

Chapter 2

Designing Intelligent Agent in Multilevel Game-Based Modules for E-Learning Computer Science Course

Kristijan Kuk, Ivan Milentijević, Dejan Rančić and Petar Spalević

Abstract Nowadays, game-based learning environments are very common environments for studying major scientific fields such as mathematics, computer science, electronics and electrical engineering. This chapter presents a game-based modules system called the game-based modules (GBMs). It combines the characteristics of computer game elements with the existing interactive multimedia environments for learning mathematics, physics and electronics. This module presents a new type of game-learning environment for teaching units of Computer Science courses. Bearing in mind that the GBMs includes interactive tasks as a form of a multi-level approach to problem solving, we have also shown an approach to evaluating student's knowledge necessary for upgrading him/her to the higher level of learning. To assess a student's knowledge level needed for the next game level in the GBMs, we have developed an intelligent agent. This illustrates how intelligent agents and fuzzy logic can help increase the quality and quantity of the most important element of e-learning and that is making a decision. The results of student's knowledge diagnosis by means of agent within the GBMs e-learning system demonstrate the possibility of applying the presented agent model in various game-based learning systems for the determination of the knowledge level performance. On the basis of the data obtained through the exams, as well as through the use of statistical reasoning methods, we have shown the efficiency of the GBMs in the learning process.

K. Kuk (✉)

School of Electrical Engineering and Computer Science Applied Studies, Department for New Computer Technologies, University of Belgrade, 11000 Belgrade, Serbia
e-mail: kukkristijan@gmail.com

I. Milentijević · D. Rančić

Faculty of Electronic Engineering, Department for Computer Science,
University of Niš, 1800 Niš, Serbia

P. Spalević

Faculty of Technical Sciences, Department for Electrical and Computing Engineering,
University of Pristina in Kosovska Mitrovica, 38220 Kosovska Mitrovica, Kosovo

Keywords E-learning system · Pedagogical agent · Game-based modules · Intelligent agent model

2.1 Introduction

Game-based learning promises to be a successful approach to teaching computer science courses. Educational games and interactive simulations can enable a student to acquire knowledge in a specific field by playing a game successfully. To a great extent educational games can be referred to as computer science disciplines. Authors Shabalina et al. [1] have implemented the educational games concept in the learning game for C# programming language. Their system is based on the common game engine architecture, but it has been extended to the use in educational games and it consists of two high-level subsystems: a game engine and a learning engine. Computer Programming course is found to be difficult and boring. Thus, learning through games seems to develop students' motivation for the subject. Authors Roslina et al. [2] describe the perceptions of students at the Malaysian university (UTM) about using educational games for the purpose of self-learning within introductory programming courses. Virtual learning environments support teaching and learning in an educational context, offering the functionality to manage the presentation, administration and assessment of coursework activities. Callaghan et al. [3] demonstrate how immersive virtual worlds can be used for a game-based strategy for the purpose of teaching electronics and electrical engineering by using a collaborative team-based competitive format. Studies conducted in a Greek High School within the Computer Science course investigated potential gender differences with respect to the game-based learning effectiveness, as well as with respect to the motivational appeal of a computer game to learning computer memory concepts [4].

A learner model, also known as a student model, refers to the model constructed from observing the interaction between a learner and a learning system or instructional environment. A student model must contain the following important information about the user: domain knowledge, learning performance, interests, preferences, goals, tasks, background, personal traits (learning styles, aptitudes...), environment (work context) and other useful features [5]. Domain-specific information is organized into a knowledge model. The knowledge model has many elements (concept, topic, subject...) which students need to learn. In this chapter we have described the learning strategy for computer science curricula as a possible model for college students.

The strategy is founded on the research conducted in the field of teaching methodology on the one hand, and game-based learning features on the other. The learning environment should be customized to the individual learner's learning styles and educational needs with the quality of the learning experience continually validated and evaluated. Software agents can be used to support instructors

and domain experts with both course design and delivery. They can also support individual learners by personalizing course materials based on learning objectives, learners' characteristics, and learners' prior knowledge, and facilitating learners' interaction. Agent technology, a combination of artificial intelligence and software engineering, represents an exciting new means of analyzing, designing and building complex software systems [6]. Agent-based systems have been successfully used in many areas such as information collection/filtering, personal assistants, network management, electronic commerce, intelligent manufacturing, health care, entertainment, etc. [7, 8]. An agent works towards its goals. The agent's goal model ensures the agent will do the right thing at the right time. Agent technology has existed for a long time, but there are few researches combined with educational values or educational technology. Actually, personalized or adaptive game-based learning has become very popular in the game-based learning and game-based testing. This chapter illustrates the easy integration of agent technology into game-based learning system and a case study based on it.

Section 2.2 discusses in more detail the related work we consider most relevant. Section 2.3 presents some pedagogical elements of the student model realized through the Net-Generation students and Game-Based Learning (GBL); Sect. 2.4 includes the use of game-based modules—the GBMs as a new teaching approach to teaching units in the Computer Science courses. This section shows two types of multi-level GBMs: (a) module for the “Unary logical operations” teaching unit and (b) module for the “Z-buffer” teaching unit. Section 2.5 describes the proposed intelligent agent model for the assessment of the knowledge level for the game-based learning system, the results of which are estimated by an equation with variable coefficients. Finally, Sect. 2.4 illustrates experimental results of estimated values obtained through the agent model versus the results obtained through classical tests and the efficiency of the GBMs as a supporting tool for the preparation of the exam questions.

2.2 Related Work

Recent research into game-based learning has identified adaptability as an area that requires further attention. Adaptability is required in game-based learning simply because each person has a different way of learning in different learning environments—one size does not fit all [9, 10]. Thus, learning may be related to and influenced by the player's preferences and customization of elements within the learning environment. The IMS Learning Design (IMS-LD) is a specification for creating Units of Learning (UoLs) which express a certain pedagogical model or strategy (e.g., adaptive learning with games). In this sense the MS-LD can be complemented with off-the-shelf components and resources integrated in Units of Learning (UoLs). Author Koper [11] in the project named <e-Adventure> shows how an adaptive IMS-LD UoL can be modeled and integrated within an external resource such as educational game. The main goal of the project was to apply a

documental approach to the development of educational adventure video games (often also referred to as point-and-click adventure games or conversational games). However, the question is what sort of adventure games should be included in the curriculum and what degree of complexity should they offer. The authors Jeetinder and Jayanthi [12] responded to this question by offering a framework for creating individual simulations (modules) and organizing them in the form of multiple levels of a game. For educational games to be effectively integrated into the curriculum, they propose to concentrate on developing simple and small games and make explicit the relation the game bears to the lessons. This is better than creating very complex games with which the student is left to figure out the relation. As a result, this chapter shows how to create simple modules in the GBMs system as independent and interrelated concepts in the field of Computer Science.

Adaptive game-based learning is a fundamental issue for the next generation of educational games where progress is controlled in accordance with the learners' behavior. The major aim of an adaptive game-based learning system is to support and encourage the learners considering their needs, strengths and weaknesses. In their paper Weng et al. [13], proposed the framework of a personalized Quiz-MASTer assessment game and the flow of a personalized quiz game. With the services provided by an intelligent agent, a user can play the personalized assessment game by means of the flow of a personalized quiz game. Actually, the personalized or adaptive game-based learning has become very popular in the game-based learning and game-based testing. In the GBMs system proposed in this chapter an adaptive system approach to users is being realized through a pedagogical agent. Within the system the agent presents the modules on the basis of the estimated knowledge of every student.

