

LEARNING ONTOLOGY FROM RELATIONAL DATABASE

MAN LI, XIAO-YONG DU, SHAN WANG

School of Information, Renmin University of China, Beijing 100872, China
E-MAIL: limanl@ruc.edu.cn

Abstract:

Ontology provides a shared and reusable piece of knowledge about a specific domain, and has been applied in many fields, such as Semantic Web, e-commerce and information retrieval, etc. However, building ontology by hand is a very hard and error-prone task. Learning ontology from existing resources is a good solution. Because relational database is widely used for storing data and OWL is the latest standard recommended by W3C, this paper proposes an approach of learning OWL ontology from data in relational database. Compared with existing methods, the approach can acquire ontology from relational database automatically by using a group of learning rules instead of using a middle model. In addition, it can obtain OWL ontology, including the classes, properties, properties characteristics, cardinality and instances, while none of existing methods can acquire all of them. The proposed learning rules have been proven to be correct by practice.

Keywords:

Ontology; Ontology Learning; OWL; Relational Database; Relational Model

1. Introduction

Ontology provides a shared and reusable piece of knowledge about a specific domain, and has been applied in many fields, such as Semantic Web, e-commerce and information retrieval, etc. More and more researchers begin to pay attention to ontology research. Until now, many ontology editors have been developed to help domain experts to develop and manage ontology. However, the construction of ontology is still time-consuming and labor-intensive even by the aid of ontology editor. It has been recognized that ontology building by hand is a very hard job and becomes the bottleneck of ontology acquisition. So ontology learning, the goal of which is to construct ontology automatically or semi-automatically by machine learning, is an important topic for ontology research.

Due to the wide use of relational database in information management, a large amount of data about various domains are organized and stored in relational

database. Data in relational database may be used as a kind of important resource for ontology learning.

OWL^[1] is the latest standard recommended by W3C. It facilitates greater machine interpretability of web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics. So the paper proposes an automatic ontology learning approach, which acquires OWL ontology based on data in relational database. Compared with existing methods, the approach can acquire ontology from relational database automatically by using a group of learning rules instead of using a middle model. In addition, it can obtain OWL ontology, including the classes, properties, properties characteristics, cardinality and instances, while none of existing methods can acquire all of them. The approach has been applied in the Semantic Web project of Renmin University of China and is proven to be practical.

The paper is organized as follows. Section 2 gives some preliminary definitions about ontology and relational database. Section 3 describes the learning rules from relational database to OWL ontology in detail. Section 4 introduces the implementation based on the approach. Section 5 analyzes current related works. Section 6 draws a conclusion and gives future works.

2. Related works

Much work has been addressed on the issue of explicitly defining semantics in database schemas^[2,3], extracting semantics out of database schema^[4] and transforming a relational model into an object-oriented model^[5]. However, the semantics obtained by previous cannot meet the requirement of constructing ontology fully. Although object-oriented model is close to an ontological theory, there are still some differences between them. For example, there does not exist hierarchies and restrictions about properties in object-oriented model. So these methods cannot be used to learn ontology from relational database directly.

There are few approaches investigating the

transformation of a relational model into an ontological model. The method in [6] creates a new database schema by analyzing the database schema firstly. Then the content of the new database schema is mapping into the ontology by reverse engineering techniques. During the transformation, the approach uses a new database schema as middle model, which is different from ours. Moreover, the semantic characteristics of the database schema are not always analyzed. The method in [7] is proposed to automate the process of filling the instances and their attributes' values of an ontology using the data extracted from external relational sources. This method uses a declarative interface between the ontology and the data source, modeled in the ontology and implemented in XML schema. However, our approach acquires ontological instances by using rules directly. The method in [8] is to build light ontologies from conceptual database schemas. It uses a mapping process, which is similar to our approach. However, its target is to construct RDF(S) ontology. RDF(S) has unclear semantics and has not inference model, which is important for automatic tasks. Our learning target is OWL ontology because OWL has clear semantics bringing by description logic systems, and OWL enjoys a well-founded inference model from some particular description logics. All the previous approaches cannot learn OWL ontology from relational database.

