# MINING RULES IN THE *ONTO4AR* FRAMEWORK FOR UPDATING DOMAIN ONTOLOGIES

Cláudia Antunes
*Department of Information Systems and Computer Engineering*
*Instituto Superior Técnico / Technical University of Lisbon*
*Av Rovisco Pais,1 – 1049-001 Lisboa, Portugal*
*claudia.antunes@dei.ist.utl.pt*

**ABSTRACT**

Despite the efforts on the area of knowledge management, the building and maintenance of ontologies continues to be almost a manual task. In the last years its automation has been pursuit, but the large number of discovered rules, in the context of pattern mining, has impaired its success, namely for representing the relationships among concepts.
In this paper, we propose the use of the *Onto4AR* framework for discovering unknown rules, and present a semi-automatic methodology for adding those discovered rules to a domain ontology. This enrichment is controlled by domain experts that incorporate their knowledge (represented by an ontology) into the framework, and control the mining process in accordance to their expectations, by making use of several categories of constraints. The reduced number of discovered rules can then be analyzed and added to the ontology.

**KEYWORDS**

Data mining framework, Association rules, Constrained Pattern mining, Ontologies, Ontology learning.

## 1 INTRODUCTION

In the information society, where changes occur all the time, the maintenance of updated knowledge repositories is a utopia. Even with the advances in the area of knowledge representation and management (with the use of domain ontologies), it is almost impossible to incorporate the new knowledge in those models, by manual means.

However, the tentative to automate the maintenance of those models have also encountered several difficulties, namely in the management of the information discovered through the use of data mining tools. Among these tools are association rules, used for discovering the relationships among concepts.

The problem of its discovery was first introduced in 1993 (Agrawal et al, 1993), for finding sets of items purchased in the same transaction a significant number of times. Since then, association rules have been applied to a variety of domains, from medicine to marketing, with a fair success. Indeed, and despite the improvements on the algorithms for this task, the massive number of discovered rules has impaired their manual analysis.

In this paper, we present the *Onto4AR* framework, and propose its use for suggesting new rules to update domain ontologies. The framework provides an environment for specifying and using constraints that enables the user to control the mining process. Moreover, it uses a domain ontology to express existing background knowledge, and allows the discovery of unknown rules that can then be incorporated in the ontology, updating its content.

The rest of the paper is organized as follows: next (in section 2), we overview the notion of ontology and the problem of association rules. In section 3, we describe the entire framework, presenting its main concepts and next, we propose a semi-automatic methodology to update domain ontologies using the framework. In section 5, we present some conclusions and guidelines for future work.

## 2 ONTOLOGIES UPDATING AND ASSOCIATION RULES

One of the great problems related to the maintenance of up-to-date knowledge bases, and of domain ontologies, in particular, is the difficulty on acquiring it complete and effectively, in a useful time range.

Fortunately, the recent developments in the area of knowledge discovery, with the advances on mining data streams, make possible the acquisition of that knowledge almost in real time. However, the large amounts of discovered information, makes impossible its manual incorporation in the knowledge bases. Before, proceeding with the succinct analysis of these difficulties, it is necessary to understand what is an ontology and what are its constituents.

### 2.1 Ontologies

An *ontology* is an explicit specification of a conceptualization (Gruber, 1998), which means that is a specification of an abstract, simplified view of the domain.

Formally, and in accordance with (Maedche, 2002) an ontology is a 5-tuple $\mathcal{O}:=\{C, \mathcal{R}, \mathcal{H}^C, rel, \mathcal{A}^O\}$, where $C$ is a set of concepts, who represent the entities in the ontology domain; $\mathcal{R}$ is a set of relations defined among concepts; $\mathcal{H}^C$ is a taxonomy or concept-hierarchy, which defines the is-a relations among concepts ($\mathcal{H}^C(C_1, C_2)$ means that $C_1$ is a sub-concept of $C_2$, or in other words $C_2$ is a parent of $C_1$); *rel* is a function, $rel: \mathcal{R} \rightarrow C \times C$, that specifies the relations on $\mathcal{R}$ (if $R \in \mathcal{R}$, $rel(R)=(C_1, C_2)$ is also written as $R(C_1, C_2)$, and means that $C_1$ is related to $C_2$, but the inverse is not necessarily true); and finally, $\mathcal{A}^O$ is a set of axioms, usually expressed in a logical language, that describe constraints on the ontology, expliciting implicit facts.

