A DATA MINING APPROACH FOR ADDING ADAPTIVE INTERVENTIONS TO EXPLORATORY LEARNING ENVIRONMENTS

by

Samad Kardan

B.Sc., Computer Engineering, Ferdowsi University of Mashhad, Iran, 2006

M.Sc., Information Technology, Amirkabir University of Technology, Iran, 2009

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

June 2017

Abstract

Due to the open-ended nature of the interaction with Exploratory Learning Environments (ELEs), it is not trivial to add mechanisms for providing adaptive support to users. Our goal is to devise and evaluate a data mining approach for providing adaptive interventions that help users to achieve better task performance during interaction with ELEs.

The general idea of this thesis is as follows:

> In an exploratory and open-ended environment, we collect interaction data of users while they are working with the system, and then find representative patterns of behavior for different user groups that achieved various levels of task performance. We use these patterns to provide adaptive real-time interventions designed to suggest or enforce the effective interaction behaviors while discouraging or preventing the ineffective ones. We test and confirm the hypothesis that as a result of these interventions, the average learning performance of the new users who work with the adaptive version of this ELE is significantly higher than the non-adaptive version.

We use an interactive simulation for learning Constraint Satisfaction Problems (CSP), the AIspace CSP applet, as the test-bed for our research and propose a framework which covers the entire process described above, called the User Modeling and Adaptation (UMA) framework. The contributions of this thesis are two-fold:

- It contributes to the Educational Data Mining (EDM) research, by devising, modifying, and testing different techniques and mechanisms for a complete data mining based approach to delivering adaptive interventions in ELEs summarized in the UMA framework. The UMA framework consists of 3 phases: Behavior Discovery, User Classification, and Adaptive Support. We assessed each of the above phases in a series of

user studies. This work is the first to fully evaluate and provide positive evidence for the use of a data mining approach for deriving and delivering adaptive interventions in ELEs with the goal of improving the user's performance.

- It also contributes to the user modeling and user-adapted interaction community by providing new evidence for the usefulness of the eye-gaze data for the purpose of predicting learning performance of users while interacting with an ELE.

Lay Summary

Exploratory learning environments such as interactive computer simulations that support open-ended interaction with the students, can be very effective tools for self-learning. However, there is evidence that only a subset of students can use these tools effectively. This dissertation proposes a framework for adding a mechanism that provides personalized support to students while they are using these educational tools. The proposed framework relies on the data collected from students and uses data mining techniques to discover the effective and ineffective patterns of behavior in student interactions. We then use these patterns to provide hints to the new students, who need them, as they are working with these tools. We evaluated this framework on two different simulations and the results show that we can effectively detect if a student needs help. Also, our experiments show that the personalized hints provided, effectively improve the learning performance of the students.

Preface

All of the work presented henceforth was conducted in the Intelligent User Interfaces (IUI) group in the Laboratory for Computational Intelligence (LCI) at the Department of Computer Science, University of British Columbia (UBC), Point Grey campus. All user experiments were approved by the University of British Columbia's Research Ethics Board [certificate # H12-03328].

The author was the primary contributor, responsible for all major areas of concept formation, data collection, analysis, and manuscript writing for all the work presented in this thesis as well as all the resulting publications. This research was conducted under the supervision of Professor Cristina Conati, and she was the supervisory author for the publications mentioned here.

Chapters 3, 4, 5, 6, 7, and 8 are based on the following published papers in peer-reviewed venues:

- Kardan, S., & Conati, C. (2015). Providing Adaptive Support in an Interactive Simulation for Learning: An Experimental Evaluation. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (pp. 3671–3680). New York, NY, USA: ACM. https://doi.org/10.1145/2702123.2702424.
- Kardan, S., Roll, I., & Conati, C. (2014). The Usefulness of Log-Based Clustering in a Complex Simulation Environment. In S. Trausan-Matu, K. E. Boyer, M. Crosby, & K. Panourgia (Eds.), *Intelligent Tutoring Systems* (pp. 168–177). Springer International Publishing. (**runner-up for best paper**)
- Kardan, S., & Conati, C. (2013a). Comparing and Combining Eye-gaze and Interface Actions for Determining User Learning with an Interactive Simulation. In S. Carberry, S. Weibelzahl, A. Micarelli, & G. Semeraro (Eds.), *User Modeling, Adaptation, and*

Personalization (pp. 215–227). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-38844-6_18. (**best paper award**)

- Kardan, S., & Conati, C. (2013). Support in an Open-Ended Learning Environment: A Pilot Study. In *AIED 2013 Workshops Proceedings Volume 2 Scaffolding in Open-Ended Learning Environments (OELEs)* (pp. 41–48). Retrieved from http://ceur-ws.org/Vol-1009/aied2013ws_volume2.pdf#page=47.
- Kardan, S., & Conati, C. (2012). Exploring Gaze Data for Determining User Learning with an Interactive Simulation. In J. Masthoff, B. Mobasher, M. Desmarais, & R. Nkambou (Eds.), *User Modeling, Adaptation, and Personalization* (Vol. 7379, pp. 126–138). Springer Berlin / Heidelberg.
- Kardan, S., & Conati, C. (2011). A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. In M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero, & J. Stamper (Eds.), Proceedings of the 4th International Conference on Educational Data Mining (pp. 159–168). Eindhoven, the Netherlands.

Section 7.6 reports some of the results published in [Fratamico, L., Conati, C., Roll, I., & Kardan, S. (2017). Applying a framework for student modeling in interactive simulations: comparing data representation granularity to handle environment complexity. *International Journal of Artificial Intelligence in Education (IJAIED)*]. After initial work on the representation of the user interactions with the target learning environment, on which the author was the lead, this project has been carried out by Fratamico for her M.Sc. thesis.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisor Dr. Cristina Conati for her support, mentoring, and guidance. I am also grateful to my supervisory committee, Dr. Alan Mackworth, Dr. Raymond Ng, and Dr. Joanna McGrenere, for their feedback and support throughout this research.

I also would like to acknowledge the funding support from the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Graphics, Animation and New Media Network of Centres of Excellence (GRAND NCE).

Finally, I wish to thank my parents for supporting me however they could, throughout my education, and my wife for her moral and intellectual support, without which, in all likelihood, this dissertation would not see the light of the day.

To my wife, Negar

and my parents,

for all of their support

Chapter 1: Introduction

1.1 Background and motivation

Advances in Human-Computer Interaction (HCI) continuously aid the creation of novel interfaces to support education and training. For example, interactive simulations are increasingly used as Exploratory Learning Environments (ELE henceforth) for learning different subjects (Frezzo, Behrens, & Mislevy, 2010; Hamza-Lup et al., 2009; Mavrikis & Gutierrez-Santos, 2010; Perkins et al., 2006; Salajan et al., 2009). These environments are designed to foster exploratory and active learning. This is done by giving students the opportunity to proactively experiment with concrete examples of concepts and processes they have learned in theory. However, it has been shown that if the students are left to experiment and explore without any additional help, the benefits of exploratory learning might be overshadowed due to (*i*) the possibility of getting overwhelmed because of the relatively large number of available options compared to step by step guided learning (Sweller, 1999) or (*ii*) lack of skills needed to explore effectively (Ploetzner, Lippitsch, Galmbacher, Heuer, & Scherrer, 2009). Hence, some students may not learn well from this relatively unstructured and open-ended form of interaction (Shute, 1993; Van Joolingen, De Jong, & Dimitrakopoulou, 2007). These students can benefit from having additional guidance when they interact with an interactive simulation (de Jong, 2006; Njoo & De Jong, 1993; Ploetzner et al., 2009). The goal of this thesis is to devise an approach that allows an ELE to provide this guidance in the form of *user-adaptive interventions* that help users explore and learn more effectively with the ELE, based on how each user interacts with the system. There are several possible ways to provide adaptive interventions described by Jameson (2009); in this work, we experimented with two forms of adaptation (*intervention types* from now on):

- *Providing explicit hints* to help the user with the current task (here, to explore a given domain with an ELE), and
- *Adapting the interface items* to reinforce the given hints when needed.

Implementing adaptive interventions requires adding two components to an ELE:

- A **user model** that determines *if* and *when* to intervene, with additional information on *which* interventions are appropriate at the time;
- An **intervention mechanism** that selects the type and content of the next intervention to be delivered to the user following an *intervention strategy* based on the assessment of the user model.

