

Contents lists available at [ScienceDirect](http://www.sciencedirect.com)

# Computers & Education

journal homepage: [www.elsevier.com/locate/compedu](http://www.elsevier.com/locate/compedu)

## Applying an online game-based formative assessment in a flowchart-based intelligent tutoring system for improving problem-solving skills

Danial Hooshyar<sup>a,\*</sup>, Rodina Binti Ahmad<sup>a</sup>, Moslem Yousefi<sup>b</sup>, Moein Fathi<sup>a</sup>,  
Shi-Jinn Horng<sup>c</sup>, Heuseok Lim<sup>d,\*\*</sup>

<sup>a</sup> Department of Software Engineering, Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia

<sup>b</sup> Center of Systems and Machines Intelligence, College of Engineering, Universiti Tenaga Nasional, 43300 Kajang, Malaysia

<sup>c</sup> Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taiwan

<sup>d</sup> Department of Computer Science and Engineering, Korea University, South Korea

### ARTICLE INFO

#### Article history:

Received 16 March 2015

Received in revised form 8 October 2015

Accepted 12 October 2015

Available online 22 October 2015

#### Keywords:

Computer programming

Intelligent tutoring system

Game-based learning

Flowchart-based environments

### ABSTRACT

Intelligent Tutoring Systems (ITSs) have been producing consistent learning gains for decades. However, a common problem with these systems is maintaining student engagement without reducing the learning benefits. In order to balance the learning benefits of ITSs with the motivational benefits of games, an online formative assessment game called tic-tac-toe quiz for single-player (TRIS-Q-SP) is incorporated in a Flowchart-based Intelligent Tutoring System (FITS), benefiting from Bayesian networks for the process of decision making, for learning computer programming. This assessment game combines tic-tac-toe with online assessment, and revises the rule of tic-tac-toe for stimulating students to use online formative assessment actively. Finally, an empirical investigation carried out to evaluate the performance of FITS indicating that considerable gains for the experimental group over the control group. The proposed system extensively enhanced students' learning interest, attitude and degree of technology acceptance, as well as improved their achievements in problem solving activities. Additionally, the findings also reveal that providing Immediate Elaborated Feedback (IEF) for each answered question in TRIS-Q-SP is the optimal design, as it facilitated the enhancement of programming knowledge acquisition when comparing it with the No Immediate Elaborated Feedback (NIEF) condition.

© 2015 Elsevier Ltd. All rights reserved.

## 1. Introduction & related works

Many works have been conducted regarding the potential use of modern technologies in transforming education and training. However, only few of these statements have been supported and examined by academic research (Lowe & Schnotz, 2008; Mayer, 2009; O'Neil & Perez, 2003, 2006; Reiser & Dempsey, 2007; Rouet, Levonen, & Biarreau, 2001; Spector, Merrill, Van Merriënboer, & Driscoll, 2008; Zillig, Bodvarsson, & Bruning, 2005). Recently, issues which include practical constraints

\* Corresponding author.

\*\* Corresponding author.

E-mail addresses: [Danial.hooshyar@gmail.com](mailto:Danial.hooshyar@gmail.com) (D. Hooshyar), [Rodina@um.edu.my](mailto:Rodina@um.edu.my) (R.B. Ahmad), [moslem\\_yousefi@yahoo.com](mailto:moslem_yousefi@yahoo.com) (M. Yousefi), [moein.fathi@gmail.com](mailto:moein.fathi@gmail.com) (M. Fathi), [horngsj@yahoo.com.tw](mailto:horngsj@yahoo.com.tw) (S.-J. Horng), [limhseok@korea.ac.kr](mailto:limhseok@korea.ac.kr) (H. Lim).

such as time and tasks load, individual meet-up sessions, and assessments, have become a challenge for teachers in large-sized classes (Wang, 2007). A study by Bloom (1984) showed that almost 98% of learners who received one-to-one instructions are able to learn two standard deviations (SD) over those who were taught using conventional teaching method. A new generation of learning systems offering 'one-to-one' individualized instruction by stimulating activities of human teachers, similar to one teacher to one student, are named Intelligent Tutoring Systems (ITS). In computer programming, several efforts have been made to develop ITS for learning programming, which include Lisp-Tutor (Anderson & Reiser, 1985) that focuses on teaching through employing a rule-based approach, SQL-Tutor (Mitrovic, 2003) for SQL programming benefitting from a constraint-based modeling approach, JITS (Sykes & Franek, 2003) that concentrates on Java programming employing a decision tree approach, the PHP intelligent tutoring system (Weragama & Reye, 2013), and so on. However, research on the problem solving ability of these ITSs is still far from adequate and ITSs can cause learners to lose their learning motivation and enthusiasm when they do not receive enough timely guidance (Tsai, Tsai, & Lin, 2015). Therefore, to improve interaction, multimedia tools such as video, animation and audio or game-based teaching tools have been proposed in creating these systems and online courses (Aldrich, 2009). Moreover, in current online learning systems, online formative assessment functions are also necessary to be included for self-enhancement and self-evaluation (Vasilyeva, Pechenizkiy, & Paul De Bra, 2008). The motivation to use online assessments can be enhanced by game-based formative assessment, which contributes to learning effectiveness improvement, as verified by Wang (2008). Since basic computer programming skills cannot be acquired through passive learning and vary from theoretical knowledge learning, developing such interactive game-based ITS should be considered to transform traditional teaching methods. Through this process, learners will be encouraged to learn basic and imperative concepts of computer programming in a flexible way, improve their problem solving abilities, and are able to raise their motivation in applying online assessments and improving learning effectiveness. As a result, the present research aims to develop an online formative assessment game called Tic-tac-toe Quiz for Single-Player (TRIS-Q-SP), on top of an existing Flowchart-based Intelligent Tutoring System (FITS) (Hooshyar, Ahmad, Yousefi, Yusop, & Horng, 2015), in order to improve students' performance in learning computer programming and also improve their problem solving abilities. The decision making process in our system is managed by a Bayesian network, which is a machine learning classification method, to handle uncertainty based on probability theory (Wong & Butz, 2001). Students who use FITS are assisted in navigating online learning materials, similar to the work conducted by Liu, Andre, and Greenbowe (2008). Nonetheless, unlike Liu et al. (2008), FITS not only promotes the idea of navigating online learning materials and updating the Bayesian network by applying an online game-based formative assessment, it also offers an adaptive and personalized flowchart development environment with the aim of improving problem solving ability, in addition to suggesting learning goals along with the appropriate reading sequences to students. As the proposed system is able to visualize the solution development for a programming problem by converting the given problem statement to its relevant flowchart while engaging users in this process, it provides students with an accurate mental model of execution.

### 1.1. Game-based learning

It is intuitively clear that games are a potentially strong motivating factor for students (Gee, 2003; Steinkuehler, 2006). Since the development of internet and technology, several studies on digital game-based learning (DGBL) (Barab, Thomas, Dodge, Carteaux, & Tuzun, 2005; Tsai, Yu, & Hsiao, 2012) have been carried out benefiting from the strategy of multi-player online games (MOG) despite the fact that Tsai, Kinzer, Hung, Chen, and Hsu (2013) came to the conclusion that multi-player DGBL may generate over-competitiveness, making the users ignore the essence of learning especially in real-time-based MOG. Thus, the present research aims to develop a single-player formative assessment game, called Tic-tac-toe Quiz for single-player (TRIS-Q-SP), in a flowchart-based ITS targeting computer programming learning and problem solving skills improvement in novice programming. The mentioned game (tic-tac-toe) is typically multi or single-player and was established in Egypt (Zaslavsky, 1982); Due to the fact that the regulations and the game itself are not complicated, the tic-tac-toe game is being played frequently around the world (Crowley & Siegler, 1993). Basically, the rules of the tic-tac-toe quiz game (TRIS-Q) are applied in TRIS-Q-SP. Nonetheless, in TRIS-Q-SP, only one user can play against the computer. Since the tic-tac-toe based game has simple regulations and there is no sense of urgency when playing the game, it seems to be more applicable and suitable for learners using the flowchart-based ITS.

