

Local Mining of Association Rules with Rule Schemas

Andrei Olaru

Claudia Marinica

Fabrice Guillet

LINA, Ecole Polytechnique de l'Universite de Nantes
rue Christian Pauc BP 50609 44306 Nantes Cedex 3

E-mail: cs@andreiolaru.ro, {claudia.marinica, fabrice.guillet}@univ-nantes.fr

Abstract

One of the central problems in Knowledge Discovery in Databases, more precisely in the field of Association Rule Mining, relies on the very large number of rules that classic rule mining systems extract. This problem is usually solved by means of a post-processing step, that filters the entire volume of extracted rules, in order to output only a few potentially interesting ones. This article presents a new approach that allows the user to explore the rule space locally, incrementally, without the need to extract and post-process all rules in the database. This solution is based on Rule Schemas, a new formalism designed in order to improve the representation of user beliefs and expectations, and on a novel algorithm for local association rule mining starting from Schemas. The proposed algorithm was tested successfully over the database provided by Nantes Habitat¹.

1. Introduction

Knowledge Discovery in Databases is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data [8]. The process of knowledge discovery is a complex one, comprising several phases that deal with problem and dataset focusing, data quality and cleaning, pattern or rule mining, evaluation, explanation, reporting and visualization of discovered knowledge. One of the most important techniques in Knowledge Discovery is Association Rule Mining [1]. An Association Rule is an implication of the form $a \rightarrow b$, that shows that the presence of itemset a in a transaction implies, with a certain confidence c , the presence of itemset b in the same transaction.

Association rules may provide valuable information in databases. Nevertheless, the number of rules extracted by

knowledge discovery systems is so large that it is impossible to inspect them manually. Moreover, most of the discovered rules are not interesting to the user, since he/she might already know or might not be interested in them, or might not be able to perform any action as a consequence of their discovery ([18], [19]). Generally, the interestingness of rules depends on statistical measures as the support and the confidence in the database (the *objective* aspect of interestingness), but, more important, it depends on the database domain, on user background knowledge and on user expectations (the *subjective* aspect of interestingness). Two main subjective measures of interest exist: unexpectedness – rules are surprising to the user – and actionability – the user is able to take action based on their discovery.

Focusing on the interesting rules can be done in the rule mining phase of the KDD process or in the post-processing phase. The second approach is suggested as a solution for the rule reduction problem in most of the papers, but its main weakness is that the entire set of association rules must be mined, and then most of them are considered as not interesting and eliminated, the whole process being quite inefficient [19].

This paper presents a new approach, in which post-mining principles are introduced into the mining step, focusing on the interesting rules without the necessity of extracting all rules existing in the database. Instead of letting the user inspect huge amounts of output containing thousands of rules, the user may explore the rule space incrementally [5], starting from his own beliefs and knowledge and discovering rules that relate to these beliefs – confirming rules, specialized rules, generalized rules or exception rules. At each step, the user inspects only a small amount of rules and he/she is able to choose the most relevant ones, for further exploration. Global post-processing is avoided in favor of local, focused rule exploration.

There are two key issues in this approach. The first one is focusing on the expectations and beliefs of the user. The solution for representing the user's beliefs and knowledge is based on the concept of General Impressions, first presented in [12] and later developed in [14]. This paper proposes

¹We would like to thank Nantes Habitat, the Public Housing Unit in Nantes, France, and especially to Ms. Cristelle Le Bouter for supporting this work.

a novel, more unitary manner of representation – the Rule Schema, that is not only more flexible and intuitive, but can also use as base elements the concepts from an ontology, providing the representation with infinite possibilities. The ontological aspect is not treated in this paper. Four Operations on Rule Schemas have been proposed, that facilitate the exploration of the rule space: Confirmation, Specialization – discovery of rules with the same conclusion but a more specific condition and a notable improvement the confidence, Generalization – discovery of rules with a more general condition and higher support, Exception – discovery of low-support regularities that contradict more general rules [7].