In the counterpart of ontologies are knowledge bases, which specify existing instantiations for a particular ontology. Formally, a *knowledge base* is a 4-tuple $\mathcal{KB}:=\{\mathcal{O}, I, inst, instr\}$, where $\mathcal{O}$ is a ontology as defined above; $I$ a set of instances; *inst* a function from $C$ to $2^I$ called *concept instantiation* (with $C$, the set of concepts in $\mathcal{O}$) and *instr* the relation instantiation function defined from $\mathcal{R}$ to $2^{I \times I}$. (It is usual to designate the concept instantiated by an instance, as its *class*).

In this manner, while ontologies try "to capture the conceptual structures of a domain of interest" (Maedche, 2002), defining its elements, and describing the relations among them; a knowledge base defines a set of elements, which can be interpreted and understand with respect to an ontology.

In a simplistic way, graphically, an ontology can be seen as a graph composed by a tree (for representing its taxonomy), and several other arcs, that relate the concepts defined in the taxonomy.

With the developments of the ontology research area and the semantic web, several ontology specification languages have been proposed. See the work by Corcho (Corcho & Pérez, 2000) for an overview and comparison of such languages.

### 2.2 Open Issues in Pattern Mining

The problem of mining association rules is defined as the discovery of "all association rules that have support and confidence greater than the user-specified minimum support and minimum confidence respectively". With an association rule corresponding to an implication of the form $A \Rightarrow B$, where A and B are propositions (sets of pairs attribute/value, most of the times named items), and expresses that when A occurs, B also occurs with a certain probability. This probability is known as the rule confidence and is given by the conditional probability $P(B|A)$. The support of the same rule is given by the number of transactions that include A and B, simultaneously.

Since its definition, several algorithms have been proposed that efficiently deal with this problem, but most of them did not give any contribution to reduce the number of discovered rules. In fact, the most effective technique to reduce the number of discovered patterns is the use of constraints. As pointed by Bayardo, constraints play a critical role in solving the trade-offs of the generality of data mining algorithms, by focusing "the algorithm on regions of the trade-off curves (or space) known (or believed) to be most promising" (Bayardo, 2002). By using constraints, the user assumes the responsibility of choosing which of those aspects are most important for the current task.

In the problem of mining association rules, we can distinguish between two main categories of constraints: interestingness measures and content constraints. Interestingness measures rank the discovered patterns or rules, by quantifying the usefulness and utility of them, discarding those with an evaluation less than a user-specified threshold. Examples of such measures are confidence, lift or the minimum improvement (Bayardo et al, 1999). With these constraints, it is possible to both improve the performance of the algorithms, by pruning uninteresting patterns, and reduce the number of discovered patterns.

The other category of constraints, content constraints, can be seen as filters over the content of the discovered patterns, instead of its relevance. While interestingness measures are quantitative metrics, content constraints are predicates on the powerset of the set of items (Pei et al, 2001). In some sense, they capture application semantics and introduce it into the mining process. An example of such constraints is the item constraint, as proposed by Srikant et al (Srikant et al, 1997). Despite the advantages coupled to the use of content constraints, by allowing the incorporation of domain knowledge, they have been applied seldom. The main reason to this scenario is the inexistence of a framework that passes the control of the mining process to the user, enabling the definition of constraints, either interestingness measures or content constraints.

# 3 THE *ONTO4AR* FRAMEWORK

Despite the developments on the creation of ontologies and knowledge representation, the large majority of mining projects does not make use of these well-formed conceptual models and despise most of the existing knowledge. On the other side, the improvements on the algorithms for mining association rules, least has contributed to incorporate background knowledge on the discovery process.

The balance between discovering unknown information and manageable quantities of patterns is far from being reached. But, as been shown, the incorporation of constraints (both in the form of interestingness measures and content constraints) is one of the few effective approaches to achieve that goal.

