Results from the Evaluation of the Effectiveness of an Online Tutor on Expression Evaluation

Amruth N Kumar
Ramapo College of New Jersey
505 Ramapo Valley Road
Mahwah, NJ 07430, USA

amruth AT ramapo.edu

ABSTRACT

Researchers have been developing online tutors for various disciplines, including Computer Science. Educators are increasingly using online tutors to supplement their courses. Are online tutors effective? Can they help students learn? If so, what features contribute to their effectiveness? We will examine these questions in the context of an online tutor that we developed for introductory Computer Science. The tutor is designed to help students learn expression evaluation in C++/Java.

We evaluated the tutor over several years, in multiple sections of Computer Science I each year. We used controlled tests with differential treatments, and used pre and post-tests to evaluate the effectiveness of the tutor. Our results show that online tutors indeed help students learn. Students who use the tutor for practice learn better than those who use a printed workbook. Students who receive both graphic visualization and text explanation learn better than those who receive only graphic visualization. Students who use graphic visualization learn better than those who receive no explanation. These results will be of interest to both developers and users of online tutors.

Categories and Subject Descriptors

K.3.1 [**Computing Milieux**]: Computer-Assisted Instruction

General Terms

Experimentation, Languages.

Keywords

Tutor, Programming, Expression, Evaluation, Feedback, Visualization, Computer Science I.

1. INTRODUCTION

The prevalent view of science education is constructivism, which maintains that information cannot simply be transferred from the teacher to the student. Instead the student must construct his or her own meaning [4]. The most productive way to help a student construct his/her own meaning is through scaffolding, where the teacher guides the student during problem-solving [23]. Online tutors are an attractive option for providing such one-on-one guidance.

Therefore, researchers have been developing online tutors for various disciplines, including Computer Science. Examples of tutors developed for Computer Science include JFLAP [21] for Automata Theory, Gateway Labs for discrete mathematics [3], and PILOT [6] for graph algorithms. These tutors engage the user in problem-solving activities, which are known to promote learning. Online tutors for introductory Computer Science courses include:

- Tutors wherein students fill out missing statements in a programming problem, and the tutor points out syntax errors and selected semantic and run-time errors. Examples include WebToTeach [2] (now TuringsCraft), InStep [20] and [24].
- Tutors that preprocess/annotate the student's program so that when the program is executed, either the state of the program is clarified or errors in the program are highlighted. Examples include CMeRun [10] and Expresso [12].
- Tutors that provide detailed feedback about the evaluation/execution of programs, so that they can be used as supplements to classroom instruction [7]. In this category, we have developed several tutors on introductory programming concepts (e.g., expressions [15], loops [9]), advanced concepts (e.g., pointers [16,17], classes [13]) and programming language design (e.g., scope [11, 18], parameter passing mechanisms [22]).

Are such online tutors effective? Do they help students learn? What features contribute to the effectiveness of tutors for Computer Science? To date, one study has shown that students who used the tutor spent less time asking teaching assistants for additional help [20]. Intelligent Tutoring Systems researchers have documented an improvement of one standard deviation through the use of problem-solving tutors in algebra, geometry and LISP [1]. We investigated these questions in the context of a tutor that we developed for expression evaluation in C++/Java.

We chose a tutor on expression evaluation for this study for two reasons: 1) All introductory Computer Science students must learn expression evaluation. Therefore, the sample sizes for experiments would be large. 2) It is easier to evaluate a students' knowledge of expression evaluation, and therefore, the

SIGCSE '04, February 23-27, 2005, St. Louis, MO, USA.

effectiveness of a tutor on expression evaluation. In Section 2, we will present our tutor on expression evaluation and its features. In Section 3, we will discuss the evaluation of the tutor, including the protocol and the results.

2. A TUTOR ON EXPRESSION EVALUATION

We developed an online tutor on expression evaluation to help students learn by solving problems. The tutor generates and presents problems to the student, obtains the student's answer, grades it, and provides detailed feedback about the correct answer. The tutor can be used in one of two modes: in tutoring mode, it generates a problem and demonstrates the step-by-step solution to the problem. In testing mode, it generates a problem, lets the user solve it and grades the user's solution. First developed in 1996, the tutor is currently in its fourth version. The work reported in this paper was carried out using version 3 (2001-02).

