Tech., Inst., Cognition and Learning, Vol. 4, pp. 00-00
Reprints available directly from the publisher
Photocopying permitted by license only

From Example Studying to Problem Solving via Tailored Computer-Based Meta-Cognitive Scaffolding: Hypotheses and Design

CRISTINA CONATI*, KASIA MULDNER AND GIUSEPPE CARENINI

Department of Computer Science, University of British Columbia
201-2366 Main Mall, Vancouver, V6T 1Z4
Canada

We present an intelligent tutoring framework designed to help students acquire problem-solving skills from pedagogical activities involving worked-out example solutions. Because of individual differences in their meta-cognitive skills, there is great variance in how students learn from examples. Our framework takes into account these individual differences and provides tailored support for the application of two key meta-cognitive skills: self-explanation (i.e., generating explanations to oneself to clarify studied material) and min-analogy (i.e., not relying too heavily on examples during problem solving). We describe the framework's two components. One component explicitly scaffolds self-explanation during example studying with menu-based tools and direct tailored tutorial interventions, including the automatic generation of example solutions at varying degrees of detail. The other component supports both self-explanation and min-analogy during analogical problem solving by relying on subtler scaffolding, including a highly innovative example selection mechanism. We conclude by reporting results from an empirical evaluation of the former component, showing that it facilitates cognitive skill acquisition when students access it at the appropriate learning stage.

Keywords: Intelligent tutoring, example studying, Analogical problem solving, Meta-cognitive skills, Self-explanation, Min-analogy, Example selection, Natural language generation, Adaptive instruction.

*Corresponding author: conati@cs.ubc.ca

INTRODUCTION

Research in cognitive science has provided extensive evidence for a natural progression in cognitive skill acquisition that centres on worked-out example solutions as learning aids. This progression starts with students studying examples *prior* to problem solving (e.g., Anderson, Fincham et al., 1997; VanLehn, 1996). As expertise increases through example studying, students progress to problem solving, but can still find it useful to refer to examples for guidance, thus engaging in what is commonly known as *analogical problem solving* (e.g., Novick & Holyoak, 1991; Reed, Dempster et al., 1985; VanLehn, 1998; VanLehn, 1999). However, research in cognitive science also indicates that how beneficial a role examples play in this progression strongly depends on the student's ability to apply meta-cognitive skills that foster learning during example-based pedagogical activities.

Two such meta-cognitive skills that have been extensively studied include *self-explanation* and *min-analogy*. Self-explanation involves elaborating and clarifying to oneself available instructional material (e.g., Chi, Bassok et al., 1989; Pirolli & Recker, 1994; Renkl, 1997; Renkl, Stark et al., 1998). Min-analogy involves not relying too heavily on examples during problem solving, that is transferring from the example only the minimum amount of information necessary to enable successful problem solving (VanLehn, 1998; VanLehn & Jones, 1993). Unfortunately, research shows that not all students can apply these meta-cognitive skills effectively, although there is some evidence that they can be explicitly coached to do so (e.g., Bielaczyc, Pirolli et al., 1995).

In this paper, we present an Intelligent Tutoring framework that can provide this coaching in an individualized manner, thus supporting an optimal example-based natural progression in cognitive skill acquisition. The framework complements Andes, a well established Intelligent Tutoring System (ITS) designed to support physics problem solving at the college level (Conati, Gertner et al., 2002; Schulze, Shelby et al., 2000), with two coaching components that support the example-based phases of skill acquisition that precede pure problem solving. The first component, known as the SE (Self-Explanation)-Coach, supports example studying prior to problem solving. The second component, known as the EA (Example-Analogy)-Coach, supports analogical problem solving (APS from now on). An earlier version of the SE Coach has been described in Conati and VanLehn (2000). In this paper we describe both an extended version of the SE-Coach and the EA-Coach in the context of an integrated framework to support learning from examples. Although there will be some overlap with Conati and VanLehn (2000),

we deem this as necessary to illustrate the scope and overall approach of this work, and to make this paper a self-contained description of our complete framework for example-based learning.

The choice of devising this framework to complement the support for physics problem solving provided by Andes is based on three main reasons. First, physics is one of the domains in which there is a well-recognized need in cognitive science for innovative educational tools, due to the extreme difficulties that students often encounter in bridging theory and practice, either because they cannot make the leap from theory acquisition to effective problem solving, or because they may learn to solve problems effectively without grasping the underlying theory (Halloun & Hestenes, 1985). Second, there were several components of the Andes architecture that we could reuse in the framework and that had been previously validated through the various evaluations of Andes (e.g., the problem-solving interface that the EA-Coach uses to support APS). Third, integrating our framework with Andes facilitates the process of devising an Intelligent Learning Environment that covers the complete progression from aided example studying, to problem solving supported by tailored examples, to pure problem solving with more traditional tailored hints and feedback.

While there has been extensive research in Intelligent Tutoring Systems (ITS) on how to use examples as learning aids, our framework for example-based learning (ExBL framework from now on) has three distinguishing features. The first is that ExBL is the only ITS that addresses both the example-studying and the analogical problem-solving phases of cognitive skill acquisition, and that does so by providing individualized coaching for the relevant meta-cognitive skills. The second is that ExBL includes one of the first attempts to apply natural language techniques for the automatic generation of example solutions that stimulate example-based learning. The third is that ExBL extends research on providing computer-based support for meta-cognitive skill acquisition both by making the first attempt to address the min-analogy meta-cognitive skill, specifically relevant to improving performance during APS and by exploring the impact of differences between a problem and an example to provide tailored stimuli for effective APS.

Because of the complexity of the ExBL framework and of its pedagogical components, we have adopted the strategy of evaluating the individual components before proceeding to a comprehensive ecological study of the complete system. In this paper, we report on a selection of the results from an empirical evaluation of the SE-Coach (Conati & VanLehn,

1999), showing that this component facilitates cognitive skill acquisition when students access it at the appropriate learning stage. Although the EA-Coach is yet to be evaluated, we will discuss how the results of the SE-Coach evaluation support some of the design choices that shape the EA-Coach pedagogical interaction.

In the rest of the paper, we first give a general overview of the ExBL framework and architecture. We then provide details on each individual Coach, including a discussion of the relevant meta-cognitive skills and of the tools each Coach uses to foster them. Next, we describe the SE-Coach evaluation. We then discuss related research, and conclude with future work.

EXBL FRAMEWORK: OVERVIEW AND ARCHITECTURE

The philosophy underlying the design of the ExBL framework is to support meta-cognitive skills for example-based learning by stimulating students to take the initiative in the learning process, rather than by enforcing a strict tutorial interaction in which students passively follow a tutor's directives. This is achieved by providing multiple levels of scaffolding, individualized to best accommodate the varied student propensity and capability for the relevant meta-cognitive skills (Chi, Bassok et al., 1989; Renkl, 1997; VanLehn, 1998; VanLehn, 1999). The scaffolding includes:

- simple reminders to engage the student in the relevant meta-cognitive processes,
- presentation of study material tailored to facilitate adequate meta-cognition,
- feedback on student performance, both at the problem solving and at the meta-cognitive level,
- interface tools designed to explicitly guide effective learning behaviours,
- directed tutorial interventions for those students who need stronger guidance.

The goal is to support students' initiative by giving them the opportunity to choose how much scaffolding they want. More proactive help, tailored to each student's specific needs, is provided only when students show suboptimal learning behaviours.

In order to tailor its scaffolding to a student's needs, the ExBL framework must be capable of monitoring and assessing each student's performance with respect to the target pedagogical tasks. Thus, the framework needs an internal representation of these tasks, against which to compare the student's problem-solving and example-studying behaviours. It also needs to encode its assessment in terms of the student's domain knowledge and relevant meta-cognitive skills in a computational student model that can then be used to guide the tailoring of instructional support.

The above requirements are implemented in the architecture shown in Figure 1. The architecture's user interface components provide interactive tools for students to study examples (SE-Coach) and to use examples during problem solving (EA-Coach). All student interface actions are monitored and assessed against the system's internal representation of the relevant problem/example solutions. This internal representation, known as the *solution graph*, is automatically built before run-time by the component labelled as *Problem Solver* in Figure 1 (left) starting from: (i) a knowledge base of physics and planning rules (*Domain and planning rules* in Figure 1) and (ii) a formal description of the initial situation for the examples/problems involved in each task (*Problem definition* in Figure 1) (Conati & VanLehn, 2000). Each solution graph is a dependency network that represents how each solution step derives from previous steps and physics knowledge.

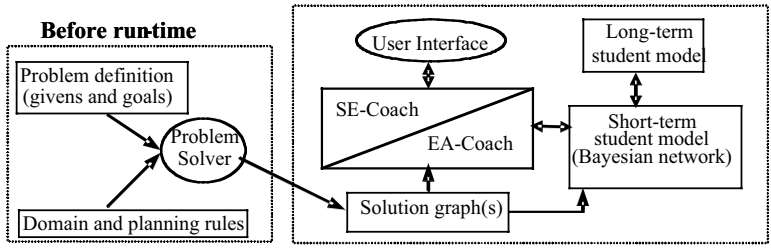


FIGURE 1
ExBL Framework architecture.

As an example of a solution graph consider, for instance, the physics example in Figure 2. Figure 3 shows the part of the solution graph that derives the first two steps mentioned in the example solution: establish the goal to apply Newton's 2nd Law and select the body to which to apply the law. In

the solution graph, intermediate solution facts and goals (*F*- and *G*- nodes in Figure 3) are connected to the rules (*R*- nodes) used to derive them and to previous facts and goals matching the rules' enabling conditions. The connection goes through rule-application nodes *(RA*- nodes in Figure 3) which explicitly represent the application of each rule in the context of a specific example. Thus, the segment of the network in Figure 3 encodes that the rule *R-try-Newton-2law* establishes the goal to apply Newton's 2nd Law (node *G-try-Newton-2law*) to solve the goal to find the force on the wagon (node *G-force-on wagon*).

The rule *R-goal-choose-body* then sets the sub-goal to find a body to apply Newton's 2nd Law (node *G-goal-choose-body*) and the rule *R-body-by-force* dictates that if one has the goals to find the force on an object and to select a body to apply Newton's 2nd Law, that object should be selected as the body. Thus, in Figure 3 this rule selects the wagon as the body for the example in Figure 2 (node *F-wagon-is the body*).

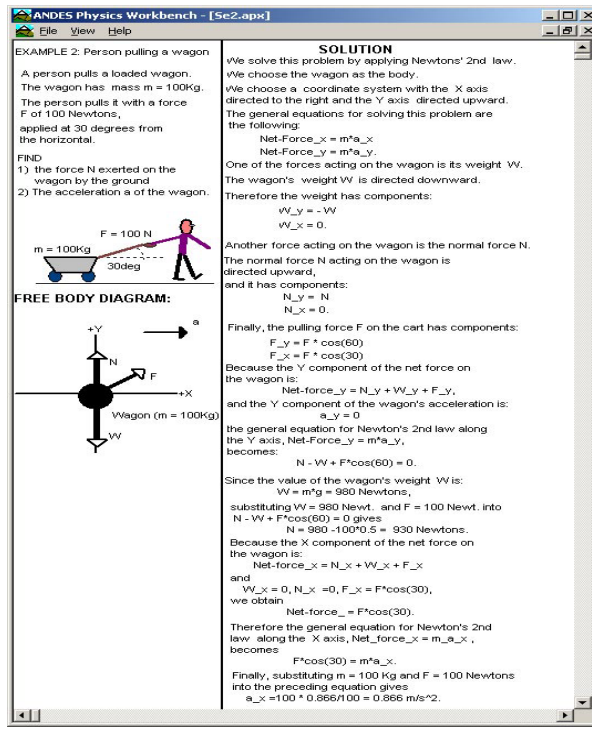


FIGURE 2
Sample SE-Coach example.

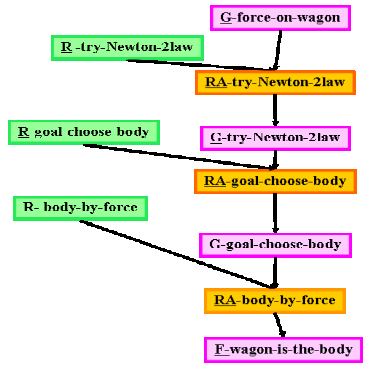


FIGURE 3
Segment of solution graph for the example in Figure 2.

Both SE-Coach and EA-Coach use the solution graph to provide feedback on students' performance during example studying and problem solving, by matching students' interface actions to elements in the solution graph. In addition to serving as the basis for the ExBL framework's ability to provide feedback, the solution graph is also used to build the framework's student models. Each time a student opens a new exercise, the corresponding solution graph provides the core structure for a Bayesian network (Pearl, 1988) that forms the short-term student model for the currently active Coach (see right side of Figure 1). The Bayesian network uses information on the student's interface actions to generate a probabilistic assessment of the student's knowledge and relevant meta-cognitive tendencies at any given point during the interaction[1]. This allows the system to generate tailored interventions to foster effective meta-cognitive skills when the model assesses that the student has knowledge gaps or that her meta-cognitive behaviours need improvement. The prior probabilities to initialise the rule nodes in the Bayesian network come from the long-term student model (see Figure 1). This model contains a probabilistic assessment of a student's knowledge of each rule in the Knowledge Base at the time when a new exercise is started. This assessment is updated every time the student finishes an exercise, with the new rule probabilities computed by the short-term model Bayesian network for that exercise.