### 1.2. Formative feedback types

Previous studies have suggested that providing online formative assessments undoubtedly improves e-learning motivation and effectiveness (Gardner, Sheridan, & White, 2002; Henly, 2003). Based on timing, feedback is divided into immediate feedback, instant feedback (after completing the assessment) and delayed feedback (feedback messages after a few minutes or longer) (Shute, 2008). Furthermore, based on the level of information presented in feedback messages, the common formative feedback types are divided into elaborated feedback, knowledge of results and knowledge of correct responses (Shute, 2008).

In this study, immediate feedback and delayed feedback timing are simultaneously adopted as they both demonstrate unique advantages in designing game-based formative assessment. After answering each question, learners are provided with immediate feedback, or they could check their answer history with delayed feedback at any time after completing the assessment. Concerning the level of feedback messages, knowledge results and elaborated feedback selected for withholding

the correct answers facilitate learning. Elaborate feedback provides relevant information to guide learners toward the correct answers; however, the knowledge result only informs whether the answers are correct or incorrect. Therefore, since three types of feedback (delayed feedback, knowledge result and elaborated feedback) are unified into game-based formative assessment, this study adopts immediate knowledge of results (IKR), delayed knowledge of results (DKR), and immediate elaborate feedback (IEF) as the feedback mechanism for designing the game-based formative assessment. In the meantime, because the feedback mechanism of IEF provides excessive messages straightaway, the authors felt it necessary to examine whether the IEF messages negatively influence learning performance when the effects of using IEF in assessment games are unknown. Consequently, two different feedback types are developed, i.e. immediate elaborate feedback (IEF) and no immediate elaborate feedback (NIEF). With the assumption of whether immediate EF negatively affects game-based assessments is unknown, it is explored how distinct feedback types influence student knowledge acquisition.

## 2. Mental model and visualization

In preliminary programming courses, a learner's progress is highly determined and impacted by developing the performance of mental models and process flow (Ramalingham, LaBelle, Weidenbeck, 2004). The significance of such models in enhancing comprehension is further highlighted by Winslow (1996). Clear mental models of algorithms without the need of any prior training are proposed to learner programmers through flowcharts. Westphal, Harris, and Fadali (2003) observed that "Even with having pseudo code, it is so hard for novices to communicate the flow of a program, unless using flowcharts or diagrams". Since UML activity diagrams are not beneficial compared to flowcharts for non-complex programs, flowcharts are well-adapted for transferring the fundamental concepts to novice programmers. By employing visualization-based tools and environments, a flowchart's performance can be enhanced to help learners in problem solving and program development, as pointed out by Bassat Levy et al. (2001). Thus, a flowchart-based intelligent tutoring system using game-based formative assessment for interaction enhancement and Bayesian networks for the process of decision making provides a novice with an accurate mental model of execution.

## 3. Architecture

An effective way to improve problem-solving skills of novice programmers is to engage them in the solution designing activities from the very first stages of computer programming, therefore in this research an online game-based formative assessment is incorporated into a flowchart-based ITS with the purpose of boosting problem solving skills and learning imperative concepts of programming. This results into two advantages in comparison with other advanced ITSs for learning computer programming. Firstly, it focuses on the procedure of problem solving activities in the form of solution designing by extending the traditional model of ITS, applying a multi-agent system in the domain model of FITS for flowchart development in an adaptive manner (see 3.2). Secondly, it applies an online formative assessment game to enhance student's motivation and improve learning effectiveness. While the majority of existing ITSs support static HTML Web pages of learning materials, FITS offers users the ability to navigate through the online learning materials in a game form, generates appropriate reading sequences, and suggests learning goals. More importantly, since Markup languages are applied in the development of FITS, it offers students different options for flowchart development with various levels of guidance based on their level of knowledge using Bayesian networks and a multi-agent system which benefits from an automatic text-to-flowchart conversion approach. This feature, in addition to the online navigation of the learning materials in the form of a tic-tac-toe based game, considerably advances FITS in comparison to other related works in problem solving skill improvement of students. The FITS entry page and its navigational menu are shown in Fig. 1.

### 3.1. Bayesian Networks in FITS

How Bayesian network is applied in FITS as an inference engine in order to assist a student's learning based on his/her level of knowledge is discussed in this sub-section.

#### 3.1.1. Problem domain modeling

The Bayesian network is used to track users' knowledge concerning each specific concept within the problem domain as well as model the problem domain structure. It is also applied in FITS to provide students with an adaptive and personalized environment for flowchart development with the goal of improving their problem solving skills. In this regard, the sub-flowchart (an incomplete flowchart) generated by FITS for every student should correspond with their knowledge level. Thus, the three objectives are addressed with the assistance of a Bayesian network. With the aim of simplifying FITS development, the scope of the problems is restricted as follow: firstly, the system is developed to tutor students in the C++ programming language; secondly, only elementary topics are covered, such as variables, assignments, and control structures, while more sophisticated topics like pointers and inheritance are excluded. In case of more complicated concepts, the instructional contents and learning materials can easily be modified without having to reformulate the whole system. Thus, all programming concepts can simply be taught using the proposed system. A set of imperative and fundamental learning concepts of the C++ programming language is chosen from a relevant textbook. Then a diagram is created based on those concepts, where each node represents a concept, as shown in Fig. 2. An edge from each concept directed towards another

Example: [Write a program to generate fibonacci series.](#) | [more](#)

Navigation Menu

- overview\_of\_programming
- programming\_language
- variable
- floating\_point\_numbers
- incredecrementoperator
- relation\_operators
- assignment
- logical\_operators
- other\_operators
- output

## Welcome to FITS for C++ Programming

briefly description

*Flowchart-based Bayesian Intelligent Tutoring System for Problem Solving Ability Improvement in C++ Programming*

Please select a study goal you wish to learn from navigation menu  
or  
click on the proceed button to develop your flowchart

Estimation of the probability knowing this knowledge concept for the student  
50%

[Proceed](#)

Fig. 1. FITS's Entry page with a navigational menu, offering study goals.

