AN AGENT BASED INTELLIGENT TUTORING SYSTEM FOR PARAMETER PASSING IN JAVA PROGRAMMING

[1]Samy Abu Naser

[1]Associate Prof., Faculty of Engineering &Information Technology,

Al-Azhar University, Gaza, Palestine

E-mail: samy@abunasser.com

ABSTRACT

We have developed an agent based intelligent tutoring system for the parameter passing mechanisms in computer science (2), an introductory Java programming language, in Al-Azhar University in Gaza. The agent based intelligent tutoring system helps students better understand parameter passing mechanisms in Java using problem based technique. In this paper, we will describe the architectural design and features of the agent based intelligent tutoring system. An initial evaluation of effectiveness of the system was carried out and the result was found to be positive. The evaluation confirmed the established hypothesis that using the intelligent tutoring system would result in an improvement in the learning of the students [5]-[9].

Keywords: Intelligent tutoring system, Agent system, E-learning, Parameter passing, Java, Programming.

1. INTRODUCTION

An Intelligent Tutoring System (ITS) provides an individualized computer-based instruction to students [2]-[3]. These systems emerged from application of artificial intelligence techniques to the computer aided instruction (CAI) systems [4]. Parameter Passing mechanisms are the most important and most confusing part in Java programming. Our students have been introduced to parameter passing in C++ in computer science (1) at the Faculty of Engineering and Information technology in Al-Azhar University in Gaza. They learned that in C++, a parameter can be passed by: value, reference, or const-reference. Students in Computer Science (2), Java programming language, try to relate parameter passing in C++ to Java. The question that is being asked the most "Does Java pass objects by reference or by value? ". The answer is no! The fact is that Java has no facility whatsoever to pass an object to any function! The reason is that Java has no variables that contain objects. The reason there is so much confusion is students tend to blur the distinction between an object reference variable and an object instance. All object instances in Java are allocated on the heap and can only be accessed through object references [10]-[12]. So if we have the following:

StringBuffer **g** = new StringBuffer ("Action");

The variable **g** does not contain the string "Action", it contains a reference (or pointer) to an object instance that contains the string "Action".

So, if we then call f(g), f is free to modify its formal parameter to make it point to another StringBuffer or to set it to null. The function **f** could also modify the StringBuffer by appending "Films" for instance. While this changes the value of that StringBuffer, the value of that StringBuffer is not the value of the actual parameter **g**.

Imagine for instance if we set **g** to null before passing it to **f**. There is no StringBuffer now to modify and **f** can in no way change the value of **g** to be non-null.

The bottom line is Java only has variables that hold primitives or object references. Both are passed by value. So, what is pass-by-value means?

Pass-by-value means that when you call a method, a copy of the *value* of each actual parameter is passed to the method. You can change that copy inside the method, but this will have no effect on the actual parameter. Unlike many other languages, Java has no mechanism for changing the value of an actual parameter. This may seem very restrictive to most students. But in Java, we can pass a reference to an object (also called a "handle") as a parameter. We can then change something inside the object; we just can't change what object the handle refers to [10]-[12].

In this paper, we have designed an agent based intelligent tutoring system that covers: the following passing mechanisms:

- **Primitive types**: Java has eight primitive data types: six number types, character and Boolean,
- **Passing arrays**: Arrays are references. This means that when we pass an array as a parameter, we are passing its handle or reference. So, we can change the contents of the array inside the method.
- **Passing strings:** When we write code with objects of class string, it can look as if strings are primitive data types.
- **Passing object references:** What if a parameter to a method is an object reference? We can manipulate the object in any way, but we cannot make the reference refer to a different object.
- **Return values:** How do we change the values of variables inside methods?

2. ARCHITECTURE OF THE TUTOR

In this section we present the models and architecture for the agent based Intelligent Tutoring System. The Knowledge base design, Expert agent module design, Feedback agent module design and the user interface design.

Knowledge Base Design

In this section we describe the Knowledge Base architectural model for the agent based Intelligent Tutoring System. In the current paper we use a small subset of Java programming language to be tutored. We have concentrated on the mechanisms of parameter passing in Java.

For every problem stored in the knowledge base, we have stored the possible solution, some possible errors for specific categories and possible hints for each error.

The following figure shows a problem example with a solution and some errors.

Figure 1: *A problem example with a solution and a few errors.*

Problem: Suppose you are given the following Java program segment?

```
public static void tryPrim(int i, double f, char c,
boolean test)
 {
  i += 10;
  c = 'z';
  if(test)
    test = false;
  else
    test = true;
  f = 1.5;
 }
```

If tryPrim is called within the following code, what will the final print statement produce?

```
  int ii = 1;
  double ff = 1.0;
  char cc = 'a';
  boolean bb = false;
  tryPrim(ii, ff, cc, bb);
  System.out.println("ii = " + ii + ", ff = " + ff +
                     ", cc = " + cc + ", bb = " + bb);
```

The Solution which is authored by Instructor:

TryPrim (At the beginning of the method call)					
i	f	c	test		
1	1.0	a	false		

TryPrim (At the end of the method call)					
i	f	c	test		
11	1.5	z	true		

Output, after the return from TryPrim					
ii	ff	cc	bb		
1	1.0	a	false		

The student first possible error: (The value of i is wrong).

