Rule mining with GBGP to improve web-based adaptive educational systems

C. Romero, S. Ventura, C. Hervás & P. González
Department of Computer Sciences and Numerical Analysis
University of Cordoba, Spain

Abstract

In this chapter we describe how to discover interesting relationships from student's usage information to improve adaptive web courses. We have used AHA! to make courses that adapt both the presentation and the navigation depending on the level of knowledge that each particular student has. We use data mining methods for providing feedback to courseware authors. The discovered information is presented in the form of prediction rules since these are highly comprehensible and they show important relationships among the presented data. The rules will be used to improve courseware, specially . We propose to use (GBGP) with multi-objective optimization techniques as rule discovery method. We have developed a specific tool named EPRules (Education Prediction Rules) to facilitate the knowledge discovery process to non-experts users in data mining.

1 Introduction

In the past years, we have seen an exponential growth in the use of web-based technology in distance learning systems. At the same time, different artificial intelligence techniques have been applied to these systems to improve students' learning, leading to what is know as Intelligent Tutoring Systems. The union of web-based learning with Intelligent Tutors has given rise to the current Web-based Educational Hypermedia Adaptive Systems [1] that allow adapting the teaching to each individual student through Internet. But the methodology used for its construction is static. Once the construction of a course is concluded and it is published in the Web for its use, the system logs information about the users' interaction with the

course. However, teachers only use this information for student evaluation. We propose a dynamic construction methodology that uses the system usage information to discover information that will allow the teacher to improve the course. The application of knowledge discovery techniques and in web-based education systems is a very novel and promising research area [2]. The same idea has been successfully used for a long time in e-commerce systems [3]. But whereas the e-commerce objective is to guide clients in making purchase decisions, the e-learning objective is to guide students in learning. Currently there are a lot of tools, both commercial and freeware, to carry out data mining tasks, and mainly rule discovery. Among all these, DBMiner [4] and Weka [5] stand out because they are very popular public domain systems and they have an integrated graphic environment that lets them carry out almost all data mining tasks. The main inconvenience is that these tools can be difficult to use for a non-expert in data mining (teachers are typically not experts in data mining). In addition they are of a general purpose nature, so they can't carry out a specific treatment of domain knowledge. In particular they do not contain features that are specific to the area of Adaptive Systems for Web-based Education (ASWEs) [1]. To resolve these problems we have developed a specific tool that has been denominated EPRules (Education Prediction Rules), to simplify the process of discovering prediction rules [6]. We have used AHA! [7] to make courses that adapt both the presentation and the navigation depending on the level of knowledge that each particular student has. We have performed several modifications in AHA! to specialize it and power it in the educational area. Our objective is to discover relations from the gathered usage data (reading times, difficulty levels and test results) from student executions and show the most interesting ones to the teacher so that he can carry out the suitable modifications in the course to improve it.

In the following, we start describing the use of rule mining techniques in e-learning systems. Next we show the process of discovering information, implemented in EPRules, and the proposed grammar-based genetic programming for rule discovery. Then, EPRules tool and the performed tests are described and some instances of the discovered rules are presented. Finally we present the main conclusions and future work.

2 Data mining in e-learning systems

is a knowledge discovery process to find previously unknown, potentially useful and non-trivial patterns from large repositories of data [4]. is the application of data mining techniques to extract knowledge from web data. There are three web mining categories: web content mining, web structure mining and web usage mining [3]. Web usage mining is the more relevant technique for e-learning systems. Web usage mining generally refers to the application of data mining techniques on web logs and meta-data. Frequently used methods in web usage mining are:

- **Association rules**. Associations between web pages visited.
- **Sequence analysis.** Analyzing sequences of page hits in a visit or between visits by the same user.
- **Clustering and classification.** Grouping users by navigation behaviour, grouping pages by content, type, access, and grouping similar navigation behaviours.

are interesting relationships discovered among data items [8]. The typical example is purchasing analysis, which can identify item pairs frequently purchased together. The use of rule mining in education is not new but was already successfully employed in several web-based educational systems. In their pioneering article, Zaïane [2] proposes the use of web mining techniques to build an agent that could recommend on-line learning activities or shortcuts in a course web site based on learner's access history to improve course material navigation as well as assist the online learning process. In other research, Wang [9] describes a set of tools for analyzing browsing log files based on data mining techniques such as association mining and collaborative filtering. Related research has been carried out by Yu et. al.[10], they use data mining technology to find incorrect student behavior. They modify traditional web logs, and apply fuzzy association rules to find out the relationships between each pattern of learner's behavior, including time spent on-line, numbers of articles read, number of articles published, number of questions asked, etc. In other research, Ha et. al. [11] proposes to use web page traversal path analysis for customized education and web page associations for virtual knowledge structures. Finally, Minaei-Bidgoli and Punch [12] introduce an approach for predicting student performance, they use clustering web-based assessment resources and discover interesting association rules within a web-based educational system. They use genetic algorithms to optimize a combination of multiple classifiers by weighing feature vectors.