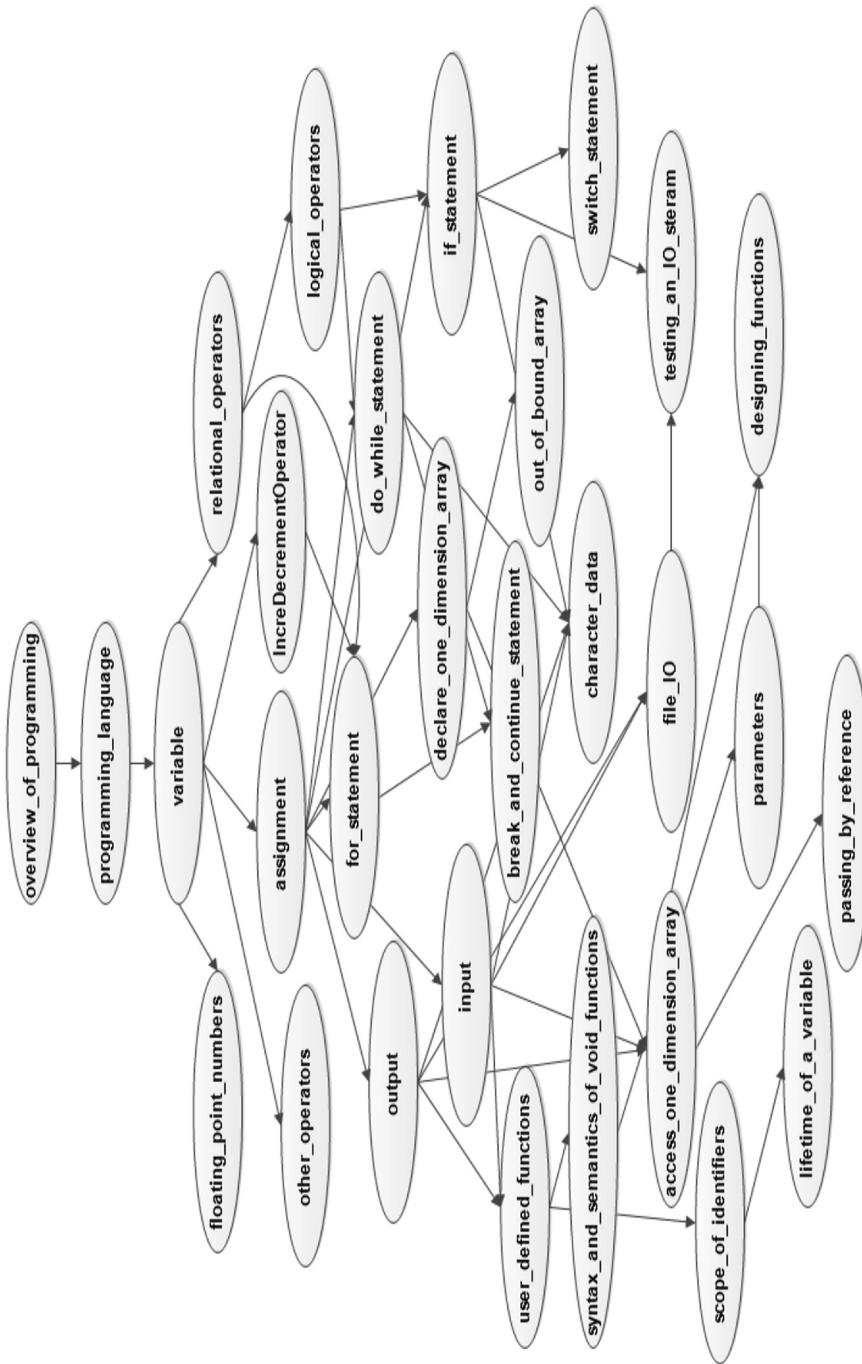


Fig. 2. All programming concepts in FITS.

displays learning dependencies among the knowledge concepts. Hence, by referring to the textbook, the directed acyclic graph (DAG) is made manually. The graph encodes the proper learning arrangements of all previously-mentioned concepts. Afterwards, the student's pre-test results are used to gain a conditional probability distribution (CPD) for the DAG. The concept being tested for each question is initially identified, and in case the question was answered correctly by the student, the concept is considered known, else unknown. Subsequently, a Bayesian network computes the probability that each notion is known  $P(v_i = \text{known})$  and then  $p(v_i = \text{known}, P_i = \text{known})$  can be measured. The Bayesian network measures a favorable CPD by employing the equation below.

$$\frac{p(v_i = \text{known} | P_i = \text{known})}{p(v_i = \text{known}, P_i = \text{known})} \cdot p(P_i = \text{known})$$

### 3.1.2. Personalized learning

To satisfy the aim of suggesting one-to-one environments in a structure, the interaction of the users with the system needs to be supervised in order to implement a profile for every one of them and to bring the Bayesian network up to date. In FITS, two different methods are employed in order to gain the profile from the users. The first approach is to answer the given questions before the entry page of FITS (pre-test). The second approach is to play TRIS-Q-SP after reading the relevant learning materials. In Fig. 3, the learning materials of the "Assignment" concept are exhibited as well as the button for the game. If the user wins the game, the Bayesian network is updated and the navigational menu is displayed one more time with the concept labeled as known (yellow light), see 3.4.1; but if the user does not win the game, the concept will be labeled as unknown (red or green light) and he/she will be advised to revise the learning materials related to the concept. Meanwhile, the Bayesian network will be updated. Hence, the Bayesian Network algorithm will constantly be updated and by referring to the degree of the knowledge of learners, the system recommends the relevant lecture notes or recommends sub-flowcharts and asks the students to complete them by providing the necessary guidance, workspace and editor, system-chat, instant feedback, and visualization notations.

## 3.2. Knowledge base

The knowledge base is classified into two distinct parts. The first section consists of the lecture notes in web page form, a repository of sample quizzes and questions along with their key solutions; while the second section includes a flowchart-based multi agent system. The lecture notes are presented when a user learns a new concept, whereas tic-tac-toe game and quizzes are introduced to decide if the user has understood a certain concept or not, see 3.4.1. These learning materials containing lecture notes and quizzes are stored by their specific programming concept which refers to the relevant nodes in the Bayesian network, see Fig. 2. Using this approach, knowledge concepts from actual instructional contents are separated and this separation is advantageous in many ways. Firstly, this separation allows multiple teachers to write parts of the instructional materials and work independently. Instructors can compose their course contents without having to be concerned about other contents; moreover, it allows them to include information resources located anywhere on the World Wide Web. Secondly, this separation enables us to apply Bayesian networks as the inference mechanisms that estimate the student's knowledge state. Thirdly, this separation facilitates changes to the content of the tutoring system. In case of adding learning materials, components in the Bayesian network should not be changed and only the learning materials need to be re-indexed accordingly. This development allows for easy modification of the instructional contents without having to reformulate the whole system.

Regarding the flowchart-based multi agent system, the system proposes two different flowchart development options to the users based on their level of knowledge, namely the Toolbar and the Guidance. In the toolbar option, users will be given an editor, a flowchart template, a workspace to complete the sub-flowchart, a sub-flowchart of the entered programming exercise, and a system chat. A screen shot of the toolbar option is shown in Fig. 4. After completing the flowchart by dragging and dropping shapes, the multi-agent system provides the users with brief feedbacks next to each of the shapes in the flowchart, as they are a classical way of making users think after a failure/error (Butler & Winne, 1995).

In the guidance option, the users are provided with a template of a flowchart having some shapes, or sub-flowcharts placed in the right positions; unlike the first option, the users are required to complete the flowchart using the editor. A screen shot of the guidance option is shown in Fig. 5. Users are not allowed to drag and drop a shape in the wrong position as the system will immediately identify any errors. Unlike the toolbar option, if the users dropped a shape in the right position at the flowchart template, the appropriate context will be extracted from the database and will be placed in the flowchart shape. Instant feedback for each drag and drop is provided by the system chat. Once the users are done dragging and dropping shapes, the system offers them a full flowchart of the programming exercise extracted from the database to allow a comparison to be made with the flowchart developed by the users.


It should also be noted with regards to the obtained profile and knowledge level evaluation by the Bayesian network for every programming concept, FITS is able to manage the guidance in order to offer users an intelligent, customized and active setting for problem solving improvement. Such feature facilitates the problem solving activities and makes them a more engaging and stimulating subject for the learners. For example, the guidance provided for user A for a programming problem

Example: [Write a program to generate fibonacci series.](#) | [more](#)

Navigation Menu

- assignment
- logical\_operators
- other\_operators
- output
- do\_while\_statement
- for\_statement
- declare\_one\_dimension\_arr
- if\_statement
- input
- user\_defined\_functions

## Welcome to FITS for C++ Programming




### Assignment

In the [C++ programming language](#), the assignment operator, '=', is the [operator](#) used for [assignment](#). Like most other operators in C++, it can be [overloaded](#).

