Ontology-Driven Association Rule Extraction: A Case Study

Andrea Bellandi[1], Barbara Furletti[1], Valerio Grossi[2], and Andrea Romei[2]

[1] IMT - Lucca Institute for Advanced Studies
Piazza S. Ponziano, 6 - 55100 Lucca, ITALY
{a.bellandi, b.furletti}@imtlucca.it

[2] Department of Computer Science - University of Pisa
Largo B. Pontecorvo, 3 - 56127 Pisa, ITALY
{romei, vgrossi}@di.unipi.it

Abstract. This paper proposes an integrated framework for extracting *Constraint-based Multi-level Association Rules* with an ontology support. The system permits the definition of a set of domain-specific constraints on a specific domain ontology, and to query the ontology for filtering the instances used in the association rule mining process. This method can improve the quality of the extracted associations rules in terms of relevance and understandability.

1 Introduction

The *Data Mining* (DM) results, i.e. the models, represent relations in the data and are usually employed for classifying new data or for describing correlations hidden in the data. In this paper, we focus on the *Association Rule Mining* as originally introduced by Agrawal et al. in [2] and on a way for improving the process results. There are several ways to reduce the computational complexity of Association Rule Mining and to increase the quality of the extracted rules: (i) reducing the search space; (ii) exploiting efficient data structures; (iii) adopting domain-specific constraints. The first two classes of optimizations are used for reducing the number of steps of the algorithm, for re-organizing the itemsets, for encoding the items, and for organizing the transactions in order to minimize the algorithm time complexity. The third class tries to overcome the lack of user data-exploration by handling domain-specific constraints. This paper focuses on these optimizations by representing a specific domain by means of an ontology and driving the extraction of association rules by expressing constraints. The aim of this work is to reduce the "search space" of the algorithm and to improve the significance of the association rules.

Paper Organization. Section 2 provides some notions of OWL ontologies, data mining and association rules. Section 3 introduces the syntax of the constraints and describes the process. Section 4 presents a case study based on a real dataset. Section 5 discusses the related works and section 6 proposes some ideas for further improvements.

2 Background knowledge

OWL Overview

OWL is a family of three ontology languages: $OWL - Lite$, $OWL - DL$, and $OWL - Full$. The first two languages can be considered syntactic variants of the $\mathcal{SHIF}(\mathcal{D})$ and $\mathcal{SHOIN}(\mathcal{D})$ description logics (DL), respectively, whereas the third language was designed to provide full compatibility with RDF(S). We focus mainly on the first two variants of OWL because OWL-Full has a nonstandard semantics that makes the language undecidable and therefore difficult to implement. OWL comes with several syntaxes, all of which are rather verbose. Hence, in this paper we use the standard DL syntax [3]. The main building blocks of DL knowledge bases are concepts (or classes), representing sets of objects, roles (or properties), representing relationships between objects, and individuals representing specific objects. OWL ontologies consist of two parts: intensional and extensional. The former part consists of a $TBox$ and an $RBox$, and contains knowledge about concepts (i.e. classes) and the complex relations between them (i.e. roles). The latter part consists of an $ABox$, and contains knowledge about entities and how they relate to the classes and roles from the intensional part. In our scenario, TBox and RBox shall provide supermarket domain knowledge, while all the supermarket items constitute ABoxes which are interlinked with intensional knowledge.

The semantics for OWL DL is fairly standard. An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a tuple where $\Delta^{\mathcal{I}}$, the domain of discourse, is the union of two disjoint sets $\Delta^{\mathcal{I}}_O$ (the object domain) and $\Delta^{\mathcal{I}}_D$ (the data domain) and $\mathcal{I}$ is the interpretation function that gives meaning to the entities defined in the ontology. $\mathcal{I}$ maps each OWL class C to a subset $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_O$, each object property P_{Obj} to a binary relation $P^{\mathcal{I}}_{Obj} \subseteq \Delta^{\mathcal{I}}_O \times \Delta^{\mathcal{I}}_O$, and each datatype property P_{Data} to a binary relation $P^{\mathcal{I}}_{Data} \subseteq \Delta^{\mathcal{I}}_O \times \Delta^{\mathcal{I}}_D$. The whole definition is in the OWL W3C Recommendation (http://www.w3.org/TR/owl-semantics/).