What we have described so far are the elements that are shared by both Coaches in the ExBL framework. We will now discuss each Coach in more detail. For each Coach, we will first introduce the meta-cognitive skills

[1]In the SE-Coach, assessment of student self-explanation behavior is done implicitly through the nodes shown in Figure 3 (Conati & Vanlehn, 2001), while the EA-Coach Bayesian network includes additional nodes that explicitly assess the relevant meta-cognitive skills (Muldner & Conati, 2005).

required to learn effectively in the targeted learning context, followed by a description of the Coach itself.

THE SE-COACH: PROVING TAILORED SUPPORT FOR EXAMPLE STUDYING

Prior to commencing problem solving, students often find it useful to study some worked-out examples. There is substantial evidence that students learn more effectively from this activity if they engage in the meta-cognitive skill known as self-explanation, i.e., generating elaborations and explanations to themselves to clarify the example solution (e.g., Chi, et al., 1989; Pirolli & Recker, 1994; Renkl, 1997). Self-explanation is believed to be beneficial both because it triggers more constructive learning processes than merely reading a provided explanation and because it allows a student to tailor the explanation to her particular needs, i.e., specific knowledge gaps or misconceptions. As a result, students who self-explain end up with more solid knowledge that they can bring to bear during problem solving. However, studies show that many students tend not to self-explain spontaneously, thus gaining limited benefits from example studying (e.g., Chi, Bassok et al., 1989; Pirolli & Recker, 1994; Renkl, 1997). The SE-Coach is explicitly designed to provide support for these students by recognizing when they are not self-explaining effectively and by generating interventions to correct this behaviour. The capability of tailoring its interventions to each student's specific needs is what distinguishes the SE-Coach from other systems that try to foster self-explanation. These systems tend to either prompt students to self-explain every step in the instructional material (Aleven & Koedinger, 2002), or make students self-explain every incorrect step in a problem solution (Mitrovic, 2003). Although these less tailored approaches have shown that they can improve student learning as compared to problem solving without explaining, they may incur the danger of being intrusive by forcing spontaneous self-explainers to generate redundant and unnecessary self-explanations. The long term goal of our research is to investigate if and how more tailored support may improve on these existing approaches, in terms of both student learning and student satisfaction.

SE Coach: Overview

Research in cognitive science has identified a variety of different types of

self-explanations that consistently correlate with learning (Atkinson, 2002; Chi, Bassok et al., 1989).

In the SE-Coach, we have focused on three of these types (discussed in more detail below), that can be automatically formalised in a computational model, given a rule-based representation of the underlying domain knowledge. This allows the SE-Coach to have an internal representation of the relevant, correct self-explanations of these three types that can be generated for each available example. The SE-Coach uses this representation to monitor a student's self-explanations, to provide feedback on their correctness and to suggest additional self-explanations that may improve learning. We have decided to enable the SE-Coach to provide feedback on self-explanation correctness despite the fact that some researchers argue that an incorrect self-explanation can be as beneficial as its correct counterpart (Chi, 2000; Ryan, 1996). The argument that incorrect self-explanation should not be discouraged is based on the fact that, although it creates flaws in the student's knowledge, these flaws may later be contradicted by other elements of the example, thus triggering self-explanations to fix the flaws and generating learning. However, this argument may not apply to those students who are not proficient in self-monitoring their understanding during study tasks. These students may not be able to detect the inconsistencies generated by their incorrect self-explanations and would thus never overcome them on their on. We argue that immediate feedback on correctness protects these students from learning wrong knowledge from incorrect self-explanation, and simply makes students who are better at self-monitoring detect the conflict sooner than they would on their own.

The three types of self-explanations targeted by the SE-Coach are:

a) Justifying a solution step in terms of the domain theory (*step correctness* self-explanation)
b) Relating solution steps to goals in the abstract plan underlying the example solution (*step utility* self-explanation)
c) Filling in missing steps (*gaps*) in the example solution (*gap-filling* self-explanation).

We label these self-explanations "domain-based" because they involve relating example steps to existing domain knowledge, as opposed to self-explanations that involve using common-sense and overly-general knowledge to infer new domain knowledge that can justify a given example

step (Chi & VanLehn, 1991; Chi et al., 1994; Bielaczyc et al., 1995; Ryan, 1996). The latter type of self-explanations, also referred to as *Explanation-Based Learning of Correctness* (EBLC), tend to be much more open-ended than domain-based self-explanations. Currently, the SE-Coach cannot provide feedback for correctness and explicitly support EBLC via tools because doing so would require much more complex domain and student models. In a subsequent section, we discuss how the EA-Coach aims to provide support for EBLC by *implicit* means that do not involve interface tools and explicit feedback.

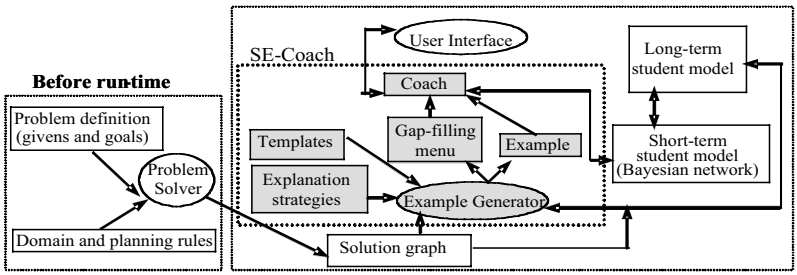


FIGURE 4
The ExBL architecture specialized for the SE-Coach.

The SE-Coach's focus on correct domain-based self-explanation is supported by the general system architecture presented in the previous section, extended as shown in Figure 4 to include components specific to the SE-Coach. In the context of the interaction with the SE-Coach, the solution graph for each available example represents a *model of correct self-explanation* for that example's solution, because for each solution step (i.e. fact in the solution graph) it encodes the three types of domain-based self-explanations needed to understand it as follows:

- *step correctness*, by encoding what domain rule generated that fact,
- *step utility*, by encoding what goal that fact fulfils, and
- *gap filling*, by showing how the fact derives from solution steps that may not be shown in the example solution).

To provide explicit monitoring and support for gap-filling self-explanation, the SE-Coach includes an example generator (see right part of Figure 4). This is a system for Natural Language Generation that can automatically tailor the level of detail in the example description to the student's knowledge, as we will describe shortly.

Interacting with the SE-Coach

In a previous section, we mentioned that one of the elements that defines the ExBL framework's pedagogy is the provision of incremental scaffolding for the meta-cognitive skills useful for effective example-based learning. As part of providing this incremental support, the SE-Coach's interface includes three different levels of scaffolding for self-explanation. These levels are designed to help students with different degrees of self-explanation capabilities self-explain more, while maintaining as much as possible the spontaneous, constructive nature of this learning activity.

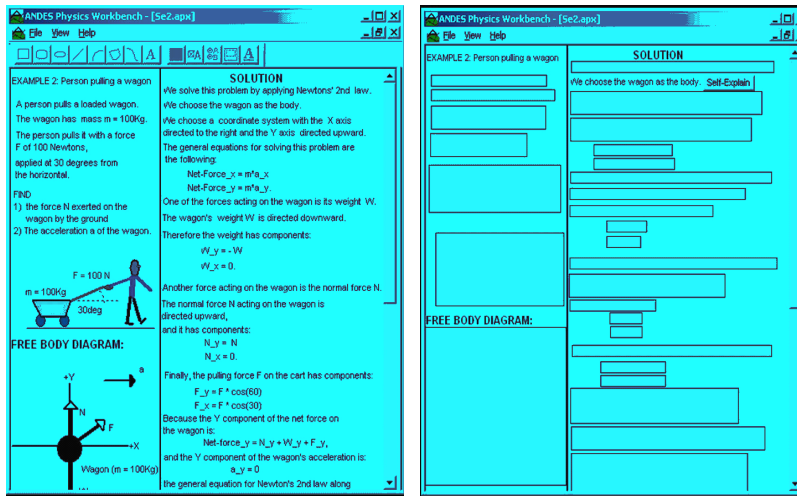


FIGURE 5
A physics example (left) presented with the SE-Coach masking interface (right).

The first level of scaffolding is given by a masking interface that presents different parts of the example covered by grey boxes (see Figure 5). In order

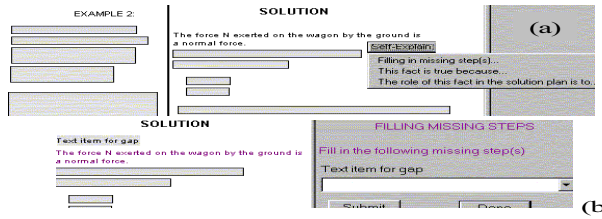


FIGURE 6
SE-Coach prompts for specific types of self-explanation.

to read the hidden text or graphics, the student must move the mouse over the corresponding box. The fact that not all the example parts are visible at once helps students focus their attention and reflect on individual example parts. Furthermore, it allows the SE-Coach to track students' attention (Conati & VanLehn, 2000). The second level of scaffolding is provided by explicit prompts to self-explain. These prompts go from a generic reminder to self-explain, that appears next to each example part when it is uncovered (see *self-explain* button in Figure 5, right), to more specific prompts for self-explanations on step correctness, step utility and gap filling that appear when a student clicks on the *self-explain* button for a given line (see Figure 6).

The third level of scaffolding consists of menu-based tools designed to provide constructive but controllable ways to generate the above self-explanations, to help those students that would be unable to properly self-explain if left to their own devices (Conati & Carenini, 2001; Conati & VanLehn, 2000). In the next sections we describe in more detail the various menu-based tools, in association with the types of self-explanation that they are meant to trigger.

Supporting self-explanations for step correctness

As Figure 6 shows, one of the prompts that appears in the self-explain menu for a given example part reads "This fact is true because...". This prompt aims to trigger self-explanations that justify why a step is correct in terms of the domain knowledge that generated it.

If a student selects the self-explanation prompt "*this fact is true because..*", a Rule Browser is displayed in the right half of the window (see Figure 7a). The rule browser contains a hierarchy of physics rules, reflecting the content of the ExBL framework's Knowledge Base. The student can browse the rule hierarchy to find a rule that justifies the currently uncovered

part. The SE-Coach will use a green check or a red cross to provide feedback on the correctness of the student's selection (see Figure 7a). To determine the correctness of a selection, the selection is matched to the rule that derives the uncovered example step in the solution graph.

If a student wants to provide a more detailed explanation of a rule, s/he can optionally click on the "Template" button at the bottom of the Rule Browser (see Figure 7a). A dialog box comes up, containing a template corresponding to a partial definition of the rule that has blanks for the student to fill in (see Figure 7b). Clicking on a blank brings up a menu of possible fillers (see right template in Figure 7b). The rule definition is in terms of the preconditions that need to be verified for the rule to be applied and of the consequences that the application of the rule generates, consistent with findings showing that this is the spontaneous way in which self-explainers tend to express their self-explanations for correctness (Chi & VanLehn 1991).

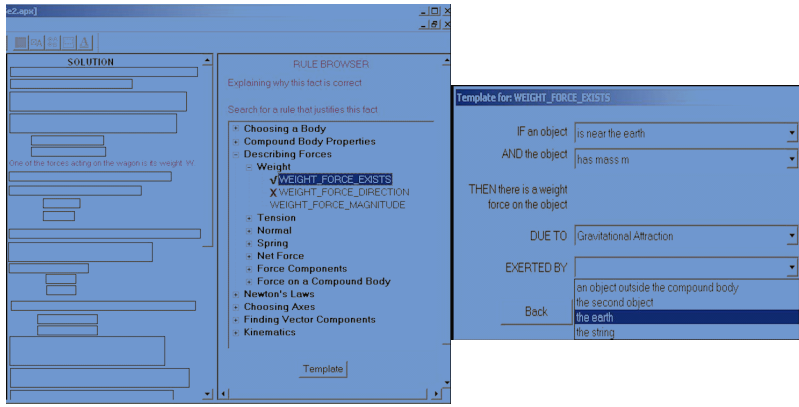


FIGURE 7
(a) Selections in the Rule Browser; (b) Template filling.

After completing a template, the student can select "submit" to get immediate feedback. The SE-Coach retrieves the definition of the corresponding rule from the Andes' Knowledge Base and uses it to verify the correctness of the student's selections. Student performance in filling in a given template is treated by the system as evidence of the student's knowledge on the corresponding rule (or lack thereof), and is therefore used

to update that rule's probability in the short-term student model for the current example.

