Student Modeling: Supporting Personalized Instruction, from Problem Solving to Exploratory Open-Ended Activities

Cristina Conati, Samad Kardan

■ *The field of intelligent tutoring systems (ITSs) has successfully delivered techniques and applications to provide personalized coaching and feedback for problem solving in a variety of domains. The core of this personalized instruction is a student model: the ITS component in charge of assessing student traits and states relevant to tailor the tutorial interaction to specific student needs during problem solving. There are however, other educational activities that can help learners acquire the target skills and abilities at different stages of learning including, among others, exploring interactive simulations and playing educational games. This article describes research on creating student models that support personalization for these novel types of interactions, their unique challenges, and how AI and machine learning can help.*

One of the main challenges in devising agents that can act intelligently is to endow them with the ability to understand the behaviors of other agents (human or artificial) in their environment. Intelligent tutoring systems (ITSs) are the ultimate example of this challenge: their goal is to provide instruction personalized to the specific needs of each learner, as good human tutors do. But understanding these needs can be extremely hard, because it entails modeling and capturing that complex ensemble of processes and states that constitutes human learning. These processes and states range from purely behavioral (for example, what actions a student takes to solve a problem), to cognitive (for example, domain knowledge, goals, cognitive load), to metacognitive (for example, domain-independent reasoning skills), to affective (for example, frustration, boredom, motivation). The more a tutor (human or artificial) understands about its learners at these different levels, the more personalized to the learners' needs its instruction can be.

Cognitive, metacognitive, and affective states, however, can be hard to assess accurately and unobtrusively from naturally occurring interaction events. Learner assessment is nontrivial even in its most basic incarnation, namely evaluating a learner's understanding of a set of domain-dependent skills from ad hoc test items (for example, Desmarais [2011]). The assessment challenges increase with the complexity of the learner's traits to be captured, because how a student behaves during an instructional activity generally provides partial and ambiguous information on the student's underlying states, and the gap between what can be observed and what a learner actually thinks and feels increases as these states go from cognitive to metacognitive and affective. In ITSs, the research field concerned with addressing these challenges is known as student modeling, and a student model is the ITS component in charge of assessing student traits and states relevant to tailor the tutorial interaction to specific student needs.

 ISSN 0738-4602

Student modeling research has made substantial progress in providing reliable learner assessment during problem solving or question-answering activities in a variety of domains (for example, programming, physics, algebra, geometry, and introductory computer science). Educational technology however, continues to produce novel environments often consisting of activities not as structured and well understood as problem solving, making the student modeling problem increasingly challenging. This article describes our research on creating student models that support personalization for these novel types of interactions, their unique challenges, and how we address them by relying on AI and machine-learning techniques.

Student Modeling for Problem Solving and Beyond

The field of ITS has made impressive progress in delivering e-learning environments that provide personalized support for problem solving in a variety of domains. In ITSs for problem solving, student modeling is generally used to provide coaching and feedback as students work on generating solutions to given problems. The student model includes a representation of one or more acceptable solutions to each available problem. This representation is used to evaluate the student's solution and provide tutorial interventions in case it deviates from the tutor's known solutions (for example, Conati, Gertner, and VanLehn 2002; Koedinger et al. 1997; Mitrovic 2012). The type of assessment needed from the student model depends, among other things, upon the type of interventions to be provided (VanLehn 2006).

At the behavioral level, solution assessment; that is, assessing the quality of the student's solution, is needed to provide feedback that helps the student improve the solution when necessary, as well as to support knowledge assessment (described later). There has also been extensive work on student models that detect instances of gaming the system. These are behaviors indicating that the student is trying to get the problem solution from the tutor (for instance by repeatedly asking for help) without trying to solve the problem on their own (Baker et al. 2008), thus the tutor may need to take actions for discouraging them (for example, Baker et al. 2006).

At the cognitive level, knowledge assessment, that is, evaluating the student's knowledge of relevant concepts and skills at specific points of the interaction (also known as knowledge tracing [Corbett and Anderson 1994]), is useful both for selecting which exercise or activity the student should do next (for example, Koedinger et al. [1997]), as well as for deciding how to provide feedback on the current student's solution (for example, Albacete and VanLehn [2000]). Plan/goal recognition, that is, understanding which solution a student is trying to follow or which part of a specific solution the student is working on, is necessary for providing help while the student is building her solution if the ITS accepts a variety of solutions to a problem or does not enforce a particular order for generating solutions (for example, Conati, Gertner, and VanLehn [2002]).

At the metacognitive level, there has been research on student modeling to assess both a student's metacognitive skills involved in effective analogical problem solving (Muldner and Conati 2010), as well as a student's ability to effectively use an ITS's help functionalities (Aleven et al. 2006). Preliminary results have shown that this information can be used by an ITS to provide interventions fostering the adequate metacognitive processes for students who do not spontaneously employ them (Muldner and Conati 2007, Roll et al. 2007).

At the affective level, researchers have looked at devising student models that can assess a student's motivation during problem solving with an ITS (for example, Qu and Johnson [2005]), as well as a variety of different emotions, including frustration, confusion, boredom (for example, Arroyo et al. [2009]; D'Mello and Graesser [2010]; Muldner, Burleson, and VanLehn [2010]). The goal is to allow an ITS to include student affect as a factor in deciding how to conduct coached problem solving (for example, D'Mello et al. [2010]), given extensive evidence in education research showing that affective factors play an important role in learning.

While providing support during problem solving is an important form of instruction, other educational activities can foster understanding at different stages of the learning process or for learners with different preferences and abilities. These activities include, among others, learning from examples, exploring interactive simulations, playing educational games, and learning with a group of peers. As is the case for problem solving, providing individualized support for these activities can be highly beneficial. However, providing this support through an ITS poses unique challenges because it requires modeling student behaviors, mental processes, and states that may not be as well defined and understood as those involved in problem solving.

For instance, because of the novelty of these educational activities, it can be difficult to judge a priori which ensembles of user-interaction behaviors are conducive to learning and which ones indicate a suboptimal interaction with the system that warrants help from the tutor. Thus, when moving beyond problem solving, even student modeling at the behavioral level becomes increasingly more ill defined and challenging. Traditional approaches based on providing the student model with a set of solutions or behaviors against which to compare the learner's performance with the system are unfeasible when there is limited knowledge of what these solutions and behaviors may be.

In this article, we describe research to provide student modeling at the behavioral level in light of these challenges. We present a user-modeling framework that can be leveraged to analyze logs of learners' interactions with a novel application, and discover classes of user types as well as their identifying behaviors and how these behaviors relate to learning. This information is then used to create a student model that can automatically classify the behaviors of new learners and whether they are associated with suboptimal task performance that may require the tutor intervention. In the

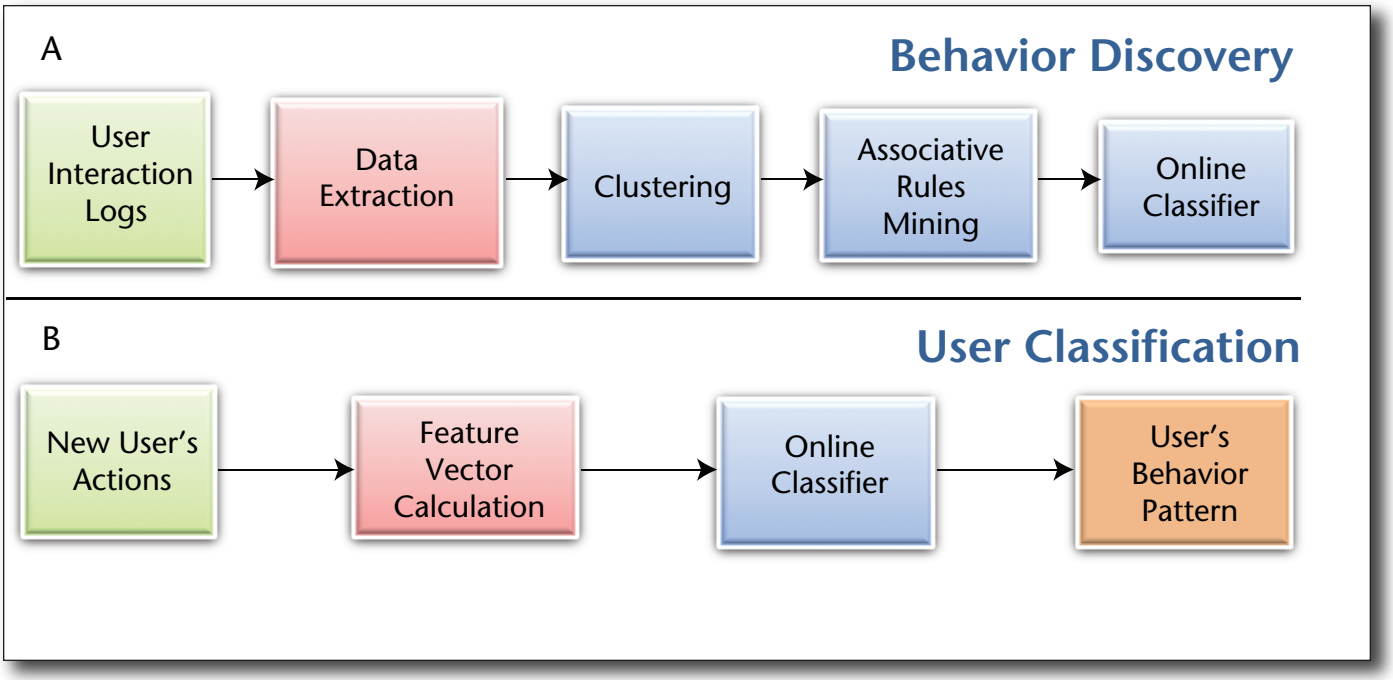

Figure 1. User-Modeling Framework.

