An Exploration Of Understanding Heterogeneity Through Data Mining

Haishan Liu
University of Oregon
Eugene, Oregon 97403
ahoyleo@cs.uoregon.edu

Dejing Dou
University of Oregon
Eugene, Oregon 97403
dou@cs.uoregon.edu

ABSTRACT

Development of internet and Web have resulted in many distributed information resources which in general are structurally and semantically heterogeneous even in the same domain. However, heterogeneity itself has not been studied in a formal way so that the representation of different kinds of heterogeneities can be generically processed by other programs automatically. Most descriptions and categorization schemes of heterogeneities were given in languages specific to different research groups. We believe that efforts invested in a thorough research of heterogeneity can ultimately benefit both data integration and data mining communities. In this paper we give a brief survey of various ways to categorize heterogeneity in the literature, and then performed a case study on detecting a specific class of heterogeneity in the setting of Semantic Web ontologies–the one that can be discovered by only data-driven approaches. Finally we propose an automatic ontology matching system that can detect this heterogeneity by using redescription mining techniques. We also believe that automatic ontology matching process is a helpful step in tasks of mining multiple information sources in the heterogeneous scenario.

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Data mining—*distributed data mining*; I.2.4 [**Knowledge Representation Formalism and Methods**]: Ontology; H.3.5 [**Information Systems**]: Information Storage and Retrieval—*data integration*

General Terms

Theory, Design

Keywords

Heterogeneity, Ontology Matching, Redescription Mining

1. INTRODUCTION

In both data integration and data mining communities, problems that might arise due to heterogeneity of multiple data resources are already well known. It is generally agreed to categorize conflicts between data resources into structural heterogeneity and semantic heterogeneity [14]. Structural heterogeneity means that different information sources store their data in different structures (e.g., relational vs. spreadsheet). Semantic heterogeneity considers differences of the content of data items and their intended meanings. In most of the distributed data mining (DDM) literatures, the heterogeneous scenario is restricted to the case where presumably different sets of attributes are defined across distributed databases [23] as disjoint models; in other words, data in each local site represent the incomplete knowledge about the complete data set. It is also termed as vertical data fragmentation [4].

There are significant progresses made by distributed data mining and information integration researchers in dealing with data heterogeneity problems. However, many challenges remain. First, Different research communities have different terms of definition and their focuses vary as well. Different languages specific to particular research groups are adopted to describe heterogeneity, which impedes effective knowledge sharing and reuse. In this paper we propose to use mapping rules in formal language to categorize heterogeneity and describe their characteristics. Second, heterogeneity is hard to discover automatically. Most of the current solution of distributed data mining and data integration systems require a step of manual specification of correspondences (matchings) in meta-data before heterogeneity resolution can be carried out. We propose an approach in this paper to discover meta-data matchings in a highly automatic way. We also observe that some kind of heterogeneity can be detected only by data-driven approaches. In the following of this paper, the attention is focused on the study of heterogeneity in the setting of Semantic Web ontologies.

In general, an ontology can be defined as the formal specification of a vocabulary of concepts and the relationships among them in a specific domain. In traditional knowledge engineering and in emerging Semantic Web research, ontologies play an important role in defining the semantics of data. We explore one kind of heterogeneity in ontologies that can be detected by data-driven approaches. A motivating example is given below.

Consider the scenario depicted in figure 1. The left (right) graph shows the structure of the source (target) ontology. The solid triangle connected to a node denotes the content of that node. The dashed dotted line depicts the matching.

As shown in the figure, *Vertebrate* can be paired to *mammal, fish, reptile, amphibian* and *bird* respectively. Any matching algorithm that explores the hypernym/hyponym relationship between the labels can discover these correspondences. However, the more accurate semantics should be *Vertebrate*$\leftrightsquigarrow$*mammal, fish, reptile, amphibian, bird*. The bi-directional arrow in the above expression denotes equivalency, meaning the set of *Vertebrate* contains

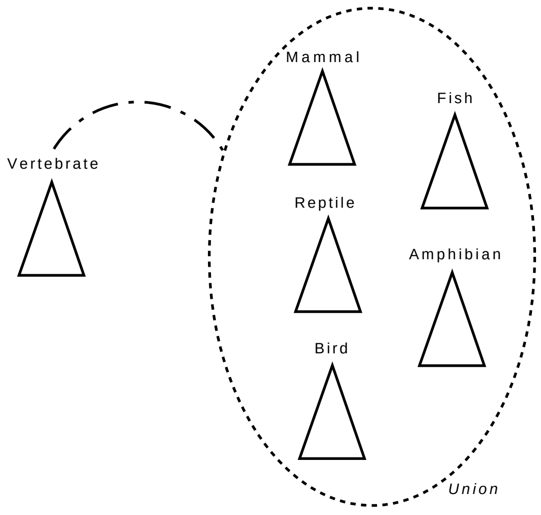

Figure 1: A Motivating Example

no more than the union of *mammal, fish, reptile, amphibian* and *bird.* This cannot be verified unless the data of source and target is examined. We developed novel methods to fulfil this task by means of matching discovery based on *redescription mining* techniques.

