Extracting Semantic Annotations from Moodle Data

Mihai Gabroveanu[1] and **Ion-Mircea Diaconescu**[2]

Abstract. The purpose of this paper is to provide a solution which allows automatic reasoning processes over Moodle activities logs, in order to obtain user-personalized recommendations. Activities logs are mined for association rules, which are the translated into Jena Rules. The information is then used by specific learning rules to create recommendations for specific users. Using this technique, additional information is obtained starting from activities database.
`Keywords:` e-Learning, association rules, Jena, RDF(S), ERDF.

1 Introduction

Nowadays, e-Learning systems are widely used, specially in schools, colleges and universities but not only. More and more corporations involve continuous learning in their management systems. A number of e-Learning systems such as Moodle, Sakai, ATutor, CLIX are available either open source or commercial either as a standalone applications or online learning platforms (such as Microsoft Learning Manager, BlackBoard).

All these systems accumulate a large amount of data suitable for analyzing the users behavior using data mining technics. The goal of extracted information is to improve the educational process.

This work describes an extension of Moodle e-Learning system, which extracts semantic metadata helpful in delivery of user personalized content. In a previous work ([6] and [13]) we improved Moodle by adding rules and semantics to enrich the reports generation. In this work we follow the idea that, additionally to the standard information that users can see, it is possible to obtain supplementary information indirectly available (i.e. obtained by processing data stored in Moodle activity logs). Using this Moodle module, users are informed about some specific changes or are advised to do some actions. For example a student can be advised to read some specific resources in order to obtain necessarily skills for a specific test. This module is user-based, meaning that all information and suggestions are made depending of which user authenticates to the system. For example may be unnecessarily to suggest for some users to follow a specific course, since they already followed that course, but for others users this can be a valid option. In order to create the module we use Weka to extract association rules from Moodle activities logs and Jena Rules to infer additional information.

The paper is organized as follows: (1) in the first part explain the steps followed to extract association rules from Moodle activities logs; (2) the second part explain the mapping from association rules to Jena Rules and discuss an improvement for Moodle activities logs; (3) finally, we describe the architecture of the module implementation.

[1] Dept. of Computer Science, University of Craiova, Romania, e-mail: mihaiug@central.ucv.ro
[2] Brandenburg University of Technology, Germany, e-mail: M.Diaconescu@tu-cottbus.de

2 Mining information from Moodle activities logs

E-Learning systems provides databases where information about students profile, courses, academic results and performed activities (reading, writing, taking tests) are stored. A huge quantity of data is collected and can be very difficult to perform a manually analyze over it. Data mining provides technics and algorithms useful to perform an automatically analyze over activities logs databases. Instructors use available data to improve the courses quality or to build recommendations for system users.

The process of discovering association rules is an important task in data mining. An association rule provides a relationship among different attributes. Our Moodle module use algorithms for mining association rules in order to identify possible relations between courses, resources, student activities. Particularly, a selection process regarding the information we are interested to mine is performed. This allows us to obtain only specific association rules which is relevant for our needs.

2.1 Basic Knowledge on Association rules

In this subsection we presents a basic introduction of concepts related to association rules and the mining process.

The initial problem of mining association rules was formulated by Agrawal in [1] and is called the *market-basket problem*.

Considering $\mathcal{T}$ to be a non-empty data table containing transactions, an *association rule* is an expression with the following form: $A \Rightarrow B$. Formally speaking, this means that transactions including A will include B as well, with a high probability. A and B are called *the antecedent*, respectively *the consequent* of the rule.

The quality of an association rule is expressed by several measures. Two of them, namely the *support* and the *confidence* are essential [1]:

- the *support* of $A \Rightarrow B$ is defined as *the percentage of transactions in $\mathcal{T}$ that contain both A and B.*
- the *confidence* of $A \Rightarrow B$ is defined as *the percentage of transactions in $\mathcal{T}$ containing A which also contain B.*

Example 1 *Table 1 contains courses followed by students. We see that student having ID* 1 *followed Web Technologies (WT), Web Applications (WA) and Web Documents (WD) courses, the student having ID* 2 *followed Web Applications (WA) and E-Business Technologies (EBT) courses, etc.*

An example of association rule is $WT \Rightarrow WD$. This express that some of the students who followed Web Technologies (WT) course, also followed Web Documents (WD) course. The support of this association rule is calculated as:

$$supp(WT \Rightarrow WD) = \frac{|\{1,3,5\}|}{|\mathcal{T}|} = \frac{3}{6} = 0.50$$

StudentID	List of courses
1	WT, WA, WD
2	WA, EBT
3	WT, WD, EBT
4	WA, WD, EBT
5	WT, WD
6	WT, EBT

Table 1. The list of courses

expressing that 50% of students followed both the Web Technologies (WT) course and the Web Documents (WD) course.

