Strategic Interface Design Can Improve Learning Efficiency in an Intelligent Tutoring System

Sutapa Dey Tithi[*]
North Carolina State University
stithi@ncsu.edu

Behrooz Mostafavi
North Carolina State University
bzmostaf@ncsu.edu

Arun Kumar Ramesh
North Carolina State University
arames25@ncsu.edu

Tiffany Barnes
North Carolina State University
tmbarnes@ncsu.edu

ABSTRACT

Educational data mining can help us discover patterns in data that can be used to help improve learning environments and assess those improvements. Here, we discovered that students were spending a significant amount of time in a tutoring system typing in newly derived problem-solving statements (about 12% of students' total time in the tutor). We designed a new problem-solving interface to allow students to select from among auto-generated multiple-choice statements and performed a between-groups study to compare students' problem-solving time and learning with and without access to auto-generated multiple-choice statements, between two different semesters. Our results showed significant time savings, with students in the control group taking 4.3 hours in total and the intervention taking 2.07 hours. These substantial time savings did not significantly impact the efficiency of student solutions (both groups had similar solution lengths) and only marginally reduced rule application accuracy on the posttest. This led to a significant difference in learning efficiency between conditions since students learned a similar amount in both groups but saved substantial time in the intervention.

Keywords

Intelligent Tutoring System, Learning Efficiency, Cognitive Load, Human-Computer Interaction

1. INTRODUCTION

Intelligent Tutoring Systems (ITS) can be a scalable alternative to human tutoring and offer user-adaptive instruction and feedback with minimal human teacher intervention [2, 9]. An effective ITS can help reduce the time for learning tasks, making the learning process more efficient.

In this study, we redesigned a section of Deep Thought (DT), an intelligent logic tutor, and explored the impact of interface design changes as a strategic intervention to reduce the training time for students. The updated interface offered students the option to choose from auto-generated multiple-choice statements[1], providing an alternative to typing solutions exclusively. We evaluated the efficacy the new interface design by answering the following research questions:

- ***RQ1 (Total training time):*** Do the students who were given multiple-choice options (MCs) to derive new statements **complete the tutor in less time** than students without MCs?
- ***RQ2 (Learning gain):*** Do students using the MCs **have similar or better learning gains** than students without MCs?

2. RELATED WORKS

Interface design in ITSs is an important element that influences students' effective learning experience as they act as a medium of communication between the learners and the learning materials [1]. An easy-to-use, engaging, and realistic interface motivates learners, whereas a poorly designed interface does not motivate students to make use of the product or learn with it [3]. Sweller et al. [8] suggest that instructional design should consider managing cognitive load to optimize learning outcomes. They discussed three primary types of cognitive load. The first, intrinsic load, pertains to the inherent difficulty of the material and may vary based on a student's prior knowledge. The second, extraneous load, concerns how information is presented and the ease with which a student comprehends it. The third, germane load, involves how we integrate and process new information into long-term memory. Since intrinsic load cannot be easily altered, it is important to design instructions to reduce the extraneous load and shift learners' attention

[*]First Author (led and carried out most of the work)

[1]Note that the new interface provides the option to choose between the multiple-choice and typing for each step derivation. The purpose of multiple choice was not to remove all typing; rather, it meant to demonstrate contextual answers to the prompt so students would understand how to answer.

S. D. Tithi, B. Mostafavi, A. K. Ramesh, and T. Barnes. Strategic interface design can improve learning efficiency in an intelligent tutoring system. In B. Paaßen and C. D. Epp, editors, *Proceedings of the 17th International Conference on Educational Data Mining*, pages 722–726, Atlanta, Georgia, USA, July 2024. International Educational Data Mining Society.

to cognitive processes directly relevant to learning. Worked examples (WE) reduce learners' extraneous load [6]. Some tutoring systems, including ours, offer WE problems along with PS (problem-solving) problems. Research studies have shown that worked examples can save students time without reducing their learning [5].