Redescription mining is a recently proposed approach for data mining tasks in domains which exhibit an underlying richness and diversity of data descriptors [26, 29, 22]. A redescription is a shift-of-vocabulary, or a different way of communicating information about a given subset of data. The goal of redescription mining is to determine the subsets that afford multiple definitions (i.e., descriptions) and to find these definitions, which is uniform with the objective of ontology matching in terms of relating concepts from different ontologies defined in the same or similar domains.

Ontology matching research is a discipline that aims at facilitating interoperability among different systems in Semantic Web and databases. Some ontology-based information integration systems have been developed to process ontology/schema matching. A survey can be found in [28]. Hence the general idea of our proposed approach is to recast the problem of ontology matching to discovering redescriptions among the named entities, including classes and properties, in different ontologies.

In terms of redescription, the matching depicted in figure 1 can be written as:

$$Vertebrate \rightsquigarrow Fish \cup Amphibian \cup Reptile \cup Bird \cup Mammal$$

This is a complex matching since it involves multiple concepts with a many-to-many correspondence. In some literature it is also referred to as mapping since it specifies the relationship among those concepts in terms of a set theory expression, which can be easily translated to *ontology mappings* in terms of other formal languages such as First-Order Logic rules. In our previous research, we defined the term "*ontology mappings*" as formal specifications of relationships of concepts from different ontologies. Ideally, they should be executable by software agents to perform tasks such as data integration/translation and can be used in distributed data mining tasks. We treated the term "ontology matching" as correspondence between concepts, which is less formal than mapping rules. Matching discovery is the first step to generate the mapping in our previous work. In this paper, we call the proposed data-driven approach. a complex matching discovery process.

During our previous work in data mining and data integration, we have collected a corpus of real data from different domains [1]. A great number of different kinds of heterogeneities have been observed.

Below is an example of how a category of heterogeneity, i.e., the naming conflicts of concepts, can be captured generally using the formal ontology mapping rule (in first-order logic form):

$\forall x P(x) \rightarrow Q(x);$

This rule states that class P in the source ontology is mapped to class Q in the target ontology, where classes in ontology are interpreted as unary predicates. For example, P and Q can be instantiated as *person* and *people*, which falls into the synonym subcategory under the naming conflict heterogeneity.

The following rule represents the conflict of property values in different ontologies by a mathematical transformation of their values:

$\forall x, y R(x, y) \rightarrow R'(x, f(y));$

Here the binary predicates R and R' denote two properties; x is the class, and y is the value of the property. For example we can instantiate this rule with R and R' being *age* and *birth_year*– both are properties of the $person$ class, and the function f means $age = current\_year - birth\_year$.

The rest of the paper is organized as follows. We first introduce some related work in Section 2 and summarize our preliminary understanding of heterogeneities based on our and other groups' previous work. Then we introduce our framework based on machine learning and data mining to discover and formally represent the heterogeneities between ontologies (in terms of complex matching). We text our methods in two case studies reported in Section 4. We conclude the paper by summarizing our contributions and discussing the future work in Section 5.

2. RELATED WORKS AND BACKGROUND

2.1 Heterogeneity Categorization

Various ways have been proposed to define different levels of heterogeneities in the literature. Goh *et al*[11] identified three main causes for semantic heterogeneity:

- Confounding conflicts occur when information items seem to have the same meaning, but differ in reality, e.g. due to different temporal contexts.
- Scaling conflicts occur when different reference systems are used to measure a value. Examples are different currencies.
- Naming conflicts occur when naming schemes of information differ significantly. A frequent phenomenon is the presence of homonyms and synonyms.

Noy *et al*[20] briefly outlined a list of semantic heterogeneity including using the same linguistic terms to describe different concepts; using different terms to describe the same concept; using different modeling paradigms (e.g., using interval logic or points for temporal representation); using different modeling conventions and levels of granularity; having ontologies with differing coverage of the domain, and so on.

Won Kim *et al* developed a framework[15] for enumerating and classifying the types of multidatabase system (MDBS) structural and representational discrepancies. The conflicts in a multidatabase system were mainly categorized in two cases: schema conflicts and data conflicts. They concluded that there are two basic causes of schema conflicts. First is the use of different structures for the same

[1]`http://aimlab.cs.uoregon.edu/benchmark`

information. Second is the use of different specifications for the same structures. The data conflicts are mainly due to 1) wrong data violating integrity constraints implicitly or explicitly, and 2) different representations for the same data.

In [12], Hammer *et al* proposed a systematic classification of different types of syntactic and semantic heterogeneities, which was then used to compose queries that make up a benchmark system for information integration systems. The classification consists of twelve cases including, for example, synonyms, simple mapping, union types, and etc.

A comprehensive scheme is proposed in [24]. Pluempitiwiriyawej *et al* classified heterogeneities of XML schemas defined in DTD files into three broad classes:

- Structural conflicts arise when the schema of the sources representing related or overlapping data exhibit discrepancies. Structural conflicts can be detected when comparing the underlying DTDs. The class of structural conflicts includes generalization conflicts, aggregation conflicts, internal path discrepancy, missing items, element ordering, constraint and type mismatch, and naming conflicts between the element types and attribute names.

