Mining Association Rules from Semantic Web Data

Victoria Nebot and Rafael Berlanga

Universitat Jaume I, Campus de Riu Sec, E-12071 Castellón de la Plana, Spain
{romerom,berlanga}@lsi.uji.es

Abstract. The amount of ontologies and semantic annotations available on the Web is constantly increasing. This new type of complex and heterogeneous graph-structured data raises new challenges for the data mining community. In this paper, we present a novel method for mining association rules from semantic instance data repositories expressed in RDF/S and OWL. We take advantage of the schema-level (i.e. *Tbox*) knowledge encoded in the ontology to derive just the appropriate transactions which will later feed traditional association rules algorithms. This process is guided by the analyst requirements, expressed in the form of a query pattern. Initial experiments performed on real world semantic data enjoy promising results and show the usefulness of the approach.

Keywords: Semantic Web, Data Mining, Instance Data, Association Rules.

1 Introduction

Thanks to the standardization of the ontology languages RDF/S[1] and OWL[2], the Semantic Web has been realized and the amount of available semantic annotations is ever increasing. This is due in part to the active research concerned about learning knowledge structures from textual data, usually referred as Ontology Learning [1]. However, little work has been directed towards mining from the Semantic Web. We strongly believe that mining Semantic Web data will bring much benefit to many domain-specific research communities where relevant data are often complex and heterogeneous, and a large body of knowledge is available in the form of ontologies and semantic annotations. This is the case of the clinical and biomedical scenarios, where applications often have to deal with large volumes of complex data sets with different structure and semantics. In this paper, we investigate how ontological instances expressed in OWL can be combined into transactions in order to be processed by traditional association rules algorithms, and how we can exploit the rich knowledge encoded in the respective ontologies to reduce the search space.

The rest of the paper is organized as follows. Section 2 gives an overview of the related work. Section 3 explains the basics of the two integrated technologies,

[1] RDF/S: http://www.w3.org/TR/rdf-concepts/rdf-schema/, '04
[2] OWL: http://www.w3.org/TR/owl-features/, '04.

N. García-Pedrajas et al. (Eds.): IEA/AIE 2010, Part II, LNAI 6097, pp. 504–513, 2010.

association rules mining and OWL DL ontologies and motivates the problem with a running example. Section 4 contains the general methodology and foundations of the approach. Section 5 shows the experimental evaluation and Section 6 gives some conclusions and future work.

2 Related Work

Most research on data mining for semantic data is based on Inductive Logic Programing (ILP) [2], which exploits the underlying logic encoded in the data to learn new concepts. Some examples are presented in [3] and [4]. However, there is the inconvenient of rewriting the data sets into logic programming formalisms and most of these approaches are not able to identify some hidden concepts that statistical algorithms would.

Other studies extend statistical machine learning algorithms to be able to directly deal with ontologies and their associated instance data. In [5], a framework is presented for designing kernel functions that exploit the knowledge of the underlying ontologies. Recently, in [6] and [7] new frequent association rules algorithms are proposed which make use of similarity functions in a similar way to the previous kernel functions.

A recent trend in data mining research is to consider more complex and heterogeneous structures than single tabular data, mainly tree and graph structured data. In this line, we can find frequent subtree [8] and graph mining [9], whose aim is to identify frequent substructures in complex data sets. Albeit interesting, these algorithms do not serve the purpose of finding interesting content associations in RDF/S and OWL graphs because they are concerned with frequent syntactic substructures but not frequent semantically related contents. Indeed, frequent graph substructures usually hide interesting associations that involve contents represented with different detail levels of the ontology. Moreover, although the underlying structure of RDFS and OWL is a graph, reasoning capabilities must be applied to handle implicit knowledge.

