Exploring Differences in Problem Solving with Data-Driven Approach Maps

Michael Eagle
North Carolina State University
890 Oval Dr, Campus Box 8206
Raleigh, NC 27695-8206
Maiku.Eagle@gmail.com

Tiffany Barnes
North Carolina State University
890 Oval Dr, Campus Box 8206
Raleigh, NC 27695-8206
Tiffany.Barnes@gmail.com

ABSTRACT

Understanding the differences in problem solving behavior between groups of students is quite challenging. We have mined the structure of interaction traces to discover different approaches to solving logic problems. In a prior study, significant differences in performance and tutor retention were found between two groups of students, one group with access to hints, and one without. The Approach Maps we have derived help us discover differences in how students in each group explore the possible solution space for each problem. We summarize our findings across several logic problems, and present in-depth Approach analyses for two logic problems that seem to influence future performance in the tutor for each group. Our results show that the students in the hint group approach the two problems in statistically and practically different ways, when compared to the control group. Our data-driven approach maps offer a novel way to compare behaviors between groups, while providing insight into the ways students solve problems.

Keywords

Approach Maps, Logic Tutor, Data Driven Models

1. INTRODUCTION

Intelligent tutors have been shown to be as effective as human tutors in supporting learning in many domains, in part because of their individualized, immediate feedback, enabled by expert systems that diagnose student's knowledge states [20]. For example, students provided with intelligent feedback in the LISP tutor spent 30% less time and performed 43% better on post-tests when compared to other methods of teaching [1]. Similarly, Eagle, and Barnes showed that students with access to hints in the Deep Thought logic tutor spent 38% less time per problem and completed 19% more problems than the control group [5]. In another study on the same data, Stamper, Eagle, and Barnes showed that students without hints were 3.6 times more likely to drop out and discontinue using the tutor [19].

Procedural problem solving is an important skill in STEM (science, technology, engineering, and math) fields. Open-ended procedural problem solving, where steps are well-defined, but can be combined in many ways, can encourage higher-level learning [2]. However, understanding learning in open-ended problems, particularly when students choose whether or not to perform them, can be challenging. The Deep Thought tutor allows students to use logic rules in different ways and in different orders to solve 13 logic proof problems for homework. In this paper, we analyze the 2009 Deep Thought data set analyzed by Stamper, Eagle, and Barnes to further understand the differences between the hint and control groups.

The rich interaction data saved by transactional tutor logs offers many avenues to explore and understand student problem solving data, particularly for problems with multiple solutions. By mapping Deep Thought transactional data into an interaction network, and applying graph mining to derive regions based on the structure of this network, we develop a new Approach Map that illustrates the approaches that groups of students take in solving logic problems. We built Approach Maps for all 13 problems in the tutor, and illustrate a detailed analysis of two of these maps to explore the differences in problem solving between the hint and control groups.

The Approach Maps for problems 1.4 and 1.5 show that the hint group explored productive regions of the interaction network, while students in the control group were more likely to explore unproductive regions that did not lead to solutions. Problem 1.4 had available hints for the hint group. Even though problem 1.5 has no hints for either group, the Approach Map shows that the two groups still explore the problem space differently, illustrating that prior access to hints had a lasting effect. The Approach Maps help us discover unproductive regions of the problem-solving space, that we believe contributed to lower retention rates for the control group. In these regions, proactive hints could be used to direct students toward more productive approaches.

In section 2, we discuss related work and the prior study with Deep Thought. In section 4, we describe our algorithm for extracting Approach Maps from data. Section 5 presents the results and illustrates two detailed Approach Maps on problems 1.4 and 1.5. Finally, we discuss the results, conclusions, and future directions for this work.

2. RELATED WORK

Although they can be very effective, the construction of intelligent tutors can be costly, requiring content experts and pedagogical experts to work with tutor developers to identify the skills students are applying and the associated feedback to deliver [12]. One way to reduce the costs of building tutoring systems is to build data-driven approaches to generate feedback during tutor problem-solving. Barnes and Stamper built the Hint Factory to use student problem-solving data for automatic hint generation in a propositional logic tutor [17]. Fossati at el. implemented Hint Factory in the iList tutor to teach students about linked lists[7]. Evaluation of the automatically generated hints from Hint Factory showed an increase in student performance and retention [19]; more details about this study are provided in section 3.1.

Although individual differences affect the ways that students solve problems [11], it is difficult to examine the overall approaches that groups of students demonstrate during problem-solving. While pre and posttests are useful for measuring the change in behavior before and after an experimental treatment, we are interested in studying not only whether a student can solve a problem, but how they are solving the problem. In this study, we use interaction networks of student behaviors to investigate how providing hints affects student problem-solving approaches.