All these current approaches use the visited pages as input to the search, and hence the discovered information describes relations between pages. In contrast, our proposed method also searches for relations between concepts and chapter units of web-based courses, and not only between pages. We are going to propose a methodology that uses evolutionary algorithms as association rule mining method for discovering interesting relationships in student's usage data to improve adaptive systems for web-based education [13]. Four main steps are distinguished in our methodology there are:

1. **Construction of the course**. The teacher builds the Hypermedia Adaptive Course providing information of the domain model, the pedagogic model and the interface module. An authoring tool is usually used to facilitate this task. Once the teacher finish the elaboration of the course, all the contents may be published on a web server.
2. **Execution of the course**. Students execute the course using a web navigator and in a transparent way the usage information is picked up and stored into the server in the log file of each student.

3. **Rule discovery**. After log files have been transferred to a database, the teacher can apply the evolutionary algorithms to discover important relationships among the gathered data. We want to discover relationships between knowledge levels, times and scores. The teacher can use our specific mining tool (EPRules) in order to facilitate this task.
4. **Improving the course**. The teacher can use the discovered relationships in order to perform the modifications that he believes more appropriate to improve the performance of the course. For example he can modify the course's original structure (joining concepts, changing concepts from level or chapter, etc.) and content (eliminating or improving bad questions, bad pages, etc.). To do it, he uses an authoring tool again.

3 Students' usage data

The usage information we have used to carry out the prediction rule discovery is the usage information captured from a Linux operating system course. We have developed the Linux course using AHA! [7] because apart from being a generic adaptive hypermedia system, it captures all the user's usage information, and its source code is available so we can (and are allowed to) modify it. We have modified AHA! in order to increase its adaptation power in education [14]. More precisely, we wanted to adapt or personalize the system to each particular student depending on his knowledge level. To do it, we have modified: the user model, the domain model and the adaptation engine of AHA!:

- **Domain model.** A course consists of several chapters with several concepts, but the concepts and the questions related with these concepts are divided in three levels of difficulty (high, medium or low).
- **User model.** The student's knowledge for each concept, initial test or final test can be only one of these values: 0 (not yet read), 10 (low), 50 (medium) and 100 (high).
- **Adaptation engine.** Before studying a new chapter the students have to do an initial adaptive test to discover their initial knowledge level. The system then presents them only the concepts with this level. Each concept has an activity to evaluate the student's knowledge about this specific concept. When the students have visited all the concepts they have to do a final (multiple-choice) test to evaluate their knowledge about the chapter at this level. If they obtain a medium or high level in the final test they can go to a higher level. If they are in the highest level already they can go to the next chapter. In each chapter everything starts again (see Figure 1): initial test, studying pages and doing activities and then the final test.

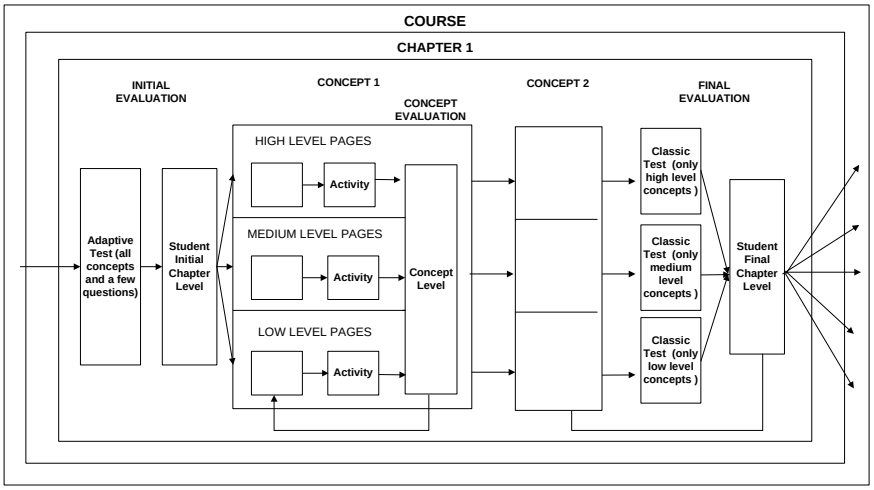

Figure 1: Modified AHA!.