The copy assignment operator, often just called the "assignment operator", is a special case of assignment operator where the source (right-hand side) and destination (left-hand side) are of the same class type. It is one of the [special member functions](#), which means that a default version of it is generated automatically by the compiler if the programmer does not declare one. The default version performs a memberwise copy, where each member is copied by its own copy assignment operator (which may also be programmer-declared or compiler-generated).

The copy assignment operator differs from the [copy constructor](#) in that it must clean up the data members of the assignment's target (and correctly handle self-assignment) whereas the copy constructor assigns values to uninitialized data members. For example:

```
My_Array first; // initialization by default constructor
My_Array second(first); // initialization by copy constructor
My_Array third = first; // Also initialization by copy constructor
second = third; // assignment by copy assignment operator
```



Estimation of the probability knowing this knowledge concept for the student

40%

Proceed

Fig. 3. Learning materials on 'assignment' concept and the game button.

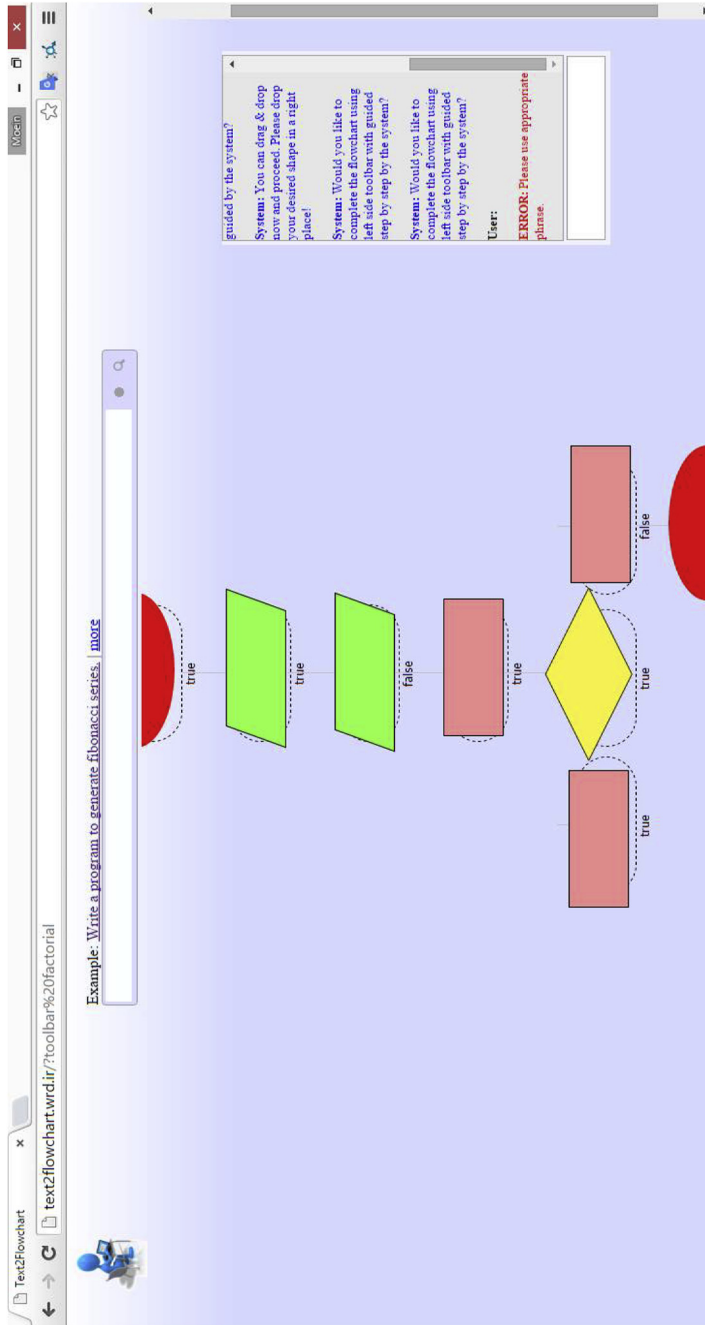


Fig. 4. The toolbar workspace with brief feedbacks.



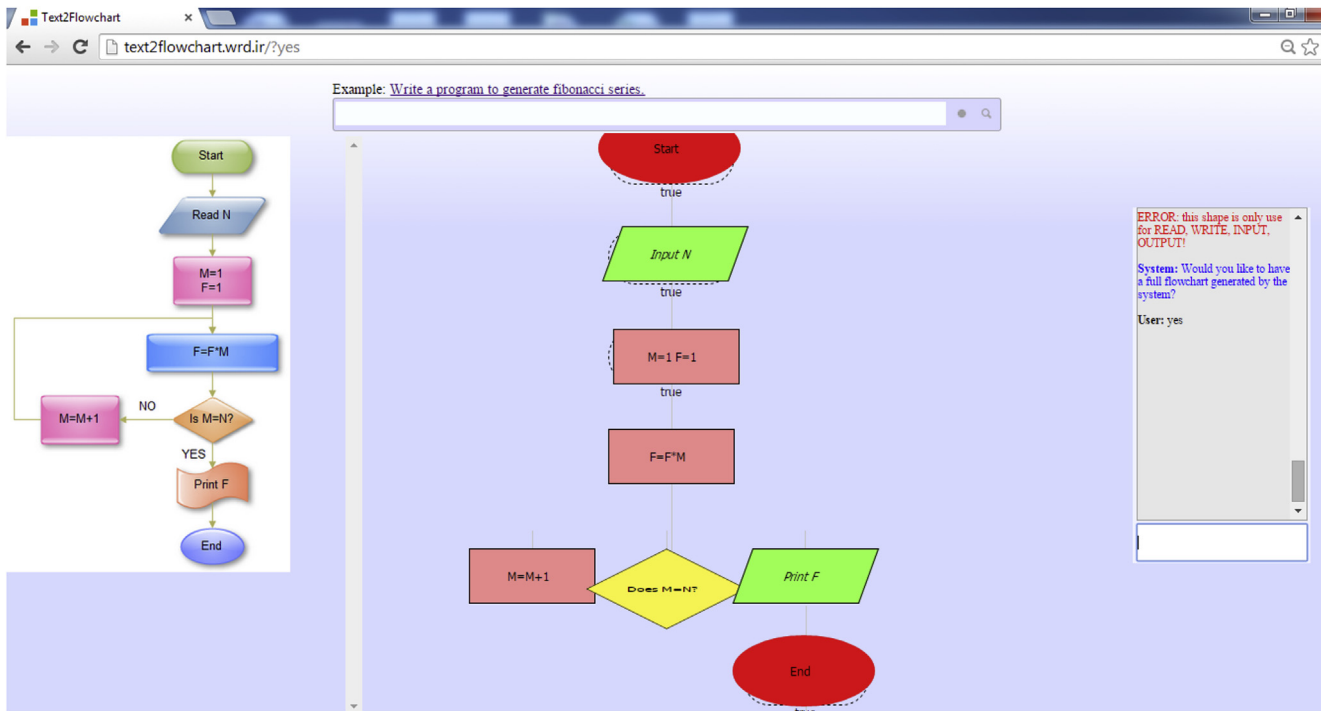


Fig. 5. Workspace provided by the guidance option along with a full flowchart from the database.

such as Factorial, is distinct from another user B; in Fig. 6, a sub-flowchart and guidance given to user A are displayed (the left figure), while in Fig. 6 (the right figure), a sub-flowchart for the same programming problem is shown indicating more guidance for user B based on his/her profile.