(A) Behavior discovery. (B) User classification.

rest of the article, we first describe the user-modeling framework. Next, we describe the interactive simulation that we used as a test bed to evaluate the framework's effectiveness, followed by the results of this evaluation. After discussing the implications of this work, and avenues for future research, we end with presenting related work and conclusions.

User-Modeling Framework

Our user-modeling approach consists of two phases: behavior discovery (figure 1a) and user classification (figure 1b). In behavior discovery, raw unlabeled data from interaction logs is preprocessed into feature vectors representing individual users in terms of their interface actions, where features consist of statistical measures that summarize the user's actions in the interfaces (for example, action frequencies; time interval between actions).[1] These vectors are the input to a clustering algorithm (that is, k-means with a modified initialization step; see Kardan and Conati [2011]) that groups them according to their similarities. The resulting clusters represent users who interact similarly with the interface. These clusters are then analyzed to determine which interaction behaviors are effective or ineffective for learning. The analysis consists of first identifying how the different clusters relate to learning outcomes (by relying either on formal test results if available, or on the judgment of a human expert) and then isolating in each cluster those behaviors that are responsible for the learning effects. Behavior identification is done by performing association rule mining (Zhang and Zhang 2002) on each cluster. This process extracts the common behavior patterns in terms of class association rules (CAR) in the form of $\mathbf{X} \rightarrow c$, where $\mathbf{X}$ is a set of feature-value pairs and c is the predicted class label (that is, the cluster) for the data points where $\mathbf{X}$ applies. We use the Hotspot algorithm (Hall et al. 2009) for association rule mining, modified to look for the optimal settings for each of the three Hotspot parameters that define which of the many association rules generated are most representative of the cluster (see Kardan and Conati [2011] for details). Essentially, the goal of this process is to find a few rules that characterize as many elements in the cluster as possible and provide an easily understandable explanation of users' behaviors for each cluster.

Understanding the effectiveness of a user's interaction behaviors is useful in itself for revealing to designers of novel educational environments how to improve them (for example, García et al. [2009b]). However, we also want to use these behaviors to guide automated adaptive support during the interaction. Thus, the CARs extracted in the behavior discovery phase are used to build a classifier student model in the user-classification phase (figure 1b). The use of association rules to construct a classifier is called associative classification mining or associative classification (Thabtah 2007). As new users interact with the system, they are classified in real time into one of the clusters generated by the behavior discovery phase, based on a measure that summarizes how well a user's behaviors match the CARs for each cluster.

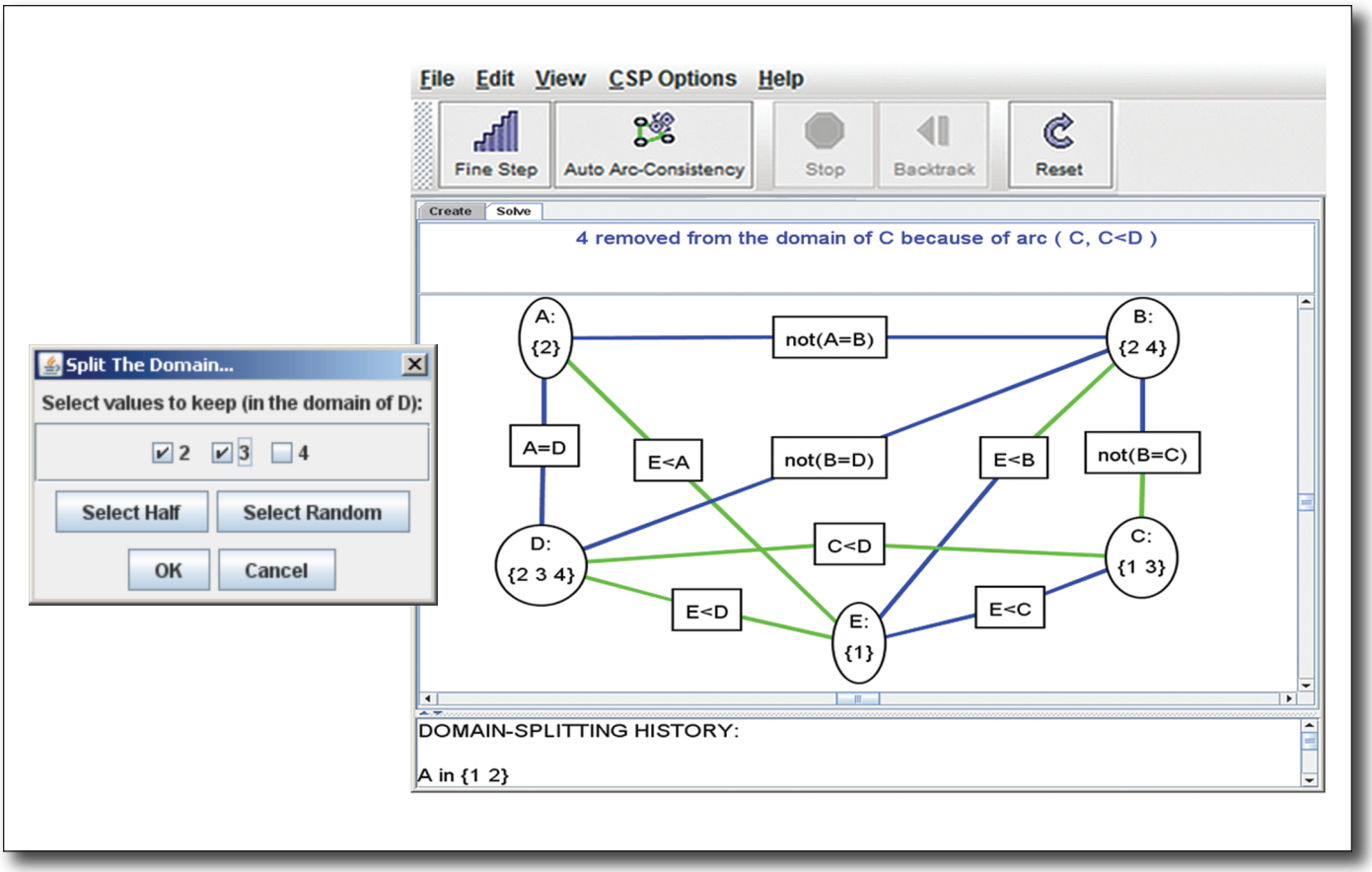

Figure 2. The CSP Applet with an Example CSP Problem.

The user-modeling framework is implemented as a toolset of modules that automate the process of going from interaction logs to generating the rule-based classifier (CAR classifier from now on). The modules are implemented in Python, with some external calls to Weka (Hall et al. 2009) through a command-line interface.[2] The framework includes a classifier evaluation module that uses cross-validation to evaluate the CAR classifier on available data sets, as follows. For each fold of the cross-validation, the data for each user in the test set is fed into the classifier trained on the training set by incrementally updating the feature vector representing the interaction behaviors of this user. Predictions are then made for the incoming vector as described earlier. The accuracy of the classifier is computed by checking (after each action in the user's log) whether each test user is correctly classified into its original cluster.

The rremainder of this article describes the results we obtained by applying the framework to build a user model for an interactive simulation designed to demonstrate the workings of an algorithm for constraint-satisfaction problems (CSPs), known as the CSP applet.

The AISpace CSP Applet

The CSP applet is one of a collection of interactive tools for learning artificial intelligence algorithms, called AIspace[3] (Amershi et al. 2008). Algorithm dynamics are demonstrated through interactive visualizations on graphs by the use of color and highlighting, and graphical state changes are reinforced through textual messages.

