Semantical Mapping of Attribute Values for Data Integration

Marcin Szymczak*† Antoon Bronselaer† Sławomir Zadrożny* Guy De Tré†
* Systems Research Institute
Polish Academy of Sciences, Warsaw, Poland
† Department of Telecommunications and Information Processing
Ghent University, Ghent, Belgium

***Abstract*—Nowadays the amount of data is increasing very fast. Moreover, useful information is scattered over multiple sources. Therefore, automatic data integration that guarantees high data quality is extremely important. One of the crucial operations in integration of information from independent databases is detection of different representations of the same piece of information (called *coreferent* data) and translation of the representation of data from one source into the representation of the other source. That translation is also known as *object mapping*. In this paper, we investigate automatic mapping methods for attributes the values of which may need semantical comparison and can be sorted by means of an order relation that reflects a notion of generality. These mapping methods are investigated closely in terms of their effectiveness. An experimental evaluation of our method shows that using different mapping methods can enlarge a set of true positive mappings.**

I. Introduction

Problem statement

One of the crucial issues to preserve high data quality in a database is the proper integration of data from different databases. Two major steps are considered in the data integration process. The first step is known as the *schema matching* problem which attempts at reconciling structural heterogeneity of data by mapping schema elements across the data sources [1], [2], [3], [4], [5], [6], [7]. The second step resolves semantic heterogeneity of data by mapping data instances across the datasets and is known as the *object mapping* problem [8], [9], [10], [11], [12], [13], [14], [15], [16].

In this paper, we present a novel approach for a specific part of the object mapping problem, namely we study automatic value mapping methods for attributes whose domains are partially ordered and where the given order relation reflects a notion of generality. We make the following three assumptions. First of all, the schema of each considered dataset contains an attribute which satifies the above condition, i.e. there exist a partial order relation defined on its domain. Next, such an order relation is known in advance. Finally, we assume that the schema matching between input datasets is established.

We consider two datasets as a running example in this contribution. They contain objects which are geographical annotations of a map that pinpoint locations of specific interest and are called points of interest (POIs). Each object is characterized by at least four attributes: name, longitude, latitude and category. The attribute `name` represents the identifier (name) of the specific POI, the `longitude` and `latitude` give the geographic coordinates of the place, and the `category` specifies the type or function of the location. The first dataset is represented by Table I which contains objects extracted from a Google database[1], called the source S, with a known partial order relation on the domain of the `category` attribute. Figure 1 presents a part of this relation. The most general concept is a root of the tree and its descendant nodes correspond to narrower concepts. For instance, the concept *establishment* in Figure 1 is the most general concept for others values of attribute `category` of dataset S and has children corresponding to more specific structures (e.g., *lodging*, etc.). The second dataset contains objects extracted from the RouteYou dataset[2], called the target T, also with a known partial order relation on the domain of the `category` attribute. Table II contains objects extracted from the target dataset, while a part of the order relation is presented in Figure 2. For instance, the concept *POI* in Figure 2 is the most general concept among the values of attribute `category` of dataset T and has children corresponding to more specific concepts (e.g., *Support*, *Accomodation*, *Shopping location*, etc.).

TABLE I. Example of objects extracted from dataset S.

Id	Name	Lon.	Lat.	Category
1	Selfstorage-Achel	5.470673	51.276789	storage
2	Campirama NV	3.251893	50.852829	campground
3	Cafe-Restaurant De Ster	4.050876	51.281777	cafe
4	Het Kouterhof	3.665122	51.034331	lodging
5	Borluut Bed Breakfast	3.657992	51.018882	lodging
6	Carlton Hotel	3.713951	51.036280	lodging
7	Snooz Inn	3.733049	51.058803	lodging

TABLE II. Example of objects extracted from dataset T.

Id	Name	Lon.	Lat.	Category
1	Pakhuis Stokholm	4.665898	51.818359	Warehouse
2	Camping De Iembarg	7.111477	52.967909	Camp Site
3	Cafe Theatre	3.722015	51.049830	Restaurant
4	Het Kouterhof	3.665140	51.034379	Hotel
5	Borluut Bed Breakfast	3.657975	51.018938	Guest room
6	Hotel City Inn	9.369670	52.329782	Hotel
7	Santellone Resort	10.16630	45.550103	Hotel

Let us consider a data integration scenario in which objects from a dataset S have to be merged with objects from a

[1]Google, http://maps.google.com
[2]RouteYou, http://www.routeyou.com/

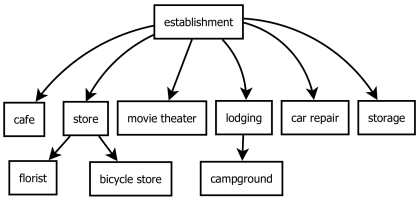

Fig. 1. A part of the partial order relation for the category attribute from the dataset S.

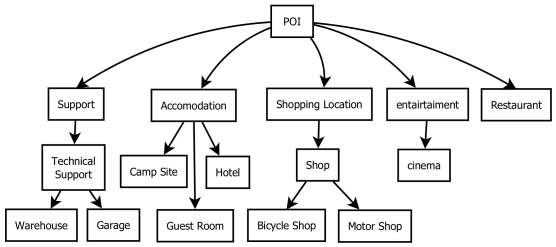

Fig. 2. A part of the partial order relation for the category attribute from the dataset T.

dataset T. Values of attribute as `name`, `longitude` and `latitude` might be stored in the target dataset T without any additional processing assuming that there is no coreference between imported values and values of objects in the dataset T for the corresponding attributes. However, importing of values of attributes such as `category`, representing information on the type of a point of interest, is less trivial as they may often refer to the same concepts presented in a different way in both datasets, called *coreferent* data. For instance, the concept *accomodation* is represented by the category *lodging* in the dataset S and by the category *Accomodation* or *Hotel* in the dataset T. Therefore, for successful data integration, it may be crucial to create mappings of values of the category attribute in the datasets S and T. Such mappings help to maintain consistency and decrease the number of duplicates in the integrated dataset which has extreme influence on data quality. This in turn descreases the cost of database maintenance.