A user modeling approach that has been extensively used in the field of Intelligent Tutoring Systems (ITS) is *knowledge engineering*, i.e., to rely on domain experts to identify the relevant knowledge and behaviours to include in the model, e.g., (VanLehn, 1988, 2006). This expert-based approach has been extensively leveraged in ITS that support structured problem-solving activities, e.g., (Jackson, Olney, Graesser, & Kim, 2006; Stamper, Eagle, Barnes, & Croy, 2013; Westerfield, Mitrovic, & Billinghurst, 2013). It has, however, several disadvantages. First, it can be expensive in terms of development cost and time needed to iterate through different viable models and evaluate how well they capture various scenarios, e.g., (Beck, Stern, & Haugsjaa, 1996; Murray & Woolf, 1992). Second, it requires a clear understanding of both the application's domain, as well as the effectiveness of different user actions with respect to task performance (e.g., learning). This requirement makes using the expert-based approach difficult in ELEs, because the novelty of these systems and the more open-ended nature of the interaction they support, compared to more structured pedagogical activities, makes it difficult to judge a priori which ensemble of user interaction behaviours are conducive to better/worse task performance (e.g., learning) and should be represented in the user model (Cocea, Gutierrez-Santos, &

Magoulas, 2008). All these issues make the expert-based approach to user modeling difficult to apply to ELEs as evidenced in (Ting, Zadeh, & Chong, 2006). Furthermore, these models are built for specific domains and rely heavily on domain knowledge (i.e., ad-hoc models). Therefore, such generated models suffer from limited transferability which is another challenge that makes it even more appealing to use an alternative approach (Amershi & Conati, 2009).

An alternative method for user modeling that is less reliant on expert knowledge, faster to develop, and more cost effective is to learn the user model from data (Amershi & Conati, 2009; Baker, 2010; Baker, Corbett, & Koedinger, 2004; Beck & Woolf, 2000). This approach has become very popular in recent years contributing to the development of a new field known as Educational Data Mining (EDM) (Romero & Ventura, 2007, 2010).

A common source of user data that has been mined for building user models in EDM is interface action logs (Amershi & Conati, 2009; Bernardini & Conati, 2010; Köck & Paramythis, 2011; Mavrikis, 2010; Shanabrook, Cooper, Woolf, & Arroyo, 2010). In this thesis, we also explore the value of eye-gaze data collected with an eye-tracking device as another source for informing the user models.

Most of the focus has been on building user models for well-structured problem-solving environments with two main goals:

- Predicting/modeling user learning and/or detecting problem-solving behaviours/strategies (Köck & Paramythis, 2011; Mavrikis, 2010; Shanabrook et al., 2010) or
- Mining the solution patterns of successful problems solvers for generating automatic hints for new users (Barnes & Stamper, 2008; Barnes, Stamper, Lehman, & Croy, 2008; Fournier-Viger, Nkambou, & Mephu Nguifo, 2009)

However, using data mining in ELEs—which enable more open-ended and generally unstructured user interaction—has received less attention. Ben-Naim, Bain, & Marcus (2009) present the mined student behaviours in an interactive simulation to teachers, so that they can evaluate and change the design of the adaptive tutorials they have authored using a content development tool called Adaptive eLearning Platform (AeLP). Amershi and Conati (2009) used clustering to find students with different levels of learning performance during interaction with an ELE for learning to solve constraint satisfaction problems called the Constraint Satisfaction Problem applet (Amershi, Carenini, Conati, Mackworth, & Poole, 2008). We refer to it as the CSP applet from now on. Amershi and Conati manually analyzed the statistical features of each cluster to understand the behaviours that differ between high achieving and low achieving clusters of students on a small dataset of users. Extending this work, Bernardini and Conati (2010) described a proof-of-concept user modeling approach that uses unsupervised clustering and class association rules mining to identify relevant user types/behaviours from the same dataset, replacing the manual cluster analysis with automatic analysis of behaviour patterns in form of association rules (Bernardini & Conati, 2010).

We extend this work by proposing a comprehensive user modeling framework that defines and streamlines the process of building a user model from an initial dataset of user interaction logs with a target ELE. The user model generated by the framework classifies users based on pre-generated clusters corresponding to the groups of users that interacted similarly with the ELE, with the groups also identifying different levels of learning performance. The user model also identifies which specific interaction behaviours are responsible for the classification of a user into a given cluster. Classification and behaviour detection are performed in real-time as users are interacting with the system. These detected behaviours are then used to provide

adaptive support during the interaction. We evaluated the accuracy of this new user modeling framework on the same environment used in (Amershi & Conati, 2009; Bernardini & Conati, 2010) but on a larger dataset (110 students vs. 24), thus providing more convincing evidence on the effectiveness of this approach for building user models. More significantly, we went a step further and have proved (by means of an experimental evaluation) that there is a learning gain derived from providing adaptive interventions (driven by the model), and to a certain extent, this has shown pedagogical efficacy.

In recent years, there has been an increasing interest in eye-tracking data as an additional source of information in modeling/predicting user performance/knowledge. Conati, Aleven, and Mitrovic (2013) provide a survey of the work done on using eye-tracking information for student modeling in ITS. Most related to this thesis on the topic of eye-tracking are studies by Conati and Merten (2007) as well as Amershi and Conati (2009). Initial results on the value of eye-tracking data in user modeling for an ELE were presented in (Conati & Merten, 2007) and later in (Amershi & Conati, 2009). They looked at eye-gaze information related to the occurrence of a simple gaze pattern *defined a priori* as being relevant for learning with an interactive simulation for learning mathematical functions. However, pre-defining gaze patterns that indicate learning, may not always be easy or possible, due to the often unstructured and open-ended nature of the interaction that interactive simulations support. We extend this work by looking at a much broader range of general eye-tracking features and try to find patterns that are relevant to predicting user performance solely by using data mining techniques.

Compared to the amount of work done on building user models from user data, there has been so far very limited work on how to use these data-based user models to provide adaptive support within an ELE (i.e., an aspect that this thesis addresses via the intervention mechanism

mentioned above). Data mining approaches have been used for generating hints during structured problem-solving activities (Barnes et al., 2008; Fournier-Viger et al., 2009). Intention to use generated models/patterns for providing a personalized experience in a learning environment is stated in some works, e.g., (Köck & Paramythis, 2011; Shanabrook et al., 2010). However, to the best of our knowledge, there is no educational ELE in which both adaptive interventions and the user model activating them are generated by mining the user interaction data. This Ph.D. thesis contributes to filling this gap.

To summarize, this thesis builds on the initial ideas presented in (Amershi & Conati, 2009; Bernardini & Conati, 2010) and expands their preliminary approach in following ways:

1- We developed a comprehensive user modeling framework that can be used to generate user models for ELEs from data. Based on user interaction data, the user models classify users in terms of their learning performance and identify their prominent behaviour patterns. Additionally, the framework was evaluated in terms of classification accuracy on a much larger dataset (110 students), thus providing more convincing evidence on the effectiveness of this approach.

2- We explored the use of eye-gaze data as a source of user modeling information complimentary to action logs. Compared to the initial work by Amershi and Conati (2009), a much broader range of general eye-tracking features was used. By using these features, relevant patterns for predicting user learning were discovered solely by applying data mining techniques, as opposed to using hand-picked pre-defined patterns. Furthermore, because these features are general and not related to the specific elements of the CSP applet's interface, their usage can be easily evaluated on other learning applications and environments. The performance of the user models built with eye-gaze

data provides further evidence that eye-tracking can be used to model learning performance of students.

3- Most importantly, we extended the approach and closed the *adaptive loop*. That is, we built a user-adaptive mechanism to provide adaptive interventions during interaction with the ELE, enabling us to fully evaluate our data mining approach for providing user-adaptive support in an ELE.

1.2 Goals and objectives

The goal of our research is to devise and evaluate a framework for adding adaptive support to ELEs. Our approach relies on using data mining techniques to (*i*) cluster users into classes that correspond to different levels of learning performance and (*ii*) identify distinctive behaviours of each class. These clusters and the corresponding behaviours are then used to create a *user model* that can classify new users as they interact with the target ELE. The output of the user modeling process is predicted user performance along with the interaction behaviours that are associated with this performance. This output will be used by an *intervention mechanism* that provides adaptive support during the interaction if the detected user behaviours are associated with suboptimal learning performance. We hypothesize that an ELE with user-adaptive interventions will be more effective compared to its non-adaptive counterpart. We test this hypothesis by building and evaluating adaptive support for an existing ELE (i.e., the CSP applet[1]).

In relation to the aforementioned goal, this thesis aims to answer the following research questions:

[1] As mentioned earlier, the CSP applet is an interactive simulation designed to help students deepen their understanding of solving Constraint Satisfaction Problems. This applet is one of a collection of interactive tools for learning common Artificial Intelligence algorithms, called AIspace (Amershi, Carenini, Conati, Mackworth, & Poole, 2008).

Q1: Can data mining techniques be used to identify from the ELE interaction data, groups of users with different performance levels and their distinguishing interaction patterns?

Q2: Is it possible to use the patterns detected in [Q1] to build a classifier user model that effectively classifies new users based on their behaviours during the interaction?

