Evaluating a Decision-Theoretic Approach to Tailored Example Selection

Kasia Muldner [1] and Cristina Conati [1,2]

[1] University of British Columbia
Department of Computer Science
Vancouver, B.C., Canada
{kmuldner, conati}@cs.ubc.ca

[2] University of Trento
Department of Information and
Communication Technology
Povo, Trento, Italy

Abstract

We present the formal evaluation of a framework that helps students learn from analogical problem solving, i.e., from problem-solving activities that involve worked-out examples. The framework incorporates an innovative example-selection mechanism, which tailors the choice of example to a given student so as to trigger studying behaviors that are known to foster learning. This involves a two-phase process based on 1) a probabilistic user model and 2) a decision-theoretic mechanism that selects the example with the highest overall utility for learning and problem-solving success. We describe this example-selection process and present empirical findings from its evaluation.

1 Introduction

Although examples play a key role in cognitive skill acquisition (e.g., [Atkinson *et al.,* 2002]), research demonstrates that students have varying degrees of proficiency for using examples effectively (e.g., [Chi *et al*., 1989; VanLehn, 1998; VanLehn, 1999]). Thus, there has been substantial interest in the Intelligent Tutoring Systems (ITS) community in exploring how to devise adaptive support to help all students benefit from example-based activities (e.g., [Conati and VanLehn, 2000; Weber, 1996]). In this paper, we describe the empirical evaluation of the Example Analogy (EA)-Coach, a computational framework that provides adaptive support for a specific type of example-based learning known as *analogical problem solving* (APS) (i.e., using examples to aid problem solving).

The EA-Coach's general approach for supporting APS consists of providing students with adaptively-selected examples that encourage studying behaviors (i.e. *meta-cognitive skills*) known to trigger learning, including:

1) *min-analogy*: solving the problem on one's own as much as possible instead of by copying from examples (e.g., [VanLehn, 1998]);
2) *Explanation-Based Learning of Correctness (EBLC*): a form of self-explanation (the process of explaining and clarifying instructional material to oneself [Chi *et al*., 1989]) that can be used for learning new domain principles by relying on, for instance, commonsense or overly-general knowledge, to explain how an example solution step is derived [VanLehn, 1999].

Min-analogy and EBLC are beneficial for learning because they allow students to (i) discover and fill their knowledge gaps and (ii) strengthen their knowledge through practise. Unfortunately, some students prefer more shallow processes which hinder learning, such as copying as much as possible from examples without any proactive reasoning on the underlying domain principles (e.g., [VanLehn, 1998; VanLehn, 1999].

To find examples that best trigger the effective APS behaviors for each student, the EA-Coach takes into account: i) student characteristics, including domain knowledge and pre-existing tendencies for min-analogy and EBCL, and ii) the similarity between a problem and candidate example. In particular, the Coach relies on the assumption that certain types of differences between a problem and example may actually be beneficial in helping students learn from APS, because they promote the necessary APS meta-cognitive skills. This is one of the novel aspects of our approach, and in this paper we present an empirical evaluation of the EA-Coach that validates it.

A key challenge in our approach is how to balance learning with problem-solving success. Examples that are not highly similar to the target problem may discourage shallow APS behaviors, such as pure copying. However, they may also hinder students from producing a problem solution, because they do not provide enough scaffolding for students who lack the necessary domain knowledge. Our solution to this challenge includes (i) incorporating relevant factors (student characteristics, problem/example similarity) into a probabilistic user model, which the framework uses to *predict* how a student will solve the problem and learn in the presence of a candidate example; (ii) using a decision-theoretic process to select the example that has the highest overall utility in terms of both learning and problem-solving success. The findings from our evaluation show that this selection mechanism successfully finds examples that reduce copying and trigger EBLC while still allowing for successful problem solving.

