.

[28] B. T. Le, R. Dieng-Kuntz, and F. Gandon. On ontology matching problems - for building a corporate semantic web in a multi-communities organization. In *ICEIS (4)*, pages 236–243, 2004.

[29] J. N. Levi. *The Syntax and Semantics of Complex Nominals*. Academic Press, New York, 1978.

[30] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic Schema Matching with Cupid. In *VLDB*, pages 49–58, 2001.

[31] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. In *ICDE*, pages 117–128, 2002.

[32] D. Moldovan, A. Badulescu, M. Tatu, D. Antohe, and R. Girju. Models for the semantic classification of noun phrases. In *In HLT-NAACL 2004: Workshop on Computational Lexical Semantics*, pages 60–67, 2004.

[33] I. Plag. *Word-Formation in English*. Cambridge Textbooks in Linguistics. Cambridge University Press, New York, 2003.

[34] L. Ratinov and E. Gudes. Abbreviation Expansion in Schema Matching and Web Integration. In *WI '04*, pages 485–489, Washington, DC, USA, 2004. IEEE Computer Society.

[35] B. Rosario and M. Hearst. Classifying the semantic relations in noun compounds. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, 2001.

[36] P. Shvaiko, F. Giunchiglia, and M. Yatskevich. Semantic matching with s-match. *Semantic Web Information Management: a Model-Based Perspective*, XX:183–202, 2010.

[37] X. Su and J. A. Gulla. Semantic Enrichment for Ontology Mapping. In F. Meziane and E. Métais, editors, *NLDB*, volume 3136 of *Lecture Notes in Computer Science*, pages 217–228. Springer, 2004.

[38] K. Taghva and J. Gilbreth. Recognizing acronyms and their definitions. *IJDAR*, 1(4):191–198, 1999.

[39] K. Toutanova and C. D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *In EMNLP/VLC 2000*, pages 63–70, 2000.

[40] J. T.-L. Wang, editor. *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*. ACM, 2008.

[41] S. Yeates, D. Bainbridge, and I. H. Witten. Using Compression to Identify Acronyms in Text. In *DCC'00*, Washington, DC, USA, 2000. IEEE Computer Society.