A CSP consists of a set of variables, variable domains, and a set of constraints on legal variable-value assignments (Rossi, Van Beek, and Walsh 2006). Solving a CSP requires finding an assignment that satisfies all constraints. The CSP applet illustrates the arc consistency 3 (AC-3) algorithm for solving CSPs represented as networks of variable nodes and constraint arcs (see figure 2). AC-3 iteratively makes individual arcs consistent by removing variable domain values inconsistent with a given constraint, until all arcs have been considered and the network is consistent. Then, if there remains a variable with more than one domain value, a procedure called domain splitting can be applied to that variable to split the CSP into disjoint cases so that AC-3 can recursively solve each case. The CSP applet provides several mechanisms to allow a learner to explore how AC-3

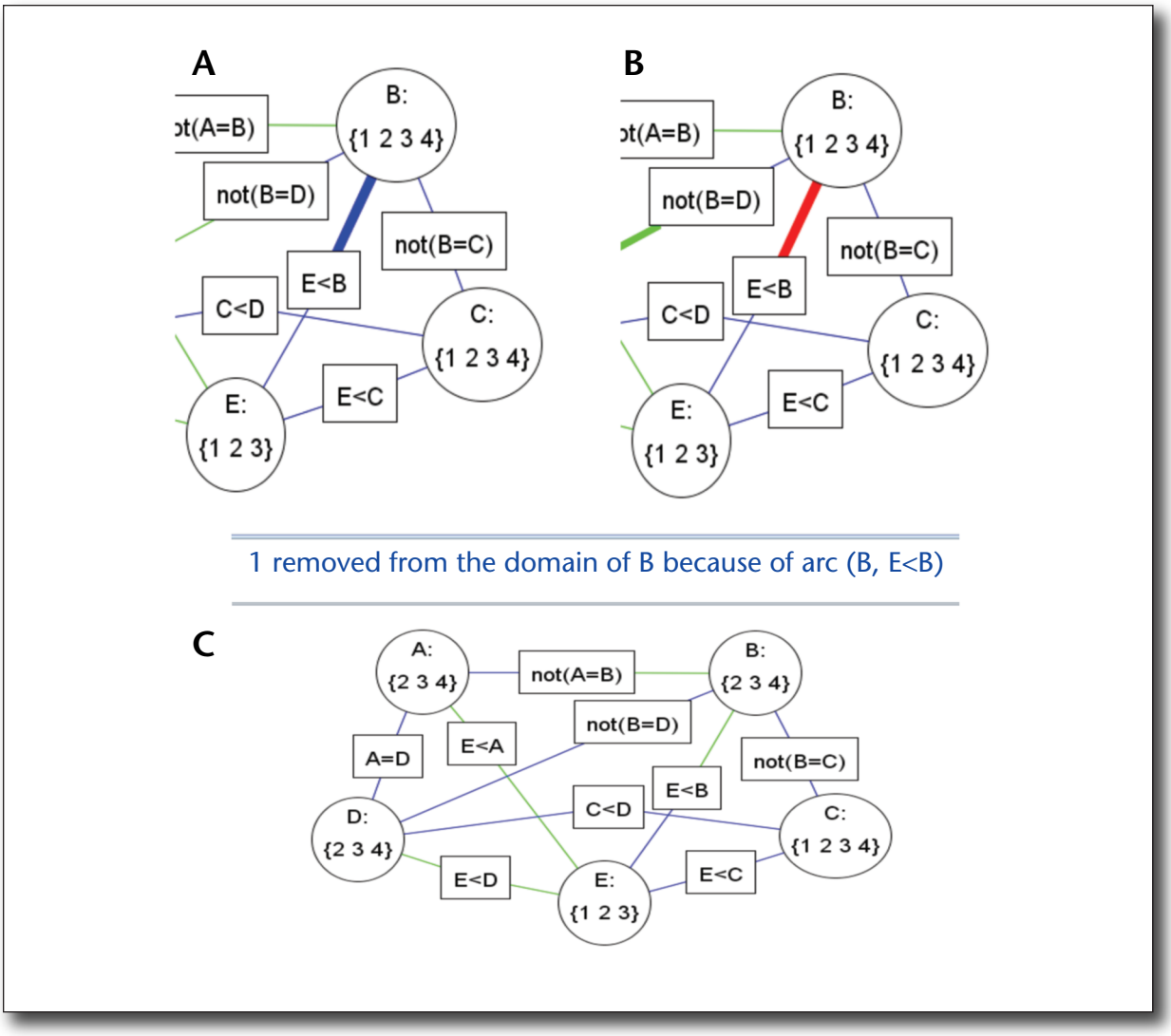

Figure 3. Basic Steps of AC-3.

solves a variety of available CPSs. These mechanisms are accessible through the toolbar shown at the top of figure 2 or through direct manipulation of graph elements. In particular, the user can perform the following: (1) Use the fine step button to see how AC-3 goes through its three basic steps: selecting an arc, testing it for consistency, removing domain values to make the arc consistent. Figure 3 shows one instance of fine stepping through the CSP shown in figure 2. In figure 3a, AC-3 selects the blue (dark) arc representing the constraint that the value selected for B should be greater than the value selected for E. In figure 3b, this arc is turned red (light) because it is not arc consistent (given the value 1 for B, there is no corresponding value in E that would satisfy the constraint). In figure 3c, the inconsistent value of 1 is removed from the domain of variable B. (2) Directly click an arc to apply all these steps at once. (3) Automatically fine step through the completion of the problem (auto arc consistency button). (4) Pause auto arc consistency (stop button). (5) Select a variable to split on, and specify a subset of its values for further application of AC-3. Figure 4a shows the domain splitting dialogue for the variable D, after all arcs in the CSP in figure 3c have been made consistent. After 3 gets selected in the dialogue box, a new CSP is generated with that value for D (figure 4b). (6) Retrieve alternative subnetworks generated by domain splitting (backtrack button). Continuing the example shown in figure 4, through backtracking the user can access one of the alternative CSPs resulting from domain splitting on D, in this case the CSP with D taking the value 4 (figure 4c). (7) Return the graph to its initial status (reset button).

As a student steps through a problem, the message panel above the graph panel reports a description of each step. Another message panel situated below the graph panel reports the history of domain splitting

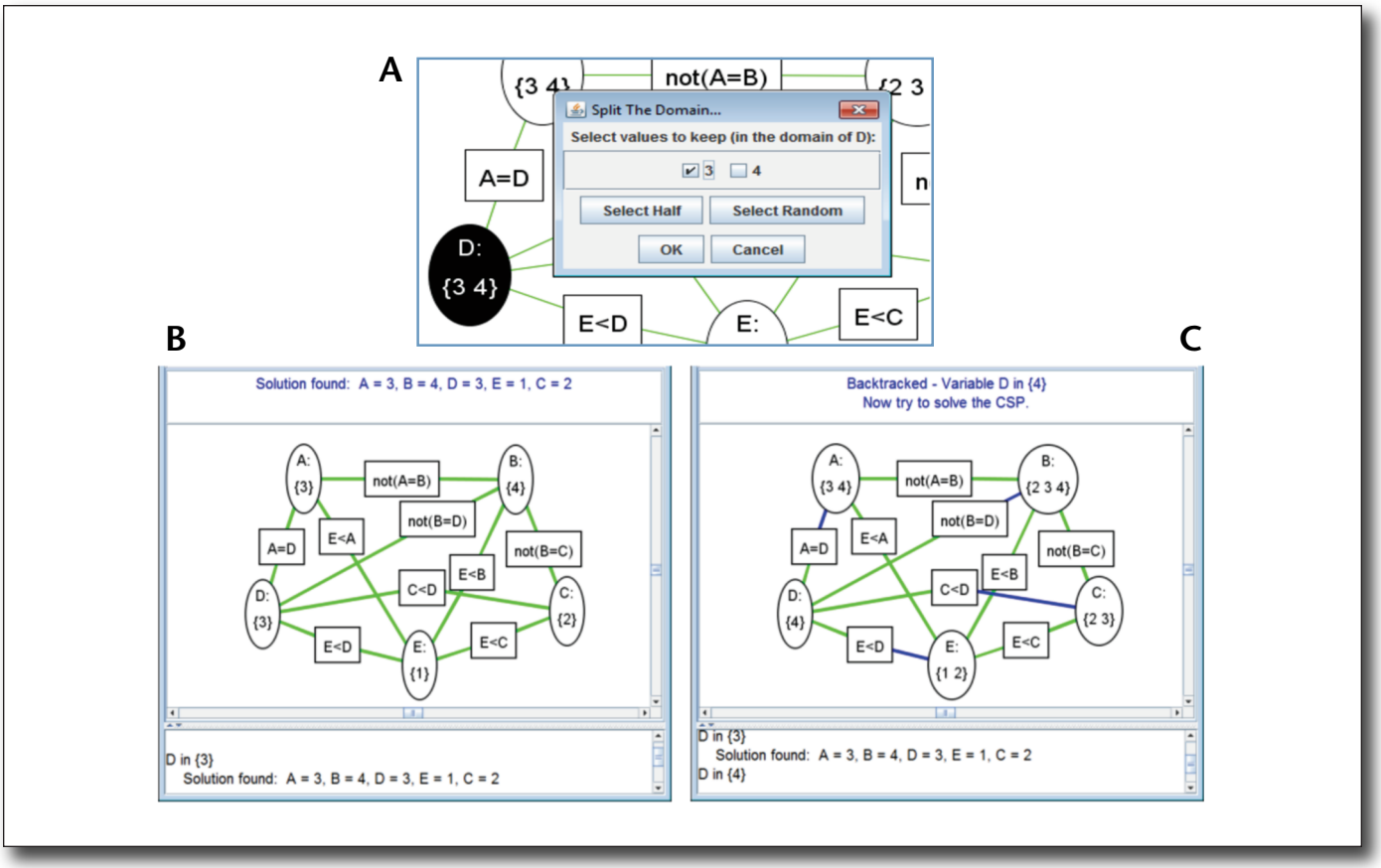

Figure 4. Domain Splitting and Backtracking.

decisions made by the user, that is, which value-variable assignment has been selected at each domain splitting point.

It should be noted that there is no specific definition of correct or incorrect behaviors or solutions when working with the CSP applet. The student can play with the simulation at will to see how AC-3 solves the set of available CSPs. The CSP applet currently does not monitor what its users do, nor does it provide any explicit support to help students learn at best from the mechanisms described above. Research however, shows that some students may benefit from this explicit support, since unaided exploration of interactive simulations can fail to help students learn (Shute 1993). The next section describes a study to test the performance of the user-modeling framework described in the previous section when applied to the CSP applet and its potential to support personalized interventions that can improve the CSP applet's effectiveness for all learners.