There are a number of ITS that, like the EA-Coach, select examples for APS, but they do not consider the impact of problem/example differences on a student's knowledge and/or meta-cognitive behaviors. ELM-PE [Weber, 1996] helps students with LISP programming by choosing examples that are as similar as possible to the target problem. AMBRE [Nogry *et al*., 2004] supports the solution of algebra problems by choosing structurally-appropriate examples. However, it is not clear from the paper what "structurally appropriate" means. Like the EA-Coach, several ITS rely on a decision-theoretic approach for action selection, but these do not take into account a student's meta-cognitive skills, nor they include the use of examples [Murray *et al.*, 2004; Mayo and Mitrovic, 2001]. Finally, some systems perform analogical reasoning in non-pedagogical contexts (e.g., [Veloso and Carbonell, 1993]), and so do not incorporate factors needed to support human learning.

In the remainder of the paper, we first describe the example-selection process. We then present the results from an evaluation of the framework and discuss how they support the EA-Coach's goal of balancing learning and problem-solving success.

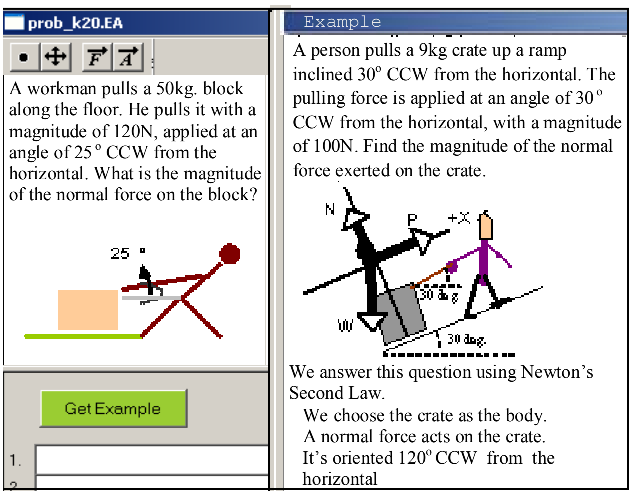

Figure 1: Fragment of the EA-Coach Interface

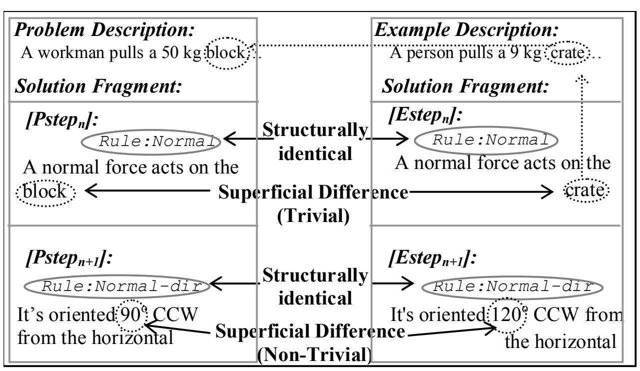

Figure 2: Sample Classification of Problem/Example Relations

2 The EA-Coach Example-Selection Process

The EA-Coach includes an interface that allows students to solve problems in the domain of Newtonian physics and ask for an example when needed (Fig. 1). For more details on the interface design see [Conati *et al.,* in press]. As stated in the introduction, the EA-Coach example-selection mechanism aims to choose an example that meets two goals: 1) helps a student solve the problem (*problem-solving success goal*) and 2) triggers learning by encouraging the effective APS behaviors of min-analogy and EBLC (*learning goal*). For each example stored in the EA-Coach knowledge base, this involves a two-phase process, supported by the EA-Coach user model: *simulation* and *utility calculation*. The general principles underlying this process were described in [Conati *et al.*, in press]. Here, we summarize the corresponding computational mechanisms and provide an illustrative example because the selection process is the target of the evaluation described in a later section.

2.1 Phase 1: The Simulation via the User Model

The simulation phase corresponds to generating a *prediction* of how a student will solve a problem given a candidate example, and what she will learn from doing so. To generate this prediction, the framework relies on our classification of various relations between the problem and candidate example, and their impact on APS behaviors. Since an understanding of this classification/impact is needed for subsequent discussion, we begin by describing it.

