Int J Artif Intell Educ
https://doi.org/10.1007/s40593-023-00338-1

ARTICLE

Investigating the Impact of Backward Strategy Learning in a Logic Tutor: Aiding Subgoal Learning Towards Improved Problem Solving

Preya Shabrina[1] · Behrooz Mostafavi[1] · Mark Abdelshiheed[1] · Min Chi[1] · Tiffany Barnes[1]

Accepted: 4 May 2023

Abstract

Learning to derive subgoals reduces the gap between experts and students and makes students prepared for future problem solving. Researchers have explored subgoal-labeled instructional materials in traditional problem solving and within tutoring systems to help novices learn to subgoal. However, only a little research is found on problem-solving strategies in relationship with subgoal learning. Also, these strategies are under-explored within computer-based tutors and learning environments. The backward problem-solving strategy is closely related to the process of subgoaling, where problem solving iteratively refines the goal into a new subgoal to reduce difficulty. In this paper, we explore a training strategy for backward strategy learning within an intelligent logic tutor that teaches logic-proof construction. The training session involved backward worked examples (BWE) and problem solving (BPS) to help students learn backward strategy towards improving their subgoaling and problem-solving skills. To evaluate the training strategy, we analyzed students' 1) experience with and engagement in learning backward strategy, 2) performance and 3) proof construction approaches in new problems that they solved independently without tutor help after each level of training and in posttest. Our results showed that, when new problems were given to solve without any tutor help, students who were trained with both BWE and BPS outperformed students who received none of the treatment or only BWE during training. Additionally, students trained with both BWE and BPS derived subgoals during proof construction with significantly higher efficiency than the other two groups.

Keywords Subgoal · Logic tutor · Intelligent tutor systems · Backward strategy · Forward strategy

✉ Preya Shabrina
pshabri@ncsu.edu

Extended author information available on the last page of the article

Published online: 21 August 2023

Introduction

Attaining the skill to define subgoals or 'subgoaling' is an important component of learning that leads to efficient problem solving. Existing literature has defined 'subgoaling' in at least two different ways. According to the classic Catrambone work (Catrambone, 1998) , 'subgoaling' is the process of identifying high-level features of a problem and decomposing the problem into smaller and easier sub-problems that conjointly achieve the overall goal. 'Subgoaling' is also defined as the process of refining the goal into subgoal(s) such that the distance between the goal and the givens of the problem is reduced and the difficulty in achieving the original goal directly is eliminated (Laird et al., 2012; Newell, 1994; VanLehn, 1988) . Experts produce subgoals during problem solving more efficiently and easily, and attaining the skill to generate subgoals can induce expert-like behavior in novices (Margulieux et al., 2016; Catrambone, 1998) . Thus, researchers from various educational domains have explored methods to help students learn to identify subgoals and/or to improve their subgoaling skills. These methods include subgoal-labeled examples (Catrambone, 1998, 1995) , expert explanations for subgoals, and asking students to write explanations for given subgoals (Margulieux and Catrambone, 2017, 2021) , etc. These studies demonstrated that these subgoal-infused tutoring methods helped to improve novice performance, using test scores to measure learning.

However, existing research has scarcely investigated the relationship between problem-solving strategies and students' subgoal generation process or skills. The two most common problem-solving strategies found in literature are forward and backward strategies. The forward strategy consists of starting from the givens in the problem description and working towards the goal at each step. On the contrary, the backward strategy consists of starting from the goal and refining the goal at each step until the refined goal can be justified by the givens. In other words, the backward strategy can be seen as refining the goal to form subgoals at each backward step. By this definition, the backward strategy is a subgoaling process. Also, experts often combine backward strategy with forward strategy during problem solving (Sweller, 1988) . While combining the strategies, they use forward strategy to identify what subgoals to derive and refine the goal into those subgoals. Then, they use forward strategy to derive those subgoals. Research on human cognitive processes suggests that students may try to work backwards more to refine the goal when they have low prior knowledge (Sweller, 1988) . This may be because working backwards can help them to reduce the space of possible next steps for working forward (Heyworth, 1999) . However, the main difficulty students face while carrying out a backward step lies in the selection of the right subgoal that will significantly reduce the distance between the goal and givens (Trafton and Reiser, 1991; Anderson, 2014; Matsuda and VanLehn, 2005) . Despite the direct relationship between backward strategy and subgoaling, and students' natural tendency to try to think backwards when in need of a subgoal, no study has been conducted investigating the impact of learning backward strategy on students' subgoaling skills. Matsuda and VanLehn (2005) investigated the impact of learning backward strategy on students' performance after they were trained with a very small set of training problems. Their findings indicate that carrying out backward strategy is hard for students and those who learned the strategy did not perform well.

However, no measures were explored to improve the training process and help students to learn the strategy. Also, the study did not investigate if learning the strategy had any impact on students' subgoaling skills. Apart from the study mentioned above, a few additional research studies have compared expert and novice problem-solving strategies (Sweller, 1988; Heyworth, 1999) , or investigated subgoal generation strategies in general (Jensen, 1987) .

To address the gaps in prior research, in this study, we designed a training session within an intelligent logic tutor, Deep Thought(DT), to support backward (BW) strategy learning as a medium to aid subgoaling and problem-solving skills. We designed and implemented two new types of problem within DT that engage students in understanding and using backward (BW) strategy. The problem types are: 1) Backward Worked Examples (BWE) to demonstrate the strategy, and 2) Backward Problem Solving (BPS) to allow students to practice the strategy. We exposed students to a training session that contains these problems. The training session contained 20 problems where students observed proof construction using backward strategy (via BWE), practiced it (via BPS), and also got additional problems to master it before taking a posttest. Then, we investigated the impact and efficacy of our training strategy based on: 1) students' experience with and response to the training; 2) students' score-based performance in new problems after training; and 3) student approaches to proof construction where we graphically presented student approaches using *Approach Maps* (Eagle and Barnes, 2014) and identified efficient subgoal derivation in those maps. In this paper, a subgoal is a proposition that reduces the difficulty of a proof construction problem when identified/derived and is often an output of a backward step. The meaning of subgoal in the context of logic proof problems within Deep Thought is detailed in "Subgoals in the Context of Logic Proof Problems in Deep Thought". The key finding of our evaluation suggests that, although backward strategy learning can be difficult for students, students can overcome the difficulties when sufficient problems are given during training. Also, learning the strategy can improve students' subgoaling skills and lead them to more efficient problem solving.

The main contributions of this study are: 1) an effective training strategy for backward strategy learning which can be easily adapted for tutors from other structured problem-solving domains, 2) demonstration of a graph-mining-based "approach map" analysis that revealed how the training session impacted students' subgoaling and problem-solving skills and approaches, and 3) important insights on the impact of BW strategy on students' problem-solving performance that could be helpful in problem and training session design within automated tutors.

Related Work

Subgoaling reduces problem-solving complexity either by breaking down a problem or by refining the goal of a problem (Simon and Newell, 1971) and thus, is an important skill to attain for efficient problem solving. Also, research claims that making subgoals

available during problem solving reduces students' cognitive load[1] and helps them perform better (Margulieux et al., 2012) . Thus, from an educational perspective, researchers have explored subgoals in problem solving mainly in two branches: 1) by attaching subgoals in instruction to improve students' problem-solving performance and 2) by exploring methods to improve students' subgoaling skills.

Many studies explored subgoal-infused learning materials for traditional or learning environment/tutor-aided structured problem solving. For example, researchers explored subgoal-labeled instructional materials in mathematics (Catrambone, 1998) and programming problem solving (Margulieux et al., 2012; Margulieux and Catrambone, 2014) . The explored materials include subgoal attached to problem description, subgoals attached to a group of steps in worked examples, etc. These studies showed that subgoal-labeled materials reduced students' cognitive load and helped them to be better at programming problem solving. Catrambone and Holyoak (1990) found evidence that providing subgoals during problem solving can also be helpful in the transfer of problem-solving skills in domains like algebra and probability. Marwan et al. (2020) and Shabrina et al. (2020) explored automatically extracted data-driven subgoals (Zhi et al., 2018) attached to programming tasks. They found evidence of better performance, higher task completion rate, and less idle time when subgoals were presented in the system. Additionally, Shabrina et al. (2020) found evidence that students closely followed the given subgoals, and tried to achieve them, which shaped their programming approach. Cody and Mostafavi (2017) provided subgoal hints during logic proof construction within an intelligent logic proof tutor. Contrary to research that found positive results with subgoals, they observed that students who received subgoal hints skipped more problems, and had a significantly higher dropout rate. Morrison et al. (2015) explored two instructional methods in introductory programming tasks: 1) attaching subgoal labels given with the task, and 2) requiring students to generate their own subgoals. They observed that the second group who generated subgoals outperformed the first group. However, they also stated that those who generated subgoals needed to spend more time which could also be the source of additional learning. In a later study, Margulieux and Catrambone (2017, 2021) showed that students learned better when they were presented with subgoals and asked to write explanations for the subgoals, when compared to generating their own subgoals. Overall the studies exploring subgoal-labeled instruction to improve students' performance, found positive results. Also, the results showed that students learn better when they generate explanations for subgoals themselves, rather than when they are given explanations. While generating explanations, students have to think about why some steps are subgoals and how they contribute to the solution overall which could trigger learning.

We found only a few studies that explored methods to improve students' subgoaling skills. Richard Catrambone conducted a series of studies (Catrambone, 1995, 1996, 1998, 1994) exploring high-level abstract subgoal labels attached to groups of steps in worked examples to aid subgoal learning. These studies showed that subgoal labels are most effective in fueling transfer and helping students to learn subgoaling, when

[1] Cognitive load refers to the load on a student's working memory during learning through problem solving (Van Gog et al., 2010)

they give away structural information about the problem solution rather than showing problem-specific steps. The results of these studies also demonstrated that students who learned with subgoal-labeled examples were more successful in test problems that involved the same subgoals learned during training. However, achieving the subgoals required a different procedure within those test problems. The Catrambone studies used students' performance and the presence of the same subgoals taught during training in the solutions of new test problems as indicators of their subgoaling skills. However, students' capability or efficiency in identifying and deriving subgoals in general and the quality of the subgoals students could derive where the same subgoals may not be involved remained unexplored.

Although researchers have shown interest in exploring methods to improve students' subgoaling skills, there is little research that links this work to backward strategy learning. There are two primary problem-solving strategies: forward and backward strategy or chaining. Forward strategy learning involves starting from the information given in the problem description (premises) and applying domain principles to these premises to derive new statements (Al-Ajlan, 2015) . Ideally, these statements should be reducing the difference between the given premises and the desired problem conclusion or goal. In backward strategy, also called backward chaining, problem solving is carried out starting from the goal, and in each step, the goal is refined to a new subgoal until the initial problem state is reached (Al-Ajlan, 2015) . Due to low prior knowledge, novices tend to use backward strategy to figure out subgoals of a problem. This is because the backward strategy possibly helps them to identify a reduced space of possible next steps, whereas the forward strategy requires them to have knowledge of all possible rules/actions and take a plausible step using that knowledge (Heyworth, 1999) . On the other hand, experts can even form subgoals while working in the forward direction due to their high prior knowledge in a domain (Sweller, 1988; Heyworth, 1999; Larkin et al., 1980; Chi et al., 1981) . However, experts often combine forward and backward strategies during problem solving, i.e. they work forwards while implicitly keeping backward subgoals in mind (Sweller et al., 1983) . Combining the strategies enable the experts to focus on narrowing the gap between the goal and the givens without explicitly working backward. Backward strategy is also used to derive subgoals in another problem-solving strategy called the means-ends analysis (Newell et al., 1972) . In means-ends analysis, backward strategy or steps are used to identify subgoals that will best help to reduce the differences between givens and the goal of a problem. Then, the backward steps are followed by a sequence of forward steps to derive those subgoals. Matsuda and Vanlehn (2004) while constructing GRAMY, a geometry tutor capable of constructing proofs, used backward strategy (within the proof construction mechanism of the tutor) to deduce subgoals. These uses of BW strategy, specifically, the adoption of this strategy by experts and within the problem-solving mechanism of automated tutors, signify that this strategy can help in subgoaling.

Despite these uses of backward strategy for subgoaling and students' natural inclination to try to work backward to identify subgoals, we found very little on the impact of learning backward strategy on students' problem-solving or subgoaling skills in the existing literature. To the best of our knowledge, the only relevant study in this regard was conducted by Matsuda and VanLehn (2005). They trained students to learn

forward and backward strategy with a geometry-theorem-proving tutor and observed that students who learned forward strategy performed better than those who learned backward strategy. They also observed that students faced difficulties while constructing proofs using backward strategy due to the need of coming up with unjustified statements (subgoals) that are to be proven next. In this study, students were trained with only 6 proof-writing problems and other non-proof problems. Since the training session contained only a few proof-writing problems, it is not clear if students' difficulties with backward strategy could be overcome with additional training or the usage of more training problems. Also, the posttest contained three problems that came from two different sets with isomorphic superficial features. I.e. all students did not get the same posttest problems. But the differences among the problems and the impact of those differences on the performance of students learning different strategies are not discussed. Note that the differences in posttest problems can impact students' posttest performance and thus, the performance may not exclusively represent the impact of learning forward or backward strategy. Moreover, although using backward strategy involves iterative subgoaling, an analysis of what subgoals students derived or how their subgoaling skills were impacted after training is also missing. Overall, the existing literature does not give sufficient insights into how students can be trained to learn the backward strategy and how learning the strategy impacts their problem-solving and subgoaling skills.