3. Preliminary definitions

There are many versions of ontology definition, and among them a popular definition is: "an ontology is an explicit, formal specification of a shared conceptualization of a domain of interest"^[9].

Definition 1. Ontological structure is a 5-tuple $O = \{C, R, H^c, rel, A^o\}$, where C is a finite set of concepts; R is a finite set of relations; H^c is called concept hierarchy or taxonomy, which is a directed relation $H^c \subseteq C \times C$, for example, $H^c(C_1, C_2)$ specifies that C_1 is a subconcept of C_2 ; rel relates concepts non-taxonomically, for example, $rel(R) = (C_1, C_2)$ specifies that C_1 and C_2 have relation R ; A^o is a set of axioms, which is expressed in an appropriate logical language, e.g. first order logic.

Based on the ontological structure, ontology comprises a set of instances, which could be seen as the extension of concepts.

Now there are many ontology description languages. This paper adopts the latest standard recommended by W3C, OWL, as the ontology description language. OWL is designed for use by applications that need to process the content of information instead of just presenting

information to humans. OWL facilitates greater machine interpretability of web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics.

Most of the elements of an OWL ontology concern *classes, properties* and *instances*. The most basic concepts in a domain should correspond to classes that are the roots of various taxonomic trees. Properties are used to assert general facts about the members of classes and specific facts about instances. A property is a binary relation, where the domain and range can be specified. OWL distinguishes two types of properties: datatype properties and object properties. Datatype properties are relations between instances of classes to datatypes. Object properties are relations between instances of two classes. In addition, property characteristics and cardinality are used to further specify properties. Both properties and classes can be arranged in a hierarchy. The members of a class are known as instances of the class.

The underlying model of relational database is the relational model^[10], in which each relation R_i is corresponding to a table. The columns of a relation are called attributes denoted as A . Here we define some functions. The function $attr(R_i)$ acquires the attributes contained in a specific relation R_i . The function $dom(A_i)$ acquires the value's range of attribute A_i , $A_i \in A$. The function $pkey(R_i)$ acquires the *primary keys* in a given relation R_i , thus $pkey(R_i) \subseteq attr(R_i)$ must hold. The function $fkey(R_i)$ acquires the *foreign keys* in a given relation R_i , thus $fkey(R_i) \subseteq attr(R_i)$ must hold.

In additions, there is dependency relationship between the data of attributes in relational model. The related definitions is as following.

Definition 2. For relations R_i and R_j in database, supposed that $A_i \subseteq attr(R_i)$ and $A_j \subseteq attr(R_j)$, $t_i(A_i)$ expresses the values of tuple t_i in attributes A_i , and $t_j(A_j)$ expresses the values of tuple t_j in attributes A_j . For each $t_i(A_i)$ in R_i , if in R_j there exists $t_i(A_i) = t_j(A_j)$, A_i and A_j are called inclusion dependency, denoted as $R_i(A_i) \subseteq R_j(A_j)$.

Definition 3. For relations R_i and R_j in database, supposed that $A_i \subseteq attr(R_i)$ and $A_j \subseteq attr(R_j)$, if there exist $R_i(A_i) \subseteq R_j(A_j)$ and $R_j(A_j) \subseteq R_i(A_i)$, A_i and A_j are called equivalence, denoted as $R_i(A_i) = R_j(A_j)$.

Besides the above entities, relational model includes some further constraints, such as NOT NULL, UNIQUE etc. All these constitute *relation schema*, which is used to describe the structure and association of data. The tuples in relations reflect the values of schema, and they are content of database.

4. Learning OWL ontology from relational database

The paper assumes that the relational schema is at least in third normal form (3NF). The assumption does not limit the applicability of our method, because most of databases are highly likely to satisfy the 3NF requirement and there is algorithm that can convert 1NF to 3NF easily^[11]. During the conversion, we try to preserve all semantics specified in the original relation definitions in database.