Data Mining and Association Rules

Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner. The relationships and summaries derived through a data mining exercise are often referred to as models or patterns. The main tasks of Data mining are generally divided in two categories: *Predictive* and *Descriptive*. The objective of the predictive tasks is to predict the value of a particular attribute based on the values of other attributes, while for the descriptive ones, is to derive patterns (correlations, trends, clusters, ...) that summarize the relationships in the data.
The Association rule mining is one of the major techniques of data mining and it is perhaps the most common form of local-pattern discovery in unsupervised learning systems. These methodologies retrieve all possible interesting patterns in the database. Given a database D of transactions, where each transaction

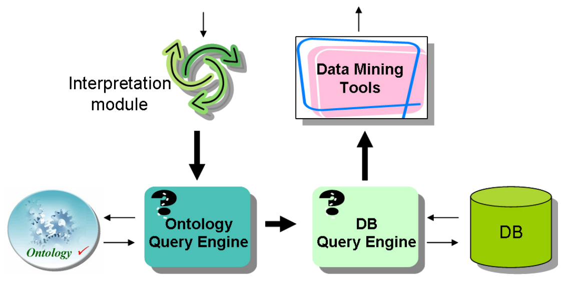

Fig. 1. The system architecture.

$T \in D$ is a set of items, an association rule is a (statistical) implication of the form $X \rightarrow Y$, where $X, Y \in D$ and $X \cap Y = \emptyset$. A rule $X \rightarrow Y$ is said to have a *support* (or frequency) factor s if and only if, at least $s\%$ of the transations in T satisfy $X \cup Y$. A rule $X \rightarrow Y$ is satisfied in the set of transactions T with a $confidence$ factor c if and only if, at least $c\%$ of the transactions in T that satisfy X also satisfy Y. The support is a measure of statistical significance, whereas the confidence is a measure of the strength of the rule. A rule is said to be "interesting" if its support and confidence are greater than user-defined thresholds sup_{min} and con_{min}, respectively, and the objective of the mining process is to find all such interesting rules [13].

3 Description of the approach

In this section, we describe our approach for guiding the extraction process of *Multi-level Constraint-based Association Rules* with an ontology support. Our scenario consists of the set of components shown in figure 1. The ontology ($O_{\mathcal{D}}$) describes the domain of interest ($\mathcal{D}$) and it is used as a means of meta-data representation. The interpretation module translates the requests of an user (*user constraints*) into a set of *formal constrains* ($Q_{\mathcal{D}}$ defined on $O_{\mathcal{D}}$) so that they can be supplied to the Ontology Query Engine by means of a suitable query language. The aim of these constraints is to exclude some items from the output association rules, or to characterize interesting items according to an abstraction level. The *user constraints* syntax is formalized in table 1. It includes both *pruning constraints*, used for filtering a set of non-interesting items, and *abstraction constraints*, which permit a generalization of an item to a concept of the ontology. By using pruning constraints, one can specify the exclusion of a set of items from the input transactions set, and, as a consequence, from the extracted rules. This kind of constraints refers either to a single item, or to an ontology concept, and they can include a condition expressed on a set of ontology properties. Abstraction constraints permit exploring different levels of the ontology concepts. The generalization to a predefined level of the hierarchy

$\mathcal{I}$ is the set of items $(i_1, i_2, ...i_n \in \mathcal{I})$.
$\mathcal{C}$ is the set of the concepts of the ontology $(c_1, c_2, ...c_n \in \mathcal{C})$.
$\mathcal{P}_c$ is the set of the properties of the concept $c \in \mathcal{C}$ $(p_1, p_2, ...p_n \in \mathcal{P}_c)$.
$cond_c$ is a Description Logic expression.
ALL represents all the instances defined in the ontology.