### 3.3. User interface module

By using the interface component, users can communicate with FITS. The interface component is split into two sub-categories which are output and input. The input section is from users to FITS and will update the Bayesian network based on the records gained from the users as explained in Section 3.1; whereas the output section is from FITS to the user which is shown in the web browser.

### 3.4. Adaptive guidance

FITS can provide users with designed options for supporting a specific user individually by using a Bayesian network and estimating the knowledge level of each user. Three types of adaptive guidance are offered to users by FITS: navigational support & tic-tac-toe game, pre-requisite recommendations, and flowchart development for problem solving skills, as discussed in 3.2.

#### 3.4.1. Navigational support & tic-tac-toe game

Each concept in the navigational menu is marked with a different color. Yellow represents 'already know the concept', red represents 'the concept is unknown and not ready to learn', and green represents 'unknown concept but ready to learn', showing the knowledge level of each user on that particular concept. When the learner clicks on the lights (Navigation menu), the corresponding learning materials will be provided for the learner as well as a tic-tac-toe quiz for single-player (TRIS-Q-SP) game button (Fig. 3). After perusing all learning materials related to the concept, the user can start the game. Once the user starts TRIS-Q-SP, he/she is able to click the distinct buttons, Fig. 7, in order to start a high level game, query the high score list, query the personal score, or query the answering history. The game-based formative assessment is shown as step 1 in Fig. 8. The basic regulations of TRIS-Q-SP are identical to the conventional tic-tac-toe game: whoever locates three adjacent tokens in a row, column or diagonally, wins the game. Regarding the TRIS-Q-SP procedure, once a user places the first token, he/she will receive a random multiple-choice test question chosen from the database according to the learning content (see step 2 in Fig. 8). The user should then click on the correct answer. However, in order to raise the playfulness and complexity of the game, the regulations are somewhat modified from the traditional tic-tac-toe. If the player gives the right answer, they can place their token (step 3–2 in Fig. 8); otherwise they will place the opponent's token (step 3–1, Fig. 8). Hence, the new game regulation prompts users to respond to the questions with more effort. Additionally, TRIS-Q-SP integrates three types of formative feedback. As seen in step 3 of Fig. 8, a window will pop up once the user has finished responding to the questions and begins to move in the TRIS-Q-SP. Immediate knowledge of results (IKR) feedback is indicated on the right of the screen, notifying the user whether they had correctly responded to the prior question. The previous answered question and Immediate Elaborated Feedback (IEF) comment appear on the left side of the screen, showing corresponding information and hints to the last question. Moreover, DKR will be provided to students when they use the query answering history function after completing a game. As shown in Fig. 8, the answering records show the DKR on the right side for all the answered questions, i.e. whether they are correct.

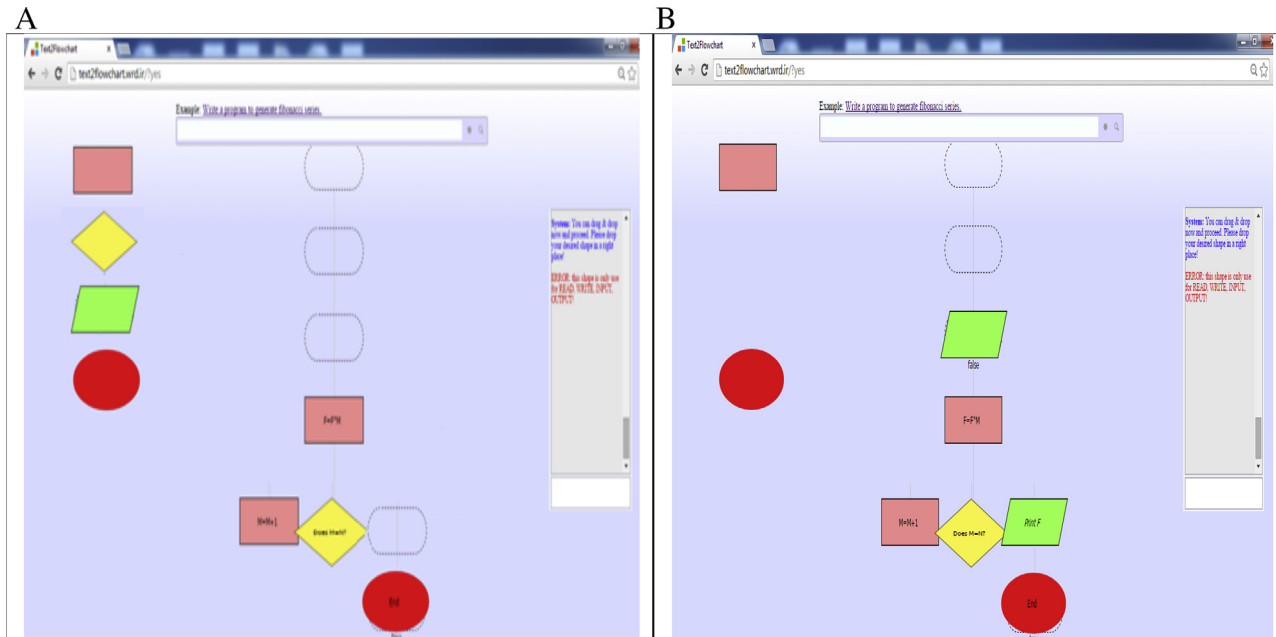
#### 3.4.2. Pre-requisite recommendations

In case the user clicks on the concepts with red, FITS recommends appropriate learning materials by considering the parent set of certain concepts in the Bayesian network. For instance, since "Assignment" and "IncreDecrementOperator" concepts are parent nodes of the "For statement" concept in the graph, FITS proposes both to the user before permitting them to proceed to the "For statement" concept (Fig. 9).

## 4. Evaluation

### 4.1. Participants & instruments

The experiment consisted of two classes of first-year undergraduate students. There were a total of 52 university students taking their first, introductory courses in Computer Science (CS). All students had no prior knowledge of computer programming. The classes were divided into one experimental group of 30 students using the proposed system and one control group of 22 students using traditional approaches. An instructor with over 6 years' experience of teaching the "Programming I" subject taught all 52 students. A test on the effect of feedback types on knowledge acquisition and learning achievement along with a questionnaire for assessing "learning interest," "technology acceptance" and "learning attitude" of the students comprised the research tools for this experiment. Two qualified teachers designed the test sheets. Students' prior knowledge of the early-stage concepts of computer programming, such as Variables, Assignments and Control Structures was assessed with a pre-test with a total score of 100, including 12 multiple-choice questions and 12 yes-or-no questions. The students' knowledge of computer programming was assessed with the post-test having a total score of 100, consisting of 6 short answer