- Domain conflicts arise when the semantic of the data sources that will be integrated exhibit discrepancies. Domain conflicts can be detected by looking at the information contained in the DTDs and using knowledge about the underlying data domains. The class of domain conflicts includes schematic discrepancy, scale or unit, precision, and data representation conflicts.

- Data conflicts refer to discrepancies among similar or related data values across multiple sources. Data conflicts can only be detected by comparing the underlying DOCs. The class of data conflicts includes ID-value, missing data, incorrect spelling, and naming conflicts between the element contents and the attribute values.

2.2 Ontology and Ontology Matching

Ontologies, which can be defined as the formal specification of a vocabulary of concepts and the relationships among them, are playing a key role to define data semantics on the Semantic Web [2] and various scientific domains, such as biological and medical data repositories. The general goal of Semantic Web is to make Web data machine-“understandable", so that web agents can process and share information automatically. Many publicly available, structurally and semantically rich resources such as databases, XML data and the Semantic Web data (e.g., RDF data) provide a unique and challenging opportunity to integrate information in new and meaningful ways. Research involving the Semantic Web is experiencing huge gains in standardization in that Web Ontology Language (OWL [1]) becomes the W3C standard for ontological definitions in web documents. OWL ontologies mainly consists of classes, datatypes and object properties and limited forms of axioms, such as subsumption, inverse relation, cardinality constraints of classes or properties. OWL also can be used to describe individual objects or data instances.

However, it is extremely unreasonable to expect that ontologies used for similar domains will be few in number [3]. For example, as the amount of data collected in the fields of Biology and Medicine grows at an amazing rate, it has become increasingly important to model and integrate the data with ontologies that are scientifically meaningful and that facilitate its computational analysis. Hence, efforts such as the *Gene Ontology* (*GO* [10]) in Biology and the *Unified Medical Language System* (*UMLS* [16]) in Medicine have been developed and have become fundamental to researchers working in those domains. However, different labs or organizations may still use different ontologies to describe their data.

Discovering semantic matchings has been one of major tasks and studied by both Semantic Web and database communities. Research in the Semantic Web has resulted in tools for ontology matching that are absolutely critical in semantic integration (see [20] for a survey). When two ontologies or two schemas do not have or do not share any data instances, the most straightforward approach is to study the similarity of names and structures of ontological concepts or schema attributes. For example, Chimaera [18] provides a ontology editor to allow user to merge ontologies. It suggests potential matchings based on the names of classes and properties. Protégé [21] gives initial ontology alignments by plugging in one of existing similarity matching algorithms. BMO [13] can generate block matchings using a hierarchical bipartition algorithm. This system builds a virtual document for each ontology and compares each pair of concepts with the information in the virtual document.

If two ontologies share data instances, the most straightforward way to compare them is to test the intersection of their instance sets. GLUE [6] is a system that employs multiple machine learning techniques to semi-automatically discover one-to-one matchings between two ontology taxonomies. The idea of the approach is to calculate the joint distributions of the classes, instead of committing to a particular definition of similarity. Thus, any particular similarity measure can be computed as a function over the joint distributions. iMAP [5] is a system that semi-automatically discovers one-to-one and even complex matchings between relational database schemas. The idea is to reformulated the matching problem as a search in a match space.

2.3 Redescription Mining

Redescription mining was first studied in [26]. Ramakrishnan *et al.* defined a redescription as a shift-of-vocabulary, or a different way of communicating information about a given subset of data instances. They also introduced CARTwheels algorithm to exploit the duality between class partitions and path partitions in an induced classification tree to model and mine redescriptions. Later in [29], Zaki and Ramakrishnan proposed an alternative algorithm to mine all minimal (non-redundant) redescriptions underlying a dataset using notions of minimal generators of closed itemset. Parida and Ramakrishnan [22] formally studied the space of redescriptions underlying a dataset and characterize their intrinsic structure. It also analyzed when mining redescriptions is feasible and how to custom-build a mining system for various biases.

In order to perform redescription mining algorithm to match ontologies, a universal set of instances of the ontologies should be specified in advance. This falls under the problem of *object reconciliation*. *Object reconciliation* problem is studied for determining whether two different data instances refer to the same real-world entity. It is closely related to instance-based matching approaches. The research in [7] proposed methods to address the reconciliation problem within the same schema. The algorithm takes three steps. It first construct the dependency graph based on the taxonomy information. Every node in the graph consists of one pair of entities which denotes a potential reconciliation decision. Second, it iteratively computes the similarity scores of reconciliation decisions. The similarity score of one reconciliation decision can both affect and be affected by the similarity score of its neighbors. The algorithm terminated when a fixed point is reached. Finally, transitive closure is computed to determine the final reconciliation decisions.

Our work is an extension of redescription mining in ontology matching to study heterogeneity. We focus on the scenario that two ontologies both have data instances and some of them refer to the same real-world entities. We extend the CARTwheels algorithm to incorporate ontology structure heuristics to guide its search in the space of redescriptions and to find more complex ontology matchings rather than one-to-one matchings. We also explore a new method to reconcile the instances described by different ontologies so that to generate a universal dataset for our redescription mining based ontology matching algorithm.