3. METHOD

3.1 Context: DT, the Intelligent Logic Tutor

DT is an intelligent logic tutor that teaches propositional logic proofs by applying valid inference rules. The interface of our tutor (DT) is shown in Figure 1a. In the tutor, students practice solving logic problems divided into four sections: introduction, pretest, training, and posttest. The introduction presents two worked examples to familiarize students with the tutor interface. Next, there is a pretest section, which is used to determine students' incoming competence. The pretest section is followed by the training section, which consists of five training levels with gradually increasing difficulty. Each training level consists of four problems, and the tutor has an expert-designed pedagogical policy that chooses whether students should solve the problem (PS) or see it as a worked example (WE). The last problem in each training level, is a level-end test problem (PS) with no access to help. Finally, students take a more difficult posttest (level 7) with 6 problems.

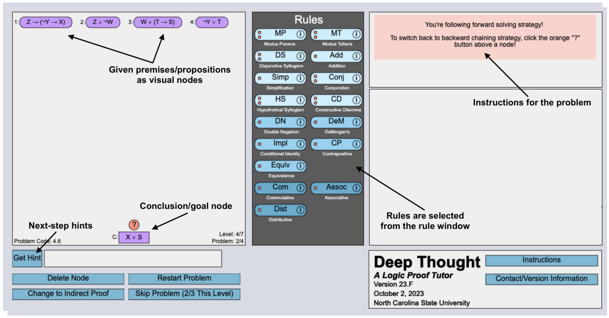
3.2 Experiment Design

We implemented two training conditions for our experiment: S22 and F23. In **condition S22**, after clicking the source nodes and a rule button, the tutor asks the student to type the resultant or derived statement in a black pop-up textbox as shown in Figure 1b. The tutor then verifies the step, and the new statement appears if the step is correct. In **condition F23**, students select the source node(s), and the tutor highlights the potentially applicable rules (only in the training phase). Next, students select a rule to apply to the source node(s). Based on the selected source node(s) and a rule, the tutor presents auto-generated potential resultant statements in the form of multiple-choice options as shown in Figure 1c, and the student chooses a solution from the provided options.

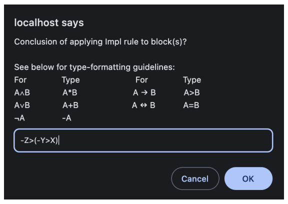
In Spring 2022 (S22) and Fall 2023 (F23), we deployed Deep Thought with 50 students each semester in an undergraduate discrete mathematics course at a large public university in the United States. This course required students to engage with Deep Thought as part of a lab assignment. While we do not have course-specific demographics, it's noteworthy that all CS majors are required to undertake discrete math. Therefore, we report the demographics of the 2020-21 graduating class of CS majors for an approximation, and this includes 21% women, 56% white, 17% Asian, 10% Non-resident Alien, 3% Black/African American, 5% Hispanic/Latinx, 9% other races. Participant consent was taken, and only the IRB-authorized researchers can access the data collected from the participants.

4. RESULTS

In F23, students had the option to either type their solution or utilize the multiple-choice window. Among the total



(a) Full Interface of Deep Thought



localhost says

Conclusion of applying Impl rule to block(s)?

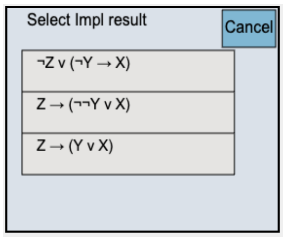
See below for type-formatting guidelines:

For	Type	For	Type
A∧B	A*B	A → B	A>B
A∨B	A+B	A ↔ B	A=B
¬A	-A		

-Z>(-Y>X)

Cancel OK

(b) S22 Typing interface



Select Impl result

Cancel

¬Z ∨ (¬Y → X)

Z → (¬¬Y ∨ X)

Z → (Y ∨ X)

(c) F23 Chooser interface

Figure 1: The full interface of DT with the comparison of the interfaces used for students to type in or choose the newly derived statement.

number of interactions to choose or type the new statement, **F23 students opted for the multiple-choice window 91.8% of the times**, implying that students overwhelmingly opted for the multiple-choice window. A Mann-Whitney U test was performed on the interaction log's data to evaluate whether their pretest performance differed by two conditions. There was no significant difference in the pretest performance ($U = 1271.50,\ p = 0.88$) between condition S22 ($Mean = 0.69,\ SD = 0.23$) and condition F23 ($Mean = 0.71,\ SD = 0.25$). We focus on students' performance in the problems where no tutor help was given: training level-end test problems and posttest problems in level 7.