In this paper, learning OWL ontology from relational database strongly depends on a group of learning rules. According to the target of learning, the rules are organized in five groups: rules for learning classes, rules for learning properties, rules for learning hierarchy, rules for learning cardinality and rules for learning instances. They are introduced below.

4.1. Rules for learning classes

During learning ontological classes, one class may derive from the information spreading across several relations (see Rule 1) or from just one relation in database (see Rule 2).

Rule 1. For relations R_1, R_2, \dots, R_i in database, supposed that $P_1 = pkey(R_1), P_2 = pkey(R_2), \dots, P_i = pkey(R_i)$ if $R_1(P_1) = R_2(P_2) = \dots = R_i(P_i)$, then the information spread across R_1, R_2, \dots, R_i should be integrated into a ontological class C_i .

Rule 1 integrates the information in several relations into one ontological class, when these relations are used to describe one entity.

Rule 2. An ontological class C_i can be created based on the relation R_i , if there does not exist a relation that can be integrated with R_i , that is to say, Rule 1 cannot be satisfied, and one of the following conditions can be satisfied:

- (i) $|pkey(R_i)| = 1$;
- (ii) $|pkey(R_i)| > 1$, and there exists A_i , where $A_i \in pkey(R_i)$ and $A_i \notin fkey(R_i)$.

Rule 2 indicates that if the relation R_i is used to describe an entity, instead of relationship between relations, then it can be mapped into one ontological class.

Example 1. A given relational database schema is shown in Table 1, which comprises three columns: Relations, Primary Key and Foreign Key. The first column is the definition of relation name and attributes, the second column indicates the primary keys of each relation, and the last column gives the foreign keys of each relation and their reference relations.

Table 1. A Relational Database Schema

Relation	Primary Key	Foreign Key
Student (StuID int, StuName string, Hometown int)	StuID	Hometown referring to CityID in City
PhdStudent(StuID int, Year data)	StuID	StuID referring to StuID in Student
City(CityID int, CityName string)	CityID	N
Department(DepID int, DepName string)	DepID	N
Study(StuID int, DepID int)	StuID, DepID	StuID referring to StuID in Student, DepID referring to DepID in Department
Staff(LectID int, Room int)	LectID	N
Staff-Default(LectID int, Address string)	LectID	LectID referring to LectID in Staff

According Rule 1, the information in relations *Staff* and *Staff-Default* should be integrated to one ontological class *Staff*, because $Staff(LectID) = Staff-Default(LectID)$. Consequently, the class *Staff* is created in OWL ontology, which is described as `<owl:Class rdf:ID=" Staff "/>`.

According Rule 2, ontological Classes *Student*, *PhdStudent*, *City* and *Department* can be created based on the relations *Student*, *PhdStudent*, *City* and *Department* in database, because Rule 1 cannot be applied and the number of primary key in these relations is one. Consequently, in OWL ontology four classes can be created as followings.

```
<owl:Class rdf:ID=" Student "/>
<owl:Class rdf:ID=" PhdStudent "/>
<owl:Class rdf:ID=" City "/>
<owl:Class rdf:ID=" Department "/>
```

4.2. Rules for learning properties and property characteristics

In OWL, there are two kinds of properties: object properties and datatype properties.

Rule 3, Rule 4, Rule 5 and Rule 6 are the rules for learning object properties.

Rule 3. For relations R_i and R_j , if $R_i(A_i) \subseteq R_j(A_j)$ and $A_i \notin pkey(R_i)$ are satisfied, then an object property P can be created based on A_i . Suppose that the classes corresponding to R_i and R_j are C_i and C_j respectively, the domain and range of P are C_i and C_j .