Thus, in this study, we aimed to aid subgoal learning by training students to learn backward strategy through demonstration (using backward worked examples, BWE), and practice (using backward problem solving, BPS) within Deep Thought. The training phase was long, involving 20 logic-proof construction problems that should provide students with enough time and practice to master the strategy. We evaluated our training procedure based on students' experience with and responses to the training. Specifically, we focused on if their difficulties with the strategy reduced over time and if they were able to efficiently use the strategy as training progressed. We also analyzed students' test score-based performance on the same posttest problems. Additionally, unlike prior studies, we investigated efficiency in subgoal derivation using approach map analysis of problems independently solved by students without tutor help.

Method

Deep Thought, The Logic Proof Tutor

We conducted our study using an intelligent logic tutor, Deep Thought (DT) [Fig. 1]. In DT, students are given logic-proof construction problems where the premises and the conclusion to be proved are given as visual nodes. A list of logic rules is provided from where students can select rules to apply on the given premises to derive new propositions to reach the conclusion. Deep Thought is organized into 7 levels. Level 1 starts with a worked example (WE) that shows how a problem is solved in the interface step by step. Then, two problem-solving (PS) type problems are given as a pretest. Levels 2-6 are training levels, each with 3 training problems of type PS or WE and ending with a training level posttest problem in PS mode. During training problems in

PS mode, students can request on-demand step-level hints. Level 7 is a posttest level with 6 PS-type problems. The tutor does not offer any hint or support during WEs, the last problem of the training levels, or the posttest problems.

Problem Solving (PS): In PS [Fig. 1], students are required to construct all the steps of a proof themselves. In PS, students can choose to use forward, backward, or both strategies at their discretion. Construction of a step in the forward (FW) direction involves selecting one or two premises and applying a rule on them to produce a new justified proposition. For example, in Fig. 1, *Simplification* rule is applied twice on node 2 to derive nodes 5 and 7. During training problems in PS, students can request on-demand hints which suggest a statement or node to derive in the hint box at the bottom of the work area. If another hint is requested, DT tells the student what nodes can be used to derive the suggested statement. If a further hint is requested, the rule needed is also suggested. On the other hand, carrying out a step in the backward direction [Fig. 3b] starts from the unjustified conclusion node. First, the students have to click the '?' button [backward derivation button] to indicate their intention to derive unjustified propositions backwards. Then, they have to select a rule and input the propositions they want to generate. For example, in Fig. 3b, nodes 6 and 7 are generated using *Modus Ponens* rule, since this rule can derive the conclusion (node 5) from the two nodes. There are no backward hints in DT. Note that an incorrect rule application either in forward or backward direction does not generate any new node.

Worked Example (WE): Figure 2 shows an example of WE where the tutor constructs a proof step by step as the students click on the '>' button, i.e. the next step button. During the construction of a step, the tutor also provides a textual explanation for the step in the text box shown in the figure. Using the '<' button, i.e. the revert

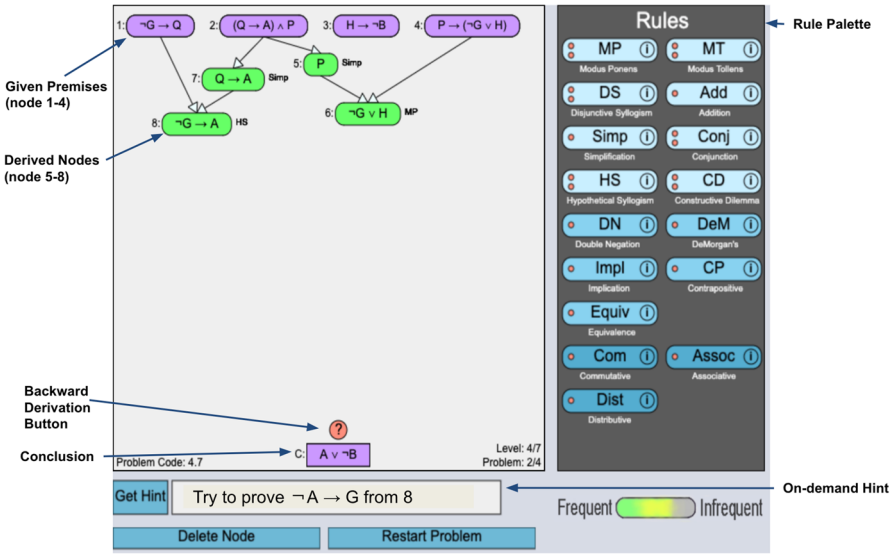


Fig. 1 Deep thought interface containing a Problem-Solving (PS) problem with a few steps derived in the forward direction

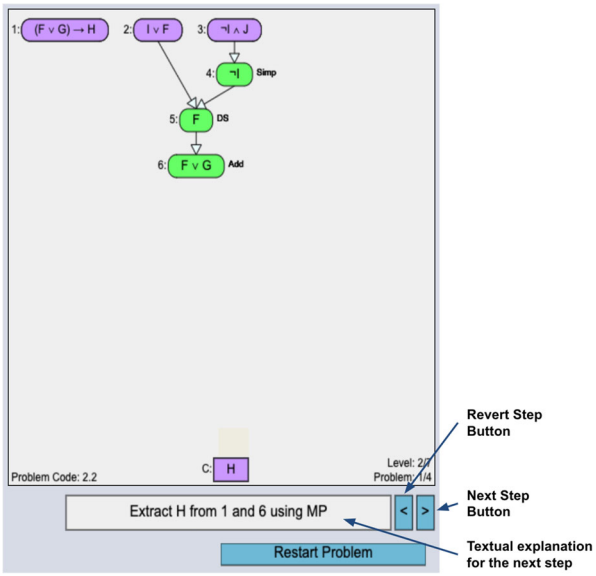
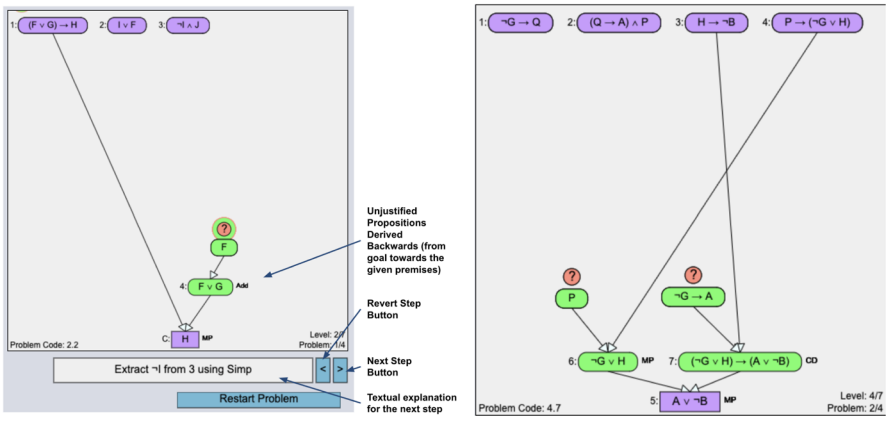


Fig. 2 Demonstration of Worked Example (WE)

button, the students can remove previously constructed steps and replay the construction of each step as many times as they want. Note that WEs are always constructed in the forward direction by the tutor starting from the given premises and moving toward the goal as shown in Fig. 2. Hints are not available for WE problems, as all steps are already worked out by the tutor for students in this problem type.

New Proof types: For this study, we designed and implemented two new proof types within DT:



(a) BWE (b) BPS

Fig. 3 New Proof Types within DT: (a) Backward Worked Example (BWE); and (b) Backward Problem Solving (BPS)

1. *Backward Worked Example* (BWE): Figure 3a shows the interface for BWEs which is similar to WEs. However, in BWEs, the tutor constructs an entire proof by using BW derivation only as students click on the > button. For example, in Fig. 3a, the tutor has derived two backward steps. The first BW step shows that H can be proved from given premise 1 and $F \vee G$ (node 4) using the 'Modus Ponens' rule. And, the second step shows that $F \vee G$ can be proved from F using the 'Addition' rule. Eventually, a justification for F from the given premises is established by the tutor in backward direction to complete the proof.
2. *Backward Problem Solving*: The interface for BPS is shown in Fig. 3b which is similar to PS. However, forward derivations are disabled in a BPS. In the figure, first, nodes 6 and 7 are generated as unjustified propositions from the conclusion (node 5) using the 'Modus Ponens' rule. Then, P is generated since given premise 4 and P can justify node 6 using the 'Modus Ponens' rule. Similarly, $\neg G \Rightarrow A$ is also generated as an unjustified predecessor of node 7. Eventually, all unjustified nodes need to be connected to the given premises in the backward direction to complete the proof. Each of these steps where a new set of unjustified nodes is generated to move towards the given premises is referred to as 'refining the goal into a new set of goals or subgoals' in this paper.

Subgoals in the Context of Logic Proof Problems in Deep Thought

In DT, all logic-proof problem solutions need to be sequentially constructed either in the forward (FW) or the backward (BW) direction. When constructing logic proofs, as in any problem-solving domain, there are some natural subgoals experts typically identify to optimally shorten the distance between the given problem parameters and the conclusion of the proof.

For example, for the problem shown in Fig. 2, the problem conclusion, or goal, 'H' needs to be proved from the given premises. Since only node 1 ($F \vee G \Rightarrow H$) contains the variable 'H', deriving facts that could be combined with node 1 to derive H is a natural, expert subgoal. Given the structure of node 1 and the domain principles, an expert would determine that 'Modus Ponens', could be used to derive H if node 1's hypothesis, the proposition $F \vee G$, were true. Therefore, an expert would set $F \vee G$ as a subgoal. DT was designed to allow students to explicitly set such subgoals by working backward by pressing the '?' button above the conclusion 'H'. Similarly, since none of the other premises contain the variable 'G', proving 'F' becomes another subgoal that can be used to derive $F \vee G$ later using the 'Addition' rule. In this proof, experts identified $F \vee G$ and F as subgoals, since identifying and deriving these propositions make the rest of the proof straightforward.

As described above, experts use BW strategy to identify subgoals by comparing the goal and givens of the problem backward. And, in prior research, using backward derivations to refine the goal while reducing the goal-givens distance has been referred to as subgoaling (Gick, 1986; Laird et al., 2012; Newell, 1994; VanLehn, 1988) . Thus, throughout the remainder of the paper, we refer to the process of working backward within DT from the problem-solving goal to derive a new subgoal as explicit **subgoaling**. For example, if in the problem shown in Fig. 2, H is refined to subgoal $F \vee G$ and then to F by working backwards, the process is called subgoaling, and the

result(s) of the refinement ($F \vee G$ or F) is called a subgoal. The backward refinements or subgoaling can be visualized in Fig. 3a.

Within DT, we also considered the case of implicit subgoaling where students identify subgoals using BW thinking. In this case, students work backward in mind but derive subgoals using only forward derivation. To identify possible cases of implicit subgoaling, we analyzed students' efficiency in deriving expert-identified subgoals during their proof construction attempts (Detailed in "Approach Map Analysis"). Overall, efficient subgoaling regardless of the visible direction of subgoal derivation can lead to efficient proof construction.

Note that although the term 'subgoaling' is used analogously to carrying out a step using 'BW strategy', there is a difference between the efficiency in using 'BW strategy' and efficiency in 'subgoaling'. In the context of DT, a student can be called efficient in 'BW strategy' if they can quickly derive a BW step. However, the efficiency in subgoaling lies in 'what propositions or subgoals students derived using BW strategy' and 'how they derived it' (for example, in less time, with fewer propositions, etc.) while constructing a proof. For example, in the problem in Fig. 3a, a student can BW derive $H \vee F$ from the conclusion 'H' with the 'Addition' rule. However, the subgoal $H \vee F$ will lead them in the wrong direction and thus, this subgoaling step is inefficient and incorrect. As explained earlier, experts use BW strategy to identify the propositions or subgoals that could optimally reduce the distance between the givens and the goal. Thus, to analyze subgoaling efficiency, we identified expert-identified subgoals in student approaches and analyzed how efficiently students derived those subgoals. The expert-identified subgoals are used as a baseline to evaluate subgoaling efficiency. The experts identified the subgoals for each problem independently without looking at student data ahead of time.

Research Questions

As discussed in "Related Work", prior literature showed that experts often use BW strategy for subgoaling. On the other hand, students may try to use it but they find it difficult. Thus, in this study, we first train students to learn and practice the backward (BW) strategy while they construct logic proofs in a training session within DT. Then, we investigated the impact of learning BW strategy on problem-solving and subgoaling skills. Also, to understand students' problem-solving experience during training, we investigated trends in students' difficulties with and adoption of BW strategy over the period of training and posttest. We measured student difficulties by multiple metrics like problem-solving time, step time, step count, etc. during problem solving using BW strategy. We also analyzed students' responses to the training in terms of their adoption of BW strategy during independent problem solving. Overall, our investigation on the efficacy and impact of backward strategy learning within DT was focused on the following three research questions:

- ***RQ1 (Students' Experience, and Response):*** How does the backward strategy training impact students' experience in DT, and how do they respond to the training?

- ***RQ2 (Impact on Performance):*** How does learning backward strategy impact students' performance in new problems?
- ***RQ3 (Impact on Subgoaling Skills):*** How does learning backward strategy impact students' subgoaling skills?