Q3: Can the discovered behaviour patterns in [Q1] be used to derive adaptive interventions that are effective in improving user's learning performance?

Q4: Is the resulting adaptive version of the exploratory environment more effective than the non-adaptive version?

Q5: What is the value (if any) of eye-gaze data as another source of data for the classifier user model?

1.3 Approach

As explained earlier, this work extends an initial data mining based approach to user modeling for exploratory learning environments (ELEs) developed by (Amershi & Conati, 2009; Bernardini & Conati, 2010), by adapting the data mining techniques commonly used in Educational Data Mining (EDM) community (i.e., clustering, association rules mining, rule-based classifiers, etc.) for the purpose of providing adaptive, real-time interventions in an ELE. This essentially means covering all four steps of the cycle of applying data mining to an educational tool[2] (i.e., EDM) as defined in (Romero, Ventura, & García, 2008):

[2] Ranging from mostly static Learning Management Systems (LMS), to more dynamic problem solving tools, to ELEs supporting exploratory and open-ended interactions.

- **Data collection:** This step usually involves running appropriate user experiments for collecting user data (e.g., interface actions and eye-gaze data in the context of this thesis) while the application is in use.
- **Preprocessing:** This step includes cleaning the collected data, calculating features based on this data, and transforming the result into an appropriate format for the mining algorithm.
- **Applying data mining:** Applying data mining algorithms that build computational models which are used to discover patterns in the data (e.g., clustering, classification, association rules mining, etc.). The output of this step can be an easy-to-understand representation of these patterns or a computational model built based on the discovered pattern in the data.
- **Interpret, evaluate and deploy the results:** The final step is to interpret the information discovered by data mining and take appropriate actions with respect to the findings about the studied application. This step can be done by an offline analysis of the findings and deploying appropriate changes in the educational tool, e.g., informing the courseware designers about the effectiveness of different activities in a Learning Management System (García, Romero, Ventura, & Castro, 2009). Additionally, it can have an online component added to the studied application, allowing it to use the findings for reacting to the user actions in real-time, e.g., providing hints during problem-solving (Barnes et al., 2008; Fournier-Viger et al., 2009).

Our work is one of the first to fully evaluate the above 4-step data mining approach as applied to ELEs, by proposing the User Modeling and Adaptation (UMA) framework that provides adaptive interventions for an ELE. Figure 1-1 shows a schematic representation of the components of our framework.

As explained earlier, the previous work done on using data mining in ELE mainly focused on building a user model for predicting user's performance/knowledge level. Thus, the main evaluation reported in these works is the accuracy of the user model (Amershi & Conati, 2009; Bernardini & Conati, 2010). While the accuracy of the classifier user model is vital to providing meaningful adaptive interventions to the users, the ultimate evaluation of this approach is to create a user-adaptive version of the environment based on the patterns discovered in the process and consequently measure the effectiveness of the support provided. We provide this level of evaluation by testing our proposed approach on the ELE used as our test-bed, the CSP applet. We selected this applet as our test-bed because it is a well-established system, which has been carefully designed and thoroughly evaluated (Amershi et al., 2008), and which is regularly used in Artificial Intelligence courses offered at the Department of Computer Science at UBC and elsewhere (see Section 2.4 for more details).

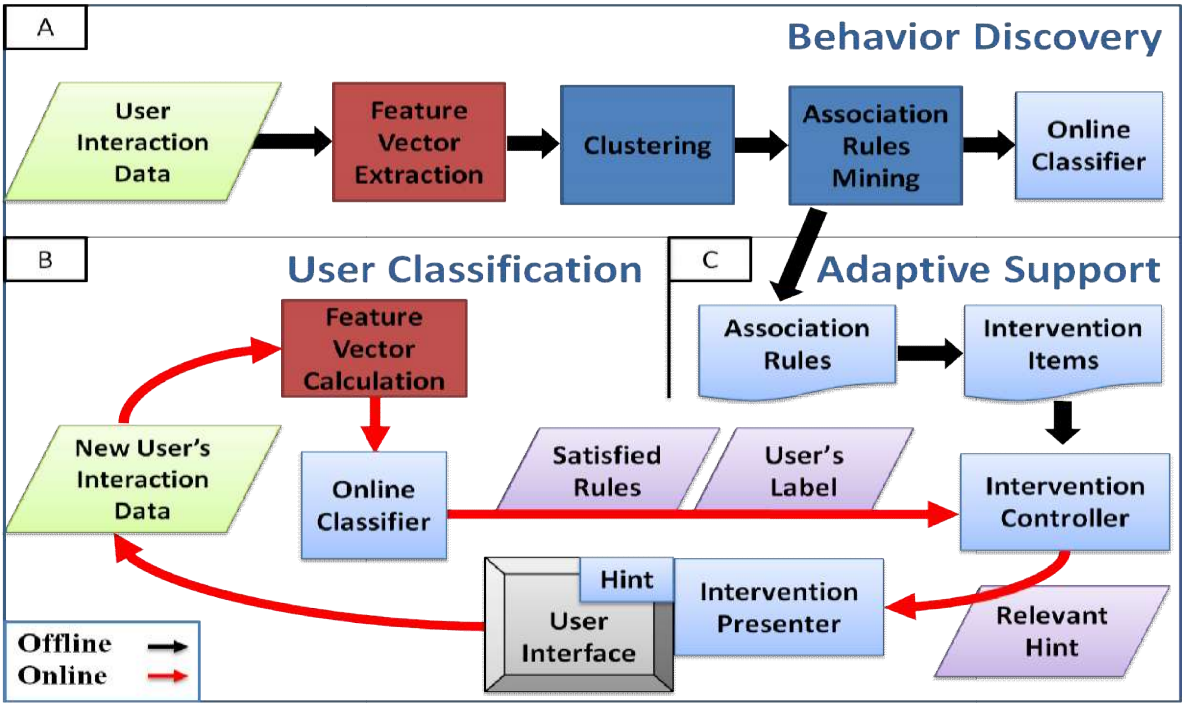


Figure 1-1 Schematic illustration of the proposed framework for providing adaptive interventions

The rest of this section briefly highlights the significant issues that were addressed in order to apply the four steps of the EDM approach for creating a new version of the CSP applet which provides adaptive support to students:

- **Data collection:** There was not an appropriate dataset available for the CSP applet for applying data mining techniques (in terms of the size of the dataset and the interaction data collected from users). Therefore, we created two new datasets for user interface actions and eye-gaze data by running two user experiments. Running user experiments was challenging and a very time-consuming process, both because we needed subjects with very specific background knowledge and because there was only one eye-tracker available. In Section 4.1 we describe the first user study where only interface actions were collected. In Section 4.3 we explain a modified version of the user study with an eye-tracker.
- **Preprocessing:** There are some existing methods for calculating features from action and eye-gaze data (Amershi & Conati, 2009; Goldberg & Helfman, 2010; Shanabrook et al., 2010; Shih, Koedinger, & Scheines, 2010) which we used as the basis of our work (as described in Section 8.2). However, for [Q5], eye-gaze data collected in a non-intrusive manner[3] by a remote eye-tracker[4] can be rather noisy[5], and the problem can be exacerbated as the sessions of collecting data gets longer (Goldberg & Helfman, 2010). Addressing this issue required devising methods for cleaning and validation of the eye-gaze data (Section 8.1 describes our solution for this issue).

[3] Not constraining the user's head movements by a chinrest

[4] Eye-trackers that track user's eyes remotely with an array of infrared sources and sensors (compare to head-mounted eye-trackers).

[5] Due to extreme head movements, objects blocking the infra red beam from reaching the eyes, subjects getting distracted, etc.

- **Applying data mining:** In our research, this step involves evaluating different data mining techniques available for (*i*) clustering student interactions and mining their behaviour patterns to find effective and ineffective interaction patterns, and (*ii*) building a user model that determines the effectiveness of user interactions during the user's interaction with an ELE. In this thesis, we addressed the following main challenges of this step:
    - We formed a data mining pipeline that includes the whole process of extracting the behaviour patterns necessary for creation and delivery of adaptive support in the CSP applet (or any other ELE) from user data. We implemented this pipeline in Python, and made the transitions between clustering, association rules mining, and building the classifier user model steps completely automated. This meant that we needed to devise mechanisms for automatically optimizing and validating the outcomes of the techniques used in each step including: finding the best parameter settings and initialization for *k*-means clustering, a method for validating the clusters, finding the best parameters for rule mining algorithm, and a weighting and aggregation scheme for the rules that form the rule-based classifier user model.

      Particularly, we had to define an appropriate multi-criteria objective function for clustering the user data. Since, we found that basic clustering is not always able to find clusters of users with different performance levels based solely on behaviour information (e.g., when clustering is based only on eye-gaze data), we defined a multi-criteria method for clustering the user data. This new method, called the hybrid approach, allows us to cluster users in a way that users who show *similar behaviours* and achieve *similar learning performance* are grouped together (see Section 3.2.2.2).

