Automatically building database from biomedical ontology

Nakhla Zina and Nouira Kaouther

BESTMOD Laboratory, High Institute of Management, University of Tunis,
41, Rue de la libert, Bouchoucha, 2000 Bardo, Tunis, Tunisia

Abstract. Ontologies give a standard presentation of specific domain knowledge in order to establish either a Knowledge Base (in the case of a moderate amount of data) or an Ontology Based Data Base (in the case of a huge amount of data).The aim of this paper is to present a new methodology to build data base automatically from a biomedical ontology.

Keywords: Ontology, Biomedical Ontology, Ontology Based Database.

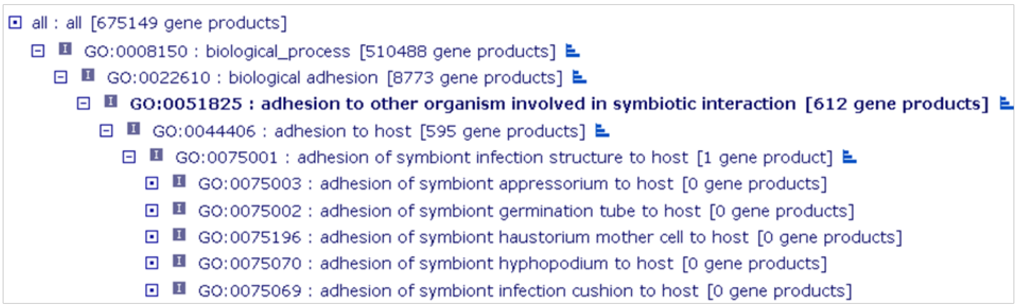
1 Introduction

As ChandraseKaran et al. in [5] "Theories in artifitiel intelligence fall into two broad categories: mechanism theories and content theories". The term mechanism includes rule systems, machine learning algorithms,... However, in some times we can't implement a mechnism without a suitable description of the domain.

Knowing that ontologies enumerate and give specific and semantic description of domain concepts, define their attributes and various relationships among them [16]. We can conclude that ontologies are typically content theories [5]. But ontology of the domain is not a goal in itself [4]. The target is to standardize the knowledge representation of a specific domain and to allow applications to use standard structured data. Several approaches and systems were proposed to store ontology together with individul instances of classes or concepts so called knowledge base [4]. But for applications manipulating a large amount of ontology based data, query performance becomes a new issue [8]. So ontology based databases (OBDB) are required. Some trials have been done to construct OBDB [2]:

1. Model based on storing ontologies elements such as concepts and properties in a single table with three columns (subject, predicate, object). This triplet used to characterize each concept by a name, a comment and its superconcept [15].
2. Model based on a table for concept model and another table to store instances [14],[3].
3. Model depends on ontology structure called Ontology DataBase (OntoDB). It uses table to represent each concept [1], [8].

These approaches are limited in the ontology type. Also, for the first model, it has no distinction between instances and concepts. For the second model query execution can engender a long delays.
In this paper, we propose to present our own process to construct OBDB. We build on our experience using Gene ontology, which is a biomedical ontology and provides terms for ”declaring molecular functions, biological process and cellular components of gene products” [17].



```
all : all [675149 gene products]
  GO:0008150 : biological_process [510488 gene products]
    GO:0022610 : biological adhesion [8773 gene products]
      GO:0051825 : adhesion to other organism involved in symbiotic interaction [612 gene products]
        GO:0044406 : adhesion to host [595 gene products]
          GO:0075001 : adhesion of symbiont infection structure to host [1 gene product]
            GO:0075003 : adhesion of symbiont appressorium to host [0 gene products]
            GO:0075002 : adhesion of symbiont germination tube to host [0 gene products]
            GO:0075196 : adhesion of symbiont haustorium mother cell to host [0 gene products]
            GO:0075070 : adhesion of symbiont hyphopodium to host [0 gene products]
            GO:0075069 : adhesion of symbiont infection cushion to host [0 gene products]
```

Fig. 1. Excerpt of Gene ontology.