Testing the Framework on the CSP Applet

In this study, 65 university students interacted with the CSP applet during sessions of two hours duration. The participants were chosen so that they had not taken an AI course but were familiar with basic graph theory. During the study, each participant was introduced to the AC-3 algorithm and then took a pretest on the topic. Next, the participant used the applet on two CSP problems of increasing difficulty and finished with a posttest. The resulting data set includes 13,078 actions over 62,752 seconds of interaction. The features calculated from these logs include usage frequency of each interface action as well as mean and standard deviation of latency between actions. Average latency is an indicator of the time spent reflecting after each action, while standard deviation of latency tells if the user was consistent or selective in the amount of time he or she spent reflecting after each action. Since we have 7 interface actions, the calculated feature vectors are 21-dimensional.

Clustering on the aforementioned data set generated two clusters (we used C-index as described by Milligan and Cooper [1985] to determine the optimal number of clusters for the data). To verify whether there is any relation between these clusters and student learning, we used proportional learning gains (PLGs) computed from each student's pre- and posttests. We found a significant difference in PLG (p = .03, as per an independent samples t-test), with a medium effect size (Cohen's d = .47). We refer to

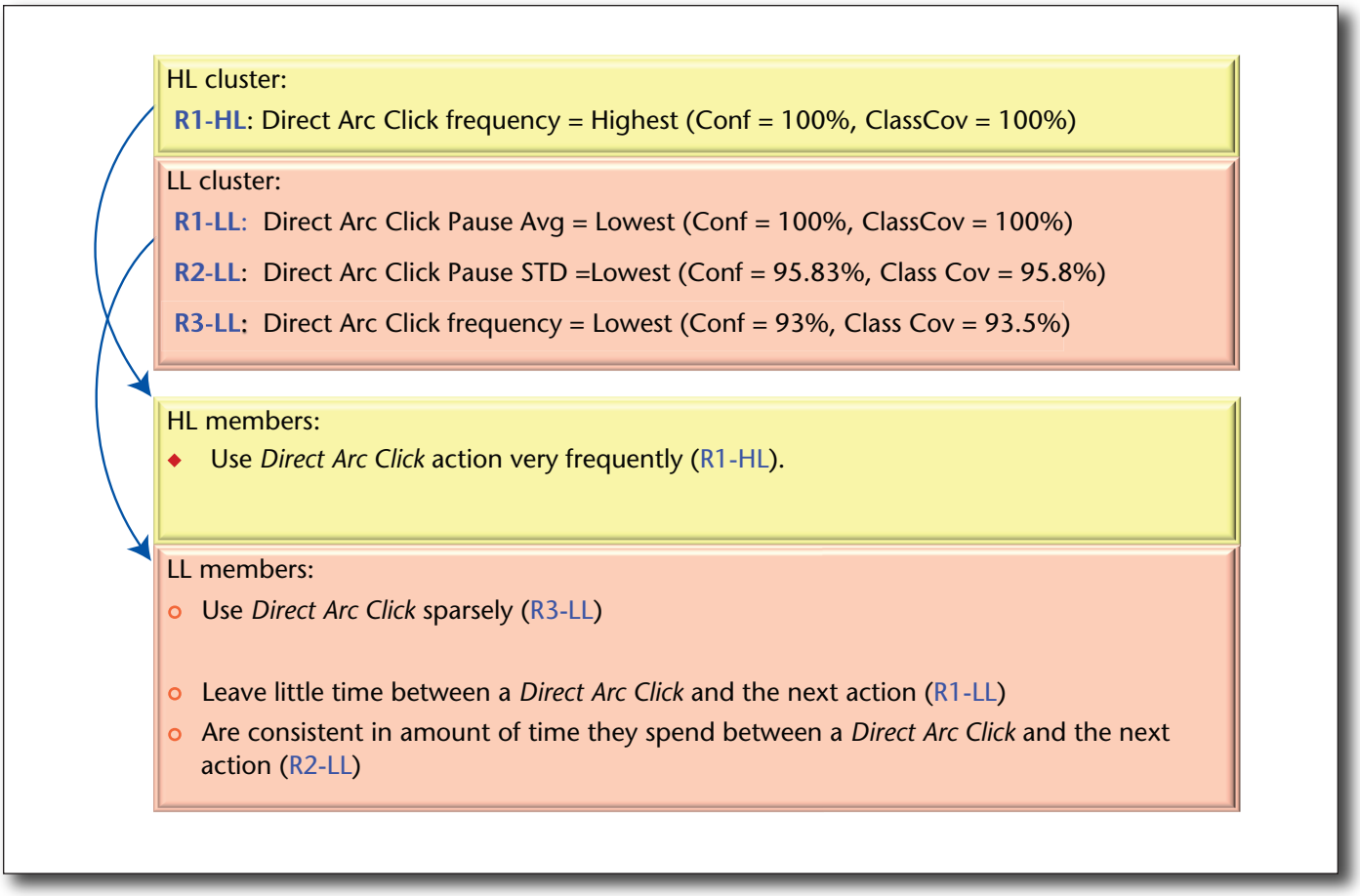

Figure 5. Sample CARs for the CSP Applet.

these clusters as high learners (HL, n = 18, M = 61.32, SD = 27.38) and low learners (LL, n = 47, M = 39.28, SD = 62.06). We found no significant difference between the average pretest scores of LL and HL. Thus, existing student knowledge cannot explain the difference in learning between the two clusters, leaving the difference in behavior patterns between the HL and LL group as the factor that affects learning.

Figure 5 shows a subset of the representative rules identified by association rule mining for the HL and LL clusters, specifically the subset related to the usage of the direct arc click functionality. The figure also shows, for each rule, its value for the two measures used for selecting the rules that are representative for a cluster, namely *rule's confidence* (conf) that is, the probability that user u belongs to this cluster if rule applies to u's behavior and *rule's support* or coverage within its cluster (class cov), that is, the probability that this rule applies to the user u, given that u belongs to this cluster.

The rules in figure 5 were generated by discretizing the continuous feature vectors in our data set into seven bins[4] (working with continuous features would generate a large number of fine-grained rules unsuitable for classification).

Direct arc click appears with values in the highest bin in the preconditions for Rule1-HL for the HL cluster, and in the lowest bin in Rule3-LL for LL cluster, indicating that LL members use direct arc click much less than HL members. The high class coverage of Rule1-HL (100 percent) indicates that high frequency of arc click pertains to all high learners, and thus it is a behavior that appears to reliably foster learning. This result makes sense pedagogically, because direct arc click requires that a student proactively choose which arc the AC-3 algorithm should select next, suggesting that the student is engaged in the exploratory process as opposed to being a passive spectator of the simulation. Thus, from the point of view of using the outcome of the student modeling process to generate didactic interventions that can help students use the CSP applet more effectively, Rule1-HL and Rule3-LL directly inform an intervention that encourage students to use direct arc click frequently if they do not so spontaneously. In addition, Rule1 and Rule2 for LL (which also have high class coverage) show low values of direct arc click pause average and standard deviation, suggesting that even when LL select arcs proactively, they consistently spend little time thinking about this action's outcome. Thus, these rules suggest

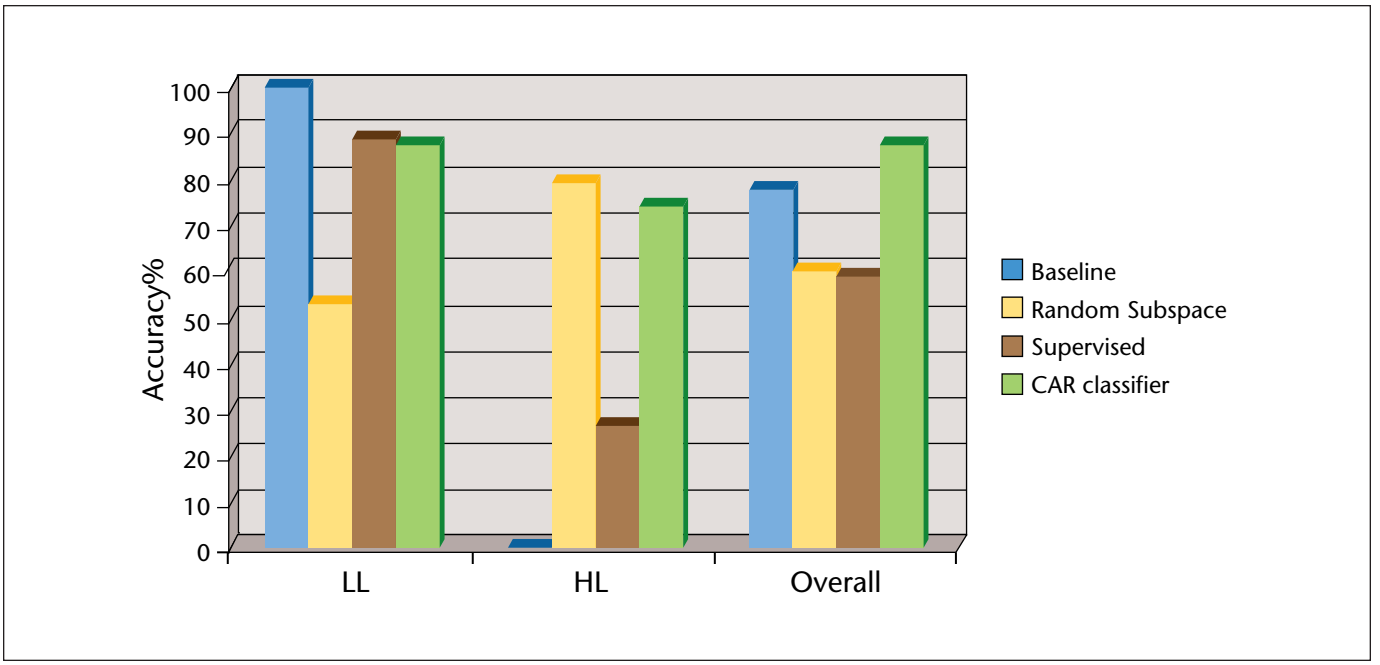

Figure 6. The Overtime Average Accuracy of Different Classifiers Compared to the New Rule-Based Classifier.