The confidence can be calculated as:

$$conf(WT \Rightarrow WD) = \frac{|\{1,3,5\}|}{|\{1,3,4,5\}|} = \frac{3}{4} = 0.75$$

and it express that: from all students who follow the Web Technologies (WT) course, 75% of them also followed the Web Documents (WD) course.

Rules having support and confidence greater than an user-specified minimum support ($minsup$) and respectively a minimum confidence ($minconf$) are named *strong association rules*.

In this work the goal is to obtain only strong association rules inferring new information relevant in our context.

To extract strong association rules many algorithms were proposed. The most popular are: Apriori [2], DHP [12], PARTITION [14], DIC [5].

2.2 Mining Logs to extract useful data

The Knowledge Discovery [10] consist in the following steps: collecting data, preprocessing data, applying the data mining algorithms and post-processing. The mining association rule process in e-Learning systems [9] follows some steps:

- *Collecting data*. The Moodle database store detailed logs with all activities that users performs.
- *Data pre-processing*. Typical tasks are performed in this phase: data selection, derivation of new attributes and selection of some attributes (new attributes are created starting from the existing ones and only a subset of relevant attributes are finally chosen), transforming the data format (to a format required by the used data mining algorithms or framework).
- *Applying the mining algorithms*. In this phase we need:
  - to choose specific association rule mining algorithm;
  - to configure the parameters of the algorithm (such as support and confidence threshold, $minsup$ and $minconf$);
  - to identify table(s) or data file are used in the mining process;
  - and to specify some other restrictions, such as the maximum number of items and what specific attributes can be present in the antecedent or consequent of the discovered rules.
- *Data post-processing*. Strong association rules which are obtained are represented in a comprehensible format.

Our interest is to extract association rules such as:

- *82% of the students who followed Web Technologies (WT) course, also followed Web Application (WA) course.*
- *70% of the students that solve home-works from Web Technologies (WT) course pass the WA exam.*
- *74% of the students that read resource A and B from course E-Business Technologies (EBT) read also resource C.*

In order to extract association rules from Moodle logs we use an existing data mining tool, namely Weka, which implements several algorithms for extracting association rules. For our purpose, we choose to we use Apriori [2], but also other algorithms can be taken into consideration. The mined models will be exported into PMML[3] (Predictive Model Markup Language). The Predictive Model Markup Language (PMML) is an XML-based language which provides a way for applications to define statistical and data mining models and to share models between PMML compliant applications.

Example 2 *Let consider the relational data table obtained from Moodle logs (Table 2) containing courses followed by students. This data table is obtained after pre-processing step and corresponding to transactional data table presented in Table 1.*

StudentID	CourseID
1	WT
1	WA
1	WD
2	WA
2	EBT
3	WT
3	WD
3	EBT

StudentID	CourseID
4	WA
4	WD
4	EBT
5	WT
5	WD
6	WT
6	EBT

Table 2. Excerpt from Moodle data

Executing the Apriori algorithm implemented in Weka over the data depicted below and providing a minimum support value (0.4) and a minimum confidence value (0.5) as parameters we obtain two association rules $WT \Rightarrow WD$ (supp=0.50, conf=0.75) and $WD \Rightarrow WT$ (supp=0.50, conf=0.75). Our module translate rules in the PMML form. By example, association rules obtained after post-processing step is depicted below:

```
<?xml version="1.0" encoding="UTF-8"?>
<PMML xmlns="http://www.dmg.org/PMML-3_1">
 <DataDictionary numberOfFields="2">
  <DataField dataType="integer" name="SudentID"
     optype="continuous">
   <Extension extender="weka" name="storageType"
     value="numeric"/>
  </DataField>
  <DataField dataType="string" name="CourseID"
    optype="categorical">
   <Extension extender="weka" name="storageType"
      value="string"/>
   <Value property="valid" value="EBT"/>
   <Value property="valid" value="WA"/>
   <Value property="valid" value="WD"/>
   <Value property="valid" value="WT"/>
  </DataField>
 </DataDictionary>
 <AssociationModel algorithmName="Apriori"
   functionName="associationRules"
   minimumConfidence="0.5"
   minimumSupport="0.4" modelName="Sudents_Courses"
   numberOfItems="2" numberOfItemsets="2"
   numberOfRules="2" numberOfTransactions="6">
  <MiningSchema>
   <MiningField name="SudentID" usageType="group"/>
   <MiningField name="CourseID" usageType="active"/>
  </MiningSchema>
  <Item id="1" value="WD"/>
```