Supporting self-explanation for step utility

The second type of prompt that appears in a self-explanation menu ("*the role of this fact in the solution plan is*…") aims to make the student abstract the function of the current step in a high-level plan underlying the example solution. When a student selects this type of prompt, a plan browser is activated. The plan browser is similar to the rule browser, but it displays a hierarchical tree representing the solution plan for a particular example instead of the SE-Coach's physics rules. For instance, Figure 8 shows the Plan Browser for the 'person-pulling-a-wagon' example (Figure 2), which displays the plan to apply Newton's Second Law (Reif, 1995). To explain the role of the uncovered fact in the solution plan, the student navigates through the goal hierarchy and selects a sub-goal that most closely motivates the fact. Pressing a "submit" button causes the SE-Coach to give immediate feedback and to send the corresponding assessment of student performance as evidence to the short-term student model. In the student model, performance assessment is accomplished by matching the student selection with the rule that established the solution graph's goal node closest to the currently uncovered step.

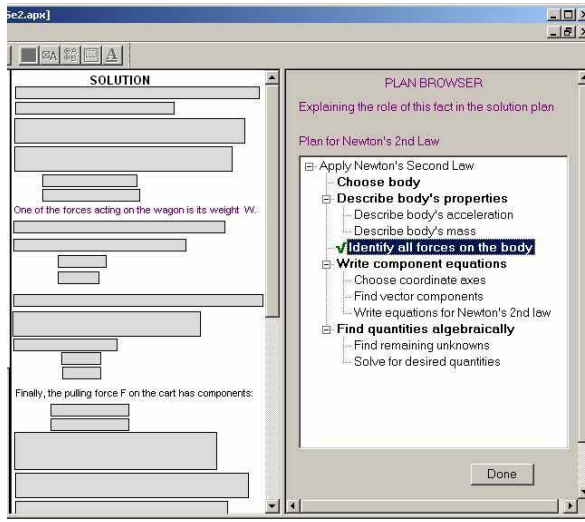


FIGURE 8
Selection in the Plan Browser.

Supporting gap-filling self-explanation

The two types of self-explanations described above are supposed to help students clarify steps that are explicitly shown in the example solution. Studying detailed example solutions has been shown to be beneficial for acquiring problem-solving skills because it reduces the high cognitive load that pure problem solving tends to induce in novice students (Sweller, 1988). This high cognitive load is caused by the fact that during problem solving one needs to both search for a reasonable solution path and understand how individual pieces of domain knowledge can be applied to implement this path. Detailed example solutions are a good starting point for novices because they eliminate the search component of the problem-solving process, allowing students to focus on understanding domain knowledge application. Self-explanations for correctness and utility have been shown to aid this process (e.g., (Chi, Bassok et al., 1989)). However, a third type of self-explanation has also been shown to aid the acquisition of problem-solving skills. If students are presented with example solutions with missing steps, they may learn by self-explaining how to fill such solution gaps (Atkinson, 2002). A likely reason for the effectiveness of this type of self-explanation is that it provides mini problem-solving sessions that help the transition from pure example studying to full-blown problem solving. We argue that this transition can be further facilitated by:

- introducing in the example solutions gaps that are not too cognitively demanding for a given student, and
- supporting student self-explanations aimed at filling these gaps.

To test our hypothesis, we have given the SE-Coach the capability to both generate examples with tailored solution gaps and coach gap-filling self-explanation over these gaps. The SE-Coach decides which gaps are suitable for a given student by monitoring how a student's knowledge changes while studying a sequence of examples, and by only selecting gaps that relate to knowledge the student seems to have mastered after studying fully detailed examples.

As shown in Figure 4, the tailored example solutions are generated by the SE-Coach Example Generator (EG from now on) with input from the long-term student model. When a student selects an example from the SE-Coach pool, EG uses the corresponding solution graph and a set of *explanation strategies* (Figure 4) to create a *text plan* for the example. This is a data structure that specifies the content, organization and rhetorical structure for

a natural language description of the complete example solution represented in the solution graph (Conati & Carenini, 2001).

Once the text plan has been generated, it is revised to possibly insert solution gaps suitable for the current student. More specifically, the revision process examines each proposition specified by a primitive communicative action in the text plan (i.e., an action that would be realized as a solution step in the example presentation). If, according to the long-term student model, there is a high probability that the student knows the rule necessary to infer that proposition, the action is de-activated. After this, all active communicative actions are realized in English by applying a set of linguistic templates (Figure 4). In contrast, de-activated actions are kept in the text plan but are not realized in the example text, thus creating solution gaps. However, as we will see shortly, de-activated actions may be realized in follow-up interactions.

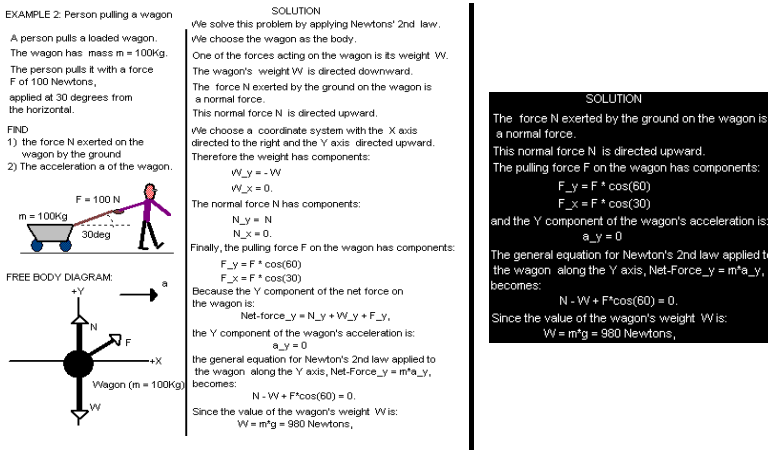


FIGURE 9
(left) Portion of Example 2 without solution gaps; (right) solution with added gaps.

As an illustration of the effects of the revision process on content selection, compare the example solutions shown in Figure 9. They are both solutions for the example shown in the figure (Example 2), but the solution to the left does not contain any solution gaps. In contrast, the same portion of the solution to the right in Figure 9 is much shorter, because it includes

several solution gaps. As previously described, EG determines what information to leave out by consulting the long-term probabilistic student model. In particular, the concise solution in Figure 9 (right) is generated by EG if the student model gives a high probability that the student knows the rules necessary to derive the missing steps (e.g., the rules that describe when to apply Newton's Second Law, how to choose a body to apply this law, when a weight force exists and with which direction, etc.), possibly because the student previously studied and self-explained other examples involving these rules. When selecting the content for Example 2, EG leaves out all the propositions derived from the rules with high probability of being known. Notice, for instance, that the concise solution in Figure 9 does not mention the solution method (Newton's second law) used, or the weight force existence/direction. Also, the choice of the body and of the coordinate system is only conveyed indirectly.

To support student self-explanation targeted at filling the inserted gaps, a specific prompt is added to the list of prompts that appear after a student clicks the *self-explain* button. The prompt is added only when the uncovered example part is enabled by solution steps corresponding to some of the primitive communicative actions that were de-activated during the revision process. The rationale behind this condition is that a solution gap with respect to an example part comprises all the solution steps that were left out, but whose understanding is a direct precondition to derive that example part. For instance, given the part of Example 2 uncovered in Figure 10a, there is only one solution gap preceding it, namely the one corresponding to the step of choosing the wagon as the body for the current problem. As shown in Figure 10a, the prompt for gap filling is generated by adding the item "*filling in missing steps*" to the self-explain menu. If the student clicks on this item, the interface inserts in the solution text an appropriate number of masking boxes, representing the missing steps (see Figure 10b, left panel, first box from top). The interface also activates a dialogue box containing a blank for each missing step, that the student can use to fill in the step (see Figure 10b, right panel). Since the interface currently does not process natural language input, the student fills each blank by selecting an item in the associated pull-down menu. EG generates the entries in this menu by realizing in natural language some of previously de-activated communicative actions in the text plan (see 'Gap filling menu' component in Figure 4). These will include both communicative actions related to other parts of the solution, which will thus represent incorrect selections, as well as the correct one.

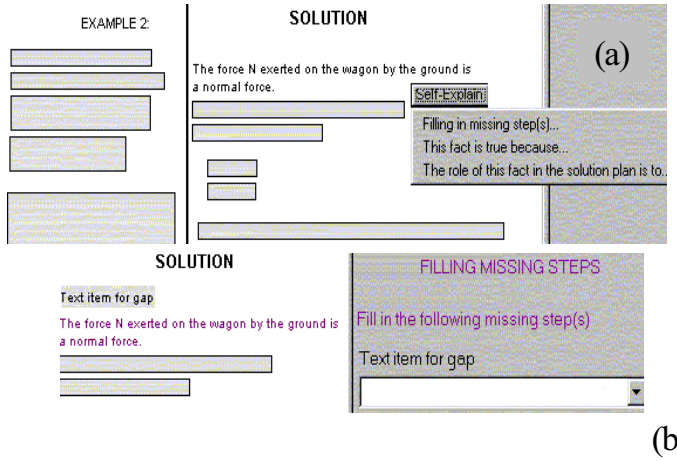


FIGURE 10
Interface tools for gap-filling self-explanation.

The student receives immediate feedback on the correctness of her selection, which is sent as evidence to the short-term student model corresponding to the Bayesian network built for the current example (see Figure 4). The network node that corresponds to the missing step is set to either true or false, depending on the correctness of the student's selection, and the network updates the probability that the student knows the corresponding rule. Thus, if the student's actions show that s/he is not ready to apply a given rule to fill a solution gap, this rule's probability will decrease in the student model. As a consequence, the next presented example involving this rule will include the related solution steps, giving the student one more chance to study the application of the rule in an example-studying condition.

The SE-Coach's Interventions

The interface tools described in the previous sections are designed to provide incremental scaffolding for self-explanation that students can access at their own discretion, to adhere to our goal of providing an environment that stimulates as much as possible the student's initiative in the learning process. However, to help those students who are not receptive to interface scaffolding because of their particularly low tendency for self-explanation, the SE-Coach can also provide more direct tutorial interventions, targeting specific limitations in a student's self-

explanation behaviour. In the rest of this section we describe the design of these tutorial interventions.

Initially, self-explanation is voluntary. However, the SE-Coach keeps track of the students' progress through the example; including how much time they looked at a solution item (through the masking interface), what they chose to self-explain via the interface tools and whether or not the self-explanations were correct. This information is collected in the SE-Coach's student model, which assesses what parts of the example may benefit from further self-explanation (Conati & VanLehn, 2001). Then, when the student tries to close the example, the SE-Coach generates tutorial interventions to make the student self-explain those parts. These interventions include:

1. A generic warning: "*You may learn more by self-explaining further items. These items are indicated by pink covers*". The warning is accompanied by changing the colour of the relevant boxes to pink (shown as dark grey in Figure 11).
2. A specific prompt attached to each pink box, such as "*Self-explain with the Rule Browser*", "*Self-explain with both the Rule and the Plan Browser*" or "*Read more carefully*", depending on what self-explanation the student model predicts to be missing for that item.

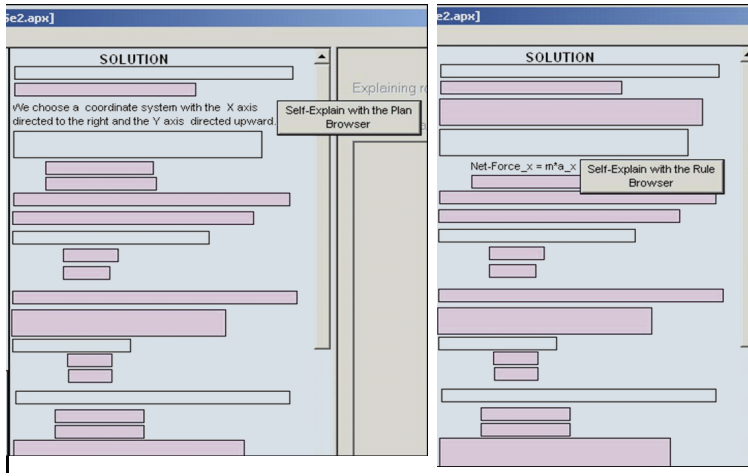


FIGURE 11
SE-Coach interventions to elicit further self-explanation.

When the item is uncovered, the more specific prompt appears in place of the simple *self-explain* button (see Figure 11). The colour of the boxes and the related messages change dynamically as the student performs more reading and self-explanation actions that change the probabilities of the corresponding nodes in the student model. The student is not forced to follow all of the SE-Coach suggestions, and can close the example whenever desired.

One of the challenges in designing the SE-Coach tutorial interventions is that they must motivate students who have low propensity to self-explain to do so. The current design based on the colouring of example lines was selected through pilot testing of various alternatives (Conati & VanLehn, 2000) because it allows the students to see at once all the parts that they should further self-explain as well as what interface tools they should use for that purpose. It also gives students suitable feedback on the progress they are making, because the colour of the solution steps and the attached hints change dynamically as students generate more self-explanations.

SE-Coach: Summary

In this part of the paper we have described the SE-Coach component of our adaptive ExBL framework for example-based learning. The SE-Coach aims to improve student problem-solving skills by helping students learn from worked-out example solutions before solving problems. To do so, the SE-Coach provides individualized scaffolding for various forms of self-explanation, a meta-cognitive skill that has been widely shown to improve the effectiveness of example-studying activities. One of the self-explanations the SE-Coach supports, gap-filling self-explanation, is meant to ease students into more direct problem-solving activities, while still in example-study mode. We will now describe how the transition from example studying to problem solving is further facilitated by the EA-Coach, the other component of the ExBL framework.