A constraint is defined on $\mathcal{I}$, $\mathcal{C}$ and $\mathcal{P}_\mathcal{C}$ in the following form:

1. **Pruning Constraints.** A pruning constraint is of one of the following forms:
   (a) $prune(e)$, where $e \in \mathcal{I} \cup \mathcal{C} \cup \{ALL\}$.
   (b) $prune_{cond_c}(c)$, where $c \in \mathcal{C} \cup \{ALL\}$.
2. **Abstraction Constraints.** An abstraction constraint is of one of the following forms:
   (a) $abstract(e, c)$, where $e \in \mathcal{I} \cup \mathcal{C}$, $c \in \mathcal{C}$ and c is a super-concept of e.
   (b) $abstract_{cond_{c_1}}(c_1, c_2)$ where $c_1 \in \mathcal{C} \cup \{ALL\}$, $c_2 \in \mathcal{C}$ and c_2 is a super-concept of c_1.
   (c) $abstract^{l}_{cond_e}(e)$, where $e \in \mathcal{I} \cup \mathcal{C} \cup \{ALL\}$, and l is a non-negative integer indicating the level of the hierarchy; $cond$ can be unspecified.

Table 1. User constraints syntax.

improves the support of association rules, and consequently avoids the discovery of a massive quantity of useless rules, especially in case of sparse data.

The ontology query engine interacts with the ontology by performing the set $Q_\mathcal{D}$ of queries. The resulting $R_\mathcal{D}$ instances set, is used by the DB query engine for retrieving the instances that contain the filtered/abstracted/pruned items (i.e., the items specified in $R_\mathcal{D}$). The data base is the repository of the data to pass in input to the data mining tools. The box "Data Mining Tools" contains the tool for analyzing and processing the data. In our context we refer to a specific algorithm for extracting association rules, but we would like to point out that the system can operate with other kinds of DM tools. The support and the confidence measures are initially provided by the user.

4 Case study: a Market Basket Analysis application

In this section we show the results of a case study by using data taken from a national supermarket, and stored in a relational database (DB). The aim of this study is to construct and test the framework described in the previous section with real data and w.r.t. specific market analysis. In this case, the data consist of a set of purchase transactions $T = [transID, item]$, where *transID* is the cash voucher identification and *item* is the purchased item. The DB contains 775,000 transactions. According to the approach proposed in sec. 3, meta-data (description of the items) and data to analyze have been organized respectively in separate structures:

The ontology - contains the description of the items and their hierarchical organization. Starting from the DB structure (tables and fields)[3], we derived the OWL ontology schema mapping the fields of the DB tables in classes and properties of the ontology. Also, we automatically filled up the ontology with about 30,000 items, and their attributes (approximately 100).

Let us consider the item *Vodka Keglevich Melon*. The correspondent hierarchical structure and the list of the item attributes are shown in the table below.

Hierarchical Structure	**Attributes of Vodka**
⊙ *owl:Things*	*hasColour*: transparent;
∇ ⊙ *XXX Supermarket*	*hasAlcoholicContent*: high;
∇ ⊙ *L0 Foodstuffs and Drinks Department*	*hasFlavour*: Melon;
∇ ⊙ *L1 Drinks*	*hasBrand*: Keglevich;
∇ ⊙ *L2 Vodka*	*isFizzy*: No;
∇ ⊙ *L3 Spicy*	*hasPrice*: EUR 7.56;
⊙ *Vodka Keglevich Melon*	*hasSize*: 70 cl;

The DB - contains the transactions T.

The experimentation has been conducted using *SeRQL* ("*Sesame* RDF Query Language") [4] language for querying the ontology and the Apriori algorithm [1] for mining association rules. *SeRQL* is an RDF/RDFS query language that is currently being developed by Aduna as part of *Sesame* [5]. It combines the best features of other (query) languages (RQL, RDQL, N-Triples, N3) and adds some of its own. Sesame is a RDF database which can be employed to manage RDF triples.