Experiment Design

Training Treatments: We implemented three training treatments for our experiment: 1) Control (C): this treatment group received only WEs and PSs, 2) Treatment 1 - BWE (T_1): this treatment group received BWEs in addition to WEs and PSs, and 3) Treatment 2 - BWE+BPS (T_2): this treatment group received both BWEs and BPSs along with WEs and PSs. Note that the control (C) [baseline] group was never shown BW strategy using examples and they were never prompted by the tutor to use BW strategy. However, they had the option to use BW strategy in PS if they want. The tutor showed the T_1 group BW strategy using BWEs but never prompted them to use it. But like the C group, they had the option to use it in PS. On the other hand, BW strategy was shown to the T_2 group using BWEs and they were prompted to use BW strategy mandatorily in BPSs. The reason for creating two separate treatments (T_1 and T_2) for BW strategy learning was to identify the appropriate level of engagement (BWE vs. BWE+BPS) necessary for the students to learn the strategy.

Problem Organization: The problem organization in the training levels is shown in Fig. 4. The blue squares indicate the problems where BWEs were given to T_1 and T_2 and the green squares indicate the problems where BPSs were given to T_2. Notice that we distributed BPSs only in the first half of the training. This was done to ensure that T_2 gets the opportunity to independently use and master BW strategy in the second half of the training phase. However, the tutor continued to give BWEs till Level 5 to remind the students that BW strategy could be useful during proof construction.

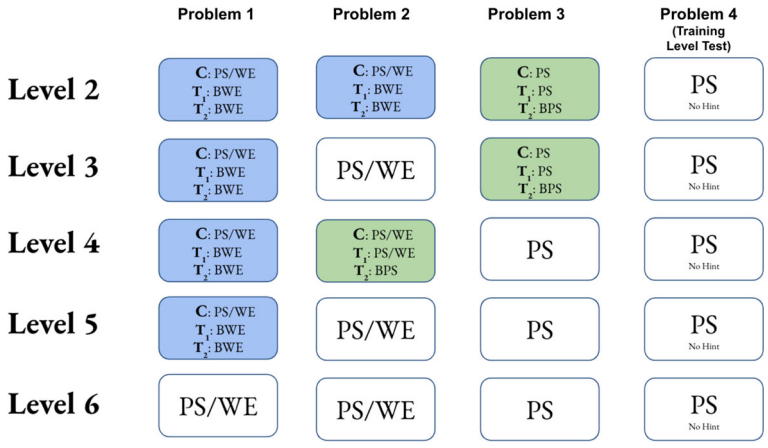


Fig. 4 Problem Organization in Training Levels for C, T_1, and T_2. **Note:** [PS/WE] notation refers to a random selection between PS and WE

Note that the problem organization shown in Fig. 4 ensures that all conditions have nearly equal numbers of examples (WE/BWE) and problem-solving (PS/BPS) type problems. Students get PS or BPS in the last two problems of each training level. For the first two problems in each training level, our problem assignment strategy ensures that the students get at least one example (WE/BWE) and one problem-solving problem (PS/BPS). If the first problem in a level is WE or BWE, the next one will be PS or BPS, and vice-versa. The only exception is level 2 where both of the first two problems for groups T_1 and T_2 are of type BWE. This was done so that groups T_1 and T_2 are exposed equally to backward strategy as the control (C) students are exposed to forward strategy through WEs. Recall that, in all conditions, one introductory WE problem is shown before the pretest PS problems, but no BWEs are shown. Overall, at most, T_1 and T_2 groups have one more example and one less problem solving than the control, C.

Deployment and Data Collection

We deployed DT with our three training treatments in a Discrete Mathematics course for computer science majors offered in a public research university in the United States. The students used DT for a take-home assignment and they were instructed to do the assignment without any collaboration with others. Each student participating in the course was assigned to one of the three conditions after they completed the pretest level. 61 students were assigned to C, 61 to T_1, and 62 to T_2. The assignment algorithm distributes students equally across the treatment groups while ensuring that the pretest scores of the three groups come from the same distribution. The possibility that the students violated the instruction by identifying those in the same training condition and collaborating with them was assumed to be low.

At the end of the experiment, 168 students completed all 7 levels of the tutor with 59 students coming from group C, 55 from T_1, and 54 from T_2. The students spent around 6 hrs on average in the training phase with each level requiring a different amount of time: level 2: 1.58 hrs; level 3: 1.16 hrs; level 4: 1.34 hrs; level 5: 0.97 hrs; and level 6: 1.24 hrs. While students worked in DT, our system collected all information required to replay and reconstruct their proof construction attempts for all problems. The data collected from the DT system are described in Table 1. Note that step counts also include steps derived in previous attempts, i.e. consider restarts. Also, note that 'Actions' [Table 1] are very granular interactions with the system, and carrying out an action requires a small amount of time (milliseconds to a few seconds). On the other hand, constructing a 'Step', or deriving a node takes much longer (10 seconds to several minutes), involves multiple actions, and actions can be even reverted in the process. For example, a student can click on the backward derivation button and then can deselect the button to derive a forward step. Thus, 'Action Count' can have a much higher value than 'Step Count' and 'Problem Time'. Certain actions are representative of students' intention to adopt a particular approach. For example, a click on the Backward Derivation Button is considered a backward (BW) action since it shows the intention to derive a step Backward. Thus, we also collected the direction (BW/FW) of each action and corresponding counts for our analysis. Note that a BW action may

Table 1 Description of collected data from deep thought

Collected data	Description
Interactions with the interface	Click, selection/deselection of rules/nodes, etc. These interactions are also referred to as Actions
Proof Steps	Derivation/deletion record of nodes with associated predecessors and rules, direction of derivation[FW/BW].
Step count	# of derived nodes during the construction of a proof
Action count	# of total actions taken during the construction of a proof. Action count is always greater than step count since each step involves multiple actions (selecting nodes/rules, BW derivation button etc.).
Step time	Time to derive each step.
Problem time	Time to complete the proof of a problem.
Restart count	# of restarts during the construction of a proof.
Session Count	# of sessions observed during the construction of proof. A new session starts if the student logs out of the system and logs in again later.
Problem Score	(1/3)*Accuracy + (1/3)*Time Function +(1/3)*Size

not lead to a successfully derived BW step. Students might start with a BW action but can switch strategy during independent problem solving due to difficulties. Thus, BW action counts can also be much higher than the count of BW steps.

For each PS problem completed by the students, they were assigned a Problem Score that is an equally weighted sum of **Accuracy**, **Time Function** [a function of Problem-solving Time], and **Size** [a function of Step Count] of the solution as shown in the last row of Table 1,
where

$$\textbf{Accuracy} = \#\text{correct rule applications}/\#\text{total rule applications} \quad (1)$$

$$\text{Time Function} = 1-((\textbf{Problem-solving Time}-Time_{25})/(Time_{75}-Time_{25})) \quad (2)$$

$$Time_{75} = 25\text{th percentile of the time taken to solve the problem} \quad (3)$$

$$Time_{25} = 25\text{th percentile of the time taken to solve the problem} \quad (4)$$

and,

$$\text{Size (Solution X of Problem Y)} = \exp(\text{Normalized Length of X}) \quad (5)$$

$$\text{Normalized Length of X} = (\textbf{Step Count(X)} - C_1)/C_2 \quad (6)$$

$$C_1 = \text{Optimal Solution Length(Y)} \quad (7)$$

$$C_2 = \text{Difference between Longest and Optimal Solution Lengths(Y)} \quad (8)$$

According to this formula, shorter solutions constructed with higher correct rule applications and in less time get higher scores. Application of this score function can also be found in prior research (Abdelshiheed et al., 2020, 2021; Sanz Ausin et al., 2020; Cody et al., 2018) .

In the subsequent sections, we describe our analyses to address the three research questions and the corresponding results. Throughout this study, we used Kruskal-Wallis tests to find significant differences across the training groups and performed post hoc pairwise Mann-Whitney U tests with appropriate Bonferroni Correction to find an ordering of the groups.

Note: To report the results of statistical analyses, we show means as a measure for central tendency, and p-values from pairwise posthoc tests as evidence while comparing the three training groups. For details check out the supplementary materials attached at the end of the manuscript.

RQ1: Students' Experience and Response

Training Phase Statistics

To understand students' experience with the training treatments, we calculated step time, problem time, step count, and restart/session counts for the PS and BPS problems given in the first three problems of each training level. These metrics are defined in Table 1. Note that higher step time, problem time, and step counts indicate higher difficulties. Also, restart counts indicate how many times the students started over the construction of a proof. Thus, a higher number of restarts indicates that the students were potentially uncertain about the correct approach to construct the proof, i.e. they were finding the construction difficult. Similarly, session counts indicate how many times the students paused or logged out and logged in before finally completing the proof of a given problem. And thus, a higher number of session counts is also indicative of difficulties.

Table 2 shows the training phase metrics values for PS/BPS problems. WE/BWEs are not included in the metric values. One limitation of this analysis is that the PS or BPS problems assigned to each student during training can be different due to random assignment. However, the problem pool for each level contains 4-6 problems and they are of similar difficulty. Thus, this analysis assumes that the problems do not introduce significant differences, but that any relative difficulties students faced were due only to different strategies (forward or backward).

We did not find any differences across the groups during pretest in these metrics. However, during the training phase, we observed significant differences in the metrics when performed Kruskal-Wallis tests with subsequent pairwise Mann-Whitney U tests with Bonferroni correction. The corrected α=0.05/3=0.017, since, for each metric, each data point was used in three pairwise tests (C:T_1, C:T_2, and T_1:T_2) to compare the groups. Recall that T_1, and T_2 both were given BWE. Groups C and T_1 received only PS to solve themselves, while T_2 additionally received BPS.

We found significant differences in the average of time-related metrics in the training problems. Both step time and overall problem time were significantly higher for T_2

Table 2 Metric Values (avg.) during Training across C, T_1, and T_2 (for T_2, metric values are shown separately for BPS, and PS)

Group	Step Time (min)	Prob. Time (min)	Step Ct.	Restarts	Sessions
C	3.3	36.6	10	0.38	4.2
T1	3.7	33.1	8	0.30	4.1
T2	PS: 2.8	PS: 39.3	PS: 9	PS: 0.32	PS: 7.2
	BPS: 5.7	BPS: 49.8	BPS: 8	BPS: 0.69	BPS: 6

than C and T_1. In the case of problem time, $P_{MW}(T_2 > C)$[2]=0.006, and $P_{MW}(T_2 > T_1) < 0.0001$. In the case of step time, $P_{MW}(T_2 > C)$=0.004, and $P_{MW}(T_2 > T_1)$=0.0001. This trend in average step and problem time of T_2 students during training was due to BPS problems [Notice BPS metrics in Table 2]. This implies that carrying out BW steps was difficult for students which required more time during training. Interestingly, T_1 and T_2 had significantly fewer step counts than C: $P_{MW} < 0.0001$, and P_{MW}=0.001 respectively for the two cases. These statistics suggest exposure to BW strategy possibly pushed T_1 and T_2 students towards shorter solution attempts. I.e., thinking/working BW potentially encouraged students to take better steps to reduce the distance between the goal, and given premises.

Additionally, we observed that T_2 students required significantly more sessions than C and T_1 students to complete training problems: $P_{MW} < 0.0001$ in both cases. T_2 students also had a significantly higher restart count than T_1 ($P_{MW} = 0.0006$) and marginally higher than C (P_{MW}=0.02). These statistics suggest T_2 students struggled during training due to BPS problems. Since BPS problems are restrictive by design, requiring students to construct proofs entirely in backward direction, it indirectly encourages the need for taking the best action at each step.

Student Engagement in Backward(BW) Strategy

As a measure of students' response to the training, we used their independent engagement in BW strategy in terms of BW action count which is representative of student intention or attempts to work backwards. We calculated the counts across the three training groups for the 4^{th} problem of each training level (level 2-6), and the 6 posttest problems in level 7. Since no training treatment or tutor help is given in these problems, this selection of problems ensures that we can understand the impact of the training interventions on students' approaches in independently solved problems. The first three problems in each training level are not specifically included in this analysis, since the students might get hints or training interventions in these problems where they do not have the opportunity to independently act. Also, note that each of these analyzed test problems is characteristically different and designed to test students' knowledge of the application of a variety of rules. According to experts, the problems are equally solvable by forward and backward strategies. Thus, we assumed any

[2] This notation indicates the p-value obtained from Mann-Whitney U test while testing the hypothesis that T_2 has a significantly higher value than C.

differences found in student approaches and performance in any of these problems only represent the impact of the training treatment they received and the skills they acquired. Sample solutions and expert subgoals for each of the problems can (Figs. 1 and 2).