- For [Q2] we needed to build an online user classifier that can classify users with acceptable and steady accuracy after observing a limited number of their interactions: It is important that the classifier user model classifies users correctly early during the interaction, so as to provide adaptive interventions early during the interaction. In Section 3.3, we present details of our proposed rule-based classifier user model which is designed to achieve this requirement. We also provide experimental evaluations showing that the user models achieve this requirement on our datasets in Sections 5.3 and 8.6.

- **Interpret, evaluate, and deploy the results:** This thesis is among the first attempts to perform a complete evaluation of data mining approach for providing adaptive interventions in ELEs. Because of this, we used an iterative process for devising new methods for this step. The two main phases of this step are:
  - Leveraging the discovered patterns in the data mining step to define proper interventions [Q3]: we proposed an algorithm for sorting the discovered patterns in terms of their potential impact on improving users' learning performance. This ranking, combined with the way each pattern is formulated, helped the human expert to generate hint messages and other relevant interventions related to each pattern with relative ease (see Section 3.4.1).
  - Devising a strategy for delivering the interventions to users as they interact with the CSP applet in a manner that is non-intrusive and yet effective in increasing their learning: These interventions include *interface adaptations* and *providing explicit advice* for the appropriate usage of available functionalities in the interface. We developed different design alternatives and used a series of pilot studies to find the

best set of techniques for this task. See Chapter 6 for the process and Section 3.4.4 for the final design.

- Finally, a third and final user study with the adaptive version of the AIspace CSP applet was conducted, to evaluate the effectiveness of our approach quantitatively and qualitatively (see Chapter 7).

1.4 Summary of contributions

The work described in this thesis helps advance the research in educational data mining (EDM), user modeling and user-adapted interactions particularly Intelligent Tutoring Systems (ITS) in different aspects. The contributions of this thesis include:

- Being the first work in ITS area to perform an in-depth and complete evaluation of using data mining for *both* user modeling *and* providing user-adaptive interventions to support open-ended interaction with Exploratory Learning Environments (ELEs). We provide both quantitative and qualitative evaluations that show the effectiveness of our approach. The evaluations of the approach are done in terms of:
  - Performance of the generated user models (user classification accuracy).
  - Effectiveness of the adaptive interventions provided to users based on the generated user models, i.e., improvement in user's learning performance as a consequence of the provided adaptive interventions.
  - User acceptance of the adaptive interventions.
- Formal evaluation and corresponding new insights on using eye-gaze data for capturing user *learning*, which provides positive evidence for the added value of eye-gaze in predicting learning, including:
  - identifying and validating meaningful eye-gaze features

- evaluating eye-gaze data as a predictor of user learning in ELE both in terms of overall accuracy as well as performance over-time
- comparing/combining eye-gaze and action data as predictors of user learning

Additionally, several practical contributions have been made during this work. The most important ones are:

- *A new dataset containing the interface actions and eye-gaze data of 45 users interacting with an exploratory environment* (AIspace CSP) made available to the EDM research community.[6]
- *A user modeling framework for ELE with the ability to generate and evaluate the performance of user models*[7] *on different user interaction data.* It is also able to extract and rank[8] behaviour patterns that are used to generate adaptive interventions for the target ELE. In addition to the data mining functionalities provided in the framework, this framework is also able to utilize the data mining functionalities provided in the Weka data mining toolkit[9], which expands its usability for mining and analyzing user data collected from different environments.

  This framework is currently used in another research project at the Intelligent User Interfaces group of Laboratory for Computational Intelligence (LCI) at UBC (Fratamico, Conati, Roll, & Kardan, 2017). More details on this work are provided in Section 7.6.

---

[6] The actions data is available online at: http://www.cs.ubc.ca/~skardan/data/index.html. The eye-gaze data was too large to be hosted on UBC-CS server; however, it is available upon request by email.
[7] Using nested cross-validation
[8] In terms of expected effectiveness on learning performance of students
[9] Weka is a publicly available collection of machine learning algorithms used for data mining. It is available online at: http://www.cs.waikato.ac.nz/ml/weka/

- *The initial versions of the Eye Movement Data Analysis Toolkit (EMDAT) for analyzing eye-tracking data.* EMDAT calculates various eye-gaze measures from the raw eye-gaze data exported from an eye-tracker. This toolkit is currently used, maintained, and expanded by different members of the Intelligent User Interfaces group of LCI. It is publicly available and has been utilized by the research community (e.g., Intelligent Computer Tutoring Group at University of Canterbury, New Zealand).

Finally, with the ever increasing number of available interactive simulations and other ELEs and emergence of Massive Open Online Courses which partially rely on these educational tools and can act as a source of user interaction data, we see a great opportunity for data mining based approaches to providing support in ELEs. By providing a fully tested data mining based approach to providing user-adaptive support in ELEs, and backed by the very promising evaluation results obtained, this thesis contributes to promoting the application of data mining techniques for adding support mechanisms to ELEs, with the goal of improving the learning experience of future generations of students.

1.5 Thesis outline

The rest of this thesis is organized as follows. First, we review the relevant literature for this work in Chapter 2. Then we provide an overview of the proposed user modeling framework in Chapter 3. Chapter 4 describes the data collection process and the resulting datasets. Chapters 5 and 6 provide details of applying the three steps of our user modeling framework to the dataset containing user interface actions logs for the CSP applet. Chapter 5 covers *behaviour discovery* and *user classification* and Chapter 6 covers *providing adaptive support*. Chapter 7 describes the experimental evaluation of the adaptive interventions in the CSP applet. Chapter 8 focuses on

utilizing eye-gaze data for modeling user learning in the CSP applet. Chapter 9 summarizes the thesis work, highlights its contributions, and outlines areas for future research. Finally, there are two appendices included at the end of the thesis. These appendices consist of materials used in our user studies.

Chapter 2: Background and Related Work

As mentioned in Chapter 1, the work done in this thesis involves elements from Educational Data Mining (EDM), Intelligent Tutoring Systems (ITS) and User-Adaptive Interactions research areas. In this chapter, we provide an overview of the existing body of research in these areas that are relevant to this thesis.

Section 2.1 focuses on the application of EDM for modeling students in learning environments based on their actions. We start with briefly describing earlier works done in traditional environments, and then we move on to works done on exploratory learning environments which are the main focus of this section. In Section 2.2, we look into works that tried using the eye-gaze information to gain insight about users in different applications especially initial ITS works that looked into using eye-gaze to assess the students' learning. In Section 2.3 we review the research done on providing support in interactive simulations for learning including the work done on the well-defined problem-solving environments and the few instances of initial works done on providing feedback in exploratory learning environments. Finally, in Section 2.4, we describe the AIspace CSP applet, the test-bed environment used in this thesis.

2.1 Mining interface action data for modeling users in learning environments

Using machine learning and data mining techniques on the data collected in different learning environments, i.e., EDM, is a new and emerging field. Please refer to (Baker & Yacef, 2009; Desmarais & Baker, 2012; Frias-martinez, Chen, & Liu, 2006; Kotsiantis, 2012; Peña-Ayala, 2014; Romero & Ventura, 2007, 2010) for general reviews of different works done in the EDM field. In this thesis, we are especially interested in using data mining techniques for modeling users based on their interactions and understanding/analyzing their behaviours.

Using data mining for understanding user's behaviours in the learning environments dates back to the end of the 1990's with early works such as (C. Tang et al., 2000; Zaïane, Xin, & Han, 1998). The early works that suggested adding a user adaptive functionality to a learning environment using patterns discovered in the usage data mainly focused on recommending links to the items to be studied next, in well structured web-based learning content delivery environments (T. Tang & McCalla, 2005; Zaïane, 2002). After these works, the focus has shifted mainly to using data mining for building users models that capture different user behaviours in more interactive environments. For instance, different supervised[10] user models have been trained to detect misusing the help mechanism (Baker, Corbett, Koedinger, & Roll, 2005), evaluate student collaboration (Anaya & Boticario, 2011), predict the performance of students in a standard test (Trivedi, Pardos, & Heffernan, 2011) or determine the effectiveness of the student's interactions for learning (Mavrikis, 2010), all based on user action data collected from the target structured problem solving environments. More relevant to our research, in the supervised category, is the work done by Kiesmueller, et al. (2010), on Kara the programmable ladybug, which is a learning environment for teaching programming to high school students. In order to be able to adapt the programming error messages to the problem-solving strategy that the student is using, Kiesmueller, et al., conducted a user study and identified four different problem-solving strategies[11] based on their observation and then trained a Hidden Markov Model with hand annotated interactions for each strategy.