Finally, we find some work aimed at integrating knowledge discovery capabilities into SPARQL[3] by extending its grammar. Some examples are [10], which can be plugged with several data mining algorithms and [11], which finds complex path relations between resources. Inspired by these works, we have also extended SPARQL grammar to define association rule patterns over the ontological data but in a less restrictive way than the one imposed by SPARQL. These patterns allow the system to focus only on the interesting features, reducing both the number and length of generated transactions.

3 Preliminaries

The problem of discovering association rules was first introduced in [12]. It can be described formally as follows. Let $I = \{i_1, i_2, ..., i_m\}$ be a set of m literals,

[3] SPARQL: `http://www.w3.org/TR/rdf-sparql-query`,'08

called items. Let $D = \{t_1, t_2, ..., t_n\}$ be a database of n transactions where each transaction is a subset of I. An itemset is a subset of items. The support of an itemset S, denoted by $sup(X)$, is the percentage of transactions in the database D that contain S. An itemset is called frequent if its support is greater than or equal to a user specified threshold value.

An association rule r is a rule of the form $X \Rightarrow Y$ where both X and Y are nonempty subsets of I and $X \cap Y = \emptyset$. X is the antecedent of r and Y is called its consequent. The support and confidence of the association rule $r : X \Rightarrow Y$ are denoted by $sup(r)$ and $conf(r)$.

The task of the association data mining problem is to find all association rules with support and confidence greater than user specified minimum support and minimum confidence threshold values [12].

DLs allow ontology developers to define the domain of interest in terms of *individuals*, *atomic concepts* (called *classes* in OWL) and *roles* (called *properties* in OWL). Concept constructors allow the definition of *complex concepts* composed of atomic concepts and roles. OWL DL provides for union ($\sqcup$), intersection ($\sqcap$) and complement ($\neg$), as well as enumerated classes (called *oneOf* in OWL) and existential ($\exists$), universal ($\forall$) and cardinality ($\geq$, $\leq$, $=$) restrictions involving an atomic role R or its inverse R^-. In OWL DL it is possible to assert that a concept C is subsumed by D ($C \sqsubseteq D$), or is equivalent to D ($C \equiv D$). Equivalence and subsumption can be also asserted between roles and roles can have special constraints (e.g., transitivity, simmetry, functionality, etc.) Regarding instance axioms, we can specify the class C of an instance a ($C(a)$), or the relations between two instances a and b ($R(a, b)$). A DL ontology consists of a set of axioms describing the knowledge of an application domain. This knowledge ranges over the terminological cognition of the domain (the concepts of interest, its *Tbox*) and its assertions (the instances of the concepts, its *Abox*).

Fig. 1 shows a fragment of the *Tbox* of a DL ontology designed for patients with arthritis-related diseases. Through semantic annotation, clinicians can annotate data sets with ontology terms and relationships from the axioms in Fig. 1, creating a repository of semantic annotations in OWL (an *Abox*) which must be consistent with the ontology axioms (*Tbox*). The right hand side of Fig. 1 shows an excerpt

Axioms
$Patient \sqsubseteq \exists\ doB.string$
$Patient \sqsubseteq \exists\ sex.Gender$
$Patient \sqsubseteq \exists\ hasReport.Report$
$Report \sqsubseteq \exists\ hasDiag.Disease$
$Report \sqsubseteq \exists\ hasSection.Section$
$hasReport \equiv belongsTo^-$
$Treat \sqsubseteq Section \sqcap \exists\ hasDrug.Drug$
$Patient \sqsubseteq ....MotherHist$
$MotherHist \sqsubseteq \exists\ hasDiag.Disease$
$MotherHist \sqsubseteq \exists\ treatedWith.Drug$
$RheumaticDis. \sqsubseteq Disease$

subject	**predicate**	**object**
PTNXZ1	doB	"20021108"
PTNXZ1	sex	Female
PTNXZ1	hasReport	RPT1
RPT1	hasDiag	PolyArthritis
RPT1	hasSection	STreat1
STreat1	type	Treat
STreat1	hasDrug	Methotrexate
PTNXZ1	...	MH1
MH1	hasDiagnosis	RheumatoidArthr.
MH1	treatedWith	NSAIDS
PolyArthritis	type	Arthritis
RheumatoidArthr.	type	Arthritis