The second important aspect of our approach is the mining algorithm. As it is inefficient to extract all rules in order to filter a few interesting ones, an algorithm is required that focuses on interesting rules at the time of extraction, in the mining step. Such an algorithm has been designed, that acts in a novel manner: based on the existing Rule Schemas and the Operations performed by the user, the algorithm generates all candidate rules – all possible rules that may result from applying the Operations to the Rule Schemas – and then checks their support against the database. This method is efficient, because the algorithm acts on a local scale, but provides globally valid results.

The proposed method has been tested on a real-life database, provided courtesy of the Nantes Habitat agency. Several interesting experiments have been carried out and relevant results have been obtained.

The paper is organized as follows. Section 2 presents the research domain and reviews related works. Section 3 describes the Rule Schema formalism and the Operations on Rule Schemas. Section 4 presents the algorithm for focusing on interesting association rules starting from Rule Schemas. Results are discussed in section 5. Finally, section 6 presents the conclusion.

2. Related Work

Subjective measures and the integration of user beliefs were first discussed in [19], in which the most important two subjective interest measures are introduced: unexpectedness and actionability. Different manners for the representation of user beliefs and expectations have been proposed, mainly related to the two measures above. [11] and [13] discuss the three cases of unexpectedness comparing discovered rules and previous knowledge: unexpected condition, unexpected conclusion, and both condition and conclusion unexpected.

The concept of rule templates is introduced in [10], as items in two lists: inclusion – rules that are interesting, and restriction – rules that are not interesting. However, all rules must be extracted and the search is not done locally.

Moreover, the formalism is very simple and not too flexible. [16] suggests a logical representation and comparison for user beliefs, but this approach is fairly limited. An important proposition for the representation of user beliefs is presented in [12] and later developed in [14]. It contains three levels of specification: General Impressions (GI), Reasonably Precise Concepts (RPC) and Precise Knowledge (PK). All three formalisms use items in a taxonomy, therefore allowing only for *is-a* relations between items. Moreover, using three different levels might be difficult to use, if the user wants to combine their features. For instance, the user might know that an item *A* leads to an item *B* and knowing that item *C* relates to them but without being sure on what side of the implication it might be. Representing this kind of knowledge is not possible with the formalism given in [14].

A very early paper in which the integration of domain knowledge is discussed is [3], that suggests using special structures for the integration of domain knowledge and constraints: *Hierarchical Generalization Trees (HG-Trees)*, *Attribute Relationship Rules (AR-rules)* and *Environment Based Constraints (EBC)*. Using item taxonomies is suggested in [20] and taxonomy items are also used in the General Impressions formalism in [14]. However taxonomies are limited to *is-a* relations. The advantages of using ontologies instead are presented in [17]. Different manners of integrating ontologies are possible [15], either in post-processing or in a pre-processing step, for filtering of transactions before mining [4].

From the great variety of pattern mining algorithms ([6], [9]) the Apriori algorithm, presented in [2], is the most popular. However, it is difficult to modify the pruning strategy in order not to extract all patterns, but only the ones that might be interesting to the user. In fact, most of the usual pattern-mining algorithms extract all patterns that have a minimum support. Moreover, most algorithms extract patterns, and not rules, association rules being built by using the mined itemsets.

Exceptions to a rule have been taken into account as interesting in [18], where the KEFIR system finds deviations from certain norms and references. [16] presents the algorithm ZoomUR that finds rules with a more specialized condition and unexpected conclusion. However in both algorithms the search is not done locally, but considers all rules in the database.

3. Focusing on interesting rules with Rule Schemas

3.1. Rule Schema formalism

Focusing on the rules that are interesting to the user means finding rules that are in a certain relation with his/her

current beliefs. This relation can be one of confirmation or one of contradiction. A formalism for representing the user's beliefs and knowledge has been designed: the *Rule Schema* is based on the three levels of specification proposed in [14], but comprises their advantages into one single level of specification. The Rule Schema represents what the user believes about the associative relations in the database. A Rule Schema is represented as follows:

$$rs(\textit{Condition} \rightarrow \textit{Conclusion} [\textit{General}]) \quad [s\% \ c\%]$$

The *Condition* and the *Conclusion* contain the items that the user believes that are present in the antecedent and, respectively, in the consequent of the rule. In complement, the *General* part contains the items that the user is not sure if they exist in the antecedent or in the consequent. The Rule Schema also contains optional constraints of support and confidence.