As mentioned in Chapter 1, data mining has also been used for generating automatic hints in problem-solving environments, e.g., (Barnes & Stamper, 2008; Barnes et al., 2008; Fournier-

[10] Using labeled data for training the model (e.g., labels provided by experts)
[11] The categories are Top down, Bottom up, Hill climbing, and Trial and error.

Viger et al., 2009). Barnes, et al. (2008), mined user solutions for generating Markov Decision Processes (MDPs) "that represent all student approaches to a particular problem" in a logic proof problem-solving environment. Afterward, a hint generation mechanism tries to match the current state of a new user in the problem-solving process, to an existing state in the resulting MDPs and proposes next actions s/he can take in the form of hints. A later quantitative study showed significant improvements in the number of completed problems and overall learning of the students who used a version of the environment that provided the automatically generated hints (Stamper, Eagle, Barnes, & Croy, 2011).

Fournier-Viger et al. (2009) used data mining to build a user/knowledge model for the RomanTutor, "a simulation-based tutoring system to teach astronauts how to operate Canadarm2, a 7 degrees of freedom robotic arm deployed on the International Space Station (ISS)". In the first step, they annotated each action taken in each solution generated by the users of the RomanTutor in a number of exercises. Annotations included different dimensions such as whether the solution is successful or not, the skills shown by the user so far in this solution, and expertise level of the user. After that, they used sequential pattern mining to find the frequent sequences in user solutions and then analyzed these common sequences to discover the relation between each frequent sequence *f*, and the label *l*, measured by how often *f* was labeled with *l* in the dataset (they call these relations, partial task models). These partial task models are then used to recognize a user's plan for solving a problem, estimate that user's skill level, and finally, find skills that the user is missing for solving the problem and provide hints based on them. To this date, there is no published report on the quantitative evaluation of the effectiveness of this type of hints in the RomanTutor. It should be noted that, while providing hints that suggest the next step for solving a problem can help the students, for ELEs designed for education and training,

there is not always a predefined next step available. More importantly, the goal is for the students to learn the target process from their interactions with the environment (exploratory learning) as opposed to receiving tutored problem-solving.

As explained in Chapter 1, it not always easy/possible for experts to provide labels for the patterns discovered as it is not the case that every domain is well explored similar to the problem-solving domain. As an alternative, instead of using experts for providing labels, some researchers in the EDM community used unsupervised machine learning techniques (e.g., clustering) for discovering different groups of users which show similar behaviours and then tried to understand how each of these groups relate to a certain available performance measure (e.g., learning performance). For example, Perera et al., (2009) used this approach to study online collaboration of groups of students in a collaborative software development tool (called TRAC). They applied clustering, to find groups of students with similar behaviour patterns and the distinguishing characteristics of each cluster. In addition to clustering, sequence mining was also used to find behaviour patterns where time plays a role. Based on the group performance scores that authors collected for each team, they were able to identify the discovered patterns as good or bad collaboration behaviours. Shanabrook et al., (2010) used sequence mining to discover repetitive sequences of actions (motifs) in user actions in a structured problem-solving environment (Wayang outpost). Subsequently, using expert knowledge, they coded the discovered motifs into seven[12] different *meaning groups* representing known positive/negative user behaviours.

[12] These seven groups are: Game-like, Frustration (guess), Frustration (hints), Not challenged, Too difficult, Skipping, and On-task.

As explained in Chapter 1, the most relevant works to our research are works done by Amershi and Conati (2009) and Bernardini and Conati (2010). In fact, our work builds upon the work done by these two. Amershi and Conati used clustering to find students with different levels of learning performance and manually analyzed the statistical features of each cluster to understand the behaviours that differ between high achieving and low achieving clusters of students on a small dataset of users working with the CSP applet (Amershi & Conati, 2009). Extending this work, Bernardini and Conati, described a proof-of-concept user modeling approach that uses unsupervised clustering and class association rules to identify relevant user types/behaviours from the same dataset, replacing the manual cluster analysis with an automatic presentation of behaviour patterns in form of association rules (Bernardini & Conati, 2010).

2.1.1 Association rules for finding user behaviour patterns

Association rules have been widely used for off-line analysis of learners' interaction patterns with educational software, e.g., to discover (*i*) error patterns that can help improve the teaching of SQL (Merceron & Yacef, 2003); (*ii*) similarities among exercises for algebra problem solving in terms of solution difficulty (Freyberger, Heffernan, & Ruiz, 2004); (*iii*) usage patterns relevant for revising a web-based educational system spanning a complete university course (García et al., 2009).

Most work on using association rules for on-line adaptation has been done within research on recommender systems. In (Changchien & Lu, 2001), for instance, association rules mining is used to match the user type with appropriate products. The main difference with our work is that in (Changchien & Lu, 2001) there is no on-line classification. Users are "labeled" based on clusters built off-line and the labels are used to guide recommendations when these users utilize the system. In contrast, we perform online classification of new users, with the goal of eventually

providing real-time adaptation. Similarly, associative classification is used in (Zhang & 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 in (Zhang & Jiao, 2007) needs labeled data, while ours can work with unlabeled datasets.

The work by Romero et al. (2009) is the most similar 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.

2.2 Mining eye-tracking data for modeling users in learning environments

Using eye-tracking to understand cognitive constructs such as intentions, plans or behaviour has received a lot of attention in psychology (Rayner, 1995, 1998). Researchers in human-computer interaction and intelligent interfaces also started looking at eye-gaze data as a source of information to model relevant cognitive processes of users during specific interaction tasks. For instance, eye-gaze data has been investigated to capture users' decision making processes during information search tasks (Rong-Fuh, 2010; Simola, Salojärvi, & Kojo, 2008), for activity recognition during working with a user interface (Courtemanche, Aïmeur, Dufresne, Najjar, & Mpondo, 2011), to predict word relevance in a reading task (Loboda, Brusilovsky, & Brunstein, 2011), to predict how well users process a given information visualization (Conati et al., 2011; Loboda & Brusilovsky, 2010), to estimate mental workload in relation to evaluating users' interruptibility (Iqbal, Adamczyk, Zheng, & Bailey, 2005), and to predict user and task characteristics in an information visualization (Steichen, Wu, Toker, Conati, & Carenini, 2014).

Muldner et al. (2009) looked at pupil dilation to detect relevant user affective states and meta-cognitive processes during the interaction with a learning environment that supports analogical problem-solving. Knoepfle et al. (2009) used eye-tracking data for comparing existing theories of how users learn to play strategies in normal-form games. The theories were compared in terms of how they could predict users' moves and attention to relevant information during interaction with a computer card game, with all theories showing limited predictive power.

In our work, we are interested in investigating whether a user's gaze patterns during interaction with an interactive simulation can be used to assess if s/he is learning. We were inspired by existing research showing that it is possible to identify distinctive patterns in the eye-gaze data of successful vs. unsuccessful users during simple problem solving and question answering tasks (Canham & Hegarty, 2010; Hegarty, Mayer, & Monk, 1995; Jarodzka, Scheiter, Gerjets, & van Gog, 2010; Tsai, Hou, Lai, Liu, & Yang, 2012). In this body of work, the attention patterns analyzed related mainly to processing the problem description (Hegarty et al., 1995) or supporting visual material (Canham & Hegarty, 2010; Jarodzka et al., 2010; Tsai et al., 2012). The main finding was that successful problem solvers pay more attention to information relevant to answer correctly, while unsuccessful problem solvers show more scattered attention patterns. Eivazi and Bednarik (2011) went a step further showing that it is possible to build a classifier that relies solely on eye-gaze data to predict users' performance during an interactive 8-tile puzzle game. Conati and Merten (2007) and Amershi and Conati (2009) present results that are even more relevant for our work, since they also looked at eye-gaze data to model student reasoning and learning during interaction with an interactive simulation. As explained earlier, the student models in (Amershi & Conati, 2009; Conati & Merten, 2007) combine simple gaze-pattern information with information on the user's interface actions, whereas in this thesis we

focus on eye-gaze data, in a broader and more generalizable manner, to better isolate its potential as a source of information for user modeling in interactive simulation.

2.3 Support in interactive simulations for learning

Providing adaptive support to students is an important part of Intelligent Tutoring Systems (ITS), where this support is intended to partially replace the guidance a student would receive in a one-to-one tutoring setting (VanLehn, 2006).

There is ample evidence that adaptive support can be beneficial in ITSs that target problem solving and other pedagogical activities where there is a well-defined set of solutions/behaviours that the ITS can target. For example, Westerfield et al. (2013) showed the effectiveness of personalized feedback on errors the students make in an ITS that provides step-by-step training on assembling a computer motherboard. Stamper et al., (2013) used a data mining approach to generate automatic solicited hints for an algebra tutor, and showed that the tutor performed better compared to a non-adaptive version in terms of both number of problems solved and overall learning. AutoTutor, an ITS that provides adaptive feedback on answers to physics questions, was successfully evaluated in (Jackson et al., 2006). It should be noted that this ITS includes a simulation that helps find the answers but no support is given on how to use it. It has also been shown that providing real-time feedback on students' help-seeking behaviour, plays a role in self-regulated and active learning (Aleven, Roll, McLaren, & Koedinger, 2016).