Fig. 1. Ontology axioms (Tbox) and semantic annotations (Abox)

of the semantic annotations associated to a patient named *PTNXZ1*. Notice that semantic annotations are represented as triples $(subject, predicate, object)$. In this paper, we separate the *Tbox* from the *Abox* for practical issues. In general, we use the term *ontology* to refer to the *Tbox* and *instance store* to refer to the *Abox*. Thus, our data mining problem is defined as follows:

Data Mining Problem: Given an OWL instance store IS consistent with the ontology O and a mining pattern Q expressed in the extended SPARQL syntax (see Section 4.1), find association rules from IS according to the mining pattern Q with minimum support and confidence threshold values.

The previous data mining problem can be thought as a classic data mining problem where the input data must be derived from the ontology in order to generate transactions according to the user specification.

4 Methodology

In this section we present a detailed view of our method along with the definitions that sustain it. Fig. 2 depicts a schematic overview of the whole process. The user specifies a mining pattern following an extended SPARQL syntax. Then, the transaction extractor is able to identify and construct transactions according to the mining pattern previously specified. Finally, the set of transactions obtained are processed by a traditional pattern mining algorithm, which finds association rules of the form specified in the mining pattern with support and confidence greater than user's specified ones.

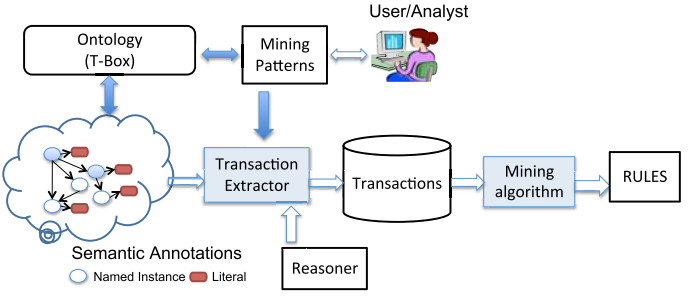

Fig. 2. Arquitecture of our proposal for mining semantic annotations

4.1 Mining Pattern Specification

The user has to specify the kind of patterns (s)he is interested in obtaining from the repository. Since semantic annotations are encoded in RDF/S and OWL, we have extended SPARQL with a new statement that allows to specify a mining pattern. The syntax is inspired by the Microsoft Data Mining Extension (DMX),

```
[1]    Query        ::=   Prologue( SelectQuery | ConstructQuery | DescribeQuery | AskQuery |
                              MiningQuery )
[2]    MiningQuery  ::=   CREATE MINING MODEL' Source  '{' Var 'RESOURCE'  'TARGET' (
                          Var ( 'RESOURCE' | 'DISCRETE' | 'CONTINUOUS' )
                          'MAXCARD1'? 'PREDICT'? 'CONTEXT'?)+ '}'
                          DatasetClause* WhereClause SolutionModifier UsingClause
[1.2]  UsingClause  ::=  'USING'  SourceSelector BrackettedExpression
```

Fig. 3. Extended SPARQL grammar for the CREATE MINING MODEL statement

```
CREATE MINING MODEL <http://krono.act.uji.es/patients_repository>
   { ?patient   RESOURCE TARGET
     ?drug      RESOURCE
     ?disease   RESOURCE PREDICT
     ?report    RESOURCE CONTEXT
   }
 WHERE
   { ?patient    rdf:type Patient .
     ?drug       rdf:type Drug .
     ?disease    rdf:type Disease .
     ?report     rdf:type Report .
   }
 USING apriori  (SUPPORT = 0.05, CONFIDENCE = 0.07)
```