4.1 RQ1: Total Training Time

Each student in F23 spent 21.6 minutes on average to select or type new expressions, whereas each student in S22 spent 32.8 minutes on average to type new expressions in the tutor. On average, each student in F23 spent **11 minutes** less to derive new statements than that in S22. Overall, the students in F23 completed the tutor in significantly less time than those in S22 ($U = 949359.00,\ p < 0.001$). The average time to complete the tutor was reduced from 4 hours 18 minutes (F23) to 2 hours 4 minutes (S22). This substantial reduction is a huge achievement in improving the tutor and the student's learning experience.

In Deep Thought, a "step" refers to deriving a new node in the proof, and a step may involve multiple actions (granular interactions with the system, such as selecting or deselecting a node or a rule). We found no significant difference in average step completion time in the pretest between conditions F23 and S22 ($U = 1139.00,\ p = 0.44$). In the level-end test problems and final posttest problems, **the students from F23 had significantly lower average step completion time** ($U = 31295.00, p < 0.001$) **than the students from S22**.

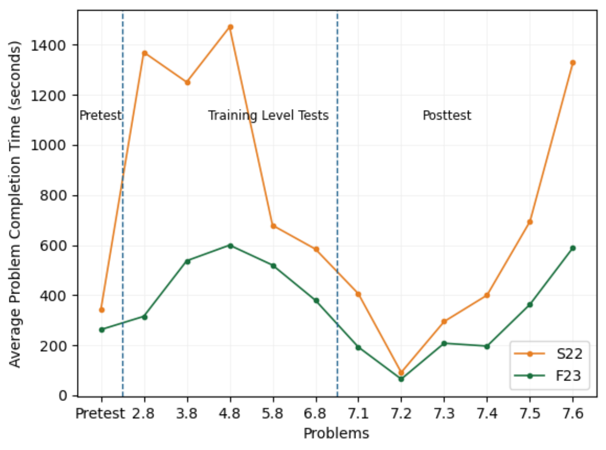


Figure 2: Trends in Average Problem-Solving Time in Pretest, Training Level Tests, and Posttest (7.1-7.6) PS Problems across the Two Training Groups

As shown in Figure 2 the average problem-solving time in F23 is also lower than in S22. For the level-end test and the post-test problems, the students from F23 took significantly less time ($U = 949359.00,\ p < 0.001$) than those from S22. Figure 3 shows the training time in F23 is significantly lower than the training time in S22 ($U = 258301.00,\ p < 0.001$).

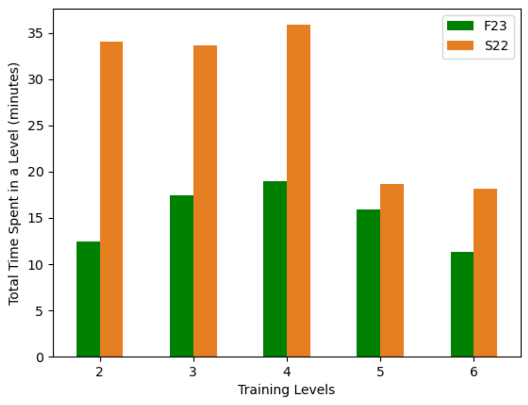


Figure 3: Total Time Spent to Complete Each Training Levels across Two Conditions

4.2 RQ2: Learning Gain

To answer our second research question on whether students in F23 have a similar or better learning gain than students in S22, we investigate their problem scores, rule application accuracy, solution length, and learning gain in the following sections.

4.2.1 Problem Score

The problem score equally weights high rule accuracy, lower time, and lower solution size to reflect learning of rules (accuracy) and efficient proof strategies (lower solution size) in shorter times (reflecting developed expertise). We also analyze each metric separately to provide a detailed understanding of student's performance and learning.