In figure 2 the Introduction chapter of the Linux course is shown at two different difficulty levels (beginner and expert). Each version has a different concept explanation that is suited to the respective knowledge level that it is expressed by the background color of the page and the text label: Grado 0(beginner), Grado 1(normal) and Grado 2(expert).

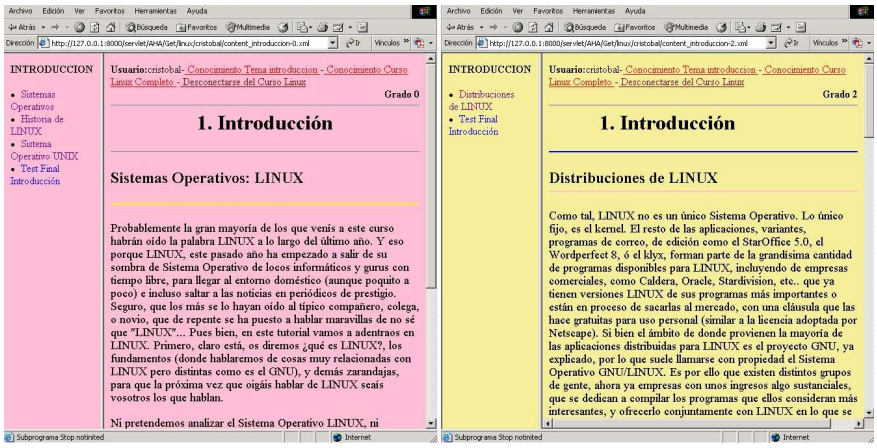

Figure 2: Two different levels of the Linux Introduction Chapter.

The AHA! system stores the usage information in two web log files (logs and model files) in which the information about user navigation and user knowledge is respectively stored. In addition, we have used another log file

(test file) to store the scores of the questions (activities and tests). The specific usage information used for data mining is the data logged for each student's interaction with the Linux course:

- **Times.** It's created from the log files and it contains information about the Web pages (contents, questions, etc.) and the time in which the student has accessed to them.
- **Levels.** It's created from the model file and it contains information about the knowledge level (high, medium, low) that the student has in each concept.
- **Success.** It's created from the test file and it contains information about the success or failure of the students in the questions (tests or activities).

Before applying rule mining algorithms we have transformed and moved all the log information to a relational database. This made (repeated) data extraction easier and increased the speed of the algorithms. During this process we have carried out the following pre-processing tasks [15]: attribute selection, data cleaning, discretization of continuous attributes and data integration.

4 Knowledge discovery process

The typical knowledge discovery process is shown in Figure 3.

Figure 3: Typical knowledge discovery process

As we can see data mining is only one step of the full process of knowledge discovery [16] that consists of:

Preprocessing. It consists of the data gathering, data cleaning, discretization of continuous data, attribute selection, data integration, etc.

Data mining. It consists of the application of a data mining task: classification, regression, clustering, rule discovery, etc.

Postprocessing. It consists of the interpretation, evaluation of the obtained results and the utilization of the discovered knowledge.

Our specific process of knowledge discovery with ASWEs is shown in Figure 4. It starts with the selection of course usage data, then it applies data mining algorithms to discover rules and finally the rules selected by the author can be used to make decisions about how to improve the course. All this process can be carried out by the teacher or author of the course, using the EPRules tool [13].

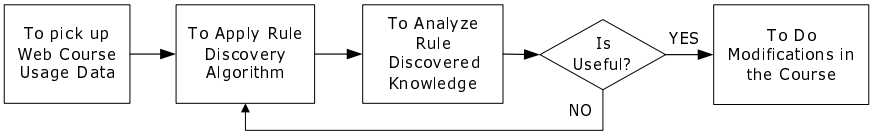

Figure 4: Specific knowledge discovery process.

The rule discovery process begins with the selection of the database where the pre-processed usage data of the course to be used are stored. Then the knowledge discovery algorithms to be applied must be selected as well as their specific parameters and both the objective and subjective restrictions that we want the discovered rules to fulfil. After finishing the algorithm execution, the group of discovered prediction rules is displayed: the elements of the rule antecedent and consequent as well as the evaluation measures of each rule are shown, and it is determined if the group of discovered rules are interesting or not. This depends on the number of rules, on their quality with respect to the different measures, and on their semantic meaning. Then it is decided which of the discovered rules are sufficiently interesting to use them to take decisions on possible modifications to the course. If the rules are not considered sufficiently interesting the algorithm is applied again, with different parameters and restrictions, in order to discover a more interesting group of rules.