Fig. 4. Example of extended SPARQL query with CREATE MINING MODEL statement

which is an SQL extension to work with data mining models in Microsoft SQL Server Analysis Services.[4]

The extended SPARQL grammar is depicted in Fig. 3, and Fig. 4 shows an example query. We extend the SPARQL grammar rule *Query* by adding a new symbol, named *MiningQuery*. This symbol expands to the keywords CREATE MINING MODEL followed by the *Source*, which identifies the input repository. The body consists of variables the user is interested in mining. Next to each variable, we specify its content type: RESOURCE for variables holding RDF resources, DISCRETE for variables holding literal values and CONTINUOUS for variables holding a continuous literal value. In case we want to find patterns with just one occurrence of the variable, we attach the keyword MAXCARD1 to the variable. By default, patterns found can contain more than one occurrence of each variable. Moreover, we specify the consequent of the rule by attaching the keyword PREDICT. Finally, the keyword TARGET denotes the resource under analysis, which must be an ontology concept. The analysis target determines the set of obtained rules. In the query example, the analysis target is a *Patient*. In the WHERE clause, we specify the restrictions over the previous variables. The good news is that we do not expect users to have an exact knowledge of the ontology structure. Therefore, users do not have to input the paths relating the pattern variables in SPARQL. Instead, they are asked to specify just the type (i.e. ontology concept) that the variables refer to. In case variables are not resources, users must specify the type of the domain of that value followed by the data type property. From now on, we refer to these ontology concepts selected

[4] DMX Reference: `http://technet.microsoft.com/en-us/library/ms132058.aspx`

by the user as the *features* set. For example, in Fig. 4 the user is interested in obtaining which drugs are associated to which diseases, so the features set is $\{Drug, Disease\}$. Finally, the *UsingClause* grammar symbol defines the name and parameters of the learning algorithm.

Since we do not ask the user to specify the exact relations, the previous query model introduces some ambiguity regarding the items that form a transaction. When the user specifies the mining pattern, (s)he is thinking about obtaining subsets of drugs that are frequently administered to patients having a certain disease. Therefore, (s)he restricted the features set to be of type *Drug* and *Disease*, respectively. However, these concepts may appear not only under the user intended context but all over the ontology. That is to say, the same conceptual entities may appear under different contexts, making it challenging for the system to automatically discover what the users' intentions really are. As previously mentioned, the user could remove this ambiguity by specifying in the SPARQL extended query the exact relation of concepts in the ontology through pattern graph triples in the WHERE clause. However, this task can be cumbersome and not always viable. In this paper, we want to relief the user from this burden and let the system handle the task of finding appropriate contexts. Thus, in order to provide the right sense to the query, user can select the intended context with the CONTEXT keyword attached to the appropriate concept.

4.2 Transaction Extractor Foundations

Since our goal is to be able to identify and construct transactions according to the user's mining pattern, in this section we present all the definitions that sustain the method we have developed.

Definition 1. *Let O be an ontology, IS an instance store consistent with O and C_T the analysis target. The target instances are the set $I_T = \{i/i \in IS,\ O \cup IS \models C_T(i)\}$.*

In the running example C_T is *Patient* and I_T is the set of all instances classified as *Patient*.

Definition 2. *$Path(C, C') = (r_1 \circ ... \circ r_n) \in Paths(C, C')$ is an aggregation path from concept C to concept C' of an ontology O iff $O \models C \sqsubseteq \exists\ r_1 \circ ... \circ r_n.C'$.*

Definition 3. *Let O be an ontology, C_T the analysis target and C_a, C_b two named concepts.*
$Contexts(C_a, C_b, C_T) = \{C''/C'' \sqsubseteq C'\}$ are least common reachable concepts and their subconcepts. That is,

(1) $\exists p_1 \in Paths(C_a, C') \wedge \exists p_2 \in Paths(C_b, C') \wedge \exists p_3 \in Paths(C', C_T)$ (C' is common reachable concept).
(2) if $\exists p_x \in Paths(C_a, E) \wedge \exists p_y \in Paths(C_b, E)$ then $\exists p_z \in Paths(C', E)$ (C' is least).