Interaction Networks describe sequences of student-tutor interactions [6]. Johnson et al. showed that visualizations of interaction networks in the InVis tool could be used to better understand how students were using the Deep Thought logic tutor [10]. Interaction networks form the basis of the data-driven domain model for automatic step-based hint generation by the Hint Factory. Eagle et al. applyied Girvan-Newman clustering to interaction networks to determine whether the resulting clusters might be useful for more high-level hint generation [6]. Stamper et al. demonstrated the differences in problem solving between the hint and control groups by coloring the edges between Girvan-Newman clusters of interaction networks based on the frequencies between two groups, revealing a qualitative difference in attempt paths [19]. In this paper we expand on these works to develop Approach Maps that concisely illustrate the approaches that students take while solving problems.

The Girvan-Newman algorithm (GN) was developed to cluster social network graphs using edge betweenness to find communities of people [8]. The technique also works in other domains. Wilkinson et al. applied GN in gene networks to find related genes [21]. Gleiser et al. used GN to discover essential ingredients of social interactions between jazz musicians [9]. We are the first to apply GN to interaction networks consisting of problem-solving steps.

In this paper, we mine the interactions from student problem solving data to summarize a large number of student-tutor transaction data into an Approach Map, demonstrating the diverse ways students solve a particular problem. We use Approach Maps to better understand the differences in behavior between two groups, students who were given access to hints, and those who were not, while completing homework in the *Deep Thought* logic proof tutor.

3. THE DEEP THOUGHT LOGIC TUTOR

In Deep Thought propositional logic tutor problems, students apply logic rules to prove a given conclusion using a given set of premises. Deep Thought allows students to work both forward and backwards to solve logic problems [3]. Working backwards allows a student to propose ways the conclusion could be reached. For example, given the conclusion B, the student could propose that B was derived using Modus Ponens (MP) on two new, unjustified (i.e. not yet proven) propositions: $A \to B, A$. This is like a conditional proof in that, if the student can justify $A \to B$ and A, then the proof is solved. At any time, the student can work backwards from any unjustified components (marked with a ?), or forwards from any derived statements or the premises. Figure 1 contains an example of working forwards and backwards with in Deep Thought.

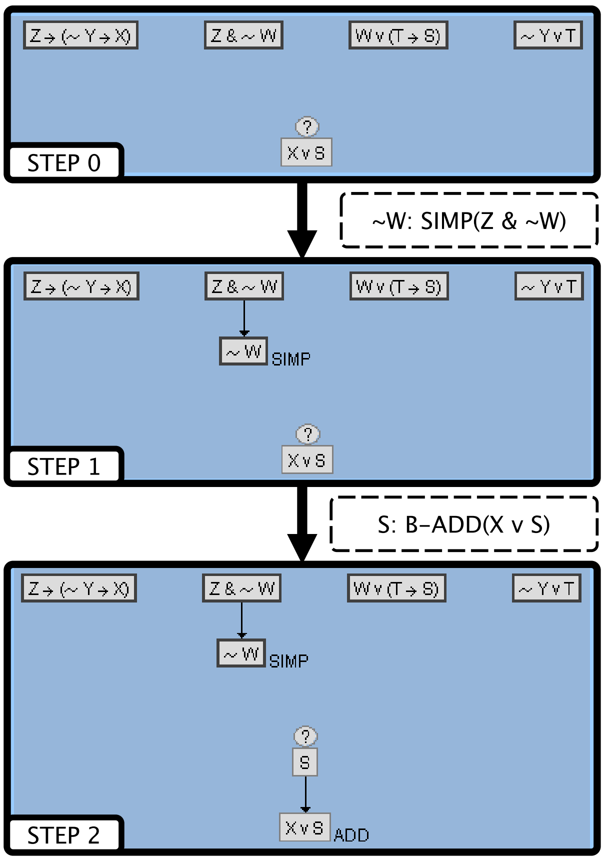

Figure 1: This example shows two steps within the Deep Thought tutor. First, the student has selected $Z \wedge \neg W$ and performed Simplification (SIMP) to derive $\neg W$. Second, the student selects $X \vee S$ and performs backward Addition to derive S.

3.1 Dataset and Prior Results

In 2012, Stamper, Eagle, and Barnes studied the effect of data-driven hints using the Spring and Fall 2009 Deep Thought propositional logic tutor dataset [19]. Data was collected from six 2009 deductive logic courses, taught by three professors. Each instructor taught one class using Deep Thought with automatically-generated hints on half of the problems (hint group, n=105) and one without access to hints on any problems (control, n=98). Students from the 6 sections were assigned 13 logic proofs in Deep Thought as a series of three graded homework assignments, with problems L1: 1.1-1.6, L2: 2.1-2.5, and L3: 3.1-3.2.

Table 1 shows retention information for each group after level L1; a χ^2 test of the relationship between group and dropout produced $\chi^2(1) = 11.05$, which was statistically significant at $p = 0.001$. The hint group completed more problems, with the effect sizes for these differences shown in Table 2. Stamper et al. found that the odds of a student in the control group dropping out of the tutor were 3.6 times more likely when compared to the group provided with automatically generated hints [19].