In our approach, on the one hand, *explicit* mappings are created by predefined mappers which are based on category *descriptions* compared using information retrieval techniques. The category description is a textual description of each category and is generated from the values of other attributes (such as the `name` attribute) or extracted from an external source (i.e. World Wide Web). Moreover, the certainty of each mapping is expressed by a possibilistic truth value (PTV) and hence based on fuzzy set and possibility theory [17], [18]. On the other hand, the order relations and the explicit mappings are used to infer *implicit* mappings.

As a consequence there are established one-to-many mappings, which means that one category from the source dataset is mapped to one or more categories from the target dataset. The examples of mappings between categories forming hierarchies shown in Figure 1 and Figure 2 are presented in Figure 3. The dotted arrows indicate the mappings of categories from the source dataset to categories of the target dataset, e.g., the mapping of *campground* to *Camp Site*.

Many problems have to be addressed while devising such

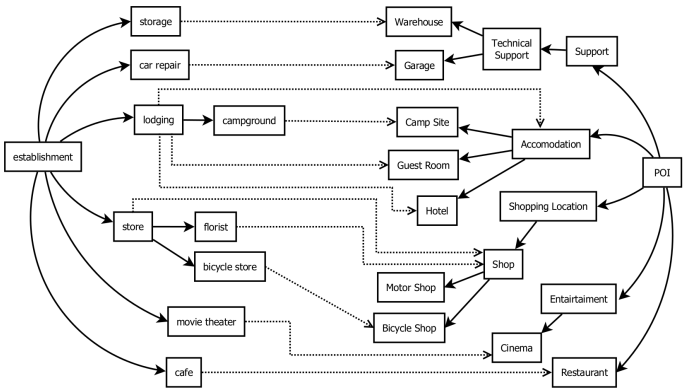

Fig. 3. Example of mappings between categories of dataset A and dataset B.

a mapping algorithm. The most important among them are the following:

- How to create a mapping between categories from heterogeneous sources?
- How can a partial order relation can be used to create a mapping between categories?
- Should the mapping be biased towards more specific or more general information?

Paper outline

The remainder of this paper is organized as follows. In Section II, overview of work related to the topic of this paper is provided. Next, in Section III some preliminary concepts are introduced that serve as a theoretical foundation of this paper. In Section IV a framework of value mapping is introduced and the algorithm is briefly described. Section V and Section VI contain the detailed description of applied mapping functions (mappers). Next, Section VII presents the results of computational experiments. Finally, we conclude and point out directions for a future work in Section VIII.

II. Related work

The problem of objects mapping has been studied in different contexts such as record linkage [12], duplicate detection [8], [19], [20], data integration [10] and knowledge base construction [21]. Most of the previous works assume that values of corresponding attributes are drawn from the same domain or at least that they bear some textual similarity that can be measured using a kind of the distance (e.g. edit distance, Jaccard). Moreover, some approaches are based on statistical information processing [11], [14], [16]. For instance, Kang et al. in [11] exploit a statistical model which captures the co-occurence of values of all attributes characterizing datasets. Next, constructed models are aligned assuming various matchings between the values of a given attribute in both datasets. The alignment with the minimum distance between thus aligned models is returned as the mapping. In [16] a strategy is presented which consists in the usage of statistical techniques to detect overlapping subsets of data present in disparate sources, through which rules for data conversion

may be extracted. In [15] domain independent string transformations are proposed to compare syntactically object's shared attributes. The established mappings depend on the mapping rules which are determined by a mapping learner and supervised by the user. In contrast, in [9], mappings is based on non-overlapping correlated attributes using a combination of profilers which contain the specific knowledge about what constitutes a typical concept.

To the best of our knowledge, none of the mentioned approaches rely on a partial order relation to establish automatically semantical mappings for attributes values with different domains. In this paper, the order relation is used for the first time in the context of object mapping.

III. Preliminaries

Within the scope of this paper it is assumed that entities from the real world are described as *objects* which are characterized by a number of *attributes* $a \in A$.

A *schema* of a given dataset is identified with the set of attributes A. For each attribute $a \in A$, let $dom(a)$ denote the domain of a (the set of possible values for attribute a).

Problem Definition

Two datasets are considered. The source dataset over the schema $A_S = \{a_1^S, \ldots, a_n^S\}$ is denoted as S, while the target dataset over the schema $A_T = \{a_1^T, \ldots, a_m^T\}$ is denoted as T. We assume that the schema matching is known:

$$f : A_S \rightarrow A_T \tag{1}$$

Moreover, at least one *category* attribute $a_C^S \in A_S$ is distinguished with values $c^S \in dom(a_C^S)$ (called categories for short), and similarly $a_C^T \in A_T$ with values $c^T \in dom(a_C^T)$. The category attribute of an object is an ordinal attribute [22] of which the values (categories) are (partially) ordered by means of a generalization/specialization relation. Besides that, the set of one-to-many categories mappings is defined as follows.