The *Onto4AR* framework is a significant step in that pathway, since it defines a formal environment for incorporating background knowledge into the pattern mining process, that is independent on the problem domain, or the nature of the data. The framework is centered in a precisely definition of constraint and the problem of constrained pattern mining, based on the notions of ontology and knowledge base, presented before. In this new context, the problem of mining association rules can be restated. In order to do this, consider a set of items $\mathcal{I}$, and an itemset $\mathcal{A}$ as a subset of $\mathcal{I}$ ($\mathcal{A} \subseteq \mathcal{I}$), and a transaction as an itemset that occurred in a particular instant of time. Additionally, consider that the I set corresponds to the set of instances of a knowledge base ($\mathcal{KB}$:={$\mathcal{O}$, $\mathcal{I}$, inst, instr}), that references an ontology $\mathcal{O}$. Suppose that inst is defined, which means that the concept implemented by each instance is known.

> *Given a set of instances I from a specific knowledge base* $\mathcal{KB}$={$\mathcal{O}$, $\mathcal{I}$, *inst*, *instr*}, *and a set of transactions* $\mathcal{D}$, *find all rules of the form* $\mathcal{A} \Rightarrow \mathcal{B}$, *where* $\mathcal{A}$ *and* $\mathcal{B}$ *are disjoint itemsets of* $\mathcal{I}$ *that may occur in the transactions of* $\mathcal{D}$, *and the itemsets* $\mathcal{A}$ *and* $\mathcal{B}$ *satisfy a set of constraints* $C_{\sigma}$, *defined over the ontology* $\mathcal{O}$.

Note that this definition only differs from the usual one, because both support and confidence are constraints, included in the set of constraints $C_{\sigma}$. Also note that with this statement, there is no imposition on the relation between $\mathcal{A}$ and $\mathcal{B}$ (beside that they do not contain repeated items and all of them belong to the set of instances), neither on the number of times that they occur together or the confidence of the rule. All depends on the constraints imposed by $C_{\sigma}$.

As in the rest of the literature (see (Pei et al, 2001)), in the context of the *Onto4AR* framework, a *constraint* is a predicate on the powerset of the set of items $\mathcal{I}$, which means, that it is a function $c: 2^{\mathcal{I}} \rightarrow$ {*true*, *false*}. An itemset $S$ is said to satisfy $c$, if and only if, $c(S)$ is *true*. In this manner, a constraint imposes some condition over the elements of the itemset, or the relations among them. These predicates can establish either qualitative or quantitative conditions. Indeed, at a first glance, two different categories of constraints can be distinguished: *interestingness constraints* (also known as *interestingness measures*) and *content constraints*.

## 3.1 Interestingness Measures

Interestingness measures are constraints that impose quantitative conditions over the set of items in the rule, like the number of times that the set of items are transactioned together, or the novelty introduced by the discovery of the rule. Formally, an interestingness measure is a composed function $f= f_\theta$ o g $[f(x)=g(f_\theta(x))]$ , with g: $2^I \rightarrow \mathbb{R}$ and $f_\theta$: $\mathbb{R} \rightarrow$ {true, false}, defined by the comparison of its argument with θ, a threshold value.

A special interestingness measure is the *existential constraint*, usually known as *support*. For the rule A⇨B, it is simply defined by substituting *g* with the function *frequency and* $f_\theta$ with the function $>=_\theta$. Where the function *frequency* counts the number of transactions in the dataset that contains both the items of A and B, and the function $>=_\theta$ that returns *true* if and only if its argument is greater than or equal to θ, and *false* otherwise.

Another usual interestingness measure is *confidence*, which in the *Onto4AR* framework can be defined by instantiating the *g* function by the conditional probability and the function $f_\theta$ again by $>=_\theta$.

With the definition of these two interestingness measures, the problem of mining association rules as initially proposed can be addressed in the *Onto4AR* framework, without any difficulty, since the set of items and the set of instances are the same things.

## 3.2 Content Constraints

Although interestingness constraints play a fundamental role on pattern mining, they only capture the knowledge about some quantity that is significant for the specific business. As such, they are not able to represent any other knowledge about the business domain.

In order to go beyond this difficulty, content constraints introduce the ability of imposing that items in the rule have some specific characteristics. These characteristics can be selected among the ones represented in the domain ontology for the mining problem.

Formally, a *content constraint* can be defined as a predicate $c_\mathcal{O}$: $2^I \rightarrow$ {*true*, *false*} that impose some qualitative condition on the items of its argument, expressed based in the domain knowledge represented in the ontology $\mathcal{O}$. For the rest of this section, consider the knowledge base $\mathcal{KB}$:={$\mathcal{O}$, $I$, *inst*, *instr*}, with $\mathcal{O}$ the ontology defined by the tuple {$C$, $\mathcal{R}$, $\mathcal{H}^c$, *rel*, $\mathcal{A}^\mathcal{O}$}.

