An Intelligent Testing System Embedded With an Ant-Colony-Optimization-Based Test Composition Method

Xiao-Min Hu, *Student Member, IEEE*, Jun Zhang, *Senior Member, IEEE*, Henry Shu-Hung Chung, *Senior Member, IEEE*, Ou Liu, and Jing Xiao

***Abstract*—Computer-assisted testing systems are promising in generating tests efficiently and effectively for evaluating a person's skill. This paper develops a novel intelligent testing system for both teachers and students. Based on the Browser/Server structure, the proposed testing system comprises a question bank and five modules, offering the features of self-adaptation, reliability, and flexibility for generating parallel tests with identical test ability. The core of the developed system is the ant-colony-optimization-based test composition (ACO-TC) method, which aims at generating high-quality tests for examinations and satisfying multiple requirements. As an advanced computational intelligence algorithm, the proposed ACO-TC method uses a colony of ants to select appropriate questions from a question bank to construct solutions. Pheromone and heuristic information is designed for facilitating the ants' selection. The system is analyzed by composing tests in different situations. The generated tests not only match the expected total completion time, the concept proportions, the average difficulty, and the score proportions of different question types, but also have high average discrimination degrees of questions. The experimental results also show that the system can always generate high-quality tests from question banks with various sizes.**

***Index Terms*—Ant colony optimization (ACO), computer assisted, intelligent education system, test composition, testing system, tutoring system.**

I. Introduction

DEVELOPMENT of computer and network technologies has brought in great advancement in the education system. Various computer-assisted application platforms have been built, such as intelligent tutoring systems (ITSs) [1]–[3], distance learning systems (DLSs) [4]–[6], virtual laboratories [7], [8], and computerized adaptive testing (CAT) systems [9]–[13]. These systems, which are now regarded as parts of the e-learning systems [14], [15], have facilitated the traditional teaching-learning-evaluation methods, thus making education more flexible and diverse.

Tests are generally the most common and effective way in evaluating a learner's knowledge or ability. Traditionally, teachers or examiners need to take days or even weeks to compose a test, but the test cannot always satisfy the need in discriminating the learners' knowledge, and the attributes such as the test completion time and the difficulty degree of a test are hard to be controlled. In modern education, computer-assisted testing systems are promising in generating tests more efficiently and effectively for evaluating a person's skill. As early as the 1990s, some social certification examinations, such as Graduate Record Examination (GRE) and Test of English as a Foreign Language (TOEFL), have adopted computerized testing systems. Multitest II [16], a program for the generation, correction, and analysis of multiple choice tests, aimed at randomly arranging questions (also called *items*) in a master test with a different order for reducing possible student cheating and helping automatic test correction and grading. Chou [9] described a computer-assisted testing and evaluation system (CATES), which focused on the usage of computer networks and the Web for testing and evaluation. Although the difficulty, discrimination, reliability, and validity of the test itself were not considered in [9], the author showed us the structure of the CATES and the students' reactions to the Web-based testing. Compared to the traditional paper-and-pencil media, computer-assisted testing platforms are more favored by students. In addition, personalized assessments tailored to each student are the developing trends [10]–[12]. Personalized CAT systems select an appropriate question from the question bank based on the examinee's answer to the previous question. Ho and Yen [12] also showed that the platforms used by the examinees, such as a PC or a personal digital assistant (PDA), did not affect the performance of the CAT.

The motivation for developing a testing system in this paper is to provide a platform for evaluating students' learning states and compose high-quality tests for examinations. Although different testing systems have been built in the literature, the questions in a test are randomly selected from the question bank [9] or selected based on a simplified optimization model [13]. In this paper, not only are the total completion time and the concept proportions considered, but also the average difficulty of the test

Manuscript received February 3, 2009; revised March 12, 2009. First published June 2, 2009; current version published October 16, 2009. This work was supported in part by the National Science Foundation (NSF) of China under Project 60573066, in part by the NSF of Guangdong under Project 5003346, in part by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, China, in part by the National Natural Science Foundation of China (NSFC) Joint Fund with Guangdong under Key Project U0835002, in part by the National High-Technology Research and Development Program ("863" Program) of China (2009–2010) no. 2009AA01Z208, and in part by a departmental general research fund of the Hong Kong Polytechnic University (G-U442). This paper was recommended by Associate Editor N. Attoh-Okine.

X.-M. Hu, J. Zhang, and J. Xiao are with the Department of Computer Science, Sun Yat-Sen University, Guangzhou 510275, China.

H. S.-H. Chung is with the Department of Electronic Engineering, City University of Hong Kong (CityU), Kowloon, Hong Kong.

O. Liu is with the Hong Kong Polytechnic University, Kowloon, Hong Kong.

Digital Object Identifier 10.1109/TSMCC.2009.2021952

and the expected score proportions of different question types are the requirements for composing tests. The developed testing system has the following characteristics.

1) Self-adaptive: Question attributes in a question bank are adaptively updated to reflect students' newest learning states. When a student logs into the system for exercises, each question will be marked after the student has finished answering the question. Then, the values of question attributes (such as the time needed for answering the question, difficulty degree, and discrimination degree of the question) will be adaptively updated. Abnormal situations will not be considered for performing the adjustment.
2) Reliable: As the attribute values of each question in the bank are adjusted according to the outcome of exercises taken by students, the input sources for the test composition are reliable. Given multiple testing requirements, the proposed system can compose tests with high assessment qualities. Therefore, the tests are reliable for evaluating students' grasp of knowledge.
3) Flexible for generating parallel tests with identical testing ability. If a student missed the exam for some reasons such as disease, a makeup exam can be scheduled in another time conveniently.

Besides, the ant-colony-optimization-based test composition (ACO-TC) method embedded in the proposed system is also a powerful searching method for selecting questions to construct high-quality tests. Since the ACO framework was first proposed by Dorigo *et al.* [17]–[19], variants of ACO algorithms have been successfully applied in various fields, such as routing [20], power electronic design [21], assignment [22], scheduling [23], subset problems [24], and other machine-learning problems [25]. However, using an ACO core for test composition is a new attempt. Based on our analysis, the construction behavior of ants in ACO suits the common process for selecting questions in a test. Details about the strategies used in ACO-TC will be presented in Section III.

The proposed testing system platform has been implemented. The system utilizes the Browser/Server (B/S) structure, and comprises a question bank, a question maintenance module, an evaluation and analysis module, an online exercising module, a test generation module, and an online testing module. Teachers and students can log into the system as the corresponding users with different rights, such as generating tests or taking exercises. Interfaces for composing tests and taking exams will be described in Section IV.

The rest of the paper is organized as follows. Section II introduces formal definitions of questions and their attributes. The self-adaptation strategy for adjusting question attributes is also described. Section III presents the ACO-TC method in detail, including descriptions on the test composition model, ants' construction behavior, and the overall implementation. The proposed testing system platform is described in Section IV. Section V analyzes the performance of the proposed system for composing tests under different requirements. Finally, in Section VI, conclusions of our research and guidelines for the further work are given.

II. Question Attributes in a Test

A. Question Attributes

A test is composed of n ($n \geq 1$) question(s) for students to answer. Each question has several attributes, such as a unique id number i, type y_i, difficulty degree d_i, discrimination degree e_i, completion time t_i, and the concept(s) that the question involves.