Two corresponding steps in a problem/example pair are defined to be *structurally identical* if they are generated by the same rule, and *structurally different* otherwise. For instance, Fig. 2 shows corresponding fragments of the solutions for the problem/example pair in Fig. 1, which include two structurally-identical pairs of steps: $Pstep_n/Estep_n$ derived from the rule stating that a normal force exists (*rule normal,* Fig. 2), and $Pstep_{n+1}/Estep_{n+1}$ derived from the rule stating the normal force direction (*rule normal-dir,* Fig. 2).

Two structurally-identical steps may be *superficially* different. We further classify these differences as *trivial* or *non-trivial.* While a formal definition of these terms is given in [Conati *et al.,* in press], for the present discussion, it suffices to say that what distinguishes them is the type of transfer from example to problem that they allow. Trivial superficial differences allow example steps to be copied, because these differences can be resolved by simply substituting the example-specific constants with ones needed for the problem solution. This is possible because the constant corresponding to the difference appears in the example/problem solutions and specifications, which provides a guide for its substitution [Anderson, 1993] (as is the case for $Pstep_n/Estep_n$, Fig. 2). In contrast, non-trivial differences require more in-depth reasoning such as EBLC to be resolved. This is because the example constant corresponding to the difference is *missing* from the problem/example specifications, making it less obvious what it should be replaced with (as is the case for $Pstep_{n+1}/Estep_{n+1}$, Fig. 2).

The classification of the various differences forms the basis of several key assumptions embedded into the simulation's operation. If two corresponding problem/example steps (*Pstep* and *Estep* respectively) are structurally different, the student cannot rely on the example to derive *Pstep*, i.e. the transfer of this step is blocked. This hinders problem solving if the student lacks the knowledge to generate *Pstep* [Novick, 1995]. In contrast, superficial differences between structurally-identical steps do not block transfer of the example solution, because the two steps are generated by the same rule. Although cognitive science does not provide clear answers regarding how superficial differences impact APS behaviors, we propose that the *type* of superficial difference has the following impact. Because trivial differences are easily resolved, they encourage copying for students with poor domain knowledge and APS meta-cognitive skills. In contrast, non-trivial differences encourage min-analogy and EBLC because they do not allow the problem solution to be generated by simple constant replacement from the example. We now illustrate how these assumptions are integrated into the EA-Coach simulation process.

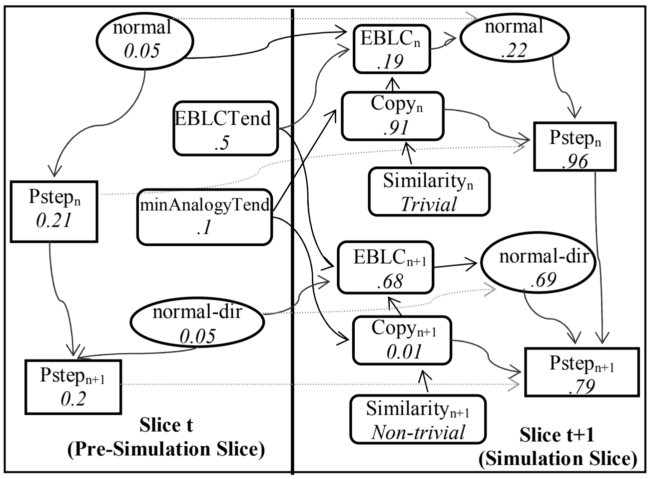

Figure 3: Fragment of the EA-Coach User Model

Simulation via the EA-Coach User Model

To simulate how the examples in the EA-Coach knowledge base will impact students' APS behaviors, the framework relies on its user model, which corresponds to a dynamic Bayesian network. This network is automatically created when a student opens a problem and includes as its backbone nodes and links representing how the various problem solution steps (Rectangular nodes in Fig. 3) can be derived from domain rules (Round nodes in Fig. 3) and other steps. For instance, the simplified fragment of the user model in Fig. 3, slice *t* (pre-simulation slice) shows how the solution steps $Pstep_n$ and $Pstep_{n+1}$ in Fig. 2 are derived from the corresponding rules *normal* and *normal-dir*. In addition, the network contains nodes to model the student's APS tendency for min-analogy and EBLC (*MinAnalogyTend and EBLCTend* in slice *t*, Fig. 3)[1].