In Table 3, we report mean BW action counts for each of the 11 problems across the three training groups and the p-values of the statistical tests. Note that the counts for the problems are not cumulative. Thus, tests for one problem are considered independent of the tests for all other problems. The reason for analyzing the counts of each problem separately was to investigate the trend over the period of training and posttest. As shown in Table 3, T_2 students carried out significantly more BW actions than group C and T_1 in most of the problems under consideration. The group that received backward examples, T_1, behaved similarly to C in terms of lower explicit usage of BW strategy. Also, in the earlier phases of training 2.4-4.4, T_2 students took too many BW actions ($\sim$31 - $\sim$59 actions) [Table 3, column 4; rows 2-4]. However, in the later phases of training, and in the posttest problems, they possibly became calculative in carrying

Table 3 BW Action Counts and % of BW Actions Among Total Actions across the Three Training Groups while Solving Pretest Problems, Training PS Problems (fourth Problem of Level 2-6), and posttest PS Problems

Prob.	C	T1	T2	p-val
Pretest (avg.)	3.9 (2.2%)	3.2 (2.4%)	4.7 (1.8%)	-
2.4	2.5 (1.3%)	8.7 (5.6%)	**58.5 (17.6%)**	T2 > C(< **0.0001**); T2 > T1(< **0.0001**)
3.4	1.6 (0.9%)	4.1 (2.4%)	**26.4 (13.6%)**	T2 > C(< **0.0001**); T2 > T1(< **0.0001**)
4.4	1.0 (0.7%)	2.1 (1.8%)	**31.1 (12.3%)**	T2 > C(< **0.0001**); T2 > T1(< **0.0001**)
5.4	0.7 (1.2%)	1.2 (1.2%)	**3.3 (4.4%)**	T2 > C(**0.015**)
6.4	1.3 (1.2%)	0.9 (1.0%)	**3.3 (3.9%)**	T2 > C(**0.0004**) ; T2 > T1(**0.015**)
7.1	0.2 (0.4%)	0.3 (0.43%)	**0.6 (0.9%)**	T2 > C(0.23) ; T2 > T1(0.09)
7.2	0.2 (0.6%)	0.1 (0.4%)	**0.3 (1.3%)**	T2 > C(0.48) ; T2 > T1(0.06)
7.3	0.3 (0.4%)	0.8 (0.9%)	**3.4 (5.3%)**	T2 > C(**0.003**) ; T2 > T1(**0.02**)
7.4	0.6 (0.6%)	0.8 (0.7%)	**4.5 (4.8%)**	T2 > C(**0.011**); T2 > T1(**0.006**)
7.5	2.6 (1.9%)	1.2 (1.7%)	**4.0 (5.7%)**	T2 > C(**0.010**); T2 > T1(**0.015**)
7.6	1.8 (0.7%)	6.3 (2.6%)	**11.2 (5.8%)**	T2 > C(**0.001**)

Interpreting p-val column: Read $T_2 > T_1$ (< 0.0001) as T_2 has significantly higher value than T_1 with p < 0.0001. $T_2 > T_1$ (0.0001) means p=0.0001 for the hypothesis. Bold-faced p-values indicate significant differences

out a BW action as indicated by the reduced count (∼3 - ∼11). Our later approach analyses ("Approach Map Analysis") confirmed that in the early phases of training, students used BW strategy inefficiently, with too many BW actions. However, with time T_2 possibly adapted to the new skill and was able to use it efficiently (fewer but correct BW steps) to refine goals to subgoals leading to better performance.

From our analysis of training phase metrics, and students' BW strategy engagement, it is evident that BPS problems posed T_2 students a significant amount of struggle. They needed more time, more sessions, and more restarts while solving BPS problems. Although T_2 students practiced BPS only in three problems, they had to do all the steps in those problems backward which were heavy-load from a cognitive point of view since students find backward derivations difficult during training (Matsuda and VanLehn, 2005) . However, when given new problems, this group voluntarily engaged in BW actions when using the strategy was not even a requirement. On the other hand, group T_1 did not seem to face many struggles. But, in terms of BW strategy usage, they behaved similarly to control C. The statistics described above suggest that only BWE may not be motivating or educational enough for students to attempt to derive propositions in the backward direction. Thus, we conclude that to successfully motivate students to engage in BW strategy, both examples (BWE), and practice (BPS) are necessary. Additionally, recall that the training phase was long as it had 20 problems altogether distributed across 5 levels, and students spent around 6 hours completing the training. And, the T_2 students did not show improved performance immediately, rather they improved gradually throughout the training levels. Thus, a training session as received by T_2 containing enough problems allowing students to sufficiently engage with the strategy might be necessary to allow students time to adapt and become efficient in using the strategy.

RQ2: Impact on Performance

In this section, we investigate students' performance in relationship with their exposure to BW strategy through BWE (T_1), or both BWE and BPS (T_2), or none (control group C). Again, we focus on the student scores in the problems where no training treatment or tutor help was given, and students solved them independently: training level test PS problems: 2.4-6.4, and posttest problems 7.1-7.6. Additionally, we looked at rule application accuracy, problem-solving time, and step count for each problem to investigate the source of higher and lower scores, since these factors impact problem scores. Note that we looked at these metrics for each problem separately to understand the trend in the metric values over the period of training and posttest. Here, we discuss the observed trends in these metrics across the three training groups using plots. Results of statistical tests to explain significant differences in each problem across the groups are reported in supplementary materials as follows: Problem Scores(Table 4), Rule Application Accuracy(Table 5), Step Counts(Table 6), and Problem Time(Table 7). The test results include the average and standard deviation of the metric values across the training groups and p-values from Kruskal Walis tests and pairwise Mann-Whitney U tests with Bonferroni Corrected α=0.017.

Problem Scores: In Fig. 5, we plot the average scores for each problem across the three training groups. As shown in the figure, problem scores across the groups in the pretest were very similar and our statistical tests found no significant differences in this phase. However, in the earlier phases of training, T_2 received significantly lower scores (in prob. 2.4), or lower scores on average (in prob. 3.4 and 4.4) than the other two groups. As the training progressed, T_2 outperformed both C, and T_1 (in prob. 5.4), or performed at least as well as them (in prob. 6.4). For the scores of prob. 2.4-6.4, notice the 'Training Level Tests' segment in Fig. 5. Since, problem score is our main performance metric, to reduce the chances of false positives, on top of statistical tests reported in the supplementary material, we performed a unified mixed-effect regression analysis (West et al., 2006; Harrison et al., 2018) for all the training level test problems to further verify the association between given training treatment (C, T_1, or T_2) and the problem scores. In the analysis, we used problem ids as a random effect variable so that the impact of the problems being different and that they were given in different training levels are eliminated. We used the treatment type as the fixed effect and problem score as the dependent variable. The result of the analysis gave a p-value=0.056[3] which indicates that students' problem scores in training level test problems were marginally impacted by the type of training treatment they received. Notice in Fig. 5 that the differences in the problem scores across the training conditions became more prominent as the students progressed from training toward posttest.

Subsequently, when we performed a similar mixed-effect regression analysis for the posttest problems, the p-value was 0.02 demonstrating a significant association between training treatment and posttest scores. In the posttest phase [notice the 'Posttest' segment in Fig. 5], T_2 consistently outperformed the other training groups in all problems. They received higher average scores in 7.1-7.2 and significantly higher average scores in 7.3-7.6.

The results of this analysis on the problem scores suggest T_2 became better at problem solving over the period of training. However, in the early phases of training (2.4-4.4), T_2 received lower scores than the other two groups. Recall that, in problems 2.4-4.4, T_2 students were observed to engage in too many BW actions [Table 3, column 4; row 2-4]. Note that, the solution to each problem in DT is 5-15 steps long, and too many backward actions indicate unnecessary propositions or actions were explored by the students. However, in the later levels, they engaged in fewer and possibly only the correct BW actions leading them toward higher scores. This trend suggests that T_2 students improved their efficiency over time in using the BW strategy. On the other hand, T_1 who only received BWE during training received insignificant higher average scores than C in problems 3.4-5.4. In all other cases, T_1 performed similarly to C and did not show consistent signs of improvement. To better understand this trend, notice the closely coupled blue and yellow curves representing the scores of C and T_1 in Fig. 5.

Ratio of Correct Rule Applications or Accuracy: As the first factor that impacts problem scores, we analyzed the rule application accuracy[4] during students' proof construction attempts. We did not find any significant differences in this accuracy

[3] p-value$<$0.05 indicates significant difference

[4] # of correct rule applications/ # of total rule applications

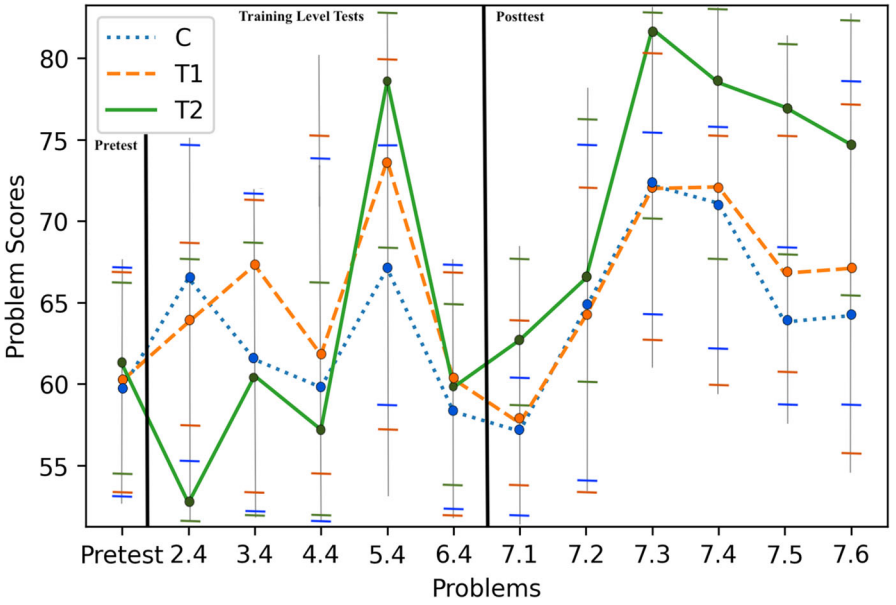


Fig. 5 Trends in Problem Scores in Pretest, Training Level Tests (2.4-6.4), and Posttest (7.1-7.6) PS Problems across the Three Training Groups. [Error bar represents min and max.]

across the three groups for any of the training-level test and posttest problems. Note that accuracy depends on students' knowledge of the logic rules and not on what strategy they learn. Thus, all the training groups were expected to improve similarly in terms of rule application accuracy over the period of training.

Problem-solving Time: To investigate the source of higher and lower scores, we also analyzed problem-solving times across the three training groups. Problem-solving times [Fig. 6a] showed a similar pattern as problem scores [Fig. 5]. In 2.4 and 4.4, T_2 took significantly more time than C and T_1 which resulted in their lower scores in these problems as shown in Fig. 5. In 2.4, the higher problem-solving times of T_2 were due to higher step times and in some cases, were due to higher unnecessary proposition counts [detailed in "Scenario 1 (Early in Training: Poor Performance of T_2 Co-occurring with Many BW actions): Training Prob. 2.4"]. In 4.4, the higher problem time of T_2 was due to higher average step time (average step time in Problem 4.4: C: 2.25 Mins.; T_1: 2.07 Mins.; T_2: 2.71 Mins.). However, as students progressed in DT, problem-solving time for T_2 decreased on average. Notice this trend in the problem time for 5.4-7.2 in Fig. 6a. For problems 7.3-7.6, T_2 took significantly lower time than T_1 and C while constructing the proofs. From the patterns in T_2 students' problem-solving time and scores, we concluded that learning BW strategy could help students to construct proofs faster and improve their scores.

Step Count: Logic-proof problems within DT can have multiple solutions with different lengths. However, possibly due to having the same level of prior knowledge measured by pretest scores, for most problems, students were observed to construct proofs with similar lengths. The details on step counts are discussed in "Approach

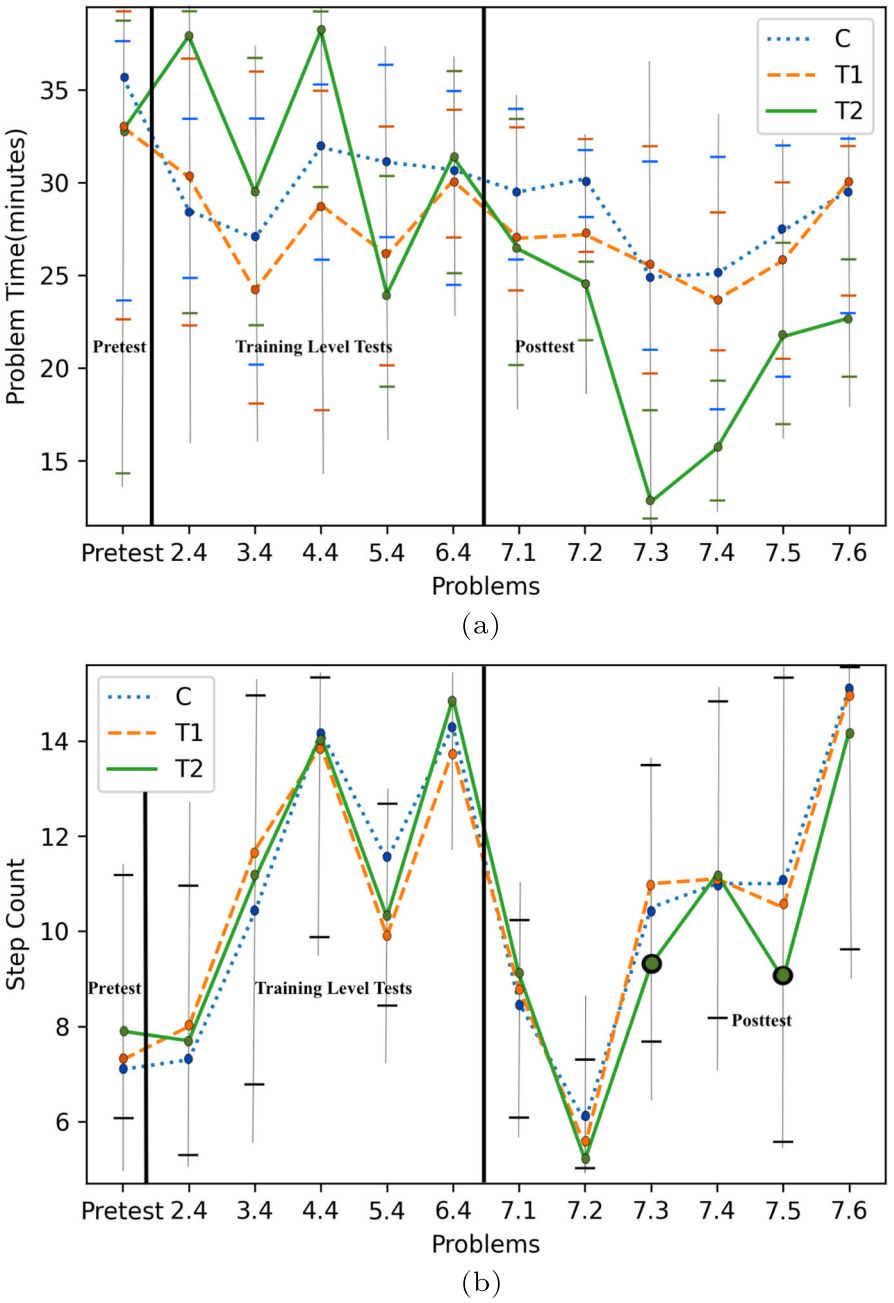


Fig. 6 Trends in (a) Problem-Solving Time; and (b) Step Counts in Pretest, Training Level Tests (2.4-6.4), and Posttest (7.1-7.6) PS Problems across the Three Training Groups