Definition 1 (One-To-Many Categories Mapping): The categories mapping is a one-to-many directional relation $R_{M1:m}$ if it maps one category $c^S \in dom(a_C^S)$ to a nonempty subset of categories $\{c_1^T, ..., c_i^T\} \subseteq dom(a_C^T)$, called the candidate categories set, representing the coreferent categories of a pair of corresponding attributes $a_C^S \in A_S$ and $a_C^T \in A_T$, where $f(a_C^S) = a_C^T$ This mapping is denoted as:

$$R_{M1:m} : dom(a_C^S) \longrightarrow_{1:m} range(a_C^T) \tag{2}$$

where dom(a_C^S) *is a domain of the attribute* a_C^S *and* range(a_C^T) *is a subset of* $dom(a_C^T)$ *comprising values of* a_C^T. *Thus, one-to-many categories mapping* $R_{M1:m}$ *is a set of pairs* $\{(c^S, c_1^T), ..., (c^S, c_i^T)\}$.

In our approach we use PTVs to express the confidence (certainty) in the validity of the mappings produced by an algorithm. Hereby, a PTV is a normalized possibility distribution [18] defined over the set of Boolean values $\mathbb{B}$ [17]. A PTV expresses the uncertainty about the Boolean value of a *proposition* p. In the context considered here, the propositions p of interest are of the form:

$$p = c^S \text{ and } c^T \text{ are coreferent}$$

where c^S and c^T are two given categories, i.e., values of the category attributes from two datasets under consideration.

Let P denote a set of all propositions under consideration. Then each $p \in P$ can be associated with a PTV denoted $\tilde{p} = \{(T, \mu_{\tilde{p}}(T)), (F, \mu_{\tilde{p}}(F))\}$, where $\mu_{\tilde{p}}(T)$ represents the possibility that p is true and $\mu_{\tilde{p}}(F)$ denotes the possibility that p is false. The domain of all possibilistic truth values is denoted $\mathcal{F}(\mathbb{B})$, i.e., is the fuzzy power set of (normalised) fuzzy sets over $\mathbb{B}$.

Let us define the order relation $\geq$ on the set $\mathcal{F}(\mathbb{B})$ by:

$$\tilde{p} \geq \tilde{q} \Longleftrightarrow \begin{cases} \mu_{\tilde{p}}(F) \leq \mu_{\tilde{q}}(F), & \mu_{\tilde{p}}(T) = \mu_{\tilde{q}}(T) = 1 \\ \mu_{\tilde{q}}(T) \leq \mu_{\tilde{p}}(T), & \text{else.} \end{cases} \tag{3}$$

IV. Mapping of categories

Before we continue to describe our method to map values of category attributes, first of all we consider different types of mappings.

A. Equivalent and Non-equivalent Mappings

Let us consider our exemplary datasets S and T shown in Fig. 1 and Fig. 2, respectively. The mappings shown in Fig. 3 may be intuitively conceived. A group may be distinguished among these mappings which is represented by: *movie theater* can be mapped to *Cinema*, *campground* to *Camp Site*, *car repair* to *Garage*. These mappings are valid in both directions, i.e., from the dataset S to the dataset T and inversely, because these categories represent coreferent information on the same level of abstraction. These mappings are called *equivalent* mappings.

In contrast, mappings such as the one between concepts *lodging* and *Hotel* are asymmetric, in a sense. On the one hand, not each *lodging* is a *Hotel*. Therefore *lodging* should be mapped to a more general concept than *Hotel*. On the other hand, each *Hotel* is a *lodging*. These categories describe different levels of abstraction, they are not equivalent, i.e. *lodging* is more general concept than *Hotel* and *Hotel* is a specialization of *lodging*. Therefore, these mappings are called *non-equivalent* mappings which are further divided into two subclasses. The first one, called *generalized* mappings, contains mappings in which the target category is a generalization of the source category and it is a valid mapping but on a different level of abstraction. In contrast to that, a mapping of which the target category is a specialization of the source category is called *specialized* mapping and it may be an invalid mapping; however, between those categories there still exists a strong semantical relation.

Due to the above described conditions, the direction of mapping has to be considered during the data integration. In this paper we consider directional mappings from the source dataset S to the target dataset T.

B. Algorithm

Our method detects coreferent categories and establishes one-to-many relationship between them (in most cases) which is defined by Definition 1. An example of such a relationship (dotted arrows) is shown in Figure 3. Such a relationship can be further processed to reduce it to a one-to-one form. Such a processing as well as classification of a mapping as equivalent or non-equivalent may be needed in data integration, but this is out of the scope of this paper and it is studied, e.g., in [23].

The Category Mapping Algorithm (Algorithm 1) creates mappings for attributes equipped with a partial order relation that reflects a notion of generality. Therefore, a partial order relation R_S on the domain of the source category attribute $a_C^S \in A_S$ and a partial order relation R_T on the domain of the target category attribute $a_C^T \in A_T$ are assumed to be given. Moreover, given are also the datasets S (source) and T (target), and an extensible set of mapping methods, called mappers. These mappers are classified as *explicit* mappers M_E and *implicit* mappers M_I and are detailed in Section V and VI respectively. In the first step of our Algorithm 1 (lines 1-3) explicit mappings are established which are based on instance data of the datasets or external source. In the second step (lines 4-6) implicit mappings are inferred which are based on the explicit mappings and the partial order relations R_S and R_T. The output of the algorithm is a relation $R_{M1:m}$ stating coreference of categories.

