Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Personalized course generation and evolution based on genetic algorithms[*]

Xiao-hong TAN[†1,2], Rui-min SHEN[1,2], Yan WANG[2]
([1]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200030, China*)
([2]*E-learning Lab, Shanghai Jiao Tong University, Shanghai 200030, China*)
[†]E-mail: xhtan@sjtu.edu.cn

Received June 4, 2012; Revision accepted Sept. 27, 2012; Crosschecked Nov. 12, 2012

Abstract: Online learners are individuals, and their learning abilities, knowledge, and learning performance differ substantially and are ever changing. These individual characteristics pose considerable challenges to online learning courses. In this paper, we propose an online course generation and evolution approach based on genetic algorithms to provide personalized learning. The courses generated consider not only the difficulty level of a concept and the time spent by an individual learner on the concept, but also the changing learning performance of the individual learner during the learning process. We present a layered topological sort algorithm, which converges towards an optimal solution while considering multiple objectives. Our general approach makes use of the stochastic convergence of genetic algorithms. Experimental results show that the proposed algorithm is superior to the free browsing learning mode typically enabled by online learning environments because of the precise selection of learning content relevant to the individual learner, which results in good learning performance.

Key words: Genetic algorithm, Course generation, Course evolution, Personalized learning, Domain ontology
doi:10.1631/jzus.C1200174 **Document code:** A **CLC number:** TP391.7

1 Introduction

E-learning systems have become an indispensable component of education in the current society. In the last decades, research in e-learning has shown that the individual characteristics of online learners differ significantly with respect to prior knowledge, preferences, skill, and learning goals (Weber and Specht, 1997; Roland, 2000; Brusilovsky and Vassileva, 2003; Chen *et al*., 2005; Dabbagh, 2007). These differences require personalized course content, including different content and different sequences through the content.

Personalized learning honors each student as an individual learner, recognizing that each student has his/her own learning style, interest, aspirations, and challenges to learning, and supports each student to learn in his/her own unique way. Considerable research has been carried out on investigating personalized services, such as how the delivery of online courses should be adapted to individual demands, such as the learner's goals, experiences, and current knowledge level (Cristea and de Mooij, 2003; Hong *et al.*, 2007; Huang *et al.*, 2007; Bai and Chen, 2008; Chen, 2008; Bhaskar *et al.*, 2010; Chu *et al.*, 2011). Online courses are the main components of e-learning systems. Personalized courseware generation is the process of assembling a sequence of learning objects that are adapted to an individual user's learning goals, preferences, and capabilities. Research on course generation started early in the history of technology enhanced learning.

The existing work on courseware generation is abundant. The early research on the dynamic courseware generator (DCG) by Vassileva and Deters (1998) set the foundations for most of today's course

[*] Project supported by the National Natural Science Foundation of China (No. 61071154) and the project FP7 "Responsive Open Learning Environments" of European Union

generation. Later approaches in course generation (Cristea and de Mooij, 2003; Karampiperis and Sampson, 2004; Méndez *et al.*, 2004; Huang *et al.*, 2008) all used rather rudimentary pedagogical knowledge (Ullrich, 2008). The Paigos system uses hierarchical task network planning (HTN-Planning) and handles different learning goals (Ullrich and Melis, 2009). Recently, some new approaches have been applied to course composition to explore personalized learning, such as genetic algorithms (GAs) (Huang *et al.*, 2007; Chen, 2008; Wang and Tsai, 2009; Chu *et al.*, 2011), artificial neural networks (Baylari and Montazer, 2009), and particle swarm optimization (Dheeban *et al.*, 2010).

The GAs (Goldverg, 1989) have shown a good performance for solving a wide variety of problems. de-Marcos *et al.* (2007) modelled the learning sequence as a classical constraint satisfaction problem (CSP). Since then, research on the application of GAs to personalized learning has emerged. Jebari *et al.* (2011) used GAs to find suitable dynamic personalized course sequences of exercises within the solution space, respecting the constraints and maximizing student success. An automatic test generation system based on GAs (Meng *et al.*, 2007) and a genetic approach enhanced auto-reply scheme in an e-learning system (Hwang *et al.*, 2008) provide supplementary systems in personalized learning. Various approaches for determining an optimal learning path based on GAs have been explored in personalized learning systems. Hong *et al.* (2007) used improved CSP based on GAs to construct an adaptive learning path according to the incorrect test responses of individual learners in a pre-test. Their approach provides benefits in terms of learning performance. Huang *et al.* (2007) applied GAs to generate personalized learning paths based on case-based reasoning (CBR) for personalized knowledge. They considered the curriculum difficulty level and the continuity of successive curriculums. Bhaskar *et al.* (2010) applied GAs to evolve learning path generation into a learning scheme generation which accommodates each learner's entire context, including their profile, infrastructure, preference, and learning contexts. The problem with these approaches is that the learning path is used to navigate learners through a range of e-learning activities (Clement, 2000), but does not generate personalized learning objects adapted to each individual learner. Chen (2008) applied GAs to propose personalized curriculum sequencing based on pre-test and pre-learning performance. The pre-test for each individual learner is based on incorrect test responses. The personalized mechanisms are based on a pre-test performed at the start of the lesson, which can collect incorrect learning concepts of learners through computerized adaptive testing (CAT) (Huang *et al.*, 2007).