THE EA-COACH: PROVIDING TAILORED SUPPORT FOR ANALOGICAL PROBLEM SOLVING

Example studying is more beneficial to learning than problem solving for students in the early stages of skill acquisition. As students gain expertise, problem-solving activities become more beneficial (Kalyuga, Chandler et al., 2001). However, as students move to problem solving, they still tend to

rely on worked-out examples to overcome impasses (e.g., (Reed & Bolstad, 1991; Reed, Dempster et al., 1985; VanLehn, 1998; VanLehn, 1999; VanLehn & Jones, 1993)). This process is referred to in the literature as analogical problem solving (APS), and can be beneficial to learning. However, as for pure example studying, APS can have different degrees of pedagogical effectiveness, depending upon the meta-cognitive skills that students bring to bear (VanLehn, 1998; VanLehn, 1999; VanLehn & Jones, 1993). For some students, APS is simply a way to overcome problem-solving impasses by copying from existing example solutions. For others, APS affords the opportunity to acquire or refine relevant domain knowledge (VanLehn, 1998; VanLehn, 1999; VanLehn & Jones, 1993). In this section, we describe how the ExBL framework discourages the former type of APS and encourages the latter.

Analogical problem solving can be characterized by two phases: example *selection* and *application* (VanLehn 1998). *Selection* involves retrieving an example that helps the student solve the target problem. Research indicates that this phase is mediated by expertise, in that novice students tend to experience difficulties finding appropriate examples (e.g., Novick, 1988). *Application* involves using the example's solution to help generate the target problem solution. The learning outcomes from this phase are heavily influenced by a number of meta-cognitive skills that can be characterized using two dimensions: *analogy-type* and *reasoning*.

The *analogy-type* dimension characterizes a student's preferred style of problem solving when examples are available (VanLehn, 1998; VanLehn & Jones, 1993) and includes *Min-analogy* and *Max-analogy*. Min-analogy identifies students who try to solve a problem on their own, and refer to an example only when they reach an impasse or to check a solution. Max-analogy identifies students who copy as much as possible, regardless of whether they can solve the problem on their own. These students miss the opportunity to practice their existing problem-solving skills, as well as the opportunity to run into impasses that would allow them to detect and overcome knowledge gaps. Empirical results based on protocol analysis demonstrate that students who prefer min-analogy tend to learn more, because it directly supports the benefits of APS, including knowledge strengthening and refinement (VanLehn, 1998; VanLehn & Jones, 1993).

The *reasoning* dimension of the APS application phase characterizes how a student tries to understand the example solution, in order to overcome a problem-solving impasse. Impasses occur either because of a lack of understanding of how to apply existing knowledge (e.g., because multiple

principles apply), or because domain knowledge is missing. When referring to an example to overcome the first type of impasse, students may use the domain-based self-explanations (described earlier) in order to gain an understanding of which rule to apply. However, when referring to an example to overcome an impasse due to missing knowledge, students need to use a different type of self-explanation to extract the necessary information from the example solution. This self-explanation is known as *Explanation-Based Learning of Correctness* (EBLC). EBLC can be used to learn new domain principles from an example's solution, either during pure example studying or during APS, by using common-sense and overly-general knowledge in conjunction with domain principles (VanLehn, 1999).

To demonstrate how EBLC works, Figure 12 (left) shows a problem and an example in the domain of Newtonian physics[2]. Let's suppose that a student 1) tries to self-explain the example before working on the problem and 2) when self-explaining solution steps 3 and 4, she reaches an impasse because she does not know about normal forces. In step 3, the example solution states that there is a normal force acting on the crate, but does not state why. Figure 12 (right) shows how commonsense and overly-general rules, in conjunction with a domain rule can be used to explain the existence of the normal force mentioned in step 3 of the example (Figure 12, left) (VanLehn, 1999). Specifically, to explain this rule, the student relies on her *existing* (1) commonsense knowledge to infer that since the crate is supported by the ramp, it pushes down on the ramp (*Commonsense* rule in Figure 12); (2) overly-general knowledge to infer that this push is an 'official physics force' (*Overly-general* rule in Figure 12); (3) domain knowledge to infer that there is a reaction force that acts on the crate and is due to the ramp (*Newton's Third Law* rule in Figure 12). This line of reasoning results in a new domain principle (*Normal-exists* rule, Figure 12). A similar process can be used to explain example step 4 (Figure 12, left) specifying the direction of the normal force. Once the student gains knowledge about normal forces and their direction through EBLC, she can apply it to generate solution steps 3 and 4 in the target problem.

Empirical results based on protocol analysis provide some indication that certain students have an inherent tendency for EBLC (VanLehn, 1999). Unfortunately, many students employ more superficial reasoning processes, which either do not result in learning, or result in shallow forms of knowledge (Reed, Dempster et al., 1985; VanLehn, 1998; VanLehn, 1999).

[2]Only a portion of the problem and example solutions is shown.

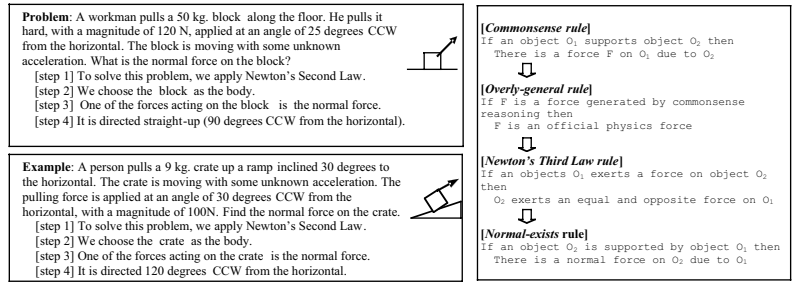


FIGURE 12
Problem & Example (left) and Reasoning via EBLC (right).

One such process employed during APS involves performing minimal adaptations of example lines so that they can be copied over to the problem solution. This can be done, for instance, by substituting example constants by problem ones (e.g., the example constant *crate* by the problem constant *block* in Figure 12). This process is known as *transformational analogy* or *mapping-application* (Anderson, 1993; VanLehn, 1998), and may prevent students from learning new rules during analogical problem solving because the correct solution is generated mostly through copying of superficially adapted example lines.

The above findings suggest that, as for example studying, there is much to gain by providing students with adaptive support for meta-cognitive skills that foster learning during APS. In the ExBL framework, the EA-Coach is designed to provide this support for students by recognizing when they are engaging in suboptimal APS behaviors and by generating interventions to trigger the more effective meta-cognitive skills.

Although both ExBL Coaches follow the philosophy of delivering tailored, scaffolded support for meta-cognitive skills, the EA-Coach differs slightly from the SE-Coach in how it delivers its support. Our hypothesis is that students should be provided with explicit types of scaffolding when they are in the early stages of skill acquisition, such as when they interact with the SE-Coach. However, some of this scaffolding should be faded eventually, to encourage students to become less reliant on tools. Thus, although the EA-Coach provides some interface scaffolding, instead of providing other tools the coach provides a more subtle form of scaffolding. This scaffolding is realized by the system-supported selection of suitable examples. However, this adds a new level of challenge, because it is hard to

assess students' self-explanation behaviour in the absence of rich interface tools. In a later section, we will see how the EA-Coach responds to this challenge by using additional information on various types of differences between the current problem and example to assess students' reasoning during APS.

The EA-Coach focuses on supporting the two key meta-cognitive skills needed for APS (i.e., min-analogy and self-explanation via EBLC). Although domain-based self-explanations may also have the potential to be beneficial during APS, we have chosen to focus only on these two skills because we have cognitive science findings to support their role in effective APS (VanLehn, 1998; VanLehn, 1999; VanLehn & Jones, 1993). Our plan is to first evaluate the current level of support provided by the EA-Coach, and then use the outcome of these evaluations to determine if and how domain-based self-explanations should be incorporated.

We will describe how the EA-Coach aims to encourage EBLC, as well as min-analogy, after introducing the overall system.

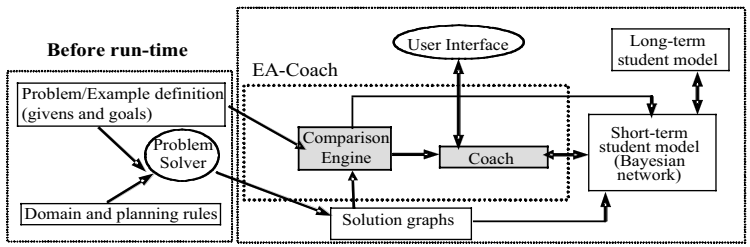


FIGURE 13
The ExBL Architecture specialized for the EA-Coach.

EA-Coach Overview

The EA-Coach provides support for min-analogy and EBLC following the general ExBL framework philosophy of providing multiple levels of scaffolding tailored to individual students' needs. The EA-Coach architecture extends the ExBL framework with components specific to the EA-Coach, as shown in Figure 13. The system contains a database of worked-out examples and problems for the student to work on (the textual description of the example solutions must currently be entered by a human being). As with the SE-Coach, the *Problem Solver* generates the system's internal representation of the example and problem solutions in the solution graph format we introduced earlier (Figure 3). In the context of the EA-Coach, it is the solution graph corresponding to the problem the student is working on that forms the basis of the EA student model (Figure

13, *short-term student model* component). As for the SE-Coach, the student model allows the EA-Coach to assess students' problem-solving performance and provide feedback on it. The solution graph also serves a second function for the EA-Coach: it is used to compare the target problem and an example to assess the differences between them (*Comparison Engine* component). This assessment serves two crucial functions in the EA-Coach:

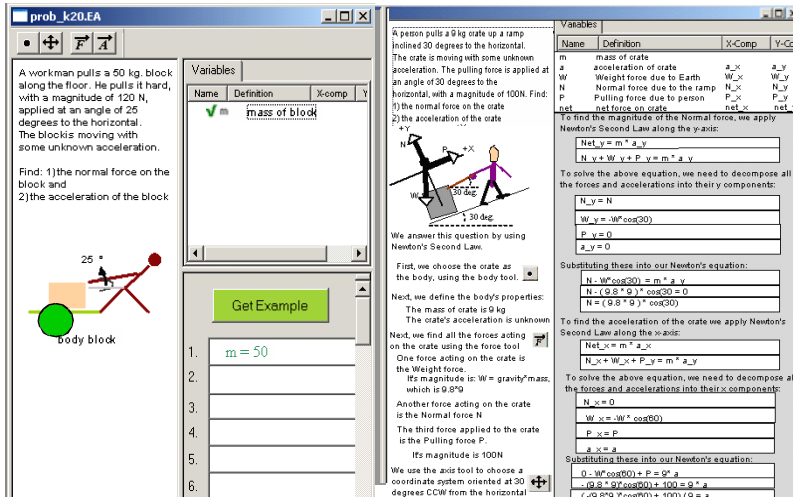
- to allow the *Coach* component to select examples that trigger effective APS behaviours, thus providing implicit support for learning through this process,
- to help assess students' cognitive and meta-cognitive states during APS.

Interacting with the EA-Coach

We will now introduce how students interact with the EA-Coach. The EA-Coach interface allows students to solve problems and refer to worked-out examples (problem and example windows, Figure 14, left and right respectively). To work on a problem, the student selects the one of her choice, which opens the problem window. The problem-solving interface is directly based on the Andes design (Conati, Gertner et al., 2002) and consists of the diagram and equation panes, which students can use to draw free-body diagrams (problem window, left, Figure 14) and type equations (problem window, right, Figure 14). To help the system assess the correctness of students' equations, any variables that students include in their equations must first be added to the variable definition pane (problem window, top, Figure 14).

During problem solving, students can ask the system to provide an example, which the EA-Coach presents in the example window (Figure 14, right). Note that the EA-Coach example format differs a little from the SE-Coach design. We initially intended to use the SE-Coach format in the EA-Coach, which is loosely based on the presentation of worked-out examples in physics text books. However, a pilot study with the EA-Coach revealed that students felt this format was not sufficiently similar to the Andes problem-solving window's design. One common complaint was related to the lack of a variable definition pane in the example window. Although this lack of similarity could be a form of scaffolding to discourage copying, we found that for several subjects it hindered problem solving. To address this issue, we decided to change the example format to more closely mirror the problem-solving format[3].

[3]To avoid discrepancy between example formats in the two coaches, we are working on implementing the same changes into the SE-Coach example presentation window. We chose to present the old SE-Coach format in this paper because it was used in the study we describe in a later section.



Problem Window Example Window

FIGURE 14
EA-Coach Problem (left) and Example (right) Interface.

As is the case with the SE-Coach, the general ExBL framework principle of providing incremental support is realized in the EA-Coach by offering several levels of scaffolding for min-analogy and EBLC. In this section, we describe the scaffolding that is embedded in the EA-Coach interface, while in the next section we introduce the scaffolding provided through tailored example selection.