The coreference of a pair of values is assumed to be a binary notion, i.e., two values are coreferent or not. However, one may be uncertain if it holds or not for a given pair of values. Thus, all the mappers associate each mapping they produce with uncertainty which is expressed by a PTV. Each mapping with PTV equal or close to (1,0) is meant as holding with high confidence. In contrast, a mapping with PTV close to (0,1) or (1,1) means that the information about compared category values are not enough to claim relation between them. Therefore, only mappings associated with a PTV above predefined threshold for necessity of truth are taken into account.

Algorithm 1 CATEGORYMAPPINGALGORITHM

Require: Dataset S, Dataset T, Order Relations R_S and R_T, Mappers M
Ensure: Relation stating coreference of categories $R_{M1:m}$
1: **for all** $m \in M_E$ **do**
2: $R_{M1:m} \leftarrow$ m.getMappings(dom(a_C^S),dom(a_C^T),S,T)
3: **end for**
4: **for all** $m \in M_I$ **do**
5: $R_{M1:m} \leftarrow$ m.getMappings($R_{M1:m}$,R_S,R_T)
6: **end for**

V. EXPLICIT MAPPINGS

The explicit mappings are created by *description mappers* and *definition mappers* which are based on different information. A *description mapper* is based on information about each category which is extracted from the source or target datasets. The *definition mapper* is based on information extracted from an external source (e.g., World Wide Web).

The considered mappers are based on a textual description of each category value (called *category description*) which is constructed from mapper-dependent information but in the similar way which is explained below.

Category description

The generation of the category description is divided into two phases: *terms preprocessing* and *terms importance calculation*. During the preprocessing phase, for each category value a representation in the form of a set of words/terms is obtained in the following way. The starting point is a collection of relevant strings, e.g., values of selected attributes, such as `name` in Tables I or II, of all objects belonging to the same category in the source/target dataset. First, special characters appearing in these strings, i.e. dash, semicolon, dot etc., are replaced by white space which gives a string of terms. Second, the strings are splitted into terms where the splitter is the white space character. Afterwards, each category is described by a set of terms which is further preprocessed by removing *stop words* and applying an algorithm for suffix stripping. *Stop words* are terms such as *a*, *and* or *to* in English. A popular predefined set of stop words contains 527 terms [24] and is used also in our computational experiments. Moreover, terms are stemmed using the Porter stemmer [25]. For instance, terms *connection*, *connecting* and *connections* are transformed to their stem which is *connect*.

The result of the term preprocessing phase is a clean and unified set of terms for each category value. The preprocessing increases the quality of terms importance calculation in the final phase of generating the category description. The importance of each term is expressed by the *tfidf* [26], [27] (term frequency and inversed document frequency) weighting scheme. This coefficient reflects that a term which occurs often but not in many other descriptions tend to be more relevant and informative than a term which appears in many descriptions. *Tfidf* weighs the frequency of a term t in a description d ($t \in d$) with a factor that discounts its importance with its appearances in the whole descriptions collection D of the single dataset, which is defined as:

$$tfidf(t,d) = tf(t,d) \times log\frac{|D|}{df(t)} \tag{4}$$

where $tf(t,d)$ is the frequency of term t in the description d and is expressed by Equation 5, $|D|$ is the size of the whole descriptions collection and $df(t)$ is a number of descriptions in which term t occurs (term t occurs at least in one description).

$$tf(t,d) = \frac{card(t,d)}{\sum_i^{|d|} card(t_i,d)} \tag{5}$$

where $card(t,d)$ is the cardinality of term t in a description d and is divided by the sum of cardinalities of all term from description d to prevent a bias towards longer documents. Moreover, $tfidf$ weights are normalized as follow [26]:

$$tfidf^*(t,d) = \frac{tfidf(t,d)}{\sqrt{\sum_i^{|d|} tfidf_i^2}} \tag{6}$$

where $tfidf^*$ is the normalized $tfidf$ and $tfidf_i$ is the $tfidf$ weight of a term from a description d. The final description category is represented as a set which consists of a unique term t_i with a normalized weight $tfidf_i^*$.

A. *Description Mapper*

The description mapper creates mappings based on the comparison of category descriptions of the source and the target categories which are generated using values of selected objects attributes. This mapper works as follows.

First of all, the category description for each category from the source and the target is constructed (see paragraph Category Description in Section V). To this aim, objects from the dataset are grouped into clusters of the same category. For each cluster, values of *selected* attributes are used to generate the category description (see paragraph Category Description in Section V). The selection of attributes can be done automatically by selecting all textual attributes. For some attributes, such as `name`, stemming is not employed because they contain many proper nouns which may mislead the algorithm.

Afterwards, constructed category descriptions of the source and target categories are pair-wise compared. This comparison estimates the possibility that two given categories (their descriptions) are coreferent and is based on *intersection*. More specifically, the detected common terms (with weights) of both descriptions are added to the *intersection* that is a common subset of both descriptions, while the remaining terms (with weights) are added to the subset called *errors* and have an influence on the computed possibility that categories are (not) coreferent.

