Development and Evaluation of an Intelligent Tutoring System for Teaching Natural Deduction

Kazuhisa Miwa [a*], Hitoshi Terai [a], Nana Kanzaki [a], and Ryuichi Nakaike[b]
[a] *Graduate School of Information Science, Nagoya University, Japan*
[b] *Graduate School of Education, Kyoto University, Japan*
*miwa@is.nagoya-u.ac.jp

Abstract: We present an intelligent tutoring system that teaches natural deduction to undergraduates. Our system was implemented on a client-server framework. An expert problem solver in the system provides basic instructional help, such as suggesting the use of a rule in the next step of solving a problem and indicating the inference drawn by applying the rule. Students learning with our tutoring system can vary the degree of help they receive (from low to high and vice versa). Empirical evaluation showed that the system enhanced the problem-solving performance of participants during the learning phase, and these performance gains were carried over to the post-test phase.

Keywords: Natural deduction, Help seeking, Client-server framework, Levels of support

1. Introduction

In this article, we present an intelligent tutoring system that teaches natural deduction (ND) to undergraduates. An ND system is a proof calculus in which logical reasoning is expressed by inference rules that are closely related to natural ways of reasoning (Barwise, 2003). Users of our tutoring system learn inference rules and strategies for applying the rules, such as strategies for inferring the proposition ¬Q => ¬P from the premise P => Q. ND is used in many universities' curricula for teaching the basics of logical thinking and formal reasoning. Figure 1 shows an example problem.

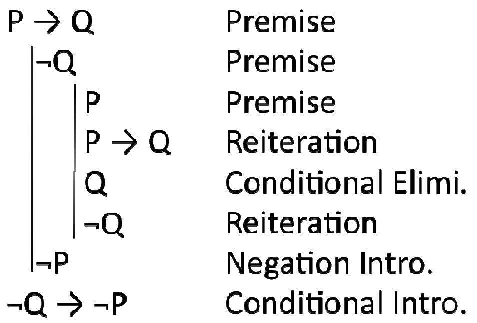

Figure 1: An example proof of contraposition from P => Q to ¬Q => ¬P.

Traditionally, ND has been taught using paper and pencil. The instructor explains the functions of the inference rules and presents example problems to students, and the students learn the inference rules as they solve the problems. However, problem solving in ND is relatively complex, because it involves selecting a single appropriate rule from various candidate rules, applying the rule to the appropriate propositions, and drawing the inference that results from the rule's application. Students often face serious difficulties because they cannot determine the correct rules to use. They may stop learning because they have no idea about how to proceed. They need the instructor's help, but in a traditional classroom setting, which may have more than 100 students, individual tutoring is sometimes impossible.

We use an intelligent tutoring system in this setting. The system provides basic instructional help, such as suggesting a possible rule for the next step when solving a problem and indicating the inference that results from the application of that rule. To provide this help, the system is equipped with technical knowledge and inference mechanisms. The system has been designed as a complete problem solver, and it acts like an ND expert.

The system performs as a complete problem solver as an ND expert, but it has neither knowledge about nor strategies for the instructions. The system is able to solve problems, and it tells students what to do next on the basis of its own attempt to solve the problem. However, the system does not know how to help students learn effectively, that is, it does not know when, how, or to what degree it should offer help. Hint presentation is an essential factor in intelligent tutoring systems (Koedinger & Aleven,2007). In our system, users are responsible for the instructional management. Although our system has no management functions for controlling the degree of help students receive, students learning with our tutoring system can vary the degree of help they receive (from low to high and vice versa). Our research question is whether the system's mode of direct instruction, that is, simply telling students what to do next, is effective compared to the traditional paper and pencil method for teaching relatively complex tasks such as performing ND.

2. The Tutoring System

Our tutoring system consists of an expert problem solver and tutor terminals. The system does not have a database containing a set of ND problems and their solutions. Rather, the expert problem solver solves each problem on demand. It is designed as a production system and consists of a working memory whose layout is consistent with the structure of ND problems and production rules corresponding to the inference rules and strategies for solving ND problems.

Our system was implemented on a client-server framework. Miwa and his colleagues developed a web-based production system architecture called "DoCoPro" that enabled the system design (Miwa et al., in press). The expert problem solver is implemented on the server and performs the complex ND inferences. Client computers for the tutor terminals that are connected to the server perform simple interface processing. User learning processes are saved as log data on the server. Using this client-server framework, our system can operate in any educational environment, where various types of computers, e. g., high spec, poorly performing, and those on different types of operating systems are used.

Students working at tutor terminals can control the levels of help that they receive. The LOSs (levels of support) can be controlled with respect to both rule selection and rule application.

<u>LOSs for rule selection</u>:

- Level 3: The system presents applicable candidate rules and strategies and indicates the propositions to which those should be applied.
- Level 2: The system presents applicable candidate inference rules and strategies.
- Level 1: The system only presents a set of inference rules and strategies (no assistance).