In the first two tests we abstract all items to two upper levels (level L2 and level L1) for verifying what categories of items are bought together. In this way we abstract all items to only 14 high level concepts in the first case and to only 4 high level concepts in the second one. These abstraction constraints can be expressed respectively as:

$$\begin{array}{l} Query\ 1 \equiv abstract^{2}(ALL) \\ Query\ 2 \equiv abstract^{1}(ALL) \end{array}$$

The third test concerns an investigation for organizing a future promotional campaign during the holidays (Christmas and Easter). The focus is on typical sweets and cakes (with well-known brands) of the two holidays, and the alcoholic drinks. The objective is to verify how those articles are related. All kinds of sweets/cakes are abstracted to *Foodstuffs* (associated with the item brand) and all kinds of alcoholic drinks to *Drinks*. These constraints can be expressed as:

$$\begin{array}{l} Query\ 3 \equiv prune_{(\exists hasBrand.=_{null})}(ALL) \wedge abstract_{(\exists hasBrand.<>_{null})}(Alcoholic,\ Drinks) \\ \quad \wedge\ abstract_{(\exists hasBrand.<>_{null})}(Sweets,\ FoodStuffs) \\ \quad \wedge\ abstract_{((\exists hasRecurrence.=_{Easter}) \sqcup (\exists hasRecurrence.=_{Christmas}))}(Sweets,\ FoodStuffs) \end{array}$$

[3] We considered the DB table named *Marketing* that, for each article, specifies a hierarchical structure w.r.t. the department organization in the supermarket.

The part of the ontology schema (i.e. the part of the DL knowledge base) related to the *Query* 3 can be expressed by the following TBox fragment:

$$\begin{aligned}\mathcal{T}_3 = {} & EatableThing \sqsubseteq (\exists hasBrand.string) \sqcap (=1 hasBrand)\\ & \sqcap\ EatableThing \sqsubseteq (\exists hasRecurrence.string) \sqcap (\geq 0 hasRecurrence)\\ & \sqcap\ (Drinks \sqsubseteq EatableThing) \sqcap (FoodStuffs \sqsubseteq EatableThing)\\ & \sqcap\ (Alcoholic \sqsubseteq Drinks) \sqcap (Sweets \sqsubseteq FoodStuffs)\end{aligned}$$

According to the interpretation function $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ defined in section 2, the semantic interpretation of the conditions expressed by the *abstract* clauses is:

$$\begin{aligned}&\Bigg(\Bigg(Alcoholic \sqsubseteq (\exists hasBrand.<>_{null})\\ &\sqcup\ Sweets \sqsubseteq (\exists hasBrand.<>_{null})\\ &\sqcap \Big((\exists hasRecurrence.=_{Easter}) \sqcup (\exists hasRecurrence.=_{Christmas})\Big)\Bigg)\Bigg)^{\mathcal{I}}\\ &= Alcoholic^{\mathcal{I}} \cap \{x_a \mid \exists y_a.(x_a, y_a) \in hasBrand^{\mathcal{I}} \wedge y_a \neq null^{\mathcal{I}}\}\\ &\quad \cup\ Sweets^{\mathcal{I}} \cap \{x_s \mid \exists y_s.(x_s, y_s) \in hasBrand^{\mathcal{I}} \wedge y_s \neq null^{\mathcal{I}}\}\\ &\qquad\qquad \cap \Big(\{z \mid \exists w.(z, w) \in hasRecurrence^{\mathcal{I}} \wedge w = Easter^{\mathcal{I}}\}\\ &\qquad\qquad\quad \cup \{h \mid \exists k.(h, k) \in hasRecurrence^{\mathcal{I}} \wedge k = Christmas^{\mathcal{I}}\}\Big)\\ &= \{\mathcal{A}\} \cap \{x_a \mid \exists y_a.(x_a, y_a) \in \{(a, b_a)\} \wedge y_a \neq null\}\\ &\quad \cup \{\mathcal{S}\} \cap \{x_s \mid \exists y_s.(x_s, y_s) \in \{(s, b_s)\} \wedge y_s \neq null\}\\ &\quad \cap \Big(\{z_s \mid \exists w_s.(z_s, w_s) \in \{(p, r_p)\} \wedge w_s = Easter\}\\ &\qquad \cup \{h \mid \exists k_s.(h_s, k_s) \in \{(q, r_q)\} \wedge k_s = Christmas\}\Big)\\ &= \{\mathcal{A}\} \cap \{(a, brand)\}\\ &\quad \cup \{\mathcal{S}\} \cap \{(s, brand)\}\\ &\quad \cap \Big(\{(p, Easter)\} \cup \{(q, Christmas)\}\Big)\\ &= \{(alcoholic, b_a)\} \cup \{(sweets_{Easter}, b_s)\} \cup \{(sweets_{Christmas}, b_s)\}\end{aligned}$$