One form of interface scaffolding corresponds to providing immediate feedback for students' problem-solving entries, realized by colouring these entries as either red or green. As evidence in cognitive science demonstrates (Chi, Bassok et al., 1989), and as we confirmed through our pilot studies, some students lack self-monitoring skills and so are unable to diagnose their own misconceptions or errors. We argue that immediate problem-solving feedback can help trigger the right APS behaviours in these students, who would otherwise continue on incorrect solution paths. For instance, if a problem-solving error is due to a max-analogy student indiscriminately copying from an example, immediate feedback can discourage this form of copying by making the student aware of max-analogy's limitations. If the error is the result of a knowledge gap the student is not aware of, then the immediate feedback can help the student realize the existence of an impasse, and encourage using an available example to overcome it.

A second form of interface scaffolding in the EA-Coach is provided by the same masking interface used by the SE-Coach. As for the SE-Coach, this interface can help focus a student's attention on individual solution lines. In the context of the EA-Coach, this interface also serves a second function: it is intended to discourage copying because of the effort needed to explicitly uncover example solution steps.

To further discourage copying, the interface includes a third form of scaffolding corresponding to the lack of 'Cut' and 'Paste' functionality between the example and problem windows. This design is based on findings from an earlier pilot study involving an interface that allowed cutting and pasting. That study showed that some students abused these functionalities to copy entire example solutions.

Supporting APS via Example Selection

In addition to the interface scaffolding described in the previous section, the EA-Coach scaffolds effective APS behaviours by selecting examples that discourage max and transformational analogy, while facilitating min-analogy and EBLC. The example selection is tailored to a student's needs, according to the ExBL principle of providing tailored support for meta-cognition. This is one of the key, original contributions of the EA-Coach, which we will discuss further in the related work section.

A fundamental factor that the system must take into account in order to find an appropriate example is the differences between it and the current problem, and how these impact APS meta-cognitive behaviours. Thus, before describing the selection process in more detail, we first describe this impact.

Impact of Differences between Problem and Example

Typically, an example is different from the problem in some ways (otherwise, the example is simply a direct solution). Our approach for providing tailored support to APS relies on assessing structural and superficial differences between the problem and the example solutions steps. This allows the system to determine whether a student can correctly transfer a solution step from an example to a problem and how easily she can do it, as we demonstrate in the next section.

Two corresponding solution steps are defined to be structurally *different* if they are not generated by the same domain principles (e.g., rules in the solution graph in Figure 3), and *identical* otherwise. Two structurally identical steps may be superficially different. We classify these differences as *trivial* or *non-trivial*, as defined below.

Given:

P = Pd + Ps where Pd = problem description , Ps = problem solution
E = Ed + Es, where Ed = example description, Es = example solution
PsolStep = a solution step in Ps containing constant cp
EsolStep = a solution step in Es containing constant ce
cp ≠ ce and PsolStep, EsolStep are structurally identical

If constant c_e is in E_d and constant c_p is in P_d and if replacing c_e by c_p in $E_{solStep}$ generates a corresponding correct solution step $P_{solStep}$ in P, then the difference between steps $P_{solStep}$, $E_{solStep}$ is classified as *trivial*. Otherwise, the difference is classified as *non-trivial*.

To make this more concrete, Figure 15 illustrates how the EA-Coach classifies the structural and superficial relations between solution steps 3 and 4 in the problem and example originally shown in Figure 12. To do so, the system relies on the solution graph structure introduced earlier, which contains all the solution steps (i.e., facts) and corresponding rules, as well as its internal representation of the problem and example description. (Figure 15 shows both the textual representation seen by the student and the system's corresponding internal representation; for the solution steps, the corresponding rules are also shown). Since problem steps 3 and 4 have a structurally identical counter-part in the example solution (i.e., are generated by the same rules, *R:Normal-exists*, *R:Normal-dir respectively*), both are classified as structurally identical. On the other hand, both steps 3 and 4 are superficially different from their corresponding example steps. The difference between step 3 in the problem and example is classified as trivial, because (i) it is due to problem/example constants *block* and *crate*, which are found in both the problem/example descriptions and solutions; (ii) the example constant *crate* can be replaced by the problem constant *block* to generate the correct solution step in the problem. On the other hand, the superficial difference between problem and example steps 4 is *non-trivial*. The difference relates to the fact that for the problem solution, a normal force is drawn straight-up, as opposed to perpendicular to the ramp for the example. This difference depends on a constant defining the incline of the surfaces on which the block and crate rest, which only explicitly appears in the example specification (see Figure 12), but which also requires additional inference in order to be reconciled (i.e., that the force is directed 90 degrees away from the surface's incline). Note that our classification of the structural and superficial relations relies on comparing the problem and

example solutions, which the EA-Coach has access to (students only have access to the example solution).

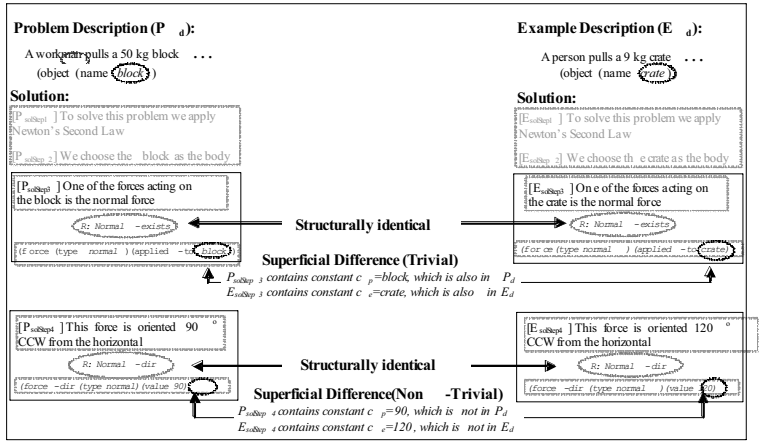


FIGURE 15
Classification of Differences.

Now the key question is: "What impact do various differences have on students' APS behaviours?" The answer to this question depends both on whether the student tends to spontaneously engage in effective APS behaviours, and on whether the student possesses the relevant knowledge.

The impact of a structural difference between a problem and example solution step depends on whether the student has the knowledge to generate the step on her own. If the student does know the rule, then the difference forces her to do pure problem solving, which can help her strengthen her knowledge through practice. If the student does not, however, the example will not be helpful for acquiring the domain principle(s) needed to generate the problem solution, and no learning gains will occur (as found in Novick, 1988).

On the other hand, superficial differences do not block students from transferring from the example, which increases the chances that they can carry on in their problem solving. This is because these types of differences afford students the opportunity to either copy the solution step directly over to the target problem, or to learn the domain principle embedded in the example's solution and apply this knowledge to generate the corresponding

step in the target problem. However, we propose that the two kinds of superficial differences we introduced above may have different impacts on whether learning occurs:

- because *trivial* differences can be resolved by simple constant substitution and so facilitate copying, they do not stimulate EBLC and min-analogy for students who do not spontaneously engage in these processes and have poor knowledge. There is some evidence backing up this assumption: students do not have difficulty reconciling trivial differences during APS to generate the problem solution, but do not always learn the underlying domain principles from doing so (Reed, Dempster et al., 1985; VanLehn, 1998), possibly because they are not using the appropriate APS skills;
- *non-trivial* differences can have a positive impact on learning for students with poor APS skills, because this type of difference:
  - can only be resolved by EBLC (and not transformational analogy). In particular, a student who tries to copy a solution line containing a non-trivial difference will not obtain a correct answer. If she is made aware of the error through feedback, she may be encouraged to reason more deeply via EBLC to learn the domain principle that generated the example solution (this newly acquired knowledge could then be applied to generate the problem solution);
  - has the potential to discourage max-analogy by making pure copying an ineffective strategy.

In the next section, we illustrate how the EA-Coach incorporates the impact of various differences between a problem and an example into its example selection mechanism.

Example Selection Process

Understanding the impact of both superficial and structural differences allows the EA-Coach to find an example that meets its general goal of providing support for APS. This goal is divided into the following two sub-goals:

- finding an example that triggers learning by encouraging min-analogy and EBLC and discouraging their ineffective counterparts, i.e., max and transformational analogy (*learning goal*),
- finding an example that helps to solve the target problem (*problem-solving success goal*).

Both of these sub-goals need to be taken into account in order to find an appropriate example. Examples that introduce certain types of differences do have the potential to encourage learning, but decrease the chances of problem-solving success by making the full solution more difficult (or impossible) to transfer. For instance, an example may encourage the learning of a particular rule due to a non-trivial superficial difference, but also include a structural difference with the problem (because its solution does not involve a domain principle required for the problem's solution). If this difference corresponds to a student's knowledge gap, the example will not be useful in helping the student overcome the problem-solving impasse caused by the knowledge gap, and thus the student will not be able to solve the problem. On the other hand, examples that virtually guarantee problem-solving success because they are very similar to the target problem facilitate max and transformational analogy and therefore may hinder learning. To find the best balance in satisfying both the learning and problem-solving success goals, the EA-Coach assesses an example's impact by taking into account the following factors:

- superficial and structural differences between the example and the target problem,
- the student's knowledge of the domain principles needed to generate the problem solution,
- the student's tendency for min-analogy and EBLC.

We will now describe how these factors are used by the EA-Coach to find an example that best satisfies the two selection sub-goals for a specific student. This involves a two-step process.

As the first step, for each example in the example pool, the EA-Coach uses its student model to simulate what a student will learn from a given example and how the example will help her solve the target problem. To see how this simulation operates, recall that the student model is a Bayesian network built upon the *problem's* solution graph (such as the one shown in Figure 3). As for the SE-Coach, this network includes nodes representing the probability that the student: 1) knows the rules that generated the problem solution (represented by the rule nodes in the solution graph), 2) can generate the corresponding solution steps (represented by the fact nodes in the solution graph). In addition, the network includes nodes that explicitly represent a student's tendency for min-analogy and EBLC reasoning, as well as nodes representing the similarity between steps in the target problem

and selected example (Muldner & Conati, 2005). To simulate how the example impacts learning of the rules involved in the problem-solving process (the first selection sub-goal), the similarity between the example and the problem solution steps is introduced as evidence to the network. This causes the network to update the probability that the student will learn the rules embedded in the problem solution by referring to the example. Learning of a specific rule is predicted to occur if:

- there is low probability that the student knows it, and
- the example solution includes the rule (i.e., there is no structural difference) and
- *either* there is high probability that the student has good APS meta-cognitive tendencies, or the example shares a non-trivial superficial difference with the problem that encourages EBLC on this rule by blocking copying of the corresponding step.

For instance, given a student with poor knowledge and poor APS meta-cognitive tendencies, and the problem/example pair shown in Figure 15, the simulation would predict that there is a low probability the student will learn the domain principle associated with the rule corresponding to step 3 (*R:Normal-exists* in Figure 15). This is because for this step, the difference with the corresponding problem step is a superficially trivial one (as illustrated in Figure 15), which allows the student to follow her tendency for max-analogy and copy the step instead of learning the rule. On the other hand, the simulation would predict that there is a higher probability the rule corresponding to step 4 (*R:Normal-dir* in Figure 15) will be learned. This is because although this student has a low tendency for min-analogy and EBLC, the system predicts that the non-trivial superficial difference between this step and the corresponding problem step encourages EBLC by making copying impossible.

To simulate how the example impacts problem-solving success (the second sub-goal of the example selection process), the similarity between the problem and example solution steps is used by the model to update the probability that the student will be able to generate the problem solution steps in the presence of the example. The simulation predicts that problem-solving success for a target solution step will occur if:

- *either* the student is able to generate the solution step on her own accord
- *or* the example solution includes the corresponding rule (i.e., there is no structural difference) and
    - *either* the example allows the student to copy the step from its solution

- *or* the student is able to generate any pre-requisite solution steps *and* the example allows the student to learn the rule needed to generate the step from its solution and apply it to the problem (although the latter generates a lower probability of problem-solving success than if the step can be directly copied).

Returning to our example of the student with poor knowledge and APS meta-cognitive tendencies, the simulation would predict that there is a high probability that the student will be able to generate the steps needed for the problem shown in Figure 15, because they all have structurally-identical counterparts in the example[4]. This probability will be slightly higher for steps that the simulation predicts this student will simply copy from the example, as is the case for trivially different steps (e.g., step 3 in Figure 15), than for steps for which this student has to learn the corresponding rule because a non-trivial difference exists (e.g., step 4 in Figure 15).

As described above, the simulation generates a prediction of how an example will help a student solve the target problem, and what will be learned during the process, corresponding to the Bayesian network's fact and rule probabilities respectively (recall that this network corresponds to the problem's solution graph). Given the outcome of this simulation, which as we said is repeated for each example in the system's example pool, the second step in the example selection process involves finding the 'best' example. To do so, the framework relies on a decision-theoretic approach (Clement, 1996), and assigns a utility to each available example. To calculate this utility, the framework uses a multi-attribute linearly-additive utility model (see Figure 16) in order to meet the learning and problem-solving success objectives. We will now describe this model in more detail.