### 3.2.1 Taxonomical Constraints

The first kind of content constraint that can be defined in this framework is the *item constraint*. As defined by Srikant (Srikant et al, 1997), an item constraint is a "Boolean expression over the presence or absence of items in the rule". And in order to be able to use abstract concepts to define the scope of item constraints in the new context, some additional notions are needed: *parent*, *child*, *descendant* and *ancestor*, like in that work.

In the *Onto4AR* context, *child* is a predicate defined over the concepts in the ontology $\mathcal{O}$, and *child* ($x$, $y$) is *true* if and only if $\mathcal{H}^c(x, y)$ is defined in the ontology. *Parent* is the inverse of *child*, and then *parent* ($x$, $y$) is *true* if and only if *child* ($y$, $x$) is *true*. The predicate *descendant* can be defined in a recursive way:

$$descendant(x,y) \Leftrightarrow child(x,y) \vee [\exists z \in C: child(x,z) \wedge descendant(z,y)]$$

which means that either $x$ is child of $y$ or exists a third concept $z$ that is parent of $x$, and is descendant of $y$, simultaneously.

With these definitions, it is now possible to define item constraints in a generic form.

In the new context, an *item constraint* is a content constraint $c_S$, implemented by the predicate $\in_S$, and then it only returns *true* if some items of its argument belong to the user-defined subset $S$ of $I$. Let an itemset $X \in 2^I$, it is said that

$$X \text{ satisfies an item constraint } c_S \Leftrightarrow \exists x \in X: x \in S$$

Note that it does not deal with the absence of some items, and the specification of $S$ has to be entered manually by the user, as it was with the specification of the Boolean expression. This specification can be done either by enumerating all accepted instances, or by describing its elements properties. Additionally,

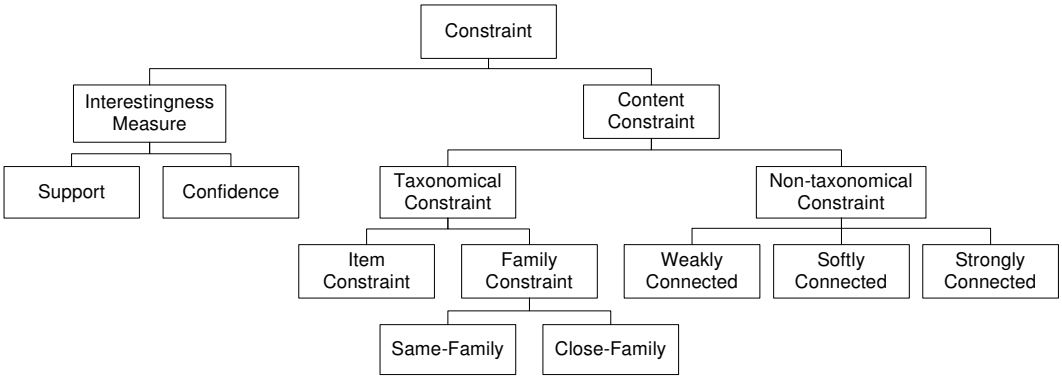


**Figure 1. Categorization of proposed constraints**

consider the notion of $n^{th}$ *cousin*. An item is a $n^{th}$ *cousin* of another item, if they are connected by a *n* degree collateral relation. More specifically,

$$\forall x, y \in I; n \in N : n^{th}\,cousin(x, y, n) = true$$
$$\Leftrightarrow \begin{cases} descendant\ (x,\ parent(y))\ ,if\ \ n = 0 \\ n^{th}\,cousin(x,\ parent(y),\ n-1),\ otherwise \end{cases}$$

For example, *x* is a $0^{th}$ cousin of *y* if it descends from *y*'s parent (it is either is "brother" or its "nephew"); *x* is a $1^{st}$ cousin of *y*, if it descends from *y*'s grandparent (one degree of collaterality); *x* is a $2^{nd}$ cousin of *y*, if it descends from *y*'s great-grandparent (two degrees of collaterality), and so on. With this definition, it is possible to define several other constraints – named *family constraints*. As expected, these constraints are implemented by the $n^{th}$*-cousin* predicate.