To summarize our literature review, although various studies have applied GAs to realize personalized learning with adaptive learning paths to improve the learning performance of individual learners, some important problems remain. Some of the research has explored personalization based on an individual learner's pre-test. However, these systems neglected the importance of the individual learner's changing performance in the learning process. Also, most research focused on exploring personalized learning paths for individual learners in the same learning system, but not the generation of different online courses including different learning objects.

This work presents a GA-based personalized course generation and evolution scheme (PCE-GA) which constructs personalized courses considering both the course difficulty level and each learner's changing performance in the learning process. It is an extension of existing research. The initial course of PCE-GA is generated according to the automatic course generation system (ACGS) (Tan *et al.*, 2010). ACGS provides course generation in a large-scale, teacher driven context, in which all the students are supposed to be in the same (virtual) class and to take exams at the same time. There, the content of the exams is set by the teachers. The content of the course is generated from a teaching outline and a domain structure constructed by teachers. The approach presented in this paper takes this initial course and evolves it based on GAs to meet individual learner's personalized needs in the learning process. The GAs are employed to construct a near optimal domain concept sequence and learning content according to the difficulty level of concepts and the time spent on them by each learner and, moreover, considering the learning performance of each learner in the learning process.

2 Course structure and system architecture

2.1 Course structure

In this course generation system, the domain concepts and the learning objects of a course are described as a domain ontology. Fig. 1 shows part of the domain ontology of the course data structure. The square nodes represent concepts and the elliptical nodes represent the learning objects. There are three types of relation, namely "including (has)", "prerequisite (requires)", and "for (is for)". The relation "including" represents the hierarchical relation of the concepts. The relation "prerequisite" represents the pre-/post-learning concepts among the courses. The relation "for" describes the relation between a learning object and a concept (namely that a learning object is for a concept, for instance, an example related to a concept). The structure of the concepts with "including" relations is a tree. The terminal nodes of this tree (for instance, "lists", "stacks", and "queues" in Fig. 1) are the core concepts, which are the smallest units of learning concepts (later, we will see that they also correspond to the smallest unit genes). The core concepts with "prerequisite" relations form a directed acyclic graph (DAG). The important part of the course generation is to construct the core concepts sequencing. The other concepts (non-terminal nodes in the tree) are added by the "including" relations to form the hierarchical index of the course. The learning objects are added by the "for" relations to generate the learning contents.

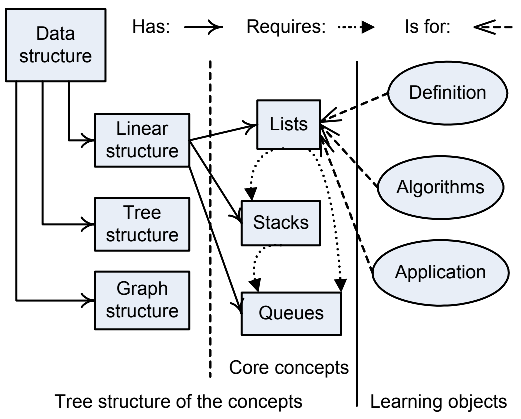

Fig. 1 An example of a course structure: part of the domain ontology of the course data structure

2.2 System architecture of the PCE-GA

The system architecture of the PCE-GA (Fig. 2) comprises an initial course generation module, an evaluation module, a module for the GA-based personalized course generation, and a course evolution module. The initial course (C_0) is generated using the course generation module ACGS. Note that the initial course (C_0) is not yet adapted to the individual student, but only to the overall course.