Table 1: Number of students that continued or dropped out of the tutor after L1

Group	Total	# Continued	# Dropped	% Dropped
Hint	105	95	10	9%
Control	98	71	27	28%
Total	203	166	37	18%

Table 2: The effect sizes of the differences between hint group and control group for completion and attempt rates by level.

	L1	L2	L3
Completed	$d = 0.51$*	$d = 0.64$*	$d = 0.39$*
Attempted	$d = 0.27$	$d = 0.44$*	$d = 0.33$*

Figure 2 charts the attempt and completion rates for hint group and control group for each problem in Deep Thought. Both groups had similar problem attempt rates, shown using solid lines, for L1 (1.1-1.5), but the hint group had significantly higher attempt rates in L2 and L3. The completion rates for each group are shown with dashed lines in Figure 2. Note that, after problem 1.4, the differences in attempt rates and completion rates seem to diverge between the groups.

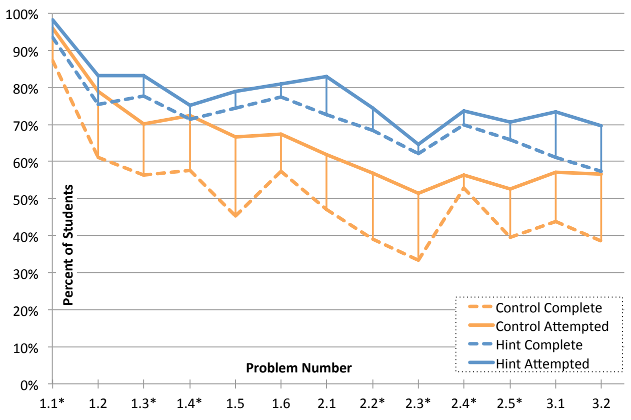

Figure 2: Attempt and complete rates per level, *indicates a problem where the hint group was given access to automatically generated hints.

We have investigated these results further. In another study, we modeled the time spent in the tutor using survival analysis [5]. In this study, we model the approaches students took to solve each problem.

4. METHODS

In Section 4.1, we describe how we use Deep Thought tutor logs to create an interaction network of all the student-tutor interactions within a single problem. We then show how we refine this network into *regions* of densely connected subgraphs (Section 4.2) using the Girvan-Newman (GN) algorithm. Finally, in Section 4.2.1 we define how we construct Approach Maps from the GN regions. For both steps in the process, we use the statistical environment *R* [14], and the complex network research library *iGraph* [4].

4.1 Constructing an Interaction Network

We construct an interaction network using all observed solution attempts to a single problem. Each solution attempt is a sequence of {state, action, resulting-state} interactions from the problem start to the last step a student performs. The *state* represents enough information to regenerate the tutor's interface at each step. An *action* is defined as a step taken, and consists of the name of the rule applied, the statements it was applied to, and the resulting derived statement. For example, Figure 1 displays two Deep Thought interactions. The first interaction works forward from STEP0 to STEP1 with action $SIMP$ (simplification) applied to $(Z \wedge \neg W)$ to derive $\neg W$. The second interaction works backward from STEP1 to STEP2 with action $B - ADD$ (backwards addition) applied to $(X \vee S)$ to derive the new, unjustified statement S.

We use a state matching function to combine identical states, that consist of all the same logic statements, but may have been derived in a different order. This way, the state for a step STEP0, STEP1, or STEP2 in Figure 1 is the set of justified and unjustified statements in each screenshot, regardless of the order that each statement was derived. We use an action matching function to combine actions, and preserve the frequency of each observed application.

If we treat the interactions used to create the networks as *samples* of observed behavior from a population, we could expect that the interaction networks constructed from different populations may have observable differences. However, rather than building two separate interaction networks and attempting to compare them, we construct a single network but keep track of the frequencies of visits by the hint and control groups for each state (vertex) and action (edge).

4.2 Extracting Regions

We partition the interaction network into densely connected subgraphs we call *regions* using Girvan and Newman's edge-betweenness clustering algorithm [8] and modularity score, a measure of the internal verses external connectedness of the regions [13]. We use following algorithm to apply region labels to nodes in a Deep Thought interaction network. First, we remove the problem start state and goal states from the Interaction Network IN to create G_1. Then, we iteratively remove all edges in G_1, in order of edge betweenness. Edge betweenness (EB) for a particular edge e is calculated by computing all shortest paths between all pairs of nodes, and counting the number of shortest paths that contain the edge e. At each GN iteration i and graph G_i, we find the edge with the highest EB, and call this bridge b_i. We remove the bridge b_i from the graph G_i, and compute the modularity score for the resulting graph G_{i+1}. The process is repeated until all edges have been removed. Then, we assign identifiers to all nodes in the disjoint regions in the intermediate graph G_n with the best modularity score. At the end of this process, we use G_n to construct the Approach Map with nodes for the original start and goal states, and a new node for each region in G_n. The Approach Map edges are the edges that connect the start state and goals to the regions, and the bridges between regions that were removed from the interaction network to create G_n.