The three parts – *Condition*, *Conclusion* and *General* – are all Expressions of the following form:

$$\textit{Expr} = \{\textit{Expr}\} \mid [\textit{Expr}] \mid \textit{Expr}? \mid \textit{Item}$$

That is, an Expression may be a disjunction of other Expressions, a conjunction of Expressions, an optional Expression or an item from the database. The optionality operator applied over an Expression means that the items that are composing it may or may not appear in the rule.

This formalism completely covers the three levels of specification presented in [14]. If the *Condition* and *Conclusion* are used, the Schema is more like a Reasonably Precise Concept or Precise Knowledge. If the *General* part is used, the Schema is more like a General Impression. The improvement is that a Rule Schema may use all three parts.

Example. If the user knows that there is an implication between *Sausage* and *Mustard*, in the form *Sausage* → *Mustard*, and he/she believes that these elements are also associated with *Bread*, but he/she does not know on which side of the implication *Bread* exists, the corresponding Rule Schema is:

$rs([Sausage] \rightarrow [Mustard] [Bread])$, with no constraints.

Moreover, if the user has a feeling there might be some influence of *Diet_Coke* in the condition, and also wants to filter the output based on some support and confidence constraints, the new elements are added to the Rule Schema:

$$rs([Sausage, Diet_Coke?] \rightarrow [Mustard] [Bread]) \\ [10\% \ 75\%]$$

Confirmation of the Rule Schema above might output the following rules:

$$Sausage, Diet_Coke, Bread \rightarrow Mustard \quad [12\% \ 87\%] \\ Sausage \rightarrow Bread, Mustard \quad [17\% \ 78\%]$$

3.2. Operations

Several operations have been designed, that allow the user to explore the rule space starting from previous beliefs and knowledge.

Confirmation is the most simple of the operations that may be performed on Rule Schemas. It finds all rules that comply with the support and confidence constraints and contain the items in *Condition* in the antecedent, the items in *Conclusion* in the consequent and the items in the *General* part in any of the two sides of the implication. The items in the *General* part may be split in any possible way between the antecedent and the consequent.

More formally, the searched rules are of the form:

$$\textit{Condition} \cup \textit{Subset} \rightarrow \textit{Conclusion} \cup (\textit{General} - \textit{Subset}), \text{ where } \textit{Subset} \subseteq \textit{General}$$

Example. The Rule Schema $rs([Sausage] \rightarrow [Mustard] [Bread, Bagels])$ is confirmed by any of the four following rules:

$$Sausage, Bread, Bagels \rightarrow Mustard \\ Sausage, Bread \rightarrow Mustard, Bagels \\ Sausage, Bagels \rightarrow Mustard, Bread \\ Sausage \rightarrow Mustard, Bread, Bagels$$

During Confirmation, the Expressions present in the Rule Schema are transformed into lists of items. Disjunctive and optional expressions result into different groups of rules, each corresponding to one element in the disjunction. For example, if the condition contains an Expression of the form $[A \ B \ C?]$, the resulting rules contain in the condition *A, C* or *B, C* or just *A* or *B*.

Specialization allows the user to find the rules that have a more particular condition and the same conclusion, with the condition that the confidence of the specialized rule must be higher than the confidence of the more general rule. That is, a specialization of $a \rightarrow b [s1, c1]$ is $a \wedge c \rightarrow b [s2, c2]$, and it is required that $c2 > c1$.