On the other hand, providing adaptive support for Exploratory Learning Environments (ELE) is in the early stages of research (Mavrikis, Gutierrez-Santos, Geraniou, & Noss, 2012). One defining characteristic of these environments is that there is usually no definition of correct behaviours. Students can explore the environment as they like. This makes it difficult to judge a priori and track which ensemble of user interaction behaviours should be the target of adaptive

support (Ting et al., 2006). Most of the work done so far on designing and evaluating adaptive feedback for learning environments that include interactive simulations has dealt with the challenge by limiting the exploratory nature of the interaction. For instance, the simulations developed by Johnson (Hussain et al., 2009) provide feedback on how to behave in pre-defined cultural/language-related scenarios with a clear definition of correct answers/behaviours. CTAT-VLab, (Borek, McLaren, Karabinos, & Yaron, 2009) provides help on well-defined steps required to run a scientific experiment. Science Assistments, (Gobert, Montalvo, Toto, Sao Pedro, & Baker, 2010) provides feedback on the specific problem of controlling for variables in an experimental design. Thus, although (Hussain et al., 2009), (Borek et al., 2009) and (Gobert et al., 2010) add some form of adaptive feedback to interactive simulations, the feedback targets pre-defined behaviours that help students find correct answer/solutions in the instructional domain. In contrast, in our work, we designed and evaluated adaptive support for a more open-ended exploratory interaction, with no prior definition for correct actions/solutions. The support aims to help students use the simulation to explore the underlying algorithm effectively, not find a correct answer/solution.

The work closest in nature to ours is eXpresser, a simulation environment for learning algebra, which leverages a set of predefined feedback strategies to provide both unsolicited and solicited feedback to students based on their interactions with the tool (Mavrikis et al., 2012). A preliminary qualitative assessment of this feedback resulted in modest positive ratings for student perception, but no quantitative study has been published so far (Mavrikis et al., 2012). Thus, our work is the first (to the best of our knowledge) to provide a rigorous evaluation of an adaptive support mechanism for open-ended exploration in an educational simulation. Moreover, while in eXpresser a significant amount of manpower has been spent on defining the behaviours to be

tracked in student model and the feedback strategies, in our work both adaptive interventions and the user model activating them are generated by mining the user interaction data with minimal human involvement.

There has been some work regarding how to design hints and recommendations in ITSs to ensure that they make the greatest impact on the user, e.g., (Santos & Boticario, 2015; Tempelaar, Rienties, & Giesbers, 2015; Walker, Rummel, & Koedinger, 2014). In this thesis, we followed an iterative design and evaluation process (Dix, 2009) with an informal focus group and 3 pilot studies to ensure that our support mechanism is effective and provides the basic functionality expected by the users. As a reference for different methods to draw the user's attention, we used (Gluck, Bunt, & McGrenere, 2007).

2.4 Test-bed: the AIspace CSP applet

The Constraint Satisfaction Problem (CSP) Applet is one of a collection of interactive tools for learning common Artificial Intelligence algorithms, called AIspace (Amershi et al., 2008). Algorithm dynamics are demonstrated via 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 (Poole & Mackworth, 2010). Solving a CSP requires finding an assignment that satisfies all constraints. The CSP applet[13] illustrates the Arc Consistency 3 (AC-3) algorithm for solving CSPs represented as networks of variable nodes and constraint arcs. 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

[13] Available at http://www.aispace.org/constraint/index.shtml

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 for the interactive execution of the AC-3 algorithm on a set of available CSP problems. These mechanisms are accessible through the toolbar shown at the top of Figure 2-1 or through direct manipulation of graph elements.

In particular, the user can:

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 (i.e., Fine Step (FS) action). Figure 2-2 shows one instance of fine stepping through the CSP. In Figure 2-2a, AC-3 selects the blue arc representing the constraint that the value selected for B should be greater than the value selected for E. In Figure 2-2b, this arc is turned red 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 2-2c, the inconsistent value of 1 is removed from the domain of variable B;

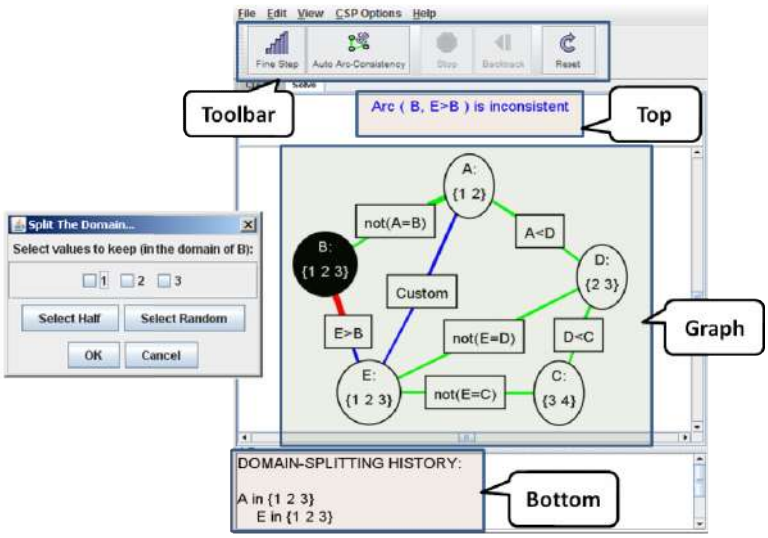


Figure 2-1 The CSP applet with an example CSP problem

2. Directly click on an arc to apply all these steps at once (i.e., Direct Arc Click (DAC) action);
3. Automatically fine step through the completion of the problem using the *Auto Arc Consistency* button (i.e., Auto AC (AAC) action);
4. Pause auto arc consistency using the *Stop* button (i.e., Stop action);
5. Select a variable to split on, and specify a subset of its values for further application of AC-3 (i.e., Domain Split (DS) action). Figure 2-3a shows the domain splitting dialog for the variable D for the CSP shown in Figure 2-2c. Domain splitting is being applied once all the arcs have been made consistent and only 3 and 4 remain in D's domain. After 3 gets selected in the dialogue box, a new CSP is generated with that value for D (Figure 2-3b);

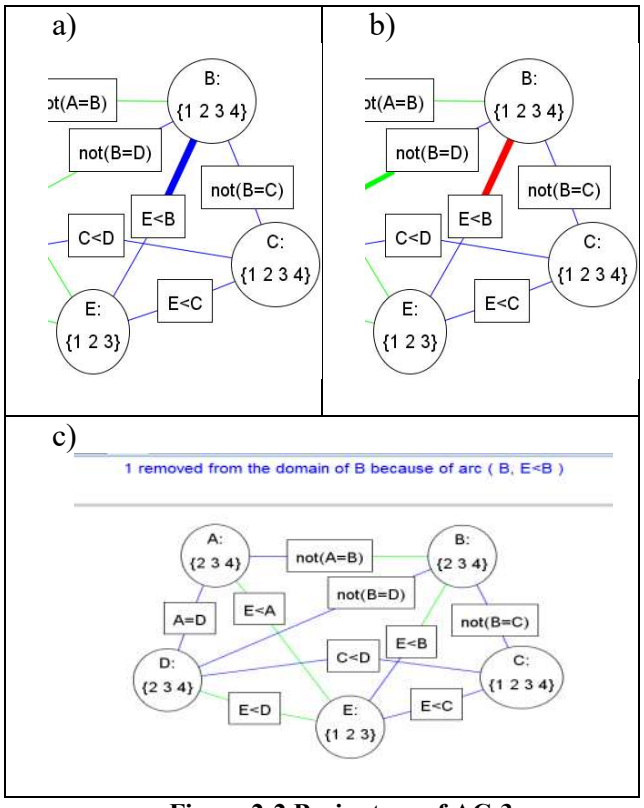


Figure 2-2 Basic steps of AC-3

6. Retrieve alternative sub-networks generated by domain splitting using the *Backtrack* button (i.e., Backtrack (BT) action). Continuing the example shown in Figure 2-3, via 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 2-3c);
7. Return the graph to its initial status using the *Reset* button (i.e., Reset action).