Rule 4. For relations R_i and R_j , two ontological objects

property “has-part” and “is-part-of” can be created, if the two conditions are satisfied:

- (i) $|pkey(R_i)| > 1$;
- (ii) $fkey(R_i) \subset pkey(R_i)$, where $fkey(R_i)$ referring to R_j .

Suppose that the classes corresponding to R_i and R_j are C_i and C_j respectively, the domain and range of “is-part-of” are C_i and C_j and the domain and range of “has-part” are C_j and C_i . Properties “has-part” and “is-part-of” are two inverse properties.

Rule 3 and Rule 4 learn object properties according to the reference relationship between relations. In relational model, a relation may be used to indicate the relationship between two relations; in this case, such relation can be mapped into an ontological object property, which is shown in Rule 5.

Rule 5. For relations R_i , R_j and R_k , if $A_i = pkey(R_i)$, $A_j = pkey(R_j)$, $A_i \cup A_j = fkey(R_k)$ and $A_i \cap A_j = \emptyset$, then two object properties P_j^i and P_j^j can be created based on the semantics of R_k . Suppose that the classes corresponding to R_i and R_j are C_i and C_j respectively, the domain and range of P_j^i are C_i and C_j and the domain and range of P_j^j are C_j and C_i . P_j^i and P_j^j are two inverse properties.

The Rule 3, Rule 4 and Rule 5 learn object properties by using the binary relationship between relations. In fact, relational model can also provide non-binary relationships representing relationships of higher degrees (n-ary), but OWL does not provide means for n-ary relations. Therefore complex relations need to be decomposed into a group of binary relations. Extending Rule 5, Rule 6 can be defined for n-ary relations.

Rule 6. For relation R_1, R_2, \dots, R_i and R_j , if $A_1 = pkey(R_1)$, $A_2 = pkey(R_2), \dots, A_i = pkey(R_i)$, $A_1 \cup A_2 \cup \dots \cup A_i = fkey(R_j)$ and $A_1 \cap A_2 \cap \dots \cap A_i = \emptyset$, then object properties $P_j^1, P_j^2, \dots, P_j^{i*(i-1)}$ can be created. The semantics of $P_j^1, P_j^2, \dots, P_j^{i*(i-1)}$ are based on the decomposition of n-ary relationship provided by R_j .

Example 2. For the given database schema in Table 1, ontological properties can be acquired.

According to Rule 3, *Hometown* can be an object property. The domain and range of it are *Student* and *City* respectively. So in OWL ontology, there will be the following descriptions.

```
<owl:ObjectProperty rdf:ID="Hometown">
  <rdfs:domain rdf:resource="# Student"/>
  <rdfs:range rdf:resource="# City"/>
</owl:ObjectProperty>
```

According to Rule 5, two object properties can be created based on the semantic of relation *Study*. We define them as *belong-to* and *has-student*, which is used to associate the ontological Classes *Student* and *Department*.

The corresponding OWL descriptions is as following.

```
<owl:ObjectProperty rdf:ID="belong-to">
  <rdfs:domain rdf:resource="# Student"/>
  <rdfs:range rdf:resource="# Department"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="has-student">
  <rdfs:domain rdf:resource="# Department"/>
  <rdfs:range rdf:resource="# Student"/>
  <owl:inverseOf rdf:resource="# belong-to"/>
</owl:ObjectProperty>
```

The rule for learning datatype properties is shown in Rule 7.

Rule 7. For an ontological class C_i and the datatype properties set of C_i denoted as $DP(C_i)$, if C_i is corresponding to relations R_1, R_2, \dots, R_i in database, then for every attribute in R_1, R_2, \dots, R_i , if it cannot be used to create object property by using Rule 3, then it can be used to create datatype property of C_i . The domain and range of each property P_i are C_i and $dom(A_i)$ respectively, where $P_i \in DP(C_i)$ and $A_i \in attr(R_i)$.

Rule 7 shows that for each attribute in relations, if it cannot be converted into ontological object property, it can be converted into ontological datatype property.

