A Diversity-Enhanced Genetic Algorithm to Characterize the Questions of a Competitive e-Learning System

Elena Verdú, María Jesús Verdú, Luisa M. Regueras, Juan Pablo de Castro
Higher Technical School of Telecommunications Engineering
University of Valladolid
Valladolid, Spain
e-mail: {elever,marver,luireg,jpdecastro}@tel.uva.es

Abstract— Nowadays, the practice of different teaching methodologies is easier thanks to the technology-enhanced learning systems. However, in order to effectively center the learning process in the student it should be adapted to the student’s progress. Adaptive e-learning systems have been proved to be valuable tools, which facilitate this adaptation. QUESTOURnament, an active and competitive Moodle tool, is being re-designed in order to become an adaptive system. One of the first steps in this adaptation is the estimation of the difficulty level of the questions proposed in this environment. This paper describes a solution based on a genetic algorithm with enhanced diversity methods that automatically characterizes the answers to the challenges. The algorithm has been tested with data registered from a contest made in a Telecommunications Engineering course. It finds diverse good solutions, from which several rules can be defined to classify the questions according to their difficulty level.

Keywords- adaptive e-learning; competitive e-learning; fitness sharing; genetic algorithms; intelligent tutoring systems

I. INTRODUCTION

Nowadays, teachers are doing an important effort to adopt new teaching methodologies in which the student is the central and active part of the learning process. There are diverse technology-based tools that facilitate the development of these methodologies such as the computer supported collaborative learning systems or those systems that support on-line competitions. QUESTOURnament is an innovative e-learning system for active and competitive learning, which allows to increase students’ motivation and academic outcomes [1]. This system, integrated into Moodle, enables the development of contests in which students compete to solve several challenges correctly and faster than their classmates.

However, being the student the main actor in the learning process and due to the fact that each student has an own learning pace and style, it is necessary to target teaching to each individual’s needs. Adaptive e-learning systems are valuable tools for teachers to manage and provide personalized teaching. These systems build a model of goals, preferences and knowledge of each student and use it right through the interaction with the student in order to adapt to the needs of that student [2].

Therefore, the QUESTOURnament system should be redesigned in order to adapt to the students’ progress but preserving the competitive nature of the system. Students’ progress can be estimated from their answers to the different challenges proposed in the contests. Then, one important task in this adaptation process is the classification of the challenges according to their difficulty. This paper describes a solution based on genetic algorithms that characterizes the answers given to the different challenges in order to automatically classify them according to the real difficulty level found by the students.

II. THE ADAPTIVE QUESTOURNAMENT SYSTEM

The QUESTOURnament system is a competitive e-learning tool, which allows teachers to organize contests, where students must solve several questions (called challenges within the system) in a time-constrained way. Turning QUESTOURnament into an adaptive system involves several strategies:

• Define how to estimate the objective difficulty level of the questions.
• Design a student modeling module that is able to infer the student’s current knowledge as well as to model the different behaviors.
• Establish different contests (that is, learning paths) adapted to the students’ current knowledge and profile. The aim is that students with similar capabilities and behaviors compete among themselves. In this way, the options of competitiveness within a contest are maintained.

The estimation of the difficulty level of questions is the base of the whole adaptive system and the first step to be accomplished. The initial data set is obtained from Moodle and QUESTOURnament logs. Interaction data, such as time and type of action (reading, submission…) will be combined with grade values to obtain the set of response patterns that will be used as input for the genetic algorithm.

III. DIVERSITY-ENHANCED GENETIC ALGORITHM

A. Genetic Algorithms and Niching Methods

Genetic algorithms are search techniques that emulate biological evolution. First, they randomly generate an initial population of chromosomes, also called individuals, which represent different possible solutions to the problem. According to their fitness, chromosomes are selected and combined with other chromosomes through a crossover operator, producing children that can even mutate. These
children make up the next generation and again, they are selected for crossover according to their fitness. The fitness of each chromosome is determined by using a function, called fitness function, which evaluates the quality of the solution represented by that chromosome. Its definition depends on the specific problem. The higher the fitness is, the higher the chances are to be selected for crossover. Therefore fitness of individuals improves generation after generation and then, at the end of the process, an optimized solution is delivered by the algorithm.

Genetic algorithms have been successfully used for search and optimization in the field of e-learning. There are examples of their application for giving automatic answers to the students’ questions [3], retrieving valuable student information [4], predicting student performance [5] and estimating the difficulty level of the exercises proposed in a web-based system [6].

