An In-Depth Exploration and Evaluation of Miss. Lindquist

Introduction and History

Intelligent tutoring systems (ITS) are computer programs that are designed to incorporate techniques from the Artificial Intelligence (AI) community in order to provide tutors which know what they teach, who they teach and how to teach it. AI attempts to produce a behavior in computer, which if performed by a human would be described as "good teaching" (Elsom-Cook, 1987). A current ITS typically has four components: expert knowledge module, student knowledge module, tutoring module, and user interface. (Nwana, 1990) On the opposite of constructivist theory which stresses "learning by doing" or "learning by design"(Duffy & Jonassen, 1992), ITS designers stress "learning by being told." (Michalski & Chilausky, 1980) Due to this nature of ITS, it shows certain advantages over traditional labs or discovery learning (Albacete, 2000) such as: clear articulation of knowledge, providing clear diagnosis of errors of students, showing how and why certain instructional techniques work or not. (Nwana, 1990) These features enable ITS to be a perfect test-bed for many theories and an ideal alternative of human tutors offering one-to-one tutoring, which was proven to be a very efficient way to attain higher achievements. In one research, 98% of students with private tutors performed better than those without private tutors. (Bloom, 1984).

Ms. Lindquist is a dialog based ITS. It is developed by Neil Heffernan and Ken Koedinger (2001) at Carnegie Mellon University. The goal of this software is to tutor students to write algebraic expressions from word questions. It initially asks the student

an algebra word question, in which there are several relations between several variables. If the student's answer is correct, the ITS will jump to the next question. If the answer is wrong, the ITS will restate the question in which some variables and relations are omitted, such that the question is shorter and simpler. The ITS also provides appropriate hints upon student's wrong answer. It is intelligent and looks like a human expert because the tutoring it provides is based on student's response.

The developer of Ms. Lindquist conducted a simple formative evaluation on the effectiveness of the software by giving 20 students a six-item posttest after 2 hours' use and asserted "Students using Ms. Lindquist did better on the post test". (Heffernan, 2001) This result is not strong enough to show the software's advantage over other computer based or traditional instructional technique. Besides, the sampling and instrument of this research seriously weakened the reliability and validity of the result.

Instructional software, like all other educational material, should be evaluated before it is used in the classroom or research laboratory. (Heller,1991) The purpose of this case study is to investigate the effectiveness of Ms. Lindquist by hands-on exploration, criticize on its advantages and weaknesses, and propose a redesign. This report is carried out towards the end of software development cycle.

Hands-on Exploration

In my case study, I asked Huong Huang to be the learner of this software. As of in February 2003, the software is migrated to a web-based platform. Questions and answers

are delivered via web browser with Java Applet technology. This new form of Ms. Lindquist is novel to us.

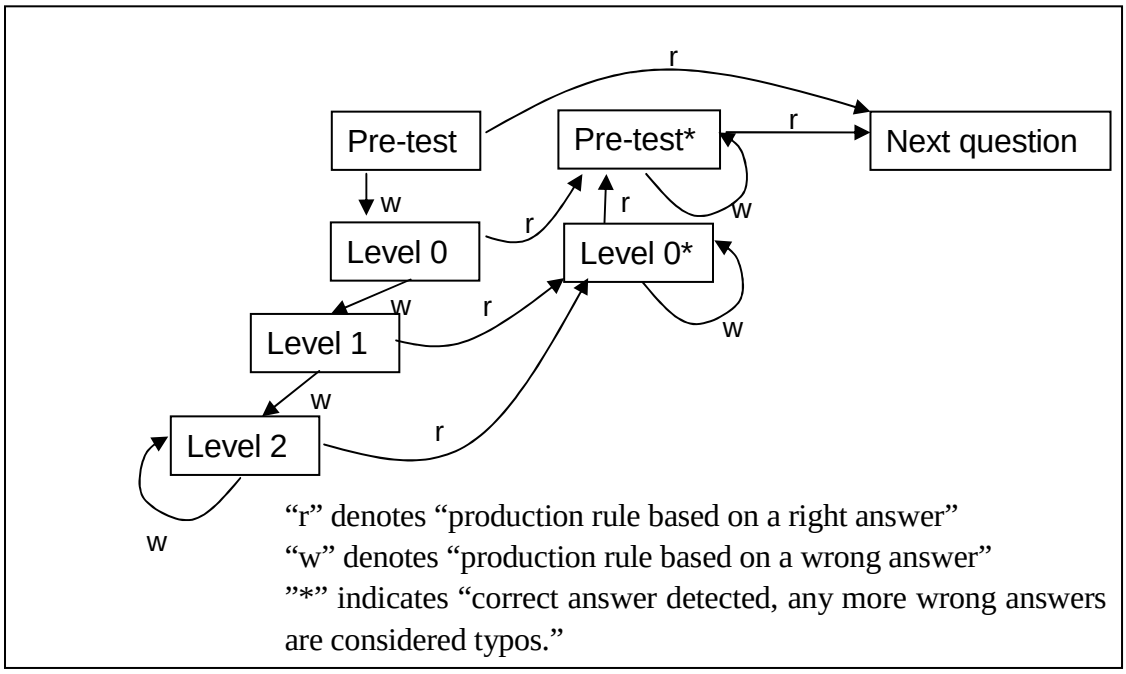
Ms. Lindquist consists of 1-operator questions and 2-operator questions. Initially, Ms. Lindquist asks students to answer 1-operator questions. If the answers are correct, advance to 2-operator questions, otherwise enter the tutoring dialogs.