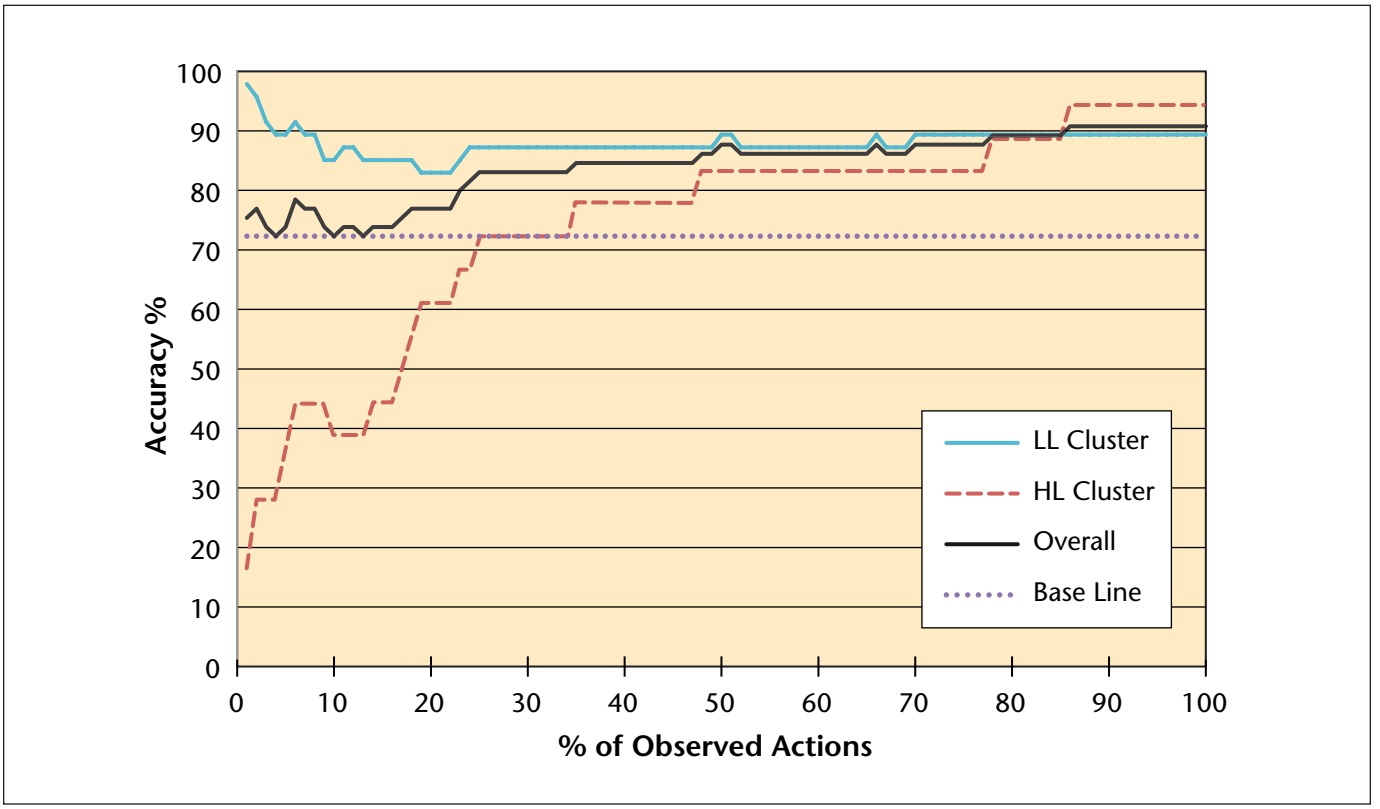

Figure 7. Accuracy of the CAR Classifier as a Function of the Percentage of Observed Actions.

an intervention designed to support low learners in paying more attention to the outcome of their direct arc clicks.

To evaluate the accuracy of the CAR classifier built upon the outcome of rule association mining, we used the evaluation component in the framework to compare its performance against the performance of (1) a baseline that always predicts the most likely label (LL in our data set); (2) the best achieving classifier among various complex classifiers available in Weka, namely the random subspace metaclassifier using C4.5 as the base classifier. Note that all these three classifiers use the categories learned through the unsupervised clustering performed using the framework. We also want to compare our approach against a fully supervised approach that starts from categories defined based on the available learning gains. To build this classifier, we calculated the median of the learning gains and labeled the students above the median as high learners and others as low learners. We then trained and tested a C4.5 classifier with these new labels. C4.5 was chosen because it was the best performing classifier among classifiers that provide information on why a label is assigned to the user, an important feature of the CAR classifier, and highly valuable for providing adaptive interventions. Figure 6 shows the overtime average accuracy of these four classifiers, both in terms of percentage of correct classifications for the individual clusters (LL and HL), and overall. The CAR classifier has the highest overall accuracy, and the differences with the other classifiers are statistically significant ($p < .001$), with a large effect size ($d > 3$). For each cluster, the accuracy of CAR classifier is comparable with the best competitor, but no other classifier achieves the same level of accuracy in both clusters. Figure 7 shows accuracy of the CAR classifier as a function of the percentage of observed actions, both overall and for the individual clusters.

For comparison, we include the overall accuracy of the baseline, which is the best performing classifier after CAR. The CAR classifier reaches a relatively high accuracy in early stages of the interaction, which is very important when the goal is to provide adaptive interventions to improve the user experience with the educational software. The overall accuracy of the CAR classifier becomes consistently higher than the baseline before observing 20 percent of user actions, and accuracy on each cluster goes above 80 percent after seeing about 50 percent of the actions, while the baseline consistently misclassifies high learners throughout.

Discussion and Future work

The results presented in the previous section provide encouraging evidence that it is possible to perform student modeling at the behavioral level even when it is difficult to define a priori the behaviors that should be captured during interaction with an e-learning environment. The empirical evaluation of our user-modeling framework on the CSP applet data set shows that it can both cluster users into groups reflecting different learning abilities based on the users' interaction behaviors, as well as classify new users accurately based on these behaviors. Furthermore, the framework generates rules that provide a fine-grained description of common behaviors for users in different clusters and appear to be suitable to guide adaptive interventions targeted at improving these behaviors when they are deemed to be suboptimal by the CAR classifier.

We are currently working on designing and dynamically generating these interventions (Kardan and Conati 2012a), with the goal of running a user study to compare a student-adaptive version of the CSP applet that includes these interventions against the standard version described earlier. This study will provide insights on how far the adaptive interventions based on class association rules can go in actually improving student learning without any assessment of the cognitive and metacognitive factors that underlie the students' behaviors.

These cognitive factors include for instance, the current student's knowledge of relevant domain concepts and skills (in case of the CSP applet, the various components of a CSP problem and of the AC-3 algorithm). Having a student model that includes an explicit representation of the connection between a student's interaction behaviors and her underlying domain knowledge allows for leveraging the observed behaviors to perform knowledge assessment. Knowledge assessment can then be used for providing richer adaptive interventions that go beyond the strictly behavioral suggestions supported by our CAR classifier student model, such as reteaching skills that the student model indicates to be poorly understood. In the CSP applet for instance, a student may not perform direct arc clicks because she does not understand how the arcs are selected by AC-3. If the student model can make this inference, it can also trigger an intervention to explain to the student that arc selection can be made randomly since selection order does not affect the outcome of arc consistency. Creating the connections between knowledge and behaviors however, is usually a laborious and time consuming process, even in relatively well understood problem-solving domains (for example, Kodaganallur, Weitz, and Rosenthal [2005]). Having the CAR approach isolate the behaviors that are important for learning (or lack thereof) in open-ended interactions may facilitate the knowledge engineering process. We want, however, to first evaluate the performance of lightweight didactic interventions based solely on behaviors, and move to more sophisticated knowledge assessment once the limitations of this approach have been clearly established.