Example 1. The ontology fragment on the left in Fig. 5 models some part of the patient's medical record. In this example we can infer that $Contexts(Disease, Drug, Patient) = \{Report, MotherHistory\}$.

Definition 4. *Let i and i' be two named instances of an instance store IS. $Path(i, i') = (r_1 \circ ... \circ r_n) \in Paths(i, i')$ is an aggregation path from instance i to instance i' in the instance store IS.*

Definition 5. *Let $i_T \in I_T$ be a target instance and i_a, i_b two named instances in an instance store IS.*
$Contexts(i_a, i_b, i_T) = \{i'\}$ are least common reachable instances. That is,

(1) $\exists p_1 \in Paths(i_a, i') \ \wedge \ \exists p_2 \in Paths(i_b, i') \ \wedge \ \exists p_3 \in Paths(i', i_{SUB})$ (i' is common reachable instance).
(2) if $\exists p_x \in Paths(i_a, i'') \wedge \exists p_y \in Paths(i_b, i'')$ then $\exists p_z \in Paths(i', i'')$ (i' is least).

Example 2. The right hand side of Fig. 5 shows an example of instance store represented as a graph that is consistent with the ontology fragment. In this example, we can infer that $Contexts(PolyArthritis, Methotrexate, PTN\_XY21) = \{RPT\_NH23\}$.

Definition 6. *Let O be an ontology and IS an instance store consistent with O. Two instances i_a, i_b, $C(i_a) \neq C(i_b)$, $C(i_a), C(i_b) \in features$ belong to the same transaction under a target instance i_T iff i_a, i_b are context compatible. That is,*
$\exists\, C_1 \in \{C(i_x)/i_x \in Contexts(i_a, i_b, i_T)\}$,
$\exists\, C_2 \in Contexts(C(i_a), C(i_b), C_T)$ such that
$O \models C_1 \sqsupseteq C_2$ where $C(i)$ is the asserted class for instance i in IS.

Example 3. In the right hand side of Fig. 5, instances *Methotrexate* and *PolyArthritis* are *context compatible*. That is, their context is *Report* both at the instance level (Abox) and at the conceptual level (Tbox).

As previously mentioned, context compatible instances may appear under very different contexts in an ontology, making it hard for the system to guess the user intended context. Currently, we allow users to specify the contexts of interest directly in the query. The concept denoting the selected context is denoted as C_{CTXT}. If no context is specified, we assume $C_{CTXT} = C_T$. Now, transactions can be unambiguosly defined as follows.

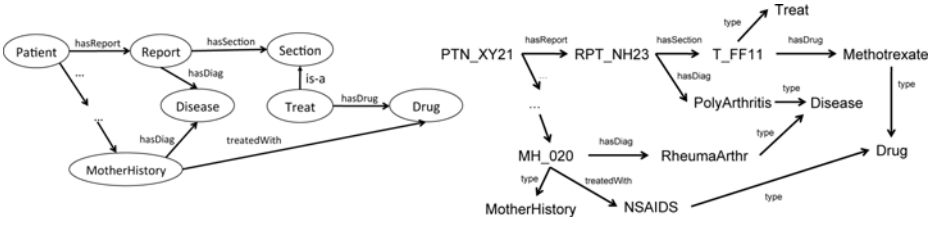

Fig. 5. Ontology graph and instance store fragment

Definition 7. *Let O be an ontology and IS an instance store consistent with O. An* instance transaction *associated to target instance i_T under context C_{CTXT} is a tuple of instances $(i_1, i_2, ..., i_n)$ such that $\forall i_x, 1 \leq x \leq n$, $O \cup IS \models C(i_x)$ where $C \in features$ and $\forall (j,k), 1 \leq j,k \leq n$, if $C(i_j) \neq C(i_k), (i_j, i_k)$ are context compatible under i_T w.r.t. the user selected context C_{CTXT}.*