This mapper generates a PTV which expresses the uncertainty about the coreference of the compared descriptions as described above. The possibility that a proposition p, stating that two categories are coreferent, is true ($\mu_{\tilde{p}}(T)$) and the possibility that p is false ($\mu_{\tilde{p}}(F)$) are calculated by the following equations:

$$\mu_{\tilde{p}}(T) = \frac{possT}{factor} \tag{7}$$

$$\mu_{\tilde{p}}(F) = \frac{possF}{factor} \tag{8}$$

where *possT*, *possF* and *factor* are equal:

$$possT = \left(\sum_{i=1}^{|intersection|} \frac{tfidf_i^{*S} + tfidf_i^{*T}}{2} \right)^{pow} \tag{9}$$

$$possF = \sum_{i=1}^{|errors|} tfidf_i^* \tag{10}$$

$$factor = \max(possT, possF) \tag{11}$$

where $|intersection|$ denotes the number of common terms, and $|errors|$ is the number of the remaining terms present in the representation of the source and target categories, $tfidf_i^{*S}$ ($tfidf_i^{*T}$) is a weight of the common term from the description of the category from the source (the target respectively) and $tfidf_i^*$ is a weight of a remaining term from the description of the category from the source or the target.

On the one hand, $possT$ is the sum of the average of common terms weights and raised to the predefine power pow which is set for this mapper to 3. On the other hand, $possF$ is computed as the sum of the weights $tfidf_i^*$ of the remaining terms (from errors that are found during comparison). Finally, $factor$ is the maximum of $possT$ and $possF$ and is used to normalize both possibilities. If the PTV of a mapping of any category description of the source and any category description of the target is above the predefined threshold then the mapping between considered categories is added to $R_{M1:m}$.

For instance, consider categories *lodging* from the source and *Hotel* from the target. Let the source dataset contain objects with category *lodging* as shown in Table I. A part of the description of *lodging* contains the following terms with $tfidf^*$: *Hotel* 0.85, *Bed* 0.22, *Breakfast* 0.22, *Resort* 0.08, *Inn* 0.03, *Carlton* 0.012, *Borluut* 0.006, etc. On the one hand, terms *Carlton* and *Borluut* get low weights because they appear infrequently in objects of category *lodging* and, moreover, they are not specific for this category. On the other hand, terms *Bed* and *Hotel* get higher weights because they are specific terms for accomodation and they appear in many objects of this category. Meanwhile, a part of description of the target category *Hotel* which is based on the names of objects from the target, shown in Table II, is the following: *Hotel* 0.96, *Inn* 0.14, *Resort* 0.08, *Bed* 0.01, *Breakfast* 0.01, *Carlton* 0.005, etc. Afterwards, the sets are compared and the common terms are detected. Finally, the PTV of this mapping is calculated using Equations (7) and (8) and equals (1,0.06) which supports high confidence in the coreference relationship between the considered categories.

B. *Definition Mapper*

In contrast to the previous mapper, this method is based on the extraction of additional knowledge from an external source. More precisily, for each category the textual description is extracted from the Web, in particular, from the Wikipedia[3]. The textual description is a webpage (e.g., for the category *lodging* it is a webpage at the URL http://en.wikipedia.org/wiki/lodging) which is parsed by JSoup HTML Parser[4]. Extracted texts are used to construct the *category description* (see paragraph Category Description in Section V). Afterwards, the category descriptions are pair-wise compared and uncertainty as to their matching is quantified like in case of the *description* mapper but with the parameter pow in (9) set to 4 what have been experimentally confirmed to make the results of both types of the mappers comparable.

The extraction works as follows. First of all, for each category a predefined number of paragraphs from a relevant Wikipedia web page is extracted. Next, hyperlinks present in the extracted paragraphs are followed and further paragraphs are extracted from the target web pages. This is continued until the predefined *reference level* is reached. The reference

[3]Wikipedia, http://www.en.wikipedia.org

[4]JSoup HTML Parser, http://www.jsoup.org

level states the limit of such a recursive hyperlinks following. We set the reference level to 1 and the number of extracted paragraphs to 2 based on experimental results.

For instance, a part of the extracted textual description of category *lodging* from the source dataset is the following (terms in bold refer to linked pages and for them recursive extraction was executed):

Lodging** (or a holiday accommodation) is a type of residential **accommodation** (*author's note: refer to ***dwelling). People who **travel** and stay away from home for more than a day need lodging for sleep, rest, safety, shelter from cold temperatures or rain, storage of luggage and access to common household functions. Lodging is done in a **hotel**, **motel**, **hostel** or **hostal**, a private home (commercial, i.e. a **bed and breakfast**, a **guest house**, a **vacation rental**, or non-commercially, with members of **hospitality services** or in the home of friends), in a **tent**, caravan/**camper** (often on a **campsite**) (...)*[5].

While a part of the textual description of category *Accomodation* from the target is the following:

***Accomodation** may refer to: a **dwelling**, a place of temporary **lodging** (...)*[6]. *A **dwelling** (also residence, abode) is an important legal concept which defines a self-contained unit of accommodation used by one or more **households** as a **home**, such as a **house**, **apartment**, (...)*[7].

These texts are used to generate category descriptions. A part of the description of *lodging* contains the following most important terms (their stems) with weights: lodg 0.3957, backpack 0.3359, hous 0.3051, room 0.2277, accommod 0.2257, residenti 0.099, facil 0.078, home 0.081, household 0.064, etc. While a part of the description of the category *Accommodation* from the target is the following: home 0.3977, lodg 0.3808, accommod 0.1458, household 0.1434, residenti 0.131, facil 0.041, etc. The descriptions comparison of *lodging* and *Accommodation* returns the mapping with PTV equal (1,0.17). The uncertainty about this mapping is low because the description of *Accomodation* contains the description of *lodging*. Thus, the results confirm intuition that *lodging* is coreferent to *accomodation*.