Pedagogical agents are embodied software agents that have emerged as a promising vehicle for promoting effective learning. They provide customized problem-solving experiences and advice that are precisely tailored to individual learners in specific contexts. However, Conati and her colleagues [14] believe that the pedagogical agent could strike a better balance between learning and engagement if, in addition to the student's knowledge, it could have access to a student's affective reactions to the game. Bayesian techniques must be used when the agent's decisions about the states of the game world are uncertain, because a Bayesian network is often used to model the agent's uncertainty about its opponents. A project by Lester [15] at North Carolina State University focuses on the development of a full suite of Bayesian pedagogical agent technologies for inquiry-based science learning. It will provide a comprehensive account of the cognitive processes and results of interacting with Bayesian pedagogical agents.

In regards to Bayesian techniques, the implemented pedagogical agent in our e-learning system uses a simple fuzzy logic deduction—"if-then". Intelligent agents in combination with fuzzy logic can help increase the quality and amount of interaction in a computer game. The storyline often demands precise control over certain creature's properties, but autonomous agents may exhibit undesirable emergent behaviors due to the absence of centralized planning and control. In order to achieve this, our agent uses the students' knowledge assessment module in

the section about pedagogical module. The mentioned pedagogical module intended for calculating the current learners' level of knowledge in the GBMs system uses a simple mathematical formula, unlike some other agent-based game design technologies which often use the given BDI.net agent framework [16].

2.3 Method

2.3.1 Problem Statement

Upon analyzing the final exam results during the examination periods in the Computer Graphics course at the School of Electrical Engineering and Computer Science Applied Studies in Belgrade, we came to the conclusion that the results of one group of questions were much more different than the others. Performing an analysis we discovered that those were the questions referring to the field of hidden surface techniques and especially to the Z-buffer algorithm. Although students had the same learning materials for the purpose of the exam preparations in all fields, the difference discovered in this teaching unit showed that this field was quite complex and abstract for students. Therefore, it was necessary to take some steps in order to improve the approach to learning regarding this teaching unit, as well as to improve the final exam results. Of all the algorithms for finding visible surface, the Z-buffer algorithm is perhaps the simplest and therefore most frequently used. Starting from the facts that this teaching unit is simple and that, in spite of that, the students show worse results in this field than in any other, we began searching for the ways how to offer our students the teaching material which would be more student-friendly.

During laboratory exercises in the Computer Architecture and Organization course students brush up and improve their knowledge acquired in lectures through concrete tasks and under the supervision and help of assistant lecturers. Taking exercises without previously acquiring the basic terms in the field being taught in the lectures directly results in difficulties with individual realization of the laboratory exercises. The possibility of visual representation of the task solving method for rehearsing materials in this course enabled their implementation in the form of an interesting game. For the purpose of motivating students to get ready for laboratory exercises and get them more active in individual task solving during the exercises, the educational game ArhiCOMP has been created. This educational game contains interactive tasks [17, 18] implemented in the graphic environment, which directly associates it with the field of application use. The game has been designed to help the students learn basic terms referring to unary logical operations and to apply them practically through solving the given examples by randomly generated content of virtual registers, which are an integral part of the arithmetic-logical unit of computer systems.

2.3.2 *Characteristics of Students*

Nowadays, elementary and secondary school pupils, as well as university students, belong to the generation born in the Internet age. Modern psychologists, sociologists and pedagogues refer to them as the Net-generation. For the above-mentioned reasons, in this work we tried to make it easier for students to learn abstract educational materials and to improve their score in the final exam by using any type of interactive multimedia applications. The Net-generation students prefer learning by being told what to do [19]. They learn successfully through discoveries—both individually and with their age-mates. They learn by doing, and not by reading instructions from the manual or by listening to lectures. The Net-generation is more comfortable in the environment abound in pictures than in text. The researchers' report that the Net-generation students will refuse to read a large quantity of text, regardless of the length of the task or instruction. Rather they merely think or speak about activities they like to perform them. Each student has a different learning style. Some students learn best by visual learning exercises. These (visual) learners benefit from a variety of ocular stimulation. One example would be the use of color. Others do best by performing intellectual activities—problem solving and reasoning. Intellectual learners like to engage in activities such as solving problems, analyzing experiences, doing strategic planning, generating creative ideas, etc. [20].

In order to make it easier for students to learn and acquire knowledge in the computer science courses that cannot be seen with the naked eye or observed on the basis of pictures, we used the characteristics of two types of students: intellectual and visual. Bearing in mind the fact that the intellectual-type learners are fonder of educational material in the form of simulations and interactive tasks, we created our environment for learning the Z-buffer algorithm as an environment in which the task solving is performed as a simulation of the operation of the mentioned algorithm. On the other hand, taking into consideration the fact that visual-type learners remember graphically presented information more easily, the environment was enhanced with various colors and shapes for concepts as integral parts of the algorithm. We reduced the definition of registers content used by the algorithm to decide upon the color, and the register itself was presented as a series of squares, where each square stands for 1 bit.

It is considered that educational games improve learning due to a better learning method that meets the needs and habits of the Net-generation students. To provide this kind of support for learning is at the same time extremely important and represents an extreme challenge. Creating special learning systems that contain educational games represents a challenge since it demands a careful analysis of stimulating learning and maintenance of positive motivation.

2.3.3 *Simulations and Games*

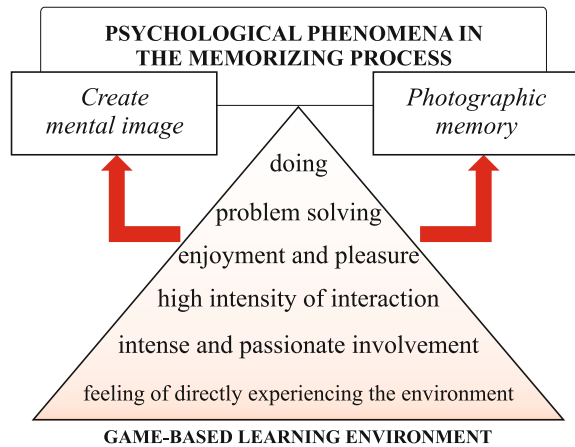
A memo-technical rule says that in order to memorize a term or a definition more easily, they should be made interesting. Motivation is also increased when we do something amusing. In this regard, one of the significant techniques is visual representation. If a piece of information can be represented by a visual picture or animation, then it should be represented in that way. Simulations and games, as highly interactive multimedia applications, can increase students' motivation for science learning, deepen their understanding of important science concepts, improve their science process skills, and advance other important learning goals. Through solving the tasks in the form of interactions the students are required to recognize the simulation of the operation of an algorithm or process, but they are not afraid of giving incorrect answers or performing wrong actions.

After performing an analysis of the existing Internet simulations and other interactive multimedia applications used in the teaching process, we came to the conclusion that it is necessary to introduce some of those contemporary teaching resources to the course in Computer Graphics, so that students could learn the planned material in the best possible way. However, since the students to whom such a type of education material is to be presented do not belong to the generation for which the existing applications and modules had mostly been made, we had to introduce additional pedagogical elements to those applications to make them more acceptable for the students. On the one hand, these applications certainly have to be amusing, while, on the other hand, they must have an educational character.

Regarding the pedagogical elements supported by the games-based learning, Norman [21] identifies seven basic requirements for a learning environment: (1) to provide a high intensity of interaction and feedback, (2) to have specific goals and established procedures, (3) to motivate, (4) to provide a continual feeling of challenge that is neither so difficult as to create a sense of hopelessness and frustration, nor so easy as to produce boredom, (5) to provide a sense of direct engagement, producing the feeling of directly experiencing the environment, directly working on the task, (6) to provide appropriate tools that fit the user and tasks so well that they aid and do not distract, and (7) to avoid distractions and disruptions that intervene and destroy the subjective experience. By adding certain inherent engaging elements, suggested by Prensky [22], we created the most acceptable learning model for our type of students. As a combination of visual and intellectual types of learning, the game-based learning environment with its characteristic given in Fig. 2.1 will improve their psychological capability of photographic memory and help them to create mental images of concepts being learnt.