4.1 Rule Discovery with GBGP

IF-THEN rules are one of the most popular forms of knowledge representation, due to its simplicity, comprehensibility and expressive power [4]. There are different types of rules depending on the knowledge they store. They are referred to as: decision rules, association rules, classification rules, prediction rules, causal rules, optimization rules, etc. The types of rules we will use are the prediction rules and Table 1 shows the precise format of the rules in EBNF (Extended Backus Naur Form).

The objective of prediction rules [17] is to predict an objective attribute depending on the values of another group of attributes. Although the syntax of prediction rules is similar to classification rules that have only one condition in the consequent, any of the attributes can appear in the consequent of the rule as it occurs in association rules. We are going to discover prediction rules, doing for that a dependence modeling task. This data mining task consists of the prediction of relationships among attributes specified (or not) by the user [15]. are very popular in data mining because they usually represent discovered knowledge at a high level of abstraction and it can be used directly in the decision making process. Dependence modeling can be considered like a generalization of discovering classification rules [18] or a specialization of association rules [8].

The rule discovery is carried out by (GBGP) [19] using techniques [20] which let us use several criteria to assess the quality of the rules. Grammar-based Genetic Programming is a genetic programming paradigm in which individuals are represented by trees derived from a grammar defined by the user to specify the solutions of the problem, in our case all the possible prediction rules. This paradigm has been chosen because it allows a high expressive power which is going to simplify the interaction with the user, restricting the grammar so the requested rules will be the only ones generated. There are plenty of metrics to evaluate the quality of the rules [21, 22], each one centered on some (different) aspects of quality. However, there isn't any metric which clearly surpasses the others in all application domains. For this reason this problem has been set out like a multi-objective optimization problem [20]. In this case there would not be a single attitude function associated to a metric, but several functions to perform the optimization at the same time. There are several ways to deal with the multi-objective optimization problem using evolutionary algorithms: the first one uses the "aggregation function", while the second one uses the Pareto Front concept. In the Pareto Front (which is the one we use) there is a vector of objectives to optimize within each individual, and the purpose of the algorithms is to make the solution for the individual converge towards the group of the best solutions (in terms of all objectives together and not in any specific objective) [23].

The evolutionary algorithm we have used consists of the following steps (Michalewicz, 96): The first step is Initialization, next Evaluation, Selection and Reproduction steps are repeated until the Finalization condition

Table 1: IF-THEN rule format in EBNF.

<rule>	::=	IF<antecedent>THEN<consequent>
<antecedent>	::=	<antecedent>AND<condition>|<condition>
<consequent>	::=	<condition>
<condition>	::=	<level-attribute> = <level-value> |
		<time-attribute> = <time-value> |
		<success-attribute> = <success-value>
<level-attribute>	::=	LEVEL.Name of a valid level attribute
<time-attribute>	::=	TIME.Name of a valid time attribute
<success-attribute>	::=	SUCCESS.Name of a valid success attribute
<level-value>	::=	BEGINNER | NORMAL | EXPERT
<time-value>	::=	HIGH | MEDIUM | LOW
<success-value>	::=	YES | NO

is fulfilled.

- Initialization consists of generating a group of initial rules specified by the user. They are generated from the most frequent values in the database. We use a Michigan approach in which each individual (chromosome) encodes a single rule. We use encoding scheme value in which a rule is a linear string of conditions, where each condition is a variable-value pair. The size of the rules depends on the number of elements in antecedent and the last element always represents the consequent. The generic format of the rules we are going to discover in Backus Naur Form (BNF) is shown in Table 1.
- Evaluation consists in calculating the fitness of the current rules. The fitness function used is made up of a 3-valued vector in which each value measures one of the three main aspects of the discovered knowledge using a data mining algorithm [15], that is, comprehensible, interestingness and accuracy. The metrics selected as partial objectives are the ones named certainty factor measure [24], interestingness measure [22] and simplicity measure [25].
- Selection chooses rules from the population to be parents to crossover or mutate. We use rank-based selection that first ranks the population and then every rule receives fitness from its ranking. The worst will have fitness 1, second worst 2, etc. and the best will have fitness N (number of rules in population). Parents are selected according to their fitness. With this method all the rules have a chance to be selected.
- Reproduction consists of creating new rules, mutating and crossing over current rules (rules obtained in the previous evolution step). Mutation consists of the creation of a new rule, starting from an older rule where we change a variable or value. We randomly mutate a variable or values in the consequent or antecedent. Crossover consists of making two new rules, starting from the recombination of two existent rules. In recombination the antecedent of a rule is joined to the consequent of another rule in order to form a new rule and vice versa (the consequent of the first rule is joined to an antecedent of the second).
- Finalization is the number of steps or generations that will be applied to the genetic process. We could also have chosen to stop when a certain number of rules are added to the final rule vector.