3. SYSTEM FRAMEWORK

Here we present our ontology-based approach using redescription mining to detect the specific class of heterogeneity discussed in sectoin 1. We assume that ontologies of given datasets are obtained in advance by either human specification or automatic extraction by machine from (semi-)structured data sources. There are several approaches proposed to investigate the transformation of relational schemas to ontologies (see [27, 17, 19]).

The proposed approach generally consists of following steps: first, parse the OWL document to extract both its the ontology structural information and data instances; second, perform the object reconciliation algorithm to construct a universal dataset; then perform the extended redescription mining process to discover complex ontology matchings.

We have also developed an ontology parser reported in [8], which can translate the OWL ontologies to our internal representation (Web-PDDL) of our ontology-based integration system. It facilitates the access and manipulation of the taxonomy information of the ontologies, and also extracts the instances from the input ontologies and prepares them as the input to the object reconciliation processor. Below we describe in detail about our object reconciliation and extended redescription mining algorithms.

3.1 Object Reconciliation

In order to perform the redescription mining algorithm to match ontologies, it is necessary to determine a universal set of data objects. To fulfil this objective, we must be able to compare ontology instances with each other and decide if they represent the same real world objects. Note that it is not guaranteed for the instances in different ontologies to be overlapping. For example, personnel ontologies used by different institutions may have totally different instances. Redescription mining is not directly suitable for matching tasks in such situations. Hence the existence of a set of equivalent instances will serve as the basic assumption in the proposed approach.

One possible approach to establish equivalence between two objects is to make use of a so-called *correspondence function* [9]: given two objects as input, the correspondence function determines the degree of equivalence between the two. However, it is very hard to design such a correspondence function since matchings between the ontologies that describe the data objects are yet unknown, thus compromising the comparison of objects in the elaborate level of granularity. We propose to address this problem by using machine learning techniques that implicitly incorporate ontology structure information and achieve satisfactory result with even very simple correspondence function.

Consider finding the overlapping instances in ontology O_1 and O_2 in Figure 2, specifically, the overlapping instances of the sets $\{t_1, t_2, t_3, t_4, t_5, t_6\}$ and $\{s_1, s_2, s_3, s_4, s_5, s_6\}$. Instead of applying the correspondence function directly to all instances in different ontologies as shown in Figure 3, we propose a way to partion instances into sub-groups and apply the correspondence function to the instances in the related sub-groups. We first train a learner based on the instances of O_1 with the target label being the classes in O_1, namely, X_1, X_2, X_3, X_4 and X_5; then we apply the learned model to predict the class labels of the instances in O_2. This process partitions the instances in O_2 according to O_1. Finally, the correspondence function is applied between all the pairs of instances in the groups that have the same class label, as shown in Figure 4, supposing s_3 and s_4 are predicted to belong to class X_4 in O_1 (the predicted class is denoted by X'_4 in the figure), and so forth.

In our implementation of the first test case described in detail in the experiment section, we used a simple string similarity measure as the correspondence function. Specifically, we treated the content of the instances in a bag-of-word representation. The classification process is essentially a text categorization process.

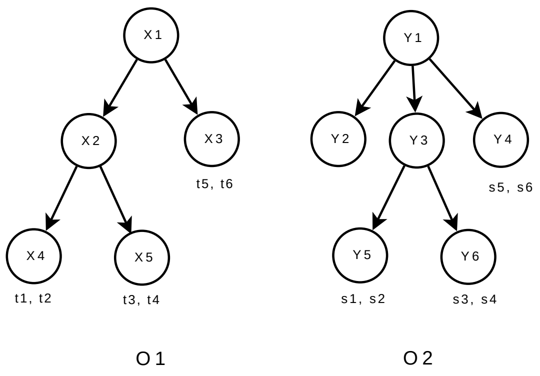

Figure 2: Two taxonomies of different ontologies.

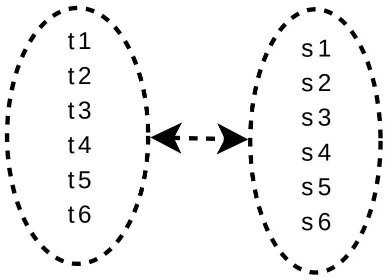

Figure 3: Applying the correspondence function directly to the instances of two ontologies.

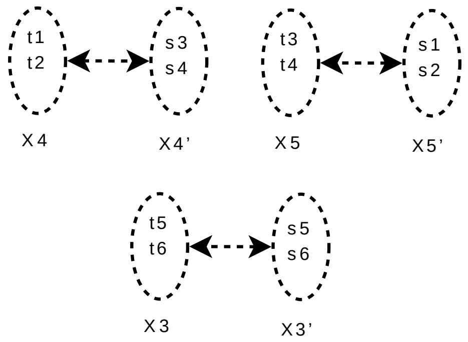

Figure 4: Applying the correspondence function to the related subsets of instances.