Stimulated by modern technologies, teachers use computer games and simulations more and more frequently in order to motivate students. Although video games can potentially be beneficial for learning, they must be aligned to specific curriculum content to achieve solid gains in learning. The three factors influencing the possible choice of the existing interactive multimedia applications for learning the subject were: content, age and learning styles.

Fig. 2.1 Characteristics of GBL environment that are used to help psychological student's phenomena



2.4 Game-Based Modules

The analysis of the existing multimedia interactive environment for learning in the field of education shows that there are three types of applications used effectively for teaching mathematics and computer engineering to students. Those types of existing educational applications are:

- Gizmos,
 - IMMEX,
 - Interactive tasks.
1. **GIZMOS:** ExploreLearning's Gizmos are an effective and engaging way to move students to inquiry-based science and math. The students get to see real simulations that they read about in textbooks. It gives real life problem-solving a whole new meaning! Interactive simulation that makes key concepts easier to understand and fun to learn [23]. World's largest and most advanced online repository of math and science simulations for grades 3–12. Gizmos are online simulations that are powerful teaching and learning tools to engage students. Their easy-to-use format makes them practical and effective. Students manipulate key variables, generate and test hypotheses, and engage in mathematical inquiry. Gizmos supplement and enhance your instruction with powerful visualizations of math concepts. Gizmos are an effective and engaging way to move students to inquiry-based science and math [24]. Gizmos move students to use higher-level thinking skills.
 2. **IMMEX:** Interactive Multimedia Exercises (IMMEX) is an online library of science simulations that incorporate assessment of students' problem-solving performance, progress, and retention. Each problem set presents authentic real-world situations that require complex thinking. Originally created for use in medical school, IMMEX has been used to develop and assess science problem

solving among middle, high school, and undergraduate science students as well as medical students. Students navigate a hierarchy of menus and submenus to select different pieces of information which are each available at a cost. This form of task structure and subtask boundaries would be expected to elicit different levels of cognitive engagement as students explore and seek out relevant information [25]. The use of IMMEX software has been scientifically shown to have significant positive effects on students' understanding of science content as well as the process of scientific investigation [26]. By integrating our multimedia simulations into a unique web-based learning platform for modeling strategic thinking and problem solving, IMMEX is able to help teachers.

3. **INTERACTIVE TASKS:** Through an adaptive approach, with visual indication of the course of performing the task, students are enabled to learn the procedures for solving tasks from various fields and of various complexity levels. The very applications are created in the well-known software package for creation of multimedia content Adobe Flash CS4. Students' interactivity with this type of applications was achieved with the use of graphic symbols (different types of buttons) and specific colors. Realized interactive tasks [27] differ in: (1) The way of giving answers, (2) Visual guidance during the task execution, (3) The level of complexity. Interactive tasks will not allow advancing to the next level of solving unless the student previously fills or selects interactive fields in the previous level accurately. From the pedagogical aspect, these applications stimulate students to make conclusions on the accuracy of achieved steps in solving of the entire task on their own. In this interesting way, through self-revealing students acquire the needed knowledge quantity and thus carry out the learning process without tutor's intervention or previous knowledge, through random guessing.

By combining characteristics of these three types of educational applications, we have created a model learning environment (Fig. 2.2) that will be acceptable for implementation of teaching units in the course Computer Science, on one side, and for previously analyzed type of students, on the other side. In order to increase the engagement and interest of students for this type of teaching material, we included game characteristics in this environment. Thus the game concept should be based on two components: (a) learners must get the course information through its interpretation in the game world; (b) learners must see the result of this algorithm in a game context. Also, besides placing a game interface into learning environment we have also applied basic game elements, such as: result, time and difficulty levels. These new modules, which include game elements, represent research multimedia learning applications and are intended for Computer engineering students. These new learning environments, which include game elements, represent research multimedia learning applications and are intended for Net-generation students, we named game-based modules (GBMs).

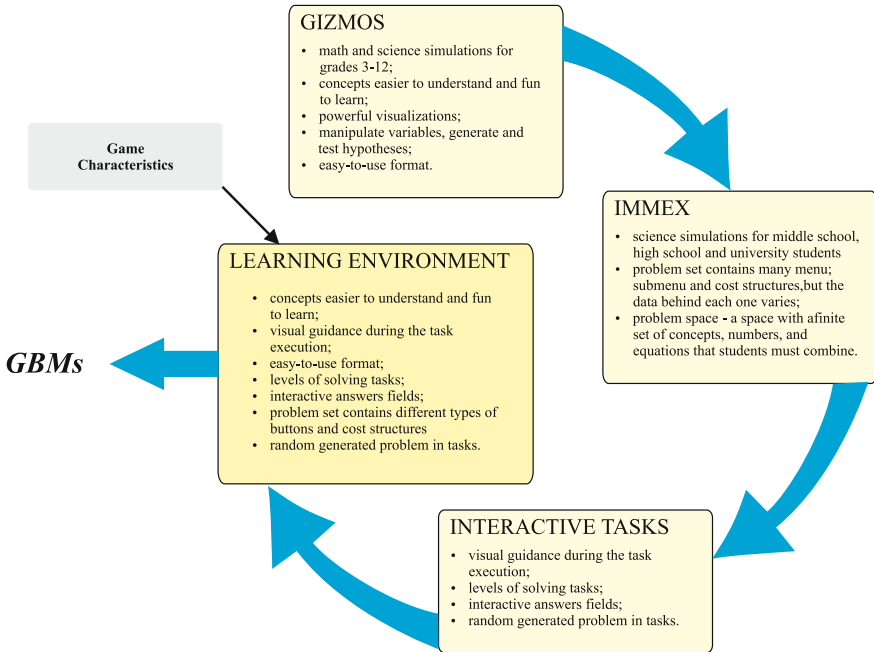


Fig. 2.2 Evolution of GBMs

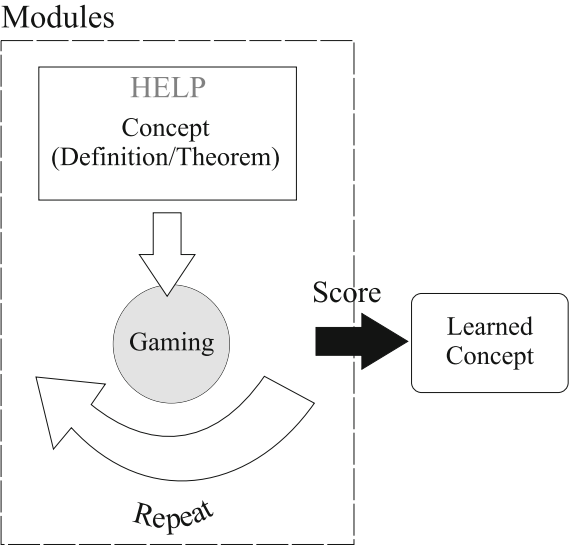
2.4.1 Implementation of GBMs

The basic terms that explain the principle of operations functioning are reached by selecting the Help option in modules. When a student starts learning with the use of the game or faces a difficulty during solving a task generated by the application, Help serves to accelerate finding the right solution. This means that formulation of definitions and theorems within Help is the key moment in designing the entire application. The purpose of learning through the game is to enable students to learn the rules and check them in practice on the example of all unary operations. Multiple repetition of tasks with performing the same operation increases the probability of learning characteristics and use of particular operation. Quality evaluation whether an operation is acquired or not is performed through visual indication of the number of successful and unsuccessful tasks (score) with the same operation, and comparison with preset criteria. The model of the e-learning system GBMs is shown in Fig. 2.3.

