Materialized views in mining ontology instances

Joanna Józefowska[1], Agnieszka Ławrynowicz[1], and Tomasz Łukaszewski[1]

Institute of Computing Science, Poznan University of Technology,
ul. Piotrowo 2, 60-965 Poznan, Poland
{jjozefowska, alawrynowicz, tlukaszewski}@cs.put.poznan.pl

Abstract. The Semantic Web technologies are used to produce growing number of ontologies and growing amount of semantically marked up data. In order to exploit such huge resource the methods are needed for mining data described by ontologies. In this work we introduce an approach, based on the concept of a materialized view, to speed up data mining methods of finding regularities in ontology instance data. Although the method is presented for mining frequent patterns it is general enough to be applied to other data mining tasks in which ontologies are used in knowledge discovery.

1 Pattern mining from the Semantic Web

The realization of the *Semantic Web* requires *semantic publishing* becoming the main form of Web publishing thus enabling an access to enormous data resource. Hence the problem of mining *from* the Semantic Web [3] will become as important as currently is data mining from local databases.

In [1] we presented a method for frequent pattern mining, exploiting background knowledge represented in a subset of OWL, which can be considered as an approach for mining from the Semantic Web. Given an ontology O with a TBox and an ABox, the method consist of finding a set of queries Q that capture the characteristics of an ABox - *frequent patterns*. Pattern is called *frequent* when it has *support* threshold above a user-specified minimum value. The support of query Q with respect to a knowledge base $\mathcal{KB}$ is defined as the ratio between the number of instances of a user-specified reference class $\hat{C}$ that satisfy query Q and total number of instances of class $\hat{C}$. The goal is to find frequent patterns in the form of conjunctive, positive $\mathcal{DL}$-safe queries over $\mathcal{KB}$ [2]. Frequent pattern $\mathcal{Q}$ contains individuals of reference class $\hat{C}$ in its answer set. Queries can be formulated using SPARQL. Below is the example pattern for *foaf:Person* being reference class $\hat{C}$:

Q=*q(x):-foaf:Person(x), swc:holdsRole(x,y1), swc:ProgrammeCommitteMember(y1)*

and its SPARQL syntax:

```
SELECT ?x
WHERE { ?x rdf:type foaf:Person .
        ?x swc:holdsRole ?y1 .
        ?y1 rdf:type swc:ProgrammeCommitteMember }
```

2 Speeding up pattern mining by materialized views

Our algorithms proceed in a systematic way, refining a query by adding one atom to it. Hence, more specific queries are very similar to their "parents" and what follows, the results of the consecutive queries may be very similar. This property leads to an idea of reusing materialized results of previous frequent queries by storing them in materialized views. The idea is motivated by the monotonicity of *query containment* which is the generality notion $\succeq$ between patterns in our approach. Query containment is monotonic w.r.t. support, that is for every pair of patterns *Q1* and *Q2*: $Q1 \succeq Q2 \Rightarrow support(Q1) \geq support(Q2)$.

In a relational database system, a *view* is a virtual table representing the result of a database query. A *materialized view* is the view cached in a real table. For SPARQL queries, CONSTRUCT clause can nicely fulfill the role of "view definition language". As an example consider the clause for query Q from the previous section:

```
CONSTRUCT { ?x rdf:type :$ref_3 . ?x :$supp_3_y1 ?y1}
WHERE { ?x rdf:type foaf:Person .
        ?x swc:holdsRole ?y1 .
        ?y1 rdf:type swc:ProgrammeCommitteMember }
```

This construct clause introduces special purpose predicates to store the bindings of each variable in the query (note that the bindings are stored also for variables previously undistinguished). For the variable in reference class $\hat{C}$, a class $\$ref\_l$ is created, where l denotes the length of a query. For each remaining variable v_i in a query, the property $\$sup\_l\_v_i$ is created.

The tests of the proposed technique on ontologies of different complexities proved that it achieves considerable speedups in computing time as presented in Figure 1[1]. Our method is general enough to be adapted to another data mining methods that consist of systematically searching the pattern space from more general to more specific patterns and that exploit the monotonicity property.

Ontology	Ref class	Support	Max_length	Speedup
FINANCIAL	Client	0.2	6	223%
LUBM	Person	0.3	4	532%
BioPax	entity	0.3	5	4147%

Fig. 1. Experimental results.

References

1. Józefowska J., Ławrynowicz A., Łukaszewski T. (2006) Frequent pattern discovery in OWL DLP knowledge bases, In Proc. of EKAW 2006, Springer, 287-302
2. Motik B., Sattler U., Studer R. (2004) Query Answering for OWL-DL with Rules. In Proc. of ISWC 2004, Springer, 549-563
3. Stumme G., Hotho A., Berendt B. (2006), Semantic Web Mining: State of the art and future directions, Journal of Web Semantics 4(2): 124-143

[1] For description of datasets see: http://www.ecmlpkdd2007.org/CD/workshops/PRICKLWM2/P_Joz/p7Final.pdf