The evaluation module evaluates each student's learning performance. Basically, for each student it assigns a score, ranging from 0 to 5 for each concept. The domain ontology with scored concepts is called the personalized domain ontology. This is the input to the PCE-GA. The personalized course generated from PCE-GA is called C_i (i=1, 2, …, with i specifying the number of updates to C_0). During the learning process, the evaluation module evaluates the student's learning performance every week or two (configurable by the teacher). Accordingly, the course C_i is updated to C_{i+1} after each evaluation. This means that PCE-GA generates a sequence of personalized courses (C_i, i=1, 2, …) for each student in the learning cycle.

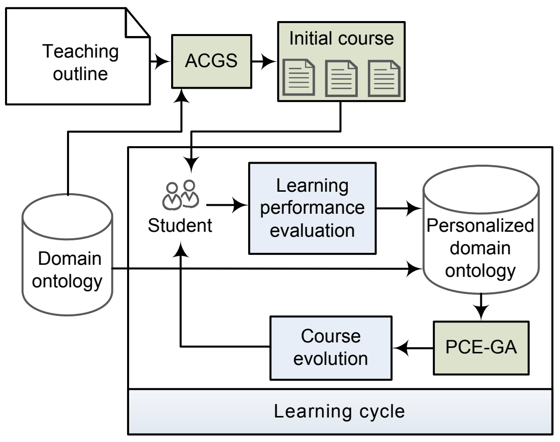

Fig. 2 Architecture and usage of the PCE-GA

3 Personalized course generation and evolution algorithm

3.1 Initial course generation

The generation of the initial course content is a basic task that has to be done before the course evolution can take place. In ACGS, the course generation is based on the domain concept graph, which is a DAG. In this system, we apply GAs to evolve the course during the learning process. Therefore, we have to consider the stochastic convergence of GAs. This means that an optimal solution should be converged from the multi-objective optimization problem, which is the process of simultaneously optimizing two or more conflicting objectives subject to certain constraints. Additionally, the sequential relationship between concepts should be retained. To address these two factors, we improve the initial course generation method of ACGS by introducing a layered topological sort (LTS) algorithm on the basis of the topological sort algorithm. A topological sort of a DAG is a linearly ordered list of its vertexes. There is not necessarily a unique linearly ordered list since some of the vertexes are exchangeable in the topological sorted list. The exchangeable vertexes in a linearly ordered list of a DAG are said to be on the same layer.

The definition of an LTS is as follows:

Let R be a partially ordered relation on the vertex set $M=(a_1, a_2, \ldots, a_n)$, with a_i being a vertex of the set M. For each vertex a_i, there is a weight l_i to mark its precedence in topological order.

Then, the vertex sequence $\boldsymbol{A}=((a_1, l_1), (a_2, l_2), \ldots, (a_n, l_n))$ is topologically ordered relative to the relation R. We call each (a_i, l_i) a binary combination.

For $0 \le i \le n-1$, for each two adjacent binary combinations (a_i, l_i) and (a_{i+1}, l_{i+1}), if it holds that $0 \le l_{i+1}-l_i \le 1$, then the element in the sequence $\boldsymbol{A}$ is defined as in layer-topological-order relative to the relation set R. The weight l_i is defined as the layer in the DAG. Vertexes with the same weight are exchangeable in the vertex sequence $\boldsymbol{A}$.

3.2 Estimation of parameters

Teachers define a general time spent on each concept when they construct the teaching outline. Since there are differences in comprehension for students, the time spent on a concept for each student is calculated as follows:

Suppose $t_{\mathrm{g}i}$ is general time spent on a concept i set by teachers. s_i is the score of this concept for a student, which is evaluated by the evaluation module in the learning process. Then t_i, the time spent on a concept i by the student, is

$$t_i=t_{\mathrm{g}i}(5-\mathrm{int}(s_i)),\ \ 0 \le s_i \le 5, \tag{1}$$

where $\mathrm{int}(s_i)$ is the integral function. For example, if $s_i=4.6$, $\mathrm{int}(s_i)=4$, $t_i=t_{\mathrm{g}i}$; if $s_i=3.5$, $\mathrm{int}(s_i)=3$, $t_i=2t_{\mathrm{g}i}$. Here a time coefficient t_e of concept i is defined as

$$t_\mathrm{e}=t_i/t_{\mathrm{g}i}. \tag{2}$$

This section gives an example of the estimation of the concept difficulty of a course. To design a course of "data structure", several experienced teachers were invited as experts to analyze the concepts and build up the teaching outline for this course. The experts wrote tests for each learning concept. Also, about 500 students who majored in computer science in the School of Continuing Education of Shanghai Jiao Tong University (SOCE-SJTU) took a test containing 100 test items covering the learning concepts of the course on data structures. Their test data were analyzed according to the item response theory in CAT (Baker, 1992) using the statistics-based BILOG program to obtain the appropriate difficulty parameters for these test items. Since the test item is related to the concept, it is assumed that the difficulty of the test item represents the difficulty of the corresponding concept.