2.4.1.1 Z-Buffer Module

The possibility of visual representation of the task solving method for rehearsing material in the course Computer Graphics enabled their implementation in the

Fig. 2.3 The model of the e-learning system GBMs



form of game-based module. The game has been designed to help students learn basic terms referring to the principle of the Z-buffer algorithm operations and practically apply them, through solving given examples with randomly generated content of buffer registers. This GBMs contains interactive tasks implemented in the graphic environment, which directly associates with the field of the application use.

By using advantages of the concept map technique, on the course in Computer Graphics we created a concept map for the teaching unit Z-buffer, with the aim to reduce the items presented in this unit to main concepts and to connect them in the simplest possible way. Having in mind the terms students should learn in order to successfully acquire knowledge about the Z-buffer algorithm functioning and thus have the needed knowledge for the exam, prior to creation of the conceptual map we marked this as the needed knowledge (Fig. 2.4).

Starting from the theoretical basis of this algorithm functioning, its integral parts (sub-concepts), without which the algorithm cannot function, are the concepts it contains—Z-buffer memory, Screen buffer memory and Scanning line. Every memory type (including the two mentioned) consists of a series of buffer registers containing a series of bits. This type of algorithm uses a 16-bit buffer, and the content of bits depends on the functioning principle of two types of tests, which are key sub-concepts of this teaching unit. In case that a Depth test is used for determining the depth of the observed polygons in relation to the observer, the test results are placed in the buffer register bits. On the other hand, results of the Color tests are also put into the buffer register as a series of 16 various bit values. As shown on the conceptual map, the basis of this algorithm functioning is determining the bit value content (1 or 0). Determining the bit that is active and whose content is to be filled determines the scanning line position in the algorithm.

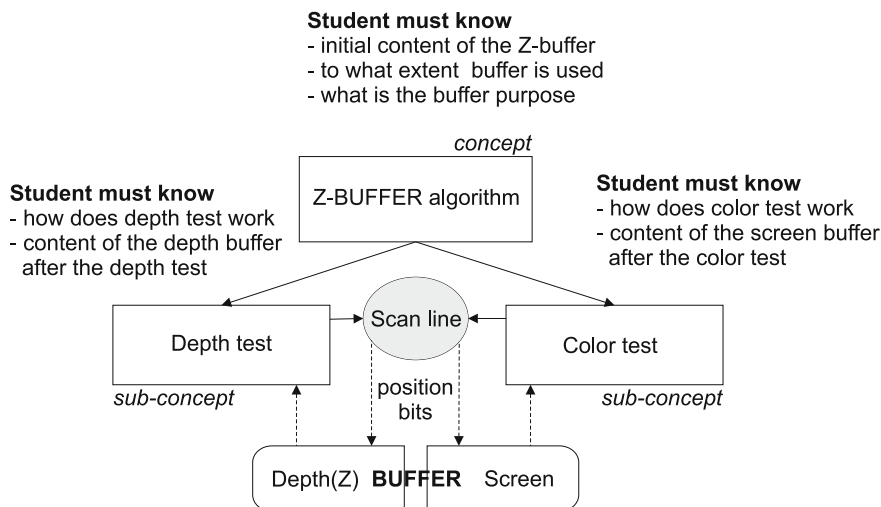


Fig. 2.4 The conceptual map for the GBMs Z-buffer module

Having in mind the concepts a student should learn, the student should connect these concepts through GBMs and successfully solve the given tasks. The tasks are given as a part of the game, and in the role of players students are motivated to solve them in order to advance in the game. Since two main concepts are given in the concept map (the depth and color test concepts), which are to be learned by students through this module, we presented them as two levels with different solving difficulty. Skill-based learning as a part of the micro game cycle completely fits into levels within the game. Many learning characteristics can occur during the game when the player attempts to go from one level to another. Adding time as a game element that contributes to the player's better ranking in the game also results in an increase of knowledge and improvement of the skills acquired through easier levels.

With the use of techniques for the first class innovative testing select/recognize [28], we came to the idea that answers in the GBMs can be given as a series of image fields on which a student should click. Since the buffer we use in the Z-buffer algorithm uses 16 bits, the task of this module is to determine the value of each bit, i.e. contents of the registry in various situations presented in the interactive task. The correct answer to fill the content of one bit is one of the proposed answers presented to students in the form of squares to be selected [29]. These squares, i.e. offered answers, are presented in two ways. In the level 1 of the module answers are offered in the form of a 13-square column (type 1). When certain square is selected in the scanning line (which shows the current active field in the task), the square falls down and fills the content of the active bit in the buffer. Answers in the level 2 are also offered as an array, but in the form of a five-square row (type 2). This row with offered answers is not constantly visible, but is

shown only when the given bit is selected as a sub-menu in the menu list. The task at this game level is to determine the color in the screen buffer by determining the resulting color in each individual bit. The resulting color should be the result of overlapping of two or three pixel colors as parts of overlapping polygons in the given scanning line.

2.4.1.2 Module ArhiCOMP

The teaching unit “Unary logical operations” is aimed at teaching students about the way of performing logical operations at the level of registers in the computer system, through comparing the register binary contents before and after performing of the given operation. When it comes to operations for moving to the left or right, what is illustrated is the way of hardware implementation of the arithmetic operations division or multiplication with a degree of number 2. If students want to know how to apply an unary logical operation, i.e. if they want to know the register contents after the applied logical operation, they have to know basic rules of the binary digit system and rules referring to unary logical operations, such as the rule for logical shift to the right. Acquisition of basic terms is facilitated with the use of appropriate graphic representations, which presents the contents of the accumulator register in the arithmetic-logical unit before and after the shift operation. The arrows show the moving direction and new positions of bit in the register, as well as the contents of Carry flag in the condition register. Good knowledge of the set of rules from the field of unary logical operations is a precondition for future successful acquisition of knowledge in other fields within the course Computer Architecture and Organization or related courses in higher years of studies.

Starting from the fact that a well designed visual environment can attract the attention of students and motivate them to spend more time solving tasks within the educational game, special attention was paid to selection of the background, which in this case consists of various forms of binary statements presented in bright colors with effects of brightness, transparency and reflection, characteristic for new “fancy” technologies. To make working in the application more interesting, components in the game are not fixed but can move on the screen independently, which gives students comfort in the process of task solving. Moving and overlapping of components such as registers enables easier defining of their contents when a complex operation is applied. The task of the game is to determine the contents of Register 2 (which represents Register 1 immediately after the selected operation) in relation to contents of Register 1, which are randomly generated and appear after selection of the unary operation, together with the task text. When the student selects a bit in Register 2, the falling menu enables entering of the particular bit in the virtual register through selection of one of the options 0 or 1. Text of the task constantly changes on the basis of randomly selected values presented in the very text. Attempts to solve the task with the same text once again are reduced to a minimum. Observed from the pedagogical side, the repeated task solving prevents mechanical solving, but stimulates students to show the real level of acquired

Fig. 2.5 The help option in the GBMs module ArhiCOM



knowledge through previous problem solving. The basic terms that explain the principle of logical operation functioning are reached by selecting the Help option in the game (Fig. 2.5). When a student starts learning with the use of the game or faces a difficulty during solving a task generated by the application, Help serves to accelerate finding the right solution. This means that formulation of definitions and theorems within Help is the key moment in designing the entire application.