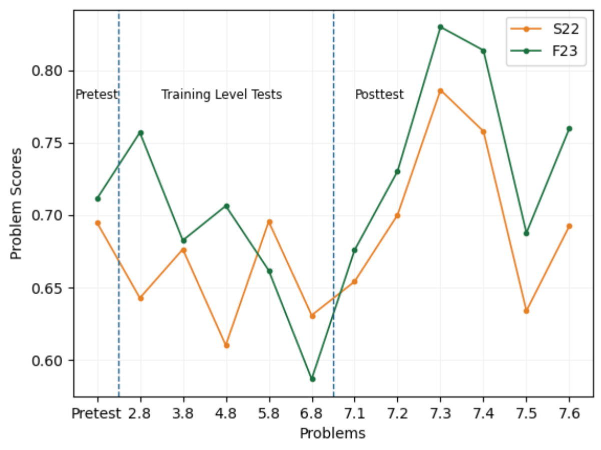


Figure 4: Problem Score (mean) in Training-level Test and Posttest Problems across Groups

$$\text{Problem Score} = \frac{1}{3} * \text{Accuracy} + \frac{1}{3} * \text{Size} + \frac{1}{3} * \text{Time} \quad (1)$$

Problem scores across the groups in the pretest were similar, and our statistical tests found no significant differences in this phase. In the posttest problems, the problem scores for the students in F23 ($Mean = 0.75,\ SD = 0.22$) were significantly higher ($U = 50518.00, p = 0.009$) than the problem scores for the students in S22 ($Mean = 0.70,\ SD = 0.22$) (Figure 4).

Upon analyzing the rule application accuracy, which depended on students' knowledge of domain rules, we found no significant differences between the two training conditions in training level-end problems. In the posttest problems, the students in S22 had a marginally higher rule accuracy than the students in F23 ($U = 41025.50,\ p = 0.06$).

Logic proofs within DT are open-ended and may have multiple solutions with different lengths. In F23, the multiple-choice window presents the solution options that are all logically correct but may not lead to the most optimal solution. A significantly lengthier solution may indicate that the students may use the multiple-choice window to derive random new statements or all possible new statements and complete the solution. A Mann-Whitney U test indicates that the solution lengths for the posttest problems between the two conditions are not significantly different ($U = 41275.50,\ p = 0.07$), dismissing the possibility that students in F23 randomly selected solutions from the multiple-choice window.

We categorized students with pretest scores below the 25th percentile as low scorers, students with pretest scores between the 25th and 75th percentile as medium scorers, and students above the 75th percentile as high scorers. As shown in Table 1, using the multiple-choice window saved time for all three pretest score groups. The low score group from

Table 1: Moderation Analysis with Posttest Performance across Two Training Conditions Categorized on Pretest Scores. [Note: **Blue**[*] indicates a significant difference, and **bold** indicates marginally significant difference.]

Learning Group	Training Group	Posttest Score Mean (SD)	Rule Accuracy Mean (SD)	Problem Time Mean (SD)	Solution Length Mean (SD)
Low Scorers (<25th percentile) (N=26)	F23	**0.712 (0.247)**[*]	0.65 (0.25)	**278.766 (374.901)**[*]	**8.346 (3.099)**[*]
	S22	0.601 (0.224)	**0.742 (0.161)**[*]	758.017 (1212.188)	10.192 (5.348)
Medium Scorers (25th-75th percentile) (N=46)	F23	0.74 (0.223)	0.693 (0.245)	**265.087 (384.897)**[*]	8.37 (3.443)
	S22	0.699 (0.227)	0.742 (0.179)	488.956 (728.04)	8.891 (3.6)
High Scorers (>75th percentile) (N=28)	F23	.801 (0.186)	0.779 (0.224)	**267.485 (439.143)**	7.869 (2.936)
	S22	0.808 (0.189)	0.789 (0.17)	407.095 (725.333)	7.464 (2.709)

F23 got the most benefits with much better posttest scores, shorter proofs, and much less problem completion time (although with a slightly lower rule accuracy) compared to the low score group in S22.