In the selected excerpt of Gene ontology (see fig.1) we can see the massive amount of biology data (675149 instances). Hence, this ontology is a good condidate on which we can apply our new process.

The paper is organised as follows: Section 2 presents the process of OBDB construction. Section 3 describes the mapping rules of OBDB, the mapping algorithm and its implementation. Finally, we give conclusions and perspectives.

2 OBDB process

To implement OBDB proces we need to define ontology and database components and show the mapping between them.

2.1 Ontology

Ontology is a formal representation of terms related to specific domain. Studer in 1995 [18] presented ontology as ”*Formal specification of a shared conceptualization*”. Perez et al. [10] summarized that an ontology induces five basic primitive components:

- Concept: is an object, idea,... In ontology, a concept is defined by its name, a textual description and relationship with other concepts (*i.e. disease, symptom*).

- Relationship: is the interaction between concepts of a domain. Ontologies usually contain only binary relations (*i.e. Has-a(disease, symptom)*).
- Function: is a kind of special relations. It is a complex structures formed from certain relations (*i.e. Symptom-Name = Name(Symptom)*).
- Axiom: is a constraints or a rules defined on concepts and attributes. The formal axioms are used to verify the consistency of ontology
  (*i.e.* $\forall$ *s Symtpom*, $\exists$ *d Disease*, $s \subset d$).

- Instance: is the concrete element of the concept and it is used to represent items in a field (*i.e.* $\{Symptom - Name : Epilepsy, diarrhea, fatigue\}$).

2.2 Ontology vs Database

A DB is an organized collection of data. The ontology is different to relational DB schema as Dehainsala and al. in [7] and Zhang and al. in [12] the difference is:

- In ontology there is a set of concepts linked by a set of binary relationships but in DB there is a set of tables. The relationships are expressed either:
  1. By the migration of a primary key of a table as foreign key in other tables.
  2. By the creation of a new table which has a combination of related tables' primary key as primary key.
- Instances in ontology are called records in DB.
- In ontology, only binary relationships exist but in relational databases, n-ary relations can be found [12].

2.3 Mapping between ontology and DB

Mapping between ontology and DB is defined by Cullot and al. [6] as *" a set of correspondences between database components and ontology components "* . The mapping process starts by treating concepts, relationships and instances [11] .

Two steps for DB construction are proposed : (1) Apply rules to ontology, (2) generate DB . Ontology is the input; rules are used to create DB and DB is the output (See fig.2).

3 Mapping rules of Database based ontology

Definition 1 (Ontology structure): An ontology structure is a quintuple [9] [13]:

$$O = (C, R, A, I, F). \tag{1}$$

Where:
C : a finite set of concepts.
R=: a finite set of relations.

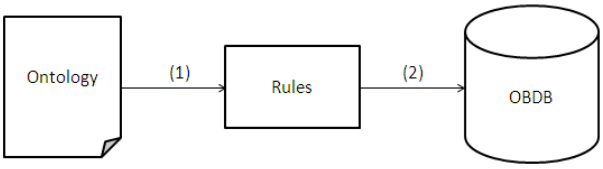


Fig. 2. Ontology constructing steps.

A : a set of finite set of axioms, which is expressed as an appropriate logical language.
I : instances.
F : a set of functions.

Definition 2 (Relational DB): Relational DB is composed of a set of tables *(T1,T2...Tn)*, and each table is composed of a set of fields *(F1, ...,Fm)* which can be primary key *(Pk)* or foreign key *(Fk)* and data *(d1, ..., dn)* [12].

Definition 3 (Concept composition): Concept is a triplet:

$$C = (Pv, Pc, Hc). \tag{2}$$

Where:

Pv (Property value): is a value constraint or the values assigned to the concept properties in order to obtain an instance.