To simulate the impact of a candidate example, a special '*simulation'* slice is added to the model (slice *t+1*, Fig. 3, assuming that the candidate example is the one in Fig. 2). This slice contains all the nodes in the pre-simulation slice, as well as additional nodes that are included for each problem-solving action being simulated and account for the candidate example's impact on APS. These include:

- *Similarity*, encoding the similarity between a problem solution step and the corresponding example step (if any).
- *Copy*, encoding the probability that the student will generate the problem step by copying the corresponding example solution step.
- *EBLC*, encoding the probability that the student will infer the corresponding rule from the example via EBLC.

During the simulation phase, the only form of direct evidence for the user model corresponds to the similarity between the problem and candidate example. This similarity is automatically assessed by the framework via the comparison of its internal representation of the problem and example solutions and their specifications. The similarity node's value for each problem step is set based on the definitions presented above, to either: *None* (structural difference), *Trivial* or *Non-trivial*. Similarity nodes are instrumental in allowing the framework to generate a fine-grained prediction of copying and EBLC reasoning, which in turns impacts its prediction of learning and problem-solving success, as we now illustrate.

Prediction of Copying episodes. For a given problem solution step, the corresponding copy node encodes the model's prediction of whether the student will generate this step by copying from the example. To generate this prediction, the model takes into account: 1) the student's min-analogy tendency and 2) whether the similarity between the problem and example allows the step to be generated by copying. The impact of these factors is shown in Fig. 3. The probability that the student will generate $Pstep_n$ by copying is high (see '$Copy_n$' node in slice *t+1*), because the problem/example similarity allows for it ('$Similarity_n$'=*Trivial,* slice *t+1*) and the student has a tendency to copy (indicated in slice *t* by the low probability of the '*MinAnalogyTend*' node). In contrast, the probability that the student will generate the step $Pstep_{n+1}$ by copying is very low (see node '$Copy_{n+1}$' in slice *t+1*) because the non-trivial difference ('$Similarity_{n+1}$'=*Non-trivial*, slice *t+1*) between the problem step and corresponding example step blocks copying.

Prediction of EBLC episodes. For a given problem rule, the corresponding EBLC node encodes the model's prediction that the student will infer the corresponding rule from the example via EBLC. To generate this prediction, the model takes into account 1) the student's EBLC tendency, 2) her knowledge of the rule (in that students who already know a rule do not need to learn it) 3) the probability that she will copy the step, and 4) the problem/example similarity. The last factor is taken into account by including an EBLC node only if the example solution contains the corresponding rule (i.e., the example is structurally identical with respect to this rule). The impact of the first 3 factors is shown in Fig. 3. The model predicts that the student is not likely to reason via EBLC to derive $Pstep_n$ (see node '$EBLC_n$,' in slice *t+1*) because of the high probability that she will copy the step (see node '$Copy_n$') and the moderate probability of her having tendency for EBLC (see node *EBLCTend* in slice *t).* In contrast, a low probability of copying (e.g., node '$Copy_{n+1}$', slice *t+1*) increases the probability for EBLC reasoning (see node '$EBLC_{n+1}$' in slice *t+1*), but the increase is mediated by the probability that the student has a tendency for EBLC, which in this case is moderate.