Suppose there are Y types of questions, e.g., multiple-choice-type-1, multiple-choice-type-2, fill in the blank, true or false, ordering, and short essay questions, etc. Each question type k corresponds to a score value $v_k (k = 1, 2, \ldots, Y)$, thus the score proportion s_k of type k in the test can be calculated as

$$s_k = \frac{\sum_{i=1}^{n} \{v_{y_i} | y_i = k\}}{\sum_{i=1}^{n} v_{y_i}} \tag{1}$$

where y_i denotes the type of question i and $\{v_{y_i} | y_i = k\}$ denotes the score of question i whose type is k. As one question has only one type, so

$$\sum_{k=1}^{Y} s_k = 1. \tag{2}$$

The question difficulty degree is generally a scoring rate of a question. The formula for computing the difficulty degree of question i is

$$d_i = \frac{\varphi_i}{\lambda_i v_{y_i}} \tag{3}$$

where φ_i is the total score earned by the students who have answered the question correctly and λ_i is the number of students who have done the question. The difficulty degree ranges in [0.00, 1.00]. If no one has achieved score from the question, the difficulty degree is 0.00. If all students can answer the question correctly, the difficulty degree is 1.00. The average difficulty degree of a test can be denoted as D, where

$$D = \frac{\sum_{i=1}^{n} d_i}{n}. \tag{4}$$

The question discrimination degree indicates a question's ability to discriminate between the students who know the knowledge and those who do not. Generally, it is computed by ranking the students according to the total score. Then based on Kelley's "27% of sample" [26], select the 27% upper scoring students and the 27% lower scoring students in terms of the total score. The discrimination degree of question i is the difference between the difficulty degree $d_i^{(\mathrm{upper})}$ of question i to the upper scoring students and the difficulty degree $d_i^{(\mathrm{lower})}$ of question i to the lower scoring students

$$e_i = d_i^{(\mathrm{upper})} - d_i^{(\mathrm{lower})}. \tag{5}$$

The value of a discrimination degree ranges in [−1.00, 1.00]. The higher the discrimination degree, the better the question does in evaluating the students' knowledge. A discrimination degree that is no smaller than 0.3 is usually regarded as acceptable. If the discrimination degree is smaller than zero, the question is not suitable for the test and should be deleted. The average discrimination degree of a test is E, where

$$E = \frac{\sum_{i=1}^{n} e_i}{n}. \tag{6}$$

As questions are used for assessing whether the student has grasped the concept(s), each question is related with one or more concept(s). Suppose M concepts are checked in the test. Using a 0/1 representation scheme, the relations between concepts and questions can be formulated as

$$\begin{cases} r_{ji} = 1, & \text{if question } i \text{ is related with concept } j \\ r_{ji} = 0, & \text{otherwise} \end{cases} \tag{7}$$

where $j = 1, 2, \ldots, M$, and $i = 1, 2, \ldots, n$. The proportion c_j of concept j in a test is

$$c_j = \frac{\sum_{i=1}^{n} r_{ji}}{n}. \tag{8}$$

As one question can involve more than one concept, so

$$\sum_{j=1}^{M} c_j \geq 1. \tag{9}$$

Each question i also has an estimated completion time t_i. The duration T of a test equals to the sum of the completion time of the n questions. The notations used in this paper are given in Table I.

TABLE I
NOTATIONS

Symbol	Meaning
n	Number of questions in a test
y_i	Type of question i
d_i	Difficulty degree of question i
e_i	Discrimination degree of question i
t_i	Completion time of question i
Y	Number of question types
v_i	Score of type i
s_k	Score proportion of type k
φ_i	Total score earned by the students who have answered the question i correctly
λ_i	Number of students who have done the question i
D	Average difficulty degree of a test
E	Average discrimination degree of a test
M	Number of concepts
r_{ji}	Relation between concept j and question i
c_j	Proportion of concept j
T	Total completion time of a test
d_i'	Updated difficulty degree of question i
e_i'	Updated discrimination degree of question i
t_i'	Updated completion time of question i
σ	Score earned by a student
ϕ_i	Total time spent by students on question i
ς	Time spent by a student for completing the question
$\tilde{D}$	Expected average difficulty of a test
$\tilde{T}$	Expected total completion time of a test
$\tilde{s}_k$	Expected score proportion of type k
$\tilde{c}_j$	Expected proportion of concept j
N	Total number of questions in a question bank
q_{l1}	Penalty of the difficulty degree of the lth test
q_{l2}	Penalty of the total completion time of the lth test
q_{l3}	Penalty of the score proportion of the lth test
q_{l4}	Penalty of the concept proportion of the lth test
$\omega_1, \omega_2, \omega_3, \omega_4$	Penalty weights
L	Number of parallel tests to be composed
Q	Set of questions in the question bank
$\Omega^{(a)}$	Candidate list of ant a
x_{li}	Indicate whether question i is selected (=1) or not (=0) to the lth test
τ_{li}	Pheromone value on question i in the lth test
ε	Small positive value
ρ	Pheromone evaporation rate
α	Pheromone reinforcement value
m	Number of ants

B. Self-Adaptation Strategy for Adjusting Question Attributes

When a new question is created and added to the question bank, it will be assigned a unique id number. The question type and the related concepts are already known, but the difficulty degree, the discrimination degree, and the completion time are still unknown. So the administrator can only assign estimated values to the attributes of the new question. Because those attribute values influence the generated test, the attribute values must be adjusted.

As there are always thousands of questions in a question bank, manual adjustments of those attribute values are unrealistic. Therefore, the proposed system opens the question bank for students as exercises, and adaptively update the question attributes according to the completion of the question. Every time when a student takes question i $(i = 1, 2, \ldots, n)$ as an exercise, the difficulty degree, the discrimination degree, and the completion time are updated as

$$d_i' = \frac{\varphi_i + \sigma}{(\lambda_i + 1)v_{y_i}} \tag{10}$$

$$e_i' = d_i'^{(\text{upper})} - d_i'^{(\text{lower})} \tag{11}$$

$$t_i' = \begin{cases} \dfrac{\phi_i + \varsigma}{\lambda_i + 1}, & \text{if } \sigma > 0 \\ t_i, & \text{otherwise} \end{cases} \tag{12}$$

where d_i', e_i', and t_i' are the updated values, σ is the score earned by the student, ϕ_i is the total time spent on the question by the previous students, and ς is the time spent by the current student for answering the question. If the answer is wrong, which means $\sigma = 0$, the completion time will not be updated.

Abnormal situations in doing exercises must be distinguished by the system. When a student logs into the system for exercises, he/she can choose the concepts related to questions, but which questions are assigned to the student is determined by the system. Each time the student can only view a question. If the student submits a wrong answer deliberately, in most cases, he/she does not know how to answer the question, so that the student's completion time is useless. Moreover, if the predefined idle timeout for answering the question exceeds, the answer to the question is deserted.

III. ACO-Based Test Composition

In this section, the ACO-based test composition method embedded in the proposed testing system is described. The user-defined test composition optimization model is first presented. Then, we detail the ants' construction behavior in the proposed method. Finally, the overall implementation is illustrated.

A. User-Defined Test Composition Optimization Model for Composing Tests

Teachers upload their requirements to the system before composing tests. Suppose a test is expected to satisfy the following requirements.

1) $\tilde{D}$: This variable represents the expected average difficulty of the test.
2) $\tilde{T}$: It represents the expected total completion time of the test.
3) $\tilde{s}_k$: It represents the expected score proportion of type k ($k = 1, 2, \ldots, Y$).
4) $\tilde{c}_j$: It represents the expected proportion of concept j ($j = 1, 2, \ldots, M$).

The optimization objective is to generate L ($L \geq 1$) test(s) with the highest average discrimination degree, satisfying the previous requirements. As stated in Section I, the third characteristic of the proposed system is its flexibility in generating parallel tests. Since the question attribute values in the bank are adaptively changed, it is impossible to generate a makeup test with the identical test ability as the original test in other time. Therefore, a function for generating multiple tests with identical test ability simultaneously is designed in the proposed system.