Metacognitive factors that may be useful to repre-

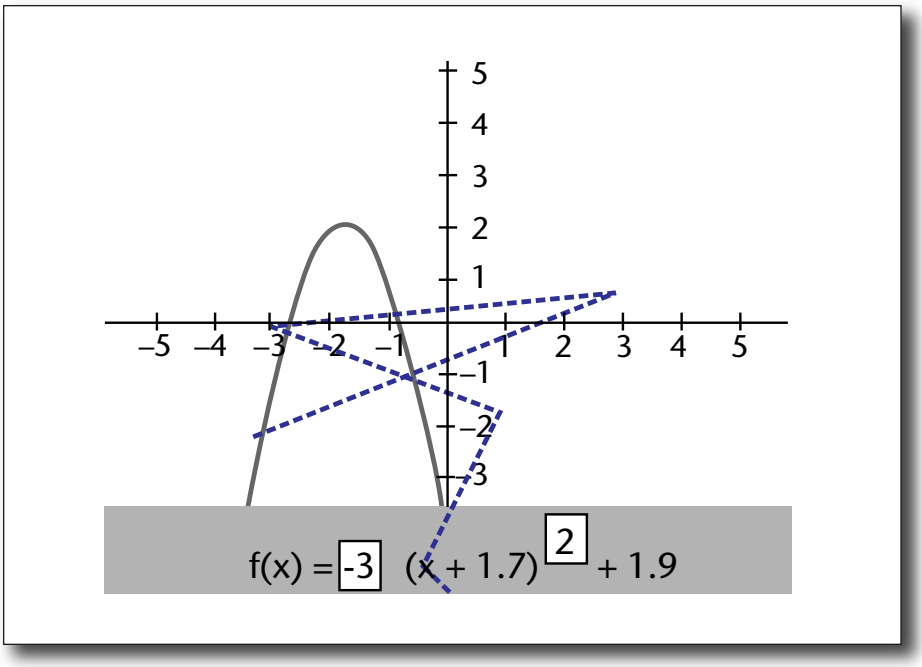

Figure 8. Sample Gaze Shift.

sent in a student model for exploratory and open-ended interactions include self-explanation (one's tendency to clarify and elaborate the available instructional material in terms of the underlying domain knowledge [Chi et al. 1994]), as well as other self-regulatory processes (for example, self-monitoring, goal setting, summarizing [Winne 2011]) that can help students structure their learning. Conati and Merten (2007) started investigating how to model students' self-explanation during an exploratory task by using as a test bed a simulation designed to help students understand simple mathematical functions. This work showed that it is possible to accurately assess students' tendency to self-explain the outcome of their exploratory actions by tracking both latency between actions as well as attention patterns indicative of reflection on action outcomes. Conati and Merten (2007) also showed that assessment of self-explanation, in turn, improves the model's accuracy in assessing the effectiveness of a student exploration. The student model used by Conati and Merten (2007) is a Bayesian network that explicitly represents the connections between exploration and self-explanation, as well as between self-explanation and its predictors (action latency and gaze patterns). The conditional probabilities defining these connections were learned from interaction and gaze data hand-labeled for instances of self-explanation by means of laborious protocol analysis (Conati et al. 2005). The gaze information tracked by this model relates to the occurrence of a simple gaze pattern defined a priori as being relevant for learning with this particular simulation: a gaze shift between two panels, one showing a function equation and one showing the related plot (see figure 8). The main exploratory action available in this simulation is to change either the equation or the plot, and see how the change affects the other component. Hence the definition of the aforementioned gaze shift as a relevant pattern to indicate self-explanation in this learning environment.

Having a student model that represents a student's self-explanation tendency, its defining behaviors, and its connection with exploration effectiveness enables an ITS to generate interventions that explicitly target this metacognitive skill (Conati 2004). But once again, this detailed student model is very laborious to build, and the process used to build it may not scale up to more complex learning environments, in which it is not as intuitive to predefine gaze patterns and interaction behaviors that reflect self-explanation. Furthermore, even when patterns can be specified, they are task specific and may not directly transfer to a different application. We are currently working on a more lightweight and generalizable approach that relies on applying the CAR framework described earlier to data that include both interaction events and eye tracking of users interacting with the CSP applet. The goal is to investigate whether we can obtain clusters of user types and a related CAR classifier that provide a representation of user effective and ineffective behaviors in terms or both actions and attention patterns. This approach is more general than the knowledge-intensive approach described earlier because the gaze data, like the action data, is expressed in terms of standard features that are either task independent (for example, fixation rate, average and standard deviation of fixation durations), or based solely on identifying the main components of the target interface (for example, average number of transitions between any two of such components) (Kardan and Conati 2012b). It is left to the CAR classifier to identify gaze patterns that are indicative of users' learning with that interface. The resulting student model does not have an explicit representation of a learner's metacognitive behaviors and tendencies. However, it may still pin down attention and action behaviors that implicitly relate to whether a learner is engaging or not in relevant metacognitive activities. We plan to investigate whether generating adaptive interventions to stimulate these behaviors may indirectly foster relevant metacognition.

We expect the approach of going for more shallow but faster to build data-based student models to become more and more prominent as the increased popularity of online learning tools will facilitate the collection of large amounts of data. For instance, in relation to providing personalized support to problem solving, researchers have been experimenting with moving away from student models that perform knowledge tracing, to rely instead on data mining to learn pat-

terns that can be leveraged to provide tutoring support, such as selecting the next problem for the student to solve (for example, Mayo and Mitrovic [2001]). Extensive research on data-based approaches has also been done for student modeling at the affective level, to leverage the availability of sensors that can detect a variety of evidential data related to affective reactions, such as increased heart rate, specific postures and facial expressions, spikes in skin conductance (for example, Arroyo et al. [2009]; D'Mello and Graesser [2010]; Muldner, Burleson, and VanLehn [2010]).

Researchers however, are also exploring the option of combining the knowledge-based and the data-based approaches to student modeling, as in Conati and Merten (2007). This option has been extensively investigated to refine existing knowledge-tracing models by using data to fine-tune model parameters (for example, Baker et al. [2010]; Gong, Beck, and Heffernan [2011]; Koedinger et al. forthcoming). It has also been explored to define affective student models that can provide information on the causes of a learner's emotional reactions, in addition to detecting these reactions (Conati and Maclaren 2009a, 2009b). This option is obviously the most labor intensive and requires having both a good theoretical understanding of the learner's states and processes to be modeled, as well as data to instantiate the details of the model. It is however, the option that can generate the most precise and informative models. Whether the added cost is worth the effort is an open research question.

Related Work

The student modeling framework described here evolved from previous versions devised in our group. Amershi and Conati (2009) present a version that does not include association rule mining of behaviors, and thus cannot identify which behaviors are responsible for user classification and should be the target of personalized tutorial interventions. Bernardini and Conati, (2010) present a proof of concept version that includes association rule mining and CAR classification. The work presented here refines that proof of concept into a comprehensive user-modeling framework that streamlines the phases necessary to generate a user classifier from an initial data set of raw interaction logs.

Association rules have been widely used for offline analysis of learners' interaction patterns with educational software; for example, to discover (1) error patterns that can help improve the teaching of SQL (Merceron and Yacef 2003); (2) similarities among exercises for algebra problem solving in terms of solution difficulty (Freyberger, Heffernan, and Ruiz 2004); (3) usage patterns relevant for revising a web-based educational system spanning a complete university course (García et al. 2009a). More relevant to the work described here is the work done by Romero et al. (2010). They applied rare association rule mining to find the relation between the online activities of students in a learning management system and their final mark. Similarly, Perera et al. (2009) used k-means clustering and sequential pattern mining to find interesting patterns for stronger and weaker students in a collaborative software development tool. However, neither of the last two works attempts to build a classifier and predict the performance of users. For additional review of works that use association rule mining for educational purposes the reader is referred to Romero and Ventura (2010).

Most work on using association rules for online adaptation has been done within research on recommender systems. In Changchien and Lu (2001), for instance, association rule mining is used to match the user type with appropriate products. In this work, however, there is no online classification of new users. Associative classification is also used by Zhang and Jiao (2007) to classify user requirements and generate personalized item recommendation in an e-commerce application. The main difference with our work is that the approach of Zhang and Jiao (2007) needs labeled data, while ours can work with unlabeled data sets. Kim and Yum (2011) used association rule mining on click stream and other interaction data in an e-commerce system for recommending items to users.

In educational context, the work by Romero et al. (2009) is the most relevant to the research described here, in that the authors aim to use clustering and sequential pattern mining to recognize how students navigate through a web-based learning environment, classify them, and use some teacher-tuned rules for recommending further navigation links accordingly. The evaluation of this work focused on analyzing the quality of the rules generated by different algorithms, but no results have yet been presented on the classification accuracy of the proposed approach.

Conclusions

In this article, we presented research on creating student models that can support student-adaptive instruction during e-learning activities beyond problem solving. Student modeling is a difficult AI problem in general, because it requires endowing a tutoring agent with the ability to understand a student's states and processes relevant for learning from often limited and ambiguous information on how the student interacts with the target learning environment. Student modeling for problem-solving activities can at least rely on some understanding of which solutions should be represented in the model, what constitutes a correct problem-solving step, and which cognitive processes and states the model should be able to assess. Educational technology, however, continues to produce novel environments often consist-

ing of activities not as well structured and understood as problem solving, making the student modeling problem increasingly challenging.

The user-modeling framework presented in this article is an attempt to address this challenge. It uses clustering and class associating rule mining to discover and recognize relevant interaction patterns during student interaction with educational software. An empirical evaluation with an interactive simulation for solving constraint satisfaction problems provided initial evidence that this framework can both (1) cluster users into meaningful groups; and (2) classify new users accurately by using classification rules that appear to be suitable to guide adaptive interventions targeted at improving interaction effectiveness. The approach is in principle applicable to a wide variety of educational environments because it does not require detailed knowledge of the pedagogical and design principles underlying each specific system and related educational activities. The potential drawback is that the interventions at the strictly behavioral level that an ITS can provide without this knowledge may not be sufficient to make a difference in student learning. Only extensive empirical studies with a variety of different e-learning environments can clearly identify the strengths and limitations of this student modeling approach. Nonetheless, we argue that this approach is worth exploring because even if it proves to be unsuitable to directly support real-time adaptive tutoring, it can help identify the relevant behaviors that affect student performance with a specific e-learning system, possibly providing the basis for more sophisticated knowledge-based learner modeling.