Map Analysis" using Approach Maps. As shown in Fig. 6b, in most of the training, and posttest problems, step counts were similar, and no significant differences were found across the three groups. However, we observed, in problems 7.3, and 7.5, T_2 had significantly fewer steps than C and T_1. Our later approach map analysis ("Scenario 3(Final: Improved Performance of T_2 (Less Time and Shorter Solution) with Fewer Effective BW Steps): Posttest Prob. 7.3", Fig. 10) showed that, in these two problems, % of T_2 students adopting the shortest possible solutions is higher than the other two training groups. These results suggest that learning BW strategy can potentially help students identify shorter optimal solutions. However, this trait could be dependent on the specific problem a student is working on since this pattern was only observed in two posttest problems.

The results of our score-based performance analysis suggest that the combination of BWE and BPS improved students' problem-solving skills, and helped them to perform better in the posttest. T_2 students obtained higher scores by constructing posttest proofs faster, or by constructing shorter posttest proofs in some cases. On the other hand, T_1 students who only received BWE, behaved and performed mostly like the control group C, who were not introduced to the backward strategy at all. In the next section, we investigate in more detail student solution approaches to identify the source of students' improved problem-solving performance as demonstrated by the test scores.

RQ3: Impact on Subgoal Learning

Approach Map Analysis

The performance analysis in "RQ2: Impact on Performance" showed that T_2 students who received both BWE and BPS during training had significant differences with C and T_1 students in performance. Control (C) group received none of BWE and BPS and T_2 group received only BWE during training. Over the period of training and posttest, T_2 eventually outperformed both C and T_1. On the contrary, T_1 showed mostly similar values as C in performance metrics. From the results of the performance analysis, we concluded that the combination of BWE and BPS helped students to learn efficient proof construction over time and achieve higher performance. To investigate how T_2 students achieved higher performance, we generated graphical representations of students' proof construction attempts using *Approach Maps* (Eagle and Barnes, 2014) for problems 2.4-6.4 (training level tests) and 7.1-7.6 (posttests). We analyzed the derivation of each proposition using statistical tests[5] to identify the instances where one training group was more efficient than another. We gave special attention to propositions that were identified as subgoals by experts for each problem to understand if there were differences in the efficiency or approach to derive these subgoals across the groups.

From our analyses of approach maps, and BW action counts ("Student Engagement in Backward(BW) Strategy"), we identified four scenarios: **1)** Poor performance of T_2 co-occurring with a lot of BW actions in the earlier phases of training (2.4-4.4); **2)**

[5] Kruskal Wallis test followed by Mann-Whitney U test with Bonferroni correction, corrected α=0.017

Better performance of T_2 with a few BW steps in the later phases of training (Prob. 5.4); **3)** Significant higher performance of T_2 due to less problem-solving time (7.3-7.6), or **4)** shorter solutions co-occurring with comparatively more BW actions during posttest (7.3, and 7.5). In the subsequent subsections, we first describe the construction method of approach maps and then, analyze approach maps of representative problems for each of the scenarios mentioned above.

From the approach maps, we highlight significant differences in 1) the adopted proof construction approach: to understand if knowing backward strategy influenced students' proof construction approach, 2) step counts: to understand if knowing backward strategy helped students to identify required steps with less exploration through unnecessary steps, and 3) step and problem time: to understand if knowing backward strategy helped students to identify subgoals or the overall approach faster. While analyzing the approach maps, we analyzed both forward and backward-directed approaches of students. The reason for analyzing the forward-directed approach is two-fold: 1) to identify where or when students tried to use BW strategy but failed and then, adopted FW strategy, and 2) to identify potential cases of *Backward Thinking* for implicit subgoaling. *Backward Thinking* refers to the event when students identify required steps or subgoals using BW strategy but carry out the steps in the forward direction in the tutor. Thus, any observed efficiency of the treatment group students (T_1 or T_2) in identifying subgoals or proof construction even in the forward direction could be the result of learning the BW strategy. On the other hand, analysis of the backward-directed approach helps to understand the explicit use of BW strategy or steps for subgoaling and the associated efficiency.

Approach Map Generation Method

An approach map, proposed by Eagle and Barnes (2014), is a graphical representation of students' problem-solving approaches. The steps to construct approach maps are briefly described below. Figures to illustrate these steps can be found in the supplementary materials attached at the end of this manuscript.

Step 1 (Construct Interaction Networks from Students' Action Logs): An interaction network (Eagle et al., 2012) for a problem is essentially a graph consisting of nodes and edges representing all students' problem-solving states and steps. For DT problems, a state is the set of all propositions a student has at any point of the construction of a proof. A state includes both justified and backward-derived unjustified propositions. A step is the addition or deletion of a proposition through the application of a logic rule. Note that the propositions in a state are lexicographically ordered and the order of their derivation is ignored. Considering the order of derivation could increase the number of states exponentially and no complex computation would be feasible on the interaction network.

As students progress in the construction of a proof, they move from state to state via steps. For example, if at state $S_0(\neg(K \wedge M),\ J \Rightarrow (K \wedge L),\ L \Rightarrow M)$, the DeMorgan's rule on $\neg(K \wedge M)$ is applied, the new state will be $S_1(\neg(K \wedge M),\ \neg K \vee \neg M,\ J \Rightarrow (K \wedge L),\ L \Rightarrow M)$. An incorrect rule application can result in the previous state and the next state being the same. The tuple (current state, step, next state) is called an interaction. So, a student's solution attempt for a logic-proof problem is a directed

graph of interactions. Our code implementation generates an interaction network for a problem by conjoining all the interactions from all student attempts to construct a proof for the problem. There is a single start node in the interaction network containing the given premises. Since there are multiple solutions for a problem, there can be multiple end states in an interaction network where each end state contains the justified goal statement. Additionally, to facilitate statistical analyses on the network, the interaction network includes data on the frequency of node and edge visits, time spent on or before each interaction, and step counts before each interaction across the three training groups.

Step 2 (Girvan-Newman Clustering Algorithm): Students do not have expert-like prior knowledge. Thus, they often require exploration leading to the derivation and deletion of unnecessary/incorrect propositions along with correct propositions during a proof construction attempt. These derivations and deletions form visible clusters or regions in the interaction network where the major outcome of these regions are the proposition(s) contributing to the final proof. Also, different approaches to solving the same problem can result in different regions. To identify these regions, the approach map technique applies the Girvan-Newman community clustering algorithm (Girvan and Newman, 2002) on interaction networks. The clustering algorithm takes as input an interaction network with start node, end nodes, and self-loops[6] removed. Additionally, edge weights are assigned based on the cumulative visit frequency of the corresponding interaction. At each iteration of the algorithm, the edge with the highest edge-betweenness is removed from the network. Edge betweenness (Cuzzocrea et al., 2012) of an edge is calculated by calculating the shortest paths between all pairs of nodes and counting the number of shortest paths that go through that edge. Then, the connectivity of the resulting graph is calculated using modularity score (Newman, 2010) . Each connected component of the resulting graph is marked as a region. This process is continued till there is no edge left to be removed. The output of the algorithm is the graph with the highest modularity score and the clusters/regions are the connected components within that graph.

Step 3 (Approach Map Generation from Clustered Network): The clustered interaction network for any logic proof problem is a fairly large graph where student approaches to solve the problem cannot be visually detected. Thus, we simplified the clustered networks to approach maps using the method adopted by Eagle et al. (2012). First, we added the start and end nodes back to the clustered interaction network and then applied the following steps: **1)** Represent each region with a single node. These regional nodes are labeled with the proposition(s) with the highest number of incoming and outgoing edges from and to other regions, and all propositions derived to generate the latter from the former. This step filters out unnecessary propositions derived by the students and keeps only the ones contributing to the proof. However, we kept a record of the counts of unnecessary propositions; **2)** Combine parallel edges, and actions between regions to a single edge with a composite action label; and **3)** Keep only unique paths between the start and goal nodes. These three steps convert a clustered interaction network to a pseudo-graph called an approach map, where the start node is connected to the goal node via region nodes. A path from the start node

[6] Edges originating and ending at the same node

to a goal node represents a solution approach, where the propositions contributing to the solution can be visually identified from the labels of the regional nodes in between.

Approach Map Presentation: In the approach maps presented in this paper, we only showed the most common student solution approaches for simplicity and used them to discuss differences found across the training groups. Different aspects of approach maps are shown in the approach map for problem 2.4 in Fig. 7. Each path that connects the start and goal nodes contains a unique set of derived correct propositions (shown within the region nodes, R1, R2, etc). These propositions contribute to an approach (labeled as A1, A2, etc.). Nodes are connected through edges with arrows showing the derivation direction (FW or BW) with BW arrows colored red. Edge thickness and color are based on frequency. Frequent edges are thicker and colored blue and non-frequent edges are colored black and of unit thickness. Note that students derived a wide variety of unnecessary propositions during proof construction. Since they are novices, most students had at least one unnecessary proposition in all problems. We did not show those unnecessary propositions in the approach maps to keep them simple. However, we recorded counts of unnecessary propositions across different groups to identify the more efficient training group. In the subsequent section, we explain approach maps for problems representing different scenarios.

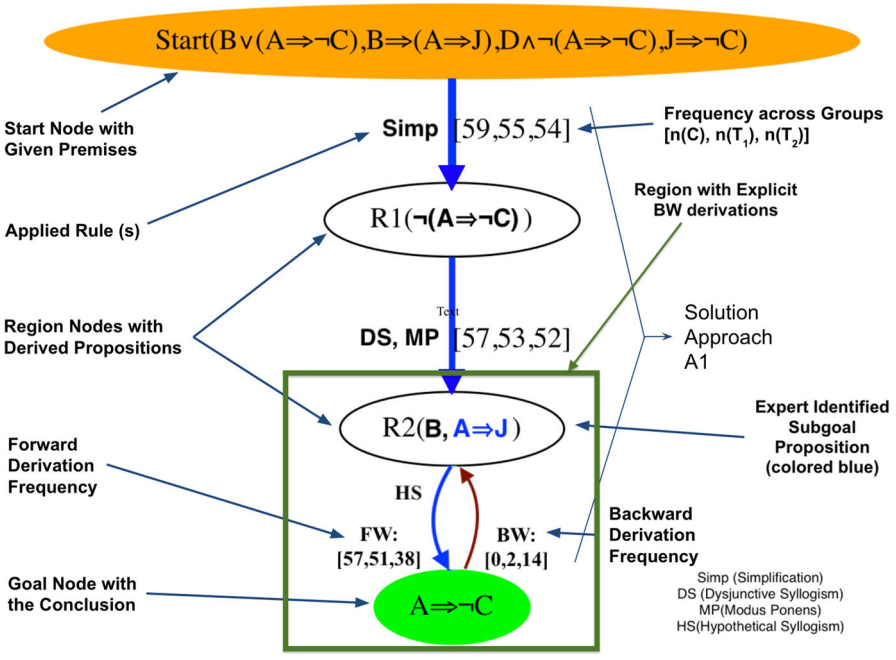


Fig. 7 Approach Map for Problem 2.4 [Note: Bold-faced propositions have significant differences in derivation efficiency across the training groups. The last node of any region with incoming BW-directed red edges is BW derived. Ex., A⇒J was BW derived by students in this problem. The green square highlights the area with BW derivations.]

Scenario 1 (Early in Training: Poor Performance of T_2 Co-occurring with Many BW actions): Training Prob. 2.4

Problem 2.4 asks students to derive the goal $A \Rightarrow \neg C$ from the given premises: $B \vee (A \Rightarrow \neg C)$, $B \Rightarrow (A \Rightarrow J)$, $D \wedge \neg(A \Rightarrow \neg C)$, and $J \Rightarrow \neg C$. In the approach map of this problem [Fig. 7], we show the most common three-proposition solution approach for this problem. This solution was adopted by 96% of the students and labeled as A1 in the figure [Start $\rightarrow$ R1($\neg(A \Rightarrow \neg C)$) $\rightarrow$ R2 ($B, A \Rightarrow J$) $\rightarrow$ Goal]. Note that in this approach, $A \Rightarrow J$ was identified as a subgoal by experts and is marked blue in Fig. 7. As explained in "Approach Map Analysis", we analyzed how each student group used forward and backward approaches for this problem.