Suppose the total number of questions in the bank is N. We use x_{li} to represent whether a question i is selected to the lth test. If question i is selected, then $x_{li} = 1$; otherwise, $x_{li} = 0$, $i = 1, 2, \ldots, N, l = 1, 2, \ldots, L$. Using n_l to symbolize the total question number in the lth test, we have

$$n_l = \sum_{i=1}^{N} x_{li}. \tag{13}$$

Thus, the optimization model for generating L test(s) can be defined as

$$\text{maximize } F = \sum_{l=1}^{L} \frac{f_l}{L} - \sum_{L \geq b_1 > b_2 \geq 1} |f_{b_1} - f_{b_2}| \tag{14}$$

where

$$f_l = \frac{\sum_{i=1}^{N} e_i x_{li}}{n_l} \tag{15}$$

subject to

$$D_l = \frac{\sum_{i=1}^{N} d_i x_{li}}{n_l} = \tilde{D} \tag{16}$$

$$T_l = \frac{\sum_{i=1}^{N} t_i x_{li}}{n_l} = \tilde{T} \tag{17}$$

$$s_{lk} = \frac{\sum_{i=1}^{N} \{v_{y_i} x_{li} | y_i = k\}}{\sum_{i=1}^{N} v_{y_i} x_{li}} = \tilde{s}_k, \quad k = 1, 2, \ldots, Y \tag{18}$$

$$c_{lj} = \frac{\sum_{i=1}^{N} r_{ji} x_{li}}{n_l} \geq \tilde{c}_j, \qquad j = 1, 2, \ldots, M \tag{19}$$

$$\sum_{i=1}^{N} x_{b_1 i} x_{b_2 i} = 0, \qquad L \geq b_1 > b_2 \geq 1. \tag{20}$$

Equation (14) requires that the average discrimination degree of the tests is maximized and the difference between any two tests is minimized. Equation (15) is the average discrimination degree of the lth test. Equations (16)–(19) are constraints of requirements to be satisfied, including the average difficulty, the total completion time, the score proportion of each question type, and the proportion of concepts. In particular, (19) requires that the proportion of each concept j should not be smaller than the expected proportion $\tilde{c}_j$, $j = 1, 2, \ldots, M$. Equation (20) restricts that no questions in the generated tests could be identical. If the earlier constraints are violated, the following penalties will be added in (15), respectively.

For the expected average difficulty degree

$$q_{l1} = |\tilde{D} - D_l|. \tag{21}$$

For the expected total completion time

$$q_{l2} = \left| \frac{\tilde{T} - T_l}{\tilde{T}} \right|. \tag{22}$$

For the expected score proportion

$$q_{l3} = \frac{\sum_{k=1}^{Y} \left\lfloor \left| \tilde{s}_k \sum_{i=1}^{N} v_{y_i} x_{li} - \sum_{i=1}^{N} \{v_{y_i} x_{li} | y_i = k\} \right| / v_k \right\rfloor}{2}. \tag{23}$$

For the expected concept proportion

$$q_{l4} = \sum_{j=1}^{M} \left\lceil \max(0, \tilde{c}_j n_l - \sum_{i=1}^{N} r_{ji} x_{li}) \right\rceil \tag{24}$$

where the symbol $\lfloor \; \rfloor$ returns the largest integer that is less than or equal to the input parameter, the symbol $\lceil \; \rceil$ returns the smallest integer that is greater than or equal to the input parameter, and the function max() returns the maximum value of the input parameters.

The first two equations [(21) and (22)] are absolute difference and relative difference between the expected values and the actual values for the average difficulty degree and the total completion time, respectively. Note that the penalties in (21) and (22) are floating-point values in the range of [0, 1]. On the other hand, the penalties in (23) and (24) are relatively larger. The penalty in (23) is the total number of questions that violate the expected score proportion of each type. To the penalty of the concept proportion in (24), if questions in the test satisfy the proportional requirement of all concepts, no penalty is added; otherwise, the penalties are the number of questions that are further needed for meeting the concept requirement.

Considering penalties, the evaluation function of the optimization problem is modified as maximizing the value of

$$\tilde{F} = \sum_{l=1}^{L} \frac{\tilde{f}_l}{L} - \sum_{L \geq b_1 > b_2 \geq 1} \left| \tilde{f}_{b_1} - \tilde{f}_{b_2} \right| \tag{25}$$

where

$$\tilde{f}_l = f_l - \omega_1 q_{l1} - \omega_2 q_{l2} - \omega_3 q_{l3} - \omega_4 q_{l4} \tag{26}$$

where ω_1, ω_2, ω_3, and ω_4 are weights for the penalties.

The generated tests are acceptable if all of the following criteria have been satisfied ($l = 1, 2, \ldots, L$).

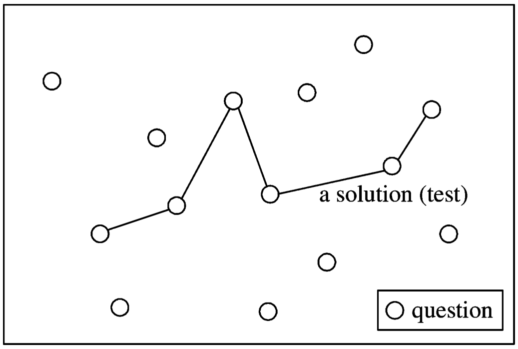


Fig. 1. Illustration of a solution constructed by an ant. The connected questions form a test.

1) $q_{l4} = 0$, which means that there is no concept proportion penalty.
2) $q_{l3} = 0$, which means that there is no score proportion penalty.
3) $q_{l2} < 10\%$, which means that the violation to the expected time is smaller than 10% of the expected time.
4) $q_{l1} < 0.1$, which means that the difference between the expected average difficulty degree and the resulting degree is smaller than 0.1 (one difficulty level).
5) $f_l > 0$, which means that the average discrimination degree of the test must be larger than zero.

B. Ants' Construction Behavior for Composing Tests

ACO simulates the foraging behavior of real ants to search for the optimal solution of a problem. In every iteration of ACO, m ants are dispatched for constructing their own solutions. As stated in [19], the ants in ACO are stochastic constructive procedures that build solutions by moving on a construction graph. The basic framework of ACO is composed of three main processes, i.e., the ants' solution construction process, an optional local search, and the pheromone update. The three processes iterate until the termination condition is satisfied. In the following part, the realization of the proposed ACO-TC is presented in detail.

1) Construction Graph: Before dispatching the ants to construct solutions, a proper construction graph should be identified. The construction graph of ACO for composing a test is denoted as $G = (Q, \Psi)$, where the set of components Q corresponds to the set of questions in the question bank, the set of connections Ψ fully connects the set of questions. Fig. 1 shows a solution (one test) constructed by an ant. The questions are scattered for better illustration in the figure. The connected lines indicate an ant's searching route, whereas the connected questions form a test.

2) Candidate List: As the set of components Q is very large, candidate lists are used to restrict the number of available choices to be considered by ant a ($a = 1, 2, \ldots, m$). Before the ant a makes a construction step, a candidate list $\Omega^{(a)}$ is built by the following three selection processes.