The operation is not performed directly on the Rule Schema, but is preceded by a Confirmation of the Schema, in order to transform expressions into item lists, to split the *General* part and to calculate confidence and support. The Specialization is performed on rules in the output of the Confirmation operation. On the basis of the rules of the form $\textit{Condition} \rightarrow \textit{Conclusion}$ resulting from the Confirmation, searched rules for a given specificity level *k* are of the form:

$$\textit{Condition} \cup \textit{Set} \rightarrow \textit{Conclusion}, \text{ where } |\textit{Set}| = k \text{ and } \textit{Set} \subseteq (I - (\textit{Condition} \cup \textit{Conclusion})), \text{ with } I \text{ the full itemset.}$$

Example. For a Rule Schema $rs([A] \rightarrow [B] [C])$ and $I = \{A, B, C, D, E\}$ the output of the Specialization operation may be:

$$\begin{aligned}
&A \rightarrow B, C \quad [s1 \ c1] \\
&A, D \rightarrow B, C \quad [s2 \ c3] \\
&A, E \rightarrow B, C \quad [s3 \ c4] \\
&A, C \rightarrow B \quad [s1 \ c2] \\
&A, C, D \rightarrow B \quad [s2 \ c5]
\end{aligned}$$

In the example above, it is required that the confidence measures satisfy $c3 \geq c1$, $c4 \geq c1$ and $c5 \geq c2$. For support, it is obvious that $s2 \leq s1$ and $s3 \leq s1$.

As presented, different levels of Specialization are possible. For example, a specialization of level 2 of the rule $A \rightarrow B, C$ above might issue the rule $A, E, F \rightarrow B, C$ where not one, but two items are added to the condition.

Generalization is the opposite of Specialization. This operation finds the rules that have a more general condition implying the same conclusion. The support is expected to be higher and the confidence slightly lower. Formally, searched rules for level of generality k are of the form:

Condition – Set \rightarrow *Conclusion*, where $|Set| = k$ and $Set \subseteq Condition$.

For example, possible generalizations of a rule $A, B \rightarrow C$ are $A \rightarrow C$ and $B \rightarrow C$.

Exception is an important operation, as it finds rules with an unexpected conclusion, in the context of a more specialized condition. That is, for rules of the form $a \rightarrow b$ exceptions are of the form $a \wedge c \rightarrow \neg b$, with the restriction that b must be a single item, so it is possible to be negated.

There are some conditions for a good exception, as specified in [7]. In this paper it is considered that if the confidence of $a \rightarrow b$ is $c1$, the confidence of $a \wedge c \rightarrow \neg b$ is $c2$ and the confidence of $c \rightarrow \neg b$ is $c3$, then $c2$ must be higher than $c1$, and $c3$ must be fairly low, because it must not be c that leads to $\neg b$, but the association of a with c .

Exception may be considered as a Specialization of the rule with the original condition and a negated conclusion. If the attribute in the conclusion is multi-modal, all values are considered, except for the one in the original conclusion.

4. Local rule mining

4.1. Algorithm description

Following the idea of integrating user knowledge and expectations into the rule mining process, a first attempt was to integrate the Rule Schemas into the Apriori algorithm described in [2]. However, modifying the pruning strategy in order to focus on interesting rules, in conjunction with the operations described in the previous section, is not very effective and does not generate good results.

Therefore, a novel approach was used. In this approach, the search for interesting rules is done *locally*, in the neighborhood of rules and associations that the user already

knows, or that the user believes to be true, specified by means of the Rule Schemas. Instead of finding all valid rules in the database and then filtering them according to interestingness criteria, the algorithm acts in an inverse manner. First, it generates all possible interesting rules – rule candidates – based on the Rule Schemas. In the second step, rule candidates are checked for support and confidence requirements, against the transaction database. The final results are presented to the user ordered by their confidence. The steps taken by the algorithm are presented in Figure 1.