Example 3. For the given database schema in Table 1, according to Rule 7, the attributes *StuID*, *StuName*, *Year*, *CityID*, *CityName*, *DepID*, *DepName*, *LectID*, *Room* and *Address* should be datatype properties of corresponding ontological classes. The related OWL segments are as following.

```
<owl:DatatypeProperty rdf:ID="StuID">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="# Student"/>
        <owl:Class rdf:about="# PhdStudent"/>
        <owl:Class rdf:about="# StuID"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="&xsd:int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="StuName">
  <rdfs:domain rdf:resource="# Student"/>
  <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>
...
```

4.3. Rules for learning hierarchy

In OWL, the classes and properties can be organized in a hierarchy. We believe that if two relations in database

have inheritance relationship, then the two corresponding ontological class or property can be organized in a hierarchy, which is shown in Rule 8.

Rule 8. For relation R_i and R_j , supposed that $P_i = pkey(R_i)$, $P_j = pkey(R_j)$, if Rule 1 does not applied and $R_i(P_i) \subseteq R_j(P_j)$ is satisfied, then the class/property corresponding to R_i is a subclass/subproperty of the class/property corresponding to R_j .

Example 4. For the given database schema in Table 1, ontological properties can be acquired. According Rule 6, the ontological Class PhdStudent is a subclass of Student. In OWL ontology, there will be the following descriptions.

```
<owl:Class rdf:ID=" PhdStudent ">
  <rdfs:subClassOf rdf:resource="# Student " />
</owl:Class>
```

4.4. Rules for learning cardinality

OWL defines property cardinality to further specify properties. The property cardinality can be learned from the constraint of attributes in relations. The rules for learning property cardinality are shown in Rule 9, Rule 10, and Rule 11.

Rule 9. For relation R_i and $A_i \in attr(R_i)$, if $A_i = pkey(R_i)$ or $A_i = fkey(R_i)$, then the minCardinality and maxCardinality of the property P_i corresponding to A_i is 1.

Rule 10. For relation R_i , and $A_i \in attr(R_i)$, if A_i is declared as NOT NULL, the minCardinality of the property P_i corresponding to A_i is 1.

Rule 11. For relation R_i , and $A_i \in attr(R_i)$, if A_i is declared as UNIQUE, the maxCardinality of the property P_i corresponding to A_i is 1.