First, utility nodes are added and linked to the network's rule nodes (e.g., '*Utility Rule1*', '*Utility Rule2*' nodes in Figure 16). The value of each utility node is the expected utility (EU) of the corresponding rule ('*Rulei*' nodes in Figure 16), which corresponds to the sum of the probability *P* of each outcome 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, $U(known(Rulei))=1$ and $U(\neg known(Rulei))=0$, the expected utility of a rule corresponds to the probability that the rule is

[4]Figure 15 only explicitly shows the structural relations for steps 3 and 4, but all four steps are classified as structurally identical to the corresponding example steps because they are generated by the same rules in the problem and example solutions.

known. The '*Utility Rulei*' nodes are linked to a multi-attribute utility (MAU) node representing the *learning* objective ('*Learning Utility*' node in Figure 16). The value of this node is the weighted sum of its input values:

$$\sum_{i}^{n} EU(Rule_i) \cdot w_i$$

Since we consider all the rules to have equal importance, all the weights w are assigned an equal value (i.e., 1/*n* , where *n* is the number of rules in the network).

A similar approach is used to obtain the measure for the problem-solving success objective: utility nodes are added and linked to the network's fact nodes (e.g., '*Utility Fact1*', '*Utility Fact2*' nodes in Figure 16), where *U*(*generated*(*Facti*))= 1 and *U*(¬*generated*(*Facti*))= 0. These utility nodes are used as inputs to the '*PS Success Utility*' node, whose value is a weighted sum of its inputs. As was the case for rule nodes, we assume that all the facts are equally important, and so the weights are assigned an equal value (i.e., 1/*n*, where *n* is the number of facts in the network).

Finally, as shown in Figure 16, the '*Learning Utility*' and '*PS Success Utility*' nodes are linked to a MAU node representing the overall utility of an example ('*Overall Utility*' node in Figure 16), whose value corresponds to the weighted sum of its inputs. These weights are currently set to the same value (i.e., 1/2), since we assume learning and problem-solving success to

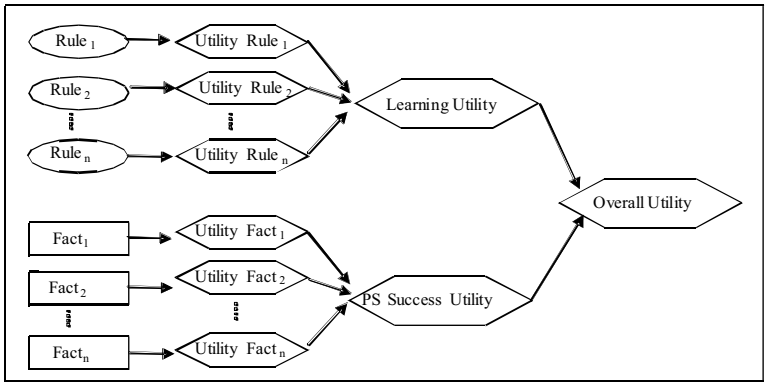


FIGURE 16
Fragment of the EA Utility Model.

be equally important, but we plan to refine this assumption through evaluations with students. The framework chooses the example with the highest '*Overall Utility*' node value and presents it to the student. A similar approach is described in (Murray & VanLehn, 2000) to select tutorial actions; here we extend this approach to the example-selection task.

Role of Example in the Student Model's Assessment

Once an example is selected, it helps the EA-Coach student model refine its assessment of the student's knowledge and APS tendencies, in conjunction with the student's interface actions (Muldner & Conati, 2005). For instance, if the student correctly generates a solution step that does not have a corresponding element in the example solution, the model can deduce that no copying took place. This increases the probability that the student knows the rule needed to generate the step. Similarly, suppose that a student assessed to have a low EBLC tendency and low physics knowledge views an example step (by uncovering it in the masking interface) that shares a trivial superficial difference with the problem (e.g., such as step 3 in Figure 15), and then generates a correct solution step. Given this scenario, the model predicts that transfer happened through transformational analogy and not EBLC. Suppose, on the other hand, that the same student generates a step after looking at an example that shares a non-trivial superficial difference with the target problem over this solution step (such as step 4 in Figure 15). In this case, the student model assesses that the student did self-explain through EBLC, because this is the only way to resolve the difference between problem and example and correctly generate the problem solution. This assessment in turn results in the model assigning a higher probability for both the physics rule corresponding to the problem step and the student's EBLC tendency.

EA-Coach: Summary

In this second portion of the paper, we have described the EA-Coach component of our adaptive framework for example-based learning. The EA-Coach aims to extend the SE-Coach tailored support for example studying to the usage of examples during problem solving (analogical problem solving, or APS), thus bridging the transition between example studying and full-fledged problem solving. Following the general philosophy of the ExBL framework, the EA-Coach targets the meta-cognitive skills that trigger effective APS (min-analogy and EBLC) and does so in a tailored manner. The crucial part of this tailoring is achieved through the individualized selection of examples that can trigger positive APS

behaviours in students who have a low tendency for them.

In the rest of the paper, we describe an initial evaluation of the ExBL framework, focusing on the pedagogical effectiveness of the SE-Coach.

EMPIRICAL EVALUATION OF THE SE-COACH

Although we have not had a chance to fully evaluate all the components of our ExBL framework, we have conducted an empirical study to test the effectiveness of the SE-Coach support for self-explanation of correctness and utility (Conati & VanLehn, 2000). Despite the fact that the evaluation targets only the SE-Coach, its results also provide initial support for decisions related to the EA-Coach design. In particular, not only do they show that complex scaffolding for self-explanation is beneficial at the initial stage of cognitive skill acquisition, but they also suggest that as student expertise increases, subtler forms of scaffolding may be more effective. This indirectly validates our choice of giving the EA-Coach less explicit ways to scaffold self-explanation as compared to the SE-Coach.

The SE-Coach study involved 56 college students, using an earlier version of the SE-Coach that covered two of the three types of self-explanations currently included in the system: step correctness (justify why a solution step is correct in terms of rules in the target domain) and step utility (explain what role a solution step plays in the high-level plan underlying an example solution). Thus this version did not include support for gap-filling self-explanation, i.e., for those explanations aimed at clarifying how a given solution fact derives from steps that are not shown in the example solution.

The ExBL framework does not provide any introductory physics instruction, because it is meant to complement regular classroom activities. Hence, to evaluate the SE-Coach adequately, subjects need to have enough knowledge to understand the topic of the examples, but not so much knowledge as to find the examples not worthy of attention. The closest we could get to enforce this constraint was to recruit college students taking introductory physics and who already had their first lecture on Newton's Second Law, but not a class test on the topic. The students came from 4 different colleges.

The one-session study comprised: 1) solving four pre-test problems on Newton's Second Law; 2) studying examples on Newton's Second Law with the system; 3) solving post-test problems equivalent but not identical to the pre-test ones. The study had two conditions. In the *experimental* (*SE*) condition, 29 students studied examples with the complete SE-Coach. In the *control* condition,

27 students studied examples using only the masking interface and the Plan Browser (the tool to support explanations for step utility)[5]. They could see the *self-explain* prompt appear when they uncovered a line, but had no access to the subsequent level of prompting for self-explanations on step correctness ("*this step is correct because*…") and step utility ("*the role of this plan in the solution step is*.."). They also had no access to the Rule Browser and Templates (the tools to support explanations for step correctness), nor feedback or coaching.

As we reported in (Conati & VanLehn, 2000), the analysis of the log files from the study shows that the SE-Coach's interface is easy to use and explicit tutorial interventions are quite successful at stimulating self-explanation. To evaluate the pedagogical effectiveness of the SE-Coach, we first compared how the students in the SE condition learned in comparison to the students in the control condition.

Table 1 shows summary statistics for pre-test scores, post-test scores and their difference (gain scores) for each condition. There were no significant differences between the pre-test scores of the two conditions, indicating that subjects had been successfully randomized to obtain two conditions with equivalent physics knowledge. The difference between gain scores of SE and control condition was also not statistically significant, as measured by an Analysis of Covariance (ANCOVA) with post-test scores as the dependent variable, the pre-test scores as the covariate and condition as the main factor.

TABLE 1
Test and gain scores for SE and Control conditions.

		N	Mean	SD
SE-condition		29		
	Pre-test		24.47	5
	Post-test		30.51	6.8
	Gain		6.04	4.49
Control condition		27		
	Pre-test		25.46	6.2
	Post-test		30.51	6.4
	Gain		5.05	4.35

[5]We let the control students access the Plan Browser because introductory physics courses usually do not address solution planning, therefore control students would have had too much of a disadvantage if they had not been able to see what a solution plan is through the Plan Browser.

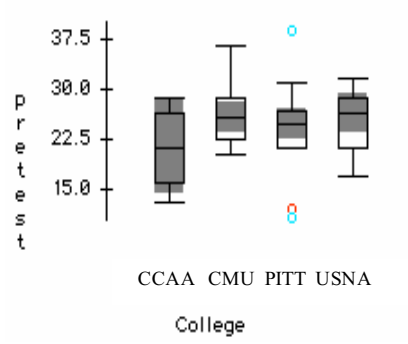
We then restricted the analysis to the subgroups of subjects coming from different colleges: Carnegie Mellon University (CMU, 14 students), Community College of Allegheny County (CCAC, 5 students), University of Pittsburgh (PITT, 20 students) and U.S. Naval Academy (USNA, 17 students). We found that the SE condition of CMU and CCAC students had gain scores substantially higher than the control condition. In contrast, in both the Pitt and USNA subgroups, students in the control condition had slightly better gain scores than students in the SE condition (see Table 2 for a summary of the mean test values for the four groups).

TABLE 2
Mean pre-test, post-test and gain scores for Control and SE group in each college.

Pre test (mean)	**CCAC**	**CMU**	**Pitt**	**USNA**
Control group	21.17	26.64	25.71	25.37
SE group	21.25	25.61	22.27	25.51

Post test (mean)	**CCAC**	**CMU**	**Pitt**	**USNA**
Control group	26.5	29.23	32.71	31.84

The commonality of behaviour between CMU and CCAC is quite surprising, because the two schools are supposed to be, respectively, the most and the least prestigious of the four colleges in the study, and this ranking is confirmed by the trends in pre-test scores (see Figure 17).



College	**Mean**
CCAC	21.2
CMU	26.13
Pitt	24.
USNA	25.45

FIGURE 17
Pre-test scores for students in the four colleges.

However, there is one characteristic that CMU and CCAC have in common and that distinguishes them from Pitt and USNA students. They start the semester at least a week later than Pitt and USNA. Therefore, although all the students participated in the experiment after they had their lectures on Newton's laws and before they took a class test on the topic, Pitt and USNA students were ahead in the course schedule and had likely spent more time on Newton's laws than CMU and CCAC students when they participated in the study. This could have generated differences in the background knowledge and/or in the way the two subgroups used the system, affecting how the students benefited from it.

To test this hypothesis, we continued our analysis by comparing the subgroup consisting of CMU and CCAC students (we'll label this *late-start* from now on) with the subgroup including Pitt and USNA students (*early-start*). An ANCOVA with post-test as the dependent variable, subgroup (*early-start* vs. *late-start*) and condition (SE vs. control) as the main factors, and pre-test as the covariate confirms that there is a statistically significant interaction of group with condition ($F(1,55) = 7.238$; $p < 0.01$), and indicates that prior physics knowledge *does not* account for the different behaviour of the two subgroups in the two conditions. Adding SAT scores (math and verbal) to the ANCOVA as covariate preserves the significance of the interaction, ($F(1,45) = 7.61$, $p < 0.01$), showing that general proficiency as measured by these tests also does not account for the different behaviour of the two groups.

Within the late-start group, the SE condition performed better than the control condition. In terms of gain scores, the SE condition has a mean of 7.5 vs. 3.2 of the control condition. An ANCOVA with post-test scores as dependent variable, condition as main factor and pre-test as covariate shows a marginally significant difference over post-test performance ($F(1,24) = 6.17$, $p = 0.021$) using the _ level of 0.0125 given by a Bonferroni adjustment for 4 post-hoc comparisons (two more of these comparisons will be discussed below). Within the early-start group the control condition performed slightly better than the SE condition (mean gain 5 vs. 6.7), but the performance difference between the two groups is not statistically significant, as shown by the corresponding ANCOVA for post-test scores over pre-test and condition ($F(1,30)= 1.6$, $p>0.2$).

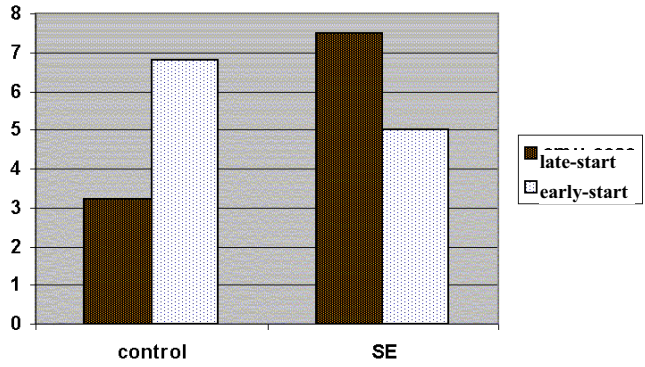


FIGURE 18
Gain scores for late-start and early-start students within the two conditions.