The tutor presents problems compiled by the instructor. It can also generate problem expressions randomly. So, it is capable of generating an infinite number of problems with which to practice - a student will never see the same problem twice. Therefore, the same tutor can be used for both tutoring and testing in a course. When used for assignments and tests, random generation of problems helps prevent plagiarism and cheating.

The student is expected to solve each expression one operator at a time. For each sub-expression, the student draws an underbrace across the sub-expression by clicking and dragging the mouse. In response, the tutor pops up a dialog box to enter the intermediate result. The student has the option to undo one or all the previous steps. Please see the left panel in Figure 1 for the problem and the incorrect answer entered by the student.

After the student enters the answer and submits it, the tutor grades the student's answer and provides detailed feedback about the correct answer. It provides two types of feedback:

- Graphic visualization, in which the answer is displayed using underbraces. Underbraces illustrate the order in which the operators in an expression are evaluated. This is intuitive and natural - similar to how a student might evaluate an expression on paper.
- Text explanation, in which each step in the evaluation of the expression is described in prose. The explanation of each step includes a description of the precedence and associativity of the operator, errors, if any in the evaluation of the operator, and finer points in the evaluation of the operator, such as integer division or the inadvisability of comparing real numbers.

Please see the right panel in Figure 1 for the graphic and text feedback provided by the tutor. The tutor provides animation controls for the student to step through the feedback. After reading the feedback, the student clicks on the 'Next Problem' button to go to the next problem and start all over again.

The tutor can be configured to provide feedback at several levels:

- No feedback – the user is just instructed to go to the next problem;
- Minimal feedback – the user is told whether his/her answer is correct.
- Detailed feedback – the user is not only told whether his/her answer is correct, but also shown graphic visualization and/or text explanation for the correct answer.

3. EVALUATION OF THE TUTOR

We evaluated the tutor (arithmetic and relational operators only) in multiple sections of our Computer Science I course from spring 2001 through fall 2002. We tested a different hypothesis each semester. We will describe the evaluation protocol in Section 3.1 and discuss the results in Section 3.2.

3.1 The Protocol

We used controlled tests to evaluate our tutors. We used a between-subjects design, randomly but evenly dividing each class into control and test groups. Further, we used the traditional pre-test, practice, post-test protocol. Both the control and test groups first answered the same pretest, then were exposed to differing treatments for practice learning, and finally came back together to answer the same post-test:

- During the pre-test, students answered a set of questions on the topic. Their score on the test reflected their level of preparation in the topic. The students were neither told whether their answers were correct/wrong, nor were they given any feedback during the test. Typically, the test lasted 6-10 minutes.
- During the practice, the test and control groups practiced with different treatments. Typically, the practice session lasted 10-12 minutes. Students did not have access to any other resource (such as text books, notes, consultation with the instructor or classmates) during the practice session.
- During the post-test, the students answered a new set of questions that were carefully matched with those on the pretest for the concepts they tested and the order in which they appeared. The change in the score from pretest to post-test reflected the effect of the tutor on the student's learning.

The whole procedure took 30-40 minutes. When analyzing the results, we calculated the average points per question rather than the total points in order to eliminate practice effect.

3.2 The Results

In spring 2001, we compared the tutor against textbooks – we tested the hypothesis that practicing with the tutor would be at least as effective as practicing with a printed workbook that included answers to problems as an appendix. The improvement in the score per attempted problem from pretest to post-test was 20.1% for workbook users (N=31) and 29.6% for tutor users (N=33), both the improvements being statistically significant (t-test 2-tailed $p < 0.05$). The effect size, calculated as (post-test average – pretest average) / standard deviation on the pretest was 1.27 for the tutor users versus 0.71 for the workbook users. Table 1 lists these figures.

Table 1: Printed Workbook Vs Tutor

Spring 2001	Pre-Test Score/Problem	Post-Test Score/Problem	% Change	Effect Size
Workbook Users (N=31)				
Average	4.35	5.23	20.10%	0.71
Std-Dev	1.23	1.03		
Tutor Users (N=33)				
Average	3.90	5.06	29.62%	1.27
Std-Dev	0.91	1.10		

In fall 2001, we compared two versions of the tutor – one that provided both graphic visualization and text explanation against one that provided only graphic visualization and no text explanation. Both the control and test groups used the tutor for practice, and both the groups viewed the visualization of the evaluation of the expression. But, only the test group received additional text explanation of the evaluation of the expression. We tested the hypothesis that students who received the text explanation would learn better than those who did not. The improvement in the score per attempted problem from pretest to post-test was 51.2% for the test group (N=33) and 39.8% for the control group (N=33), both improvements being statistically significant ($p < 0.05$). The effect size was 1.30 for those who received both graphic visualization and text explanation versus 1.00 for those who received only graphic visualization. Table 2 lists these figures.