Regions represent sets of steps that are highly connected to one another. When a solution attempt is within a region, new actions will stay within the region, or take a bridge edge into another region or goal. If an attempt is in a region with no goal bridges, the student must take a bridge to another region to reach a goal. Therefore, paths on the Approach Map can be interpreted as a high-level approaches to solving the problem. We hypothesize that we can use the Approach Map to discover different problem-solving approaches. In the next section, we investigate Approach Maps for two problems in Deep Thought, after which the hint and control groups diverged in performance.

4.2.1 *Approach Map*

Here we provide a more detailed description of the algorithm we use to generate an Approach Map from the interaction network for a problem after its nodes have been labeled with region identifiers. A region A (or action a) *dominates* a region B if every path from the start of the problem to B, must go through A (or a).

1. Combine all nodes with the same region identifier into a single region node labeled with the identifier, and remove all the edges with the same region identifier.
2. Combine all goal states that are dominated by a single region into a single goal node.
3. Calculate chi-squared to find in-edge frequencies that are different than expected between the groups (described in more detail below).
4. Combine parallel bridge edges between two regions into complex edges that represent the combination of the actions.
5. Label each region with the post conditions (derived statements) that result from the most frequent in-edge actions.
6. Provide new region identifiers that indicate the significant regions by the group with larger than expected frequency, with a number indicating the order in which the region was formed. For example, the regions the hint group visits more than expected are H1, H2, ..., the regions the control group visits more than expected are C1, C2, ..., and those that are visited as expected by both groups are labeled N1, N2, etc.

We use a two-tailed chi-squared test to look for differences between the hint and control groups in how they visit regions in the Approach Map. The null hypothesis is that there is no difference in the frequency of entering a particular region between attempts in the hint group and the control group. The alternative hypothesis is that the groups enter regions with different than expected frequency. We use Bonferroni correction [15] to compensate for the number of tests that we run. When the p is less than the Bonferroni-corrected alpha, we label the regions H1, H2, etc., blue for significantly higher than expected participation by the hint group. Regions C1, C2, etc., are bordered in orange and represent regions where the control group was represented more frequently than expected. Regions N1, N2, etc., satisfy the null hypothesis in that both groups visit these regions as expected.

The Approach Map for problem 1.4 is shown in Figure 3. Each region node contains statements derived on the most frequent in-edge. The bridge edges are those actions that most frequently lead into and out of each region. The edges are labeled with the action(s) taken and the number of attempts using these actions. A bridge and its resulting region can be read as, this many students performed the following action(s) to derive the following proposition(s). For clarity we do not draw edges with frequency less than ten, and we delete actions and regions that become disconnected due to these edge removals. The edges on the map are colored on a spectrum based on the ratio between the groups from blue (hint group) to orange (control group.) Paths in the Approach Map can be interpreted as empirically-observed problem solving approaches.

Each approach map is accompanied by a region table which provides more detail about the frequencies of observed solution attempts from each group. The columns Hint and Control are the total frequencies of in-edges by each group, or in other words, the number of solution attempts from each group that visit at least one node in the region. Time refers to the mean time a solution attempt stays in the region before exiting. Goals refers to the sum of the frequencies of out-edges that lead to goal states. The p values are the results of the chi-squared tests to compare group representation to expected values.

5. RESULTS & DISCUSSION

We perform our experiments on the Spring and Fall 2009 Deep Thought propositional logic tutor dataset as analyzed by Stamper, Eagle, and Barnes in 2012[19]. The data set is made up of 4301 student-attempts which contain 85454 student-tutor interactions across 13 problems. The prior study compared the performance between the hint (n=105) and control (n=98) groups, showing that students with available hints on the first 5 problems in L1 were 3.6 times more likely to complete the tutor. In addition, the hint group spent about 12 minutes per problem in the tutor, while the control group took 21 minutes per problem. Although the average total time in tutor between groups was not significantly different, more in-depth analysis of time revealed that this was because many students in the control group dropped out of the tutor, and were less likely to complete problems attempted in levels L2 and L3 [5]. In this section we present the results of applying Approach Maps to 11 problems in this data set, and illustrate the Approach Maps to two problems 1.4 and 1.5, just before the retention gap begins between the hint and control groups.

Table 3 summarizes our results from constructing Approach Maps for 11 of the 13 Deep Thought problems (records for problems 1.6 and 2.1 have not been normalized into our standard format). It is difficult to summarize the information from each map in to a single row in a table, however we have selected a few measures that provide an overview. In Table 3, the Hint and Control columns count the number of problem attempts for each group. Regions refers to the total number of regions in each Approach Map. Sig-H and Sig-C denote the number of regions visited significantly more than expected by the hint and control groups respectively. Sig-G denotes the number of significant regions that were also goal regions. This table shows that most problems in Deep Thought have 10-17 regions. In problems 1.4 and 1.5, more than half of the regions were visited more than expected by the hint or control groups.

Table 3: Summary of Approach Maps for 11 Deep Thought tutor problems. An asterisk (*) indicates problems where the hint group had access to hints.

