PUSHING CONSTRAINTS IN ASSOCIATION RULE MINING: AN ONTOLOGY-BASED APPROACH

Andrea Bellandi
IMT – Lucca Institute for Advanced Studies
Piazza S. Ponziano, 6 – 55100 Lucca, ITALY

Barbara Furletti
IMT – Lucca Institute for Advanced Studies
Piazza S. Ponziano, 6 – 55100 Lucca, ITALY

Valerio Grossi
Department of Computer Science – University of Pisa
Largo B. Pontecorvo, 3 – 56127 Pisa, ITALY

Andrea Romei
Department of Computer Science – University of Pisa
Largo B. Pontecorvo, 3 – 56127 Pisa, ITALY

ABSTRACT

This paper proposes an integrated framework for the extraction of constraint-based multi-level association rules with the aid of an ontology. The latter, which represents an enriched taxonomy, is used to describe the application domain by means of data properties. Defining or updating these properties is a simple task and does not imply changing the items hierarchy, or the implementation level of our framework. The system enables the definition of domain-specific constraints, by using the ontology to filter the instances used in the association rule mining process. This can improve the quality of the extracted association rules and make them more interesting and easy to understand. We describe our framework, also including examples of queries based on real-data.

KEYWORDS

Data Mining, association rules, ontology, domain-specific constraints.

1. INTRODUCTION

Data Mining (DM) is an emerging field that covers a wide range of application domains, such as marketing, finance, e-commerce, biology and privacy among the others. DM results are typically models representing relations in the data. The extracted models are usually employed to classify new data, or to describe correlations hidden in the data. The accuracy of these models is heavily influenced by data quality.

In this paper, we focus on *Association Rule Mining* as originally introduced in (Agrawal, R. et al, 1993). There are several ways to reduce the computational complexity of Association Rule Mining and to increase the quality of the extracted rules: (i) by covering the search space more efficiently; (ii) by exploiting efficient data structures; (iii) by 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. For example, (Ng, R., T. et al, 1998) supports constraint-based human-centered exploratory mining of association rules. This paper

focuses on these domain-specific optimizations providing a set of constraints, which enable the users to clearly specify which associations are to be mined during the extraction process.

With the advent of the semantic web (Berners-Lee, T. et al, 2001), ontologies have set to play a key role, by providing a source of shared and precisely defined terms, which can be used to describe resources and domains of applications. They express the concepts in the domain, as well as the relationships holding between them. In our view, they are used to capture the knowledge about a domain of interest and to increase the reliability and utility of the models.

The ontology represents a taxonomy enriched with data properties. A concept models a category of a set of items, e.g. *bread* and *milk* are kinds of *food*. Each data property describes an item feature at the considered level, e.g. *hasTypeFlour* is a feature of the items belonging to the *bread* concept only, while *hasPrice* is a feature of the items belonging to the *food* concept. It is important to point out that the ontology is not the repository of data, but it represents the component for modelling the domain (i.e. the metadata). Adopting an ontology, the domain expert is able to focus the analysis only over a single fragment of the data, without considering all specific items. This enables to reduce the number of generated association rules and, usually, to increase the quality of the extracted model.

In order to do this, the domain expert introduces a set of constraints, which can be divided into two categories, namely *abstraction constraints* and *pruning constraints*. The former are used to define a generalization of some items on a concept of the ontology. The latter category is used to exclude items from the analysis. Both are defined on the basis of conditions expressed on data properties. In this work we do not consider other classes of constraints, such as convertible (Pei, J. et al, 2001), succinct (Han, J. et al, 2002) or loose (anti-)monotone (Bonchi, F. et al, 2005).

Paper Organization. Section 2 introduces the syntax of the constraints and the description of the process for pushing constraints in Association Rule Mining. Section 3 presents an example of use based on a real-life dataset. Section 4 discusses related works. Finally, section 5 concludes the paper and suggests some ideas for further improvements. In the remaining of the paper, we use the terms of *ontology*, *hierarchy* and *taxonomy* as synonymous.

2. DESCRIPTION OF THE APPROACH

In this section, we describe our approach to guide the process of multi-level constraint-based Association Rule Mining. Our scenario consists of an ontology O_D, describing the domain of interest D and representing a hierarchy of concepts, and of a set of constraints V_D, defined on O_D. As mentioned earlier, we essentially consider membership constraints in the context of the hierarchy.