<u>LOSs for rule application</u>:

- Level 2: The system generates a proposition that was inferred automatically.
- Level 1: The system presents partial information about an inferred proposition. Users provide the terms for the complete proposition that is to be inferred.

Figure 2 shows an example screenshot of the tutoring system. The system provides users with lists of inference rules and strategies. Once, the user selects one of the rules or strategies from the lists, then the system automatically runs the rule. The system presents the complete inference result or a template in which the result is partially blanked. The system supports students by offering help about the rule and strategy selection.

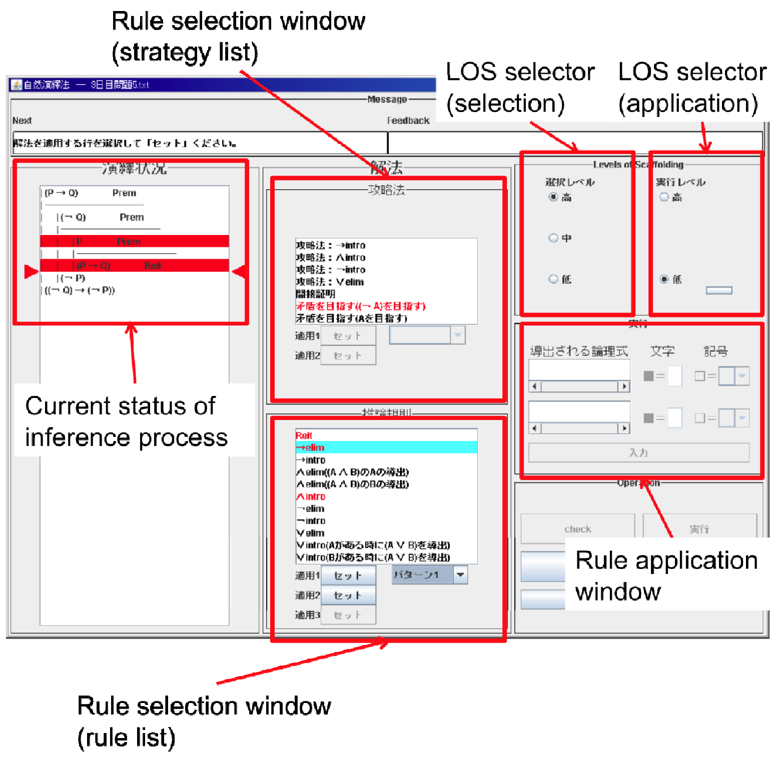
Rule selection window (strategy list)
LOS selector (selection)
LOS selector (application)
Current status of inference process
Rule application window
Rule selection window (rule list)

Figure 2: An example screenshot.

3. Evaluation

We verified that the system can successfully solve all 35 problems that are contained in a representative textbook (Todayama, 2000) that is used for ND education in Japan.

Our system is currently being used for teaching ND in university classes run by the first author. A class of 29 undergraduates in fall semester (2010) and 49 undergraduates in spring semester (2011) learned ND using our system, successfully acquiring the necessary skills for constructing ND proofs.

We designed and conducted an experiment to evaluate our system's usefulness. The purposes of the experiment were to verify whether learning with our tutoring system is an improvement over learning in the traditional paper and pencil manner. A total of 49 participants participated in the experiment. Of these, 33 were assigned to the experimental group and the other 16 to a control group.

In the initial phase of the experiment, the participants in both groups learned the basics of ND with handout materials and an instructional video. Next, in the 80-min learning phase, the participants in the experimental group learned ND by solving six example problems with our tutoring system. At the same time, the participants in the control group learned ND by solving the same six problems using handout materials, without the support of our tutoring system; the handout materials contain complete information about how to solve the problems. After the learning phase, the participants in both groups solved two post-test problems printed on the test sheets. One was an easy problem that requires a first-order subproof and the other was a difficult problem that requires a second-order subproof.

We analyzed the post-test scores in order to answer our first research question. In the learning phase, the participants in the experimental group solved an average of 5.21 example problems, but the participants in the control group solved only an average of 2.69 problems. The statistical analysis shows that the difference between the two groups is significant ($t(47) = 5.71$, $p < 0.01$), which means that the tutoring system successfully enhanced the participants' problem-solving performances during the learning phase. A crucial question is whether the performance gain was carried over to the post-test scores, where the tutoring system's assistance was not available.

Figure 3 shows the result of the post-test. The statistical analysis shows that the average score for solving the easy problem was higher in the experimental group than in the control group ($t(47) = 2.59$, $p < 0.05$), but the average score for solving the difficult problem was not ($t(47) = 1.42$, n.s.). This result shows an enhanced performance of those who learned with our tutoring system, but only for solving the easy problem.