Example. Suppose the user wants to find all rules that are the level 1 Specialization of the Rule Schema $rs([A \rightarrow \{B, C\}][D])$ [10%, 60%]. That is, A leads to either B or C , and they are associated with D . Support and confidence must be over 10% and 60% respectively. The full item set is $I = \{A, B, C, D, E, F\}$. The algorithm work as follows:

- regardless of the operation, a set of rule candidates is created that result from the Expressions in the Rule Schema and from splitting the *General* part between condition and conclusion. The result is:

$$\begin{aligned}
&[A] \rightarrow [B \ D] \\
&[A \ D] \rightarrow [B] \\
&[A \ D] \rightarrow [C] \\
&[A] \rightarrow [C \ D]
\end{aligned}$$

- all results are tested against the database for support and confidence. Candidates failing to comply with support requirements (support is lower than 10%) are pruned:

$$\begin{aligned}
&[A] \rightarrow [B \ D] \quad [25\% \ 67\%] \\
&[A \ D] \rightarrow [B] \quad [25\% \ 44\%] \\
&[A] \rightarrow [C \ D] \quad [8\% \ 26\%] - \text{pruned} \\
&[A \ D] \rightarrow [C] \quad [8\% \ 35\%] - \text{pruned}
\end{aligned}$$

- based on the rules that result after pruning, and considering the items in the full item set I that are not already in the rules, all rule candidates are generated, obtaining a first level specialization. Support and confidence are checked and candidates are pruned if the measures do not comply with the requirements of the schema or if the confidence is not greater than the confidence of the more general rule.

$$\begin{aligned}
&[A \ C] \rightarrow [B \ D] \quad [7\% \ 20\%] - \text{pruned for support} \\
&[A \ E] \rightarrow [B \ D] \quad [17\% \ 62\%] - \text{pruned: confidence is not higher than confidence of ancestor} \\
&[A \ F] \rightarrow [B \ D] \quad [20\% \ 83\%] \\
&[A \ D \ C] \rightarrow [B] \quad [7\% \ 35\%] - \text{pruned for support} \\
&[A \ D \ E] \rightarrow [B] \quad [9\% \ 48\%] - \text{pruned for support} \\
&[A \ D \ F] \rightarrow [B] \quad [14\% \ 66\%]
\end{aligned}$$

- the remaining rules are sorted according to their confidence:

$$\begin{array}{l} [A F] \rightarrow [B D] \quad [20\% \ 83\%] \\ [A D F] \rightarrow [B] \quad [14\% \ 66\%] \end{array}$$

4.2. Algorithm efficiency

There are a number of advantages that this approach has, compared to the Apriori algorithm. They result in great part from the fact that the Rule Schemas are partially instantiated, so the search space is greatly reduced. Moreover, the more the user refines current Rule Schemas, the lower is the number of generated candidate rules.

Compared to Apriori, the number of passes through the databases is lower. Once all candidate rules are generated, only one pass through the database is necessary, to check the support of the candidates. For complexity reasons, in the case of multi-level operations one pass per specificity / generality level is necessary.

One important issue in the presented approach is the number of generated candidate rules. This depends on the operation and on the properties of particular Rule Schemas, as shown below.

For the operation of Confirmation on a Rule Schema $rs(X \rightarrow Y \ [Z])$ (where X, Y, Z are item sets), the number of generated rules is equal to the number of possibilities of splitting the Z set into subsets. The subsets are added to the left and right sides of the implication. The number of possibilities is equal to $2^{|Z|}$, the number of subsets in Z : for each subset S in Z , S is added to the condition and $Z - S$ to the conclusion. As searches are performed in the close neighborhood of the rule, the number of items in Z is fairly low.

For the operation of Specialization, all the rules of the form $X \cup S \rightarrow Y$ are generated, where S is not included in $X \cup Y$ and $|S|$ is the specificity level. Normally, the number of candidate rules would be $|I|^{|S|}$ where I is the set containing all items that appear in the database. However, there is a way of implementing this operation more efficiently, based on the idea of the Apriori algorithm. First, all the candidate rules with one item added to the condition are generated, then checked for support. This prunes most of the level 1 candidates. The level 2 candidates are obtained by adding one more item to the condition of the level 1 results and then are checked for support. This approach greatly reduces the total number of generated candidate rules.