3.2 Incorporating Ontological Constraints and Heuristics in Redescription Mining

We extend the CARTwheels algorithm introduced in [26] to generate redescriptions for ontologies. The idea of CARTwheels is to search the space of possible set-theoretic expressions by growing two trees in opposite directions, so that they are matched at the leaves. The decision conditions in the first tree are based on set membership checks in entries from X and the bottom tree is based on membership checks in entries from Y, thus matchings of leaves corresponds to a potential redescription. The top tree is then removed and a new tree is grown to match the bottom tree in the similar manner. This process keeps iterating with new redescriptions mined along the way.

The search process is driven by maintaining entropy in the process of generating classification trees as opposed to traditional tree induction which is motivated at reducing entropy. The stop criterion for the search is a tunable parameter n in the algorithm that controls the number of unsuccessful consecutive alternations. It is possible that the CARTwheels stops alternations prematurely before some interesting redescriptions are found due to the improper value n or just takes too long to reach convergence before the interesting redescriptions are mined. In order to overcome this disadvantage, we propose to incorporate the semantic relationships between the given descriptors (i.e., classes in given ontologies) as heuristics to guide the exploration. We name the extended algorithm Onto-CART in this paper.

Examples of the heuristics that can be used in the approach include:

- H1: It's likely to find redescriptions at the parent level of the class nodes where redescriptions are found.
- H2: It's likely to find redescriptions among the siblings of the class nodes where redescriptions are found.
- H3: It's likely to find redescriptions at the child level of the class nodes where redescriptions are found.

Besides, we also introduce the heuristic for property matching after obtaining class level matchings:

- H*: Properties are likely to match if their classes are matched.

Suppose after applying the object reconciliation process to the ontologies O_1 and O_2 which have the taxonomies as depicted in Figure 2, we obtain the universal set $D = \{d_1,d_2,d_3,d_4\}$, as shown in Table 1.

object	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	class
d_1	√	×	×	√	×	√	X_1
d_2	×	×	√	×	√	×	X_4
d_3	×	√	×	×	√	×	X_3
d_4	√	×	√	×	√	√	X_5

Table 1: Datasets in initialization

Here, the set X corresponds to the set of descriptors (i.e., classes) $\{X_1,X_2,X_3,X_4,X_5\}$ from the O_1 ontology and Y corresponds to $\{Y_1,Y_2,Y_3,Y_4,Y_5, Y_6\}$ from O_2. We start to initialize the algorithm by preparing a dataset derived from X, Y and D for the classification tree induction. The details of how to generate the dataset can be found at [26]. The idea is to correspond the entries to the objects (i.e. value of a boolean feature are determined by whether the object is described by the descriptor); the boolean features are derived from one of X or Y, and the classes are derived from the other. Table 1 illustrates the the dataset for the first alternation in Onto-CART.

A classification tree can now be grown from the initial dataset. The paths of the tree induce partitions to the original dataset. We then prepare another dataset with X as the features and the partitions found as the classes. The second tree matches at the leaves with the first tree and produces redescriptions according to the corresponding partitions "read off" along the matched paths.

Algorithm 1 Alternation Process in Onto-CART Algorithm

```
Input: objects $D$, descriptor sets $\{X_i\}$,$\{Y_i\}$
Output: redescriptions $\mathcal{R}$
  Parameters and Initialization:
  $\theta$, $d$, $\rho$, $\eta$
  set answer set $\mathcal{R}$ = { }
  Alternation:
  while (count < $\eta$) do
    if $\mathcal{R}_{new}$!={ } then
      for each $r$ in $mathcalR_{new}$ do
        $\{D',X_i',Y_i'\}$=get_siblings($r$)
        Onto-CART($D',X_i',Y_i'$)
        $\{D',X_i',Y_i'\}$=get_parents($r$)
        Onto-CART($D',X_i',Y_i'$)
        $\{D',X_i',Y_i'\}$=get_children($r$)
        Onto-CART($D',X_i',Y_i'$)
      end for
    end if
    $\mathcal{G} = \{X_i\}$
    $\mathcal{F} = \mathcal{G}$
    if flag=false then
      $\{X_i\} = \mathcal{G}; \mathcal{G} = \{Y_i\}$
    else
      $Y_i = \{G\}; \mathcal{G} = \{X_i\}$
    end if
    $\mathcal{D}$=construct_dataset($\mathcal{O}, \mathcal{F}, \mathcal{C}$)
    $t$=construct_tree($\mathcal{D}$, $d$)
    if all leaves in $t$ have same class $c \in \mathcal{C}$ then
      set $l$ = random leaf in $t$ having non-zero entropy
      impurify($t$, $l$)
    end if
    $\mathcal{R}_{new} = eval(t, \theta)$
    if $\mathcal{R}_{new} = \{\}$ then
      count=count+1
    else
      count=0
      for each$c \in \mathcal{C}$ do
        if $c$ is involved in some $r \in \mathcal{R}\_new$
        $\mathcal{H}$=descriptors($c$)
        for each descriptor $g \in \mathcal{G} \cap \mathcal{H}$ do
          increase $g$'s class participation count
          if $g$'s class participation count > $\rho$
          remove $g$ from $\mathcal{G}$
        end for
      end for
    end if
    $\mathcal{G} = \mathcal{R} \cup \mathcal{R}_{new}; flag = \neg(flag)$
    $\mathcal{C}$=paths_to_classes(t)
  end while
```

Note that the classification tree is generated by maintaining entropy in some form, since impurity drives exploration. This is where we incorporate ontological heuristics to guide the exploration in order to avoid randomicity and unproductive termination.