In fall 2002, we compared two different versions of the tutor – one that provided graphic visualization (but no text explanation), versus one that did not provide any explanation except for the correct final answer. Our hypothesis was that those who received graphic visualization only would still learn better than those who received no explanation. The improvement in the score per attempted problem from pretest to post-test was 53.1% for the test group that received graphic visualization (N=24) and 33.4% for the control group that received no explanation (N=24), both improvements being statistically significant ($p < 0.05$). The effect size was 1.35 for those who received graphic visualization versus 0.84 for those who received no explanation. Table 3 lists these figures.

Table 2: Graphic Visualization with and without Text Explanation

Fall 2001	Pre-Test Score/Problem	Post-Test Score/Problem	% Change	Effect Size
Graphic Visualization (N=33)				
Average	3.44	4.80	39.80%	1.00
Std-Dev	1.36	1.05		
Graphic Visualization + Text Explanation (N=33)				
Average	3.12	4.72	51.22%	1.30
Std-Dev	1.23	1.26		

Table 3: No Explanation Vs Graphic Visualization

Fall 2002	Pre-Test Score/Problem	Post-Test Score/Problem	% Change	Effect Size
No Explanation (N=24)				
Average	3.06	4.08	33.43%	0.84
Std-Dev	1.22	1.51		
Graphic Visualization (N=24)				
Average	3.02	4.63	53.13%	1.35
Std-Dev	1.19	0.98		

3.3 Discussion

Students who used the tutor for practice learned better than those who used a printed workbook. We have replicated this result in several other studies [9,13,14]. Clearly, online tutors are at least as effective as textbooks, the traditional source of exercise problems. Online tutors have several advantages over textbooks:

1. Unlike textbooks, online tutors can generate an unlimited supply of problems, thereby providing for as much practice as desired by the learner.
2. Unlike textbooks, online tutors can instantaneously grade the learner's answer.
3. Unlike textbooks, online tutors can provide detailed feedback.

Therefore, online tutors such as ours may be used for learning, reinforcement, assessment, and self-assessment, both in and after class.

Students who received graphic visualization and text explanation learned better than those who received only graphic visualization. This confirms the earlier result in literature that in order to be effective, visualization must be extended with explanation [8,19].

Students who received only graphic visualization still learned better than those who received no explanation. This highlights the importance of providing explanation in online tutors. Other studies that we have conducted also support this conclusion [11,14,16].

The improvement in learning that we observed with our online tutor was in the range of 30-60% after accounting for practice effect. The effect size was over 1.25. This compares favorably with the result that one-on-one human tutoring can improve student learning by two effect sizes over normal classroom instruction [5].

4. ONGOING AND FUTURE WORK

Since testing our tutor, we have made several changes in keeping with the feedback we received from students and teachers who used the earlier versions of the tutor:

- Problem generation in the tutor is now adaptive – the tutor keeps track of the topics that the student has not yet mastered, and generates problems on only those topics.
- The tutor does not generate expressions randomly any more, since it is harder to control the learning outcomes of such expressions. Instead, the tutor now generates problems based on parameterized templates. Since a large number of problem instances can be generated from each template, the tutor is still capable of generating an endless supply of problems. Since the templates are indexed by learning outcomes, the generated problems are better matched to the needs of the learner.

We are currently extending the tutor to cover logical, assignment and bitwise operators. We plan to continue to evaluate the tutor. The tutor is available for general use. Access to the tutor can be obtained by contacting the author.

5. ACKNOWLEDGMENTS

Partial support for this work was provided by the National Science Foundation's Course, Curriculum and Laboratory Improvement Program under grant DUE-0088864 and the Combined Research and Curriculum Development and Educational Innovation Program under grant CNS-0426021.

6. REFERENCES

[1] Anderson J.R., Corbett A.T., Koedinger K.R. and Pelletier R., "Cognitive Tutors: Lessons Learned", The Journal of the Learning Sciences, Vol No 4(2), 1995, Lawrence Erlbaum Associates, Inc., pp 167-207.