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

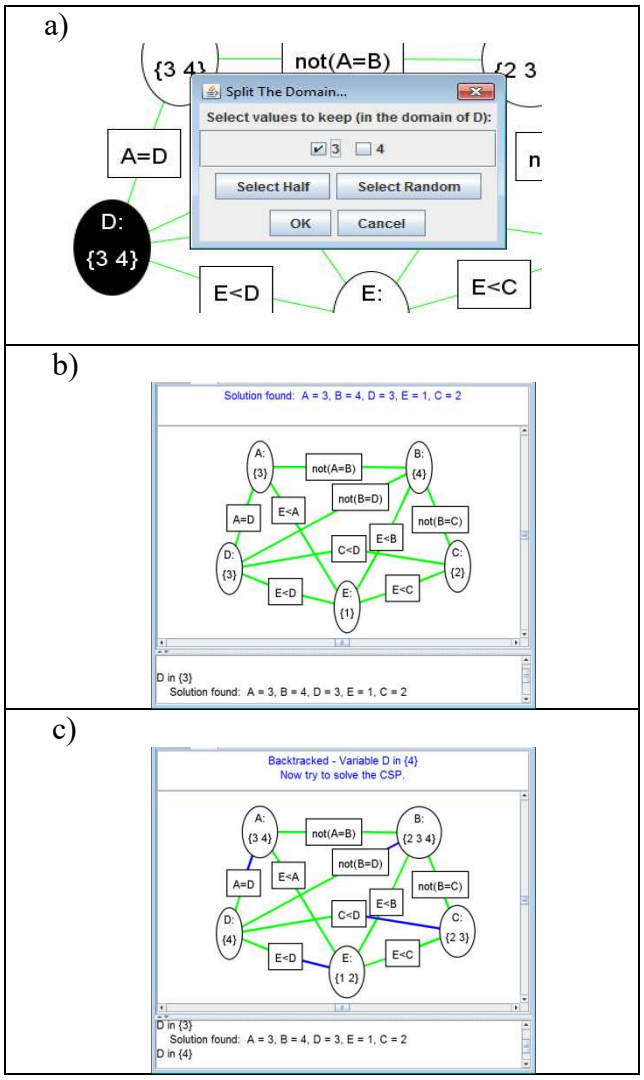


Figure 2-3 Domain-splitting and Backtracking

The original CSP applet did not provide any explicit support to help students learn more effectively from the mechanisms described above. Research, however, shows that students may benefit from this support since unaided exploration of interactive simulations often fails to help students learn (Shute, 1993). As mentioned in Chapter 1, we used the CSP applet to evaluate our proposed User Modeling and Adaptation framework and added a support mechanism to the CSP applet described in Chapter 6.

Chapter 3: Overview of the Proposed User Modeling and Adaptation Framework

As explained earlier, the main motivation for this thesis is based on the idea that students may benefit from having additional guidance when they interact with an Exploratory Learning Environment (ELE). This chapter describes the proposed User Modeling and Adaptation (UMA) Framework which aims to provide this guidance adaptively, based on the real-time evaluation of whether a user's interaction behaviour with an ELE is conducive to learning or not.

3.1 An overview of the UMA framework

Our user modeling approach consists of three major phases: *Behaviour Discovery* (Figure 3-1A), *User Classification* (Figure 3-1B), and *Adaptive Support* (Figure 3-1C).

In the behaviour discovery phase (Figure 3-1A), data from existing interaction logs is preprocessed into feature vectors where features consist of statistical measures that summarize the user's interaction with an ELE (e.g., action frequencies, the time interval between actions, and gaze patterns). Each vector summarizes the behaviours of one user. A clustering algorithm then groups these vectors according to their similarities, thus identifying users who interact similarly with the interface. Next, association rules mining is applied to each cluster to extract its common behaviour patterns, i.e., rules in form of X → *c*, where X is a set of feature-value tuples and *c* is the predicted cluster for the data-points to which X applies. Clusters are then analyzed to identify how they relate to student learning performance. Thus, the behaviour discovery phase generates groups of users who are associated with different levels of learning performance, as well as sets of interaction behaviours representative of each group.

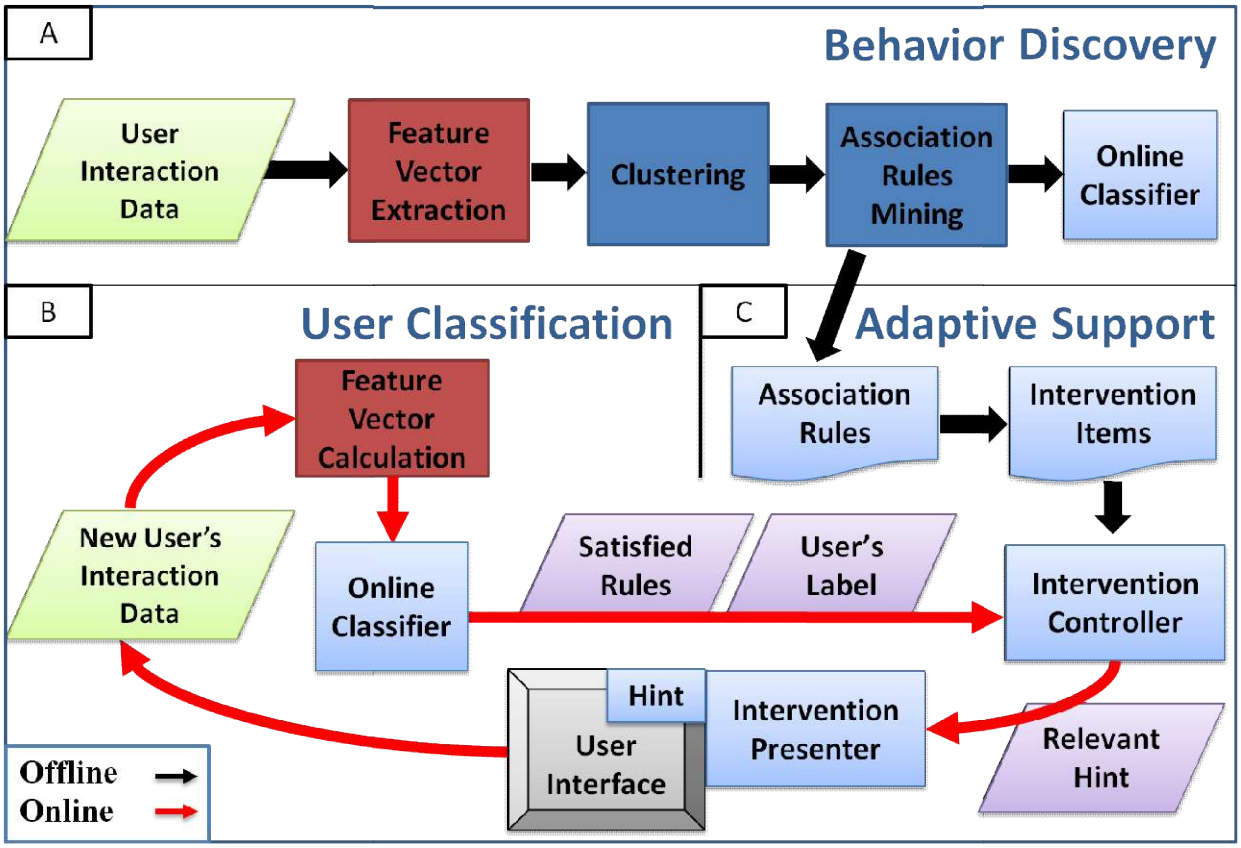


Figure 3-1 Schematic illustration of the proposed User Modeling and Adaptation Framework

Understanding the effectiveness of a user's interaction behaviours is useful in itself for revealing to developers how the application can be improved (Hunt & Madhyastha, 2005). However, we also want to use these behaviours to guide automated adaptive support during the interaction. Thus, the clusters and association rules identified in the behaviour discovery phase are used to build an online classifier user model in the next phase of the framework.

The user classification phase (Figure 3-1B) uses the clusters and class association rules extracted in the behaviour discovery phase to build an online classifier user model. This classifier assesses in real-time the (possibly evolving) learning performance of a new user by (*i*) incrementally building a feature vector based on the interaction events seen so far; (*ii*) classifying

this vector in one of the available clusters. Note that the classification can change over time depending on the evolution of the user's interaction behaviours.

The classifier generated in user classification phase (Figure 3-1B) is used to assess the performance of a new user based on her interaction behaviours in real-time. This assessment then guides the delivery of adaptive support in the last phase of the framework.

The goal of the adaptive support phase (Figure 3-1C) is reinforcing the effective behaviours discovered in the behaviour discovery phase and discouraging the ineffective ones. The adaptive support relies on the class assigned to a user interacting with the ELE by the classifier user model, as well as the satisfied association rules causing that classification decision (Figure 3-1C) to decide which intervention to provide and when. The intervention mechanism consists of two components: *intervention controller* and *intervention presenter* (Figure 3-1C). Intervention controller selects the next intervention to be given and intervention presenter delivers a correspondingly appropriate hint message (or any other type of suitable intervention) to the user.