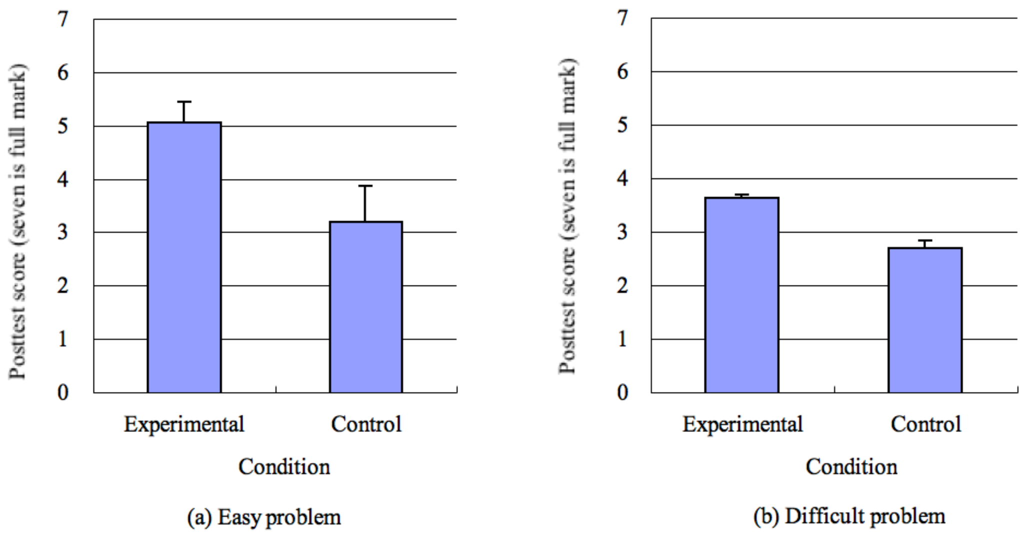

Figure 3: Average score of the post-test for the experimental and control groups.

4. Discussion and conclusions

Our research question concerned the usefulness of our tutoring system. Our results show that the participants in the control group solved only less than half of the problems within the learning phase. This means that they faced serious impasses during this phase. In contrast, the participants in the experimental group solved almost all the problems, which means that the system successfully eliminated the impasses. Although many preceding studies have indicated that learners behave irrationally in seeking help (Wood & Wood, 1999; Aleven & Koedinger, 2000), our overall results implies that the participants adaptively managed the LOSs while using our tutoring system.

The advantage of learning with the tutoring system was carried over only to solving the easy problem in the post-test; the enhanced performance did not reach a statistically significant level for the difficult problem. This may indicate a limitation of the system's direct mode of instruction, such as simply indicating the next step to be performed. An important part of our future research will involve addressing this limitation.

Other tutoring systems exist for teaching ND. One is a system called “Fitch” that provides students with templates for reasoning and automatically performs the reasoning (Barwise, 2003). However, this system does not provide information about which rule should be applied for each phase of the reasoning. As we mentioned above, instructional assistance in rule selection is more important for introductory students than in rule application, and so this is a critical limitation.

Croy et al. (2007) developed an intelligent tutoring system for teaching how to construct propositional proofs. To identify the best assistance for a specific stage of reasoning, they drew on previously acquired log data for students’ reasoning processes and used a Markov Decision Model to infer the next step in reasoning. In this way, they implemented a “Hint Factory” and were able to successfully improve introductory students’ learning in deductive logic courses. When our system provides help, it sometimes narrows the possible rules or strategies to a few candidates rather than a single one, because of the limitations of its inference abilities. Compared to our tutor, Croy et al.’s tutor appears to always provide one best candidate rule for the next step in reasoning. However, our system can handle new problems that have not been solved in advance. Croy’s system must accumulate training data for the Markov Model, and accumulating such data is time consuming, even though the researchers have tried to shorten the preliminary trials (Stamper, 2010). We have begun to expand our system for learning through problem posing, in which students generate their own problems and test the validity of the original problems by having our tutoring system solve those problems.

References

[1] Aleven, V., & Koedinger, K. R. (2000). Limitations of Student Control: Do Student Know when they need help? Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000. pp. 292-303.

[2] Barwise, J., & Etchemendy, J. (2003). Language, Proof and Logic. CSLI Publications.

[3] Croy, M., Barnes, T., & Stamper, J. (2007). Towards an Intelligent Tutoring System for propositional proof construction. Proceedings of European Computing and Philosophy Conference, 145-155.

[4] Koedinger, K. R., & Aleven, V. (2007). Exploring the Assistance Dilemma in Experiments with Cognitive Tutors. Educational Psychology Review, 19, 239-264.

[5] Miwa, K., Morita, J., Nakaike, R., & Terai, H. in press. Learning through Intermediate Problems in Creating Cognitive Models. Interactive Learning Environments.

[6] Stamper, J., Barnes, T., & Croy, M. (2010). Enhancing the Automatic Generation of Hints with Expert Seeding. Proceedings of the 10th International Conference on Intelligent Tutoring Systems (ITS 2010), 31-40.

[7] Todayama, K. (2000). Learning Logic through Building it. Nagoya University Publisher.

[8] Wood, H., & Wood, D. (1999). Help seeking, learning and contingent tutoring. Computers and Education, 33, 153-169.