where $\{\mathcal{A}\}$ and $\{\mathcal{S}\}$ are the instances sets of the classes *Alcoholic* and *Sweets* respectively, with $a \in \{\mathcal{A}\}$ and $s, p, q \in \{\mathcal{S}\}$; b_a, b_s are any well-known brands of *Alcoholic* and *Sweets* respectively. The semantic expressed by *prune* clause is very similar to *abstract* so we omit it for lack of space.
In the last test, we consider the case in which the supermarket augments its services by introducing a new department (*Assisted Service*). This event introduces an innovation in the supermarket domain, so we have to modify the ontology[4] i.e. we have to introduce a new data property, for some category (*typeOfService* (ToS) with enumerated type *Assisted Service*, *Take Away*, *Free Service*). We abstract to level *L*2 all the items with *typeOfService* equals to *Assisted Service* or *Take Away*, ignoring the others. This constraint can be expressed as:

[4] Notice that, the introduction of a new property does not imply the re-engineering of the structure, but only the introduction of the property in the higher classes so that the property is inherited by each subclasses.

$$
\begin{aligned}
Query\ 4 \equiv\ & abstract^2_{(\exists hasToS.=_{AssistedService})}(ALL) \\
& \wedge\ abstract^2_{(\exists hasToS.=_{TakeAway})}(ALL) \\
& \wedge\ prune_{((\exists hasTos.<>_{AssistedService})\sqcap(\exists hasTos.<>_{TakeAway}))}(ALL)
\end{aligned}
$$

For the lack of space we omit the semantic interpretation of the *Query*4.

For evaluating our framework we submitted to the system the queries introduced above. Our framework automatically translates these constraints into *SeRQL* language for querying the ontology. In all tests we applied the Apriori implementation of the KDDML System [10], setting the support threshold to 1%[5], and confidence to 50%. In Table 2, the five rows represent the results of the tests. The first query labeled no$_{constraints}$ represents the request without any constraints. #*Trans* reports the number of transactions that satisfy the constraints, #*Items* reports the total number of different articles that compose the transactions, #*Itemsets* and #*Rules* report the number of itemsets and the rules computed by the Apriori, respectively. Furthermore *LI* and *AI* contain statistical information about the number of items contained in the largest transaction, and the average number of items contained in a transaction. In figure 2

Query ID	**#Trans**	**#Items**	**#Itemsets**	**#Rules**	**LI**	**AI**
no constraints$_{Query0}$	91563	123	176	50	76	7.68
Test 1$_{Query1}$	80765	11	524	1248	12	3.86
Test 2$_{Query2}$	76323	4	15	31	4	2.83
Test 3$_{Query3}$	352	33	60	9	6	2.17
Test 4$_{Query4}$	69534	10	200	258	10	4.03

Table 2. Queries summary results

we report the supports graph of the queries. In the abscissa there are the top 50 frequent itemsets, while in the ordinate there is the support related to the i^{th} frequent item. As you can notice, in the picture the result of Test 2 has not been reported because it contains only 15 frequent itemsets. The use of real data typically brings issues related to the quality of the extracted model. Items at the lower levels of the taxonomy may not have enough support to appear in any frequent itemsets. This aspect is underlined in figure 2 in which we can notice that the *Query* 0 retrieves only itemsets with a very low support. This is mainly due to the large number of articles. Moreover, rules extracted at the lower levels of a concept, are too much specific, and may not be interesting. Consider for example the following rule extracted at low level:

$$\{bread,\ red\ wine,\ ham,\ chocolate\ cake\} \Rightarrow \{roasted\ chicken,\ cooked\ lasagne\} \quad [supp = 0.02, conf = 0.57].$$

The rule is not relevant due to the low support. Consider instead the following rule, that corresponds to the previous, but at an higher level of abstraction, and

[5] This low support threshold is dued to the large number of items.