However, one problem in traditional genetic algorithms is the possible convergence to a local solution, which is sub-optimal. This problem, called premature convergence, can occur when diversity is lost and chromosomes in the population become almost identical; then population stops evolving in early stages, causing a premature stagnation of the search [7, 8].

On the other hand, traditional genetic algorithms converge to a single optimal solution while real optimization problems often require the localization of multiple solutions, either global or local [9]. Niching methods allow genetic algorithms to converge to multiple, highly fit, and significantly different solutions, as well as to avoid premature convergence in cases in which only one solution is required [10]. These methods aim at the formation and maintenance of stable subpopulations (niches) within a single population [11].

Two frequently used niching methods are fitness sharing and crowding. When implementing fitness sharing, as its name indicates, similar individuals share fitness. Specifically, the shared fitness of an individual is equal to its similarity of the individual to the rest of the population. As its name indicates, similar individuals share fitness.

Therefore, each possible solution to the problem is represented by the following function [16]:

$$f(C_i) = \frac{n_j}{n_j + w_{t1} \cdot w_{g1} \cdot w_{a1}}$$  

(1)

where $n_j$ is the number of patterns of difficulty $j$, which belong to the ranges of time, grade and accesses represented by the chromosome $C_i$, and $n_j$ is the total number of patterns of difficulty $j$. Then, $n_j$ divided by $n_j$ corresponds to the support measure. Additionally, three weights $w_{t1}$, $w_{g1}$ and $w_{a1}$ are used in order to award narrowest ranges and penalize widest ranges of time, grade and accesses, respectively. Logically widest ranges will cover almost all the domain and will have high support. The interest is to find medium and narrow ranges that cover most patterns.

The algorithm ends after several generations and generates optimized solutions to the problem: a rule for each difficulty level. However, for the present problem, it is necessary to consider that the response pattern for a student depends not only on the difficulty level of the question but also on the knowledge degree and behavior of the individual student. Thus, it is interesting to recover not only the best solution but also some other solutions, different from the best one, which are also good remarkable sets of ranges. Therefore, some additional mechanisms have been incorporated into this genetic algorithm in order to obtain diverse solutions.

C. Diversity-Enhanced Genetic Algorithm

The enhanced version of the genetic algorithm incorporates niching methods in order to promote diversity and propose several optimized solutions for each rule. A solution based on sharing has been implemented to solve the present problem, including some additional mechanism that also increases diversity. On the one hand, as in [12-14], the
The proposed algorithm inserts new individuals into the population, who replace similar individuals. A genotypic distance measure determines the similarity of the chromosomes based on their structure. On the other hand, as in the sharing approach, similar individuals share fitness and then fitness is reduced in densely populated regions.

The different steps of the enhanced algorithm are shown in Fig. 1. It includes the typical phases of a genetic algorithm (such as fitness evaluation, selection, crossover and mutation), and other additional phases to promote diversity. During the niche assignment stage, several niches with similar chromosomes are created. In the next step, those similar chromosomes that are closest to the best one in their niche are deleted and replaced by new randomly generated individuals, thus increasing diversity. On the other hand, during the fitness scaling step, the fitness values of the rest of individuals of the niche are divided by a quantity that measures the similarity between that individual and the rest of individuals in the population. Specifically, the modified or shared fitness function \( f' \) of a chromosome \( C_i \) is calculated using the most widely used sharing function [9]:

\[
f'(C_i) = \frac{f(C_i)}{\mu \times \sum_{j=1}^{\mu} sh(d(C_i, C_j))}
\]

where \( f(C_i) \) is the fitness of the chromosome \( C_i \), \( \mu \) is the population size, \( d(C_i, C_j) \) is the distance between the chromosome \( C_i \) and the chromosome \( C_j \) and \( sh(d) \) is the sharing function, which measures the similarity between two elements.

Sharing is a fitness scaling mechanism that is applied prior to the selection process of a new population. Fig. 2 shows the effect of the sharing mechanism on the selection method of the present algorithm after ten generations. This selection method is the proportionate selection method, also known as the roulette wheel method as it emulates the operation of a roulette wheel. The wheel is divided into as many portions as the number of elements in the population and the size of each portion is proportional to the fitness of the corresponding element. The selection of the new population is done by spinning the wheel as many times as the population size and selecting each time the element indicated by the pointer [18, 19].