Output, after the return from TryPrim					
ii	ff	cc	bb		
11	1.5	z	true		

The student second possible error: (The value of f is wrong).

Output, after the return from TryPrim					
ii	ff	cc	bb		
11	1.5	z	true		

The student third possible error: (The value of c is wrong).

Output, after the return from TryPrim					
ii	ff	cc	bb		
11	1.5	z	true		

The student first possible error: (The value of test is wrong).

Output, after the return from TryPrim					
ii	ff	cc	bb		
11	1.5	z	true		

The problems stored in the knowledge base are classified into four levels of difficulties: easy, medium, difficult, and very difficult. The expert agent model determines which problem level should be given to the student.

In order for the student to solve a problem, he must first fill the table: tryPrim in the beginning of the method. Second, he must fill the table: tryPrim at the end of the called method. Third, he must fill the final output table: Output, after the return from TryPrim as shown if figure 1.

Usually textbooks do not present the steps required to solve a problem, but using visualizations and intelligent tutoring systems, student can learn and solve problems comfortably[1].

Expert Agent Module

The expert agent module determines the student level of understanding from the problem statement, the problem specification and student's answers. When the student reach a certain score answering at the current level of difficulty; for example 80 percent or more, the expert module increase the level of difficulties of problems to be given for the student. On the other hand, if the student did not reach a minimum score at certain level say 50 percent, the expert agent model branches the student to the tutorial on the subject involved in the problem. When the student finishes the tutorial, the expert agent module permits the student to go back to the questions mode. The score of the student, the level of difficulties and his name are all stored in a database for further analysis in the future when the student comes back and use the Tutor. So, the database reflects the actual level of every student as a result of all previous sessions.

Feedback Agent Module

Effectiveness of the system depends heavily upon its feedback timing and style. Timing refers to when the student is given a response to the solution. When the feedback is presented to the student should be governed by what the student have done. Tutors are better than teachers in this respect in that they can provide a student with timely feedback better than most teachers.

Using the example in figure 1, the following dialogue between the Tutor and the student would come up:

Student solution

Output, after the return from TryPrim					
ii	ff	cc	test		
11	1.5	z	true		

Tutor: "The value of ii is wrong. It should be 1; because the changes will not be seen outside tryPrim". Is that what you have thinking of?
Student: Yes
Tutor *makes the correction and proceeds with rest of the table.*

Student solution

Output, after the return from TryPrim					
ii	ff	cc	bb		
1	1.5	z	true		

Tutor: "The value of ff is wrong. It should be 1.0; because the changes will not be seen outside tryPrim". Is that what you have in mind?
Student: Yes
Tutor *makes the correction and proceeds with rest of the table.*

Student solution

Output, after the return from TryPrim					
ii	ff	cc	bb		
1	**1.0**	z	true		

Tutor: "The value of cc is wrong. It should be a; because the changes will not be seen outside tryPrim". Is that what you have in mind?
Student: Yes
Tutor *makes the correction and proceeds with rest of the table.*

Student solution

Output, after the return from TryPrim					
ii	ff	cc	bb		
1	**1.0**	a	true		

Tutor: "The value of bb is wrong. It should be false; because the changes will not be seen outside tryPrim". Is that what you have in mind?
Student: Yes
Tutor *makes the correction and proceeds with other tasks.*

User Interface

The interface of intelligent tutoring systems is a very important factor that we gave it a careful consideration during the design of the Tutor. The user interface is based on a presentation format implemented in many popular Integrated Development Environments used by professional programmers. Upon connecting to the Tutor website, the student's browser displays the working environment for the Tutor. An appropriate skill-level problem is selected by the expert agent module or the problem that last attempted is presented to the student.

The student solves the problem in the Student Solution window. When student press Check button, the expert agent module will determine the appropriate response based on the diagnosis of the student solution. The feedback of the expert agent module is sent to the student's output window.
The student, at any time, may explicitly request from the tutor to view the solution, Exit from the current problem and ask for a new one; furthermore, the student can view his performance based on statistics including problems attempted, problems solved, number of attempts on a problem and problem difficulty. The Tutor user interface is shown in figure 2.