[2] Arnow D. and Barshay, O., WebToTeach: An Interactive Focused Programming Exercise System, In proceedings of FIE 1999, San Juan, Puerto Rico (November 1999), Session 12a9.

[3] Baldwin, D. Three years experience with Gateway Labs. In Proceedings of ITiCSE '96 (Barcelona, Spain, June 1996), ACM Press, 6-7.

[4] M. Ben-Ari, Constructivism in Computer Science, Proceedings of 29th SIGCSE Technical Symposium, March 1998, 257-261.

[5] Bloom, B.S.: The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. Educational Researcher, Vol 13 (1984) 3-16.

[6] Bridgeman, S., Goodrich, M.T., Kobourov, S.G., and Tamassia, R. PILOT: An Interactive Tool for Learning and Grading. in Proceedings of SIGCSE '00 (Austin, TX, March 2000), ACM Press, 139-143.

[7] Brusilovsky, P. and Su, H.: Adaptive Visualization Component of a Distributed Web-Based Adaptive Educational System, Proceedings of the 6th international conference on Intelligent Tutoring Systems, June 2002, LNCS 2363, Springer Verlag, 229-238.

[8] Brusilovsky, P., Explanatory Visualization in an educational programming environment: connecting examples with general knowledge. In: B. Blumenthal, J. Gornostaev and C. Unger (eds.) Human Computer Interaction. LNCS 876. Berlin: Springer-Verlag, 202-212.

[9] Dancik, G. and Kumar, A.N., A Tutor for Counter-Controlled Loop Concepts and Its Evaluation, Proceedings of Frontiers in Education Conference (FIE 2003), Boulder, CO, 11/5-8/2003, Session T3C.

[10] Etheredge, J. CMeRun: Program Logic Debugging Courseware for CSi/2 Students. Proceedings of 35th SIGCSE Technical Symposium on Computer Science Education, March 2004, 22-25.

[11] Fernandes, E. and Kumar, A.: A Tutor on Scope for the Programming Languages Course, Proceedings of 35th SIGCSE Technical Symposium, Norfolk, VA, (March 2004), 90-95.

[12] Hristova, M., Misra, A., Rutter, M, and Mercuri, R. Identifying and Correcting Java Programming Errors for Introductory Computer Science Students. Proceedings of 34th SIGCSE Technical Symposium on Computer Science Education, February 2003, 153-156

[13] Kostadinov, R. and Kumar, A.N. A Tutor for Learning Encapsulation in C++ Classes, Proceedings of ED-MEDIA 2003 World Conference on Educational Multimedia, Hypermedia and Telecommunications}, Honolulu, HI, 6/23-28/2003, 1311-1314.

[14] Kumar, A.N., Learning Programming by Solving Problems, in Informatics Curricula and Teaching Methods, L. Cassel and R.A. Reis ed., Kluwer Academic Publishers, Norwell, MA, 2003, 29-39.

[15] Krishna, A., and Kumar A.: A Problem Generator to Learn Expression Evaluation in CS I and Its Effectiveness, The Journal of Computing in Small Colleges, Vol 16, No. 4, (May 2001), 34-43.

[16] Kumar, A.N., A Tutor for Using Dynamic Memory in C++, Proceedings of 2002 Frontiers in Education Conference (FIE 2002), Boston, MA, 11/6-9/2002, Session T4G.

[17] Kumar A. Learning the Interaction between Pointers and Scope in C++, Proceedings of The Sixth Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2001), Canterbury, UK, (June 2001), 45-48.

[18] Kumar A.N.: Dynamically Generating Problems on Static Scope, Proceedings of The Fifth Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2000), Helsinki, Finland, (July 2000), 9-12.

[19] Naps, T.L., Eagan, J.R. and Norton L.L. (2000) JHAVE – an environment to actively engage students in Web-based algorithm visualizations. Proceedings of 31st SIGCSE Technical Symposium on Computer Science Education, March 2000, 32(1), 109-113.

[20] Odekirk-Hash, E. and Zachary, J.L. Automated Feedback on Programs Means Students Need Less Help from Teachers. Proceedings of 32nd SIGCSE Technical Symposium on Computer Science Education, February 2001, 55-59

[21] Rodger, S., and Gramond, E., JFLAP: An Aid to Study Theorems in Automata Theory, Proceedings of ITiCSE 98, Dublin, Ireland, August 1998, 302.