For the operation of Exception, the rules of the form $X \cup S \rightarrow \bar{y}$ are generated, where y is a single item and S is a set of items not in X and not containing y . The number of candidate rules is of the same order with the Specialization operation, but, in the case of multi-modal attributes, it is multiplied with the number of modalities of the attribute y , minus 1. The same approach as with Specialization may be used for Exception.

For the operation of Generalization on a schema $X \rightarrow$

Y , candidate rules with fewer items in the condition are generated. For all levels of generality, $2^{|X|}$ candidate rules are generated, but X is not likely to have more than 5 or 6 items.

One more factor is added to the complexity of candidate rule number. If the expression lists contain disjunction operators, then candidate rules are generated for each term in the disjunction. However this is not likely to significantly increase the number of candidate rules as the expected number of terms in the disjunctions is quite low.

5. Results and discussion

An application was developed that implements the algorithm described above and allows the management of Rule Schemas. The application was tested on a real-life database. The database used was provided by Nantes Habitat, a public office that manages social accommodations in Nantes, France. Each year, 1500 Nantes Habitat clients (out of a total of 50000) answer a questionnaire about the quality of their accommodation. The questionnaire contained 78 questions in 2006. Nantes Habitat offered for data mining experiments the results obtained between the years 2003-2006. Each question refers to a certain element related to the accommodation and may be answered with one of 4 degrees of satisfaction.

In the database, there exists one attribute for each question. The possible values for the attributes are 1 to 4 for the satisfaction level, 95 and 96 for "not applicable", 99 for "don't know" and 0 if the client did not answer the question. The database contains one transaction for each questionnaire. The transaction is represented by the answers (or the values representing the lack of answer) to each of the questions. Because the algorithm needs binary attributes, items of the form question_answer are formed, for example the item Q35_1 represents an answer of "very satisfied" to the question number 35 – about the lighting of common spaces. Therefore there are a total of 624 items.

For testing two databases were used, from year 2003 and 2006, containing 1490 transactions each. Using the provided databases, some simple examples have been created and encouraging results have been obtained. The most interesting ones are presented below.

Example 1. The analyst is interested on the dissatisfaction related to common spaces in the proximity of the accommodation. The search starts with unsatisfied answers (value "4") to two questions related to the problem: Question 7 – "The quantity of green spaces is sufficient" and Question 65 – "The cohabitation difficulties are treated efficiently". The analyst already knows the direction of the implication (which is quite obvious) and creates a Rule Schema of the form:

Figure 1. Rule candidate generation and matching algorithm

```

create empty lists of rule candidates rList and rListS

// Confirmation
for each side Sd in {Condition, Conclusion, General}
    create set C[Sd] containing all item lists resulting from the Expressions in Sd
for each item list Cond ∈ C[Condition], Cons ∈ C[Conclusion], Gen ∈ C[General]
    for each subset GS of Gen
        add to rList a new rule candidate RC(Cond ∪ GS → Concl ∪ (Gen - GS))
for each transaction T in the database
    for each rule candidate RC ∈ rList
        check support s and confidence c for RC
remove from rList all rules with s < minsup

// Specialization
for each rule candidate RC(X → Y) ∈ rList
    for each item i ∈ I - X ∪ Y
        add to rListS the RC(X ∪ {i} → Y)
for each transaction T in the database
    for each RC(X ∪ {i} → Y) ∈ rListS
        check support s and confidence c1 for RC and check confidence c2 for {i} → B
remove from rListS all rules with s < minsup or c1 < minconf or c1 < c2
return rListS

```



$rs([Q7_4] \rightarrow [Q65_4] [])$
with no support and confidence constraints.

An operation of level 1 Specialization has the following output:

```

62 results.
[Q7_4, Q72_4] -> [Q65_4] [ 1.2% 81.8% ]
[Q7_4, Q93_4] -> [Q65_4] [ 1.2% 78.2% ]
[Q7_4, Q37_4] -> [Q65_4] [ 0.9% 76.4% ]
[Q7_4, Q55_4] -> [Q65_4] [ 0.8% 75.0% ]
[Q7_4, Q22_4] -> [Q65_4] [ 0.5% 70.0% ]
...