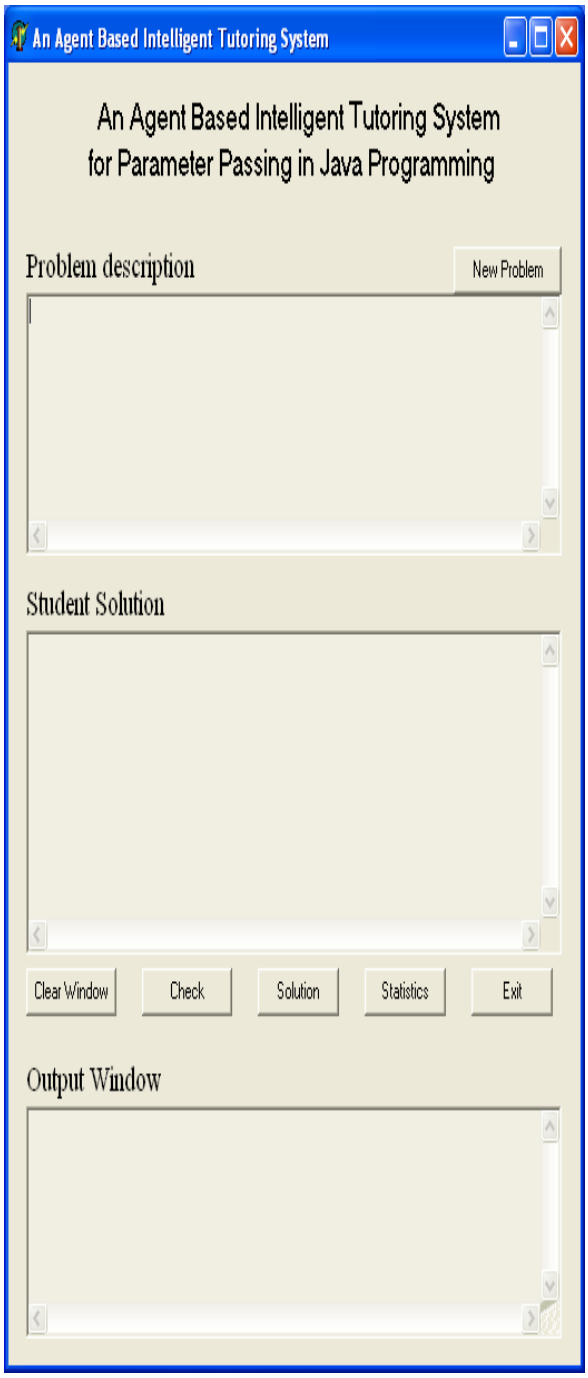

Figure2: Tutor User Interface

3. CONCLUSIONS

In this paper, we have presented recent developments related to the agent based Intelligent Tutoring System, which is based on sound theories, pattern recognition techniques, error detection and correction strategies.
An intial evalution shows that using the tutor enhanced the student performacnce and knowledge in the mechanisms of papameter passing in Java. A through evaluation is secduled over the next two semester to get feedback from the students to enable us improve the Tutor. This research is significant since it has the potential to be applied to many programming courses at the university level. Furthermore, it is being considered to be used for homework solution and quizzes. Since the expert module agent store all information about each student uses the tutor and the score he reached.

4. FUTURE WORK

Additional intelligent tutoring systems for Java programming should be invested to enable students overcome their difficulties that are faced in first programming languages.

5. REFERENCES

[1] S. S. Abu Naser, Developing Visualization Tool for Teaching AI Searching Algorithms, Informat techonology Journal, Vol. 7(2), 2008, pp. 350-355.

[2] S. S. Abu Naser, Intelligent Tutoring System (ITS) for Teaching Database to Sophomore Students in Gaza and its Effect on their Performance, Information techonology Journal, Vol.5(5), 2006,pp. 916-922.

[3] S. S. Abu Naser, A comparative study between Animated Intelligent Tutoring Systems "AITS" and Video-based Intelligent Tutoring Systems "VITS", Al-Aqsa University Journal, Vol. 5 No. 1 Part (1). 2001, pp. 72-96.

[4] S. S. Abu Naser, O. Sulisel, The Effect Of Using Computer Aided Instruction on Performance of 10th Grade Biology In Gaza", Journal of the college of education, Vol. 4 No. 1, 2000, pp. 9-37.

[5] J. R. Anderson, A. T. Corbett, K. R. Koedinger, & R. Pelletier, Cognitive Tutors: Lessons learned. The Journal of the Learning Sciences, 4, 1995, pp.167-207.

[6] B. Woolf,. P. Beck, J. Eliot,C. & Stern, M., Growth and maturity of intelligent tutoring systems, in K. D. Forbus & P. J. Feltovich (Eds.), Smart machines in education, (Cambridge, MA: MIT Press, 2001, pp.100-144).

[7] A. C. Graesser, N. K. Person, Teaching tactics and dialog in auto tutor, International Journal of Artificial Intelligence in Education, *12, 2001, pp.* 12-23.

[8] K. R. Koedinger, K. D. Forbus & P. J. Feltovich, Cognitive tutors in (Eds.), Smart machines in education, (Cambridge, MA: MIT Press), 2001, pp. 145-167.

[9] A. C. Scott, J. E. Clayton, & E. L. Gibson, A Practical Guide to Knowledge Acquisition (Menlo Park, CA: Addison-Wesley), 1991.

[10] D. Y. Liang, Introduction to Java Programming-Comprehensive Version (7th Edition) Prentice Hall, 2008.

[11] W. Thomas, An Introduction to Object-Oriented Programming with JAVA, 4/e Tata McGraw-Hill Publicatons, 2006.

[12] Horstmann, "Big Java", 2nd Edition, John Wiley & Sons, 2006.