The process starts by specifying the ontology driven by a domain expert. In our framework, the ontology is not meant as an input data repository, but is used to model the domain of interest. Once the ontology has been outlined, the analyst[1] can submit a set of constraints related to concepts and properties of the defined domain. The aim of these constraints is to exclude some items from the extracted association rules, or to characterize interesting items according to an abstraction level.

The syntax of the two kinds of constraints introduced in this paper is formalized in figure 1. It includes both pruning constraints, i.e. used to filter a set of non-interesting items, and abstraction constraints, i.e. which enable a generalization of the item on a concept of the ontology. We now informally describe each constraint by an example.

By using pruning constraints, one can specify a set of items which can be excluded 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 can include a condition expressed on a set of ontology properties. There exist two kinds of pruning constraints.

1. The *prune(e)* constraint excludes a single item or all the items belonging to an ontology concept expressed by *e*. As one could expect, removing an ontology concept $c \in C$ also implies excluding all the items belonging to the descendent concepts of *c*.

[1] The domain expert and the analyst can be different users. In the remaining of the paper, these terms are used as synonymous.

$\mathcal{I}$ 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 combination (conjunction or disjunction) of triples (p, θ, v), where $p \in \mathcal{P}_c$, θ is one of the boolean operators $=, \neq, <, \leq, >, \geq$ and v is a constant from the domain of p.

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) $abstract_{cond<c1>}(c_1, c_2)$ 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.

Figure 1. Syntax of constraints constructs

2. The $prune_{cond<c>}(c)$ constraint is similar to *prune(e)*, except that uninteresting items are selected according to a condition defined on a set of properties of *c*. The condition can also refer to a property of an ancestor of *c*.

For example, let us suppose to have a simple ontology, modelling DVD movies sale domain. Every item represents a DVD title and has only one property, *hasLength*, indicating movie run time. The ontology has the form of a taxonomy organized according to the genre (e.g. adventure, thriller, etc.) of the movie. The predicates *prune(thriller)* and $prune_{hasLength>120}(romantic)$ allow us, respectively, to prune all the "thriller" movies and all the "romantic" movies longer than 120 minutes from the input transactions. Notice that *hasLength* is a property of the upper concept *movie*, and *romantic* is a sub-concept of *movie*.

Abstraction constraints enable to explore different levels of the ontology concepts. The generalization to a predefined level of the hierarchy increases the support of association rules, and consequently avoids the discovery of a massive quantity of useless rules, especially in case of sparse data. More interesting is the exploitation of rules in which items are extracted at different levels of the hierarchy. The system offers three different predicates:

1. The *abstract(e, c)* predicate generalizes a single item *e* (or all the items belonging to the concept *e*) to the super-concept *c*.
2. $abstract_{cond<c1>}(c_1, c_2)$ is similar to the previous one, but it generalizes items according to a condition ($cond<c_1>$) defined on a set of properties of the concept c_1. Only the items satisfying the condition are abstracted to the super-concept c_2. As for $prune_{cond}(c)$, this condition can also refer to a property of an ancestor of c_1.
3. Finally, $abstract^l_{cond<e>}(e)$ generalizes every element satisfying the condition $cond<e>$ to the super-concept placed at level *l* of the hierarchy[2]. In this case, *e* can be either a single item or a concept. If the condition is not specified, all the items belonging to *e* are generalized to the super-concepts at level *l*.

The symbol '*ALL*' in figure 1 can be used to select all the items defined in the ontology.

Let us assume that *fantasy* is a sub-concept of *adventure*. The predicate *abstract(fantasy, adventure)* generalizes all movies belonging to *fantasy* as a more generic item *adventure*. It is equivalent to the predicate $abstract^2(fantasy)$, if *adventure* is placed at level 2 of the ontology.

The set of constraints V_D is therefore a conjunction of pruning and abstraction constraints.

Once the ontology and the constraints have been defined, the *constraints interpretation phase* starts. This step is achieved by querying the ontology, in order to retrieve all the items involved which satisfy the constraints. From the logical point of view, the result of the interpretation phase is a set of pairs $\mathsf{S} = \{(\mathsf{S}_1, \mathsf{S}_2) \mid V_D\}$, where $\mathsf{S}_1 \in \mathit{I}$ and $\mathsf{S}_2 \in \mathit{I} \cup \mathit{C} \cup \{\mathtt{nil}\}$. Information on how to treat items during the rule extraction

[2] We suppose that the root of the hierarchy has level 0.

phase is annotated to each item by means of the second element of the pair. Thus, an item can be (i) abstracted to a super concept $c \in C$; (ii) ignored during mining (by using *"nil"* as second element of the pair) or (iii) considered at the lowest level of the taxonomy.