5 EPRules tool

The EPRules tool [6] is a visual tool to discover prediction rules and it is oriented to be used by the teacher or author of the course. It has been implemented in the Java programming language and its main characteristic is its specialization in education through attributes, filters and specific restrictions for the ASWE domain. Furthermore, it is a dynamic tool, because it lets the (advanced) user add new rule discovering algorithms and new rule evaluation measures, by modifying only some configuration files (Java prop-

erties files) and by selecting the new types in the algorithm directory or the new evaluation measures. The graphic interface of the EPRules application consists of four main windows:

- **Data input:** In this window (Figure 5) you can either open an existing database with the usage data of a course or create a new one and add new students to it. The course usage files (students' log files) must be selected in order to be pre-processed and integrated into a relational database. We have also transformed continuous attributes into discrete attributes using one of the following unsupervised global methods: equal-width method, equal-frequency method and manual method (in which you have to specify the cut-off points.). We have only done discretization of the time attribute, assigning three values: HIGH, MEDIUM and LOW.

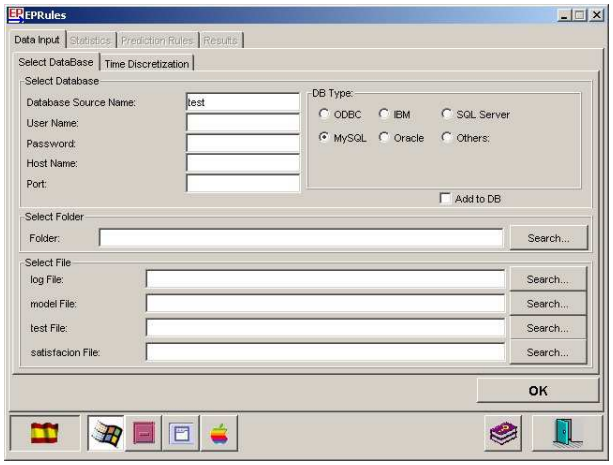

Figure 5: Data input window.

- **Data view:** If a course database has been opened from this window all students' pre-processed usage data can be visualized. These data are about the access times, correct answers and knowledge levels obtained by students for the different web pages (activities and contents) that make up the course. You can select either visualizing all students' data or a specific student's data, or just about a particular chapter of the course or about a specific concept of the chapter, or the visibility and difficulty level of a particular chapter (high, normal, low), or a type of particular information (time, level or correct answers).
- **Rule discovery:** This is the most important part of the tool because this is where the different algorithms for rule discovery are applied. The implemented algorithms are algorithms for decision tree building

like ID3 [18], an algorithm for association rule discovery like Apriori [8], an algorithm for induction rules like Prism [26] and different versions of evolutionary algorithms, specifically the grammar based genetic programming algorithm with or without multi-objective optimization (Pareto). You can select the algorithm to use and its particular parameters of execution (figure 6) and also the subjective restrictions that the rules have to fulfil (figure 7), so that the rules finally shown to the user are really interesting for him.

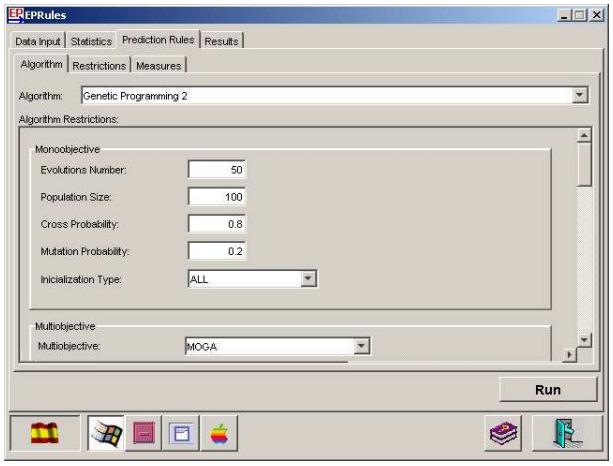

Figure 6: Algorithms window.