First, select a concept. The probability for selecting a concept j for the lth test is

$$p_{\mathrm{c}}^{(a)}(j) = \frac{\eta_{\mathrm{c}}^{(a)}(j)}{\sum_{i=1}^{M} \eta_{\mathrm{c}}^{(a)}(i)} \tag{27}$$

where

$$\eta_{\mathrm{c}}^{(a)}(j) = \begin{cases} \max\left(\varepsilon, \tilde{c}_j - c_{lj}^{(a)}\right), & \text{if } \Omega_{\mathrm{c}}^{(a)}(j) \neq \emptyset, \quad n_l^{(a)} > 0 \\ \tilde{c}_j, & \text{if } \Omega_{\mathrm{c}}^{(a)}(j) \neq \emptyset, \quad n_l^{(a)} = 0 \\ 0, & \text{otherwise} \end{cases} \tag{28}$$

where $\Omega_{\mathrm{c}}^{(a)}(j)$ is the set of unselected questions by ant a that are related with concept j, $n_l^{(a)}$ is the total number of the selected questions by ant a for the lth test, and ε is a small positive value in case that the value of $\eta_{\mathrm{c}}^{(a)}(j)$ equals to or be smaller than zero. The bigger the value of $\tilde{c}_j - c_{lj}^{(a)}$, which means that the gap between the expected concept proportion and the actual concept proportion is bigger, the larger the probability for ant a to select concept j.

Second, select a question type. After choosing a concept j, the probability for selecting type k is

$$p_{\mathrm{ty}}^{(a)}(k) = \frac{\eta_{\mathrm{ty}}^{(a)}(k)}{\sum_{i=1}^{Y} \eta_{\mathrm{ty}}^{(a)}(i)} \tag{29}$$

where

$$\eta_{\mathrm{ty}}^{(a)}(j) = \begin{cases} \dfrac{\max\left(\varepsilon, \left|\tilde{s}_k - s_{lk}^{(a)}\right|\right)}{v_k}, & \text{if } \Omega_{\mathrm{ty}}^{(a)}(j,k) \neq \emptyset, \quad n_l^{(a)} > 0 \\ \dfrac{s_{lk}^{(a)}}{v_k}, & \text{if } \Omega_{\mathrm{ty}}^{(a)}(j,k) \neq \emptyset, \quad n_l^{(a)} = 0 \\ 0, & \text{otherwise} \end{cases} \tag{30}$$

where $\Omega_{\mathrm{ty}}^{(a)}(j,k)$ is the set of unselected questions by ant a that are related with concept j and have the type k.

Third, select a difficulty level. We consider the difficulty degree in 0.00–0.09 as difficulty level 0.0, 0.10–0.19 as level 0.1, and so on. The probability for selecting difficulty level u is

$$p_{\mathrm{diff}}^{(a)}(u) = \frac{\eta_{\mathrm{diff}}^{(a)}(u)}{\sum_{i=1}^{10} \eta_{\mathrm{diff}}^{(a)}(i)} \tag{31}$$

where

$$\eta_{\mathrm{diff}}^{(a)}(u) = \begin{cases} \dfrac{1}{\max\left(\varepsilon, \left|\tilde{D} - \left(\sum_{i=1}^{N} d_i x_{li}^{(a)} + u\right)/(n+1)\right|\right)}, & \text{if } \Omega_{\mathrm{diff}}^{(a)}(j,k,u) \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

where $\Omega_{\mathrm{diff}}^{(a)}(j,k,u)$ is the set of unselected questions by ant a that are related with concept j, type k, and difficulty level u.

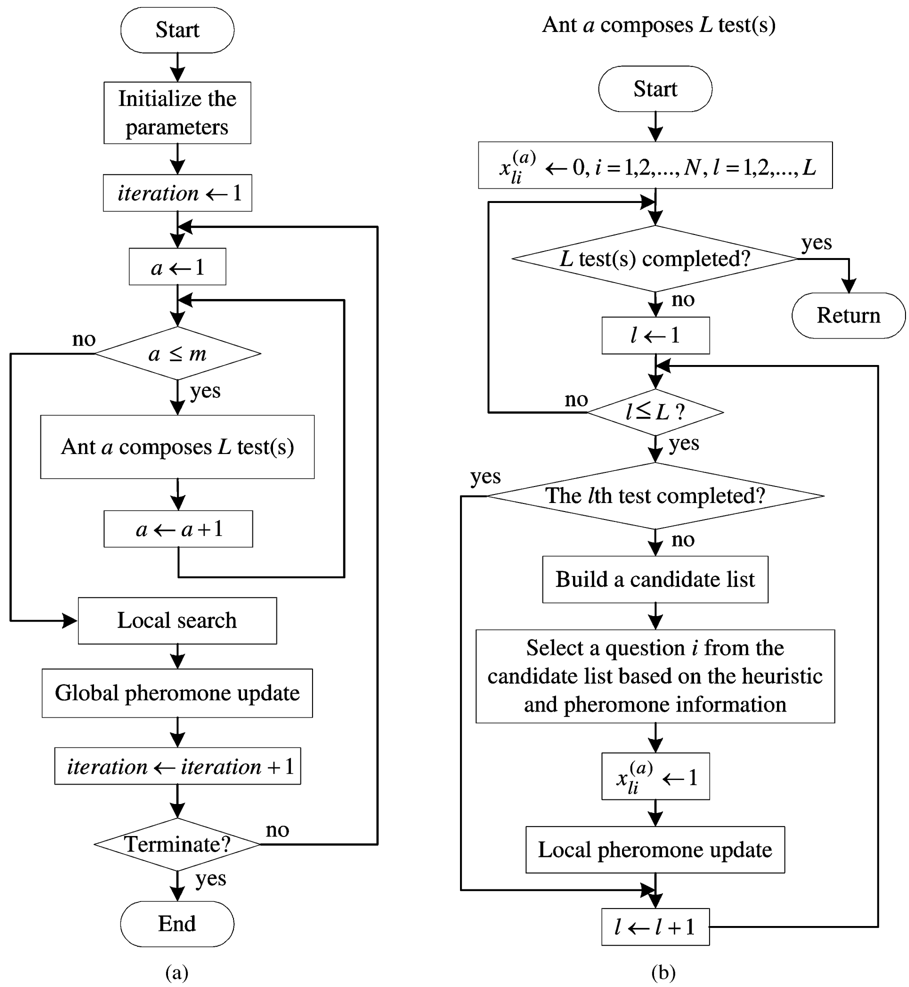


Fig. 2. Flowcharts of the proposed ACO-TC. (a) Overall flowchart of ACO-TC. (b) Flowchart of an ant's construction procedure for generating a solution.

After a concept j, a type k, and a difficulty level u have been selected, the unselected questions by ant a that have the attributes with the concept j, the type k, and the difficulty level u form a candidate list $\Omega^{(a)} = \Omega^{(a)}_{\text{diff}}(j, k, u)$.

3) Construction Step: During the solution construction process, each ant a ($a = 1, 2, \ldots, m$) selects a question from the candidate list according to the heuristic and pheromone information, which is termed one construction step. The heuristic information for selecting question i is defined as the discrimination degree of the question. The higher the discrimination degree, the larger the heuristic value is. The pheromone information is associated with each question, referring to the desirability of adding the question to the current partial solution. The probability for ant a to select a question i from the candidate list $\Omega^{(a)}$ is

$$p_{\text{item}}(i) = \frac{e_i \tau_{li}}{\sum_{j \in \Omega^{(a)}} e_j \tau_{lj}} \tag{33}$$

where τ_{li} is the pheromone value of question i in the lth test, $l = 1, 2, \ldots, L$. When a new question i is selected, ant a moves to the question and marks it selected as $x_{li}^{(a)} \leftarrow 1$.

4) Local Pheromone Update: Based on the ant colony system (ACS) framework [18], after an ant has chosen a question i, the pheromone value on the selected question will be decreased as

$$\tau_{li} \leftarrow (1 - \rho)\tau_{li} + \rho\tau_{\text{initial}} \tag{34}$$

where ρ ($0 < \rho < 1$) is the pheromone evaporation rate, τ_{initial} is the initial pheromone value.