Specifically, for each redescription r obtained from the last alternation, we construct the new dataset using X' and Y', which are subsets of X and Y that reside in the same level as the descriptors participated in r, together with the objects D' described by X' and Y'.

We then recursively invoke Onto-CART with X', Y' and D' being the input. Similarly, we build datasets for the descriptors in the parent, children class and property levels of the descriptors in r and perform Onto-CART recursively on them. The pseudo-code of the alternation process of Onto-CART is given in Algorithm 1, where θ is the Jaccard's coefficient; d is the depth of trees; ρ is the number of class participation allowed; and η is the max number of consecutive unsuccessful alternation. Functions such as *construct_tree, impurify, and paths_to_classes* and the initialization process of the algorithm are described in detail in [26].

4. EXPERIMENT

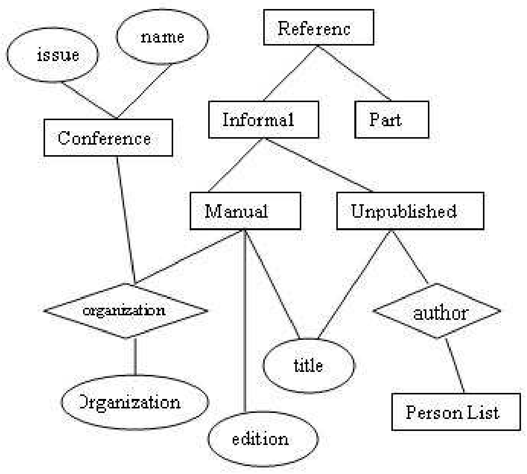

Figure 5: Segment of Bibliographic Ontology 101.

We first evaluate our system on the synthetic ontologies (ontology 101 and 201) from the EON ontology alignment test dataset [2]. both of them are equipped with data instances. Ontology 101 is the baseline ontology for the domain of Bibliographic references. Ontology 201 is a mirror ontology that has exactly the same structure as 101 but with each label or identifier's name (name of classes and properties) replaced by a pseudo one (random string). So a matcher based only on the naming similarity at the meta-data level will not work at all in this scenario. Figure 5 shows a part of the bibliography ontology 101.

This test case contains only the naming conflict heterogeneity. Our system can still detect it by establishing one-to-one matchings among the concepts.

The object reconciliation processor uses an SVM learner and produces all 53 correct reconciliation decisions. The reason it achieves such high accuracy is that the instances of ontology 201 highly resemble those of ontology 101 in a bag-of-word representation. The Onto-CART generate 26 matchings. Together with the matchings that are inferred based on Onto-CART's decision using domain-independent constraints such as "two class nodes match if their children also match" and "two property nodes match if the classes or the characteristics of datatype they link also match," the total number of generated matching sums up to 70. There are actually 91 mappings specified manually by human. The reason why our system did not produce the complete matching result is due to the lack of training data; some of the classes have no instances at all.

The second test is conducted on the People Ontology [3] from UMD and the Person & Employee Ontology [4] from CMU. Figure 6 shows part of the two ontologies. We crafted the ontologies a bit by adding a "College" class with one "Course" property as the subclass of "Organization" to the UMD ontology (on the left side), and add a "University" class with two properties "Undergraduate_course" and "Graduate_course" to the CMU ontology (on the right side). We manually generated instances for the ontologies. Our system produced 9 matchings as depicted in dotted line in Figure 6 including one complex matching represented by the following redescription:

$$Course \leftrightsquigarrow Undergraduate\_course \cup Graduate\_course$$

This can be also represented in the following FOL form as mapping rules:

$\forall x\ Course(x) \rightarrow Undergraduate\_course(x);$
$\vee\ Graduate\_course(x)$
$\forall x\ Graduate\_course(x) \rightarrow Course(x);$
$\forall x\ Undergraduate\_course(x) \rightarrow Course(x).$

To our knowledge, very few of the existing matching system can automatically generate such kind of matchings involving equivalence relationship between union of concepts. This matching result also shows that it successfully captures the specific heterogeneity that we propose to study.

5. CONCLUSION AND RESEARCH DIRECTIONS

We present in this paper an automatic and considerably accurate framework for discovering complex ontology matching in order to understand heterogeneity problem. The heterogeneity problem is essential in both distributed data mining and data integration research communities. We also try to formally represent the discovered heterogeneities in formal language, which we believe to be reusable by other DDM or data integration systems.

Our main contributions are:

- An attempt to represent heterogeneity in formal language. Our approach is capable to discover and formally represent complex many-to-many matchings, especially unions and intersections, thus being able to capture more complex semantic heterogeneities while most of the other works focus on finding one-to-one or group-to-group matchings.

- We performed a case study on detecting a specific class of heterogeneity in ontologies–the one that can be discovered by only data-driven approaches.