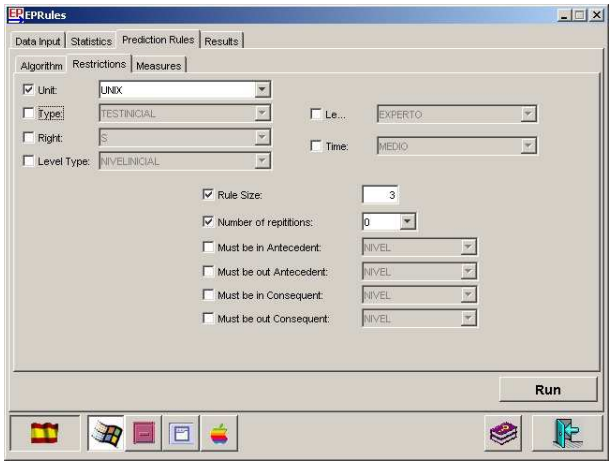

Figure 7: Restrictions window.

- **Results:** This window appears automatically after finishing the algorithm execution and lets us visualize all the discovered prediction rules (Figure 8). For each discovered prediction rule the conditions of the rule antecedent and consequent are shown and then all the values for each rules evaluation measure [22, 21] (certainty factor, interestingness, simplicity, confidence, support, interest, gini, laplace, etc., currently 40 different available measures). In a predetermined way, they appear ordered from the first discovered one to the last one, but they can be rearranged taking into account a condition or the value of any measure by simply clicking the desired column.

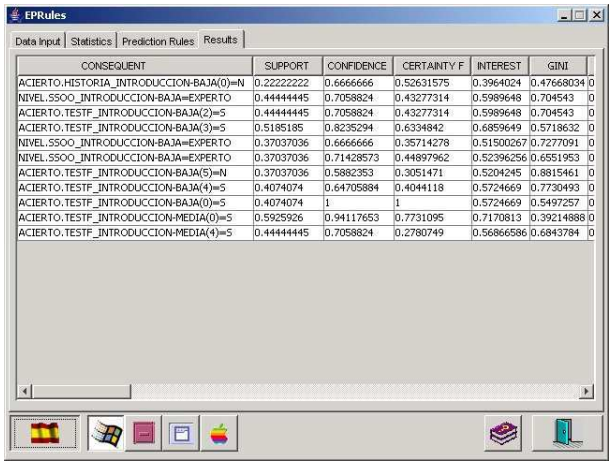

Figure 8: Results window.

6 Experimental results

To carry out the tests we have used the log information of 50 students of Computer Science Engineering at the Cordoba university taking a course about the LINUX operating system. The course was developed with (and served through) the AHA! system [7]. Different tests have been carried out in order to compare the results that each implemented algorithm produces in the task of knowledge discovery. The objective is to compare the number of discovered rules in each case and the quality of them based on the previously set out measures about accuracy, interestingness and comprehensibility. Because evolutionary algorithms are not deterministic, the evolutionary algorithms have been executed 10 times, and we have used the average values of all the executions. Furthermore, three different tests have been carried out: first using all data, then only the frequent data (those

with a support higher than 0.5) and finally, the range data (those with a support higher than 0.2 and lower than 0.9).

The obtained results show (Table 2, 3, 4) that, in general, evolutionary algorithms generate a lower number of rules but with higher interest than classic algorithms, making them more suitable to be used on-line, like for example in the extraction of knowledge in an adaptive system of education. Classic algorithms, and specially Apriori, produce very exact rules, but fail when generating rules with a higher interest and, furthermore, the length of the produced rules makes these rules difficult to understand. In addition, when we use all data (this could happen when the user wants to extract global information about the system without applying any type of restriction over the group) it generates a group of rules so large that it becomes impossible to exploit the rules later. The obtained results using the pro-

Table 2: Number of discovered rules.

Algorithm	**All**	**Range**	**Frequent**
ID3	474	131	89
Prism	657	172	62
Apriori	5960	491	70
AE-GBGP	198	162	51

Table 3: Percentage of exact rules.

Algorithm	**All**	**Range**	**Frequent**
ID3	46,0	51,9	60,3
Prism	71,9	53,7	91,9
Apriori	84,3	90,0	93,0
AE-GBGP	76,5	86,1	96,3

Table 4: Percentage of interesting rules.

Algorithm	**All**	**Range**	**Frequent**
ID3	1,5	7,6	15,6
Prism	2,5	11,6	49,3
Apriori	3,6	7,9	53,1
AE-GBGP	21,9	60,4	76,6

posed evolutionary algorithms show that, in general, these algorithms produce a lower number of rules than classic algorithms. Moreover, the use of algorithms based on Pareto Front (MOGA and NSGA) let us optimize the three objectives at the same time, producing in all the executions the higher proportion of accurate, comprehensible and interesting rules.