Approach with Forward Steps: Most students in the three training groups (with 57, 51, and 28 students in groups C, T_1, and T_2, respectively) worked forward (FW) to derive all 3 propositions. The FW derivations are shown by the blue edges in Fig. 7. In this problem, T_2 group tried to work backward more and had more BW actions [Table 3], but 14 of them were able to successfully derive backward steps – these students are discussed in the next paragraph. Students in the T_2 group who worked forward also derived more unnecessary propositions before deriving the correct steps [before $\neg(\mathbf{A} \Rightarrow \neg\mathbf{C})$, mean unnecessary proposition count (C, T_1, T_2)= ~1,~1, and ~4; $P_{MW}(T_2 > C)$=0.001; $P_{MW}(T_2 > T_1)$=0.002]; and before **B**, mean (C, T_2)= ~4, and ~6; $P_{MW}(T_2 > C)$=0.009]. Because of these extra steps and BW actions, students in the T_2 group who worked forward also took almost twice as long to derive each of the three propositions [$\neg(\mathbf{A} \Rightarrow \neg\mathbf{C})$ [mean(C, T_1, T_2)= 4.98, 5.77, and 9.36 mins.; $P_{MW}(T_2 > C)$=0.0005; $P_{MW}(T_2 > T_1)$=0.0008], **B** [mean(C, T_1, T_2)= 4.33, 4.37, and 9 mins.; $P_{MW}(T_2 > C)$=0.003; $P_{MW}(T_2 > T_1)$=0.011], and $\mathbf{A} \Rightarrow \mathbf{J}$ [mean(C, T_1, T_2)= 6.98, 5.77, and 12.36 mins.; $P_{MW}(T_2 > C)$=0.006; $P_{MW}(T_2 > T_1)$=0.015]. We concluded that in the early levels of training, these T_2 students were still adapting to the BW skill, as shown by failing to derive BW steps from BW actions. They also derived extra unnecessary propositions and took extra time to derive the correct propositions needed to solve the problem. In the instances of failed BW derivation attempts, group T_2 switched to a forward strategy.

Approach with Backward Steps: In problem 2.4, 0, 2, and 14 students had successful backward derived steps across C, T_1, and T_2 respectively. As shown by the backward-directed edge from the conclusion to region R2 in Fig. 7, these students only backward derived $A \Rightarrow J$ which is an expert-identified subgoal for this problem. They derived the other 2 propositions, $\neg(A \Rightarrow \neg C)$ and B, forward. The 2 T_1 students who had BW steps had an average step count of 7.15 and problem time of 35.1 minutes. Since only a few students in C and T_1 derived BW steps, a valid statistical comparison between them and T_2 students with BW steps could not be carried out. Therefore, we compare the backward T_2 approaches with the forward C and T_1 approaches. The 14 T_2 students who derived $A \Rightarrow J$ backwards had significantly fewer unnecessary steps than C and T_1 [mean unnecessary step counts(C, T_1, T_2)= 7, 6, and

4; $P_{MW}(T_2 < C)$=0.0005; $P_{MW}(T_2 < T_1)$=0.011]. But, T_2 students still needed more time to construct the proof with BW steps than what C and T_1 groups took to derive the proof in the forward direction [mean BW derivation time for $T_{2_{BW}}$ = 38.9 mins.; $P_{MW}(T_2 > C)$=0.001; $P_{MW}(T_2 > T_1)$=0.006]. These comparisons between $T_{2_{BW}}$ students' approaches and C or T_1 students' forward-directed approaches emphasized that those who successfully learned BW strategy with BWE+BPS and were successful in applying it required less exploration before figuring out the right subgoal than those who did not learn the strategy. However, this problem was only given after the first level of training and thus, most T_2 students were still not efficient enough in using the BW strategy. Also, T_2 students used BW derivation only to derive the subgoal and derived the rest of the steps using FW derivation, unlike the training BPS/BWE where all steps are done backward. Recall that combining strategies in this way is often observed in expert problem-solving approaches as mentioned in prior literature. Note that in this analysis, overall, no significant differences were found between the solution attempts of T_1 and C.

Scenario 2 (Later in Training: Improved performance of T_2 with only a Few BW Steps): Training Prob. 5.4

Problem 5.4 asks students to derive the goal $\neg J$ from the given premises: $\neg(K \wedge M)$, $J \Rightarrow (K \wedge L)$, and $L \Rightarrow M$. From the approach map shown in Fig. 8, we identified 6 solution approaches, labeled A1 - A6. The approaches and associated expert-identified subgoals are shown in the figure.

Approach with Forward Steps: Among the 6 approaches, A2 and A4 are the two most common strategies that cover 71% of all students. For exact student counts across different training groups who followed these approaches notice the frequen-

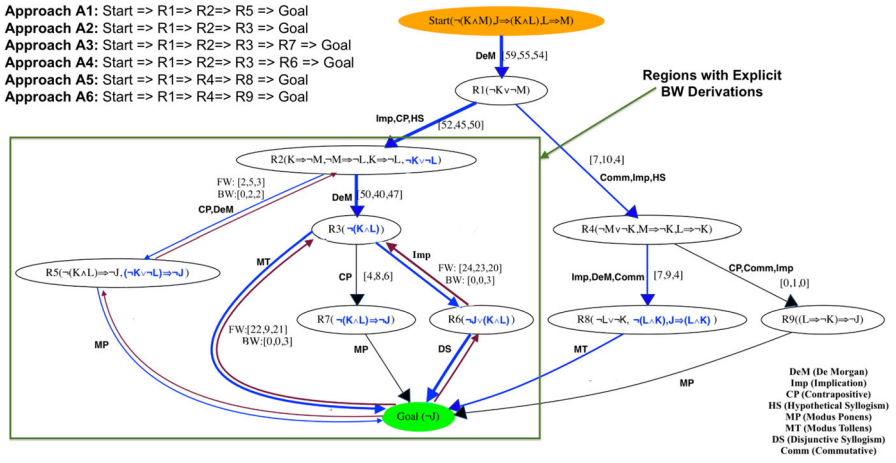


Fig. 8 Approach Map for Problem 5.4. [On the top-left, contributing regions for different approaches or paths are shown. Edges that are missing frequencies have the same frequencies as the edges immediately above them. For example, R3⇒R7 and R7⇒Goal have the same frequencies across training groups]

cies attached to the edges. In these two approaches, the common expert subgoal is to derive $\neg(K \wedge L)$ using Modus Ponens on $\neg K \vee \neg L$. The subgoal is marked blue in region R3 in the figure. When we analyzed student approaches where all steps used forward derivations, we observed that group T_2 derived both of these propositions in significantly fewer steps than group C [$\neg\mathbf{K} \vee \neg\mathbf{L}$: mean(C, T_2)= 9.16, and 7.42; $P_{MW}(T_2 < C)$=0.011; $\neg(\mathbf{K} \wedge \mathbf{L})$: mean(C, T_2)= 10.44, and 8.17; $P_{MW}(T_2 < C)$=0.003]. This suggests that group C required more exploration than T_2, as measured by unnecessary steps before figuring out what to derive to achieve the final goal. We did not find any significant differences in the step counts of T_1 and T_2 while deriving these two subgoal propositions. The third most frequent approach is A5. In the final step of approach A5, Modus Tollens is applied on $\neg(L \wedge K)$, and $J \Rightarrow (L \wedge K)$ to derive the final goal $(\neg J)$. Experts identified $\neg(L \wedge K)$ [in region R8] as a subgoal in this approach. Group T_2 outperformed both C, and T_1 in deriving $\neg(L \wedge K)$ in terms of problem-solving time [mean(C, T_1, T_2)= 26.07, 20.97, and 15.48 mins.; $P_{MW}(T_2 < C)$=0.002; $P_{MW}(T_2 < T_1)$=0.014].

Approach with Backward Steps: We found only 2 T_1 students and 8 T_2 students derived expert-identified subgoals using BW derivation. Here, the subgoals are $(\neg K \vee \neg L, \neg K \vee \neg L \Rightarrow \neg J)$ in A1, or $\neg(K \wedge L)$ in A2, or $(\neg(K \wedge L), \neg J \vee (K \wedge L))$ in A4. The BW derivations are shown by the BW-directed edges and corresponding frequencies in A1, A2, and A4 in Fig. 8. In this problem, T_2 again used BW derivation to only derive the subgoals. They derived the rest of the steps in the forward direction. T_2 students who derived these subgoals using BW derivation outperformed C in terms of both step count, and problem-solving time [**Step Count:** mean(C, T_2)=11.2, and 8.5; $P_{MW}(T_2 < C)$=0.0001; **Problem-Solving Time:** mean(C, T_2)= 31.1, and 25.85 mins.; $P_{MW}(T_2 < C)$= 0.008]. However, the 2 T_1 students who used the backward strategy had average step counts (10.9) and times (29.7 mins.) that are similar to C but higher than T_2.

Overall, our approach analysis of problem 5.4 showed that most T_2 students did not explicitly derive BW steps. However, they were efficient in deriving the subgoal propositions even when they only used FW steps. This improvement was mostly observed for T_2 students who received BWE+BPS during training. Thus, we concluded that possibly T_2 students used their improved skills on backward strategy implicitly (i.e. *BW Thinking*) to identify subgoals and outline the entire proof. Then, they constructed the proof using forward steps in less time and with fewer unnecessary steps. Also, the few students who explicitly derived subgoals using BW steps were more successful than they were in the previous problem (prob. 2.4) where they needed more time to work backward. In this problem, T_2 students with BW steps had fewer unnecessary propositions and less time. We concluded that by the 5th level of training, T_2 students who received both BWE and BPS, were better adapted to using BW strategy for explicit or implicit subgoaling. However, in this problem, although T_1 received higher average scores than C [Fig. 5], the difference in the scores of C and T_1 was not significant according to statistical tests. This suggests that BWE is not as effective as BWE+BPS in helping students effectively learn BW strategy by level 5.

Scenario 3(Final: Improved Performance of T_2 (Less Time and Shorter Solution) with Fewer Effective BW Steps): Posttest Prob. 7.3

In the final posttest level, problem 7.3 asks students to derive the goal $\neg H$ from the given premises: $\neg(K \wedge E)$, $A \Rightarrow E$, and $H \Rightarrow (K \wedge A)$. The approach map for this problem [Fig. 9] shows the three most commonly adopted solution approaches labeled A1, A2, and A3. The blue-colored propositions refer to the expert-identified subgoals. These approaches cover 96% of all students.

Approach with Forward Steps: The common expert-identified subgoal in the approaches A1, A2, and A3 is $\neg(K \wedge A)$ or $\neg(A \wedge K)$ in region R4. And, while constructing proofs with only forward steps, T_2 derived $\neg(K \wedge A)$ or $\neg(A \wedge K)$ with significantly fewer unnecessary steps than group C and T_1 [**for** $\neg(\mathbf{K} \wedge \mathbf{A})$, mean(C, T_1, T_2)= 9.45, 9.46, and 7.38; $P_{MW}(T_2 < C)$=7.91e-05; $P_{MW}(T_2 < T_1)$=0.007; **for** $\neg(\mathbf{A} \wedge \mathbf{K})$, mean(C, T_2)= 9.19, and 7.64; $P_{MW}(T_2 < C)$=0.01662]. Moreover, T_2 derived $\neg A \vee E$ significantly earlier in their solution attempt than C [mean(C, T_2)= 9.60, and 4.53 mins.; $P_{MW}(T_2 < C)$=0.009]. From the approach map, we can see that $\neg(K \wedge A)$ comes from $\neg K \Rightarrow \neg A$ which is derived by applying Hypothetical

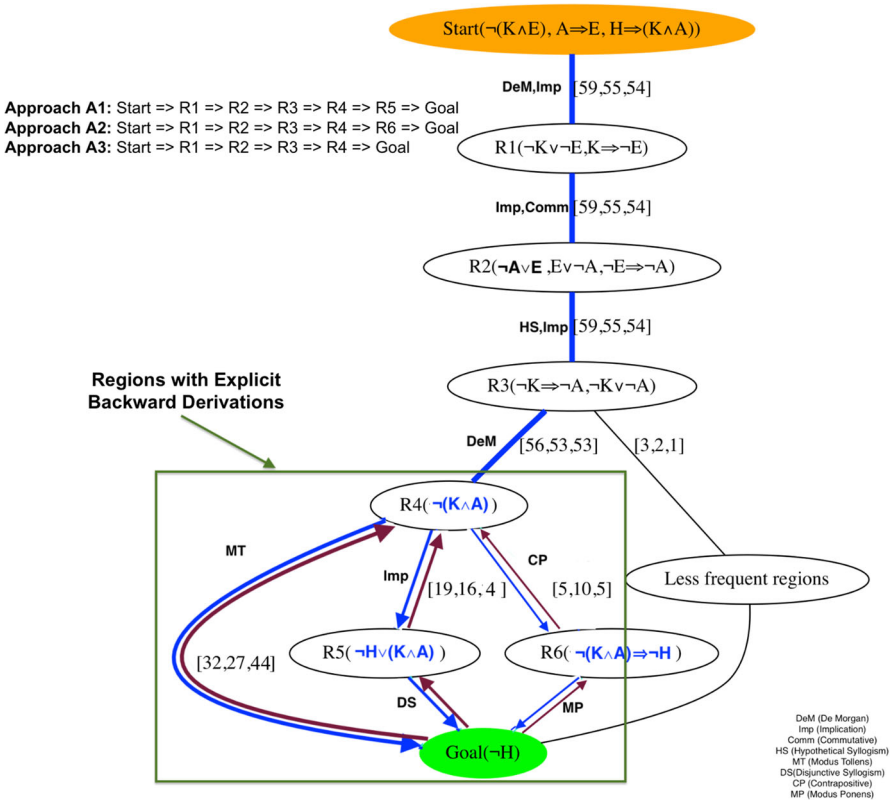


Fig. 9 Approach Map for Problem 7.3.