**Fig. 6.** The guidance given to user A (the left Figure) and B (the right figure) for the same programming problem.

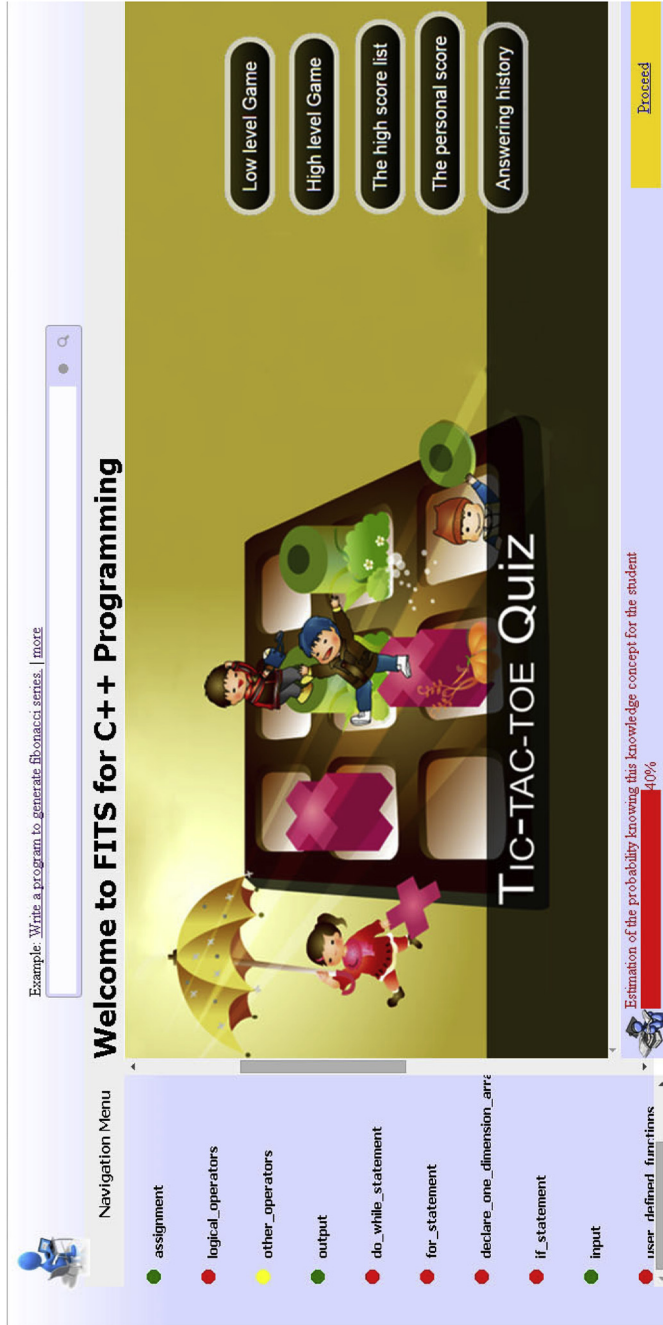



Fig. 7. The Game menu of TRIS-Q-SP.



Fig. 8. The three step of playing TRIS-Q-SP.


Example: [Write a program to generate fibonacci series.](#) | [more](#)

## Welcome to FITS for C++ Programming



To learn this concept, the system recommends you would better start according the following sequence.  
 Concepts which are parents in BN for the concept you choose.

<p>☹️ Not known: You should learn firstly!</p>	<p><b>assignment</b> <b>inrecrementoperator</b></p>
<p>😊 Already known: Don't need to study again, Good!</p>	<p><b>relation_operators</b></p>

Estimation of the probability knowing this knowledge concept for the student  
 11%

[Proceed](#)

Fig. 9. A learning sequence generated by FITS for the “for statement” concept, since it is not ready to learn (Red light).

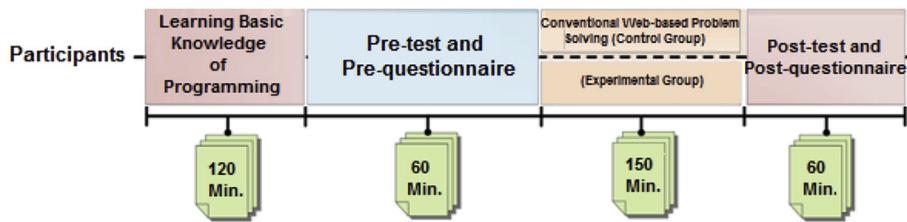


Fig. 10. Diagram of experiment design.

questions, 12 multiple-choice questions, and 4 matching items. The Cronbach's alpha values of 0.972 and 0.981 were obtained for the pre-test and post-test respectively. For the 8-item learning attitude questionnaire with a 5-point rating scheme, 0.84 was achieved for Cronbach's alpha. The learning interest questionnaire having 14 items and a rating scheme of 6 points produced 0.89 as the Cronbach's value. The 14-item questionnaire of technology acceptance includes 2 dimensions with a rating scheme of 5 points. Seven items were dedicated to "usefulness" and another 7 to "ease of use." The Cronbach's alpha values of 0.88, 0.97 and 0.91 were attained for the questionnaire, and its dimensions of "usefulness" and "ease of use" respectively.

#### 4.2. Experimental procedure

The procedure of the experimental study is illustrated in Fig. 10. It includes a 2-h course on the basics of computer programming concepts (Variables, Assignments and Control Structure), which is part of the existing Programming I course in CS. The learning activities began with a 1-h pre-test and completing the questionnaire on learning attitude. Afterwards, the experimental group took part in the web-based learning activities with the proposed system, whereas the control group was exposed to the conventional problem solving method. Each group had 150 min. At the end, a 1-h post-test was performed to assess and monitor any changes in learning attitudes, learning achievements, technology acceptance, knowledge acquisition with different feedback types and learning reception.

### 5. Experimental results

#### 5.1. Learning achievement analysis

As one of the primary objectives of this study, the efficiency of the proposed system was investigated considering its role in improving students' learning achievement. To circumvent the inevitable prior knowledge difference between the two groups, the post-test and pre-test scores were regarded as dependent and covariate variables after implementing one-way analysis of covariance (ANCOVA). Having applied a homogeneity test, the post-test scores of the control and experimental groups were found to be homogeneous ( $F = 0.30$ ,  $p = 0.61 > 0.05$ ), therefore justifying the use of ANCOVA. The tuned mean values of the post-test scores for the experimental and control groups were 81.32 and 59.17 respectively. The ANCOVA results are presented in Table 1. Furthermore, the control and experimental groups ( $F = 56.04$  and  $p < 0.05$ ) exhibited considerable difference, indicating that the proposed system had positive and drastic impact on the experimental group's learning achievements in the programming I course.

#### 5.2. Learning attitude analysis

In order to define the effectiveness of the proposed system, its efficacy in improving students' learning attitudes toward computer programming after doing problem solving tasks was examined as another objective of this study. Based on the experimental results, in terms of the learning attitude questionnaire, the mean value and standard deviation of 3.39 and 0.29 were obtained for the control group before the learning activities, while 3.55 and 0.30 were obtained for the experimental group. The t-test result ( $t = 1.14$ ,  $p > 0.05$ ) indicates no considerable difference between the control and experimental groups before taking part in the problem solving tasks. However, as illustrated in Table 2, the result obtained from the t-test following the learning activity ( $t = 3.57$ ,  $p < 0.05$ ) indicates that students' learning attitudes in the experimental group considerably enhanced compared to the control group. As a result, the proposed system enhanced students' attitudes toward the computer programming course along with their learning achievements.

Table 1

ANCOVA of the post-test results and descriptive data; Experimental Group (EG); Control Group (CG).

Group	N	Mean	S.D.	Adjusted mean	Std. error	F
EG	30	81.10	6.90	81.32	1.65	56.04*
CG	22	59.46	11.22	59.17	2.24	

\* $p < 0.05$ .

**Table 2**  
Learning attitude questionnaire t-test result.

	Group	N	Mean	S.D.	t
Post-test	EG	30	4.39	0.51	3.57*
	CG	22	3.16	1.29	

\*p < 0.05.

**Table 3**  
Learning interest questionnaire t-test result.

Group	N	Mean	S.D.	t
EG	30	4.07	0.43	3.60*
CG	22	3.42	1.19	

\*p < 0.05.