The distribution of the niches in the roulette wheel before applying the sharing method can be observed in Fig. 2 (a). The numbers above the percentages identify the different niches. Elements of niche 1 have higher fitness and cover a big part of the roulette. If sharing is not applied the result is that most elements in next generations will be members of niche 1 and then, diversity is lost and premature convergence can occur. However, when fitness is shared among similar individuals the result is that the roulette portions are more equally distributed as it is shown in Fig. 2 (b). Therefore, in this case the probability of selecting members of niche 1 is reduced and the probability of selecting members of other different niches, although with lower fitness, is increased. So, diversity is promoted and an improvement of the search and localization of individuals in optimal regions of the fitness landscape is obtained.

Finally, after a number of generations, the algorithm returns several high-fit chromosomes: the highest fitness chromosome and those chromosomes, belonging to other niches, whose fitness value is higher than half the highest fitness. As the returned chromosomes are from different niches, it is guaranteed that they are different good solutions.

### IV. EXPERIMENTS AND RESULTS

The algorithm has been tested with real data registered from a contest made with the QUESTOURNament tool in a Telecommunications Engineering course during the year 2008-2009. The contest was a compulsory activity, which was worth 10 percent of the final mark and lasted 3 weeks. Each week, 4 challenges were proposed at the beginning of a laboratory session. The challenges were IP addressing and routing exercises, some of them taken from exams of
previous years. Although many of the challenges could be solved during the laboratory session, all of them were almost one week long. Anyway, most students tried to solve the challenges as soon as possible as score varied with time. The contest consisted of 12 challenges. Initially, the teacher classified 4 challenges as easy, 3 as moderate, and 5 as hard. Although students did not know this initial classification, they knew the initial score of a challenge, which was set by the teacher according to the estimated difficulty level of the challenge. This initial score increased until reaching a maximum score or until a student answered the challenge correctly, when the score started decreasing. Then, the final score obtained by a student when answering a challenge depended on its difficulty, the moment when another student answered it correctly, the moment when the student answered and the grade obtained by the answer. The number of students participating was 38 and the total number of available answers for the easy, moderate and hard levels was 134, 103 and 169, respectively.

Table 1 shows the outputs of four executions of the genetic algorithm. It works with a population of 400 chromosomes. The number of generations is 15.

As it is shown in Table 1, the hard questions are characterized by two main sets of ranges. On the one hand, many answers have a very low grade and a high number of accesses. For example, according to execution 1, many answers to the hard questions were submitted between the minutes 1 and 48, with poor marks between 0 and 2 out of 100, and the corresponding questions were read by the students between 3 and 4 times. On the other hand, there is another important set of answers with a medium mark, around 50. Executions 3 and 4 found a third set of ranges for the hard questions, corresponding to higher grades and fewer accesses. In these two executions, the fitness value for these solutions is slightly higher than half the best fitness value obtained, which is the established limit to consider diverse solutions. Due to this proximity to the limit, these ranges of high grades are not always considered and selected by the algorithm, as it can be observed in executions 1 and 2.

For the moderate questions, three sets of intervals are found too. In this case answers with grades between 30 and 50 are predominant. For the easy questions, there is only one solution, which is substantially better than the other solutions. As expected, the easy questions are characterized by high grades and few accesses.

The differently characterized sets of answers may correspond to different students’ profiles or different knowledge levels, for example. Fig. 3 shows the fitness for the grade and time ranges obtained in execution 1. As it is expected, the range with highest fitness moves to lower grades as the difficulty level increases. Moreover, for the moderate questions, grades are more spread over the domain and then three peaks can be distinguished. The moderate questions are correctly answered by a considerable number of students and incorrectly answered by other important number of students. Besides, those students who devote more time to solve these questions get higher grades. For the hard questions, many students get very low grades and some get higher grades around 50. Only few of them get high grades.

The ranges found by the algorithm fit the real data so the algorithm performs correctly, even with the handicap of having a small data set.