VI. Implicit mappings

Besides the explicit mappings, our method returns implicit mappings (lines 4-6 in Algorithm 1). We consider two types of implicit mappings. The first type of implicit mapping (Implicit Mapping I) is established when for the specific category c_1^S from the source there does not exist any explicit mapping. It is a mapping of the first one-level more general concept c_2^S in a partial order relation R_S of the considered category $c_1^S \in dom(a_C^S)$ from the source dataset for which there exists an explicit mapping $m \in R_{M1:m}$. The implicit mapping for c_1^S is thus $c^T \in dom(a_C^T)$ such that c^T is the target category to which c_2^S is mapped via m. It should be stressed that the quality of the implicit mappings depends on the correctness of explicit mappings and the order relation.

For instance, R_S in Figure 1 contains the following pairs of elements: (*establishment*, *store*), (*store*, *florist*), where *establishment* is the most general concept. Suppose that the set of already detected mappings contains mapping (*store*, *Shop*). If the category *florist* is not mapped explicitly then the algorithm returns an implicit mapping of *florist* to *Shop*.

Moreover, the set of mappings is extended by the second type of implicit mappings (Implicit Mapping II) which work as follow. For each target category of a mapping categories from the target partial order relation R_T which are generalizations of the considered category are extracted. Next, there are established mappings between the considered category and extracted generalizations.

For instance, suppose that there exists mapping between *lodging* from the source and *Hotel* from the target. Using the order relation shown in Fig. 2 the generalizations of *Hotel* are categories *Accomodation* and *POI*. Thus, mappings are established between them and the source category *lodging*.

VII. Evaluation and Discussion

The evaluation of our algorithm is conducted based on two real-world datasets. The source dataset contains around 200000 objects and a partial order relation on the set of 100 categories which are extracted from the Google Maps database by the Google Places API[6]. The target contains around 430 000 objects and a partial order relation on the set of 502 categories which are shared by RouteYou[7].

Our algorithm employs two thresholds to decide on categories coreference which are set to 0.2 for $\mu_{\tilde{p}}(F)$ of Definition Mapper (0.1 of Description Mapper respectively) and to 0.5 for $\mu_{\tilde{p}}(T)$ of both mappers based on experimental results. If $\mu_{\tilde{p}}(F)$ is lower than the threshold then the coreference is declared. If $\mu_{\tilde{p}}(T)$ is lower than the threshold then the lack of coreference is declared. Finally, if both of the thresholds are exceeded then the coreference status is declared as *unknown*.

For both datasets in total our algorithm suggested 263 mappings between the categories from the source and the target using mappers which are described in Sections V and VI. A part of the results is presented in Table III. Each mapping consists of four values: the source category, the target category, PTV and the name of the mapper which produced the specific mapping. There can be distinguished equivalent mappings (i.e. *lodging* and *Accomodation*, *establishment* and *POI*), generalized non-equivalent mappings (i.e. *church* and *Place of Worship*) and specialized non-equivalent mappings (i.e. *lodging* and *Hotel*).

The quality of our method is evaluated using two standard measures of recall and precision. The precision is a fraction of detected real coreferent mappings among all detected mappings, the recall is a number of detected real coreferent mappings divided by the number of all real coreferent mappings. The sets of possible real coreferent equivalent and non-equivalent mappings are provided manually by experts. Table IV presents the quality measures of our approach for equivalent mappings (column 2), equivalent and generalized

[5]The text is extracted from the definition of *lodging* in Wikipedia, http://en.wikipedia.org/wiki/Lodging

[6]The text is extracted from the definition of *Accomodation* in Wikipedia, http://en.wikipedia.org/wiki/Accomodation

[7]The text is extracted from the definition of *dwelling* in Wikipedia, http://en.wikipedia.org/wiki/Dwelling

[6]Google Places, http://developers.google.com/places/

[7]RouteYou, http://routeyou.com/

non-equivalent mappings (column 3), equivalent and all non-equivalent mappings (column 4).

Additionally, the last column in Table IV contains of recall and precision for *any but not the most general* mappings. They show if there is established at least one coreferent mapping for each category from the source which is not a mapping for the most general category in the order relation R_S, called *root* which is *POI* for R_S (by Definition 1 each category from the source can be mapped to the root). Thus, in this case the precision and recall are calculated as follows. The precision is a number of distinct source categories (different from the root) of detected real coreferent mappings divided by the number of distinct source categories (different from the root) of all detected mappings, the recall is a number of distinct source categories (different from the root) of detected real coreferent mappings divided by the number of distinct source categories (different from the root) of all real coreferent mappings.

Firstly, the recall and the precision in Table IV are calculated for explicit mappers (above the bar), for which high thresholds are set. As consequence they return high precision but low recall for equivalent and non-equvalent mappings (column 4). The high precision is important because these mappings are used by the implicit mappers: any false positive mappings are propagated by implicit mappers what lowers the overall result. Besides that, it is not crucial to detect all posible coreferent mappings but it is sufficient if the algorithm creates at least one coreferent mapping for each category which turns out to be the proper mapping (the selection of mappings is out of the scope of this paper and is investigated in [23]). The results in the last column confirm that, for around half of the categories from the source at least one coreferent mapping is established which is different from root with the precision equals 0.93.