4.2.2 *NLG and Learning Efficiency*

Normalized learning gain is calculated using the average problem scores on the pre- and posttest problems with the equation [7]:

$$\text{NLG} = \frac{(\text{posttest score} - \text{pretest score})}{\sqrt{(1 - \text{pretest score})}} \tag{2}$$

where, NLG is scaled between 0 and 1. We define Learning Efficiency (LE) by the following equation:

$$\text{LE} = \frac{\text{NLG}}{\text{Tutor completion time}} \tag{3}$$

The 12 students who had perfect pretest scores were discarded from our study as a perfect pretest score implies a strong understanding of concepts already; therefore, learning gain may not be relevant to them. We found no significant difference in the average NLG between students from F23 and S22 ($U = 1141.0$, $p = 0.14$). As shown in Table 2, students from F23 had **significantly higher LE** than those from S22 ($U = 1791.00$, $p < 0.001$). In Figure 5, we observed that the mean LE for F23 exceeds that of S22 and that the LEs for F23 exhibit a shift towards higher positive values compared to S22.

Table 2: Normalized Learning Gain (NLG) across the Two Training Groups [Note: **Blue**[*] indicates a significant difference.]

Group (N)	Pretest Score Mean (SD)	Posttest Score Mean (SD)	NLG Mean (SD)	LE Mean (SD)
F23 (42)	0.66 (0.21)	**0.74 (0.12)**[*]	-0.016 (0.60)	**0.34 (0.25)**[*]
S22 (46)	0.66 (0.25)	0.69 (0.14)	-0.107 (0.46)	0.06 (0.08)

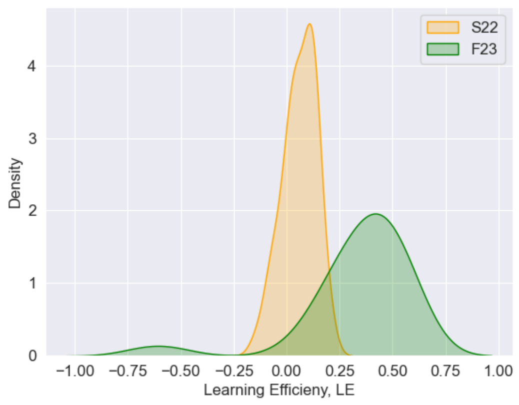


Figure 5: Distribution of Learning Efficiency across Two Conditions

5. CONCLUSION

Overall, our analysis showed that students choosing a derived statement from multiple choice options (F23) took less than half the time to complete the tutor than those required to type the derived expressions (S22). The reduced training time and improved learning efficiency may suggest a reduced extrinsic cognitive load for the students in F23 since they could see the probable solutions as options in the multiple-choice window and did not need to think about the syntax or symbols expected by the tutor.

Students in F23 had a marginal decrease in rule application accuracy. We investigated the scores for individual rules further, and students in F23 had similar or higher scores for all the other rules except MP, MT, and DS. Interestingly, these rules have only a single solution and don't require typing or selecting any solution in either version of the tutor. Thus, we could not relate the use of a multiple-choice window with the decrease in scores in these rules. Additionally, we can also introduce rule application practice early in the tutor and fade it as rule accuracy improves.

One limitation in our study is that due to IRB restrictions, we are unable to access any grades outside the tutor and measure long-term learning impacts. However, prior research suggests that tutor completion leads to good scores on a delayed post-test [4]. A second limitation is there may be uncontrolled differences between student groups from the two semesters. However, there was no significant difference in the pretest performance between groups. A third limitation is the study explores design changes to reduce cognitive load in only one tutor. However, this study suggests ways that other systems could be analyzed using data mining techniques to identify time and interactions that may be sources of extraneous load.

6. ACKNOWLEDGMENTS

The work is supported by NSF grant 2013502.

7. REFERENCES

[1] N. Kamaruddin and S. Sulaiman. Understanding interface design principles and elements guidelines: A content analysis of established scholars. In *Proceedings of the Art and Design International Conference (AnDIC 2016)*, pages 89–100. Springer, 2018.

[2] S. Kardan and C. Conati. Providing adaptive support in an interactive simulation for learning: An experimental evaluation. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3671–3680, 2015.

[3] R. Lewis, S. Stoney, and M. Wild. Motivation and interface design: maximising learning opportunities. *Journal of Computer Assisted Learning*, 14(1):40–50, 1998.