In the *transformation phase*, S is used to transform the transactions, according to constraints provided by the analyst. This phase scans the database, and, for each item, it filters (or abstracts) the items, as stated in S. Typically, at the end of this step, the total number of transactions is reduced. The correctness of this task is guaranteed, if every item in the database transaction is described in the ontology: if, and only if, an item appears in the set S_1, it must be abstracted (or deleted) from the transactions. Obviously, this step can radically change the value of itemset support, and consequently the frequent itemsets discovered. In other words, it introduces a sort of approximation on the generated rules, since some items are deleted, while others can be abstracted or remain the same.

Finally, the transformed data are used as input to the rule extraction algorithm, together with the support and confidence measures provided by the user (*rules extraction phase*). The result of this step is a set of association rules satisfying the constraints defined in the *constraints interpretation phase*. This output is returned back to the analyst. Figure 2 summarizes the process.

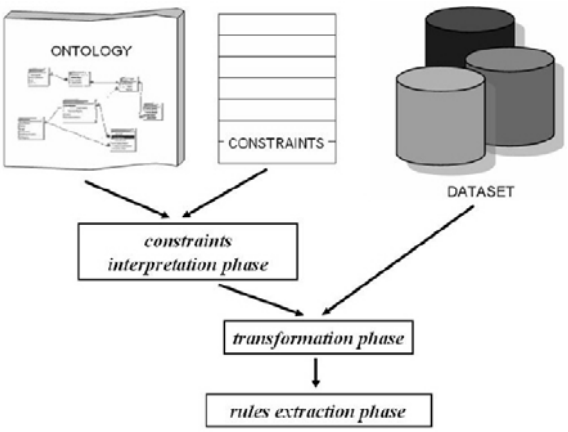

Figure 2. The three phases approach of the framework

3. AN EXAMPLE ON THE MARKET BASKET ANALYSIS DOMAIN

In this section we introduce an example of usage with data taken from a national supermarket, and stored in a relational database (DB). The aim is to construct and test the framework described in the previous section with real data and related to a specific market analysis. The data consist of a set of tuples *T = [transID, item]*, where *transID* is the transaction identification and *item* is the purchased product. The DB contains 775,000 transactions.

According to the approach proposed in section 2, metadata have been organized into an ontology, well-separated from the transactions to analyze. The ontology does not contain only the hierarchical organization of the items, but also their properties. Starting from the DB structure, we derived the ontology schema mapping the fields of the DB tables in classes and properties of the ontology. The latter was filled up with about 30,000 items, with approximately 100 attributes. The ontology has been built by using RDF (Resource Description Framework) syntax, the W3C (World Wide Web Consortium) recommended standard.

As an example, let us consider the item *"Vodka Keglevich Melon"*. Its properties and the corresponding hierarchical structure are reported below:

Hierarchical Structure	Attributes of Vodka
⊙ *owl:Things* ▽ ⊙ *XXX Supermarket* ▽ ⊙ *L0 Foodstuff and Drinks Department* ▽ ⊙ *L1 Drinks* ▽ ⊙ *L2 Vodka* ▽ ⊙ *L3 Spicy* ⊙ *Vodka Keglevich Melon*	*hasColour*: transparent; *hasAlcoholicContent*: high; *hasFlavour*: Melon; *hasBrand*: Keglevich; *hasRecurrence*: none; *hasType*: Alcoholic; *hasPrice*: EUR 7.56; *hasSize*: 70 cl;

Example 3.1. In the first test we abstract all the items to an upper level (level L2), to verify what categories of items tend to be bought together. Thus, we abstract all the items to only 14 high-level concepts. This constraint can be expressed as:

Query 1 ≡ $abstract^2(ALL)$

By way of example, if the ontology is made by alcoholic products, and we are not interested in the specific brands of alcoholics which are bought together, using Query 1, we can discover only the rules related to alcoholic types, such as vodka, wine and beer.