TABLE I. TIME, GRADE AND ACCESSES RANGES FOUND BY THE DIVERSITY-ENHANCED GENETIC ALGORITHM FOR EACH DIFFICULT LEVEL

<table>
<thead>
<tr>
<th>Execution</th>
<th>Hard Level</th>
<th>Moderate Level</th>
<th>Easy Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time 1</td>
<td>[1,48] [13,36]</td>
<td>[0,2] [44,60] [3,4] [1,2]</td>
<td>0.143 0.107</td>
</tr>
<tr>
<td>Grade 1</td>
<td>[0] [2] [2] [2]</td>
<td>[0] [0] [0] [0]</td>
<td>[0] [0] [0] [0]</td>
</tr>
<tr>
<td>Accesses 1</td>
<td>[1,2] [1,2] [1,2] [1,2]</td>
<td>[1,2] [1,2] [1,2] [1,2]</td>
<td>[1,2] [1,2] [1,2] [1,2]</td>
</tr>
<tr>
<td>Fitness 1</td>
<td>0.143 0.107</td>
<td>0.176 0.107</td>
<td>0.143 0.107</td>
</tr>
<tr>
<td>Time 2</td>
<td>[2,80] [1,47]</td>
<td>[2,3] [3,4] [2,3] [1,]</td>
<td>0.138 0.132</td>
</tr>
<tr>
<td>Grade 2</td>
<td>[2,3] [2,3] [2,3] [2,3]</td>
<td>[2,3] [2,3] [2,3] [2,3]</td>
<td>[2,3] [2,3] [2,3] [2,3]</td>
</tr>
<tr>
<td>Accesses 2</td>
<td>[0,2] [1,2] [1,2] [1,2]</td>
<td>[0,2] [1,2] [1,2] [1,2]</td>
<td>[0,2] [1,2] [1,2] [1,2]</td>
</tr>
<tr>
<td>Fitness 2</td>
<td>0.138 0.132</td>
<td>0.138 0.132</td>
<td>0.138 0.132</td>
</tr>
<tr>
<td>Time 3</td>
<td>[3,68] [1,1,40]</td>
<td>[0,8] [1,2] [2,3] [1,2] [2,3] [1,2]</td>
<td>0.122 0.076</td>
</tr>
<tr>
<td>Grade 3</td>
<td>[0,8] [1,2] [2,3] [1,2] [2,3] [1,2]</td>
<td>[0,8] [1,2] [2,3] [1,2]</td>
<td>[0,8] [1,2] [2,3] [1,2]</td>
</tr>
<tr>
<td>Accesses 3</td>
<td>[0,8] [1,2] [2,3] [1,2] [2,3] [1,2]</td>
<td>[0,8] [1,2] [2,3] [1,2]</td>
<td>[0,8] [1,2] [2,3] [1,2]</td>
</tr>
<tr>
<td>Fitness 3</td>
<td>0.122 0.076</td>
<td>0.122 0.076</td>
<td>0.122 0.076</td>
</tr>
<tr>
<td>Time 4</td>
<td>[4,1,2] [1,5,64]</td>
<td>[0,10] [1,2] [2,3] [1,2]</td>
<td>0.143 0.107</td>
</tr>
<tr>
<td>Grade 4</td>
<td>[0,10] [1,2] [2,3] [1,2]</td>
<td>[0,10] [1,2]</td>
<td>0.107</td>
</tr>
<tr>
<td>Accesses 4</td>
<td>[0,10] [1,2] [2,3] [1,2]</td>
<td>[0,10] [1,2]</td>
<td>0.107</td>
</tr>
<tr>
<td>Fitness 4</td>
<td>0.143 0.107</td>
<td>0.143 0.107</td>
<td>0.143 0.107</td>
</tr>
</tbody>
</table>

V. CONCLUSIONS

This paper contributes with an innovative work that incorporates important differences from other approaches based on genetic algorithms.

The most similar system in the literature is the one described in [6], which estimates the difficulty level of the exercises proposed in a web-based system. However, the questions in this system can be solved through multiple attempts and receiving different hints, and then, most input parameters to the algorithm are not applicable to the present problem. Moreover, this algorithm needs an initial set of rules to characterize the different difficulty levels that must be defined by the teacher.

The proposed algorithm allows characterizing the challenges in the QUESTOURnament system, finding diverse good solutions. From these solutions, several rules are automatically defined to reclassify the questions according to the real difficulty found by the students.

Moreover, these solutions may be useful information to characterize students’ knowledge and behavior.
Next future work includes:

- Post-processing of the output of the genetic algorithm. From the ranges that the genetic algorithm delivers, several classification rules have to be defined.
- Definition of the process for re-classification of the difficulty level of the challenges, according to the previous post-processing solution.
- Improvement of the exploitation phase of the algorithm through the implementation of a simple local search method (e.g. hill-climbing) in order to refine the output of the genetic algorithm.
- Inclusion of more parameters related to the student’s profile in the structure of the response patterns.
- Studying the possibility of using the different solutions obtained by the enhanced genetic algorithm to detect and classify students with different knowledge levels and behaviors.

REFERENCES