[22] Shah, H. and Kumar, A.N., A Tutoring System for Parameter Passing in Programming Languages, Proceedings of The Seventh Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2002), Aarhus, Denmark, June 2002, 170-174

[23] Wood, David and Wood, Heather, Vygotsky, Tutoring and Learning, Oxford review of Education 22(1): 5-16, 1996

[24] Yoo, J.P., Seo, S.J. and Yoo, S.K. Designing an Adaptive Tutor for CS-I Laboratory. Proceedings of the 5th International Conference on Internet Computing, Las Vegas, NV, 2004.

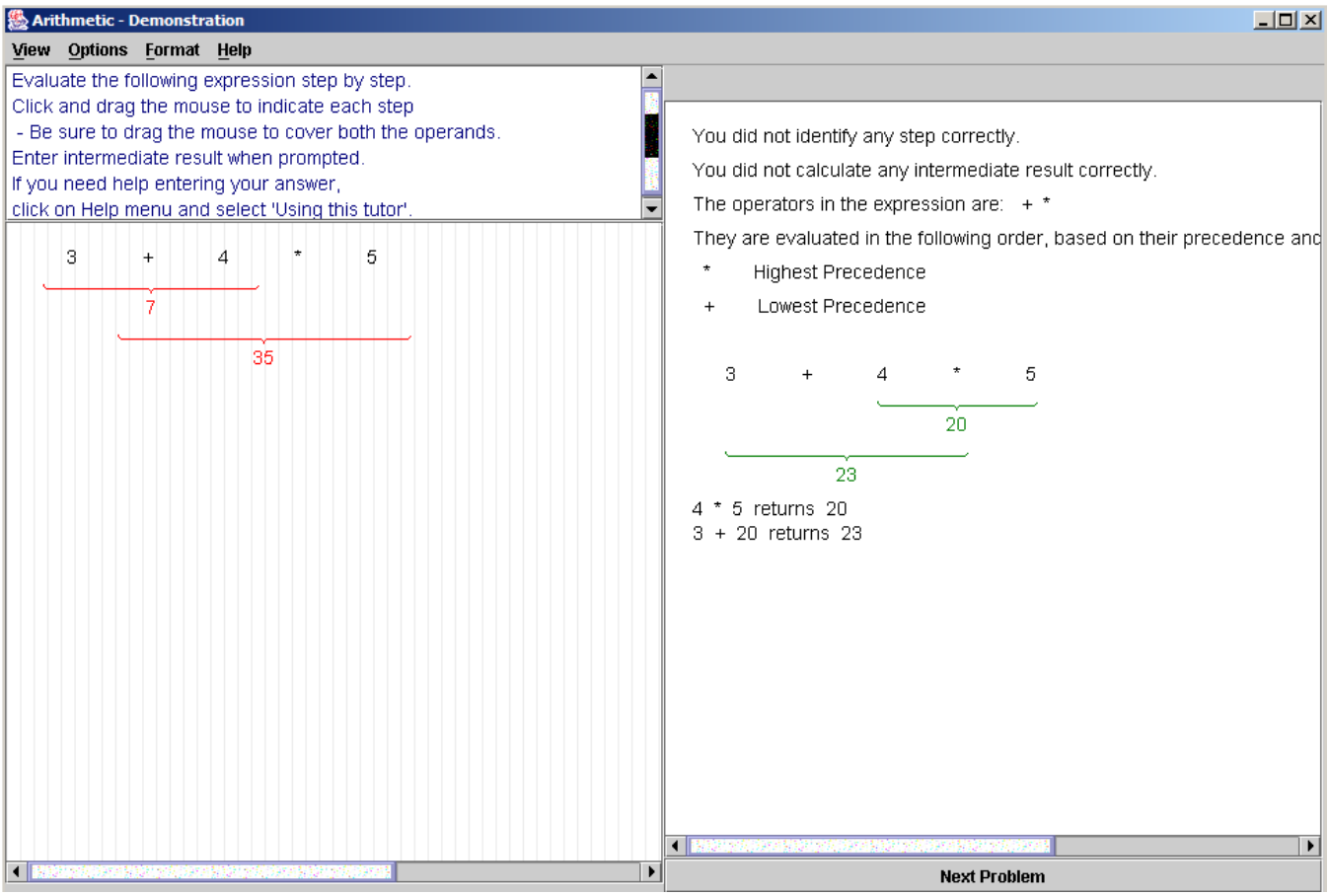

Figure 1: Snapshot of the Expression Evaluation Tutor (Version 4)