Example 3.2. The second test concerns an investigation to organize an advertising campaign during the holidays (Christmas and Easter). The focus is on typical sweets and alcoholic drinks (with well-known brands) of the two holidays, to verify how those products are related. Therefore, all kinds of sweets with *hasRecurrence* not equal to *'Easter'* or *'Christmas'* are abstracted to the *Foodstuff* concept. In the same way alcoholic drinks are abstracted to the *Drinks* concept. The remaining products are simply abstracted to level L2. This constraint can be expressed as:

Query 2 ≡ $prune_{(hasBrand=null)}(ALL) \wedge abstract^2_{(hasType\neq'Alcoholic') \wedge (hasType\neq'Sweets')}(ALL) \wedge$
$abstract_{(hasRecurrence\neq'Easter') \vee (hasRecurrence\neq'Christmas')}(Alcoholic, Drinks) \wedge$
$abstract_{(hasRecurrence\neq'Easter') \vee (hasRecurrence\neq'Christmas')}(Sweets, Foodstuff)$

Example 3.3. We consider the case in which the supermarket expands its service, by opening a new department (assisted service). This event introduces an innovation in the supermarket domain, so we have to modify the ontology. This new property does not imply the re-engineering of the structure, but only the introduction of the property in the opportune higher classes, so that the property is inherited by each subclass. In this example, we have added a new data property, only for some product categories, namely *typeOfService (ToS)* with enumerated type *'Assisted Service'*, *'Take Away'*, *'Free Service'*. Then, all the items with *typeOfService* equal to *'Assisted Service'* or *'Take Away'* are abstracted to level L2, while the others are ignored. This query can be expressed as:

Query 3 ≡ $abstract^2_{(hasToS='AssistedService')}(ALL) \wedge$
$abstract^2_{(hasToS='TakeAway')}(ALL) \wedge$
$prune_{(hasToS\neq['TakeAway', 'AssistedService'])}(ALL)$

The experiment has been conducted following the steps shown in figure 2 and explained below. It must be pointed out that this process is completely automatic.

```
SELECT myItems, TypeOfService
FROM { } rdf:type {market: EatableThing};
                  market: hasToS {TypeOfService}
WHERE  TypeOfService = "AssistedService" OR
                  TypeOfService = "TakeAway"
USING NAMESPACE market = <http:\\www.supermarketOntology.com>
```

Figure 3. SeRQL translation of Query 3

Constraints interpretation phase. It consists in the translation of the constraints above defined into SeRQL (Sesame RDF Query Language) (Broekstra, J. and Kampman, A., 2004) language for querying the ontology.

Table 1. Queries summary results.

Query ID	# Trans	# Items	# Itemsets	# Rules	LI	AI
No constraints	91.563	123	176	50	76	7,68
Query 1	80.765	11	524	1248	12	3,86
Query 2	352	33	60	9	6	2,17
Query 3	69.534	10	200	258	10	4,03

SeRQL is an RDF/RDFS query language which is currently being developed by Aduna as part of Sesame (Broekstra, J. et al, 2002). It combines the best features of other (query) languages (RQL[3], RDQL[4], N-Triples[5], N3[6]) and adds some of its own. Sesame, a RDF database which can be employed to manage RDF triples, is here used to load and query the ontology, in order to retrieve and label the items which satisfy the constraints. The translation of Query 3 into SeRQL is shown in figure 3.

The result of this phase is a set of pairs as described in section 2. By way of example, the interpretation of Query 2 produces (among the others) the pairs *(XXX Ice Cream, Foodstuff)*, *(XXX Energy Drink, Drinks)* where *XXX Ice Cream* and *XXX Energy Drink* belong to the *Sweets* and to the *Alcoholics* concept, respectively.

Transformation phase. In this phase, the transactions are read from the database and, according to the label provided at the previous step, each item of the transaction is filtered out, abstracted to a higher level of the hierarchy or not transformed.

Rule extraction phase. We apply the Apriori (Agrawal, R. et al, 1994) implementation of the KDDML System (Romei, A. et al, 2006), setting the support threshold to 1%, and confidence to 50%. In table 1, the five rows represent the results of the queries. The first query represents the request without any constraints. *#Trans* reports the number of transactions that satisfy the constraints, *#Items* reports the total number of different products that compose the transactions, while *#Itemsets* and *#Rules* report the number of itemsets and the rules calculated by the Apriori, respectively. Furthermore, LI and AI contain statistical information about the number of items of the largest transaction, and the average number of items contained in a transaction.