```

It is the rule with the highest confidence that shows the relationship with Question 72 – “In case of works performed in the common spaces, are you satisfied with the information received”. This relation with maintenance works leads the analyst to want to add another item to the search: the unsatisfied answer to Question 58 – “Delays of the interventions to repair degradation of common spaces”. However the analyst does not know how this answer relates to the other elements in the rule, so he just puts it in the *General* part of the Schema. An operation of Confirmation on the resulted Rule Schema $rs([Q7_4] \rightarrow [Q65_4] [Q58_4])$ shows that item Q58_4 is most likely a condition rather than a conclusion:

```

2 results.
[Q7_4, Q58_4] -> [Q65_4] [ 1.4% 75.0% ]
[Q7_4] -> [Q65_4, Q58_4] [ 1.4% 18.2% ]

```

Now the analyst runs a level 1 Specialization on the same Rule Schema. The results are interesting:

```

78 results.
[Q7_4, Q39_4, Q58_4] -> [Q65_4] [ 0.8% 100.0% ]
[Q7_4, Q49_4, Q58_4] -> [Q65_4] [ 0.9% 93.3% ]
[Q7_4, Q25_4, Q58_4] -> [Q65_4] [ 0.9% 92.9% ]
[Q7_4, Q93_4, Q58_4] -> [Q65_4] [ 0.7% 91.7% ]
[Q7_4, Q57_4, Q58_4] -> [Q65_4] [ 0.7% 90.9% ]
[Q7_4, Q50_4, Q58_4] -> [Q65_4] [ 0.5% 88.9% ]
[Q7_4, Q6_1, Q58_4] -> [Q65_4] [ 0.5% 87.5% ]
[Q7_4, Q63_4, Q58_4] -> [Q65_4] [ 0.9% 86.7% ]
[Q7_4, Q5_1, Q58_4] -> [Q65_4] [ 0.8% 85.7% ]
[Q7_4, Q2_1, Q58_4] -> [Q65_4] [ 1.1% 84.2% ]
...

```

A number of relevant conclusions can be drawn related to the dissatisfaction about common spaces. Although the rules extracted have fairly low support, it is the high confidence that shows that they are important and must be considered. The new items in the conditions of the extracted rules show that the problem relates to unsatisfied answers to other questions like Q39 – on the building’s ambiance, Q25 – state of building’s proximity or Q57 – important reparations on common parts. However the last rules in the first 10 results are unexpected: the problem relates with very satisfied answers (value “1”) to the three questions Q2, Q5 and Q6 – access to the city center, administrative services and parking spaces. This can mean only one thing: that the problems of common spaces and

repair works in the common spaces are most likely related to the accommodations closer to the center. That may be considered as valuable information.

Example 2. The analyst knows that there is a relation between questions Q13 and Q2 – “There are sufficient spaces for children” and “Access to the city center”, more precisely between the “very satisfied” answers to the two questions. A Confirmation operation on the Rule Schema $rs([\] \rightarrow [\] [Q13.1\ Q2.1])$ shows that the implication is from the first to the second:

```
2 results.
[Q13_1] -> [Q2_1] [ 15.30% 75.50% ]
[Q2_1] -> [Q13_1] [ 15.30% 23.46% ]
```

The first rule that resulted is quite strong, so the analyst wants to know if there are any exceptions to the rule. He/she performs a first level Exception operation and he/she gets the result below:

```
1 result.
[Q1_4, Q13_1] -> [Q2_2] [ 0.20% 100.00% ]
```

Although the resulting rule has a low support, it has a very strong confidence, which is much higher than the confidence of the initial rule, even if the support is smaller. Moreover, the confidence is higher than the confidence of the rule $Q1.4 \rightarrow Q2.2$ [0.87% 56.52%], so it is the *association* of Q1.4 and Q13.1 that leads to the result.