[1] Unless otherwise specified, all nodes have Boolean values

Prediction of Learning & Problem-Solving Success. The model's prediction of EBLC and copying behaviors influences its prediction of learning and problem-solving success. Learning is predicted to occur if the probability of a rule being known is low in the pre-simulation slice and the simulation predicts that the student will reason via EBLC to learn the rule (e.g., rule *normal-dir*, Fig. 3). The probabilities corresponding to the *Pstep* nodes in the simulation slice represent the model's prediction of whether the student will generate the corresponding problem solution steps. For a given step, this is predicted to occur if either 1) the student can generate the prerequisite steps and derive the given step from a domain rule (e.g. $Pstep_{n+1}$, Fig. 3) or 2) generate the step by copying from the example (e.g., $Pstep_n$, Fig. 3).

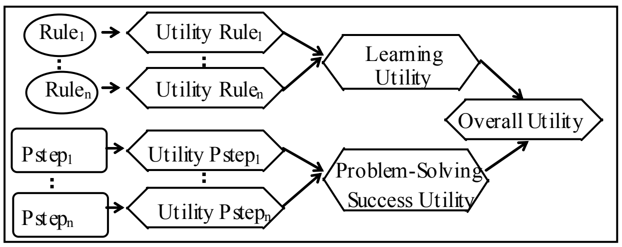

Figure 4: Fragment of the EA Utility Model

2.2 Phase 2: The Utility Calculation

The outcome of the simulation is used by the framework to assign a utility to a candidate example, quantifying its ability to meet the *learning* and *problem-solving success* objectives. To calculate this utility, the framework relies on a decision-theoretic approach that uses the probabilities of *rule* and *Pstep* nodes in the user model as inputs to the multi-attribute linearly-additive utility model shown in Fig. 4. The expected utility (EU) of an example for learning an individual rule in the problem solution corresponds to the sum of the probability *P* of each outcome (value) for the corresponding rule node multiplied by the utility *U* of that outcome:

$$EU(Rule_i) = P(known(Rule_i)) \cdot U(known(Rule_i)) + P(\neg known(Rule_i)) \cdot U(\neg known(Rule_i))$$

Since in our model, $U(known(Rule_i))=1$ and $U(\neg known(Rule_i))=0$, the expected utility of a rule corresponds to the probability that the rule is known. The overall learning utility of an example is the weighted sum of the expected learning utilities for all the rules in the user model:

$$\sum_i^n EU(Rule_i) \cdot w_i$$

Given that we consider all the rules to have equal importance, all weights *w* are assigned an equal value (i.e., *1/n*, where *n* is the number of rules in the user model). A similar approach is used to obtain the problem-solving success utility, which in conjunction with the learning utility quantifies a candidate example's overall utility.

The *simulation* and *utility calculation* phases are repeated for each example in the EA-Coach's knowledge base. The example with the highest overall utility is presented to the student.

3 Evaluation of the EA-Coach

As we pointed out earlier, one of the challenges for the EA-Coach example-selection mechanism is to choose examples that are different enough to trigger learning by encouraging effective APS behaviors (*learning goal*), but at the same time similar enough to help the student generate the problem solution (*problem-solving success goal*). To verify how well the two-phase process described in the previous section meets these goals, we ran a study that compared it with the standard approach taken by ITS that support APS, i.e., selecting the most similar example. Here, we provide an overview of the study methodology and present the key results.

3.1 Study Design

The study involved 16 university students. We used a within-subject design, where each participant 1) completed a pencil and paper physics pre-test, 2) was introduced to the EA-Coach interface (*training* phase), 3) solved two Newton's Second Law problems (e.g., of the type in Fig. 1) using the EA-Coach, (*experimental* phase) and 4) completed a pencil and paper physics post-test. We chose a within-subject design because it increases the experiment's power by accounting for the variability between subjects, arising from differences in, for instance, expertise, APS tendencies, verbosity (which impacts verbal expression of EBLC).