6.1 Description of the discovered information

The main objective of our work is to discover a group of useful and interesting rules and to present them to the teacher so that he can easily take decisions about how to improve the course. Semantically, the discovered rules express the following relationships:

```
IF Level|Time|Success  AND ...
THEN Level|Time|Success
```

Where Level, Time and Success are expressions referring to users' attained knowledge state (BEGINNER, NORMAL, EXPERT), the reading time for pages (HIGH, MEDIUM, LOW), and to information on students' successes and failures in the test and activities questions (YES, NO). Taking the discovered rules as a basis the teacher can decide which of the expressed relationships are desirable or undesirable, and what can be done to strengthen or weaken them (namely changing or modifying the contents, structure and adaptation of the course). The discovered rules show different types of relationships depending on which attributes are in the rule consequent:

- **Time.** It shows which attributes (in the rule antecedent) have an influence on the time (attribute of the rule consequent).
- **Level.** It shows which attributes (in the rule antecedent) have an influence on the level (attribute of the rule consequent).
- **Success.** It shows which attributes (in the rule antecedent) have an influence on the success (attribute of the rule consequent).

These relationships can make reference to chapters (levels obtained in initial and final tests) or to concepts (times, successes and levels obtained in exposition content pages and evaluation activities) of web-based adaptive courses. Using these discovered relations a teacher can make decisions about which modifications in the course are the most appropriate in order to increase the relationship (if he considers it to be desirable) or on the contrary to eliminate the relationship (if he considers it not to be desirable), changing or modifying the contents, the structure or the adaptation of the course.

Next we are going to describe the meaning and the possible use of several discovered rules.

```
IF LEVEL.interface-network-high = EXPERT THEN
LEVEL.tcpip-telnet-medium = EXPERT
(Interest=0.57, Factor Certainty=0.75, Simplicity=1)
```

This rule shows that the knowledge level obtained in the evaluation activities of the two mentioned concepts have been simultaneously very high (EXPERT). This indicates that the concepts (NETWORK, with HIGH difficulty level in the INTERFACE chapter, and TELNET, with MEDIUM difficulty level in the TCPIP chapter) may be related to each other. In this case, the teacher should check the presentation content of both concepts and try to find what the reason of the relationship is. And he should then decide if joining both concepts into a single concept, putting both concepts in the same chapter, setting them to the same level of difficulty, correcting the rules that assign levels, or any other modification is most appropriate. In this particular rule example we have considered that both concepts should have the same level of difficulty. But if the levels refer to initial or final test instead of activities, it can be concluded that the chapters are related. Then the teacher can join the chapters, or put them one after the other, or on the contrary, create more distance between them.

```
IF SUCCESS.characteristic-introduction-high(2) = NO AND
TIME.characteristic-introduction-high(2) = HIGH THEN}
LEVEL.characteristic-introduction-high = EXPERT
(Interest=0.65, Factor Certainty=0.87, Simplicity=0.5)
```

This second rule shows that students, evaluated as EXPERT in the concept CHARACTERISTIC in the INTRODUCTION chapter at the HIGH difficulty level, fail question number two of the activity of this concept, and they also need a HIGH time to answer that question. So, this rule shows that something is wrong with this question (bad or not clear enunciate, several or none correct answers, etc.) and the teacher should review that question to see what happen.

7 Conclusions and future work

In this paper we have introduced a visual tool to discover knowledge in the form of prediction rules in order to help teachers to improve adaptive Web-based courses. Particularly, we have proposed the use of Grammar-based genetic programming with multi-objective techniques. The quality of the results, depending on the number of rules obtained and their interestingness, accuracy and comprehensible factor of the rules, is higher than for a number of classic algorithms that use only one measure or a composition of some of them to evaluate the rules. Regarding the usability of the discovered rules for taking decisions about the modifications to be carried out in ASWEs, the different types of rules, and the usefulness that they can have to improve the course have been described and illustrated using specific instances of the discovered rules in a Linux course. A specific tool, named EPRules, has been developed in order to simplify the whole process of knowledge discovery. This tool lets us carry out the pre-processing of usage data in web courses, the selection of restrictions on the type of information to be discovered as

well as the application of data mining algorithms to extract the rules and to show them. Currently we are working on the following issues:

- Complete automation of the knowledge discovery process in ASWEs, so that the discovered rules can be applied directly on the course, without manual intervention by the teacher or author of the course, except for accepting or rejecting the changes proposed by the rules.
- Use other metrics related to the subjective interest of professionals in the discovered rules. Some works has been done using evolutionary algorithms [27] in which there isn't an aptitude function but individuals are assessed by an expert in each cycle of the algorithm.
- Use parameter-free rule mining algorithms [28]. In this way, we don't need to ask to the courseware authors for the specific values of the algorithm's parameters, and they don't need to understand what is the role of these parameters in the data mining process.