2.5 Student Modelling in E-Learning System GBMs

In tutoring systems, students can learn new concepts and recognize the relationships between the previously learned and new concepts. This knowledge is represented as a conceptual map in the GBMs system. Hwang [30] introduced the “Concept Effect Graphs” where the subject materials can be viewed as a tree diagram comprising chapters, sections, sub-sections and key concepts to be learned. In order to obtain maximum benefits from the material, the teacher must perform a very difficult task referring to the use of conceptual maps in the realization of the teaching units. In the process of designing and creating the teaching units a lecturer may find concept maps very useful. Global “macro maps” can also be made, showing the main ideas we want to present during the entire course, or specific “micro maps”, showing the structure of knowledge for specific fields. The concept maps are graphic tools for organizing and presenting the knowledge base.

The conceptual map approach offers an overall cognition of the subject contents. The diagnosis process can be easily implemented through this approach. If a student fails to learn the concept “sub-concept 1/level 1”, it is possible that the student did not learn the concept “sub-concept 2/level 2”. Therefore, the system suggests that the student needs to study the sub-concept 2 again. For this reason, the GBMs knowledge base of the must show the relationships between concepts. To do this, a conceptual map-based notation is proposed. Let us suppose that C_i and C_j are two concepts and if the concept C_i is a prerequisite for the concept C_j then the concept-effect relationship $C_i \rightarrow C_j$ exists.

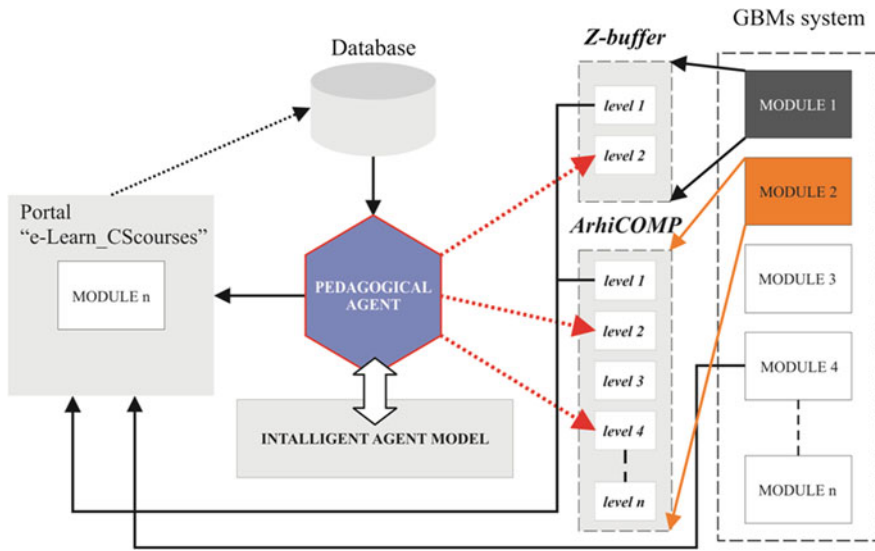


Fig. 2.6 Pedagogical agent in GBMs

The repository of knowledge stored in the knowledge base of the GBMs is structured as follows: a set of sub-concepts that represent atomic content teaching unit parts—concepts, a concept map that describes the interdependence between the sub-concepts. Each sub-concept is composed as a module including interactive tasks as game levels. Bearing in mind that the GBMs have interactive tasks implemented as multi-level problem solving, we have also shown the model for evaluating the knowledge necessary for upgrading to the next level, as shown in Fig. 2.6. To assess a student's knowledge level needed for the next game level in the GBMs, we have developed an intelligent agent model for the knowledge level assessment. By means of the intelligent agent model we have managed to assess a student's knowledge at the current game level, which has provided us with the basis upon which it is possible to decide if a student should move to the next level of learning or if he should stay at the same level.

2.5.1 Pedagogical Agent and System

The student is introduced to the set of all realized GBMs through the Internet portal “e-Learn_CScourses”. The GBMs represent separate concepts that are being loaded on to the portal. Every module contains two or more separate files loaded within the module. The files represent the external *swf* files, while each one of them comprises a sub-concept within the module. Upon a student’s registration and logging on to the portal, the Internet browser shows the Flash-supported

environment as well as the possibility of choosing a pre-defined teaching units list. When a student chooses a certain teaching unit an adequate module based on the chosen unit appears on the portal shell. Afterwards the portal itself guides the student through other teaching units, loading the external *swf* files until it finishes displaying the associated sub-concepts in the knowledge domain. Up to this point the student is guided by the portal without being able to choose the files which are to be loaded on to the shell.

A teacher defines in advance the order in which the file within the GBMs is loaded, as well as the modules within the portal shell. In the process of creating the order of presentation the teacher divides the course into sets of teaching units. Afterwards, these are causally connected, thus creating the knowledge domain. Subsequently, the teacher ranks them according to their difficulty and complexity. The unit which does not require the knowledge of other units becomes a candidate for the easiest level and it is displayed on the list students can choose by themselves. The units at the more difficult levels are ranked according to their complexity and they are themselves loaded on to the portal. Upon finishing the ranking of units or concepts the same rule applies to ranking sub-concepts within the concepts. The example of sorted Computer Science course concepts can be seen in Table 2.1.

Since in the process of learning a student needs to show adequate knowledge about a specific concept or sub-concept, the next step in the assessment of causal relationships relating to the domain is to determine the student's knowledge necessary for the transition to the next concept, i.e. to the module itself or some of the game levels within the module. On the portal this task is performed by a pedagogical agent. The pedagogical agent assesses a student's current level of knowledge in the system and on the basis of this assessment it loads external databases into the system. Within the system the agent is never revealed to the student nor does the student know that the agent is being used by the system.

Table 2.1 Classification of certain concepts within the computer science course

Concepts	Sub-concepts	Complexity	Precondition
Data representation		1	No
Computer organization		1	No
Binary arithmetic	Add	1	No
	Subtract		
	Multiply		
	Divide		
Operation on bits	Unary logical operations	2	Binary arithmetic
	Binary logical operators		
Data manipulation		2	Computer organization
Machine language programming		3	Data manipulation
			Computer organization
Display systems		1	No
Z-buffer algorithm	Depth test	2	Display systems
	Color test		

Related modules are loaded independently and students do not decide upon the loading order. It is done by the agent on the basis of a conceptual map given by the teacher as well as on the basis of the knowledge assessment model.

While using the loaded module, on the basis of a student's logging on to the portal the data on his interaction with the modules are stored in the system database. Upon finishing the module game level the pedagogical agent receives the information about a student's interaction with the system and then by means of a simple "if-then" rule reaches the decision to let the student move to the next game level in the same or different module (thus loading another module on to the shell). The fuzzy rules are interpreted as:

if ($P(X) > = \text{necessary_KNOWLEDGE}$) *then* ($SHELL = \text{next_LEVEL}$)
else ($SHELL = \text{current_LEVEL}$)

If the agent decides that a student does not possess sufficient knowledge to move to the next level (sub-concept), the agent does not load the next level (sub-concept) within the same module (concept), but forces the student to return to the same level. The teacher determines the level of knowledge— $P(X)$ —that the student needs to master on the basis of the analysis of the connection between the concepts in the knowledge domain. However, the question is how to calculate the current student's knowledge in the GBMs system? The answer to this question lies in the pedagogical model for calculating the current level of a student's knowledge which is used by the agent to display the module on the portal.