Syllogism on $K \Rightarrow \neg E$ and $\neg E \Rightarrow \neg A$. And, early derivation of $\neg A \vee E$ suggests that T_2 students also figured out the approach to derive $\neg(K \wedge A)$ quicker than other groups. These statistics show that as in later phases of training (prob. 5.4), T_2 students derived subgoals in problem 7.3 efficiently while working in the forward direction. This efficiency could be the result of implicit subgoaling using a BW strategy.

Approach with Backward Steps: We identified 22 T_2 students ($\sim$ 41% of T_2 students) who explicitly backward-derived $\neg(K \wedge A)$ defining it as a subgoal. For this derivation, they used Modus Ponens with $\neg(K \wedge A) \Rightarrow \neg H$, Modus Tollens with $H \Rightarrow (K \wedge A)$ given, or Disjunctive Syllogism with $\neg H \vee (K \wedge A)$. T_2 students who derived subgoals using BW steps had significantly less step count and spent less time than the average of groups T_1 and C [**step count:** mean(C, T_1, T_2)= 10.5, 11.0, and 7.62; $P_{MW}(T_2 < C)$=1.28e-05; $P_{MW}(T_2 < T_1)$=0.0006; **problem time:** mean(C, T_1, T_2)= 24.9, 25.5, and 15.6 mins.; $P_{MW}(T_2 < C)$=0.0001, $P_{MW}(T_2 < T_1)$=0.009]. We did not find any significant differences in step counts and problem-solving time between T_2 students who used BW derivation (problem time: 15.6 mins., step count: 7.62) and those who did not (problem time: 13.9 mins., step count: 9.6). These statistics suggest that there are no significant differences between the efficiency of T_2 students who used BW steps explicitly and those who did not use the strategy or might be using it implicitly. However, C and T_1 students had significantly higher step count and time than those of T_2 indicating their lower efficiency in proof construction. We found one group C student and three T_1 students who derived $\neg(K \wedge A)$ in the backward direction. However, we observed these 4 students had an average step count of 16.20 and a problem-solving time of 31.7 mins. which is a lot higher than T_2 students using BW derivation. This indicates that group C or T_1 was not efficient in BW derivation.

Apart from the efficient subgoaling in both forward and backward-directed approaches, as shown in Fig. 10, most T_2 students adopted the shortest 8-step approach A3 while solving this problem. A similar pattern was observed for problem 7.5 where 65% T_2 students adopted the shortest 6-step solution, whereas the percentages for C, and T_1 were only 35%, and 41% respectively.

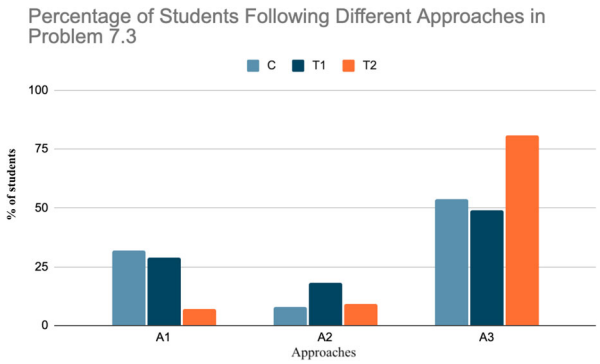


Fig. 10 Percentage of Students across the Three Training Groups Adopting Different Approaches

Overall, in this problem, T_2 students engaged in BW derivations comparatively more than C and T_1, and they did so efficiently. They also continued to show improved subgoaling behavior both in forward and backward-directed approaches. In addition to fewer unnecessary steps and less time, in this problem, we observed T_2 students be more driven toward the shortest solution. However, none of the students constructed the entire proof backward. They used backward steps to derive the subgoals only.

Scenario 4(Final: Improved Performance of T_2 (Less Time) with Fewer Effective BW Steps): Posttest Prob. 7.6

Problem 7.6 asks to derive $D \Rightarrow C$ from $A \vee \neg D$, $A \Rightarrow (B \Rightarrow C)$, and B. The approach map for this problem [Fig. 11] shows the two most commonly adopted solution approaches, A1 and A2, adopted by $\sim$ 79% of all students.

Approach with Forward Steps: The common expert-identified subgoal in both A1 and A2 is $D \Rightarrow A$ (from region R1), which is only two steps away from the start point. We did not find any differences in the derivation of this subgoal across our training groups. The other expert subgoal for A1 is $\neg D \vee C$ (from region R3). When students only used forward steps, T_2 students required significantly less time than the other two groups to derive $\neg D \vee C$ [mean(C, T_1, T_2)= 21.7, 26.2, and 14.2 mins.; $P_{MW}(T_2 < C)$=0.003; $P_{MW}(T_2 < T_1)$=0.008;]. Also, T_2 students adopting approach A2, derived $A \Rightarrow C$, the expert-identified subgoal for this approach, in significantly less time than the other two groups [mean(C, T_1, T_2)= 22.5, 24.9, and 17.1 mins.; $P_{MW}(T_2 < C)$=0.001; $P_{MW}(T_2 < T_1)$=0.004].

Approach with Backward Steps: Twelve (12) T_2 students who adopted approach A1 derived the subgoal $\neg D \vee C$ in BW direction. We observed that group T_2 derived this

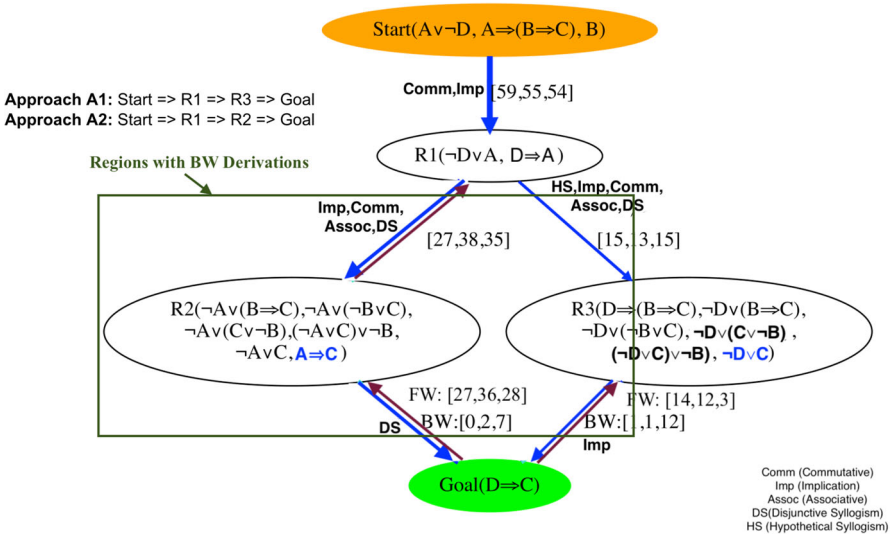


Fig. 11 Approach Map for Problem 7.6.

subgoal with significantly fewer steps than C, and T_1 [mean(C, T_1, T_2)= 16.8, 17.62, and 10.4; $P_{MW}(T_2 < C)$=0.001; $P_{MW}(T_2 < T_1)$=0.002]. They were also observed to derive propositions required to derive $\neg D \vee C$ significantly **earlier** in their solution attempt than T_1 and C [Ex., **for** $\neg$**D** $\vee$ (**C** $\vee$ $\neg$**B**), mean(C, T_1, T_2)= 22.6, 25.2, and 15.3 mins.; $P_{MW}(T_2 < C)$=0.009; $P_{MW}(T_2 < T_1)$=0.007; **for** ($\neg$**D** $\vee$ **C**) $\vee$ $\neg$**B**, mean(C, T_1, T_2)= 23.0, 25.6, and 18.08 mins.; $P_{MW}(T_2 < C)$=0.009; $P_{MW}(T_2 < T_1)$=0.005]. Additionally, T_2 had significantly less time gap in between consecutive steps than C [for example, between $\neg$**D** $\vee$ (**C** $\vee$ $\neg$**B**) and its preceding step, mean(C, T_2)= 7.10, and 3.25 mins.; $P_{MW}(T_2 < C)$=0.012]. Seven (7) T_2 students adopting approach A2 explicitly derived subgoal $A \Rightarrow C$ backwards. Overall, when approach A2 was adopted, T_2 students discovered subgoal $A \Rightarrow C$ with significantly fewer unnecessary proposition derivations than that of C [mean unnecessary proposition count before deriving $A \Rightarrow C$: for C, and T_2 = 14.90, and 11.25; $P_{MW}(T_2 < C)$=0.001].

These analyses of students' forward and backward-directed approaches in problem 7.6 show that like previously described problems, T_2 students using only FW steps continued to show improvement in subgoaling. When using BW steps, T_2 students identified complex subgoals early in their solution attempts with fewer unnecessary propositions, and in less time. They were also able to figure out a plan to derive those subgoals as indicated by early derivation of prerequisite propositions and quicker consecutive steps. We concluded that having the BW skill motivated implicit or explicit subgoaling, and helped to identify an outline of the solution to the problems in less time. However, explicit BW derivations were still low and observed only in some steps in 19 out of 54 ($\sim$ 35%) T_2 students' solutions. Since it is common for experts to work forward with implicit BW thinking, we consider this a success for T_2 learning.

Summary of Approaches in Posttest Problems

Overall, in the posttest problems, more T_2 students (17-41% students per problem) used explicit backward steps in their final solution than the other two groups. Only a few students in group C (0-3 students per problem) and T_1 (0-4 students per problem) had backward steps in the posttest. Figure 12a shows the counts of students across the three training groups who had at least one BW-derived step in their final solutions of the posttest problems. However, T_2 students with and without explicit BW steps were more efficient than C and T_1 in the posttest. Overall, in all posttest problems, T_2 outperformed the other two groups. They constructed the proofs with significantly less average step time (T_2:2.04, T_1:2.91, and C: 3.08 mins.; P_{KW}=0.014) and marginally less unnecessary proposition count (T_2:3.3, T_1:3.9, and C:4; P_{KW}=0.045) than the other two groups during posttest [Fig. 12b]. Thus, we conclude that T_2 students trained with BWE+BPS were able to identify required subgoals and outline the correct approach to construct proofs faster by using BW strategy implicitly (i.e. BW thinking) or explicitly. They also required less exploration through incorrect propositions, which also contributed to their improved performance.

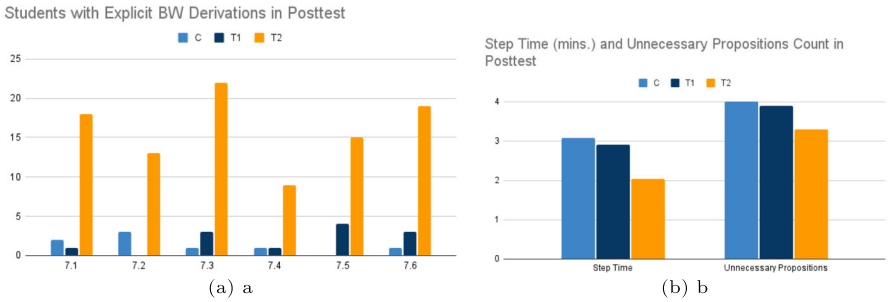


Fig. 12 Histograms for (a) Student Counts with Explicit BW Derivation, and (b) Average Step Time (minutes) and Unnecessary Proposition Counts in Posttest Problems across C, T_1, and T_2

Discussion

In this study, we explored BWE and BPS within an intelligent logic tutoring system to help students adapt to backward strategy use with an aim of improving their subgoaling and problem-solving skills. Our results showed the effectiveness of training students with a combination of BWE and BPS for backward strategy learning and revealed important insights on how backward strategy learning impact students' competence in problem solving. We have summarized the major findings below.