The *same-family*$_n$ constraint accepts an itemset if all their items share a common ancestor and all of them are $n^{th}$ cousins to each other. This means that

$$X\ satisfies\ same\text{-}family_n \Leftrightarrow \forall x,y \in X:\ x \neq y \wedge n^{th}cousin(x,y,n)$$

Similarly, the *close-family*$_n$ constraint accepts an itemset if every item in the set has an $n^{th}$-cousin in the set.

$$X\ satisfies\ close\text{-}family_n \Leftrightarrow \forall x \in X \exists y \in X:\ x \neq y \wedge n^{th}cousin(x,y,n)$$

Note that both item constraints and family constraints can be defined using the taxonomy on the ontology, so they are examples of *taxonomical constraints*.

### 3.2.2 Non-taxonomical Constraints

The incorporation of an ontology in the mining process, instead of a simple taxonomy, allows for the definition of other constraints that extend the notion of item constraint. Those new constraints are based on the relations among concepts, described in the ontology.

Consider some additional predicates over items and itemsets with two or more items (consider an itemset $X \in 2^I$), needed for defining those new constraints. First, two items are *related* if there is a known non-taxonomical relation *r* (with $r \in \mathcal{R}$) among their classes. Formally,

$\forall x,y \in \mathcal{I}$: $related(x,y)=true \Leftrightarrow \exists r \in \mathcal{R}$: $r(inst^{-1}(x), inst^{-1}(y))$.

Based on this notion, an itemset is *weakly connected* if all its items are related with at least another item in the set. This is,

$X$ is *weakly connected* $\Leftrightarrow \forall x \in X\ \exists y \in X$: $x \neq y \wedge related(x,y)$

An itemset can either be *softly connected*. This happens when there is a chain of relations among all its items. More specifically

$X$ is *softly connected* $\Leftrightarrow$

$\exists x,y \in X$: $x \neq y \wedge related(x,y) \wedge X \setminus \{x\}$ is *softly connected*

An itemset is *strongly connected* if all its items are related to all the other items in the set, which can be stated as follows:

$X$ is *strongly connected* $\Leftrightarrow \forall x,y \in X$: $x \neq y \wedge related(x,y)$

These three predicates can be used to define new content constraints: *weak*, *soft* and *strong* constraints, respectively. All of them can be categorized as non-taxonomical constraints, since they are based on non-taxonomical relations.

### 3.2.3 Composition of Content Constraints

Beside the existence of taxonomical and non-taxonomical constraints, the Onto4AR framework provides another important artifact – the composition of content constraints.

The *composition* of two content constraints, say $f$ and $g$, results in a new content constraint, $f$ o $g$, which only accepts itemsets accepted by at least one of the original constraints. Consider an itemset $X \in 2^I$,

$$f \text{ o } g : 2^I \rightarrow \{true, false\} \text{ with } f \text{ o } g(X) = true \Leftrightarrow f(X) = true \vee g(X) = true$$

It is important to note that the composition of any pair of the non-taxonomical constraints proposed above, always result in the most restrictive of the constraints. For example, composing weak and soft constraints will result on the last one, since all itemsets accepted by a soft constraint are also accepted by weak constraints. Indeed, the soft constraint is more restrictive than the weak constraint, and the strong constraint is more restrictive than the soft constraint.

Interesting compositions can result from combining non-taxonomical with taxonomical constraints. For example, the composition of weak and item constraints would only accept itemsets that have some specific items and, simultaneously, all their items are related with at least another item in the set.

In addition to the composition of content constraints, the *Onto4AR* framework allows the definition of other constraints, either as subclasses of proposed ones or by defining new kinds of constraints with different scopes. Examples of the last ones are temporal and structural constraints, used to identify relevant temporal patterns and to identify structured patterns, like frequent sequences or graphs. In order to extend the framework, it is only need to stat the new constraints in accordance to the definitions presented above.

In this manner, the categorization of the constraints presented in this paper, illustrated in Figure 1, is just a small picture of the potentialities introduced by the *Onto4AR* framework.

## 4 A METHODOLOGY FOR ONTOLOGIES UPDATING

The main challenges on updating a domain ontology is related with three main issues: first, the discovery of the new information, second, the analysis of such information, and third, its incorporation in the ontology.

In this manner, the methodology proposed in this paper comprises three steps: information acquisition, information into knowledge (IIK) and updating, as illustrated on Figure 2.