Pc (Property constraint): denotes the cardinality constraint. We have two kinds of cardinality maxCardinality and minCardinality *Pc=(minCard, maxCard)*, *minCard* is a restriction containing constraint describes the number of individuals that have at least *minCard=n* semantically distinct values; *maxCard* is a restriction containing constraint describes the number of instances that have at most *maxCard=N* semantically distinct values.

Hc (Hierarchical concept): is called concept hierarchy, which is a directed relation C1:=H(C2) denotes that *C2* is a subconcept of *C1* [12]. There are two kinds of hierachical relation Parent relation (*C1:=P(C2)* means that *C1* is the parent of *C2*) and child relation (*C2:=Ch(C1)* means that *C2* is the child of *C1*).

Based on general features of relational DB and ontology, we propose the specific rules which map ontology to database:

Rule1: Mapping approach converts ontology into DB tables:

$$DB := map(O). \tag{3}$$

Rule2: When C is a parent concept then C is mapped to table T:

$$T := Tab(C) \tag{4}$$

and
Pk(T) := id(C).

Where:
Tab(X):Maps the concept X to table.
id(Y): Returns the field of concept Y, which is not null and unique as identifier.
Pk(Z): Refers to the primary key of concept Z.

Rule3: When *C1* and *C2* are two concepts mapped to tables (respectively *T1* and *T2*), and *C1= H(C2)* then the primary key of the table *T2* will be a foreign key in table *T1*.

$$When\ T1 = Tab(C1)\ and\ T2 = Tab(C2), \tag{5}$$

then Fk(T1):=PK(T2).

Where:
Fk(X): refers foreign key of concept X.
PK(Y): returns the primary key of concept Y.

Rule 4: When two concepts *C1* and *C2* map to tables (respectively *T1*, *T2*) in DB and R is a relationship between *C1* and *C2* (*R(C1, C2) =True*), then R maps to relationship between table *T1* and table *T2*.

$$When\ R(C1, C2) = True\ and\ T1 = Tab(C1),\ T2 = Tab(C2) \tag{6}$$

then R is preserved in DB.

Rule 5: When two concepts related with relationship R and (*C1* and/or *C2*) not mapped to tables then R is not preserved in DB.

Rule 6: A pure concept is mapped to field in DB table related to parent concept.

$$When\ C2 = Ch(C1)\ and\ Ch(C2) = Null \tag{7}$$

then T:=Tab(C1) and T:=Field(T,C2).

Where:
Field(X,C): affects to table X the pure concept C as field.

Rule 7: In general case, instances are mapped to records when related concepts are mapped to tables. But, when concept maps to attribute so its instances map to values of this attribute (that means: instances represent the list of values which this attribute can take).

Figure 3 presents an example which describes the relation between ontology and DB. DB contains two tables: *disease* and *symptom*. Each table has a set of fields. For example, table *symptom* has three fields: *idSym, nameSym, typeSym*. The underlined fields indicate the primary keys. Ontology contains four concepts: *Symptom, Disease, Cardiac_symptom, Nervous_symptom. Cardiac_symptom, Nervous_symptom* are two subconcepts. Some concepts in ontology are not mapped to tables, they are mapped to attributes in tables. Such as *Cardiac_symptom* and *Nervous_symptom* are mapped to *nameSym* attribute in *Symptom* table.

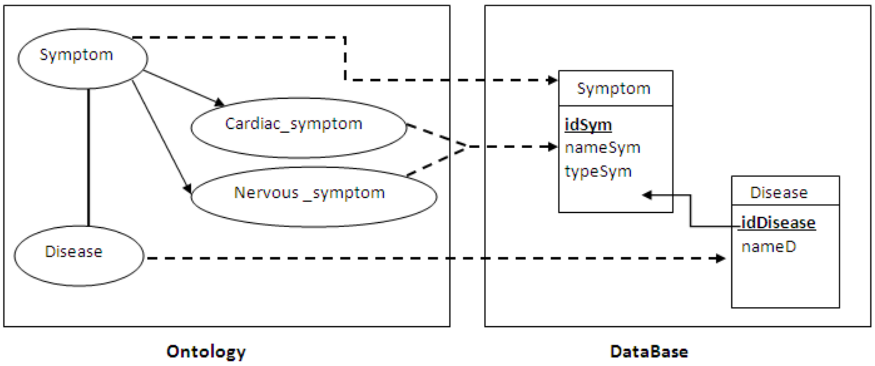


Fig. 3. Mapping example.