3.3 Metadata of a gene

The core concepts in the DAG of Fig. 1 are the smallest units of genes of the chromosome. Since all the core concepts will appear in the chromosome, the chromosome coding on concept arrangement recodes the position of each gene.

To consider the learners' personalized needs in the fitness function, the gene is represented as a tuple (id, t_e, d, w_t, w_d, r), with id being the concept identifier, t_e being the time coefficient defined as in Eq. (2), d being the difficulty level of the concept, w_t being the time weight and w_d being the difficulty weight, and r being the general rank. The meaning of these symbols is as follows:

The time coefficient t_e and the difficulty attribute d are two important parameters used to calculate the adaptability of a chromosome. The domains of t_e and d are all [0, 5]. The time weight w_t and the difficulty weight w_d are two control parameters used by the developer to manually adjust the chromosome to fine-tune the time-base or the difficulty scale, in order to determine an order suitable to the learners' actual needs. For example, for students with good pre-knowledge and good understanding, we adjust w_d to a higher value and w_t to a lower value to enhance the course difficulty. Conversely, for students with weak comprehension and preferring to learn more, we adjust w_t to a higher value and w_d to a lower value to construct more learning objects in the course. The domains of w_t and w_d are all [0, 1]. The general rank r is the eigenvalue of the gene. In this setting, the value of r represents the comprehensive learning cost of a concept, and is calculated as

$$r=t_e w_t+d w_d. \tag{3}$$

According to the domain of the parameters, the domain of r is [0, 10]. The value of r plays an important role when designing the fitness function. In the following introduction of the fitness function, we explain in detail how to use the parameters.

3.4 Personalized course evolution

This section explains how to evolve the course based on GAs to meet each learner's needs. In general, GAs perform the optimization process in four stages: initialization, selection, crossover, and mutation (Davis, 1991). As for the optimization of learning contents, the process is solving the population of an abstract representation (named chromosome) of a certain quantity of concepts, evolving towards an organic combination of the optimal learning content sequence. The course evolution process is described in the flowchart (Fig. 3).

3.4.1 Initial population size

The population size affects the precision and performance of the GA directly. Generally, the initial population size can be determined according to the complexity of the solved problem. According to Huang *et al.* (2007), Bhaskar *et al.* (2010), and Jebari *et al.* (2011), a typical value of a population size is from 20 to 100. A large initial population size will slow down the search speed, but will improve the probability of optimization. Considering the scale of this course generated system, we take 50 as the default population size.

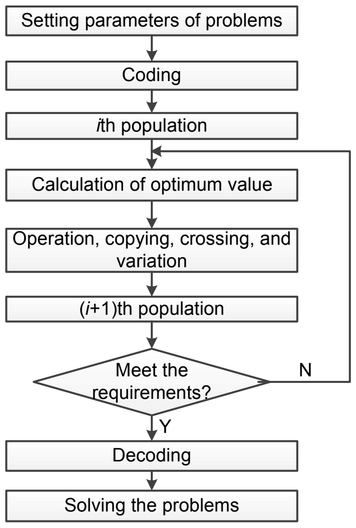
Setting parameters of problems
Coding
ith population
Calculation of optimum value
Operation, copying, crossing, and variation
(i+1)th population
Meet the requirements?
N
Y
Decoding
Solving the problems

Fig. 3 Flowchart of GA-based course evolution

3.4.2 Selecting the fitness function

The fitness function is applied to judge the quality of the generated learning concept sequence. To generate a personalized course, it is necessary to consider the difficulty level of a concept and the time spent on it for each student. In this paper, the value of general rank r of the gene represents these two important attributes. Therefore, r is considered to determine the fitness function. The average square deviation (variance) of r is adopted for measurement. The method is as follows: given a chromosome composed of a set of genes, then each gene has its own eigenvalue r_i, which is calculated as Eq. (3) specified in Section 3.3. The eigenvalue of the whole chromosome is $\boldsymbol{R}=(r_1, r_2, \ldots, r_n)$. Then, the calculation of the average square deviation is as follows:

First, calculate the average value of all the genes' eigenvalues in the whole chromosome:

$$\overline{r}=\frac{1}{n}\sum_{i=1}^{n} r_i. \tag{4}$$

Then, group the genes and put two neighboring genes in the same group, and calculate the average eigenvalue of each group. This is defined as the local gene eigenvalue:

$$r'=\frac{r_{i-1}+r_i}{2},\quad i=2,3,\ldots,n. \tag{5}$$

Finally, calculate the variance between local gene eigenvalues and the average value of the gene eigenvalues of the whole group of chromosomes:

$$f=\sum_{i=2}^{n}(r_i'-\bar{r})^2, \tag{6}$$

where f is the proposed fitness function for personalized course generation by the GAs. The smaller the value of f, the more evenly are the genes distributed in a complete chromosome.

3.4.3 Selection operation

To evolve the learning content sequence in the learning process according to an individual's learning performance, the individuals with a small fitness function value have a relatively high probability to be selected for the next generation. The tournament selection is a useful and robust selection mechanism (Miller and Goldberg, 1995). After repeated experiments, we choose the tournament selection rank-based fitness assignment to perform the reproduction operation in this study.

3.4.4 Crossover operation

The learning concept sequence rearrangement, which is the crossover operation, aims to combine two parent chromosomes to generate a child chromosome with better performance. In this system, considering the requirement that all the genes of the chromosomes have to appear and there should be no repetition, we choose the mapped partially matched crossover (PMX) method (Sivanandam and Deepa, 2007). In this method, two crossover points are selected at random to give a matching segment. The matching segment gives the mapping relations among genes. According to the mapping relationship, parents are mapped to each other to generate two child individuals. This method retains a segment of the genes, which is beneficial for the transmission of excellent characteristics. Meanwhile, the mapping relations ensure uniqueness of every gene in the chromosome.

This crossover operation was compared with the common single-point and two-point crossovers. In experiments, the generation was set to 1000 and the crossover probability was set to 0.5. With the same data set, the convergence properties of different crossovers were observed. The single-point crossover operator converged 382 times, the two-point crossover 396 times, and the proposed PMX crossover 478 times. The results show that the convergence property of the PMX crossover is more efficient than the common crossover operators.

3.4.5 Procedure for personalized course generation

In summary, the procedures of the proposed PCE-GA are as follows:

Step 1: Two excellent chromosomes of the parent population are run out through the tournament selection function described above.

Step 2: To generate a random value and to judge according to the crossover probability whether the crossover operation will be conducted. Either the crossover function will be used or the child chromosome will inherit the genes of the parent chromosomes directly.

Step 3: To judge according to the mutation probability whether to make a mutation operation or not. If a mutation operation is to be made, a mutation function will be used.

After the generation of the complete child chromosomes, we sort them with the rank population function, retain the setting of population size, and empty the present population, and then the child population becomes the current population.

4 Experiments

We carried out experiments to repeatedly test and determine the mutation probability and the crossover probability. In the experiments, the population size was set to 50.

First, we set the crossover probability to 0.5, took different values for the mutation probability, and observed the impact on the average fitness of the population and the generation of an optimal-adaptive chromosome. The test results (Fig. 4) show that the mutation probability has little influence on the population's average fitness. In contrast, a high average fitness will change the trend in population evolution.

Second, we set the mutation probability to 0.10, took different values for the crossover probability, and observed the impact on the average fitness and the generation of an optimal-adaptive chromosome. It is obvious that chromosomes with a larger crossover probability have a better performance during the search for an optimal solution (Fig. 5).