5) Completion and Evaluation of a Solution: An ant a adds a new question to a test step by step until the total completion time has exceeded the expected total time. After L tests have been composed, the solution is evaluated by using the evaluation function (25). If the violation to the expected completion time is smaller by deleting the last added question, the last added question is deleted and the solution is evaluated once more. After m ants have completed building m solutions, the recorded best-so-far solution is updated and the algorithm proceeds to perform local search and the global pheromone update.

6) Local Search: A local search method is applied to the best-so-far solution. The aim of local search is to change a small part of questions in the test, and to see whether a better solution will be generated. Some randomly selected questions in the best-so-far solution are removed. Then new questions are added to the partial solution until the solution is completed. The

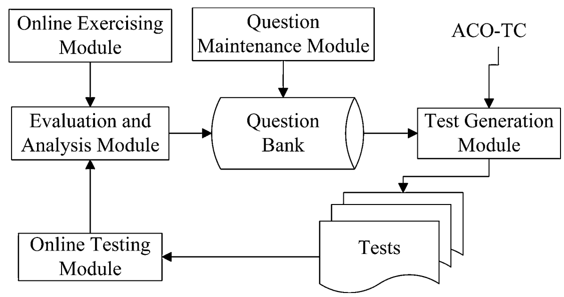


Fig. 3. Architecture of the proposed testing system.

new solution is evaluated, and the best-so-far solution is updated if a better solution is found.

7) Global Pheromone Update: Global pheromone update as (35) is performed in order to reinforce the pheromone values of questions in the best test generated so far. For all questions i in the best-so-far solution

$$\tau_{li} \leftarrow (1-\rho)\tau_{li} + \alpha \tag{35}$$

where α ($\alpha > \rho$) is the predefined pheromone reinforcement value, $l = 1, 2, \ldots, L$.

C. Implementation of ACO-TC

A complete flowchart of the proposed ACO-TC is presented in Fig. 2(a). Initially, the parameters of the algorithm are initialized. In every iteration, m ants are dispatched for constructing solutions. Each solution is composed of L tests, satisfying the preset requirements. At the end of each iteration, local search and global pheromone update are carried out. The algorithm terminates when the maximum iteration number or the maximum evaluation number is reached. The detailed illustration of an ant's construction procedure is shown in Fig. 2(b). The proposed ACO-TC is embedded in an intelligent testing system for generating tests.

IV. Intelligent Testing System

The proposed intelligent testing system is composed of a question bank, a question maintenance module, an evaluation and analysis module, an online exercising module, a test generation module, and an online testing module. Fig. 3 shows the architecture of the proposed system. There are three kinds of users: administrator, teacher, and student. Different users are given different rights. The users who can use the module are indicated in parentheses in the following description.

1) Question maintenance module (administrator): This module maintains questions in the question bank, such as question addition, deletion, and revision.
2) Question evaluation and analysis module (administrator, teacher): It evaluates students' answers to the questions; analyzes questions for the difficulty, discrimination, and completion time; adaptively adjusts the question attribute values.
3) Online exercising module (administrator, student): This module provides a platform for students to exercise; transfers the answer to the evaluation and analysis module for scoring.
4) Test generation module (administrator, teacher): It implements the proposed ACO-TC to generate tests.
5) Online testing module (administrator, teacher, student): It provides a platform for students to take a test; transfers the answers to the evaluation and analysis module for scoring.

The system utilizes the B/S network structure. Users can use a web browser and log into the system. Fig. 4 shows the interface for teachers to specify the requirements of the generated tests, including the course name, the test title, the expected total completion time, the expected average difficulty degree, the expected concept proportion, the expected score proportion of each type, and the number of tests to be generated simultaneously.

Fig. 5 shows the interface for students to answer a question in a test. The example shown in Fig. 5 is a multiple-choice question, which can also be a fill in the blank question, or a true or false question. After the answer is submitted, it will be evaluated and marked.

V. Performance Evaluation for the Test Composition

Whether a system is of high quality depends on its reliability of achieving good results. Therefore, this section analyzes the performance of the proposed system in two aspects. One is the system's performance to different question bank sizes. The other is the performance to different requirements for the test composition.

A. Settings of the Question Bank and Parameter Values

In order to facilitate the analysis of the proposed system, we generate experimental question banks with different features. After setting the number of questions n, the number of question types Y, and the number of concepts M, a question bank is generated by the following steps.

For $i := 1$ to n

Step 1) Select a uniform random integer number in $\{1, 2, \ldots, Y\}$ as the type y_i of question i.

Step 2) Select a uniform random float-point number in [0.00, 1.00) as the difficulty degree d_i of question i.

Step 3) Generate two different uniform random float-point numbers b_1 and b_2 in [0.00, 1.00), and set the discrimination degree e_i as $|b_1 - b_2|$.

Step 4) Generate a standard Gaussian random number b_3 and set the completion time t_i as $180b_3 + 180$. If the value of t_i exceeds the range of (10, 1000), the value of b_3 should be generated again to compute the value of t_i.

Step 5) Set the relation between concepts and questions as

Step 5-1) Set $r_{ji} := 0, j = 1, 2, \ldots, M$.

Step 5-2) Generate a random integer $b_4 \in \{1, 2, \ldots, M\}$.

Step 5-3) Set $r_{b_4,i} := 1$.

Step 5-4) Generate a random integer $b_5 \in \{1, 2, \ldots, M\}$.

Step 5-5) If $b_5 > b_4$ and $\delta_1 > 0.8$, the value of $r_{b_5,i}$ is set as 1, δ_1 is a uniform random number in [0, 1].

Step 5-6) Generate a random integer $b_6 \in \{1, 2, \ldots, M\}$.

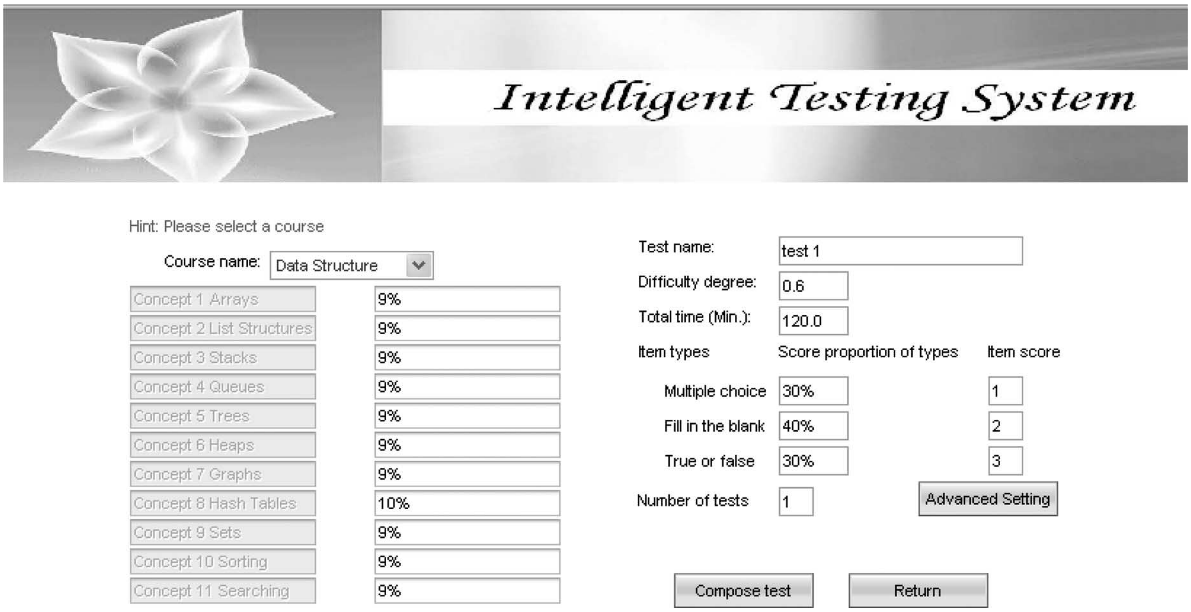


Fig. 4. Interface for teachers to specify the requirements of the generated tests.