Example 4. Given $C_T = Patient$ and $features = \{Drug, Disease\}$ we have three options. If the user does not select a context (i.e. $C_{CTXT} = Patient$) the transaction $\{Methotrexate, PolyArthritis, RheumatoidArthritis, NSAIDS\}$ would be generated. If $C_{CTXT} = Report$, the transaction $\{Methotrexate, PolyArthritis\}$ would be generated. Finally if $C_{CTXT} = MotherHistory$, the generated transaction would be $\{RheumatoidArthritis, NSAIDS\}$.

5 Evaluation

The current implementation of the transaction extractor has been developed on the top the ontology indexing system proposed in [13], which also provides a simple reasoning mechanism over the ontology indexes. In order to show the usefulness of our proposal, we test the method over a real-world instance store holding OWL annotations about patient's follow-ups. These annotations have been generated in the context of the Health-e-Child project[5], and they are consistent with an ontology similar to the one used as example in Fig. 1. The semantic annotations contain information about 588 patients with very heterogeneous structure. The total number of semantic annotations is 629.000, which gives more than 1000 semantic annotations per patient on average.

Table 1 shows some query pattern examples[6] along with the user selected context, the number of transactions generated by our method and the number of rules generated with the *Apriori* algorithm [14] for mining association rules. The query patterns specify the ontology concepts acting as interesting features, leaving the PREDICT attribute unmarked. As it can be observed, the first query is executed in for the context of *Patient*, which means a transaction will be generated for each patient holding features of type *Disease*, *Drug* and *Finding*. The selection of the context is crucial because it determines both the number and contents of each transaction, and therefore, the obtained rules. Notice the number of transactions is very reduced thanks to the context selection and, consequently, the number of generated rules.

Table 1 also shows some examples of the rules generated from the previous queries. The obtained rules are very clear and useful thanks to the query pattern restrictions and latter transaction generation, which extremely reduces the features' search space thus, the complexity and overload produced by uninteresting features. To corroborate this fact, we generated transactions with all the possible features at the context of *Patient* and the *apriori* algorithm obtained more than 400.000 rules, of which the first hundred were uninteresting. Notice

[5] `http://www.health-e-child.org/`

[6] We omit full syntax of the query pattern due to space restrictions.

Table 1. Examples of rules obtained with different queries and contexts (between brackets). All the queries are performed with minimum confidence of 0.7. LOM stands for limitation of motion, ANA for anti-nuclear antibody and RF for Rheumatoid Factor.

Query	**Examples of rules**	**Sup.**	**Conf.**
$Drug \wedge Finding \implies$	LargeJointsAffected $\implies$ Oligoarthritis	0.19	0.91
Disease [Patient]	PresenceOfANA $\implies$ Oligoarthritis	0.14	0.76
225 transactions	PrevDrugEtanercept $\implies$ PrevDrugMethotrexate	0.12	0.9
22 rules (sup=0.1)	PrevDrugPrednisone $\implies$ SystemicArthrities	0.11	1.0
	Fever , Rash $\implies$ SystemicArthrities	0.11	1.0
$Finding \implies$	LargeJointsAffected $\implies$ Oligoarthritis	0.087	0.91
Disease [Report]	PresenseOfANA $\implies$ Oligoarthritis	0.063	0.76
492 transactions	Fever , Rash $\implies$ SystemicArthrities	0.05	1.0
12 rules (sup=0.05)			
$Finding \implies$	ANA_Negative $\implies$ RF_Negative	0.36	0.86
Finding [Med.Ev.]	LOMRightShoulder $\implies$ LOMRightWrist	0.08	0.81
84 transactions	LOMLeftHip $\implies$ LOMRightHip	0.08	0.73
21 rules (sup=0.08)			

for queries 1 and 2 we get the same rule stating that the presence of ANA implies oligoarthritis disease. However, the support is different because their respective contexts are different, which generates different transaction sets.