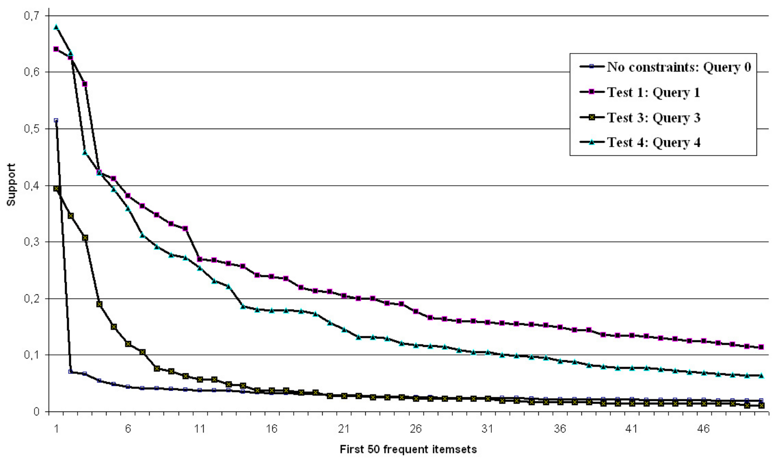

Fig. 2. Compared Supports.

satisfying *Query* 4:

$\{FoodStuffs\ AssistitedService\} \Rightarrow \{FoodStuffs\ TakeAway\}$

$[supp = 0.26, conf = 0.68]$.

This rule abstracts all the items to level L2 of the ontology and each of them is selected by the *typeOfService* property. The information extracted from this association rule can suggest that the assisted service department has to provide to the customers also (take away) cooked meals ($roastedchicken, cookedlasagne$). In general, items abstracted at the higher levels, tend to have higher support counts. This fact increases the quality of the extracted rules, and as consequence, helps the analyst in the decision support. Association rules related to *Query* 3, for example, emphasize the concept of multi-level rule correlating concepts at different abstraction level. For example the concept *FoodStuffs* (level L2) with $BAULI$ and $MOTTA$ as brands, and *Drinks* (level L2) with $ASTI$[6], are related to *Red Meats* (level L7) slaughtered and packed by the supermarket. It can suggest to the analyst some marketing decisions on these products during Easter or Christmas period.

The study of multi-level association rules is well-known in literature, and in this context, our work may not seem innovative. The focus of our approach is the introduction of the expressive power of ontologies for constraint-based multi-level association rule mining. The main advantages can be summarized in terms of *extensibility* and *flexibility*. Our framework is extensible because data properties and concepts can be introduced in the ontology without either changing the relational database containing the transaction, or the implementation of our framework. The flexibility is guaranteed from the separation of the data to

[6] $MOTTA$, $BAULI$ and $ASTI$ are Italian food and drink brands.

analyze (the transactions) from the meta-data (description of the data). Furthermore it interesting to point out that our approach is general, and can be adapted to further data mining analysis.

5 Related Works

Methods to define and integrate item constraints are originally introduced by Srinkant and Agrawal in [11] and by Han and Fu in [7]. Recently, in [12] and [9], we can find the attempt to integrate the item-constraints evaluation directly in the rule extraction algorithm. In [12], the authors concentrate on improving the Apriori algorithm, while in [9] the authors focus on the definition of a two-phase approach: specification of the constraint association queries, and submission of the constraints in the mining process.

Our approach follows the research line proposed by the cited works, nevertheless it introduces three main differences: (i) we employ an ontology to represent an item taxonomy; (ii) constraints can be defined on the basis of specific properties of the items; (iii) by using an ontology instead of a taxonomy, a new item property or a concept can be added without re-engineering the (meta-data) representation model or the relational database.