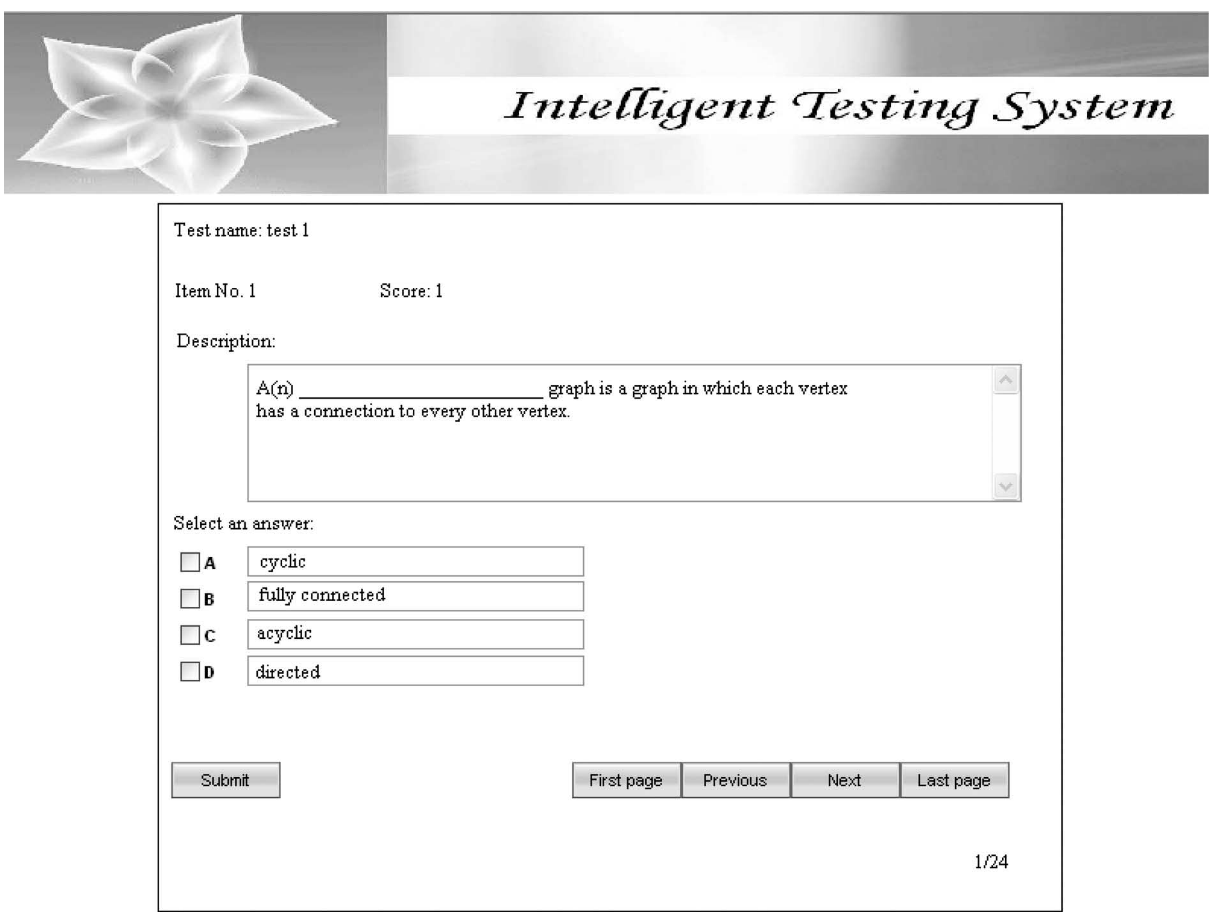


Fig. 5. Interface for students to answer a question in a test.

TABLE II
PARAMETER SETTINGS OF ACO-TC

m	5	ω_2	1.0
ρ	0.1	ω_3	1.0
α	10.0	ω_4	1.0
τ_{initial}	1.0	ε	0.001
ω_1	1.0	*MaxEvals*	10000

Step 5-7) If $b_6 > b_5$ and $\delta_2 > 0.8$, the value of $r_{b_6,i}$ is set as 1, δ_2 is a uniform random number in [0, 1].

End For

The number 180 in Step 4) represents 180 s, i.e., 3 min, so that the time distribution of questions approximates the Gaussian distribution $N(180, 180^2)$ in the range of (10, 1000). Each question in the bank is associated with at least one concept, but no more than three concepts.

The parameter settings of the proposed ACO-TC method are empirically set as in Table II, including the number of ants m, the pheromone evaporation rate ρ, the pheromone reinforcement value α, the initial pheromone value τ_{initial} for each question, the penalty weights $\omega_1, \omega_2, \omega_3, \omega_4$, a predefine positive value ε, and the maximum number of solution evaluations (*MaxEvals*). Since the question bank contains a large number of questions, the pheromone reinforcement to the questions in the best solution should be large enough. So the pheromone reinforcement value α is set as 10.0. The setting of the pheromone evaporation rate ρ is set as 0.1, following the recommended setting by Dorigo and Gambardella [18]. In each iteration, five ants are dispatched for constructing solutions. The algorithm terminates when the solution evaluation number is larger than *MaxEvals*.

B. Analysis on Using Question Banks With Different Sizes

Suppose a teacher wants to compose a test with the total completion time $\tilde{T} = 120 \times 60$ s, the average difficult degree $\tilde{D} = 0.6$, and the proportions of three concepts as 25%, 25%, and 50%, respectively. Only one type of questions is tested.

Fig. 6 shows the average discrimination degree from 20 independent runs calculated by ACO-TC, a random test composition method, and the genetic algorithm (GA) proposed by Hwang *et al.* [13], respectively. Question banks with sizes from 500 to 100 000 are checked. Note that the problem addressed in [13] does not consider the average difficulty and the types of questions. Therefore, the penalties for the violation of the expected difficulty degree and the types are not considered, which makes the problem much easier for GA than for ACO-TC and the random method. Using the same value of *MaxEvals*, it can be seen that ACO-TC always outperforms the GA [13] and the random method with much higher average discrimination degrees. On the other hand, the solution quality obtained by ACO-TC is always high even though the bank size changes, which indicates that the algorithm is robust and effective.

Table III also lists more information about the previous experiment. The average discrimination degree of the whole question bank (Bank Discri) is shown. The "mean" represents the mean evaluation function value and "disc" denotes the average discrimination degree of the resulting tests. The three methods can compose tests above the average discrimination degree of the question bank. The average discrimination degree of tests obtained by ACO-TC is in accordance with the mean evaluation function value, which means that the resulting test satisfies the requirements quite well. The "avgQ" shows the average question number in a composed test by running the algorithm 20 times independently. Although the numbers of questions in the tests composed by the random method and by the ACO-TC are similar, the quality of the tests by the random method is the lowest among the three algorithms.