Prob	Hint	Control	Regions	Sig-H	Sig-C	Sig-G
1.1*	348	447	16	1	7	2
1.2	196	187	16	1	2	1
1.3*	171	152	15	2	3	0
1.4*	138	219	16	5	4	2
1.5	155	218	18	4	6	2
2.2*	150	150	15	4	4	1
2.3*	129	108	14	4	3	1
2.4*	99	80	10	3	1	1
2.5*	112	79	10	3	0	1
3.1	173	114	17	0	1	0
3.2	147	100	12	1	2	0

We present detailed Approach Maps for problems 1.4 and 1.5 for three reasons. First, they occur before a large increase in control group dropout, as shown in Figure 2 in Section 3.1. After these problems, the odds of the control group dropping (no longer logging into the tutor) was 3.6 times that of the hint group [19]. Second, these problems stand out in Table 3, with high goal regions and more than half the extracted regions being significantly different between the groups. Third, in problem 1.4 the hint group had access to hints, however in problem 1.5 neither group received hints. This allows us to look for differences in behavior between the groups when working in the tutor on equal terms. For each of these problems we generated the Approach Map and corresponding reference table and visualization as described in Section 4.2.1.

5.1 Problem 1.4

Problem 1.4: Prove $X \vee S$
Given: $Z \rightarrow (\neg Y \rightarrow X), Z \wedge \neg W, W \vee (T \rightarrow S), \neg Y \vee T$

Problem 1.4 was designed to teach the Constructive Dilemma (CD) rule $[((P \rightarrow Q) \wedge (R \rightarrow S)) \wedge (P \vee R)] \rightarrow (Q \vee S)$. For this problem, students in the hint group had access to hints. Table 4 describes the regions of the Approach Map. Figure 3 shows the Approach Map for problem 1.4. To show differences in more detail, we have provided the most common attempts for each group in figure 4. In particular, this figure shows that the control group has derived an unjustified statement T that cannot be proven.

Hints were available for the hint group on problem 1.4; Table 5 shows the number of hint requests at depths D1 to D4, where students could request up to four consecutive hints while in a single state. In Table 5, R is the region, D1–4 is the depth of the hint, Target Proposition refers to the proposition the student is directed to derive, and Rule is the rule that the student is directed to use. Depth D1 hints direct students to the Hint column, while depth D2 hints direct the students to the Rule column. Depth D3 tells the student the preconditions needed to derive the target proposition. The depth D4 hint is a bottom out hint that directly tells the student what interface elements to click to derive the target step.

Table 4: Detailed information on the regions in the 1.4 Approach Map shown in Figure 3.

Region	Hint	Control	Time	Goals	p
H1	109	65	1.59	2	<0.001
H2	89	43	1.71	81	<0.001
H3	19	3	1.34	22	<0.001
C1	9	106	0.41	0	<0.001
C2	6	68	0.41	0	<0.001
C3	5	62	1.95	0	<0.001
C4	24	134	0.32	0	<0.001
N1	22	51	0.9	0	0.089
N2	9	15	1.41	20	0.811
N3	10	23	1.47	2	0.261
N4	14	38	0.13	0	0.056

Table 5: Number and depth of hints used by the hint group in each region; PS=Problem Start

R	D1	D2	D3	D4	Target Proposition	Rule
PS	50	13	13	4	$\neg W$	SIMP
H1	36	17	11	5	Z	SIMP
H1	32	16	11	5	$T \rightarrow S$	DS
H1	29	17	10	3	$\neg Y \rightarrow X$	MP
H2	36	19	18	2	$(\neg Y \rightarrow X) \wedge (T \rightarrow S)$	CONJ
H2	21	17	12	3	$X \vee S$	CD

There are three obvious paths in the Approach Map in Figure 3, one for the hint group, one for the control, and one with no differences between the groups. Figure 4 shows the most common solution paths for the hint and control group, with the same edges as the Approach Map. The Hint group tends to work forward using simplification (SIMP) (H1 to H2), while the control group was more likely to work backwards with addition (B-ADD) (C4 to C1). This backward addition path is a buggy strategy, that does not lead to any goals. We note that there are no backwards hints given in Deep Thought, so students on this path do not get hints regardless of group. Data on hint usage, shown in Table 5 and the statements derived in the H1-H3 regions suggest that students in the Hint group are being "routed" toward a successful strategy.