### 5.3. Technology acceptance and learning interest analysis

Regarding learning interest, mean values of 3.42 and 4.07 and standard deviation of 1.19 and 0.43 were obtained for the control and experimental groups respectively. The results obtained from the t-test ( $t = 3.60$ ,  $p < 0.05$ ) presented in Table 3 indicate a noteworthy difference between the control and experimental groups. Essentially, higher learning interest was revealed for students in the experimental group who utilized the proposed system in the learning process. This finding is in agreement with that for learning attitudes, which indicates that by using the online proposed system students are situated in a more enjoyable learning context than a conventional approach. In terms of students' technology acceptance through using the proposed system, the t-test result given in Table 4 emphasizes the considerable difference between the control and experimental groups in both "ease of use" ( $t = 2.91$ ,  $p < 0.05$ ) and "usefulness" ( $t = 4.16$ ,  $p < 0.05$ ) dimensions after the learning activity; this represents the students' acceptance of the system and approach. Moreover, the interviews conveyed similar findings, whereby over 85% of students in the experimental group sensed the helpfulness of the proposed system in improving their learning effectiveness.

### 5.4. Analysis of the effect of different feedback types on knowledge acquisition

Considering the influence of various feedback types on acquiring knowledge, the effectiveness of feedback types, IEF and not IEF, are explored on enhancing the knowledge acquisition of learners through the process of learning. ANCOVA was utilized to investigate the discrepancies between the post-test scores of the computer programming knowledge of both teams with the subjects' pre-test scores as covariate, and the objective of comparing the effects of two formative feedback situation in TRIS-Q-SP. Based on the uniformity of the regression test, the two teams' regression uniformity was the same ( $F = 0.30$ ,  $p = 0.61 > 0.05$ ) implying that the coefficient is homogeneous and the initial hypothesis of ANCOVA is conformed. Table 5 presents a summary of the ANCOVA findings, through which the post-test scores of adjusted mean values were 59.40 for the control team and 78.15 for the empirical team. Furthermore, a remarkable discrepancy was observed between the control and experimental groups with  $F = 58.12$  and  $p < 0.05$ . This suggests that providing IEF messages throughout game-based evaluation compared with not providing any, appeared more helpful in enhancing the performance of learners' knowledge acquisition through the process of learning.

## 6. Discussion and conclusions

Simulating human teachers in implementing one-to-one personalized teaching to a certain extent, is an intense and difficult topic in research on learning environment design. Extending the traditional architecture of ITS, exploring new methods of modeling a student's learning process and performance, and interaction enhancement are three key issues in launching e-learning. Finding the solution to these three issues will contribute to the induction of e-Learning. Therefore, in this study an attempt was made to develop an online formative assessment game called tic-tac-toe quiz for single-player (TRIS-Q-SP) on top of an existing Flowchart-based Intelligent Tutoring System (FITS) for exploring the game's effectiveness in learning computer programming as well as problem-solving skill improvement. The decision-making process in the

**Table 4**  
Technology acceptance t-test result for the EG and CG.

	Group	N	Mean	S.D.	t
Easy to use	EG	30	5.52	0.54	2.91*
	CG	22	4.49	1.62	
Usefulness	EG	30	5.14	0.52	4.16*
	CG	22	4.32	1.70	

\*p < 0.05.



**Table 5**  
Descriptive data and ANCOVA of the post-test results.

Group	N	Mean	S.D.	Adjusted mean	Std. error	F
EG	30	78.10	6.90	78.15	1.50	58.12*
CG	22	59.56	10.30	59.40	2.21	

\* $p < 0.05$ .

proposed system operates on a Bayesian network. This proposed system facilitates the creation of a mental model of execution for students, as it is able to visualize the solution development for a programming problem by converting the given problem statement to a relevant flowchart while engaging users in the process. To the best of our knowledge, the proposed system is the first flowchart-based ITS using an online formative assessment game to enhance students' motivation in learning the basic concepts of computer programming. FITS enables students to improve problem-solving skills through solution designing activities by offering various personalized options in flowchart development along with step-by-step guidance. Due to the fact that FITS takes full advantage of Bayesian networks along with markup language, it can manipulate its guidance and change the algorithms and sub-flowchart components for each user, so as to provide an intelligent, personalized and dynamic environment for problem-solving improvement. According to the empirical findings, FITS has positive and drastic impact on the experimental group's learning achievements over the control group. As some of the tasks in the pre and posttest consisted of flowchart development for the given programming problem, the positive impact of FITS on students' learning achievement shows its capability of yielding improvements in problem-solving skills besides learning programming concept. FITS is also able to enhance students' attitudes toward the computer programming course along with their learning achievements. Concerning learning interest, students in the experimental group who utilized FITS in the learning process experienced higher learning interest. This finding is in agreement with that for learning attitude, which indicates that by using the online flowchart-based ITS students are set in a more enjoyable learning context than with conventional approaches. In addition, the results from technology acceptance and learning interest analysis showed that over 85% of students in the experimental group sensed the helpfulness of the proposed system in improving their learning effectiveness. As a result, the students were remarkably involved in the problem-solving tasks while their intrinsic motivation also enhanced through using the online formative assessment game (TRIS-Q-SP). The findings additionally revealed that not only were the students in a learning state with full enjoyment and engagement by using FITS, but their learning attitudes and achievement also improved. Furthermore, the findings revealed that if the system provides immediate elaborated feedback during the TRIS-Q-SP game, it significantly increases the level of student performance in computer programming.

## Acknowledgments

This research has been financially funded by the University of Malaya with project number of RG327-15AFR.

## References

- Aldrich, C. (2009). *Learning online with games, simulations, and virtual worlds*. San Francisco, CA: Jossey-Bass.
- Anderson, J. R., & Reiser, B. J. (1985). *The LISP tutor, byte* (Vol. 10(4), pp. 159–175).
- Barab, S., Thomas, M., Dodge, T., Carteaux, R., & Tuzun, H. (2005). Making learning fun: quest Atlantis, a game without guns. *Educational Technology Research and Development*, 53(1), 86–107.
- Ben-Bassat Levy, R., Ben Ari, M., & Uronen, P. (2001). An extended experiment with Jeliot 2000. In *Proceedings of the first International program visualization workshop* (pp. 131–140). Porvoo Finland: University of Joensuu Press.
- Bloom, B. S. (1984). The search for methods of group instruction as effective as one-to-one tutoring. *Educational Leadership*, 41(8), 4–17.
- Butler, D. L., & Winne, P. H. (1995). Feedback and self-regulated learning: a theoretical synthesis. *Review of Educational Research*, 65(3), 245–281.
- Crowley, K., & Siegler, R. S. (1993). Flexible strategy use in young children's tic-tac-toe. *Cognitive Science*, 17, 531–561.
- Gardner, L., Sheridan, D., & White, D. (2002). A web-based learning and assessment system to support flexible education. *Journal of Computer Assisted Learning*, 18, 125–136.
- Gee, J. P. (2003). *What video games have to teach us about learning and literacy*. New York, NY: Palgrave Macmillan. <http://dx.doi.org/10.1145/950566.950595>.
- Henly, D. C. (2003). Use of web-based formative assessment to support student learning in a metabolism/nutrition unit. *European Journal of Dental Education*, 7(3), 116–122.
- Hooshyar, D., Ahmad, R. B., Yousefi, M., Yusop, F. D., & Horng, S.-J. (2015). A flowchart-based intelligent tutoring system for improving problem-solving skills of novice programmers. *Journal of Computer Assisted Learning*, 31, 345–361.
- Liu, C. H., Andre, T., & Greenbowe, T. (2008). The impact of Learner's prior knowledge on their use of chemistry computer simulations: a case study. *Journal of Science Education and Technology*, 17(5), 466–482.
- Lowe, R., & Schnotz, W. (2008). *Learning with animation: Research implications for design*. New York: Cambridge University Press.
- Mayer, R. E. (2009). *Multimedia learning* (2nd ed.). New York: Cambridge University Press.
- Mitrovic, A. (2003). An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education*, 13(2–4), 197–243.
- O'Neil, H. F., & Perez, R. S. (2003). *Technology applications in education: A learning view*. Mahwah, NJ.: Erlbaum.
- O'Neil, H. F., & Perez, R. S. (2006). *Web-based learning: Theory, research, and practice*. Mahwah, NJ.: Erlbaum.
- Ramalingam, V., LaBelle, D., & Weidenbeck, S. (2004). *Self-efficacy and mental models in learning to program*. *SIGCSE bulletin* (Vol. 36(3), pp. 171–175). New York – NY –US, pp: ACM Press.
- Reiser, R. A., & Dempsey, J. V. (2007). *Trends and issues in instructional design and technology*. Upper Saddle River, NJ: Pearson Merrill Prentice Hall.
- Rouet, J. F., Levonen, J. J., & Biardeau, A. (2001). *Multimedia learning: Cognitive and instructional issues*. Oxford, UK: Pergamon.
- Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research*, 78(1), 153–189.
- Spector, J. M., Merrill, M. D., Van Merriënboer, J., & Driscoll, M. P. (2008). *Handbook of research on educational communications and technology*. New York: Erlbaum.
- Steinkuehler, C. A. (2006). Why game (culture) studies now? *Games and Culture*, 1, 97–102. <http://dx.doi.org/10.1177/1555412005281911>.