Prior to the experimental phase, each subject's pre-test data was used to initialize the priors for the rule nodes in the user model's Bayesian network. Since we did not have information regarding students' min-analogy and EBLC tendencies, the priors for these nodes were set to 0.5. During the experimental phase, for each problem, subjects had access to one example. For one of the problems, the example was selected by the EA-Coach (*adaptive-selection condition*), while for the other (*static-selection condition*), an example most similar to the target problem was provided. To account for carry-over effects, the orders of the problems/selection conditions were counterbalanced. For both conditions, subjects were given 60 minutes to solve the problem, and the EA-Coach provided immediate feedback for correctness on their problem-solving entries, realized by coloring the entries red or green. All actions in the interface were logged. To capture subjects' reasoning, we used the think-aloud method, by having subjects verbalize their thoughts [Chi *et al.*, 1989] and videotaped all sessions.

3.2 Data Analysis

The primary analysis used was univariate ANOVA, performed separately for the dependent variables of interest (discussed below). For the analysis, the within-subject *selection* factor (*adaptive* vs. *static*) was considered in combination with the two between-subject factors resulting from the counterbalancing of selection and problem types. The results from the ANOVA analysis are based on the data from the 14 subjects who used an example in *both* conditions (2 subjects used an example in only one condition: one subject used the example only in the static condition, another subject used the example only in the adaptive condition).

3.3 Results: Learning Goal

To assess how well the EA-Coach adaptively-selected examples satisfied the learning goal as compared to the statically-selected ones, we followed the approach advocated in [Chi *et al*., 1989]. This approach involves analyzing students' *behaviors* that are known to impact learning, i.e., copying and self-explanation via EBLC in our case. Although this approach makes the analysis challenging because it requires that students' reasoning is captured and analyzed, it has the advantage of providing in-depth insight into the EA-Coach selection mechanism's impact. For this part of the analysis, univariate ANOVAs were performed separately for the dependent variables *copy* and *EBLC* rates.

Copy Rate. To identify copy events, we looked for instances when students: 1) accessed a step in the example solution (as identified by the verbal protocols and/or via the analysis of mouse movements over the example) and 2) generated the corresponding step in their problem solution with no changes or minor changes (e.g., order of equation terms, constant substitutions). Students copied significantly less from the adaptively-selected examples, as compared to the statically-selected examples ($F(1,14)=7.2$, $p=0.023$; on average, 5.9 vs. 8.1 respectively).

EBLC rate. To identify EBLC episodes, we analyzed the verbal protocol data. Since EBLC is a form of self-explanation, to get an indication of how selection impacted explanation rate in general, we relied on the definition in [Chi *et al*., 1989] to first identify instances of self-explanation. Students expressed significantly more self-explanations while generating the problem solution in the adaptive selection condition, as compared to in the static condition ($F(1, 10)=6.4$, $p=0.03$; on average, 4.07 vs. 2.57 respectively). We then identified those self-explanations that were based on EBLC (i.e., involved learning a rule via commonsense and/or overly-general reasoning, as opposed to explaining a solution step using existing domain knowledge). Students generated significantly more EBLC explanations in the adaptive than the static condition ($F(1, 10)=12.8$, $p=0.005$; on average, 2.92 vs. 1.14 respectively).

Pre/Post Test Differences. With the analysis presented above, we evaluated how the EA-Coach selection mechanism impacts learning by analyzing how effectively it triggers APS behaviors that foster it. Another way to measure learning is via pre/post test differences. In general, students improved significantly from pre to post test (on average, from 21.7 to 29.4; 2-tailed $t(15)=6.13$, $p<0.001$). However, because there was overlap between the two problems in terms of domain principles, the within-subject design makes it difficult to attribute learning to a particular selection condition. One way this could be accomplished is to 1) isolate rules that only appeared in one selection condition and that a given student did not know (as assessed from pre-test); 2) determine how many of these rules the student showed gains on from pre to post test. Unfortunately, this left us with very sparse data making formal statistical analysis infeasible. However, we found encouraging trends: there was a higher percentage of rules learned given each student's learning opportunities in the adaptive condition, as compared to the static one (on average, 77% vs. 52% respectively).