4 Implementation

Ontology-DB mapping algorithm is iterative. It analyses each ontology component and applies rules explained previously. Algorithm treats two cases, one concerns parent concept and other concerns pure concept. In each case, we analyse ontology, apply rules and generate DB.

Ontology Mapping Algorithm

```
Ontology (Input)
Relational Database (Output)
1. BEGIN
```

```
For each concept
Select concept
  If concept == parent
     Apply rule 2
     Apply rule 3
     Apply rule 4
  Else
    Apply rule 6
    Apply rule 5
    Apply rule 7
  End If
 End For
END.
```

We developed our algorithm in Java Language using Eclipse and OWLAPI (Onology Web Language API). The resulting programme is applied to *Gene ontology*, which contains thousands of concepts and properties.

Figure 4 illustrates the mapping between a part of *Gene ontology* and the resulted OBDB. For example, concepts *biological process, biological adhesion, adhesion to other organism, adhesion to host* and *adhesion of symbiont* map to tables in DB (applying **rule 1**). **Rule 3** is used to map the relationship between them. The subconcepts of *adhesion of symbiont* are pure, we use **rule 6** to map them to field in *adhesion of symbiont* table.
The result of proposed algorithm is a meaningful OBDB which can be used in the development of computer applications in biomedical domain.

5 Conclusion

This paper presents a new method for automatically generating relational DB from biomedical ontology. Some mapping rules are proposed. This method needs to be tested on different types of ontologies and different domains in order to confirm its effectiveness.

References

1. *Integration de bases de donnees heterogenes par articulation a priori d'ontologies: application aux catalogues de composants industriels.* PhD thesis, Ecole Nationale Superieure de Mecanique et d'Aerotechnique, 2002.
2. *OntoQL, un langage d'exploitation des bases de donnees a base ontologique.* PhD thesis, Ecole Nationale Superieure de Mecanique et d'Aerotechnique, 2007.
3. S. Alexaki, V. Christophides, G. Karvounarakis, D. Plexousakis, and K. Tolle. The ics-forth rdfsuite: Managing voluminous rdf description bases, 2001.
4. B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems and Their Applications*, 14(1):20–26, 1999.