The Approach Map in Figure 3 shows that the control group is more likely to visit regions that do not contain successful goals. It seems that the effect of hints is to keep students

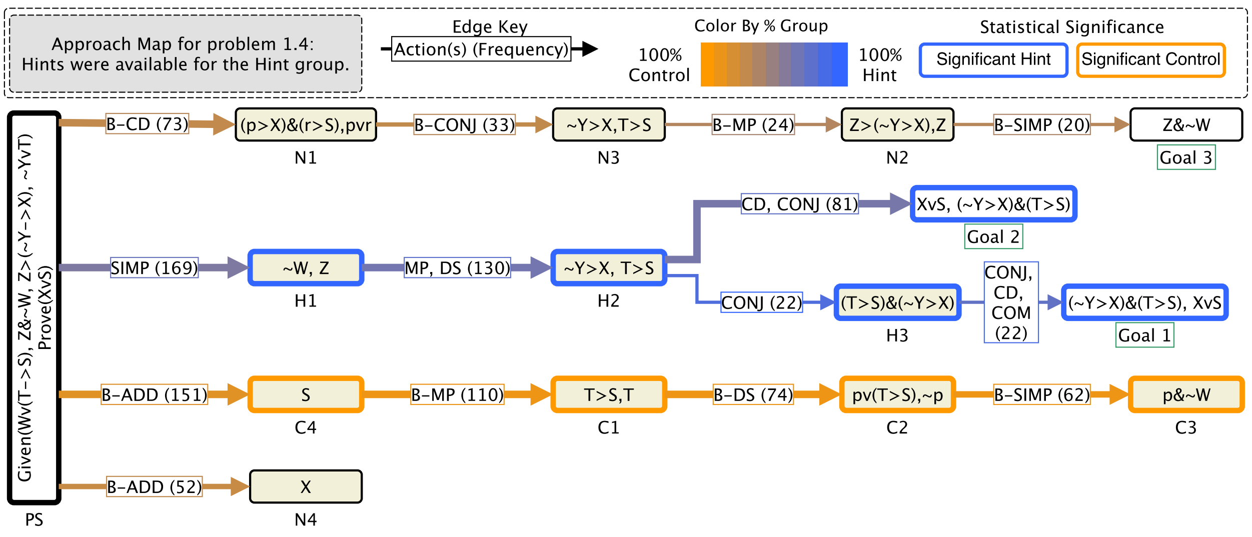

Figure 3: The Approach Map for problem 1.4. Edges and vertexes can be read as the number of students who performed action(s) to derive proposition(s). Three main approaches are revealed, with the hint group strongly preferring to work the problem forwards. The control group often attempts to solve the problem by wards with addition, there are no goals along this path. More detail is given in Table 4.

along a particular solution path, or prevent them from following the unproductive one taken by the control group. As a prior study of this data suggests [18], these students without hints are likely to abandon the tutor altogether. We hypothesize that hints help students achieve small successes and remain in the tutoring environment.

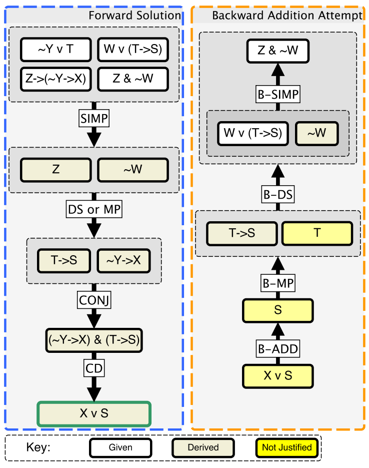

Figure 4: The most common attempt paths for each of the main approaches in the approach map for problem 1.4 (figure 3.) The highlighted nodes represent unjustified propositions.

5.2 Problem 1.5

Problem 1.5: Prove $A \vee \neg C$, given: $B \to (A \to E), B \vee (A \to \neg C), D \wedge \neg(A \to \neg C), E \to \neg C$.

Problem 1.5 was designed to teach the Hypothetical Syllogism (HS) axiom $[(P \to Q) \wedge (Q \to R)] \to (P \vee R)$. Problem 1.5 is interesting, as this problem had no hints, but still has large differences between the groups. The Approach Map is shown in Figure 5, and additional information on the regions is available in Table 6.

Table 6: Detailed information on the regions in the 1.5 Approach Map shown in Figure 5.

Region	Hint	Control	Time	Goals	p
H1	53	39	0.42	82	0.002
H2	89	58	1.16	26	<0.001
C1	17	55	0.69	0	0.002
C2	36	106	0.19	0	<0.001
C3	24	65	2.41	0	0.005
C4	16	51	0.72	0	0.003
C5	30	81	1.2	0	0.002
C6	3	19	0.22	0	0.007
N1	7	14	2.27	0	0.434
N2	7	8	1.16	11	0.700
N3	2	12	3.38	0	0.037
N4	8	15	0.26	0	0.498

The Hint group approaches problem 1.5 by working forward using simplification (SIMP) on $D \wedge B$ to derive the separate statements D and B; this could be a result of the forward directed hints they received in the earlier problems. The hints may have helped students develop a preference to working forwards, as doing so allowed them to request help if they became stuck. This preference carried over to the problems where hints were not available.