Discussion. As far as the learning goal is concerned, the evaluation showed that the EA-Coach's adaptively-selected examples encouraged students to engage in the effective APS behaviors (min-analogy, EBLC) better than statically-selected examples: students copied less and self-explained more when given adaptively-selected examples. This supports our assumption that certain superficial differences encourage effective APS behaviors. The statically-selected examples were highly similar to the target problem and thus made it possible to correctly copy much of their solutions, which students took advantage of. Conversely, by blocking the option to correctly copy most of their solution, the adaptively-selected examples provided an incentive for students to infer via EBLC the principles needed to generate the problem solution.

3.4 Results: Problem-Solving Success Goal

The problem-solving success goal is fulfilled if students generate the problem solution. To evaluate if the adaptive example-selection process met this goal, we checked how successful students were in terms of generating a solution to each problem. In the static condition, all 16 students generated a correct problem solution, while in the adaptive condition, 14 students did so (the other 2 students generated a partial solution; both used the example in both conditions). This difference between conditions, however, is not statistically significant (sign test, $p=0.5$), indicating that overall, both statically and adaptively selected examples helped students generate the problem solution.

We also performed univariate ANOVAs on the dependent variables *error rate* and *task time* to analyze how the adaptively-selected examples affected the problem solving *process,* in addition to the problem solving *result*. Students took significantly longer to generate the problem solution in the adaptive than in the static selection condition ($F(1, 10) =31.6$, $p<0.001$; on average, 42min., 23sec. vs. 25min., 35sec. respectively). Similarly, students made significantly more errors *while* generating the problem solution in the adaptive than in the static selection condition ($F(1, 10)=11.5$, $p=0.007$; on average, 22.35 vs. 7.57 respectively).

Discussion. As stated above, the problem-solving success goal is satisfied if the student generates the problem solution, and is not a function of performance (time, error rates) *while* doing so. The fact that students took longer/made more errors in the adaptive condition is not a negative finding from a pedagogical standpoint, because these are *by-products* of learning. Specifically, learning takes time and may require multiple attempts before the relevant pieces of knowledge are inferred/correctly applied, as we saw in our study and as is backed up by cognitive science findings (e.g., [Chi, 2000]).

However, as we pointed out above, 2 students generated a correct but incomplete solution in the adaptive selection condition. To understand why this happened, we analyzed these students' interaction with the system in detail. Both of them received an example with non-trivial superficial differences that blocked copying of some solution steps, because the user model predicted this would trigger learning via EBLC. This prediction is mediated by the model's assessment of the student's EBLC tendency, to which we had assigned a generic prior probability of 0.5 for both students due to lack of more accurate information. This appeared to have been inaccurate for one of these students, who showed no desire to engage in any in-depth reasoning during the study (i.e., likely had a very low EBLC tendency). The other student, however, generated a number of EBLC self-explanations, indicating that inaccurate prior on EBLC tendency was not the reason for suboptimal example selection in terms of problem-solving success. This student invested considerable effort and did learn *some* of the rules needed to solve the problem (as we found by comparing her pre and post-test answers on related questions). However, although the simulation predicted she would learn all the necessary rules and thus generate the full problem solution, she was unable to do so within the allotted 60 minutes, mostly because she sometimes required several attempts to infer a correct rule. We can't predict whether this student would have eventually generated a full solution or whether she would have become overwhelmed and frustrated by the process. There is a fine line between taking extra time to learn from one's errors, and *floundering*, i.e. engaging in too many trial and error attempts that obstruct learning. Thus, even if students have good APS tendencies there is no guarantee that they will learn all the rules needed to generate a full problem solution. This suggests that the system could be improved by the addition of more explicit scaffolding of correct EBLC to help students when they are floundering.