[3] PMML - `http://www.dmg.org/pmml-v3-1.html`

```
  <Item id="2" value="WT"/>
  <Itemset id="1" numberOfItems="1" support="0.667">
   <ItemRef itemRef="1"/>
  </Itemset>
  <Itemset id="2" numberOfItems="1" support="0.667">
   <ItemRef itemRef="2"/>
  </Itemset>
  <AssociationRule id="1" antecedent="1"
    consequent="2" support="0.5" confidence="0.75"/>
  <AssociationRule id="2" antecedent="2"
    consequent="1" support="0.5" confidence="0.75"/>
 </AssociationModel>
</PMML>
```

An advantage os using this representation (PMML) is that it is XML based, it has a schema and it is easy to translate to another representation types, XML-based or not. Our solution use an XSLT transformation to map association rules from PMML representation to Jena rules syntax.

3 Generate Recommendations in Moodle

In this section, we describe a translation from association rules, extracted from Moodle activities logs based on *support* and *confidence* factors, to Jena rules. Using such rules, complex reports and recommendations on the page of each authenticated user are created.

3.1 Brief introduction to Jena Rules

Jena is a framework which allows reasoning over RDF(S) ([8], [4]). It use a triple based syntax for rules (e.g. `(?x rdf:type moodle:Student)`), and built-ins to represent user defined operations (actions). Atoms are represented by RDF nodes, and the used syntax for representing URI's, variables, blank nodes and literals (plain or typed) is based on SPARQL. In Jena rules, components of a triple are: (1) the *subject* - is the first node, and it can be variable, URI reference or blank node; (2) the *predicate* - the second node of the triple, is expressed by using a variable or an URI reference; (3) the *object* - the last node of the triple, can be a variable, an URI reference, a blank node or a literal.

Jena rules offers support for a form of `negation-as-failure`, expressed by using the `noValue` built-in, who's parameters are the nodes of the triple (e.g. `noValue(?x moodle:passedExam moodle:WT)`). Conjunction is used by default and disjunction is not supported. Three types of rules are supported by Jena Rules engines, namely *forward*, *backward* and *hybrid* (forward rules having backward rules in the head). This paper describe a Moodle improvement dealing with forward rules executed by a RETE [7] forward engine.

3.2 Mapping association rules to Jena Rules

In order to generate reports and recommendations, we use Moodle activities logs as knowledge base and a translation from the extracted association rules to Jena rules is performed. We consider the association rule, obtained above in the mining process:

82% of the students who followed Web Technologies (WT) course, also followed Web Application (WA) course.

Using such rules, we can recommend to some students, *who already followed* WT *course and do not followed yet the* WA *course*, to consider follow that course, (e.g. in the next semester). Particularly, for currently authenticated student *Tom Miller*, we can recommend him to consider the *WA* course for the next semester but this is not an available information for another authenticated student *John Smith* who already followed *WA* course. Newly obtained data is not stored into Moodle database. Instead, it is used by the Moodle view module when the page for this student is generated.

The above association rule translate into the following Jena rule:

```
[R:
(?x rdf:type moodle:Student)
(?x moodle:username moodle:Tomy)
(?x moodle:takenCourse moodle:WT)
noValue(?x moodle:takenCourse moodle:WA)
->
(?x moodle:followCourse moodle:WA)]
```

We have to note that the second triple of our rule is dynamically created when the user access a page, and isn't part of the association rule. The subject of the triple `(?x moodle:username moodle:Tomy)` is obtained by using the *usernames* of the current logged users. Using this technique, we can express user-based recommendations. We don't want to recommend a course for a student which already followed that course. For the rule expressed in Jena, we use the `noValue` builtin to check in the working memory the triple denoted by the built-in parameters and it fails if the triple is found. Assuming that our rule is expressed as $WT \Rightarrow WA$, and denoting with D_{WT} the set of all students who followed WT course and with D_{WA} the set of all students who followed the *WA* course, then have to analyze four possible situations:

- *x is a positive example* - $x \in D_{WT} \wedge x \in D_{WA}$ - for our case, this express that the student already followed the WT course and also the WA course. This situation is covered: if the student already followed both courses, then the rule do not fire.
- *x is a non-positive example* - $x \notin D_{WT} \vee x \notin D_{WA}$ - for our case, this express that the student hasn't followed the WT course or hasn't followed the WA course. In our rule, if the student hasn't followed the WT course, the rule will do not fire, and if the student hasn't followed the WA course then the rule fire only if he followed the WT course.
- *x is a negative example* - $x \in D_{WT} \wedge x \notin D_{WA}$ - in this case, the rule fire and we recommend for that student to consider the WA course.
- *x is a non-negative example* - $x \notin D_{WT} \vee x \in D_{WA}$ - in this case the rule don't fire, since the student either hasn't followed WT course or already followed WA course.

We can note that in the case of a *positive* and *non-negative* example, the rule do not fire and in the case of a *negative* example the rule always fire. For the case of *non-positive* example, the rule fire only if the student followed the WT course but not followed the WA course. We have this situation because we want to recommend some actions only to students which cover the conditions of the boolean association rule but do not cover all conclusions.

The rule from the above example has a simple structure: only one antecedent atom (translating into a condition) and one precedent atom (translating into a conclusion). Sometimes, rules are more complex:

74% of the students who get WT course and passed the test T2 have accessed resource Res1 and solved assignment A1.

A result of applying a reasoning using such a rule, can be to guide the student to read some resources and to do some specific actions in order to prepare himself for a specific test.

There is a need to to make suggestions only for students which accomplish all conditions and we need to recommend only those parts from conclusion which are not accomplished yet. For the student *Tom Miller* who follows the *WT* course, and already accessed resource *Res1*, we need only to suggest him to consider solving the assignment *A1*. For this case we have also a supplementary condition, expressing that he hasn't passed yet the test *T2*. For the rest of the students, this recommendation will not be useful. Such statements, from association rules, translate into negated conditions in Jena rules.

We translate this rule into two Jena rules:

```
[R1:
(?x rdf:type moodle:Student)
(?x moodle:username moodle:Tomy)
(?x moodle:takenCourse moodle:WT)
noValue(?x moodle:passTest moodle:T2)
noValue(?x moodle:accessedResource moodle:Res1)
->
(?x moodle:accessedResource moodle:Res1)]

[R2:
(?x rdf:type moodle:Student)
(?x moodle:username moodle:Tomy)
(?x moodle:takenCourse moodle:WT)
noValue(?x moodle:passTest moodle:T2)
noValue(?x moodle:solvedAssignment moodle:A1)
->
(?x moodle:solvedAssignment moodle:A1)]
```

Consider having the student *Tom Miller*, with the username *Tomy*. It has followed the *WT* course (but not the *WA* course), hasn't passed the *T2* test, hasn't accessed the *Res1* resource and also hasn't solved the *A1* assignment. According with those statements, Tom Miller accomplish conditions from rules **R**, **R1** and **R2**. Conform with the rule **R**, we recommend him to consider follow the *WA* course, and according with **R1** and **R2** we recommend him to solve the assignment *A1* and to read information refereed by *Res1*. All those information are inferred and cannot by obtained directly from the activities logs.

For the general case, a boolean association rule:

$$A_1 \wedge A_2 \wedge ... \wedge A_n \Rightarrow B_1 \wedge B_2 \wedge ... \wedge B_m$$

translate into many Jena rules:

$$R_1 : A_1 \wedge A_2 \wedge ... \wedge A_n \wedge \neg B_1 \Rightarrow B_1$$
$$.....$$
$$R_m : A_1 \wedge A_2 \wedge ... \wedge A_n \wedge \neg B_m \Rightarrow B_m.$$

The general case, for boolean association rules, already implies extraction of simple rules having the same conditions and each of them having the conclusion formed by one of the atoms from the association rule conclusion:

A simple association rule:

$$A_1 \wedge A_2 \wedge ... \wedge A_n \Rightarrow B$$

translate in the Jena rule:

$$R_1 : A_1 \wedge A_2 \wedge ... \wedge A_n \wedge \neg B \Rightarrow B$$

In order to obtain Jena rules from association rules, select only association rules having good a probability (*strong association rules*). For this reason, a minimum value is selected for both support and confidence measure factors. For different rule sets, different values for maximum and minimum factors are set.