Next, the recall and the precision are calculated for explicit and implicit mappings combined. The implicit mappers decrease the precision for equivalent mappings (column 1 in Table IV) because they create mostly the non-equivalent mappings. Thus, for non-equivalent mappings (column 3 and 4 in Table IV) these mappers increase the precision and the recall, i.e. the precision from 0.41 to 0.57 and the recall from 0.11 to 0.4 for equivalent and generalized mappings (column 3); the precision from 0.78 to 0.79 and the recall from 0.12 to 0.33 for equivalent and non-equivalent mappings (column 4). Besides that, for almost half of categories from the source there is established at least one coreferent mapping which is different from root with the precision equals 0.98.

TABLE III. SOME RESULTS OF ALGORITHM: MAPPINGS OF VALUES FROM THE SOURCE AND TARGET DATASETS.

Category from S	**Category from T**	**PTV**	**Mapper**
art gallery	Art Museum	(1,0)	Definition
art gallery	Museum	(1,0)	ImplicitII
art gallery	Cultural Centre	(1,0)	ImplicitII
art gallery	Tourist Attract	(1,0)	ImplicitII
art gallery	Recreation	(1,0)	ImplicitII
art gallery	Building Constr	(1,0)	ImplicitII
art gallery	POI	(1,0)	ImplicitII
restaurant	Eatery	(1,0)	Definition
restaurant	Eat and Drink	(1,0)	ImplicitII
restaurant	POI	(1,0)	ImplicitII
restaurant	Road Restaurant	(1,0.003)	Definition
restaurant	Hotel	(1,0.09)	Description
amusement park	Themepark	(1,0)	Definition
amusement park	Amusement Park	(1,0)	Definition
amusement park	Recreation	(1,0)	ImplicitII
amusement park	POI	(1,0)	ImplicitII
church	church	(1,0)	Definition
church	Place of Worship	(1,0)	ImplicitII
church	POI	(1,0)	ImplicitII
church	Place of Worship	(1,0.1)	Description
storage	Warehouse	(1,0.001)	Definition
storage	POI	(1,0.001)	ImplicitII
establishment	POI	(1,0.015)	Description
natural feature	Waterfall	(1,0.05)	Definition
natural feature	Landscape Element	(1,0.05)	ImplicitII
natural feature	POI	(1,0.05)	ImplicitII
natural feature	Dunes	(1,0.12)	Definition
lodging	Hotel	(1,0.06)	Description
lodging	Accomodation	(1,0.06)	ImplicitII
lodging	Accom Shelter	(1,0.06)	ImplicitII
lodging	POI	(1,0.06)	ImplicitII
lodging	Accomodation	(1,0.17)	Definition
food	Restaurant	(1,0.1)	Description
food	Eatery	(1,0.1)	ImplicitII
food	Eat and Drink	(1,0.1)	ImplicitII
food	POI	(1,0.1)	ImplicitII
meal delivery	Restaurant	(1,0.1)	Implicit
meal delivery	Eatery	(1,0.1)	ImplicitII
meal delivery	Horeca	(1,0.1)	ImplicitII
meal delivery	Eat and Drink	(1,0.1)	ImplicitII
meal delivery	POI	(1,0.1)	ImplicitII

VIII. CONCLUSION AND FUTURE WORK

We present a novel automatic method to establish one-to-many semantical mappings between attributes values from different domains. This method applies an extensible set of mappers which are based on the constructed textual descriptions of considered values and employs information retrieval techniques for further processing. Moreover, we have also shown how a known partial order relation defined on the domain of considered attributes can be used to create mappings.

Our approach generates the set of mappings where each value from the source dataset is related to at least one value from the target dataset. It, in general, produces alternative mappings for the same value. However, it may be crucial to select for each value from the source exactly one from the target. Thus, more sophisticated selection method than the method which is based on the lowest uncertainty about mapping have to be investigated [23]. Moreover, our method can be easily applied for data integration or interoperability tasks.

In the future we plan to extend the set of mappers, e.g., by a duplicates detection based mapper or a domain specific mapper which will be able to exploit also other attributes such as those representing the coordinates of an object. This should increase the number of detected correct mappings and improve the quality of our approach.

ACKNOWLEDGMENT

This contribution is supported by the Foundation for Polish Science under International PhD Projects in Intelligent Com-

TABLE IV. RESULTS OF THE EVALUATION: PRECISION AND RECALL.

Mapper	Equivalent		Equivalent & Generalized		Equivalent & Non-equivalent		Any Not The Most General	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Description	0.13	0.03	0.31	0.01	0.94	0.02	1.00	0.09
Definition	0.39	0.52	0.44	0.10	0.76	0.10	0.92	0.39
Desc., Def.	0.34	0.54	0.41	0.11	0.78	0.12	0.93	0.45
Desc., Implicit I	0.09	0.03	0.32	0.02	0.77	0.03	0.83	0.11
Desc., Implicit II	0.09	0.08	0.40	0.06	0.93	0.08	1.00	0.09
Desc., Implicit I & II	0.06	0.08	0.41	0.09	0.78	0.10	0.83	0.11
Def., Implicit I	0.39	0.52	0.44	0.10	0.76	0.10	0.92	0.39
Def., Implicit II	0.17	0.56	0.63	0.35	0.77	0.25	0.97	0.41
Def., Implicit I & II	0.19	0.56	0.63	0.35	0.77	0.25	0.97	0.41
Desc., Def., Implicit I	0.33	0.54	0.42	0.12	0.78	0.13	0.93	0.47
Desc., Def., Implicit II	0.15	0.59	0.58	0.38	0.79	0.32	0.98	0.47
All	0.14	0.59	0.57	0.40	0.79	0.33	0.98	0.49

puting. Project financed from The European Union within the Innovative Economy Operational Programme 2007-2013 and European Regional Development Fund.