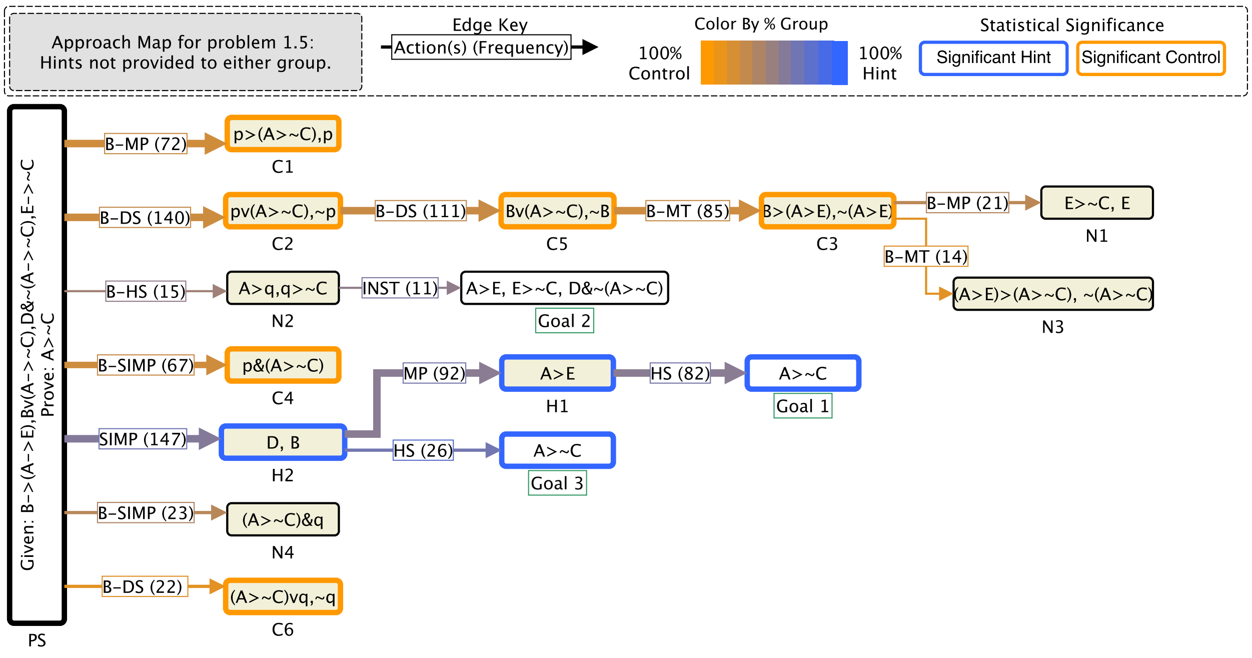

Figure 5: Even in the absence of the automatically generated hints, the hint group still prefers a forward solution. The control group explores regions that do not lead to goals. Details are given in Table 6.

When working problem 1.5, the control group systematically derives statements that do not lead to goals. The most common attempt is to work backwards with disjunctive syllogism (B-DS) (region C2) to derive $B \vee (A \rightarrow \neg C), \neg B$ from the conclusion $A \rightarrow \neg C$. This is likely because connecting with the premise $B \vee (A \rightarrow \neg C)$ seems like a promising direction. However, it is not possible to justify the proposed proposition $\neg B$ in this problem. This discovery is important as interventions can be added to warn away from regions that do not lead to goals. For example, we could offer a message that warns them that most students who attempt the same type of proof are not successful. Fossati et al. showed that human tutors helping students with the iList tutor, suggest that students delete unproductive steps [7].

5.3 Working Backwards and Trailblazing

Although working backwards seems be unproductive for the control group, we note that there are productive approaches that work backwards, for example N1-N4 regions in problem 1.4 explored evenly by both groups. There are some advantages to working backwards in *Deep Thought*. When a student works backwards, *Deep Thought* asks whether they would like to target the premises (extraction) or construct their own hypothesized statement from the conclusion. Then, the student clicks on one of just a few rules that can be used backwards, limiting the search space for the next step. Next, students are prompted to fill in the blanks in statements derivable from the chosen rule.

Region N1 in Figure 3, shows variables p and r that students can set to any proposition. Should the newly derived statements seem to match the patterns of existing premises, students keep them; otherwise they delete and try again. *Deep Thought* will sometimes warn students when they try to work backwards with something that is not justifiable. However, this may lead students to think that the tutor can always determine when working backwards is a viable strategy. In this case, students might mistakenly suppose that if there is no error message, they are closer to the solution. This is not the case, as *Deep Thought* has no built-in measures to determine closeness to completion. Rather, a few buggy rule applications are included in Deep Thought's automated error detection.

5.3.1 *Trailblazing Effect*

Barnes and Stamper proposed that hints might limit the breadth of student approaches to problems, causing a hint 'trailblazing' effect that might bias students toward expert solutions when originally building the Hint Factory[16]. In this analysis, we see some evidence of this effect. The difference in solution breadth between the two groups seems to be significant on several problems. The hints provided were limited to working forward, and the hint group demonstrated a strong preference for working forward. It remains to be seen whether providing hints for working backward will allow for more breadth of the search space. In any case, our results suggest that hints can cause a trailblazing effect, even when no hints are provided. Therefore, hints should be carefully constructed to include the diversity that a tutor designer wishes to promote in the tutor.