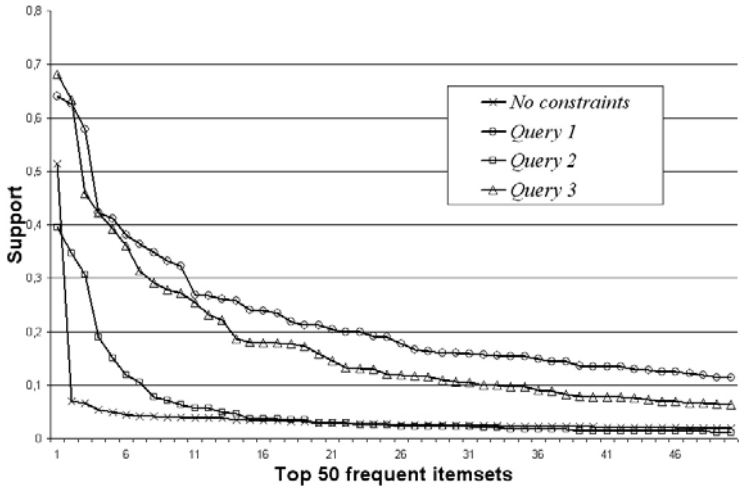

Figure 4. Compared Top 50 Frequent Itemsets

Figure 4 reports the graph of the queries. In the abscissa, there are the top 50 frequent itemsets ordered by support value, while the ordinate presents the support related to the ith frequent itemset. The use of real data typically introduces 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. Typically, this is mainly due to the sparsity of the transactions. This aspect is underlined in figure 4, which shows that the query without constraints retrieves only itemsets with a very low support. However, we can observe that items abstracted at the higher levels tend to have higher support counts, as highlighted in figure 4. This fact usually increases the quality of the extracted rules and, as a consequence, helps the analyst in the decision support.

Discussion. The study of multi-level association rules is well-known in the literature and in this context our work may not seem innovative. The core 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, since data properties and concepts can

[3] http://139.91.183.30:9090/RDF/RQL/
[4] http://www.w3.org/Submission/RDQL/
[5] http://www.w3.org/2001/sw/RDFCore/ntriples/
[6] http://www.w3.org/DesignIssues/Notation3

be introduced in the ontology without changing the relational database containing the transaction, or the implementation level of our framework. Flexibility, on the other hand, is guaranteed by the separation of the data to analyze (the transactions) from the meta-data (description of the data). Adding a new user-defined constraint on a concept is quite simple, since it involves only updating the ontology with a new property placed in correspondence of the desired concept of interest (see example 3.3). As this simple Market Basket Analysis application has highlighted, Association Rule Mining tasks can be coupled with an ontology to find more relevant rules. As one could expect, the quality of the ontology describing the domain is quite important to obtain good results. Furthermore, it is interesting to point out that our approach is general, and can be adapted to a wide range of data mining analyses.

4. RELATED WORKS

In the literature, several works focus on the discovery of association rules with item constraints. The latter have a double application. On the one hand, they are employed to limit candidate generation and counting support in finding frequent itemsets. On the other hand, they improve the comprehension of the extracted rules, with the aid of a taxonomy. Methods to define and integrate item constraints are originally introduced by Srinkant and Agrawal in (Srikant, R. and Agrawal, R., 1995) and by Han and Fu in (Han, J. and Fu, Y., 1995). (Srikant, R. et al, 1997) and (Ng, R., T. et al, 1998) can be viewed as attempts to integrate item-constraints evaluation directly in the rule extraction algorithm. In (Srikant, R. et al, 1997), the authors concentrate on improving the Apriori algorithm, while (Ng, R., T. et al, 1998) focuses on defining a two-phase approach, first proposing a model whereby the user specifies a constraint association query to the system, and after pushing the constraints inside the mining process.

Even though our approach is similar to the ones previously outlined, 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; and more importantly, (iii) compared to the ordinary use of a taxonomy, in our framework a new specific item (or concept) property can be added without changes in the system implementation level.

Recent studies have been conducted to merge the field of Association Rule Mining with the ontology domain. In (Zhou, X. and Geller, J., 2007), the authors use an ontology to improve counting support during the association rule mining phase, by means of concept hierarchy raising. Another interesting approach is outlined in (Hou, X. et al, 2005), where an ontology-based algorithm is employed to discover rules of product fault causes, in an attempt to mine 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 both papers, all the items are generalized by going up to the specified level of abstraction of the ontology.

In addition, our framework also enables 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.