We then performed pairwise comparisons of the late-start and early-start groups within each of the instructional conditions. Let's start by analyzing performance. Figure 18 gives the pairwise performance comparison for the two groups within each condition in terms of gain scores. In the SE condition late-start students performed better than early-start students. However, the performance difference is not statistically significant as per an ANCOVA with post test scores as the dependent variable, group as the main factor, and pre-test as the covariate ($F(1,28) = 2.17$, $p > 0.1$). In the control condition, early-start students performed considerably better than late-start students, and the corresponding ANCOVA with group as main factor showed a marginally significant difference in post test performance when controlled for pre test ($F(1,26) = 5.58$, $p = 0.027$), as per the aforementioned Bonferroni adjustment. As a matter of fact, early-start students in the control condition performed almost as well as late-start students in the SE Condition.

One reason for these results could be that the early-start students in the control condition were actually self-explaining. The reader should remember that this condition did have a minimal form of scaffolding provided by the masking interface and associated *self-explain* button. It is possible that, because of the more advanced learning stage of the early-start students, this minimal scaffolding was sufficient to trigger self-explanation, making the early-start students in the control condition self-explain more than late-start students in the same condition did.

Verifying this hypothesis is not easy, because students in the control condition could not communicate their explanations to the control version

of the SE-Coach. The only measures that may indicate self-explanation in the log files include: (i) mean and standard deviation of multiple accesses to example lines; (ii) mean and standard deviation of the time spent on each example line; (iii) mean number of accesses and selections in the Plan Browser. To explore the potential influence of these measures on post-test performance in relation to learning stage, we ran a multiple regression analysis of post test scores on these measures within each of the two control sub-groups, early-start and late-start.

TABLE 3
Regression of post test on mean and stardard deviation of line accesses for Early Start group.

Dependent variable is: **post**
cases selected according to **early-start**
27 total cases of which 13 are missing
R squared = 72.6% R squared (adjusted) = 64.4%
s = 4.306 with 14 - 4 = 10 degrees of freedom

Source	Sum of Squares	df	Mean Square	F-ratio
Regression	491.163	3	163.721	8.83
Residual	185.421	10	18.5421	

Variable	Coefficient	s.e. of Coeff	t-ratio	prob
Constant	11.7794	5.138	2.29	0.0448
pre	0.719348	0.1772	4.06	0.0023
mean-acces	2.96072	1.54	1.92	0.0834
sd-acc	-3.64745	1.698	-2.15	0.0573

For the early-start control group, the model we obtained (see Table 3) is able to account for 64.4% of the variance in post-test performance (i.e., adjusted R2 = 64.4), with the following predictor variables yielding a significant or marginally significant correlation with the dependent variable: the mean number of line accesses (variable *mean-acces* in Table 3, correlation coefficient 2.96, t(10) = 1.92, p = 0.083), standard deviation of multiple line accesses (*sd-acc* in Table 3, correlation coefficient -3.65, t(10) = -2.15, p = 0.0573) and pre-test scores (pre in Table 3, correlation coefficient -0.72, t(10) = 4.06, p = 0.0448). None of the aforementioned variables that may indicate self-explanation yield a significant or marginally significant correlation in the analogous multiple regression for the late-start

control group (13 students). Although these results are not strong enough to conclude that early-start control students self-explained more than late-start control students, the hypothesis that early-start students self-explained more in the control condition is consistent with the fact that these students had started studying Newton's Laws earlier and had probably gained more knowledge on the topic. It is possible that, although this knowledge was not strong enough to make early-start control students better problem solvers, it was sufficient to enable them to generate effective self-explanations under the minimal scaffolding provided by the masking interface.

One might wonder why we didn't see a similar effect in the SE condition, i.e., why we did not see more learning in early-start students in this condition (17 students), given that they not only had potentially more familiarity with the topic than late-start students in the SE condition (12 students), but they also had more help in generating useful self-explanations than their control counterpart. To try and answer this question, we compared early and late-start students in the SE Condition with respect to time on task and statistics on usage of the SE tools to see if we could find any difference that could explain the comparatively limited benefits the early-start students got from the interaction with the SE-Coach. We did not find any difference in terms of time on task or frequency of SE-tools usage, but we did find a statistically significant difference ($p = 0.011$, 2-tailed t-test with Bonferroni adjustment) in the average number of attempts tried before giving up on a Template explanation, with early-start students showing significantly fewer attempts. This difference suggests that early-start students might have had a lower level of motivation to learn from the SE-Coach, because they started learning Newton's second laws earlier than late-start students and thus perceived studying examples on this topic to be less useful, consistent with the findings described in (Nguyen-Xuan, Bastide et al. 1999). Browser selections and template filling are recall tasks not as constructive as generating explanations verbally, unless students actively reflect on the result of their actions. It is plausible that, because early-start students were less motivated to study the SE-Coach examples, they may have reasoned less on what they were doing with the SE-Coach interface. Thus, they did not learn as much as they could have from the SE tools, although they did not access them less than late-start SE students did.

SE-Coach Evaluation: Summary

In this section, we have described the results of a controlled study designed to test the pedagogical effectiveness of the SE-Coach support for

self-explanation for step correctness and utility. These results show that the SE-Coach's multiple levels of scaffolding for self-explanation improve students' problem solving when students are in the early stage of cognitive skill acquisition. However, the results also suggest that the full-fledged system is not as effective for students who are at a more advanced learning stage. As a matter of fact, there is indirect evidence that the milder form of scaffolding provided by the SE-Coach masking interface and untailored reminders may be sufficient to trigger effective self-explanation for these more advanced students. These results provide initial support for our design decision of confining strong self-explanation scaffolding to the early learning stage of example studying supervised by the SE-Coach, and to fade this scaffolding when students are advancing to the analogical problem solving targeted by the EA-Coach. We will of course need to run more user studies to understand how effective the EA-Coach design is, but the study we have presented provides initial evidence that we are on the right track.

RELATED WORK

The ExBL framework is to the best of our knowledge the only framework that integrates tailored support for both example studying prior to problem solving and APS, and that does so by targeting the meta-cognitive skills involved in these instructional activities. However, there are systems that target one of the two ExBL phases in isolation (Aleven & Ashley, 1997; Kashihara, Hirashima et al., 1995; Nogry, Jean-Daubias et al., 2004; Renkl, Atkinson et al., 2002; Weber, 1996), or that target the ExBL meta-cognitive skills in a different instructional context (Aleven & Koedinger, 2002; Mitrovic, 2003).

Renkl, Atkinson et al. (2002) describe a tutor that aims to coach self-explanation during example studying like the SE-Coach. However, this tutor focuses exclusively on gap-filling self-explanation. The gaps are inserted following a fixed progression, instead of being tailored to a student's knowledge. The framework presented by Kashihara, Hirashima et al. (1995) also tries to foster gap-filling self-explanation, in the context of studying instructional text on word meaning. In this system, like in the SE-Coach, gap selection is tailored to reduce a student's cognitive load. However, to generate the corresponding instructional material the system relies on a simple user model, while the SE-Coach uses a sophisticated model of the student's inferential capabilities.

The natural language generation (NLG) field has extensively studied the process of producing text tailored to a model of the user's inferential capabilities (e.g., Horacek, 1997; Korb, McConachy et al., 1997; Young, 1999). However, the application of NLG techniques in ITS tends to be limited either to managing and structuring the tutorial dialog (e.g., Freedman, 2000; Moore, 1996) or to making the selected content more fluent (Di Eugenio, Glass et al., 2002; Haller & Di Eugenio, 2003), rather than on tailoring the presentation of the instructional material to a detailed model of the student's inferential capabilities, as we do in the SE-Coach.

There are two ITS that support self-explanation during pure problem solving. The Geometry Explanation Tutor (Aleven & Koedinger, 2002) supports self-explanation during geometry theorem proving, while Normit-SE (Mitrovic, 2003) targets data normalization in the database domain. Unlike the ExBL framework, neither of these tutors tailors its support to specific shortcomings in student self-explanation behaviour. The Geometry Explanation tutor prompts students to self-explain every problem-solving step, while Normit-SE prompts students to self-explain every *new* or *incorrect* problem-solving step.

Both tutors target only self-explanations for step correctness and provide interface tools similar to the SE-Coach's to scaffold it. Normit-SE provides menu-based tools, while the Geometry tutor asks students to select an explanation from a list of geometry theorems. The Geometry tutor has recently been extended to allow students to type free-form self-explanations (Aleven, Ogan et al., 2004). This is an important addition given that all cognitive science studies reporting on the effectiveness on this meta-cognitive skill targeted free-form verbal self-explanation, closer in nature to free-form typed explanations than menu-based ones. However, a study comparing the two versions of the Geometry tutor revealed no overall difference in learning between self-explaining by selecting from a menu vs. typing free-form explanations.

There are a number of systems that, like the EA-Coach, select examples for students during problem solving. However, none of these systems take into account the impact of various differences between the problem and example on a student's knowledge and/or meta-cognitive skills. ELM-PE (Weber, 1996) helps students solve LISP programming problems by choosing examples that are as similar as possible to the problem the student is working on. CATO (Aleven & Ashley, 1997) helps students build legal arguments by dynamically generating examples of how an expert would argue about a particular case. In (Nogry, Jean-Daubias et al., 2004) the

authors claim that their AMBRE system supports the solution of algebra problems by choosing structurally-appropriate examples. However, it is not clear from the paper what "structurally appropriate" means. SPIEL (Burke & Kass, 1995) supports learning of social skills by retrieving and presenting examples, referred to as stories, intended to illustrate first-person narratives about actual experiences. To recognize when a story is relevant, SPIEL relies on a set of rules representing story-telling strategies (for instance, a story could be relevant because it explains alternative perspectives to the student).

Recently, Aleven, McLaren et al., (2004) expanded the Geometry tutor with a Help-seeking Tutor to provide tailored support for the meta-cognitive skill of effective help-seeking. One aspect of this skill involves not abusing available help by asking for detailed hints too lightly and too frequently. The Help-Seeking tutor relies on a production rule model that captures both correct help-seeking behavior and incorrect help-seeking behavior that negatively correlates with learning, in order to provide tailored support to effective help-seeking. Referring to examples during problem-solving activities is also a form of help seeking. The EA-Coach discourages max-analogy, which is also a form of 'help abuse', but does so by selecting appropriate examples.

In this paper, we have argued that prompts to encourage students to reason effectively should be tailored, because tailored prompts have the potential to both trigger better learning and be less intrusive than untailored ones. The only work that has compared tailored and non-tailored prompts is Hausmann and Chi (2002). This work found no learning difference between receiving tailored vs. untailored prompts in a Wizard of Oz experiment involving students studying instructional text in biology. However, this work does not provide conclusive evidence on this issue because it also showed that students learned well without any prompts to self-explain, possibly because the instructional material was intended for a much younger population (i.e., college students were studying grade 8 text). Furthermore, the study did not include any results on student preferences for tailored vs. untailored interventions, even though student satisfaction can often have a strong impact on system acceptance and consequent effectiveness.

SUMMARY & FUTURE WORK

In this paper we have presented ExBL, an ITS framework designed to help students effectively acquire problem-solving skills from pedagogical

activities involving worked-out example solutions. The rationale underlying this work is that students tend to spontaneously rely on examples when learning new skills through traditional pedagogical material (i.e. textbooks), and they do so by (i) studying examples prior to problem solving in the very early stages of skill acquisition, (ii) using examples as aids during problem solving (analogical problem solving, or APS) as their expertise increases. Thus, it seems highly valuable to provide computer-based support for these activities, to complement the extensive availability of computer-based support for pure problem solving.

However, like many other pedagogical activities, there is great variance in how well different students learn by using examples. Cognitive science studies have shown that this variance strongly depends on individual differences in the ability to apply meta-cognitive skills that positively affect example-based learning.

ExBL is designed to take into account these individual differences and provide tailored support for the application of self-explanation and min-analogy, two of the meta-cognitive skills related to example-based learning that have been most extensively studied in cognitive science. The subject matter targeted by the ExBL framework is Newtonian physics, one of the domains in which there is a well-recognized need for innovative educational tools, due to the extreme difficulty students often encounter in bridging theory and practice, either because they cannot make the leap from theory acquisition to effective problem solving, or because they may learn to solve problems effectively without grasping the underlying theory.

In this paper, we have described the two coaching components of ExBL, that separately address the two phases of example-based learning: (i) the SE-Coach, that scaffolds self-explanation during example studying; (ii) the EA-Coach, that supports both self-explanation and min-analogy during problem solving.

There are three main contributions in our work:

1. It describes an ITS with the unique characteristic that it supports the use of examples both before and during problem solving, and that it does so by adapting not only to individual students' cognitive traits, but also to their specific meta-cognitive needs.
2. It includes one of the first attempts to apply natural language generation techniques for the automatic generation of example solutions at varying degrees of detail, a functionality used by the SE-Coach to support self-explanation aimed at filling gaps in sparse example solutions.

3. It extends research on providing computer-based support for meta-cognitive skill acquisition both by making the first attempt to address the min-analogy meta-cognitive skill, specifically relevant to improving performance during APS and by exploring the impact of differences between a problem and an example to provide tailored stimuli for effective APS.