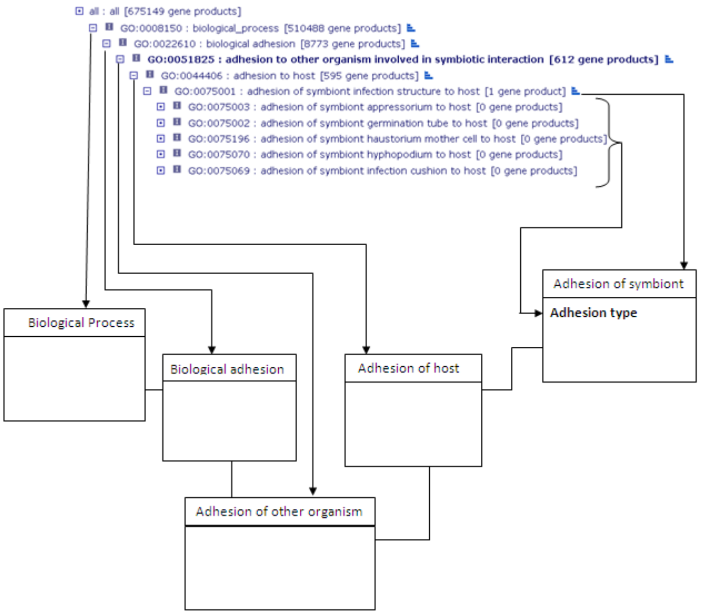

all : all [675149 gene products]
GO:0008150 : biological_process [510488 gene products]
GO:0022610 : biological adhesion [8773 gene products]
GO:0051825 : adhesion to other organism involved in symbiotic interaction [612 gene products]
GO:0044406 : adhesion to host [595 gene products]
GO:0075001 : adhesion of symbiont infection structure to host [1 gene product]
GO:0075003 : adhesion of symbiont appressorium to host [0 gene products]
GO:0075002 : adhesion of symbiont germination tube to host [0 gene products]
GO:0075196 : adhesion of symbiont haustorium mother cell to host [0 gene products]
GO:0075070 : adhesion of symbiont hyphopodium to host [0 gene products]
GO:0075069 : adhesion of symbiont infection cushion to host [0 gene products]
Adhesion of symbiont
Adhesion type
Biological Process
Biological adhesion
Adhesion of host
Adhesion of other organism

Fig. 4. Mapping example of Gene ontology.

5. J. R. Chandrasekarn, B. What are ontology, and why do we need them? In *IEEE Intelligent System*, 1999.
6. N. Cullot, R. Ghawi, and K. Ytongnon. Db2owl : A tool for automatic database-to-ontology mapping. In M. Ceci, D. Malerba, and L. Tanca, editors, *SEBD*, pages 491–494, 2007.
7. H. Dehainsala, G. Pierra, and L. Bellatreche. L.: Ontodb: An ontology-based database for data intensive applications. In *In Proc. of the 12th Int. Conf. on Database Systems for Advanced Applications (DASFAA'07). LNCS*, pages 497–508. Springer, 2007.
8. H. Dehainsala, G. Pierra, and L. Bellatreche. Ontodb: An ontology-based database for data intensive applications. In *IN PROC. OF THE 12TH INT. CONF. ON DATABASE SYSTEMS FOR ADVANCED APPLICATIONS (DASFAA'07). LNCS*, pages 497–508. Springer, 2007.
9. W. S. Du Xiaoyong, Li Man. A survey on ontology learning research. *Journal of software*, 17(9).
10. A. Gomez-Prez, M. Fernandez-Lopez, and O. Corcho. Ontological engineering with examples from the areas of knowledge management, e-commerce and the semantic web. *Springer*, 2004.
11. J. Z. J. D. Huizhang Shen, Jingwen Hu. Ontology-based modeling of emergency incidents and crisis management. *Modeling of Emergency Incidents Management*, 2012.
12. J. L. Lei ZHANG. Automatic generation of ontology based on database. *Journal of Computational Information Systems*, 7.

13. W. S. Li M, Du XY. Learning ontology from relational database. In *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, pages 3410–3415. IEEE Press, 2006.
14. L. Ma, Z. Su, Y. Pan, L. Zhang, and T. Liu. Rstar: an rdf storage and query system for enterprise resource management. In D. A. Grossman, L. Gravano, C. Zhai, O. Herzog, and D. A. Evans, editors, *Proceedings of the 2004 ACM CIKM International Conference on Information and Knowledge Management, Washington, DC, USA, November 8-13, 2004*, pages 484–491. ACM, 2004.
15. B. McBride. Jena: Implementing the RDF model and syntax specification. In S. Staab, editor, *Proc. of the 2. Int. Workshop on the Semantic Web - SemWeb'2001*, volume 40 of *CEUR-WS*, Hongkong, China, 2001.
16. N. F. Noy. What do we need for ontology integration on the semantic web position statement. In *Second International Semantic Web Conference*, 2003.
17. D. L. Rubin, N. H. Shah, and N. F. Noy. Biomedical ontologies: a functional perspective. *Brief Bioinform*, 9(1):75–90, Jan. 2008.
18. G. Studer, B. Richard, and D. Fensel. Knowledge enginerinng: principales and methods. *Data Knowl. Eng*, 1998.

.