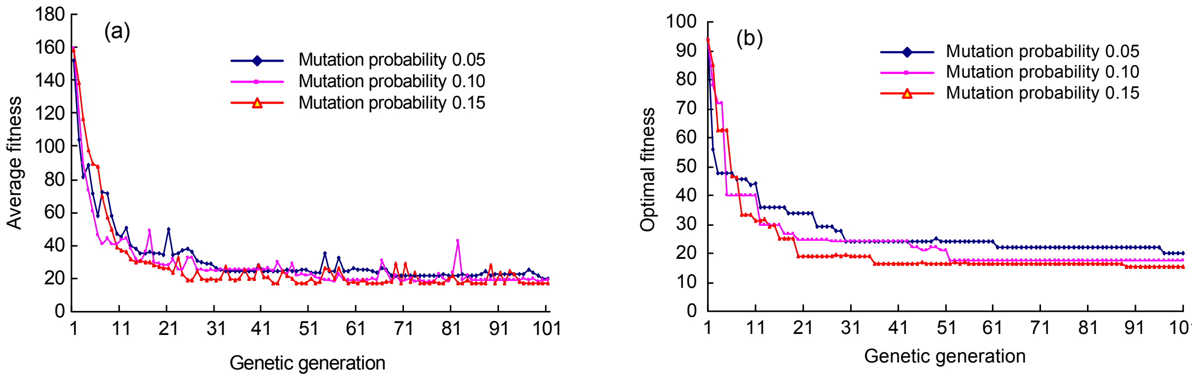

Fig. 4 Impact of mutation probabilities on average fitness (a) and optimal fitness (b) (crossover probability is 0.5)

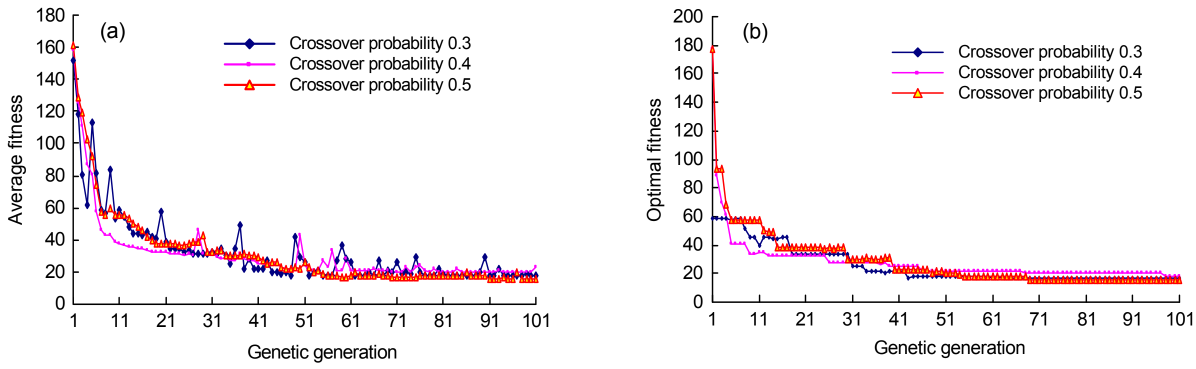

Fig. 5 Impact of crossover probabilities on average fitness (a) and optimal fitness (b) (mutation probability is 0.10)

With respect to the crossover probability and the mutation probability, larger values are more favorable, as they help to avoid premature convergence of a population. However, the large size of a newly generated chromosome also means a great deal of computation and the degradation of population evolution performance. Therefore, after the experiments, we decided to choose a crossover probability of 0.5 and a mutation probability of 0.15.

5 Implementation and evaluations

For assessing the usability of the PCE-GA, we used the system usability scale (SUS) (Brooke, 1996), which is a method frequently used by the human-computer interaction (HCI) community. It consists of 10 items used to measure user satisfaction with the usability of a system. It is easy to administer and there is a standard way to calculate the satisfaction score. All 10 questionnaire items are scored (0–4) on a five-point scale ranging from “strongly agree” to “strongly disagree”. For our questionnaire, we used the original 10 items in the SUS adapted slightly according to our learning system context: (1) I thought that I would like to use this system to learn frequently; (2) I found the system unnecessarily complex; (3) I thought the system was easy to use; (4) I thought that I would need the support of a technical person to be able to use this system; (5) I found that the various functions in this system were well integrated; (6) I thought that there was too much inconsistency in this system; (7) I would image that most learners would learn to use this system very quickly; (8) I found the system very cumbersome to use; (9) I felt very confident of using the system; (10) I need to learn a lot of things before I could get going with this system.

The SUS score of the PCE-GA was calculated in two steps: first calculating the SUS score of each respondent, and then averaging the scores of the total 290 respondents (from the 295 students who participated in the survey). We followed the standard method to calculate the SUS score of each respondent. This means first summing up the score contributions from each item, which ranged from 0 to 4. For Items 1, 3, 5, 7, and 9, the score contribution is the scale position minus 1, while for Items 2, 4, 6, 8, and 10, the contribution is 5 minus the scale position. We then multiplied the sum of the scores by 2.5 to obtain the overall value of SUS. Consequently, SUS scores ranged from 0 to 100. The higher the score, the more robust and reliable is the system.