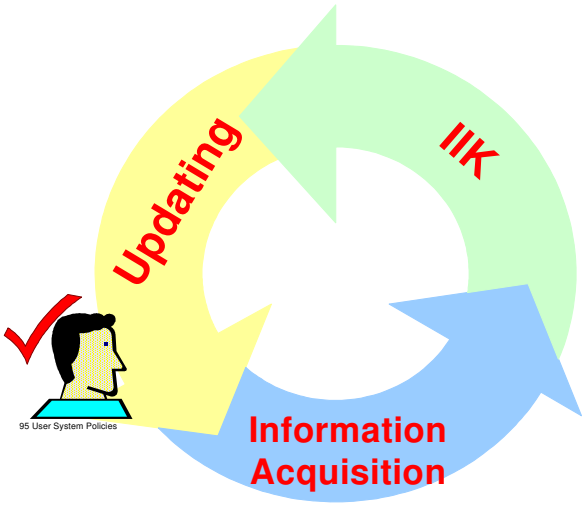


**Figure 2. Methodology for ontologies updating**

The first step, aims for identifying the new information through the use of efficient techniques for the discovery of frequent patterns, that are able to mine continuous, high-volume, open-ended data streams as they arrive. One of such techniques is the *estDec* method (Chang and Lee, 2004), which is able to find recent frequent itemsets over online data streams by decaying the weight of old transactions as time passes. In this manner, the changes in the data stream are adaptively reflected in already known patterns.

However, in the context of the *Onto4AR* framework, and due to human limitations on the analysis of large amounts of discovered information, this method has to be adapted to collect the rules discovered with any of possible constraints. In order to optimize it, no item constraints can be applied.

The next step is to identify removed and new rules, and suggest them to the domain expert. The third and last step is manual, and consists on incorporating valid rules in the ontology.

With this simple iteration, it is possible to suggest a little number of discovered relations to add, at each time, giving to domain experts the opportunity to contextualize the set of rules, understanding them and incorporating them into the ontology.

# 5 CONCLUSION

In this paper we describe the *Onto4AR* framework for mining association rules, and propose its use to update domain ontologies.

The framework is based on the use of a domain ontology, to represent existing domain knowledge, and makes use of it to guide the mining process. In order to accomplish this goal, it defines a set of constraints that conditioned the discovery process in accordance to user expectations, aiming for only identifying relations useful for the user. The term useful means that the user can make use of the information, since he is able to understand it and accommodating it in his mind frame.

The methodology proposed to update domain ontologies, takes advantage of the reduction on the number and scope of the discovered information, providing a simple tool to support the semi-automatic maintenance of domain ontologies.

Since the framework is extensible, in order to be able to address other problems, the next step is to define other relevant categories of constraints, like temporal and structured ones. Another important issue is the efficient adaptation of stream pattern mining algorithms to deal with constraints.

# REFERENCES


Agrawal, R.et al., 1993, A. Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the ACM SIGMOD conference on Management of Data*, Washington DC, USA. 207-216

Bayardo, R.J.et al, 1999. Constraint-Based Rule Mining in Large, Dense Databases. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (KDD'99), San Diego, California, USA. ACM Press. 145-154

Bayardo, R.J., 2002. The Many Roles of Constraints in Data Mining. In *SIGKDD Explorations*, vol. 4, nr. 1 pp. i-ii.

Corcho, O., and Gómez-Pérez, A., 2000. A Roadmap to Ontology Specification Languages. In the *Proceedings of 12th International Conference on Knowledge Engineering and Knowledge Management* (EKAW'00), Juan-les-Pins, French Riviera.Springer. 80-96

Chang, J.H. and Lee, W.S., 2004. A Sliding Window Method for Finding Recently Frequent Itemsets over Online Data Streams. In *Journal of Information Science and Engineering* 20-4. 753-762

Gruber, T.R., 1998. A Translation Approach to Portable Ontology Specifications. In *Knowledge Acquisitions*, 5, 2. Academic Press. 21-66

Maedche, A., 2002. *Ontology Learning for the Semantic Web*, Kluwer Academic Publisher.

Pei, J. et al, 2001. Mining Frequent Itemsets with Convertible Constraints. In *Proceedings of 2001 IEEE International Conference on Data Engineering* (ICDE'01), Heidelberg, Germany. 433–332.

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