Several parts of the ExBL architecture rely on the architecture of Andes, a well established ITS for physics problem solving. This has two main advantages. The first is that some of the ExBL modules have been empirically validated through the studies performed on Andes. These modules include the ExBL physics knowledge base, as well as the components used by the problem-solving part of the EA-Coach (i.e., the Andes interface, and its mechanisms for feedback and knowledge assessment). The second advantage is that the ExBL framework is ready to be integrated into a larger environment that covers all phases of cognitive skill acquisition, from pure example studying with the SE-Coach, to APS with the EA-Coach, to pure problem solving with Andes.

In the paper, we have discussed how both ExBL coaches rely on the philosophy of providing multiple levels of tailored scaffolding for their target cognitive skills, although the intensity of the scaffolding is faded from the SE-Coach to the EA-Coach. The SE-Coach includes fairly explicit scaffolding for building useful self-explanations in the form of menu-based tools and direct tailored tutorial interventions. Conversely, the EA-Coach relies on subtler scaffolding means, including a highly innovative example selection mechanism. This mechanism is tailored to student needs and chooses examples that can trigger the appropriate meta-cognitive behaviours because of their level of similarity (or lack thereof) with the target problem.

We have also presented results from a study on the effectiveness of the SE-Coach component of the ExBL framework. The study targeted an earlier version of the system that supported only two types of self-explanation: (i) step correctness (justify why a solution step is correct in terms of rules in the target domain); (ii) step utility (explain what role a solution step plays in the high-level plan underlying an example solution). The study showed that the SE-Coach tailored scaffolding improved students' problem-solving performance if students accessed the system at an early learning stage. The study also provided preliminary evidence that simpler, untailored scaffolding may be sufficient to support self-explanation for more expert

students. Because these are the students that are more likely to be ready to move to the APS phase of cognitive skill acquisition, these preliminary results are consistent with our choice of a milder form of scaffolding in the EA-coach.

Naturally, one of the immediate next steps in this research is to directly evaluate the EA-Coach, as well as the newer version of the SE-Coach complete with scaffolding for self-explanation aimed at filling gaps in sparse example solutions.

In addition to testing the overall pedagogical effectiveness of these components, we intend to design the studies so that they will provide insights on the following issues.

- Is the subtler form of scaffolding provided by the EA-Coach sufficient or do we need more direct interventions, in the SE-Coach style?
- The EA-Coach currently addresses only one type of self-explanation, *Explanation-Based Learning for Correctness* (EBLC). EBLC allows students to learn new rules by applying common-sense and overly-general knowledge to explain unknown steps in the example solution. We chose to focus the EA-Coach on EBLC because this is the type of self-explanation on which we have cognitive science findings in relation to its role in effective APS. However, it is possible that the other kinds of self-explanations (for correctness, utility and gap filling), currently addressed only by the SE-Coach may still play a role during APS. We would like the EA-Coach study to give us insights on what this role might be and if and how it will need support during interaction with the EA-Coach.
- The EA-Coach relies on supporting learning during problem solving by encouraging min-analogy and EBLC through example selection. A benefit of this strategy is that it affords students the opportunity to practice the application of meta-cognitive skills, which could help them become more effective learners. A related drawback, however, is that it places much of the responsibility during the learning process on the student. This may be problematic for some students, especially for the ones who have very low meta-cognitive tendencies. An alternative approach for supporting problem-solving activities, taken by Andes, is to provide direct problem solving hints *instead* of examples (i.e., to support pure problem solving). It would be interesting to compare the two approaches to gain a better understanding of how each coach supports learning, and if and how student characteristics and expertise (e.g., stage in cognitive skill acquisition) impact the learning outcomes.

- A more general question related to the above point is whether there are optimal trajectories that students can follow to go from pure example studying (SE-Coach), to APS (EA-Coach), to pure problem solving (Andes). If so, how can they be defined and supported in an intelligent learning environment that includes all three coaches?

In the longer term, we are planning to investigate how to apply the natural language generation techniques currently used by SE-Coach to the EA-Coach to generate examples with adequate solution steps. Currently, the textual description of each available example solution in the EA-Coach must be entered by a human being. Natural-language generation could be used to build this textual description automatically from the solution graph, as it is currently done in the SE-Coach.

ACKNOWLEDGEMENTS

This research was supported by a graduate fellowship from the University of British Columbia, by the Walter C. Sumner Memorial Foundation, and the B.C. Advanced Systems Institute. The authors would like to thank Andrea Bunt and the anonymous reviewers for their helpful comments on earlier versions of the manuscript.

REFERENCES

Aleven, V. and K. D. Ashley (1997). *Teaching case-based argumentation through a model and examples: Empirical evalution of an intelligent learning environment*. Artificial Intelligence in Education, Kobe, Japan.

Aleven, V. and K. Koedinger (2002). "An effective meta-cognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor." *Cognitive Science* **26**(2): 147-179.

Aleven, V. and K. R. Koedinger (2002). "An Effective Meta-Cognitive Strategy: Learning by Doing and by Explaining with a Computer-Based Cognitive Tutor." *Cognitive Science* **26**(2): 147-179.

Aleven, V., B. McLaren, et al. (2004). *Toward Tutoring Help Seeking: Applying Cognitive Modeling to Meta-Cognitive Skills*. Intelligent Tutoring Systems, Brasil. 227-239

Aleven, V., A. Ogan, et al. (2004). *Evaluating the Effectiveness of a Tutorial Dialogue System for Self-Explanation*. Intelligent Tutoring Systems: 7th International Conference, Maceió, Alagoas, Brazil. 443 - 454

Anderson, J., J. Fincham, et al. (1997). "The Role of Examples and Rules in the Acquisition of a Cognitive Skill." *Journal of Experimental Psychology: Learning, Memory and Cognition* **23**(4): 932-945.

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

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

Bielaczyc, K., P. Pirolli, et al. (1995). "Training in self-explanation and self-regulation strategies: Investigating the effects of knowledge acquisition activities on problem-solving." *Cognition and Instruction* **13**(2): 221-252.

Burke, R. and A. Kass (1995). "Supporting Learning through Active Retrieval of Video Stories." *Expert Systems with Applications* **9**.

Chi, M. T. H. (2000). Self-Explaining: The dual process of generating inferences and repairing mental models. *Advances in Instructional Psychology*. R. Glaser, Lawrence Erlbaum Associates**:** 161-238.

Chi, M. T. H., M. Bassok, et al. (1989). "Self-explanations: How students study and use examples in learning to solve problems." *Cognitive Science* **15**: 145-182.

Chi, M. T. H. and K. VanLehn (1991). "The content of physics self-explanations." *The Journal of the Learning Sciences* **1**: 69-105.

Clement, R. (1996). *Making Hard Decisions*, Duxberry Press.

Conati, C. and G. Carenini (2001). *Generating Tailored Examples to Support Learning via Self-explanation*. IJCAI '01, the Seventeenth International Joint Conference on Artificial Intelligence, Seattle, WA

Conati, C., A. Gertner, et al. (2002). "Using Bayesian Networks to Manage Uncertainty in Student Modeling." *Journal of User Modeling and User-Adapted Interaction* **12**(4): 371-417.

Conati, C. and K. VanLehn (1999). *Teaching meta-cognitive skills: implementation and evaluation of an tutoring system to guide self-explanation while learning from examples.* AIED'99, 9th World Conference of Artificial Intelligence and Education, Le Mans, France.

Conati, C. and K. VanLehn (2000). "Toward Computer-based Support of Meta-cognitive Skills: A Computational Framework to Coach Self-Explanation." *International Journal of Artificial Intelligence in Education* **11**.

Conati, C. and K. VanLehn (2001). *Providing adaptive support to the understanding of instructional material*. IUI 2001, International Conference on Intelligent User Interfaces, Santa Fe, NM, USA.

Di Eugenio, B., M. Glass, et al. (2002). The DIAG experiments: Natural Language Generation for Intelligent Tutoring Systems. Harriman, NY.

Freedman, R. (2000). *Plan-based dialog management in a physics tutor*. Sixth Applied Natural Language Processing Conference, Seattle, WA

Haller, S. and B. Di Eugenio (2003). *Minimal Text Structuring to Improve the Generation of Feedback in Intelligent Tutoring Systems*. FLAIRS 2003, the 16th International Florida AI Research Symposium.

Halloun, I. and D. Hestenes (1985). "The Initial Knowledge State of College Physics Students." *American Journal of Physics* **53**: 1043-1055.

Hausmann, R. and M. Chi (2002). "Can a Computer Interface Support Self-explaining." Cognitive Technology **7**: 4-14.

Horacek, H. (1997). *A Model for Adapting Explanations to the User's Likely Inferences*. UMUAI. 1-55.

Kalyuga, S., P. Chandler, et al. (2001). "When Problem Solving is Superior to Studying Worked Examples." *Journal of Educational Psychology* **93**(3): 421-428.

Kashihara, A., T. Hirashima, et al. (1995). "A Cognitive Load Application in Tutoring." *User Modeling and User-Adapted Interaction* **4**: 279-303.

Korb, K., McConachy, et al. (1997). *A cognitive model of argumentation*. Nineteenth Annual Conference of the Cognitive Science Society, Mahwah NJ USA, Lawrence Erlbaum Associates.400-405.

Mitrovic, A. (2003). *Supporting Self-Explanation in a Data Normalization Tutor*. In Supplementary proceedings AIED.

Mitrovic, T. (2003). Supporting Self-Explanation in a Data Normalization Tutor. Supplementary Proceedings of AIED2003. 565-577.

Moore, J. D. (1996). "Discourse generation for instructional applications: Making computer-based tutors more like humans." *Journal of Artificial Intelligence in Education* **7**(2): 181-124.

Muldner, K. and C. Conati (2005). *Using Similarity to Infer Meta-Cognitive Behaviors During Analogical Problem Solving*. To appear in UM2005 User Modeling: Proceedings of the Tenth International Conference, Edinburgh, UK.

Murray, C. and K. VanLehn (2000). *DT Tutor: A decision-theoretic dynamic approach for optimal selection of tutorial actions*. ITS 2000, Montreal, Canada.

Nguyen-Xuan, A., A. Bastide, et al. (1999). *Learning to solve polynomial factorization problems: by solving problems and by studying examples of problem solving, with an intelligent learning environment. AIED '99, 9th World Conference of Artificial Intelligence and Education*, Le Mans, France.

Nogry, S., S. Jean-Daubias, et al. (2004). *ITS Evaluation in Classroom: The Case of Ambre-AWP*. Intelligent Tutoring Systems 2004:, Brasil. 511-520.

Novick, L. R. (1988). "Analogical transfer, problem similarity and expertise." *Journal of Experimental Psychology: Learning, Memory and Cognition* **14**: 510-520.

Novick, L. R. and K. J. Holyoak (1991). "Mathematical problem solving by analogy." *Journal of Experimental Psychology: Learning, Memory and Cognition* **17**(3): 398-415.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo, CA, Morgan-Kaufmann.

Pirolli, P. and M. Recker (1994). "Learning strategies and transfer in the domain of programming." *Cognition and Instruction* **12**(3): 235-275.

Reed, S. K. and C. A. Bolstad (1991). "Use of examples and procedures in problem solving." *Journal of Experimental Psychology: Learning, Memory and Cognition* **17**(4): 753-766.

Reed, S. K., A. Dempster, et al. (1985). "Usefulness of analogous solutions for solving algebra word problems." *Journal of Experimental Psychology: Learning, Memory and Cognition* **11**: 106-125.

Reif, F. (1995). *Understanding basic mechanics*, Wiley & Sons.

Renkl, A. (1997). "Learning from worked-examples: A study on individual differences." *Cognitive Science* **21**(1): 1-30.

Renkl, A., R. Atkinson, et al. (2002). "From Example Study to Problem-Solving: Smooth Transitions Help Learning." *The Journal of Experimental Education* **70**(4): 293-315.

Renkl, A., R. Stark, et al. (1998). "Learning from worked-out examples: the effects of example variability and elicited self-explanation." *Contemporary educational psychology* **23**: 90-108.

Ryan, R. (1996). Self-explanation and adaptation. *Psychology*. Pittsburgh, University of Pittsburgh.

Schulze, K. G., R. N. Shelby, et al. (2000). "Andes: An intelligent tutor for classical physics." *The Journal of Electronic Publishing* **6**(1).

Sweller, J. (1988). "Cognitive load during problem solving: Effects on learning." *Cognitive Science* **12**: 257-285.

VanLehn, K. (1996). Cognitive skill acquisition. *Annual Review of Psychology, Vol. 47*. J. Spence, J. Darly and D. J. Foss. Palo Alto, CA, Annual Reviews**:** 513-539.

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

VanLehn, K. (1999). "Rule-Learning Events in the Acquisition of a Complex Skill: An Evaluation of Cascade." *The Journal of the Learning Sciences* **8**(1): 71-125.

VanLehn, K. and R. M. Jones (1993). Better learners use analogical problem solving sparingly. *Machine Learning: Proceedings of the Tenth Annual Conference*. P. E. Utgoff. San Mateo, CA, Morgan Kaufmann**:** 338-345.

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

Young, M. (1999). "Using Grice's maxim of Quantity to select the content of plan descriptions." *Artificial Intelligence Journal* **115**(2): 215-256.