Acknowledgements

The authors gratefully acknowledge the financial support provided by the Spanish Department of Research of the Ministry of Science and Technology under TIC2002-04036-C05-02 Projects.

References

[1] Brusilovsky, P., Adaptative educational systems on the world-wide-web: A review of available technologies. *Int. Conf. on Intelligent Tutoring System*, 1998.

[2] Zaiane, O.R., Web usage mining for a better web-based learning environment. Technical report, 2001.

[3] Spiliopoulou, M., Web usage mining for web site evaluation. *Communicacions of the ACM*, 2000.

[4] Klosgen, W. & Zytkow, J., (eds.) *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, 2001.

[5] Witten, I.H. & Frank, E., *Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.

[6] Romero, C., Ventura, S. & de Bra, P., Discovering prediction rules in aha! courses. *9th International Conference on User Modeling*, Johnstown, PA, USA, pp. 25–34, 2003.

[7] de Bra, P. & Ruiter, J., Aha! adaptive hipermedia for all. *Proc. of the WebNet Conference*, pp. 262–268, 2001.

[8] Agrawal, R., Imielinski, T. & Swami, A., Mining association rules between sets of items in large databases. *ACM SIGMOD International Conference on Management of Data*, 1993.

[9] Wang, F., On analysis and modeling of student browsing behavior in

web-based asynchronous learning environments. *International Conference on Web-based Learning*, Hong Kong, pp. 69–80, 2002.

[10] Yu, P., Own, C. & Lin, L., On learning behavior analysis of web based interactive environment. *International Conference ICCEE*, Oslo, 2001.

[11] Ha, S., Bae, S. & Park, S., Web mining for distance education. *APAN Conference*, Beijing, 2000.

[12] Minaei-Bidgoli, B. & Punch, W., Predicting student performance: an application of data mining methods with the educational web-based system lon-capa. *IEEE Frontiers in Education*, pp. 1–6, 2003.

[13] Romero, C., Ventura, S. & de Bra, P., Knowledge discovery with genetic programming for providing feedback to courseware author. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, **14(5)**, pp. 425–465, 2005.

[14] Romero, C., de Bra, P., Ventura, S. & de Castro, C., Using knowledge levels with aha! for discovering interesting relationship. *World Congress ELEARN*, Montreal, 2002.

[15] Freitas, A.A., *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer Verlag, 2002.

[16] Mitraa, S., S.K., S.P. & Mitra, P., Data mining in soft computing framework: A survey. *IEEE Transaction on Neural Networks*, **13(1)**, pp. 3–14, 2001.

[17] Noda, E., Freitas, A. & Lopes, H.S., Discovering interesting prediction rules with a genetic algorithm. *Conf. on Evolutionary Computation*, 1999.

[18] Quilan, J.R., Generating production rules from decision trees. *Proceeding of IJCAI-87*, 1987.

[19] Whigham, P.A., Gramatically-based genetic programing. *Proceedings of the Workshop on Genetic Programming*, pp. 33–41, 1995.

[20] Coello, C., Veldhuizen, D. & Lamount, G., *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer, 2002.

[21] Lavrac, N., Flach, P. & Zupan, B., Rule evaluation measures: A unifying view. *ILP-99*, Berlin Heidelberg, 1999.

[22] Tan, P. & Kumar, V., Interesting measures for association patterns. Technical Report TR00-036, Department of Computer Science, University of Minnnesota, 2000.

[23] Fonseca, C.M. & Fleming, P.J., Genetic algorithms for multiobjective optimization: Formulation, discusin and generalization. *Conf. on Genetic Algorithms*, San Mateo, CA, 1993.

[24] Shortliffe, E. & Buchanan, B., A model of inexact reasoning in medicine. *Mathematical Biosciences*, **23**, pp. 351–379, 1975.

[25] Liu, J.L. & Kwok, J.T., An extended genetic rule induction. *Conf. On Evolutionary Computation*, 2000.

[26] Cendrowska, J., Prism: an algorithm for inducing modular rules. *Journal of Man-Machine Studies*, **27**, pp. 349–370, 1987.

[27] Williams, G.J., Evolutionary hot spots data mining. an architecture

for exploring for interesting discoveries. *Conf. on Knowledfe Discovery and Data Mining*, 1999.

[28] Keogh, E., Lonardi, S. & Ratanamahatana, C., Towards parameter-free data mining. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, pp. 22–25, 2004.