Following are the tutoring dialogs of a 1-operator question:

- Level 0 (question): Each person at the party gets two scoops of ice cream. Write an expression for the number of scoops of ice cream needed if there are "p" people at the party. (Answer: 2*p)
- Level 1: In stead of asking students to type an expression, it asks students to select the first variable (i.e. total number of people, total scoops of ice cream, scoops of ice cream per person), the operator (i.e. plus, minus, times, divided by), and the second operator.
- Level 2: Students are still asked to make choices in 3 drop-down menus. However, this time Ms. Lindquist just simply tells the answer in a sentence: “The number of scoops of ice cream needed equals the number of people times the number of scoops per person.”

At any screen, if the student gives a wrong answer, then go forward to the next screen.

Level 2 screen is the farthest screen that a student can explore. If the answer is still wrong, then level 2 screen is looped until the student gives a correct answer. If the answer is correct, then Ms. Lindquist return to level 0 screen and assume that the tutoring process has finished. If the student makes any mistakes again, Ms. Lindquist won't provide any tutoring. It just considers it as a typo and shows the expression on the screen. Following is a screen transition diagram based on our experience of exploration:



The above diagram shows transitions of dialog screens of a 1-operator question. It is the basic construction of this software. Although 2-operator questions have more complex transition diagrams and much more production rules, their structure is just the combination of two 1-operator transition diagrams. Therefore, if the student makes a mistake in the first operator, he/she will enter the left sub-tree of the diagram; if the student makes a mistake in the second operator, he/she will enter the right sub-tree of the diagram. However, 2-operator tutoring structure and productions rules do require much more sophisticated design because following issues must be implemented properly:

1. Equivalence of expressions. i.e. x+y*z is equivalent to x+(y*z) or z*y+x
2. Errors in variables, not in operators. i.e. if in x+y*z, y is wrong, which production rule will be applied?

In our hands-on exploration, we found that Ms. Lindquist handles the first issue properly. The second issue is also nicely implemented by reminding the student of what that variable represents, if wrong again, then just telling the correct symbol the student should use. Since this tutoring in variables is just a linear process, it doesn't add too many entangling production rules into the already complex structure.

The software also keeps tracking and recording each student's performance by a unique user-name, so that students may retrieve their personal history from the database when they log-on later.