To evaluate the efficiency of the learning system with personalized course generation, we performed a descriptive analysis of participants' and nonparticipants' grades. A sample of 530 students enrolled in the DS course in the 2011 fall semester. They were selected and categorized into two groups: participants who took the DS course in the system with PCE-GA, and nonparticipants who took the DS course in the normal e-learning platform at SOCE-SJTU. After learning, all learners took part in the same test and were assigned grades. To determine which learning model was more effective, grades were compared between the two groups in the two different learning models. Because the grades of the two groups appear as skewed distributions, the Mann-Whitney U test (a common nonparametric test for independent samples) was used to compare the differences between the two independent groups. Table 1 shows the U-test results. The mean rank of participants was 312.56, while the mean rank of nonparticipants was 246.32. The results show that the participants' grades were significantly higher than the nonparticipants' grades (n=530, U=39169.8, α=0.05, P<0.0001). This strongly indicates that the personalized learning courses from PCE-GA are efficient.

Table 1 Mann-Whitney U test for comparing the performance of participants and nonparticipants

Group	N	Rank sum	Mean rank	U
Participants	295	92205.2	312.56	20779.8
Nonparticipants	235	57885.2	246.32	39169.8

95% confidence; Mann-Whitney U: 39169.8

6 Conclusions and future work

The aim of this system is to provide a personalized online learning course for individual learners in large scale online education. A PCE-GA method is proposed. In the PCE-GA presented, the domain concepts and the learning objects of a course are described as a domain ontology. To consider the stochastic convergence of GAs, an LTS method is applied. The generated courses consider not only the concept difficulty level and the time spent on each concept, but also the learning performance of individual students during the learning process. The personalized course can be evolved in the learning process according to the updated personalized concept graph.

Further work will be devoted to extending the personalized mechanism to handle more complex individual characteristics and behaviors of the learners. We will also research how to help learners learn more effectively in the personalized learning environment.

References

Bai, S.M., Chen, S.M., 2008. Automatically constructing concept maps based on fuzzy rules for adapting learning systems. *Expert Syst. Appl.*, **35**(1-2):41-49. [doi:10.1016/j.eswa.2007.06.013]

Baker, F.B., 1992. Item Response Theory: Parameter Estimation Techniques. Marcel Dekker, New York, p.440.

Baylari, A., Montazer, G.A., 2009. Design a personalized e-learning system based on item response theory and artificial neural network approach. *Expert Syst. Appl.*, **36**(4):8013-8021. [doi:10.1016/j.eswa.2008.10.080]

Bhaskar, M., Das, M.M., Chithralekha, T., Sivasatya, S., 2010. Genetic algorithm based adaptive learning scheme generation for context aware E-learning. *Int. J. Comput. Sci. Eng.*, **2**(4):1271-1279.

Brooke, J., 1996. SUS—A Quick and Dirty Usability Scale. *In*: Jordan, P.W., Thomas, B., Weerdmeester, B.A., *et al.* (Eds.), Usability Evaluation in Industry. Taylor & Francis, London.

Brusilovsky, P., Vassileva, J., 2003. Course sequencing techniques for large-scale education. *Int. J. Cont. Eng. Educ. Lifelong Learn.*, **13**(1-2):75-94.

Chen, C., 2008. Intelligent Web-based learning system with personalized leaning path guidance. *Comput. Educ.*, **51**(2): 787-814. [doi:10.1016/j.compedu.2007.08.004]

Chen, C., Lee, H., Chen, Y., 2005. Personalized e-learning system using item response theory. *Comput. Educ.*, **44**(3): 237-255. [doi:10.1016/j.compedu.2004.01.006]

Chu, C., Chang, Y., Tsai, C., 2011. PC^{2}PSO: personalized e-course composition based on particle swarm optimization. *Appl. Intell.*, **34**(1):141-154. [doi:10.1007/s10489-009-0186-7]

Clement, J., 2000. Model based learning as a key research area for science education. *Int. J. Sci. Educ.*, **22**(9):1041-1053. [doi:10.1080/095006900416901]

Cristea, A.I., de Mooij, A., 2003. Adaptive Course Authoring: My Online Teacher. Proc. 10th Int. Conf. on Telecommunications, p.1762-1769. [doi:10.1109/ICTEL.2003.1191699]

Dabbagh, N., 2007. The online learner: characteristics and pedagogical implications. *Contemp. Issues Technol. Teach. Educ.*, **7**(3):217-226.