***RQ1* (Training Struggle and Increased BW Strategy Usage):** Our results showed that to solve BPS problems during training T_2 students showed signs of struggle - needing more time, sessions, and restarts especially in the earlier training levels. T_1 students who were simply shown BWE but were not required to perform BPS, avoided these struggles and therefore did not learn the BW strategy by the end of the tutor. Note that prior studies showed that backward derivations are difficult for students during training, or when they are still learning (Matsuda and VanLehn, 2005) . And, BPS problems required backward derivations in all the steps. Thus, the increased time, session, and restart counts in BPS problems given during training conform to the results of prior research. However, the performance of T_2 over the period of training and posttest indicates that they eventually overcome the difficulties associated with BW derivation and were able to use it for efficient proof construction. Also, note that our ultimate goal of the training was to improve students' subgoaling and problem-solving skills so that they can efficiently solve new problems. And, T_2 students who learned backward strategy using both BWE and BPS performed better in posttest than those who received only BWE (T_1) and those who were not exposed to BW startegy at all (control C). Thus, we concluded that although BPSs were time-consuming and difficult for students during training, they had a key role in improving T_2 students' skills to solve new problems. Matsuda and VanLehn (2005) taught students the BW strategy using only 6 training proof-construction problems and other non-proof problems and then, explored its impact using posttest problems. In our study, the problem organization for T_2 had 4 differences from the training session shown in the Matsuda and VanLehn study: 1) the training session contained 20 training proof-construction problems, 2) example proofs (BWEs) solved using backward strategy were shown before having T_2 students solve BPSs, 3) then, three

BPSs were given in the first half of the training, each of which engaged the T_2 students in backward strategy in all the steps, and 4) in the second half of the training, T_2 students were given the opportunity to use any strategy during proof construction at their own discretion. No BPS were given in this phase and the students got a chance to adapt the BW strategy independently. However, BW action counts across the training groups demonstrated that, when T_2 students solved problems independently in the test problems, they voluntarily engaged in backward derivations comparatively more than the other two groups and eventually outperformed the other groups. Thus, we also conclude that the combination of demonstration through examples (BWEs), practice (through BPSs), and opportunity to adapt to the strategy might be necessary before students could use backward strategy to solve problems efficiently.

***RQ2* (Improved Problem Solving Achieved Over Time):** The results from our performance analyses showed that the combination of demonstration (BWE), practice (BPS), and additional problems in the second half of the training to use backward strategy independently helped students to learn the strategy better. This training strategy helped students to adapt with the strategy and use it to improve their problem-solving performance. The improved performance was observed in higher scores, decreased problem-solving time, and step counts. However, improved performance was not observed immediately after students were exposed to the BW strategy. In the earlier phases of training (2.4-4.4), we observed T_2 to spend significantly more time while solving simple problems leading to lower scores. As training progressed, T_2 increasingly became more efficient in problem solving and outperformed C, and T_1. Recall that BWE/BPSs were given to students mostly during the first half of training [Fig. 4]. However, T_2 students continued to improve throughout later phases of training, and posttests. This pattern suggests that to be efficient in using the BW strategy, students may need additional problem solving during training where they can explore the BW strategy independently. The additional training could help them to become proficient with the strategy and successfully incorporate it into their problem-solving method.

***RQ3* (Improved Subgoaling Skill):** Our approach map analysis revealed that T_2 students derived expert-identified subgoal propositions more efficiently. They derived subgoals with less time and fewer unnecessary derivations than the other two groups. On the other hand, T_1 did not show any significant or consistent evidence of improved subgoaling, possibly due to superficial exposure to BW strategy through BWEs only.

We observed, although T_2 students engage in BW derivations more than control C, and T_1, they never constructed entire proofs backward. Rather, they used explicit BW derivations to subgoal only and FW steps to derive those subgoals. Also, not all T_2 students derived *explicit* BW steps. The majority of T_2 students demonstrated only FW derivations. However, we have demonstrated that T_2 showed efficient subgoaling even when they used only forward derivations. This suggests that these students are potentially applying an implicit BW strategy, forming subgoals using BW thinking, but generating nodes only in the forward direction, much like experts often do. But, one major limitation of this analysis is that we can not verify the implicit use of BW strategy from log data collected from the tutor. However, since the improved subgoaling behavior was only observed for T_2 students and not for the other two training groups,

we conclude that backward strategy learning through a training session as the one T_2 received may help improve students' problem-solving and subgoaling skills.

Conclusion and Future Works

In this paper, we explored backward strategy learning as a way to improve students' subgoaling and problem-solving skills. From our results, we conclude that a combination of examples, practice through problem solving, and enough training with additional independent problem solving is necessary to help and motivate students to learn to use backward strategy. Students who learned backward strategy in this fashion (i.e. the T_2 students) demonstrated efficiency in deriving subgoals. They identified and derived subgoals in less time, and less exploration through unnecessary propositions which led to an improved problem-solving performance. Thus, this training strategy can be adopted by tutors for structural problem solving like logic, mathematics, geometry, etc. However, in this study, most T_2 students showed hints of implicit BW strategy use rather than carrying out explicit BW derivations. To validate the reasoning that the implicit use of backward strategy led to efficient subgoaling, future studies are required that involve interviewing students or a talk-aloud protocol where students explain their process of deriving subgoals after learning backward strategy. Also, as a future work, mixed FW/BW problems where students need to carry out some steps backward instead of all backward steps can be explored for backward strategy learning. Mixing FW and BW strategies might ensure an appropriate level of cognitive engagement and complexity promoting learning while reducing training struggles.

Supplementary information Details of statistical analyses results and Approach map generation procedure can be found in this link: https://1drv.ms/b/s!AiFP3IlZKTk2gQCFUDDcqxdmkiH4?e=jMp3DF. Demonstration videos for BWE and BPS can be found in the following links: **BWE** (https://1drv.ms/v/s!AiFP3IlZKTk2cG3tMr0XCm9CnCY); **BPS** (https://1drv.ms/v/s!AiFP3IlZKTk2cWNBVuCfzmaEdsE)

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Preya Shabrina. The first draft of the manuscript was written by Preya Shabrina and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding The work was supported by NSF grant 2013502.

Declarations

Conflicts of interest The authors have no relevant financial or non-financial interests to disclose.

References

Abdelshiheed, M., Maniktala, M., Ju, S., Jain, A., Barnes, T., & Chi, M. (2021). Preparing unprepared students for future learning. In: *Proceedings of the 43rd Annual Conference of the Cognitive Science Society*, vol. 43

Abdelshiheed, M., Zhou, G., Maniktala, M., Barnes, T., & Chi, M. (2020). Metacognition and motivation: The role of time-awareness in preparation for future learning. In: *Proceedings of the 42nd Annual Conference of the Cognitive Science Society*, vol. 42

Al-Ajlan, A. (2015). The comparison between forward and backward chaining. *International Journal of Machine Learning and Computing, 5*(2), 106.

Anderson, J. R. (2014). The geometry tutor and skill acquisition. In: *Rules of the Mind* (pp. 175–192). Psychology Press, ???

Catrambone, R. (1994). Improving examples to improve transfer to novel problems. *Memory & cognition, 22*(5), 606–615.

Catrambone, R. (1995). Aiding subgoal learning: Effects on transfer. *Journal of educational psychology, 87*(1), 5.

Catrambone, R. (1996). Generalizing solution procedures learned from examples. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 22*(4), 1020.

Catrambone, R. (1998). The subgoal learning model: Creating better examples so that students can solve novel problems. *Journal of experimental psychology: General, 127*(4), 355.

Catrambone, R., & Holyoak, K. J. (1990). Learning subgoals and methods for solving probability problems. *Memory & Cognition, 18*(6), 593–603.

Chi, M. T., Feltovich, P. J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive science, 5*(2), 121–152.

Cody, C., & Mostafavi, B. (2017). Investigating the impact of unsolicited nextstep and subgoal hints on dropout in a logic proof tutor. In: *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 705–705)

Cody, C., Mostafavi, B., & Barnes, T. (2018). Investigation of the influence of hint type on problem solving behavior in a logic proof tutor. In: *International Conference on Artificial Intelligence in Education* (pp. 58–62). Springer

Cuzzocrea, A., Papadimitriou, A., Katsaros, D., & Manolopoulos, Y. (2012). Edge betweenness centrality: A novel algorithm for qos-based topology control over wireless sensor networks. *Journal of Network and Computer Applications, 35*(4), 1210–1217.

Eagle, M., & Barnes, T. (2014). Exploring differences in problem solving with datadriven approach maps. In: *Educational Data Mining 2014*

Eagle, M., Johnson, M., & Barnes, T. (2012). Interaction networks: Generating high level hints based on network community clustering. *International Educational Data Mining Society*

Gick, M. L. (1986). *Problem-solving strategies. Educational psychologist, 21*(1–2), 99–120.

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences, 99*(12), 7821–7826.

Harrison, X. A., Donaldson, L., Correa-Cano, M. E., Evans, J., Fisher, D. N., Goodwin, C. E., Robinson, B. S., Hodgson, D. J., & Inger, R. (2018). A brief introduction to mixed effects modelling and multi-model inference in ecology. *PeerJ, 6*, 4794.

Heyworth, R. M. (1999). Procedural and conceptual knowledge of expert and novice students for the solving of a basic problem in chemistry. *International Journal of Science Education, 21*(2), 195–211.

Jensen, R. J. (1987). Stuck? don't give up! subgoal-generation strategies in problem solving. *The Mathematics Teacher, 80*(8), 614–634.

Laird, J., Rosenbloom, P., & Newell, A. (2012). *Universal Subgoaling and Chunking: The Automatic Generation and Learning of Goal Hierarchies* (Vol. 11). USA: Springer.

Larkin, J., McDermott, J., Simon, D. P., & Simon, H. A. (1980). Expert and novice performance in solving physics problems. *Science, 208*(4450), 1335–1342.

Margulieux, L. E., & Catrambone, R. (2014). Improving problem solving performance in computer-based learning environments through subgoal labels. In: *Proceedings of the First ACM Conference on Learning@ Scale Conference* (pp. 149–150)

Margulieux, L., & Catrambone, R. (2017). Using learners' self-explanations of subgoals to guide initial problem solving in app inventor. In: *Proceedings of the 2017 ACM Conference on International Computing Education Research* (pp. 21–29)

Margulieux, L. E., & Catrambone, R. (2021). Scaffolding problem solving with learners' own self explanations of subgoals. *Journal of Computing in Higher Education,* 1–25

Margulieux, L. E., Guzdial, M., & Catrambone, R. (2012). Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. In: *Proceedings of the Ninth Annual International Conference on International Computing Education Research* (pp. 71–78)

Margulieux, L. E., Catrambone, R., & Guzdial, M. (2016). Employing subgoals in computer programming education. *Computer Science Education, 26*(1), 44–67.

Marwan, S., Gao, G., Fisk, S., Price, T. W., & Barnes, T. (2020). Adaptive immediate feedback can improve novice programming engagement and intention to persist in computer science. In: *Proceedings of the 2020 ACM Conference on International Computing Education Research* (pp. 194–203)

Matsuda, N., & VanLehn, K. (2005). Advanced geometry tutor: An intelligent tutor that teaches proof-writing with construction. In: *AIED*, vol. 125, (pp. 443–450)

Matsuda, N., & Vanlehn, K. (2004). Gramy: A geometry theorem prover capable of construction. *Journal of Automated Reasoning, 32*(1), 3–33.

Morrison, B. B., Margulieux, L. E., & Guzdial, M. (2015). Subgoals, context, and worked examples in learning computing problem solving. In: *Proceedings of the Eleventh Annual International Conference on International Computing Education Research* (pp. 21–29)

Newell, A. (1994). *Unified Theories of Cognition*. USA: Harvard University Press.

Newell, A., Simon, H. A., et al. (1972). *Human Problem Solving* (Vol. 104). NJ, USA: Prentice-hall Englewood Cliffs.

Newman, M. (2010). Networks: An Introduction Oxford Univ. *Press*

Sanz Ausin, M., Maniktala, M., Barnes, T., & Chi, M. (2020). Exploring the impact of simple explanations and agency on batch deep reinforcement learning induced pedagogical policies. In: *International Conference on Artificial Intelligence in Education* (pp. 472–485). Springer

Shabrina, P., Marwan, S., Chi, M., Price, T.W., & Barnes, T. (2020). The impact of data-driven positive programming feedback: When it helps, what happens when it goes wrong, and how students respond. In: *Proceedings of the 4th Educational Data Mining in Computer Science Education (CSEDM) Virtual Workshop*

Simon, H. A., & Newell, A. (1971). Human problem solving: The state of the theory in 1970. *American psychologist, 26*(2), 145.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive science, 12*(2), 257–285.

Sweller, J., Mawer, R. F., & Ward, M. R. (1983). Development of expertise in mathematical problem solving. *Journal of Experimental Psychology: General, 112*(4), 639.

Trafton, J., & Reiser, B. (1991). Providing natural representations to facilitate novices' understanding in a new domain: Forward and backward reasoning in programming. In: *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*(pp. 923–927). Lawrence Erlbaum Associates, Inc. Hillsdale, NJ

Van Gog, T., Paas, F., & Sweller, J. (2010). Cognitive load theory: Advances in research on worked examples, animations, and cognitive load measurement. *Educational Psychology Review, 22*(4), 375–378.

VanLehn, K. (1988). Toward a theory of impasse-driven learning. In: *Learning Issues for Intelligent Tutoring Systems* (pp. 19–41). Springer, ???

West, B. T., Welch, K. B., & Galecki, A. T. (2006). Linear Mixed Models: a Practical Guide Using Statistical Software. Chapman and Hall/CRC, ???

Zhi, R., Price, T.W., Lytle, N., Dong, Y., & Barnes, T. (2018). Reducing the state space of programming problems through data-driven feature detection. In: *Educational Data Mining in Computer Science Education (CSEDM) Workshop@ EDM*, vol. 18

Authors and Affiliations

Preya Shabrina[1] · Behrooz Mostafavi[1] · Mark Abdelshiheed[1] · Min Chi[1] · Tiffany Barnes[1]

Behrooz Mostafavi
bzmostaf@ncsu.edu

Mark Abdelshiheed
mnabdels@ncsu.edu

Min Chi
mchi@ncsu.edu

Tiffany Barnes
tmbarnes@ncsu.edu

[1] Computer Science, North Carolina State University, Raleigh 27695, North Carolina, USA