Example 5. For the given database schema in Table 1, according Rule 9, the minCardinality and maxCardinality of attributes StuID, DepID and LectID corresponding to their own domain is 1. The related OWL segments are as the following.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="# StuID"/>
  <owl:minCardinality">1</owl:minCardinality>
  <owl:maxCardinality">1</owl:maxCardinality>
</owl:Restriction>
<owl:Restriction>
  <owl:onProperty rdf:resource="# DepID"/>
  <owl:minCardinality">1</owl:minCardinality>
  <owl:maxCardinality">1</owl:maxCardinality>
</owl:Restriction>
...
```

4.5. Rules for learning instances

For an ontological class C_i , its instances consist of tuples in relations corresponding to C_i and relations between instances are established using the information contained in the foreign keys in the database tuples. The rule is shown in Rule 12.

Rule 12. For a ontological class C_i , if C_i is corresponding to relations R_1, R_2, \dots, R_i in database, then every tuple $t_i, t_i \in R_1 \bowtie R_2 \bowtie \dots \bowtie R_i$, can be a instance of C_i .

According to the Rule 1 to Rule 12, OWL ontology can be constructed based on the data in relational database automatically.

5. Implementation

The overall framework of our ontology learning is presented in Figure 1. The input for the framework is data stored in relational database. The framework uses a database analyzer to extract schema information from database, such as the primary keys, foreign keys and dependencies, etc. Then the obtained information is transferred to ontology generator. The ontology generator will generate ontology based on the schema information and rules. At last, user can modify and refine the obtained ontology by the aid of ontology reasoner and ontology editor. The frame is a domain/application independent and can learn ontologies for general or specific domains from relational database.

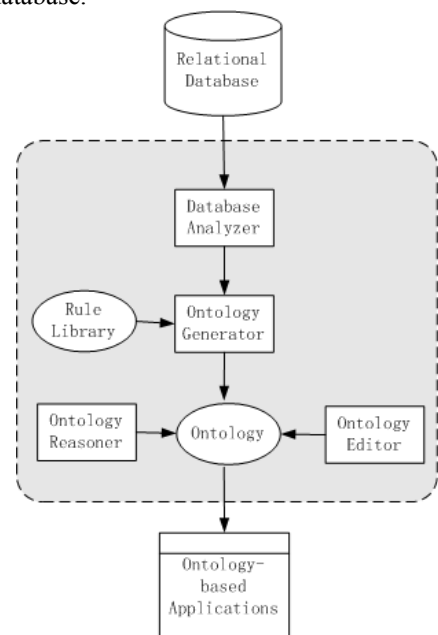


Figure 1. Ontology Learning Framework

Now we have applied the framework and learning rules introduced in this paper to the Semantic Web project in Renmin University. Our goal is to acquire OWL ontology automatically from existing digital library of Renmin University, the huge resources of which is stored in relational database. The digital library comprises about thirty thousand resources on economics and we choose them as the source of ontology learning. The result of ontology learning comprise about twenty classes, four hundred properties and thirty thousand instances. The classes and properties are organized in hierarchy. The non-taxonomical relations of classes are embodied by object properties.

6. Conclusion and future works

Ontology plays an important role for those knowledge-intensive applications as a source of formally defined terms for concise communication. Research on ontology is becoming increasingly widespread in the computer science community. The major difficulties in building ontology are a mass of handwork. So the automatic ontology construction by ontology learning is a good solution.

In this paper, an ontology learning approach is proposed to construct OWL ontology automatically based on data in relational database. The related learning rules are discussed in detail. It can be seen that the approach is practical and helpful to the automation of ontology building. In the future, we plan to integrate other learning techniques into our approach to obtain better learning result.

Acknowledgements

The work was supported by the National Natural Science Foundation of China (Grant No. 60496325) and "211" project.

References

- [1] OWL, Web Ontology Language (OWL), <http://www.w3.org/2004/OWL/>, 2004.
- [2] Rishé N. Database design: the semantic modeling approach. McGraw-Hill, 1992.
- [3] Biskup J. Achievements of relational database schema design theory revisited. *Semantics in Database*, LCNS 1358, Springer Verlag, 1998.
- [4] Chiang R, Barron T, Storey V. Reverse engineering of relational databases: extraction of an EER model from a relational database. *Journal of data and knowledge engineering*, 1994, 12(2):107-142.
- [5] Vermeer M, Apers P. Object-oriented views of relational databases incorporating behaviour. DASFAA, 1995.
- [6] Kashyap V. Design and creation of ontologies for environmental information retrieval. *Proc. of the 12th Workshop on Knowledge Acquisition, Modeling and Management*. Alberta, Canada. 1999.
- [7] Rubin DL, Hewett M, Oliver DE, Klein TE and Altman RB. Automatic data acquisition into ontologies from pharmacogenetics relational data sources using declarative object definitions and XML. *Proc. of the Pacific Symposium on Biology*. 2002.
- [8] Stojanovic L, Stojanovic N, Volz R. Migrating data-intensive Web Sites into the Semantic Web. *Proc. of the 17th ACM symposium on applied computing*. ACM press, 2002. 1100-1107.
- [9] Gruber TR. A translation approach to portable ontology specifications. *Technical Report KSL 92-71*. Knowledge System Laboratory. 1993.
- [10] Codd EF. A relational model of data for large shared data banks. *CACM* 13 No.6, 1970.
- [11] Salzberg B. Third normal form made easy. *Sigmoid Record*, 1986, 15(4):2~18.