2.5.2 Designing Intelligent Agent Model

One of the most common solutions for the student diagnosis in ITS is testing. Generally speaking, test-based diagnosis systems use heuristic solutions to infer students' knowledge. In contrast, Computerized Adaptive Testing (CAT) is a well-founded technique, which uses a psychometric theory called Item Response Theory (IRT) [31]. The IRT supplies several methods to estimate students' knowledge. All of them calculate a probability distribution curve $P(\theta|u)$, where $u = u_1 \dots u_n$ is the vector of items administered to students. When applied to adaptive testing, the knowledge estimation is accomplished every time the student answers each item posed, obtaining a temporal estimation. The distribution obtained after posing the last item of the test becomes the final student knowledge estimation. One of the most popular estimation methods is the Bayesian method [32]. It applies the Bayes theorem to calculate students' knowledge distribution after the posing an item i .

The students' knowledge level in an adaptive hypermedia application [33] is measured by using the answers to the questions previously presented to the student. The decision whether a student has solved the task presented in our proposed intelligent agent model is made on the basis of a formula which, aside from the

correct answers, includes two additional parameters (time and the number of used Help options [34]).

In the ITS approach [35], user model content variables are used for keeping records on user interaction with the ITS and for adjusting the content presentation to the user profile. These learning style variables are a part of the Bayesian Network—BN for drawing conclusions about the student. There is a list of variables for each topic: spent time, topic depth level, wrong answers and correct answers. The variables relevant to deciding on students' knowledge in the GBMs system also include spent time and correct answers. However, the time needed for solving the tasks within our approach is divided into: (1) time used to read contents of the Help window and (2) time needed for giving the answers when the Help windows were not used.

The idea of the Neuro-Fuzzy Reasoner (NFR) system was the initial inspiration to create the model for students' knowledge diagnosis in a game-based learning system. The NRF system is relatively simple, supports creation of high-level pedagogical strategies, and can be easily adapted to individual teacher's preferences. The NRF model for student classification is based on test results and the time needed to complete the test. Modification of the NRF system parameters is made by adding a new parameter—time needed for reading the contents of the Help window. The learning model we have used in this education game is based on the operation principle of the NRF presented by Sevarac in his work [36]. The intelligent agent model has been extended with one more input variable—the Help window, because this component has been used by students in the educational game to a great extent. The initial model which we started with and which we used in the module N was based on the NRF.

Since the initial model had not produced the expected results, we applied the new model for knowledge level estimation which used the coefficients as variable values. The rule for determining the percentage of knowledge applies basic arithmetical operations to input parameters of the model. The significance of the input values for final knowledge estimation is determined on the basis of empirical coefficient values, given by the teacher. The knowledge level that student possesses after playing the education game is given by the following formula [37]:

$$P_i(X = \text{Mastered}) = \left(a \times \frac{A_i}{N} - h \times \frac{H_i}{N} - t \times \frac{t_a \cdot (A_i - H_i) - t_h \cdot H_i}{T_{\max}} \right) \times 100 \quad (2.1)$$

where

- a , h and t are coefficients that should be estimated,
- A_i Number of correct answers of i -th student,
- H_i Number of opened help windows of i -th student,
- t_a Average time a student needs to give an answer without using help window,
- t_h Average time a student needs to give an answer when using help window,
- N Total number of answers,
- T_{\max} Maximum duration of the game.

The values of coefficients: a , h and t can have a range from 0 to 1. The first part of the formula has the most significant role in calculating a student's final knowledge level, since it uses the number of correct answers— A_i . The second important part for knowledge estimation is the second part of the formula, which represents the number of used Help windows during the game— H_i . This part of the formula has the negative sign, since it decreases the probability of the final knowledge level. The part of the formula which depends upon time has the least importance for knowledge calculation. When a student uses the Help window, the time needed for giving an answer increases, which results in reduction of the student's total knowledge.

The basic terms of knowledge that a student has to present on a test are written in the help windows in the game. When a student starts learning by using the game or faces a difficulty during solving a task generated by the application, the Help window serves to accelerate finding the right solution related to a particular task. This means that an appropriate formulation of definitions and theorems within the Help window is the key moment in designing the entire application. There is a need to optimize the maximal duration of the game T_{\max} . We have selected the approach to limit T_{\max} by setting $T_{\max} \geq N_{th}$. This admission is based on the assumption that when a student does not know the answer to a single question, he will use the Help window for each, namely N th. The inequality in relation is set, because there is some additional time provided for a student to decide if he will use the Help window, (if he is not sure enough that he has a correct answer). If a student thinks that he has the correct answer to the query, he will provide an answer in shorter time t_a and estimated knowledge level will be higher according to the Eq. (2.1).

Based on teachers' estimation, the values referring to the significance of coefficients in the given formula are estimated as:

- $a=0.50$,
- $h=0.35$,
- $t=0.15$.

where the sum coefficients must be 1. Through repeated comparison of the results obtained through classical paper test and results obtained through a computer game by using the estimation of the knowledge level with the new model (Eq. (2.1) and empirical coefficients) we have come to very encouraging results. With the use of the new model we have managed to reduce the error in estimating the students' knowledge level by 20 %. The new error value is 29.97 % ($\varepsilon = 0.3$) and it is reached by means of the above given formula and the empirical coefficient values.

The goal of the intelligent agent model is to estimate the knowledge of students as sufficient enough to let them pass the current level of module (sub-concepts) and go to the next one. Another goal is to compare and narrow the difference between the results obtained through playing a game and results obtained through classical examination methods. The results of classical examinations are graded into three groups: bad, good and excellent. Thus, we have graded the results obtained through using intelligent agent model in the same manner. The output of the model

estimates students' knowledge sorted into three grades: bad ($P(X) \leq 45$), good ($45 < P(X) < 85$) and excellent ($P(X) \geq 85$). In our experiments we have obtained results through classical paper-based tests that deviate a lot from the results obtained through playing a computer game. For example, some students passed the classical examination test with the highest scores ($P(X) = 100$) but they answered less than 3 out of 8 level tasks in playing the game during maximum time of game duration. Or quite the contrary—some students achieved very bad results in classical examination tests ($P(X) = 0$) and they answered correctly to more than 5 out of 8 level tasks in playing the game.

The selection of optimal coefficients is performed by using algorithm for minimizing the root mean square (RMS) error [shown at Eq. (2.2)] between the classical test results of students and the result of estimation model calculated by the Eq. (2.1):

$$a_{opt}, h_{opt}, t_{opt} = \min_{\{a,h,t\} \in (0,1)} \sqrt{\frac{1}{N} \sum_{i=1}^N (R_{i \text{ classic}} - R_{i \text{ est}})^2} \quad (2.2)$$

where $R_i \text{ classic}$ is result of i th student obtained by classical paper-based examination and $R_{i \text{ est}}$ is result of i -th student estimated by intelligent agent model. The C++ like code of algorithm implemented [38] is given in Fig. 2.7. At the output of this algorithm the optimal values of a , h and t are calculated according to RMS rule.

```

minError = 1000;
for(a=0; a<=1; a+=0.01){
    for(h=0; h<=1; h+=0.01){
        for(t=0; t<=1; t+=0.01) {
            ErrSum = 0;
            for (nStud=0; nStud<NSt; nStud++){
                Rest = (a*A[i]/N - h*H[i]/N - t*Ttot[i])*100;
                if (Rest > 85) Rest = 100;
                else if (Rest >= 45 && Rest < 85) Rest = 50;
                else Rest = 0;
                ErrSum +=(Rest - Rclassic[i])*(Rest -
Rclassic[i]);
            }
            Err = sqrt(ErrSum/NSt);
            if (Err < minError){
                aopt=a; hopt=h; topt=t;
            }
        }
    }
}

```

Fig. 2.7 C++ like code for calculation of optimal coefficients

The similar results may be obtained if other error criteria (like minimum average absolute error, maximal absolute error, etc.) are selected. The values of coefficients a , h and t calculated by the Eq. (2.2) are 0.55, 0.30 and 0.15. This leads to the error of 25.35 % ($\varepsilon = 0.25$). This is the main reason why the error of intelligent agent model estimation is even lesser.