5.4 Conclusions and Future Work

In this paper, we have presented Approach Maps, a novel representation of student-tutor interaction data that allows for the comparison of problem-solving approaches on open-ended logic problems. The Approach Map visualization results in a significant reduction in the space needed to describe a large amount of student-tutor data. It does this by reducing the student attempts into regions that we can consider as higher-level approaches to problem-solving. Deep Thought problems each had an average of 330 solution attempts, which were made up of about 6.5 thousand interactions. Using our Approach Maps, we partition problems into about 15 regions each (including 2–3 goal regions, as shown in Table 3).

We have shown that we can use Approach Maps annotated with frequencies of visits by two groups to identify regions where a particular study group was over-represented. This allowed us to examine the approaches each group took to solving each proof. As we predicted, the automatically generated hints seemed to direct the students in the hint group down a common path, and we were able to detect this with the Approach Maps. Interestingly, even in problem 1.5, where neither group had hints, the hint group still showed a preference for working forwards, providing some evidence for a persistent effect of the hints. Analyzing Approach Maps also facilitated another important discovery that control group tended enter and remain in unproductive (or buggy) regions. These observed differences help explain how the automatically-generated hints produced the difference in tutor performance and retention in the 2009 Deep Thought study. Our investigations suggest that the patterns of behavior exhibited by students do result in meaningful regions of the solution attempt search space. We believe that, since the algorithms we applied to derive Approach Maps work on general graphs, we may be able to apply Approach Maps to understand problem-solving in domains where students solve open-ended problems in a procedural way.

In our future work, we plan to use Approach Maps to provide students with hints towards target sub-goals rather than simple step-based hints. We could also combine this with expert-created subgoals. We hypothesize that these more abstract hints will help encourage student planning. We also plan to use Approach Maps to provide proactive feedback to students when they enter unproductive regions. We will also apply Approach Maps to other open-ended problems to investigate their generalizability to other STEM fields.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0845997. The authors wish to thank John Stamper for providing the 2009 Deep Thought dataset for this analysis.

7. REFERENCES

[1] J. R. Anderson and B. J. Reiser. The lisp tutor. *Byte*, 10(4):159–175, 1985.

[2] B. S. Bloom. *Taxonomy of Educational Objectives: The Classification of Educational Goals.* Taxonomy of educational objectives: the classification of educational goals. Longman Group, New York, 1956.

[3] M. J. Croy. Problem solving, working backwards, and graphic proof representation. *Teaching Philosophy*, 23:169–188, 2000.

[4] G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[5] M. Eagle and T. Barnes. Survival analysis on duration data in intelligent tutors (under review). In *Intelligent Tutoring Systems*, Honolulu, Hawaii, 2014.

[6] M. Eagle, M. Johnson, and T. Barnes. Interaction Networks: Generating High Level Hints Based on Network Community Clustering. *educationaldatamining.org*, pages 1–4.

[7] D. Fossati, B. Di Eugenio, S. Ohlsson, C. Brown, L. Chen, and D. Cosejo. I learn from you, you learn from me: How to make ilist learn from students, 2009.

[8] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proc. National Academy of Sciences*, 99(12):7821–7826, June 2002.

[9] P. Gleiser and L. Danon. Community structure in jazz. *arXiv preprint cond-mat/0307434*, 2003.

[10] M. Johnson, M. Eagle, and T. Barnes. Invis: An interactive visualization tool for exploring interaction networks. In *Educational Data Mining (EDM2013)*, pages 82–89, 2013.

[11] D. Jonassen. Toward a design theory of problem solving. *Educational Technology Research and Development*, 48(4):63–85, 2000.

[12] T. Murray. Authoring intelligent tutoring systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education (IJAIED)*, 10:98–129, 1999.

[13] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[14] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2012. ISBN 3-900051-07-0.

[15] J. P. Shaffer. Multiple hypothesis testing. *Annual review of psychology*, 46(1):561–584, 1995.

[16] J. Stamper, T. Barnes, L. Lehmann, and M. Croy. The hint factory: Automatic generation of contextualized help for existing computer aided instruction. *Intelligent Tutoring Systems YRT*, pages 71–78, 2008.

[17] J. Stamper, T. Barnes, L. Lehmann, and M. Croy. A pilot study on logic proof tutoring using hints generated from historical student data. *Educational Data Mining (EDM 2008)*, pages 197–201, 2008.

[18] J. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental evaluation of automatic hint generation for a logic tutor. In *Artificial Intelligence in Education*, AIED'11, pages 345–352, Berlin, Heidelberg, 2011. Springer-Verlag.

[19] J. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental evaluation of automatic hint generation for a logic tutor. *International Journal of Artificial Intelligence in Education (IJAIED)*, 22(1):3–18, 2012.

[20] K. VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

[21] D. M. Wilkinson and B. A. Huberman. A method for finding communities of related genes. *Proc. National Academy of Sciences*, 101(Suppl 1):5241–5248, 2004.