Given our society's increasing need for high quality teaching and training, e-learning and online learning are becoming increasingly critical to complementing human tutoring in a large variety of fields and settings. The research presented in this article is a step toward providing comprehensive student modeling and personalized instruction for a range of e-learning activities that can foster student understanding at different stages of the learning process and for learners with different preferences and abilities. In this paper, we focused on activities related to exploring interactive simulations. Other new forms of intelligent computer-based tutoring that are being investigated include, among others, support for collaborative learning (for example, Isotani and Mizoguchi [2008]), teachable agents that can help students learn by acting as peers that students can tutor (for example, Leelawong and Biswas [2008]) intelligent support for learning from educational games (for example, Johnson [2010]; Manske and Conati [2005]), and intelligent tutoring for ill-defined domains (for example, Lynch et al. [2008]). Providing these forms of intelligent tutoring, like providing intelligent support for exploring interactive simulations, poses unique challenges for student modeling, because it requires understanding and formalizing domains as well as student behaviors and mental states often not as well understood as those involved in traditional problem solving. Advances in AI techniques for reasoning under uncertainty, machine learning, decision-theoretic planning, as well as the increasing availability of rich interaction and sensor data that can help capture the relevant user states, are promising means for the field to tackle these challenges and contribute online personalized instruction to our society's ever-increasing need for high-quality teaching and training.

Notes

1. Another approach is to create data points from sequence mining. This approach has been successfully applied when there are few high-level types of actions (for example, a successful attempt on the first step of a problem, asking for hints, and so on), and when action order is unambiguously important (for example, attempt a problem step before asking for help). In contrast, we are interested in interactions encompassing finer-grained interface actions that can be done in any order, which makes looking for recurring action sequences computationally expensive without a clear added value.

2. The functionalities used from Weka are standard algorithms that can be replaced by any other standard tool or implemented internally. They are transparent to the final user of the framework.

3. AIspace is available at www.aispace.org.

4. The framework includes a functionality that selects the optimal number of bins by computing the cross-validated classification accuracy of a simple rule-based classifier known to be sensitive to feature discretization (decision-table), using different numbers of bins.

References

Albacete, P. L., and VanLehn, K. 2000. The Conceptual Helper: An Intelligent Tutoring System for Teaching Fundamental Physics Concepts. In *Intelligent Tutoring Systems,* Lecture Notes in Computer Science, ed. G. Gauthier, C. Frasson, and K. Vanlehn, 564–573. Berlin: Springer. dx.doi.org/10.1007/3-540-45108-0_60

Aleven, V.; McLaren, B.; Roll, I.; and Koedinger, K. 2006. Toward Metacognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence Education* 16(2): 101–128.

Amershi, S.; Carenini, G.; Conati, C.; Mackworth, A. K.; and Poole, D. 2008. Pedagogy and Usability in Interactive Algorithm Visualizations: Designing and Evaluating Cispace. *Interacting with Computers* 20(1): 64–96. dx.doi.org/10.1016/j.intcom.2007. 08.003

Amershi, S., and Conati, C. 2009. Combining Unsupervised and Supervised Classification to Build User Models for Exploratory Learning Environments. *Journal of Educational Data Mining* 1(1): 18–71.

Arroyo, I.; Cooper, D. G.; Burleson, W.; Woolf, B. P.; Muldner, K.; and Christopherson, R. 2009. Emotion Sensors Go to School. In *Proceeding of the 2009 Conference on Artificial Intelligence in Education,* 17–24. Amsterdam: IOS Press.

Baker, R. S. J. d; Corbett, A. T.; Gowda, S. M.; Wagner, A. Z.; MacLaren, B. A.; Kauffman, L. R.; Mitchell, A. P.; and

Giguere, S. 2010. Contextual Slip and Prediction of Student Performance after Use of an Intelligent Tutor. In *User Modeling, Adaptation, and Personalization,* Lecture Notes in Computer Science, ed. P. D. Bra, A. Kobsa, and D. Chin, 52–63. Berlin: Springer

Baker, R. S.; Corbett, A. T.; Roll, I.; and Koedinger, K. R. 2008. Developing a Generalizable Detector of When Students Game the System. *User Modeling and User-Adapted Interaction* 18(3): 287–314. dx.doi.org/10.1007/s11257-007-9045-6

Baker, R. S. J. d.; Corbett, A. T.; Koedinger, K. R.; Evenson, S.; Roll, I.; Wagner, A. Z.; Naim, M.; Raspat, J.; Baker, D. J.; and Beck, J. E. 2006. Adapting to When Students Game an Intelligent Tutoring System. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 392–401. Berlin, Heidelberg: Springer-Verlag. dx.doi.org/10.1007/11774303_39

Bernardini, A., and Conati, C. 2010. Discovering and Recognizing Student Interaction Patterns in Exploratory Learning Environments. In *Intelligent Tutoring Systems,* ed. V. Aleven, J. Kay, and J. Mostow, 125–134. Berlin: Springer. dx.doi.org/10.1007/978-3-642-13388-6_17 PMid:20843738

Changchien, S., and Lu, T. C. 2001. Mining Association Rules Procedure to Support Online Recommendation by Customers and Products Fragmentation. *Expert Systems with Applications* 20(4): 325–335. dx.doi.org/10.1016/S0957-4174(01)00017-3

Chi, M. T. H.; De Leeuw, N.; Chiu, M. H.; and Lavancher, C. 1994. Eliciting Self-Explanations Improves Understanding. *Cognitive Science* 18(3): 439–477. DOI: 10.1207/s15516709cog1803_3

Conati, C. 2004. Toward Comprehensive Student Models: Modeling Metacognitive Skills and Affective States in ITS. In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, Lecture Notes in Computer Science, 902. Berlin: Springer. dx.doi.org/10.1007/978-3-540-30139-4_114

Conati, C.; Gertner, A.; and VanLehn, K. 2002. Using Bayesian Networks to Manage Uncertainty in Student Modeling. *User Modeling and User-Adapted Interaction* 12(4): 371–417. dx.doi.org/10.1023/A:1021258506583

Conati, C., and MacLaren, H. 2009a. Modeling User Affect from Causes and Effects. In *User Modeling, Adaptation, and Personalization,* ed. G.-J. Houben, G. McCalla, F. Pianesi, and M. Zancanaro, Lecture Notes in Computer Science, 4–15. Berlin: Springer. dx.doi.org/10.1007/978-3-642-02247-0_4

Conati, C., and MacLaren, H. 2009b. Empirically Building and Evaluating a Probabilistic Model of User Affect. *User Modeling and User-Adapted Interaction* 19(3): 267–303. dx.doi.org/10.1007/s11257-009-9062-8

Conati, C., and Merten, C. 2007. Eye-Tracking for User Modeling in Exploratory Learning Environments: An Empirical Evaluation. *Knowledge-Based Systems* 20(6): 557–574. dx.doi.org/10.1016/j.knosys.2007.04.010

Conati, C.; Merten, C.; Muldner, K.; and Ternes, D. 2005. Exploring Eye Tracking to Increase Bandwidth in User Modeling. In *User Modeling 2005,* ed. L. Ardissono, P. Brna, and A. Mitrovic, Lecture Notes in Computer Science, 357–366. Berlin: Springer. dx.doi.org/10.1007/11527886_47

Corbett, A. T., and Anderson, J. R. 1994. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction* 4(4): 253–278. dx.doi.org/10.1007/BF01099821

D'Mello, S.; Lehman, B.; Sullins, J.; Daigle, R.; Combs, R.; Vogt, K.; Perkins, L.; and Graesser, A. 2010. A Time for Emoting: When Affect-Sensitivity Is and Isn't Effective at Promoting Deep Learning. In *Proceedings of the 10th International Conference on Intelligent Tutoring System,* Volume Part I, 245–254. Berlin: Springer.

D'Mello, S. K., and Graesser, A. 2010. Multimodal Semi-Automated Affect Detection from Conversational Cues, Gross Body Language, and Facial Features. *User Modeling and User-Adapted Interaction* 20(2): 147–187. dx.doi.org/10.1007/s11257-010-9074-4

Desmarais, M. C. 2011. Performance Comparison of Item-to-Item Skills Models with the IRT Single Latent Trait Model. In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization*, 75–86. Berlin: Springer. dx.doi.org/10.1007/978-3-642-22362-4_7

Freyberger, J.; Heffernan, N.; and Ruiz, C. 2004. Using Association Rules to Guide a Search for Best Fitting Transfer Models of Student Learning. In *Workshop on Analyzing Student–Tutor Interactions Logs to Improve Educational Outcomes at Its Conference,* 1–4. Berlin: Springer.

García, E.; Romero, C.; Ventura, S.; and Castro, C. 2009a. An Architecture for Making Recommendations to Courseware Authors Using Association Rule Mining and Collaborative Filtering. *User Modeling and User-Adapted Interaction* 19(1–2) 99–132. dx.doi.org/10.1007/s11257-008-9047-z

García, E.; Romero, C.; Ventura, S.; and Castro, C. De 2009b. An Architecture for Making Recommendations to Courseware Authors Using Association Rule Mining and Collaborative Filtering. *User Modeling and User-Adapted Interaction* 19, 99–132.