- We extended the redescription algorithm, Onto-CART, by incorporating the use of ontology structure information. Various constraints and heuristics involving semantic relationships are explored.

- The exploration of machine learning techniques for object reconciliation by discovering overlapping instances in differ-

[2]http://oaei.ontologymatching.org/2004/Contest/#101

[3]http://www.cs.umd.edu/projects/plus/DAML/onts/personal1.0.daml

[4]http://www.daml.ri.cmu.edu/ont/homework/atlas-cmu.daml

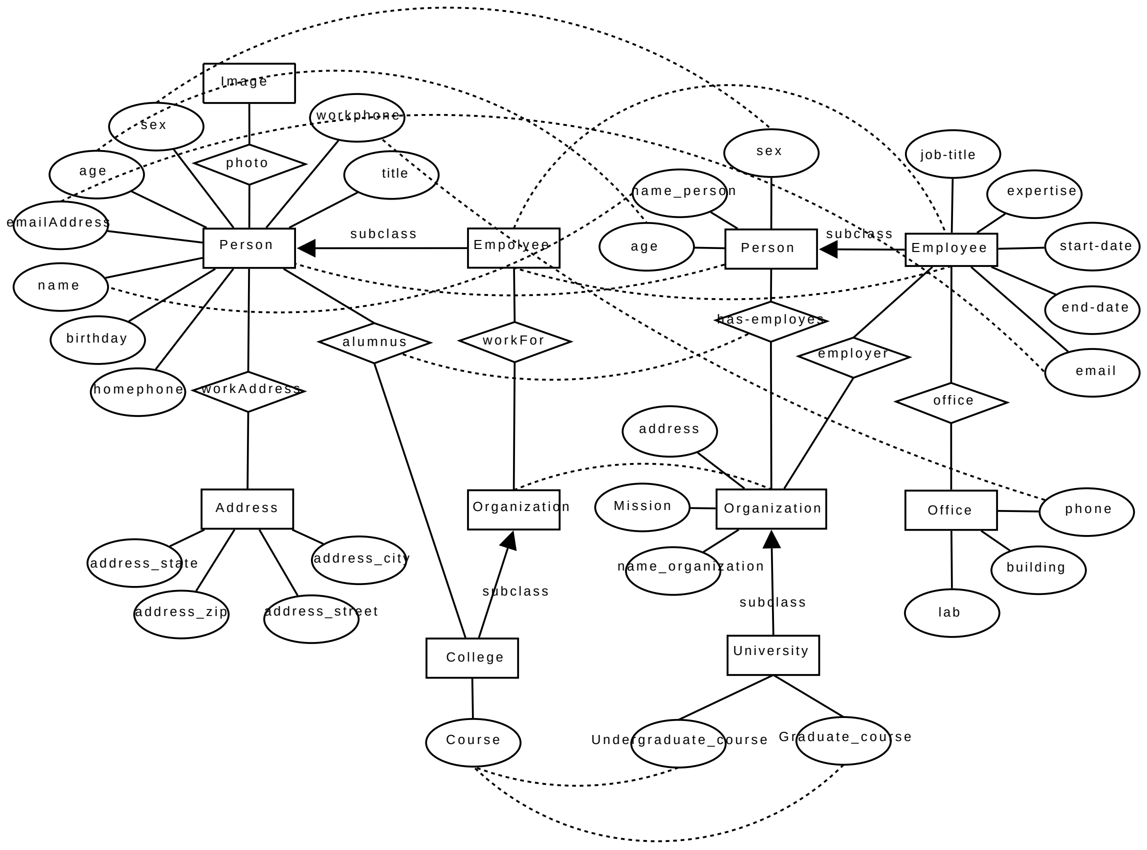

Figure 6: UMD and CMU person ontologies.

ent ontologies, and thus enabling the construction of a universal reconciled dataset for further mining algorithms.

In the future work, we plan to extensively explore the spectrum of heterogeneities with more real-world data from different domains stored in relational databases, XML documents or spreadsheets. We will further study the formal methodology to represent and categorize data heterogeneities and continue to develop algorithms to automatically resolve heterogeneity. We also plan to discover ontology matchings by combining the conceptual level and instance level approach in an effective and efficient manner. In our future work, there are mainly several challenges that need to be addressed:

- One big difficulty for the ontology matching system that adopts instance level approach is the orthogonality of data. If there are no overlapping instances supplied by the ontologies under consideration, it will be very hard to perform mining algorithms. What are the more intrinsic characteristics of data and how to study them effectively deserve further study.

- Not all data heterogeneities can be represented in formal language as mappings or the expressions with set theories. On the other hand, even with sophisticated design, it is hard to get 100% accurate mappings without human involvement. Each mapping may have some associated probabilistic values. Similar to what we have reported in [25], the mapping rules discovered by inductive logic programming have accuracy for each rule.

- Although we argue that formal mappings rules can be used by DDM or data integration systems because the rules are machine processable, it is an open question whether existing DDM or data integration systems can process all discovered data heterogeneities. We may need to design different DDM or data integration algorithms for different kinds of heterogeneities.