Finally, in (Shen, B. et al, 2006) the authors implement a prototype system by using Protégé tools. Basically, they mainly offer an intelligent rule retrieval service that operates on various forms of association rules, stored in distributed heterogeneous rulebases.

5. 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, which represents a specific domain. Our system enables the definition of constraints and it applies the ontology to preprocess the transactions used as input of the association rule mining phase. The framework has an architecture in which the data mining engine is decoupled with the domain specification. The main advantages of the proposed technique can be summarized in terms of extensibility and flexibility: one could change the algorithmic details or extend the ontology, without changing the implementation level nor the database transactions.

Ontologies express the concepts in the domain, as well as the relationships holding between them. These relationships can bind two concepts (object property), or one concept to a data type (data property). In this work, only data properties are used. Since object properties enhance expressiveness with respect to the use of data properties only, it is interesting to study: (i) how object-properties can be employed in our framework; (ii) what aspects they can improve. In this view, another important future work is represented by the possibility to include more expressive constraints in our framework, as those proposed in (Pei, J. et al, 2001), (Han, J. et al, 2002), (Bonchi, F. et al, 2005). Furthermore, we could improve the system, by integrating the constraints evaluation directly during the mining phase.

ACKNOWLEDGEMENT

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

REFERENCES

Agrawal, R. et al, 1993. Mining association rules between sets of items in large databases. *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD '93)*. San Diego, CA, pp. 207-216.

Agrawal, R. et al, 1994. Fast algorithms for mining association rules in large databases. *Proceedings of the 20th International Conference on Very Large Databases (VLDB '94)*. Santiago de Chile, Chile, pp. 478-499.

Berners-Lee, T. et al, 2001. The Semantic Web, *Scientific American*, Vol. 284, No. 5, pp. 34-43.

Bonchi, F. et al, 2005. Pushing Tougher Constraints in Frequent Pattern Mining. *Proceedings of the Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'05)*. Hanoi, Vietnam, pp. 114-124.

Broekstra, J. et al, 2002. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. *Proceedings of the first International Semantic Web Conference (ISWC 2002)*. Sardinia, Italy, pp. 54-68.

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

Grahne, G. et al, 2000. Efficient mining of constrained correlated sets. *Proceedings of 16th International Conference on Data Engineering (ICDE' 00)*. San Diego, CA, pp. 512-524.

Han, J. and Fu, Y., 1995. Discovery of multiple-level association rules from large databases. *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95)*. San Francisco, CA, pp. 420-431.

Han, J. et al, 2002. Mining Top-k frequent closed patterns without minimum support. *Proceedings of 2002 IEEE International Conference on Data Mining (ICDM'02)*. Maebashi City, Japan, pp. 211-218.

Hou, X. et al, 2005. Application of Data Mining in Fault Diagnosis Based on Ontology. *Proceedings of the 3rd Conference on Information Technology and Applications (ICITA '05)*. Sydney, Australia, pp. 260-263.

Ng, R., T. et al, 1998. Exploratory mining and pruning optimizations of constrained associations rules. *Proceedings of the 1998 ACM SIGMOD international conference on Management of data (SIGMOD '98)*. Seattle, WA, pp. 13-24.

Pei, J. et al, 2001. Mining frequent item sets with convertible constraints. *Proceedings of 17th International Conference on Data Engineering (ICDE'01)*. Heidelberg, Germany, pp. 433-442.

Romei, A. et al, 2006. KDDML: a middleware language and system for knowledge discovery in databases, *Data and Knowledge Engineering*, Vol. 57, No. 2, pp. 179-220.

Shen, B. et al, 2006. Ontology-based Association Rules Retrieval using Protégé Tools. *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006)*. Hong Kong, China, pp. 765-769

Srikant, R. and Agrawal, R., 1995. Mining Generalized Association Rules. *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95)*. San Francisco, CA, pp. 407-419.

Srikant, R. et al, 1997. Mining Association Rules with Item Constraints. *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining (KDD '97)*. Newport Beach, CA, pp. 67-73.

The Resource Description Framework home page: http://www.w3.org/RDF/.

The World Wide Web Consortium (W3C): http://www.w3.org/.

Zhou, X. and Geller, J., 2007: Raising to Enhance Rule Mining in Web Marketing with the Use of an Ontology. *Data Mining with Ontologies: Implementations, Findings and Frameworks*, editors: Nigro, H., O., Cisaro, S., G., Xodo, D., IGI Global, USA, to appear.