3.3 Jena Rules inference submodule

After obtaining Jena rules starting from association rules, the next step is to use those rules, and Moodle activities logs to infer supplementary information. Jena API contains a module capable of extracting models used in the reasoning process, directly from a MySQL database. The inference submodule directly link to the Moodle database and extract all necessarily data from tables, creating RDF triples which are stored in the working memory. Those triples represents the initial facts base. At the next step, Jena rules obtained from association rules, are loaded by the engine into a *RuleStore* object. When the inference process in finished, the working memory contains new facts obtained by applying rules over initial facts base. New information (facts) are used to create additional information and recommendations in the user view page. The new information is temporarily stored, and is processed by the view submodule of the extension.

3.4 Adding strong negation for Moodle data

We saw that in Jena Rules we use `noValue` built-in for checking the existence of some specific facts in the working memory (it implements a form of *negation-as-failure*). Assuming that our goal is to find out for some accessed resources (from a specific course), which of them are considered useful by students, it is possible to obtain both useful not useful resources. Also it is possible to have an overlap. The meaningful recommendations address useful resources, therefore we want to suggest only those resources which are considered useful by the others. This can be naturally expressed, by using negative facts (e.g. we can have facts expressing that a resource is marked as not useful by some of the students). This can't be expressed by using *negation-as-failure*, since a student can mark the resource as useful, other student mark the same resource as not useful and other student don't mark at all. Using *negation-as-failure*, we may conclude that a resource is *not useful* just because it was not marked as *useful*. This is not always true: not marking as useful, sometimes means that the student has not marked the resource since it has no opinion about that resource at the moment of questioning. This is related to Open World Assumption (OWA) and Closed World Assumption (CWA). In the case of CWA, not marking the resource means that we have a not useful resource. In the case of OWA, not marking the resource means that it's status is *undetermined*.

Introduced in [3], and based on Partial Logic[11], ERDF comes with a solution to allows such facts. It use *strong negation* in order to represent negative information, e.g. *not-useful* resources. In this way, the property `moodle:usefulResource` is represented as a *partial property*, and it can represent positive information, negative information, ambiguous resource, or don't represent information at all (undetermined). Moreover, ERDF supports closed and open world assumption. Some predicates are closed (are totally represented in the knowledge base), and for those we can infer negative information if positive information can't be inferred. Other predicates are partial, and for those we can express multiple truth values (true, false, overdetermined and undetermined).

A prototype of an ERDF engine was developed and is available for online[4] testing. It is based on Jena and supports strong negation and a form of negation-as-failure.

4 System Architecture and Implementation

Figure 1 illustrates the overall architecture of the system. The system contains four specific parts (modules):

- *Mining Association Rules module* - extract association rules using Weka API;

[4] `http://oxygen.informatik.tu-cottbus.de/JenaRulesWeb`

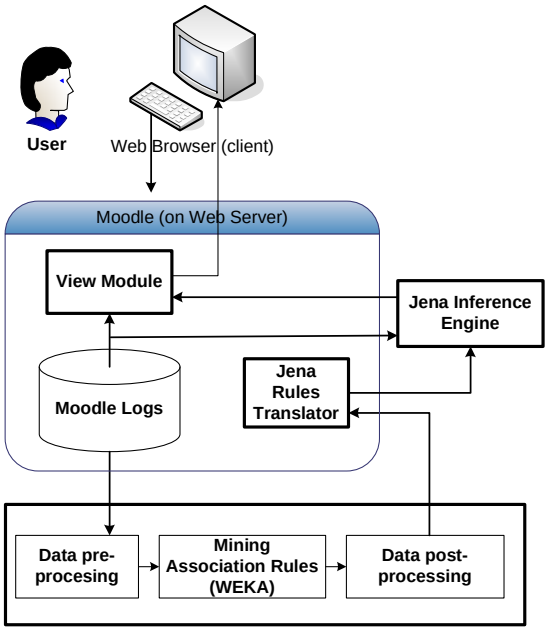

Figure 1. System Architecture

- *Jena Rules Translator module* - maps association rules to Jena rules.
- *Inference Engine module* - interact with the Jena inference engine. It uses the Jena rules obtained from the previous module, and Moodle activities logs as initial working memory.
- *View module* - improve user views, by adding new information obtained from the inference process and possible obtained recommendations.

The *Mining Association Rules module* connects to Moodle database, obtains activities logs, select and prepare data in order to extract association rules. For the mining process, WEKA is used.