2.6 Evaluation Results and Discussion

Modeling students' knowledge in educational games involves a high level of uncertainty. For this reason, the presented agent model could be applied through the pedagogical agent to game-based e-learning systems as a student knowledge diagnosis engine. Game-based learning represents one kind of software applications that uses games for the purpose of learning or education. The aim of such an application is to help the students understand the topics by visual representations of pertinent processes. In our evaluation we were interested in studying two factors:

- the search for the best coefficient values to make the model as precise as possible,
- how well the students learn the concepts from the GBMs environments.

Within our first study, we have analyzed the results achieved for each sub-concept through the classical paper-based test by the students who used the modules of our portal for the purpose of studying. The results of this test were collected in order to be compared with the results of students' knowledge obtained by means of intelligent agent model while playing the GBMs. The parameters were recorded during the playing modules of 71 students. The analysis has taken into consideration only the group of 50 students who used the modules in a time span lesser than maximum time allotted T_{\max} . Also, this group did not include the results of the students who did not use the Help option while using the module itself. The reason for neglecting the Help option is the fact that the students who already had the knowledge about the given concepts used the module simply to practice what they had already mastered.

On the other hand, there are the students who wanted to finish the task in the module in the shortest possible time disregarding the potential inaccuracy of the given answers and thus ignoring the possibility of using the Help option. In order to calculate a student's knowledge by means of formula (2.1), the intelligent agent model in this study uses the following optimal values calculated by the Eq. (2.2): $a = 0.62$, $h = 0.25$ and $t = 0.13$. The mistake made by the agent model compared to the results of classical examinations on a paper test was very small $\varepsilon = 0.18$, as shown Fig. 2.8.

On the basis of the represented optimal values for the coefficients in this study the pedagogical agent has created the following relations in the knowledge

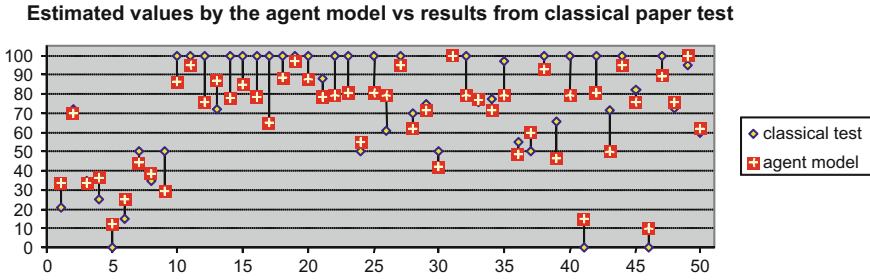


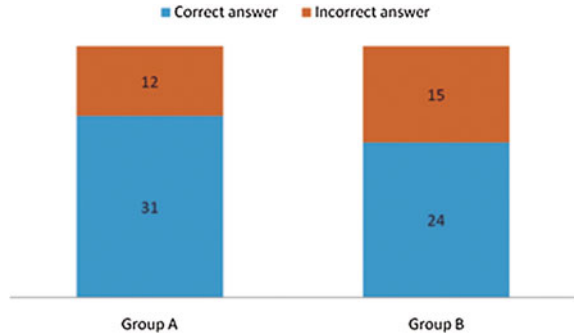
Fig. 2.8 The mistake made by the agent model compared to the results of classical test

assessment: the knowledge of 9 students has been equally estimated, the knowledge of 35 students has been underestimated while the knowledge of 15 students has been overestimated. Generally speaking, the results obtained through classical paper-based tests are better than values estimated by the agent model. In the intelligent agent model, each of the input parameters is weighted and the main task has been to calculate weighting coefficients in order to match the results estimated in this model to the results obtained through classical paper-based test. By comparing the results with the same set of the archive data, we also came to a conclusion that the values in formula (2.1) are partially sensitive to certain changes. The accuracy of the evaluation stays almost the same if, for example, we change the value of the rate t from 0.10 to 0.20 and the value of the rate h from 0.30 to 0.55. Slightly larger sensitivity of evaluation has been noticed while changing the value of the rate a . In order to maintain the model accuracy in a student's knowledge evaluation, the value of the rate a can vary from 0.50 up to 0.75.

Our second study dealt with the efficiency of the use of the GBMs as a supporting tool for preparing the exam questions in the field of the Z-buffer algorithm and the Unary logical operations. The total number of students attending was 183. However, the students that gave the “I don't know” answer were not taken into consideration in the analysis of the achieved results referring to this exam question. After the exam, we analyzed the results achieved by the students. We investigated the difference in the use of the GBMs between two groups of students, Group A and Group B. The group of students that used the GBMs for the purpose of the exam preparation was marked as Group A, while the other group that did not use the GBMs was marked as Group B. There were 82 students in each group. The exam results achieved by both groups are given in Fig. 2.9.

As shown in Fig. 2.9, Group A achieved better results than Group B. The difference in the number of correct answers is bigger than the difference in the number of incorrect answers. On the basis of these data, we can assume that Group A that used the GBMs was more successful than Group B that did not use the GBMs. To check this assumption we will use the method of statistical hypothesis testing in our research. In statistical research, the starting points are two mutually exclusive, opposite assumptions regarding the testing result—zero (H_0) and alternative (H_a) hypothesis:

Fig. 2.9 The exam results achieved by both groups



H_0 There is no statistically significant difference in distribution of correct and incorrect answers between Group A and Group B.

H_a There is statistically significant difference in distribution of correct and incorrect answers between Group A and Group B, and it is not accidental.

To determine whether the differences we observed within the comparative table are statistically significant (whether the distribution of values in rows and columns is independent), we used the X^2 test (Pearson Chi Square). The following values were obtained: $X^2 = 1.03$, $df = 1$, $p = 0.310$. From the Chi Square distribution table we read X^2 values for the selected significance level and the corresponding number of the freedom level. The corresponding probability $X^2 (\alpha)$ is 0.455. Since $X^2 > X^2 (\alpha)$ we automatically accepted the alternative hypothesis as the truth and concluded: the difference is statistically significant and it probably occurred under the influence of the experimental factor, which is in our case the GBMs. We have thus justified the use of the GBMs as a significant factor in the learning process. The presented results and conclusions drawn in this chapter attach considerable significance to the use of the GBMs in the fields that are abstract and difficult to understand, especially if we bear in mind that Net-generation students are not quite interested in the classic manner of learning. In that sense, the GBMs can represent good teaching material for other technical science courses as well.

Although we have done a lot of research on the intelligent agent model developed and used in the adaptive game-based learning, as well as virtual learning environment, our survey is not complete. A more thorough survey of the current trends in the applicability of knowledge management to e-learning system such as Moodle needs to be done. In the future we plan to create many GBMs for other units of Computer Science course and implement Moodle platform as classroom resources. In Moodle, any resource may be hidden by selecting option Hide from teacher. We have tried to develop Moodle plug-in based on the intelligent agent model which would automatically show the modules on grounds of estimated knowledge. The amount of knowledge $P(X)$ necessary for students to reach the next module should be determined by the teacher. A large amount of data is to be gathered to find the optimal coefficients for the proposed model.