The result shows that, although generally the satisfaction about spaces for children leads to a satisfaction about the access to the center, in the special case where there is dissatisfaction related to transportation (Question 1), people tend to be less satisfied about access to the center. This quite interesting association might be considered as valuable information.

6. Conclusion

In this paper we have presented a new solution for local association rule mining that integrates user beliefs and expectations. The solution has two important components. The Rule Schema formalism, based on the concepts introduced in [14], helps the user focus the search for interesting rules, by means of a flexible and unitary manner of representation. The local mining algorithm that was developed does not extract all rules and then post-process them, but, instead, searches interesting rules in the vicinity of what the user believes or expects. This way, the user can explore the rule space in a local and incremental manner, global processing being avoided.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216, 1993.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. *Advances in knowledge discovery and data mining*, 1:307–328, 1996.

[3] S. S. Anand, D. A. Bell, and J. G. Hughes. The role of domain knowledge in data mining. *Proceedings of the fourth international conference on Information and knowledge management*, 1:37–43, 1995.

[4] A. Bellandi, B. Furlotti, V. Grossi, and A. Romei. Ontology-driven association rule extraction: A case study. *Proceedings of the Workshop "Context & Ontologies: Representation and Reasoning 2007*, pages 1–10, 2007.

[5] J. Blanchard, F. Guillet, and H. Briand. Interactive visual exploration of association rules with rule-focusing methodology. *Knowledge and Information Systems*, 13:43–75, 2007.

[6] A. Ceglar and J. F. Roddick. Association mining. *ACM Comput. Surv.*, 38(2):5, 2006.

[7] B. Duval, A. Salleb, and C. Vrain. On the discovery of exception rules: A survey. *Quality Measures in Data Mining*, pages 77–98, 2007.

[8] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. *Advances in Knowledge Discovery and Data Mining*, 1:1–34, 1996.

[9] J. Hipp, U. Gntzer, and G. Nakhaeizadeh. Algorithms for association rule mining - a general survey and comparison. *ACM SIGKDD Explorations Newsletter*, 2(1):58–64, 2000.

[10] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. *Proceedings of the third international conference on Information and knowledge management*, pages 401–407, 1994.

[11] B. Liu and W. Hsu. Post-analysis of learned rules. *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, 1:828–834, 1996.

[12] B. Liu, W. Hsu, and S. Chen. Using general impressions to analyze discovered classification rules. *Proc. 3rd Int. Conf. Knowledge Discovery & Data Mining*, 1:31–36, 1997.

[13] B. Liu, W. Hsu, L.-F. Mun, and H.-Y. Lee. Finding interesting patterns using user expectations. *Knowledge and Data Engineering, IEEE Transactions on*, 11(6):817–832, 1999.

[14] B. Liu, W. Hsu, K. Wang, and S. Chen. Visually aided exploration of interesting association rules. *PAKDD '99: Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, pages 380–389, 1999.

[15] H. O. Nigro, S. G. Cisaró, and D. Xodo. *Data Mining with Ontologies: Implementations, Findings, and Frameworks*, chapter Preface, pages xi–xv. IGI Global, 2007.

[16] B. Padmanabhan and A. Tuzhilin. A belief-driven method for discovering unexpected patterns. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, 1:94–100, 1998.

[17] J. Phillips and B. G. Buchanan. Ontology-guided knowledge discovery in databases. *Proceedings of the international conference on Knowledge capture*, pages 123–130, 2001.

[18] G. Piatetsky-Shapiro and C. J. Matheus. The interestingness of deviations. *Proceedings of the AAAI-94 Workshop on Knowledge Discovery in Databases*, 1:25–36, 1994.

[19] A. Silberschatz and A. Tuzhilin. On subjective measures of interestingness in knowledge discovery. *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 275–281, 1995.

[20] R. Srikant and R. Agrawal. Mining generalized association rules. *Future Generation Computer Systems*, 13:161–180, 1995.