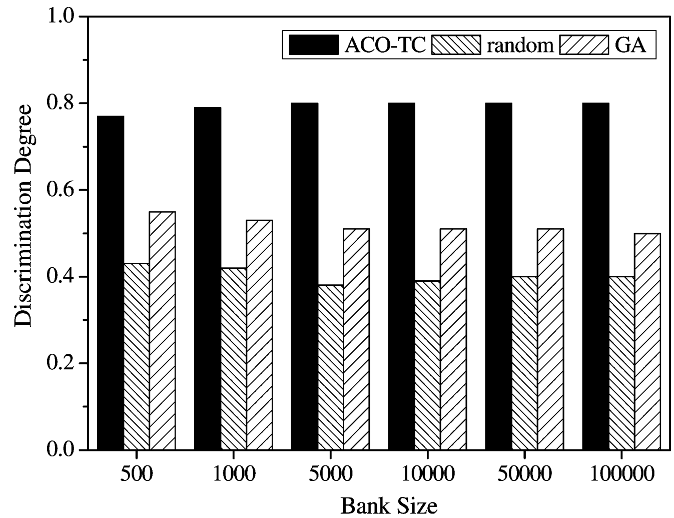


Fig. 6. Comparison between the average discrimination degree of ACO-TC, a random method, and the GA [13]. Question banks with different sizes are checked. The questions in the bank relate to three concepts and have only one type. Only one test is generated each time.

TABLE III
COMPARISON BETWEEN ACO-TC, A RANDOM METHOD, AND GA [13] WITH DIFFERENT QUESTION BANK SIZES FOR GENERATING ONE TEST

Bank Size	Bank Discri	ACO-TC			Random method			GA
		mean	disc	avgQ	mean	disc	avgQ	disc
500	0.36	0.77	0.77	27.6	0.43	0.47	28.3	0.55
1000	0.35	0.79	0.79	27.9	0.42	0.46	28.6	0.53
5000	0.33	0.80	0.80	29.7	0.38	0.42	29.9	0.51
10000	0.34	0.80	0.80	30.4	0.39	0.43	29.6	0.51
50000	0.33	0.80	0.80	29.4	0.40	0.44	29.3	0.51
100000	0.34	0.80	0.80	29.5	0.40	0.43	29.6	0.50

The questions in the bank relate to three concepts and have only one type. "Bank Discri" stands for the average discrimination degree of the whole question bank; "mean" stands for the mean evaluation function value; "disc" stands for the average discrimination degree of the resulting tests; "avgQ" stands for the average question number in a composed test. Each algorithm is run for 20 times independently.

Under the same requirements except for generating two tests simultaneously, Table IV tabulates the results with the same question banks. The GA [13] is not proposed for generating parallel tests, so it is not compared. The listed results in the table are the best, worst, and mean evaluation function values from 20 independent runs. The "%ok" in the table indicates the rates of generating acceptable solutions. The results obtained by ACO-TC are much better than those by the random method, especially for the acceptable rates. The acceptable rates of ACO-TC are all 100%, whereas the acceptable rates of the random method vary from 50% to 80%.

TABLE IV
EVALUATION FUNCTION VALUES OF ACO-TC AND THE RANDOM METHOD WITH DIFFERENT QUESTION BANK SIZES FOR GENERATING TWO TESTS SIMULTANEOUSLY

Bank Size	ACO-TC				Random method			
	best	worst	mean	%ok	best	worst	mean	%ok
500	0.68	0.60	0.63	100	0.33	0.18	0.24	80
1000	0.69	0.61	0.65	100	0.26	0.16	0.21	65
5000	0.68	0.62	0.65	100	0.31	0.08	0.18	50
10000	0.69	0.61	0.65	100	0.26	0.09	0.18	65
50000	0.67	0.58	0.63	100	0.27	0.14	0.20	75
100000	0.65	0.56	0.62	100	0.29	0.14	0.20	80

The questions in the bank relate to three concepts and have only one type. Here, "best" stands for the best evaluation function value; "worst" stands for the worst evaluation function value; "mean" stands for the mean evaluation function value; "%ok" indicates the rates of generating acceptable solutions in 20 independent runs.

TABLE V
EVALUATION FUNCTION VALUES OF ACO-TC AND THE RANDOM METHOD FOR GENERATING ONE TEST UNDER DIFFERENT REQUIREMENTS

Instance	Bank Size	ACO-TC				Random method			
		best	worst	mean	%ok	best	worst	mean	%ok
c3t1	1000	0.83	0.76	0.79	100	0.49	0.38	0.42	100
	10000	0.83	0.76	0.80	100	0.44	0.37	0.39	100
c3t3	1000	0.72	0.64	0.68	100	0.31	-0.75	0.07	25
	10000	0.76	0.68	0.72	100	0.31	-0.66	0.09	40
c12t4	1000	0.66	0.48	0.58	100	-4.41	-6.75	-5.77	0
	10000	0.71	0.58	0.65	100	-4.76	-6.75	-5.60	0

Here "best" stands for the best evaluation function value; "worst" stands for the worst evaluation function value; "mean" stands for the mean evaluation function value; "%ok" indicates the rates of generating acceptable solutions in 20 independent runs.

C. Analysis on Satisfying Different Requirements

Three instances of requirements for composing tests are experimented. Each instance is denoted as "cxty," which means that the test covers x concepts and the questions involve y types. Question banks with 1000 questions and 10 000 questions are used. The experimental results of ACO-TC and the random method for generating one test are compared in Table V. For the instance c3t1, ACO-TC can obtain a test with an average discrimination degree about 0.8, whereas the random method can generate a test with only an average discrimination degree about 0.4. When the numbers of selected concepts and types become larger, e.g., instances c3t3 and c12t4, the performance of the random method deteriorates rapidly. However, ACO-TC still achieves very good results.

VI. CONCLUSION AND FUTURE WORK

The approach of using computer-assisted testing systems to release teachers from the burden of composing tests and improve the assessment quality of tests is significant and promising in modern education. This paper builds an intelligent testing system, which embeds an ACO-TC method for generating high-quality tests for examinations. The ACO-TC method simulates the foraging behavior of natural ants for selecting appropriate questions to a test. The reliability of the test composition system has been experimented by applying the system in question banks with various sizes and by satisfying different requirements for the test composition.

The proposed system not only provides a way in composing high-quality tests, but also gives students a platform to exercise and evaluate themselves. The question attributes in a question bank are adaptively adjusted, always reflecting students' learning states. The function of generating parallel tests with identical test ability is also a novelty of the proposed system. It can be useful for makeup tests if someone cannot take the test in the scheduled time.

The prospect of computer-assisted technologies in education is bright. Much room is left to be explored. On the whole, the traditional education mode is in transition to be automated, intelligent, and personalized. Personalization is high demanding in the modern education system, which has not been considered in this paper. However, the combination of intelligence and personalization is the future direction, which will be addressed in the forthcoming work.

ACKNOWLEDGMENT

The authors would like to thank the associate editor and reviewers for their valuable comments and suggestions that improved the paper's quality. Jun Zhang is the corresponding author, e-mail: junzhang@ieee.org.

REFERENCES

[1] B. P. Butz, M. Duarte, and S. M. Miller, "An intelligent tutoring system for circuit analysis," *IEEE Trans. Educ.*, vol. 49, no. 2, pp. 216–223, May 2006.

[2] S. Piramuthu, "Knowledge-based web-enabled agents and intelligent tutoring systems," *IEEE Trans. Educ.*, vol. 48, no. 4, pp. 750–756, Nov. 2005.

[3] G.-J. Hwang, "On the development of a cooperative tutoring environment on computer networks," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 32, no. 3, pp. 272–278, Aug. 2002.

[4] G.-J. Hwang, "A tutoring strategy supporting system for distance learning on computer networks," *IEEE Trans. Educ.*, vol. 41, no. 4, pp. 343–343, Nov. 1998.

[5] V. B. Devedžić, "Key issues in next-generation web-based education," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 33, no. 3, pp. 339–349, Aug. 2003.