- Sykes, E. R., & Franek, F. (2003). A prototype for an intelligent tutoring system for students learning to program in java. In *Proceedings of the 3rd IEEE International conference on advanced learning technologies* (pp. 485–486). Athens, Greece.
- Tsai, F. H., Kinzer, C., Hung, K. H., Chen, C. C., & Hsu, I. Y. (2013). The importance and use of targeted content knowledge with scaffolding aid in educational simulation games. *Interactive Learning Environments*, 21(2), 116–128.
- Tsai, Fu-H., Tsai, C.-C., & Lin, K.-Y. (2015). The evaluation of different gaming modes and feedback types on game-based formative assessment in an online learning environment. *Computers & Education*, 81, 259–269. <http://dx.doi.org/10.1016/j.compedu.2014.10.013>.
- Tsai, F. H., Yu, K. C., & Hsiao, H. S. (2012). Exploring the factors influencing learning effectiveness in digital game-based learning. *Educational Technology & Society*, 15(3), 240–250.
- Vasilyeva, E., Pechenizkiy, M., & De Bra, P. (2008, January). *Adaptation of elaborated feedback in e-learning*. In *adaptive hypermedia and adaptive web-based systems* (pp. 235–244). Berlin Heidelberg: Springer.
- Wang, T. H. (2007). What strategies are effective for formative assessment in an e-learning environment? *Journal of Computer Assisted Learning*, 23, 171–186.
- Wang, T. H. (2008). Web-based quiz-game-like formative assessment: development and evaluation. *Computers & Education*, 51(3), 1247–1263.
- Weragama, D., & Reye, J. (2013). The PHP intelligent tutoring system. *Artificial Intelligence in Education Lecture Notes in Computer Science*, 7, 583–586.
- Westphal, B., Harris, F., & Fadali, M. (2003). Graphical programming: a vehicle for teaching computer problem solving. In *33rd ASEE/IEEE frontiers in education conference* (pp. 19–23). Boulder Colorado: IEEE.
- Winslow, L. E. (1996). Programming pedagogy-A psychological overview. *SIGCSE Bulletin*, 28, 17–22.
- Wong, S. K. M., & Butz, C. J. (2001). Constructing the dependency structure of a multi-agent probabilistic network. *IEEE Transactions on Knowledge and Data Engineering*, 13(3), 395–415.
- Zaslavsky, C. (1982). *Tic-tat-toe and other three in a row games from ancient Egypt to the modern computer*. New York: Thomas Y. Crowell.
- Zillig, P. L. M., Bodvarsson, M., & Bruning, R. (2005). *Technology-based education*. Greenwich, CT: Information Age Publishing.



**Danial Hooshyar** is a Ph.D. student and research assistant in the Software Engineering Departments at the University of Malaya, where he conducts research on novice programmers and flowchart-based environments for improving problem solving skills. He has already published several articles on aforementioned area based on the conducted experimental studies.



**Associate Prof. Dr. Rodina Binti Ahmad:** She has received a Petronas scholarship in 1984 to pursue her bachelor degree in United States of America. She pursued her master degree in 1989 under Petronas scholarship as well in the Computer Science discipline. She started teaching in University Technology Malaysia in computer science department in 1991. She moved to University Malaya to continue her teaching career in 1993. She received her PhD in the area of Information Systems Management from National University of Malaysia in 2006. She has been involved in various research activities in the area of Requirements Engineering and Organizational analysis and modeling.



**Moslem Yousefi** is currently working as a senior lecturer in the department of mechanical engineering, Universiti Tenaga Nasional (UNITEN), Kajang, Malaysia. He is a member of the science and engineering institute (SCIEI) and etc. Dr. Moslem did his PhD at Universiti Teknologi Malaysia (UTM) (2010–2013) where he worked on developing a new evolutionary-based design approach for constrained optimization of thermal systems. He completed his Masters and degree both in fluid mechanics from Tarbiat Modares University, Iran (2008) and Iran University of Science and Technology, Iran (2005) respectively. Dr. Yousefi has been publishing numerous research papers in the field of evolutionary computation, multi-objective optimization, and constrained optimization in reputed journals.



**Moein Fathi** is currently a master student in the field of software engineering in Malaysia. He is a professional programmer and a young researcher who has collaborated in many research projects, i.e. flowchart-based programming environments for improving problem solving skills.



**Shi-JINN HORNG** received the B.S. degree in electronics engineering from the National Taiwan Institute of Technology, the MS degree in information engineering from the National Central University, and the Ph.D. degree in computer science from the National Tsing Hua University in 1980, 1984, and 1989, respectively. He was a professor and dean of the College of Electrical Engineering and Computer Science, National United University, Miaoli, Taiwan from 2006 to 2009. Currently, he is the Chair and a distinguished professor in the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology. He has published more than 190 research papers and received many awards; especially, the Distinguished Research Award between 2004 and 2006 from the National Science Council in Taiwan; Outstanding IT Elite Award, in 2005; Outstanding EE Prof. Award, the Chinese Institute of Electrical Engineering; and the Outstanding Research and Invention Award between 2006 and 2008 from National Taiwan University of Science and Technology. He was also promoted to the Chair professor in National United University in 2008. His research interests include Information Security, VLSI design, multiprocessing systems, and parallel algorithms.



**Heuseok Lim** received his B.S., M.S. and Ph.D. degrees in Computer Science from Korea University, Seoul, in 1992, 1994 and 1997, respectively. He worked at SITEC of Samsung for two years after his PhD. He is a Professor in the Department of Computer Science and Engineering at Korea University in Korea. In 2012 year, He was a visiting scholar in the department of Computer Science of University of Colorado at Boulder. Now, he has been a member of ACL and many other research groups. His research interests are in brain-neuro language processing, natural language processing, AI in education, and educational data mining.