References

1. Shabalina, O., Vorobkalov, P., Kataev, A., Tarasenko, A.: Educational games for learning programming languages. *System* pp. 79–83 (2008)
2. Roslina, I., Wahab, S., Che Mohd Yusof, R., Khali, K., Jaafa, A.: Students perceptions of using educational games to learn introductory programming. *Comput. Inf. Sci.* **4**(1), 205–216 (2011)
3. Callaghan, M.J., McCusker, K., Losada, J., Harkin, J., Wilson, S., Dugas, J., Demots, S., Desbois, F., Fouquet, A., Sauviat, F.: Game-based strategy to teaching electronic and electrical engineering in virtual worlds. *International IEEE Consumer Electronics Society's Games Innovations Conference (ICE-GIC)*, pp. 1–8, 21–23 (2010)
4. Papastergiou, M.: Exploring the potential of computer and video games for health and physical education: A literature review. *Comput. Educ.* **53**(3), 603–622 (2009)
5. Nguyen, L., Do, P.: Learner model in adaptive learning. *Proc. World Acad. Sci. Eng. Technol.* **35**, 396–401 (2008)
6. Wooldridge, M., Jennings, N.R.: Intelligent agents: Theory and practice. *Knowl. Eng. Rev.* **10**(2), 115–152 (1995)
7. Guttman, R., Moukas, A., Maes, P.: Agent-mediated electronic commerce: A survey. *Knowl. Eng. Rev.* **13**, 147–159 (1998)
8. Shen, Z.Q., Gay, R., Miao, Y.: *Agent-based E-Learning System—a Goal-based Approach, Business and Technology of the New Millennium*, EdCTLeondes, Kluwer Academic Press International, vol. 3, Chap. 6, (2004)
9. Kelly, D., Tangney, B.: Adapting to intelligence profile in an adaptive educational system, *Interacting with Computers*. Elsevier **18**, 385–409 (2006)
10. Villaverde, J.E., Godoy, D., Amandi, A.: Learning styles' recognition in e-learning environments with feed-forward neural networks. *ISISTAN Research Institute, UNICEN University, Campus Universitario, Paraje Arroyo Seco, Tandil, Argentina* (2006)
11. Burgos, D., Moreno-ger, P., Sierra, J. L., Fernández-Manjón, B., Specht, M., Koper, R.: Building adaptive game-based learning resources: The marriage of IMS learning design and <e-adventure>. *Simul. Gaming* **39**, 414–431 (2008)
12. Jeetinder, S., Jayanthi, S.: Creating educational game by authoring simulations, *Proceedings of the 17th International Conference on Computers in Education [CDROM]*. Hong Kong: Asia-Pacific Society for Computers in Education (2009)
13. Weng, M.M., Fakinlede, I., Lin, F., Shih, T. K., Chang, M.: A conceptual design of multi-agent based personalized quiz game, *advanced learning technologies (ICALT)*, 11th IEEE international conference on, pp. 19–21 (2011)
14. Conati, C., Gertner, A., Vanlehn, K.: Using Bayesian networks to manage uncertainty in student modeling. *User Model. User-Adap. Inter.* **12**(4), 371–417 (2002)
15. Sabourin, J.L., Mott, B.W., Lester, J.C.: Modeling learner affect with theoretically grounded dynamic Bayesian networks. *Proceedings of the 4th international conference on affective computing and intelligent interaction*, pp. 286–295 (2008)
16. Li, Y., Musilek, P., Wyard-Scott, L.: Fuzzy logic in agent-based game design. *Proceedings of the 2004 annual meeting of the North American fuzzy information processing society*. Banff, Alberta, Canada (2004)
17. Kuk, K., Prokin, D., Dimic, G., Stanojevic, B.: Interactive tasks as a supplement to educational material in the field of programmable logic devices. *Electron. Electr. Eng.* **2** (98), 63–66 (2010)
18. Kuk, K., Prokin, D., Dimić, G., Spalević, P.: Learning unary logical operations through the modern interactive educational application—Arhicomp. In: *10th anniversary international scientific conference, UNITECH'10*, vol. 3, pp. 303–308. Gabrovo. Bulgaria (2010)
19. Tapscott, D.: *Growing up digital: The Rise of the Net Generation*. McGraw-Hill, NewYork (1998)

20. Felder, R.: Reaching the second tier: Learning and teaching styles in college science education. *J. Coll. Sci. Teach.* **23**(5), 286–290 (1993)
21. Norman, D.: Things that make us smart. Addison-Wesley, MA (1993)
22. Prensky, M.: Digital Game-Based Learning. McGraw-Hill, New York (2001)
23. Marlene, S.: The future: Gizmos. Technology and Children. *J. Elementary School Technol. Edu.* **13**(2) (2008)
24. Pitler, H., Hubbell, E., Kuhn, M., Malenoski, K.: Using technology with classroom instruction that work. ASC, Alexandria (2007)
25. Iqbal, S.T., Bailey, B.P.: Leveraging characteristics of task structure to predict the cost of interruption. CHI 2006 Proceedings: Using Knowledge to Predict and Manage. Montreal, Canada (2006)
26. Cox, C.T.: An investigation of the effects of interventions on problem solving strategies and abilities. (PhD, Clemson University) (2006)
27. Kuk, K., Rancic, D., Spalevic, P., Trajcevski, Z., Micalovic, M.: Use game based interactive multimedia modules to learning basic concepts on courses for computing science. *Przegląd Elektrotechniczny* **88**(5B), 150–153 (2012)
28. Parshall, C.G., Davey, T., Pashley, P.: Innovative item types for computerized testing. In: van der Linden, W.J., Glas, C.A.W. (eds.) Computerized adaptive testing: theory and practice. Kluwer Academic Publishers, The Netherlands (2000)
29. Kuk, K., Spalevic, P., Caric, M., Panic, S.: Game based learning module “Z-Buffer” on a course in computer graphics. In Proceedings YUInfo 2011/ICIST 2011. Kopaonik, 6–9 March (2011)
30. Hwang, G.J.: A conceptual map model for developing intelligent tutoring systems. *Comput. Educ.* **40**(3), 217–235 (2003)
31. Lord, F.M.: Applications of item response theory to practical testing problems. Lawrence Erlbaum Associates, Hillsdale (1980)
32. Owen, R.J.: A Bayesian sequential procedure for quantal response in the context of adaptive mental testing. *J. Am. Stat. Assoc.* **70**(350), 351–371 (1975)
33. Font, M.J., Manrique, D., Ríos, J.: Evolutionary construction and adaptation of intelligent systems. *Expert Syst. Appl.* **37**(12), 7711–7720 (2010)
34. Kuk, K., Milentijević, I., Rančić, D., Spalević, P.: Pedagogical agent in multimedia interactive modules for learning—MIMLE. *Expert Syst. Appl., Elsevier Sci.* **39**(9), 8051–8058 (2012)
35. Fang, W., Blank, G.D.: Student modeling with atomic bayesian networks. Paper presented at the 8th international conference on intelligent tutoring systems. Jhongli, Taiwan, June 26, 30 (2006)
36. Sevarac, Z.: Neuro fuzzy reasoner for student modeling. IEEE computer society, Washington (2006)
37. Kuk, K., Spalević, P., Ilić, S., Carić, M., Trajčevski, Z.: A model for student knowledge diagnosis through game learning environment. *Tech. Technol. Educ. Manage.—TTEM* **7**(1), 103–110 (2012)
38. Kuk, K., Ilic, S., Spalevic, P., Panic, S.: Student knowledge diagnosis in game-based learning applications. In: 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics (SISY), pp. 455–459 (2012)

E-Learning Paradigms and Applications

Agent-based Approach

Ivanović, M.; Jain, L.C. (Eds.)

2014, XVII, 273 p. 97 illus., Hardcover

ISBN: 978-3-642-41964-5