Other studies concern the merging of the association rules mining with a domain ontology. In [6], the authors use an ontology to improve the counting support during the association rule mining phase by using a taxonomy. Another interesting approach is presented in [8], where an ontology-based algorithm is employed for discovering rules of product fault causes, in an attempt to discover high-level clearer rules. In this case, the system enables the user only to specify an ideal level of generality of the extracted rules. In addition, our framework also enables the users to specify different levels of abstraction for different items, depending on the specific properties of such items. A concise syntax has been defined to this aim. In our view, the use of an ontology enforces constraints definition, enabling us to use data properties in domain-specific constraints.

6 Conclusions and future works

We proposed an integrated framework for the extraction of constraint-based multi-level association rules with the aid of an ontology. Our system permits the definition of domain-specific constraints by using the ontology for filtering the instances used in the association rule mining process. The main advantages of the proposed framework can be summarized in terms of *extensibility* and *flexibility*.

In our case study, the supermarket domain is modeled only by classes and data properties and it would be very interesting to study: (i) how object properties (and more complex logical relationships) can be employed in our framework; (ii) what aspects they can improve. Other important future works are the possibility of modeling the antecedent and the consequent of an association rule as ontology concepts in order to express constraints on the association rules structure. Furthermore we could improve the system by integrating the constraints evaluation directly in the mining algorithm.

Acknowledgement. This work is supported by *MUSING* project (www.musing.eu/).

References

[1] Agrawal, R., Methta, M., Shafer, J., and Srikant, R.: Fast algorithms for mining association rules in large databases. In Proceedings of the 20th International Conference on Very Large Databases (VLDB '94), Santiago de Chile, Chile, pp. 478-499.

[2] Agrawal, R., Srikant, R., and Swami, A.: Mining association rules between sets of items in large databases. In Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD '93), San Diego, CA, pp. 207-216.

[3] Baader, F., Calvanese, D.,McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook. Cambridge University Press (2003).

[4] Broekstra, J., Kampman, A.: SeRQL: An RDF Query and Transformation Language, http://www.cs.vu.nl/ jbroeks/papers/SeRQL.pdf, 2004.

[5] Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In Ian Horrocks and James Hendler, editors, Proceedings of the first International Semantic Web Conference (ISWC 2002), number 2342 in Lecture Notes in Computer Science, pages 5468, Sardinia, Italy, June 9 12, 2002. Springer Verlag, Heidelberg Germany.

[6] Chen, X., Zhou, X., Scher, R., and Geller, J.: Using an interest Ontology for Improved Support in Rule Mining. In Proceedings of the 5th International Conference of Data Warehousing and Knowledge Discovery (DaWaK 2003), Prague, Czech Republic, pp. 320–329

[7] Han, J., and Fu, Y.: Discovery of multiple-level association rules from large databases. In Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95) San Francisco, CA, pp. 420–431

[8] Hou, X., Gu, J., Shen, X., and Yan, W.: Application of Data Mining in Fault Diagnosis Based on Ontology. In Proceedings of the 3rd Conference on Information Technology and Applications (ICITA '05), Sydney, Australia, pp. 260–263.

[9] Ng, R., T., Lakshmanan, L., V., S., Han, J., and Pang A.: Exploratory mining and pruning optimizations of constrained associations rules. In Proceedings of the 1998 ACM SIGMOD international conference on Management of data (SIGMOD Seattle, WA, pp. 13–24.

[10] Romei, A., Ruggieri, S., and Turini, F.: KDDML: a middleware language and system for knowledge discovery in databases, Data and Knowledge Engineering, 57(2):179–220, 2006.

[11] Srikant, R., and Agrawal, R.: Mining Generalized Association Rules, In Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95), San Francisco, CA, pp. 407–419.

[12] Srikant, R., Vu, Q., and Agrawal, R: Mining Association Rules with Item Constraints In Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining (KDD '97), Newport Beach, CA, pp. 67–73.

[13] Tan, P.N., Steinbach, M., and Kumar, V.: Introduction to Data Mining, Pearson International Edition - Addison Wesley, 2006.