Using the second module, association rules which are obtained from the mining process are then translated to Jena rules. The inference engine runs as a servlet and uses Jena API and rules obtained before in order to obtain new information. Finally, a PHP module improve the final view of the authenticated user with new information and possible recommendations obtained after the inference process.

Some operations are dynamical (e.g. the reasoning process, creating views), and others are created timely by a *cron* process (e.g. mining logs to obtain boolean association rules, translate association rules into Jena rules). Before Jena rules are passed to the inference engine, for each rule, a triple expressing the identity of the currently logged user is added (e.g. `(?x moodle:username moodle:Tomy)`). Also, new triples regarding authenticated users are added to memory when a user login to the Moodle system. Multiple users authentication is supported by adding a new triple for each new authenticated user. Those triples allow us to identify relevant information for specific users.

5 Conclusion and future work

The paper describes a Moodle extension used to create improved views for users by adding recommendations based on the existing data about user activities. The view is created by using the user-data as input for a rule-based learning recommendation processing.

Future work include representation of negative facts in Moodle activities and using fuzzy association rules instead of boolean association rules. In addition, we intend to develop a rule designer module which allows (for tutors) to create general/specific interest rule based on diverse criteria. General rules apply to all users of the system (e.g. create a message for every student which has not passed an exam).

REFERENCES

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami, 'Mining association rules between sets of items in large databases', in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, eds., Peter Buneman and Sushil Jajodia, pp. 207–216. ACM Press, (1993).

[2] Rakesh Agrawal and Ramakrishnan Srikant, 'Fast algorithms for mining association rules', in *Proc. 20th Int. Conf. Very Large Data Bases, (VLDB)*, eds., Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, pp. 487–499. Morgan Kaufmann, (12–15 1994).

[3] Anastasia Analyti, Grigoris Antoniou, Carlos Viegas Damasio, and Gerd Wagner, 'Negation and Negative Information in the W3C Resource Description Framework', *Annals of Mathematics, Computing and Teleinformatics*, **1**(2), 25–34, (2004).

[4] D. Brickley and R.V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation February 2004. http://www.w3.org/TR/rdf-schema/.

[5] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur, 'Dynamic itemset counting and implication rules for market basket data', in *Proceedings ACM SIGMOD International Conference on Management of Data*, pp. 255–264. ACM Press, (May 1997).

[6] Mircea Diaconescu, Sergey Lukichev, and Adrian Giurca, 'Semantic Web and Rule Reasoning inside of E-Learning Systems', in *Proceedings of 1st International Symposium on Intelligent and Distributed Computing*, eds., C. Badica and M. Paprzycki, Studies in Computational Intelligence, Craiova, Romania, (18-20 October 2007). Springer.

[7] C. Forgy, 'Rete – A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem', *Artificial Intelligence*, **19**, 17–37, (1982).

[8] Klyne G. and Caroll J.J. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-concepts/.

[9] Enrique Garcia, Cristobal Romero, Sebastian Ventura, and Toon Calders, 'Drawbacks and solutions of applying association rule mining in learning management systems', in *Proceedings of the International Workshop on Applying Data Mining in e-Learning (ADML'07)*, (September 2007).

[10] Jiawei Han, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[11] Heinrich Herre, Jan O. M. Jaspars, and Gerd Wagner, 'Partial Logics with Two Kinds of Negation as a Foundation for Knowledge-Based Reasoning', in *What is Negation?*, eds., D.M. Gabbay and H. Wansing, Kluwer Academic Publishers, (1999).

[12] Philip S. Yu Jong Soo Park, Ming-Syan Chen, 'An effective hash-based algorithm for mining association rules', in *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pp. 175–186, San Jose, Canada, (1995).

[13] Sergey Lukichev, Adrian Giurca, and Mircea Diaconescu, 'Empowering moodle with rules and semantics', in *Proceedings of 3rd Workshop on Scripting for the Semantic Web (SFSW2007)*, eds., T. Heath S. Auer, C. Bizer and G. A. Grimnes, Innsbruck, Austria, (6 June 2007).

[14] Ashok Savasere, Edward Omiecinski, and Shamkant B. Navathe, 'An efficient algorithm for mining association rules in large databases', in *Proceedings of 21th International Conference on Very Large Data Bases (VLDB'95)*, eds., Umeshwar Dayal, Peter M. D. Gray, and Shojiro Nishio, pp. 432–444. Morgan Kaufmann, (September 1995).