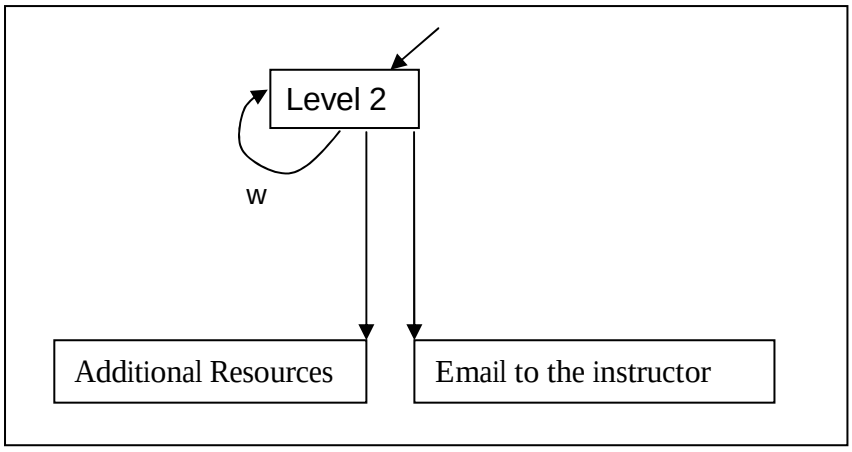
Critiques and Proposals for A Redesign

Generally, Ms. Lindquist has shown its effectiveness and efficiency in teaching algebra word questions (Heffernan, 2001). Comparing with other computer-based teaching, this ITS-based software is innovative and more effective. From hands-on explorations and the transition diagram above, I found that Anderson's model tracing design is lying beneath. Specifically, "there should be a production rule model of the underlying skill incorporated into the tutor. This is a model which would perform the task the student

was expected to perform. At each point in the problem solving the model is capable of generating a set of production sequences which represent correct solutions of the problem." (Anderson, et al. 1995) The software also provides sufficient feedback and help messages to guide students to the correct solution. Besides, the establishment of a student performance and history database is a very useful design.

However, I also propose following issues which may be improved in a redesign:

1. The off-path actions may be improved. Currently, if a student keeps giving wrong answers, Ms. Lindquist will lose its patience and keep looping the same screen (which states or shows the correct solution) at a certain level. The designer assumes that every student can understand this "easiest" level of screen. But what if a student just cannot understand it? Repeating the same information or telling the correct answer doesn't help a lot. Therefore, I propose that some additional resource information or actions (e.g. a web link or an auto-generated email sent to the instructor) may be provided in this screen.



2. The interface looks simple. Only standard Windows interface components (e.g.

text-field, drop-down menu, button) are used. Some graphics or animations may be added to support word statements so that questions, feedbacks, help information can be delivered in a more vivid and clear way.

Discussions and Conclusions

Building an ITS is not an easy job. It requires not only a lot of programming work (even for only one question), but also, more importantly, a comprehensive and precise design of production rule models. We must consider every possible action from students and provide proper feedback and tutoring information based on that. Continuous formative evaluation should be carried out so that the software's functionality and interface can be improved to achieve higher usability and usefulness.

References

Anderson, J. R., Corbett, A. T., Koedinger, K., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. The Journal of Learning Sciences, 4, 167-207.

Bloom, BS, (1984). The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring, Educational Researcher 4-16

Duffy, T. M., & Jonassen, D. H. (Eds.). (1992). Constructivism and the technology of instruction: A conversation. Hillsdale NJ: Erlbaum

Elsom-Cook, M. (1987) Intelligent Computer-aided instruction research at the Open University. Technical Report No: 63. Computer-Assisted Learning Research Group, The Open University, Milton Keynes.

Heffernan, N. T (2001) Intelligent Tutoring Systems are Forgotten the Tutor: Adding a Cognitive Model of Human Tutors. Dissertation. Computer Science Department, School of Computer Science, Carnegie Mellon University. Technical Report CMU-CS-01-127

Heller, R. S. (1991). Evaluating software: A review of the options, Computers & Education, 17 (4), 285-291

Mark M. A. & Greer J. E. (1993): Evaluation methodologies for intelligent tutoring systems. Journal of Artificial Intelligence and Education, 4 (2/3), 129-153.

Michalski R. S. and Chilausky R. L..(1980) Learning by being told and learning from examples: An experimental comparison of the two methodsofknowledge acquisition in the context of developing an expert system for soybean disease diagnosis. International Journal of Policy Analysis and Information Systems, 4(2)

Nwana H.S. (1990). "Intelligent Tutoring Systems: an overview ". Artificial Intelligence Review, 4, p. 251-277.

Patricia L. Albacete, Kurt A. VanLehn, (2000). Evaluating the Effectiveness of a Cognitive Tutor for Fundamental Physics Concepts, Proceedings of the 22nd Annual Meeting of the Cognitive Science Society