REFERENCES

[1] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *The VLDB Journal*, vol. 10, no. 4, pp. 334–350, Dec. 2001.

[2] M. Szymczak, S. Zadrozny, and G. De Tré, "Coreference detection in xml metadata," in *2013 Joint IFSA World Congress NAFIPS Annual Meeting, Proceedings*, W. Pedrycz and M. Reformat, Eds., 2013, pp. 1354–1359.

[3] J. Madhavan, P. A. Bernstein, and E. Rahm, "Generic schema matching with cupid," in *Proceedings of the 27th International Conference on Very Large Data Bases*, ser. VLDB '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 49–58.

[4] H. hai Do and E. Rahm, "Coma - a system for flexible combination of schema matching approaches," in *In VLDB*, 2002, pp. 610–621.

[5] A. Bilke and F. Naumann, "Schema matching using duplicates," in *Proceedings of the 21st International Conference on Data Engineering, ICDE 2005, 5-8 April 2005, Tokyo, Japan*. IEEE Computer Society, 2005, pp. 69–80.

[6] R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos, "imap: discovering complex semantic matches between database schemas," in *in: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, ACM*. Press, 2004.

[7] M. Szymczak and J. Koepke, "Matching methods for semantic annotation-based xml document transformations," in *K. Atanassov, et al. (Eds.), New Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics. Applications. Volume II*. SRI PAS, 2012, pp. 297–308.

[8] R. Ananthakrishna, S. Chaudhuri, and V. Ganti, "Eliminating fuzzy duplicates in data warehouses," in *Proceedings of the 28th International Conference on Very Large Databases (VLDB 2002)*, 2002.

[9] A. Doan, Y. Lu, Y. Lee, and J. Han, "Object matching for information integration: A profiler-based approach," in *In: Proceedings of the IJCAI-03 Workshop on Information Integration on the Web. (2003*, 2003, pp. 53–58.

[10] F. Naumann, A. Bilke, J. Bleiholder, and M. Weis, "Data fusion in three steps: Resolving inconsistencies at schema-, tuple-, and value-level," in *IN BULLETIN OF THE TECHNICAL COMMITTEE ON DATA ENGINEERING*, 2006, pp. 21–31.

[11] J. Kang, D. Lee, and P. Mitra, "Identifying value mappings for data integration: An unsupervised approach." in *WISE*, ser. Lecture Notes in Computer Science, A. H. H. Ngu, M. Kitsuregawa, E. J. Neuhold, J.-Y. Chung, and Q. Z. Sheng, Eds., vol. 3806. Springer, 2005, pp. 544–551.

[12] I. P. Fellegi and A. B. Sunter, "A theory for record linkage," *Journal of the American Statistical Association*, vol. 64, pp. 1183–1210, 1969.

[13] M. Craven, D. Dipasquo, D. Freitag, A. Mccallum, T. Mitchell, and K. Nigam, "Learning to construct knowledge bases from the world wide web," *Artificial Intelligence*, vol. 118, pp. 69–113, 1999.

[14] W. W. Cohen, "Integration of heterogeneous databases without common domains using queries based on textual similarity," 1998, pp. 201–212.

[15] S. Tejada, C. A. Knoblock, and S. Minton, "Learning domain-independent string transformation weights for high accuracy object identification," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '02. New York, NY, USA: ACM, 2002, pp. 350–359.

[16] H. Lu, W. Fan, C. H. Goh, S. E. Madnick, and D. W.-L. Cheung, "Discovering and reconciling semantic conflicts: A data mining perspective." in *DS-7*, 1997, pp. 409–427.

[17] H. Prade, "Possibility sets, fuzzy sets and their relation to Lukasiewicz logic," in *Proc 12th Int Symp on Multiple-Valued Logic*, 1982, pp. 223–227.

[18] L. A. Zadeh, "Fuzzy sets as a basis for a theory of possibility," *Fuzzy Sets Syst.*, vol. 100, pp. 9–34, Apr. 1999.

[19] M. Weis and F. Naumann, "Detecting duplicate objects in xml documents." in *IQIS*, F. Naumann and M. Scannapieco, Eds. ACM, 2004, pp. 10–19.

[20] ——, "Dogmatix tracks down duplicates in xml." in *SIGMOD Conference*, F. zcan, Ed. ACM, 2005, pp. 431–442.

[21] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery, "Learning to construct knowledge bases from the world wide web," *Artificial Intelligence*, vol. 118, pp. 69 – 113, 2000.

[22] S. S. Stevens, "On the theory of scales of measurement," *Science*, vol. 103, no. 2684, pp. 677–680, 1946.

[23] M. Szymczak, A. Bronselaer, S. Zadrozny, and G. De Tré, "Selection of semantical mappings of attribute values for data integration," in *To appear in IEEE Intelligent Systems 2014, Proceedings*. Springer, 2014, pp. 1–12.

[24] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009.

[25] M. F. Porter, "Readings in information retrieval," K. Sparck Jones and P. Willett, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, ch. An Algorithm for Suffix Stripping, pp. 313–316.

[26] G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval," Ithaca, NY, USA, Tech. Rep., 1987.

[27] W. B. Frakes and R. Baeza-Yates, Eds., *Information Retrieval: Data Structures and Algorithms*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.