Overall, by specifying a query pattern and selecting the intended context, the features' search space is extremely reduced. That is, the generated transactions contain only the interesting stated and inferred features for the user at the level of granularity specified in the context. Therefore, this process eases the task of the association rule algorithm, which extracts small and very useful subsets of interesting rules.

6 Conclusions

We have presented a novel method for mining association rules from heterogeneous semantic data repositories expressed in RDF/S and OWL. To the best of our knowledge, this problem has only been considered to a minor extend. The intuition under the method developed is to extract and combine just the interesting instances (i.e. features) from the whole repository and flatten them into traditional transactions while capturing the implicit schema-level knowledge encoded in the ontology. Then, traditional association rules algorithms can be applied. We believe this type of learning will become increasingly important in future research both from the machine learning as well as from the Semantic Web communities. Initial experiments on real world Semantic Web data enjoy promising results and show the usefulness of our approach. As future work, we would like to apply generalized query patterns by using the ontology axioms, as well as to automatically discover interesting contexts and their association rules.

References

1. Buitelaar, P., Cimiano, P., Magnini, B. (eds.): Ontology Learning from Text: Methods, Evaluation and Applications. Frontiers in Artificial Intelligence and Applications, vol. 123. IOS Press, Amsterdam (2005)
2. Muggleton, S., Raedt, L.D.: Inductive logic programming: Theory and methods. J. Log. Program 19/20, 629–679 (1994)
3. Lisi, F.A., Esposito, F.: Mining the Semantic Web: A logic-based methodology. In: Hacid, M.-S., Murray, N.V., Raś, Z.W., Tsumoto, S. (eds.) ISMIS 2005. LNCS (LNAI), vol. 3488, pp. 102–111. Springer, Heidelberg (2005)
4. Hartmann, J., Sure, Y.: A knowledge discovery workbench for the Semantic Web. In: Workshop on Mining for and from the Semantic Web at the ACM SIGKDD (August 2004)
5. Bloehdorn, S., Sure, Y.: Kernel methods for mining instance data in ontologies. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 58–71. Springer, Heidelberg (2007)
6. Dánger, R., Ruiz-Shulcloper, J., Llavori, R.B.: Objectminer: A new approach for mining complex objects. In: ICEIS (2), pp. 42–47 (2004)
7. Rodríguez-González, A.Y., Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A., Ruiz-Shulcloper, J.: Mining frequent similar patterns on mixed data. In: Ruiz-Shulcloper, J., Kropatsch, W.G. (eds.) CIARP 2008. LNCS, vol. 5197, pp. 136–144. Springer, Heidelberg (2008)
8. Chi, Y., Muntz, R.R., Nijssen, S., Kok, J.N.: Frequent subtree mining - an overview. Fundam. Inform. 66(1-2), 161–198 (2005)
9. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: Cercone, N., Lin, T.Y., Wu, X. (eds.) ICDM, pp. 313–320. IEEE Computer Society, Los Alamitos (2001)
10. Kiefer, C., Bernstein, A., Locher, A.: Adding data mining support to SPARQL via statistical relational learning methods. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 478–492. Springer, Heidelberg (2008)
11. Kochut, K., Janik, M.: SPARQLeR: Extended SPARQL for semantic association discovery. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 145–159. Springer, Heidelberg (2007)
12. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: SIGMOD Conference, pp. 207–216. ACM Press, New York (1993)
13. Nebot, V., Llavori, R.B.: Efficient retrieval of ontology fragments using an interval labeling scheme. Inf. Sci. 179(24), 4151–4173 (2009)
14. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB, pp. 487–499. Morgan Kaufmann, San Francisco (1994)