[4] B. Mostafav and T. Barnes. Exploring the impact of data-driven tutoring methods on students' demonstrative knowledge in logic problem solving. *International Educational Data Mining Society*, 2016.

[5] B. Mostafavi, G. Zhou, C. Lynch, M. Chi, and T. Barnes. Data-driven worked examples improve retention and completion in a logic tutor. In *Artificial Intelligence in Education: 17th International Conference, AIED 2015, Madrid, Spain, June 22-26, 2015. Proceedings 17*, pages 726–729. Springer, 2015.

[6] F. Paas, A. Renkl, and J. Sweller. Cognitive load theory and instructional design: Recent developments. *Educational psychologist*, 38(1):1–4, 2003.

[7] P. Shabrina, B. Mostafavi, S. D. Tithi, M. Chi, and T. Barnes. Learning problem decomposition-recomposition with data-driven chunky parsons problems within an intelligent logic tutor. 2023.

[8] J. Sweller, J. J. Van Merrienboer, and F. G. Paas. Cognitive architecture and instructional design. *Educational psychology review*, 10:251–296, 1998.

[9] M. Ueno and Y. Miyazawa. Irt-based adaptive hints to scaffold learning in programming. *IEEE Transactions on Learning Technologies*, 11(4):415–428, 2017.

APPENDIX

A. INTERFACE FOR TWO PROBLEM TYPES

In the study, the tutor presents two different problem types: Problem Solving (PS) and Worked Example (WE). In Figure 6, the PS interface shows how students iteratively derive intermediate nodes to complete the solution. There is a "Get Hint" button at the bottom.

In Figure 7, the WE interface is shown where, at the bottom of the interface, students click the Next and Previous arrows to proceed through the solution. The explanation for each worked step is displayed below the solution workspace.

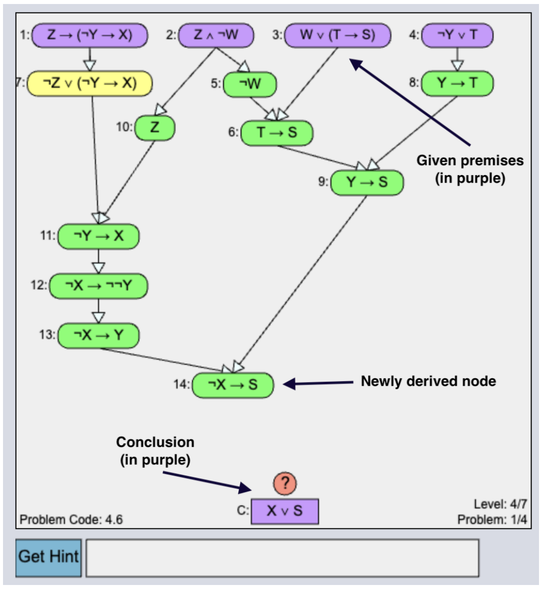


Figure 6: Problem Solving (PS) Interface

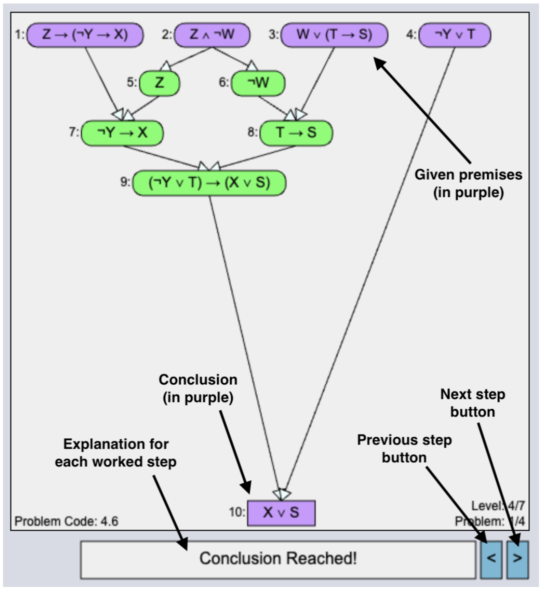


Figure 7: Worked Example (WE) Interface