4 Conclusions & Future Work

We have presented the evaluation of a framework that provides support for APS through its example-selection mechanism. This mechanism relies on a decision-theoretic approach to find examples that encourage effective APS behaviors while helping the student to generate the problem solution. To do so, the mechanism relies on the assumption that examples including certain types of superficial differences with the target problem discourage copying and thus encourage students to learn the underlying domain principles via EBLC. This is in contrast to existing approaches for example selection, which present the student with the most similar example. The findings from our evaluation support our approach, by showing that choosing examples with appropriate differences triggers the effective APS behaviors. However, we also found that for some students, just using examples to trigger these behaviors may have detrimental side-effects, such as excessively increasing problem-solving time. Thus, as our next step, we plan to explore if and how more explicit forms of scaffolding on the target APS behaviors may address this problem, as well as in general help students who have very poor APS tendencies. We also plan to integrate the EA-Coach with two other ITS that target different phases of the problem-solving spectrum, i.e., studying examples before problem solving, and pure problem solving without the aid of examples.

References

[Anderson, 1993] J. R. Anderson. *Rules of the Mind*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1993.

[Atkinson *et al.*, 2002] R. Atkinson, A. Renkl and D. Wortham. Learning from Examples: Instructional Principles from the Worked Examples Research. *Review of Educational Research*, 70(2):181-214, 2002.

[Veloso and Carbonell, 1993] M. Veloso and J. Carbonell. Derivational Analogy in PRODIGY: Automating Case Acquisition, Storage, and Utilization. *Machine Learning*, 10(3):249-278, 1993.

[Chi, 2000] M. T. H. Chi. Self-Explaining: The dual process of generating inferences and repairing mental models. *Advances in Instructional Psychology*. Lawrence Erlbaum Associates, Hillsdale, NJ, pages 161-238, 2000.

[Chi *et al.*, 1989] M. T. H. Chi, M. Bassok, M. W. Lewis, P. Reimann and R. Glaser. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, 15:145-182, 1989.

[Conati *et al.*, in press] C. Conati, K. Muldner, and G. Carenini. From Example Studying to Problem Solving via Tailored Computer-Based Meta-Cognitive Scaffolding: Hypotheses and Design. *Journal of Technology, Instruction, Cognition & Learning*, in press.

[Conati and VanLehn, 2000] C. Conati, and K. VanLehn. Toward Computer-based Support of Meta-cognitive Skills: A Computational Framework to Coach Self-Explanation. *Int. Journal of Artificial Intelligence in Education*, 11:389-415, 2000.

[Mayo and Mitrovic, 2001] M. Mayo and A. Mitrovic. Optimizing ITS behavior with Bayesian networks and decision theory. *Int. Journal of Artificial Intelligence in Education*, 12:124-153, 2001.

[Murray *et al*, 2004] C. Murray, J. Mostow and K. VanLehn. Looking ahead to select Tutorial Actions: a Decision-Theoretic approach. *Int. Journal of Artificial Intelligence in Education*, 14(3-4):235-278, 2004.

[Nogry *et al.*, 2004] S. Nogry, S. Jean-Daubias and N. Duclosson. ITS Evaluation in Classroom: The Case of Ambre-AWP, In *Proc. of Intelligent Tutoring Systems*, pages 511-520, Maceio, Brazil, 2004. Springer.

[Novick, 1995] L. Novick. Some Determinants of Successful Analogical Transfer in the Solution of Algebra Word Problems. *Thinking and Reasoning*, 1(1):5-30, 1995.

[VanLehn, 1998] K. VanLehn. Analogy Events: How Examples are Used During Problem Solving. *Cognitive Science*, 22(3):347-388, 1998.

[VanLehn, 1999] K. VanLehn. Rule-Learning Events in the Acquisition of a Complex Skill: An Evaluation of Cascade. *Journal of the Learning Sciences*, 8:71-125, 1999.

[Weber, 1996] G. Weber. Individual Selection of Examples in an Intelligent Learning Environment. *Int. Journal of Artificial Intelligence in Education*, 7(1):3-33, 1996.