- Besides the existing repositories that contain heterogeneous data, it is hard to find appropriate datasets for the purpose of testing and benchmarking the performance of systems that designed to detect and resolve heterogeneity. Most of the repositories do not have golden standards for heterogeneity resolution.

  On the other hand, the existing data repository (e.g., our collected data[5]) has only limited resources so that only a subset of heterogeneities is exhibited in the data. For some specific study in DDM or data integration, researchers may require different kinds of combinations of heterogeneities, which is not easily satisfied by any existing datasets. A benchmark system with the capability to automatic generate synthetic data according to user's demand of certain heterogeneities will be very helpful.

6. ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their insightful comments and helpful suggestion for the improvements of our work.

[5]http://aimlab.cs.uoregon.edu/benchmark

7. REFERENCES

[1] OWL Web Ontology Language. http://www.w3.org/TR/owl-ref/.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5), May 2001.

[3] J. D. Bruijn and A. Polleres. Towards an Ontology Mapping Specification Language for the Semantic Web. Technical report, Digital Enterprise Research Institute, June 2004.

[4] D. Caragea, J. Z. 0002, J. Bao, J. Pathak, and V. Honavar. Algorithms and software for collaborative discovery from autonomous, semantically heterogeneous, distributed information sources. In *ALT*, pages 13–44, 2005.

[5] R. Dhamankar, Y. Lee, A. Doan, A. Y. Halevy, and P. Domingos. iMAP: Discovering Complex Mappings between Database Schemas. In *Proceedings of the ACM Conference on Management of Data*, pages 383–394, 2004.

[6] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Learning to Map Between Ontologies on the Semantic Web. In *International World Wide Web Conferences (WWW)*, pages 662–673, 2002.

[7] X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD*, pages 85–96, 2005.

[8] D. Dou, D. V. McDermott, and P. Qi. Ontology Translation on the Semantic Web. *Journal of Data Semantics*, 2:35–57, 2005.

[9] D. Fang, J. Hammer, and D. McLeod. The identification and resolution of semantic heterogeneity in multidatabase systems, 1994.

[10] T. Gene Ontology Consortium. Creating the Gene Ontology Resource: Design and Implementation. *Genome Research*, 11(8):1425–1433, 2001.

[11] C. H. Goh. *Representing and reasoning about semantic conflicts in heterogeneous information systems*. PhD thesis, 1997. Supervisor-Stuart E. Madnick.

[12] J. Hammer, M. Stonebraker, and O. Topsakal. Thalia: Test harness for the assessment of legacy information integration approaches. In *ICDE*, pages 485–486, 2005.

[13] W. Hu and Y. Qu. Block matching for ontologies. In *Proc. of 5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006, LNCS 4273*, 2006.

[14] W. Kim and J. Seo. Classifying schematic and data heterogeneity in multidatabase systems. *Computer*, 24(12):12–18, 1991.

[15] W. Kim and J. Seo. Classifying schematic and data heterogeneity in multidatabase systems. *Computer*, 24(12):12–18, 1991.

[16] D. Lindberg, B. Humphries, and A. McCray. The Unified Medical Language System. *Methods of Information in Medicine*, 32(4):281–291, 1993.

[17] L. Lubyte and S. Tessaris. Extracting ontologies from relational databases. In *Description Logics*, 2007.

[18] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. The Chimaera Ontology Environment. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1123–1124, 2000.

[19] B. Motik, I. Horrocks, and U. Sattler. Bridging the gap between owl and relational databases. In *WWW*, pages 807–816, 2007.

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

[21] N. F. Noy and M. A. Musen. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In *Proceedings of the National Conference on Artificial Intelligence*, pages 450–455, 2000.

[22] L. Parida and N. Ramakrishnan. Redescription mining: Structure theory and algorithms. In *AAAI*, pages 837–844, 2005.

[23] B. Park and H. Kargupta. Distributed data mining: Algorithms, systems, and applications, 2002.

[24] C. Pluempitiwiriyawej and J. Hammer. *A Classification Scheme for Semantic and Schematic Heterogeneities in XML Data Sources*. University of Florida, Gainesville, FL, technical report tr000-004 edition, September 2000.

[25] H. Qin, D. Dou, and P. LePendu. Discovering Executable Semantic Mappings Between Ontologies. In *Proceedings of the International Conference on Ontologies, Databases and Application of Semantics (ODBASE)*, pages 832–849, 2007.

[26] N. Ramakrishnan, D. Kumar, B. Mishra, M. Potts, and R. F. Helm. Turning cartwheels: an alternating algorithm for mining redescriptions. In *KDD*, pages 266–275, 2004.

[27] L. Stojanovic, N. Stojanovic, and R. Volz. Migrating data-intensive web sites into the semantic web. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, pages 1100–1107, New York, NY, USA, 2002. ACM Press.

[28] H. Wache, T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hubner. Ontology-based integration of information: A survey of existing approaches. In *IJCAI–01 Workshop: Ontologies and Information Sharing*, pages 108–117, 2001.

[29] M. J. Zaki and N. Ramakrishnan. Reasoning about sets using redescription mining. In *KDD*, pages 364–373, 2005.