Davis, L.D., 1991. Handbook of Genetic Algorithms. van Nostrand Reinhold, New York.

de-Marcos, L., Pages, C., Martinez, J.J., Gutierrez, J.A., 2007. Competency-Based Learning Object Sequencing Using Particle Swarms. 19th IEEE Int. Conf. on Tools with Artificial Intelligence, p.111-116. [doi:10.1109/ICTAI.2007.14]

Dheeban, S.G., Deepak, V., Dhamodharan, L., Elias, S., 2010. Improved personalized e-course composition approach using modified particle swarm optimization with inertia-coefficient. *Int. J. Comput. Appl.*, **1**(6):102-107. [doi:10.5120/134-252]

Goldverg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Hong, C.M., Chen, C.M., Chang, M.H., Chen, S.C., 2007. Intelligent Web-Based Tutoring System with Personalized Learning Path Guidance. 7th IEEE Int. Conf. on Advanced Learning Technologies, p.512-516.

Huang, M., Huang, H., Chen, M., 2007. Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach. *Expert Syst. Appl.*, **33**(3):551-564. [doi:10.1016/j.eswa.2006.05.019]

Huang, Y., Chen, J., Huang, T., Jeng, Y., Kuo, Y., 2008. Standardized course generation process using dynamic fuzzy Petri nets. *Expert Syst. Appl.*, **34**(1):72-86. [doi:10.1016/j.eswa.2006.08.030]

Hwang, G.J., Yin, P., Wang, T., Tseng, J., Hwang, G.H., 2008. An enhanced genetic approach to optimizing auto-reply accuracy of an e-learning system. *Comput. Educ.*, **51**(1):337-353. [doi:10.1016/j.compedu.2007.05.014]

Jebari, K., EI Moujahid, A., Bouroumi, A., Ettouhami, A., 2011. Genetic Algorithms for Online Remedial Education Based on Competency Approach. Int. Conf. on Multimedia Computing and Systems, p.1-6. [doi:10.1109/ICMCS.2011.5945603]

Karampiperis, P., Sampson, D., 2004. Adaptive Instructional Planning Using Ontologies. Proc. IEEE Int. Conf. on Advanced Learning Technologies, p.126-130. [doi:10.1109/ICALT.2004.1357388]

Méndez, N.D.D., Ramírez, C.J., Luna, J.A.G., 2004. IA Planning for Automatic Generation of Customized Virtual Courses. Proc. European Conf. on Artificial Intelligence and Applications.

Meng, A., Ye, L., Roy, D., Padilla, P., 2007. Genetic algorithm based multi-agent system applied to test generation. *Comput. Educ.*, **49**(4):1205-1223. [doi:10.1016/j.compedu.2006.01.012]

Miller, B.L., Goldberg, D.E., 1995. Genetic algorithms, tournament selection, and the effects of noise. *Compl. Syst.*, **9**(3):193-212.

Roland, H., 2000. Logically optimal curriculum sequences for adaptive hypermedia systems. *LNCS*, **1892**:121-132. [doi:10.1007/3-540-44595-1_12]

Sivanandam, S.N., Deepa, A.N., 2007. Introduction to Genetic Algorithms. Springer Publishing Company.

Tan, X., Ullrich, C., Wang, Y., Shen, R., 2010. The Design and Application of an Automatic Course Generation System for Large-Scale Education. IEEE 10th Int. Conf. on Advanced Learning Technologies, p.607-609. [doi:10.1109/ICALT.2010.172]

Ullrich, C., 2008. Pedagogically Founded Courseware Generation for Web-Based Learning—An HTN-Planning Based Approach Implemented in PAIGOS 5260.

Ullrich, C., Melis, E., 2009. Pedagogically founded courseware generation based on HTN-planning. *Expert Syst. Appl.*, **36**(5):9319-9332. [doi:10.1016/j.eswa.2008.12.043]

Vassileva, J., Deters, R., 1998. Dynamic courseware generation on the WWW. *Br. J. Educ. Technol.*, **29**(1):5-14. [doi:10.1111/1467-8535.00041]

Wang, T.I., Tsai, K.H., 2009. Interactive and dynamic review course composition system utilizing contextual semantic expansion and discrete particle swarm optimization. *Expert Syst. Appl.*, **36**(6):9663-9673. [doi:10.1016/j.eswa.2008.12.010]

Weber, G., Specht, M., 1997. User Modeling and Adaptive Navigation Support in WWW-Based Tutoring Systems. Proc. 6th Int. Conf. on User Modeling, p.289-300.