[6] O. G. Bellmunt, D. M. Miracle, S. G. Arellano, A. Sumper, and A. S. Andreu, "A distance PLC programming course employing a remote laboratory based on a flexible manufacturing cell," *IEEE Trans. Educ.*, vol. 49, no. 2, pp. 278–284, May 2006.

[7] M. M. Albu, K. E. Holbert, G. T. Heydt, S. D. Grigorescu, and V. Truşcă, "Embedding remote experimentation in power engineering education," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 139–143, Feb. 2004.

[8] I. T. Christou, S. Efremidis, T. Tiropanis, and A. Kalis, "Grid-based virtual laboratory experiments for a graduate course on sensor networks," *IEEE Trans. Educ.*, vol. 50, no. 1, pp. 17–26, Feb. 2007.

[9] C. Chou, "Constructing a computer-assisted testing and evaluation system on the world wide web—The CATES experience," *IEEE Trans. Educ.*, vol. 43, no. 3, pp. 266–272, Aug. 2000.

[10] E. Guzmán and R. Conejo, "Self-assessment in a feasible, adaptive web-based testing system," *IEEE Trans. Educ.*, vol. 48, no. 4, pp. 688–695, Nov. 2005.

[11] A. Kavčič, "Fuzzy user modeling for adaptation in educational hypermedia," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 34, no. 4, pp. 439–449, Nov. 2004.

[12] R.-G. Ho and Y.-C. Yen, "Design and evaluation of an XML-based platform-independent computerized adaptive testing system," *IEEE Trans. Educ.*, vol. 48, no. 2, pp. 230–237, May 2005.

[13] G.-J. Hwang, B. M. T. Lin, H.-H. Tseng, and T.-L. Lin, "On the development of a computer-assisted testing system with genetic test sheet-generating approach," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 4, pp. 590–594, Nov. 2005.

[14] D. Dagger, A. O'Connor, S. Lawless, E. Walsh, and V. P. Wade, "Service-oriented e-learning platforms: from monolithic systems to flexible services," *IEEE Internet Comput.*, vol. 11, no. 3, pp. 28–35, May/Jun. 2007.

[15] S.-C. Wang and Y.-H. Liu, “Software-reconfigurable e-learning platform for power electronics courses,” *IEEE Trans. Ind. Electron.*, vol. 55, no. 6, pp. 2416–2424, Jun. 2008.

[16] P. Lira, M. Bronfman, and J. Eyzaguirre, “Multitest II: a program for the generation, correction, and analysis of multiple choice tests,” *IEEE Trans. Educ.*, vol. 33, no. 4, pp. 320–325, Nov. 1990.

[17] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: optimization by a colony of cooperating agents,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996.

[18] M. Dorigo and L. M. Gambardella, “Ant colony system: A cooperative learning approach to the traveling salesman problem,” *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997.

[19] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA: MIT Press, 2004.

[20] S. S. Iyengar, H.-C. Wu, N. Balakrishnan, and S. Y. Chang, “Biologically inspired cooperative routing for wireless mobile sensor networks,” *IEEE Syst. J.*, vol. 1, no. 1, pp. 29–37, Sep. 2007.

[21] J. Zhang, H. S.-H. Chung, W.-L. Lo, and T. Huang, “Extended ant colony optimization algorithm for power electronic circuit design,” *IEEE Trans. Power Electron.*, vol. 24, no. 1, pp. 147–162, Jan. 2009.

[22] K. Socha, M. Sampels, and M. Manfrin, “Ant algorithms for the university course timetabling problem with regard to the state-of-the-art,” in *Proc. EvoWorkshops ‘03* (Lecture Notes in Computer Science 2611). Berlin, Germany: Springer-Verlag, 2003, pp. 334–345.

[23] W.-N. Chen and J. Zhang, “An ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements,” *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 39, no. 1, pp. 29–43, Jan. 2009.

[24] Y.-C. Liang and A. E. Smith, “An ant colony optimization algorithm for the redundancy allocation problem (RAP),” *IEEE Trans. Rel.*, vol. 53, no. 3, pp. 417–423, Sep. 2004.

[25] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas, “Data mining with an ant colony optimization algorithm,” *IEEE Trans. Evol. Comput.*, vol. 6, no. 4, pp. 321–332, Aug. 2002.

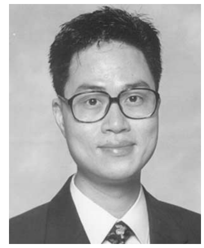
[26] T. L. Kelley, *Statistical Method*. New York: Macmillan, 1923.

Xiao-Min Hu (S’06) received the Bachelor’s degree in network engineering in 2006 from Sun Yat-Sen University, Guangzhou, China, where she is currently working toward the Ph.D. degree in the Department of Computer Science.

Her current research interests include artificial intelligence, evolutionary computation, swarm intelligence, and their applications in design and optimization on intelligent education, network routing, and biological information.

Jun Zhang (M’02–SM’08) received the Ph.D. degree in electrical engineering from the City University of Hong Kong (CityU), Kowloon, Hong Kong, in 2002.

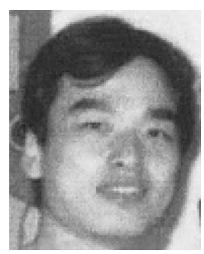
From 2003 to 2004, he was a Brain Korean 21 Postdoctoral Fellow in the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST). Since 2004, he has been with the Sun Yat-Sen University, Guangzhou, China, where he is currently a Professor of the Department of Computer Science. His current research interests include genetic algorithms, ant colony system, particle swarm optimization, fuzzy logic, neural network, nonlinear time series analysis and prediction, and design and optimization of power electronic circuits. He has authored four research book chapters and over 60 technical papers in his research areas.



Henry Shu-Hung Chung (S’92–M’95–SM’03) received the B.Eng. degree in electrical engineering in 1991 and the Ph.D. degree in 1994 from the Hong Kong Polytechnic University, Kowloon, Hong Kong.

Since 1995, he has been with the City University of Hong Kong (CityU), Kowloon. He is currently a Professor in the Department of Electronic Engineering and the Chief Technical Officer of e.Energy Technology Limited—an associated company of CityU. His current research interests include time- and frequency-domain analysis of power electronic circuits, switched-capacitor-based converters, random-switching techniques, control methods, digital audio amplifiers, soft-switching converters, and electronic ballast design. He has authored four research book chapters, and over 250 technical papers including 110 refereed journal papers in his research areas, and holds ten patents.

Dr. Chung was awarded the Grand Applied Research Excellence Award in 2001 from the CityU. He was the IEEE Student Branch Counselor and was a Track Chair of the technical committee on power electronics circuits and power systems of the IEEE Circuits and Systems Society during 1997–1998. He was an Associate Editor and a Guest Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, PART I: FUNDAMENTAL THEORY AND APPLICATIONS during 1999–2003. He is currently an Associate Editor of the IEEE TRANSACTIONS ON POWER ELECTRONICS and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, PART I: FUNDAMENTAL THEORY AND APPLICATIONS.



Ou Liu received the Ph.D. degree in information systems from the City University of Hong Kong (CityU), Kowloon, Hong Kong, in 2006.

Since 2006, he has been with the Hong Kong Polytechnic University, Hong Kong, where he is currently an Assistant Professor in the School of Accounting and Finance. His current research interests include business intelligence, decision support systems, ontology engineering, genetic algorithms, ant colony system, fuzzy logic, and neural network.

Jing Xiao received the B.S. and M.S degrees in computer science from Wuhan University, Wuhan, China, in 1997 and 2000, respectively, and the Ph.D. degree from the National University of Singapore, Singapore, Singapore, in 2005.

She is currently a Lecturer with the Department of Computer Science, Sun Yat-Sen University, Guangzhou, China. Her current research interests include evolutionary computation and text mining.