Gong, Y.; Beck, J. E.; and Heffernan, N. T. 2011. How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis. *International Journal of Artificial Intelligence Education*. 21(3): 27–45.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. H. 2009. The Weka Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter* 11(1): 10–18. dx.doi.org/10.1145/1656274.1656278

Isotani, S. and Mizoguchi, R. 2008. Theory-Driven Group Formation Through Ontologies. In *Intelligent Tutoring Systems,* Lecture Notes in Computer Science, ed. B. P. Woolf, E. Aïmeur, R. Nkambou, and S. Lajoie, 646–655. Berlin: Springer. dx.doi.org/10.1007/978-3-540-69132-7_67

Johnson, W. L. 2010. Serious Use of a Serious Game for Language Learning. *International Journal of Artificial Intelligence in Education* 20(2): 175–195.

Kardan, S., and Conati, C. 2011. A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. Paper presented at the 4th International Conference on Educational Data Mining, Eindhoven, The Netherlands, 6–8 July.

Kardan, S., and Conati, C. 2012a. Providing Adaptive Support in an Exploratory Learning Environment by Mining User Interaction Data. Paper presented at the 5th International Workshop on Intelligent Support for Exploratory Environments (ISEE 2012), Chania, Greece, 14–18 June. PMid:22634040

Kardan, S., and Conati, C. 2012b. Exploring Gaze Data for Determining User Learning with an Interactive Simulation. In *User Modeling, Adaptation, and Personalization,* Lecture Notes in Computer Science, ed. J. Masthoff, B. Mobasher, M. Desmarais, and R. Nkambou, 126–138. Berlin: Springer. dx.doi.org/10.1007/978-3-642-31454-4_11

Kim, Y. S., and Yum, B.-J. 2011. Recommender System Based on Click Stream Data Using Association Rule Mining. *Expert Systems with Applications* 38(10): 13320–13327. dx.doi.org/10.1016/j.eswa.2011.04.154

Kodaganallur, V.; Weitz, R. R.; and Rosenthal, D. 2005. A Comparison of Model-Tracing and Constraint-Based Intelligent Tutoring Paradigms. *International Journal of Artificial Intelligence in Education* 15(2): 117–144.

Koedinger, K. R.; Anderson, J. R.; Hadley, W. H.; and Mark, M. A. 1997. Intelligent Tutoring Goes to School in the Big City. *International Journal of Artificial Intelligence in Education* 8(1): 30–43.

Koedinger, K.; Brunskill, E.; Baker, R. S. J. d.; McLaughlin, E. A.; and Stamper, J. 2013. New Potentials for Data-Driven Intelligent Tutoring System Development and Optimization. *AI Magazine* 34(3).

Leelawong, K., and Biswas, G. 2008. Designing Learning by Teaching Agents: The Bet-

ty's Brain System. *International Journal of Artificial Intelligence in Education* 18(3): 181–208.

Lynch, C.; Pinkwart, N.; Ashley, K.; and Aleven, V. 2008. What Do Argument Diagrams Tell Us about Students' Aptitude or Experience? A Statistical Analysis in an Ill-Defined Domain. Paper presented at the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains: Assessment and Feedback in Ill-Defined Domains. Montreal, Canada, 23 June.

Manske, M., and Conati, C. 2005. Modelling Learning in an Educational Game, In *Proceedings of the 2005 Conference on Artificial Intelligence in Education: Supporting Learning Through Intelligent and Socially Informed Technology,* 411–418. Amsterdam: IOS Press.

Mayo, M., and Mitrovic, A. 2001. Optimising Its Behaviour with Bayesian Networks and Decision Theory. *International Journal of Artificial Intelligence in Education* 12(2): 124–153.

Merceron, A., and Yacef, K. 2003. A Web-Based Tutoring Tool with Mining Facilities to Improve Teaching and Learning. Paper presented at the 11th Conference on Artificial Intelligence in Education, Sydney, Australia, 20–24 July.

Milligan, G. W., and Cooper, M. C. 1985. An Examination of Procedures for Determining the Number of Clusters in a Data Set. *Psychometrika* 50(2): 159–179. dx.doi.org/ 10.1007/BF02294245

Mitrovic, A. 2012. Fifteen Years of Constraint-Based Tutors: What We Have Achieved and Where We Are Going. *User Modeling and User-Adapted Interaction* 22(1–2): 39–72. dx.doi.org/10.1007/s11257-011-9105-9

Muldner, K.; Burleson, W.; and Vanlehn, K. 2010. "Yes!": Using Tutor and Sensor Data to Predict Moments of Delight During Instructional Activities. In *User Modeling, Adaptation, and Personalization,* ed. P. Bra, A. Kobsa, and D. Chin, 159–170. Berlin: Springer. dx.doi.org/10.1007/978-3-642-13470-8_16

Muldner, K., and Conati, C. 2007. Evaluating a Decision-Theoretic Approach to Tailored Example Selection. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence,* 483–488. Menlo Park, CA: AAAI Press.

Muldner, K., and Conati, C. 2010. Scaffolding Metacognitive Skills for Effective Analogical Problem Solving Via Tailored Example Selection. *International Journal of Artificial Intelligence Education* 20(2): 99–136.

Perera, D.; Kay, J.; Koprinska, I.; Yacef, K.; and Zaïane, O. R. 2009. Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Transactions on Knowledge and Data Engineering* 21(6): 759–772. dx.doi.org/10.1109/TKDE.2008. 138

Qu, L., and Johnson, W. L. 2005. Detecting the Learner's Motivational States in an Interactive Learning Environment. In *Proceedings of the 2005 Conference on Artificial Intelligence in Education: Supporting Learning Through Intelligent and Socially Informed Technology,* 547–554. Amsterdam: IOS Press.

Roll, I.; Aleven, V.; McLaren, B. M.; and Koedinger, K. R. 2007. Can Help Seeking Be Tutored? *Searching for the Secret Sauce of Metacognitive Tutoring,* 203–210. Amsterdam: IOS Press.

Romero, C.; Romero, J. R.; Luna, J. M.; and Ventura, S. 2010. Mining Rare Association Rules from E-Learning Data. Paper presented at the Third International Conference of Educational Data Mining, Pittsburgh, PA, 11–13 June.

Romero, C., and Ventura, S. 2010. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics,* Part C: Applications and Reviews 40(6): 601–618. dx.doi.org/ 10.1109/TSMCC.2010.2053532

Romero, C.; Ventura, S.; Zafra, A.; and Bra, P. 2009. Applying Web Usage Mining for Personalizing Hyperlinks in Web-Based Adaptive Educational Systems. *Computers and Education* 53(3): 828–840. dx.doi.org/ 10.1016/j.compedu.2009.05.003

Rossi, F.; Van Beek, P.; and Walsh, T. 2006. *Handbook of Constraint Programming*. Amsterdam: Elsevier Science.

Shute, V. J. 1993. A Comparison of Learning Environments: All That Glitters. In *Computers and Cognitive Tools, Technology in Education,* 47–73. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Thabtah, F. 2007. A Review of Associative Classification Mining. *The Knowledge Engineering Review* 22(1): 37–65. dx.doi.org/ 10.1017/S0269888907001026

VanLehn, K. 2006. The Behavior of Tutoring Systems. *International Journal of Artifical Intelligence Education* 16(3): 227–265.

Winne, P. H. 2011. A Cognitive and Metacognitive Analysis of Self-Regulated Learning. *Handbook of Self-Regulation of Learning and Performance,* 15–32. London: Routledge.

Zhang, C., and Zhang, S. 2002. *Association Rule Mining: Models and Algorithms*. Berlin: Springer. dx.doi.org/10.1007/3-540-46027-6

Zhang, Y., and Jiao, J. R. 2007. An Associative Classification-Based Recommendation System for Personalization In B2C E-Commerce Applications. *Expert Systems with Applications* 33(1): 357–367. dx.doi. org/10.1016/j.eswa.2006.05.005

Cristina Conati is an associate professor of computer science at the University of British Columbia, Vancouver, Canada. She received a "Laurea" degree (M.Sc. equivalent) in computer science from the University of Milan, Italy (1988), as well as an M.Sc. (1996) and Ph.D. (1999) in intelligent systems at the University of Pittsburgh. Conati's research goal is to integrate research in artificial intelligence, cognitive science, and human computer interaction to make complex interactive systems increasingly more effective and adaptive to the users' needs. Her areas of interest include intelligent user interfaces, user modeling, user-adaptive systems, and affective computing. Her research has received awards from the International Conference on User Modeling, the International Conference of AI in Education, the International Conference on Intelligent User Interfaces (2007), and the *Journal of User Modeling and User Adapted Interaction* (2002). Conati is an associate editor for the *Journal of AI in Education,* for the *IEEE Transactions on Affective Computing*, and for the *ACM Transactions on Intelligent Interactive Systems*.

Samad Kardan is a Ph.D. candidate in computer science at the University of British Columbia, Vancouver, Canada. He is a member of the Intelligent User Interfaces research group at the Laboratory for Computational Intelligence (LCI). His current research involves applying data mining on user-interaction data with the goal of providing personalized support in open-ended learning environments. His previous research includes modeling learning and student knowledge using different sources of user-interaction data.