The rest of this chapter is organized as follows. Section 3.2 describes the components of the behaviour discovery phase. Section 3.3 discusses the details of the user classification phase. Section 3.4 provides an overview of the objectives, guidelines, and algorithms used in the adaptive support phase. Finally, Section 3.5 provides a summary.

3.2 UMA framework: behaviour discovery

In this section, we describe different components of the behaviour discovery phase, namely: data extraction, clustering, and association rules mining.

3.2.1 Data extraction

The first step in behaviour discovery phase is to create a set of data-points from user interaction data. For instance, in the datasets used in this thesis, data-points are vectors of

features consisting of statistical measures that summarize the user's interactions with the interface based on both interface actions (e.g., action frequencies, time interval between actions, etc.) and eye-gaze data (e.g., fixation rate, mean of fixation duration, etc.).

Handling noise in the interaction data is also part of the data extraction step. For example, eye-tracking data can be rather noisy, especially if collected in a setting that does not constrain the user's head. Details on our proposed procedures for clean-up and noise reduction in eye-gaze data are provided in Section 8.1.

3.2.2 User clustering and the hybrid approach

As mentioned before, we are interested in finding groups of students who behave in a similar way, and we employ clustering to achieve this goal. Our first choice for clustering was the *k*-means algorithm (Bishop, 2007), which is a common algorithm in EDM research (Baker & Yacef, 2009). The simplicity of the *k*-means algorithm, its effectiveness, and acceptable efficiency makes it a popular choice. To refine the clustering step, we experimented[14] with other clustering algorithms available in the Weka data mining package (Hall et al., 2009), including Hierarchical Clustering and Expectation Maximization (Bishop, 2007). However, similar to Perera et al. (2009), none of these alternatives produced substantially different outcomes in terms of clusters in our datasets (described in Chapter 4). We thus decided to retain *k*-means as the clustering algorithm for our approach, but devised a method to ensure faster convergence to a good set of clusters.

One of the issues when using the *k*-means algorithm is setting good initial centroids, so that the algorithm can quickly converge to a stable set of clusters with a small within-cluster error.

[14] Using the CSP-Action-65 dataset described in Chapter 4

However, the implementation available in Weka tended to converge slowly on our datasets. We thus experimented with Genetic Algorithms (GA) to initialize the centroids for *k*-means based on an approach suggested in (Kim & Ahn, 2008). This approach relies on using "chromosomes" to mold initial cluster centroids as needed. These chromosomes represent different initial values for each feature of the initial centroids. Through mutation and crossover, in each iteration, new initial centroids are generated and the ones with lower corresponding within-cluster error for the resultant clusters are retained for the next iteration.

The method proposed by Kim and Ahn (2008) primarily deals with binary features and proposes discretizing continuous variables and using multiple-bit genes to represent these variables (14 bits per feature). The user modeling tasks targeted by our framework typically need to handle a substantial number of continuous features. For instance, there are at least 21 continuous features in the datasets used in this thesis; therefore, the method proposed by Kim and Ahn (2008) is inefficient because it requires chromosomes with too many extra bits to discretize the features without major loss of information (e.g., with 21 continuous features we need $21 \times 14 = 294$ bits for each chromosome). In general, with m continuous features and k clusters, $14mk$ bits are needed following the method proposed by Kim and Ahn (2008).

We thus changed the method for building the chromosomes to make it more suitable for datasets with a high number of continuous features. Instead of each chromosome representing the initial value of the centroids, in our version, each chromosome represents one permutation of membership assignment of data-points to clusters (e.g., data-point 1 belongs to cluster 1, data-point 2 belongs to cluster *k*, etc., where *k* is the desired number of clusters). Then, each chromosome is used to generate a set of centroids that initialize a different run of *k*-means. In

this setting, each chromosome needs $\lceil log_2(k^n) \rceil = \lceil n\, log_2(k) \rceil$ bits where k is the number of desired clusters and n is the number of data-points in the dataset.

Given the above calculations, we can determine that when the ratio of number of data-points over continuous features (i.e., n/m) is less than $14k/log_2\,(k)$, our approach would definitely result in smaller chromosomes and less computation. Also, knowing that $k \geq 2$, the lower bound for this ratio is 28, which is a relatively high ratio for the existing datasets in the ELE community. For ratios above that, the more efficient method is the one proposed by Kim and Ahn (2008).

Given that in our datasets k was expected to be less than or equal to 5 and n is at most 110 while m is at least 21, our proposed approach is clearly more efficient, e.g., a maximum of $\lceil \log_2(5^{110}) \rceil = 256$ bits are needed to represent one complete cluster-set in our approach which is much smaller than $14mk = 14 \times 21 \times 5 = 1470$ bits needed for k chromosomes generated following the method proposed by Kim and Ahn (2008). More interestingly, we do not have to discretize the features and there is no need to handle multi-bit chromosomes during the crossover and mutation actions (described below).

The proposed approach uses the following steps. We generate a random population of 100 initial chromosomes (where each chromosome represents one permutation of membership assignment of data-points to clusters), each used to generate a set of centroids that initialize a different run of *k*-means. We then select the half of the chromosomes that led to clusters with the lowest within-cluster error and use these to generate the next generation by crossover (i.e., selecting two chromosomes and choosing the upper half bits of one chromosome and the lower half of the other chromosome to form a new one) and mutation (i.e., selecting a chromosome and

randomly changing one of its bits). We repeat this process until there is no improvement for a certain number of generations or we reach the maximum number of iteration limit.

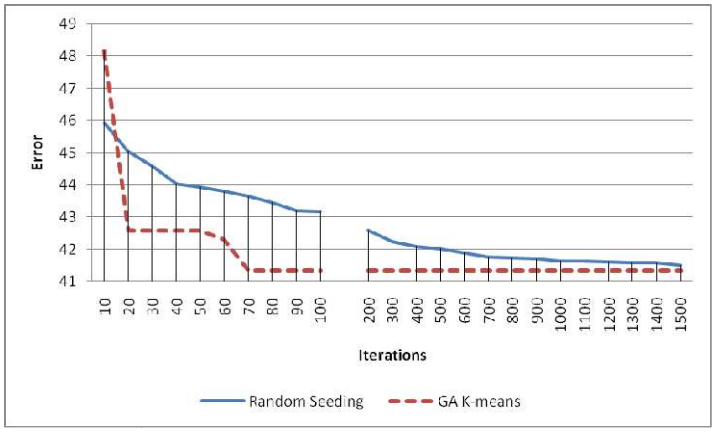


Figure 3-2 Convergence of within-cluster error for GA *k*-means compared to the standard Random Seeding of *k*-means (for *k* = 2) on the CSP-Action-65 dataset

Our experimental results show that, although this approach does not guarantee finding the global minimum for the within-cluster error, it converges faster than the standard random seeding method. For instance, Figure 3-2 shows a comparison between the performances of GA *k*-means and the *k*-means from Weka on one of our datasets (CSP-Action-65, see Chapter 4) averaged over 30 different runs. GA *k*-means converges after 70 iterations, while the standard seeding method does not reach that same error level even after 1500 iterations (here, iterations are the number of times that the basic *k*-means is used for both cases).

To determine the optimal number of clusters (i.e., *k*) we use the minimum *k* suggested by the following measures: C-index, Calinski and Harabasz (Milligan & Cooper, 1985), and Silhouettes (Rousseeuw, 1987). These are the best performing measures of clustering validity in the literature (Milligan & Cooper, 1985; Rousseeuw, 1987) in terms of determining the true number of clusters in the data.

3.2.2.1 Clusters and learning performance

In order to associate behaviours with learning performance, it is first necessary to establish how the user groups generated by clustering relate to learning. This can be done in different ways, depending on whether information on the users' learning performance is available or not:

- If learning performance measures are not available, we face an unsupervised learning problem. In this case, clustering is done using GA *k*-means. It is then left to the judgment of a human expert to evaluate how each cluster and associated behaviours may relate to learning. Since we have access to a learning performance measure, this case is not considered in this work.
- If learning performance measures are available, one possible approach is to generate the clusters solely based on interaction data, and then assign a label for each cluster by comparing the average learning performance of the users in that cluster with the performance of the users in the other clusters. This is the approach we successfully adopted for interface actions data for the CSP applet described in Chapter 5 (called the *basic clustering* from now on). It is possible, however, that clustering solely based on behaviours does not generate groups with a clear (i.e., statistically significant) difference in learning performance, making it difficult to assign labels to the clusters automatically. To tackle this situation, we propose a solution that leverages user performance data to guide the clustering process, thus creating a hybrid approach.
- If learning performance measures are available, the conventional method for creating a training set of labeled classes is to divide the performance spectrum into different ranges and putting users within each range into one group (e.g., median split for creating 2 classes). The number of classes is an arbitrary choice, and data-points at the boundaries are also somewhat

arbitrarily forced into one class. When grouping users together, the hybrid approach (see Section 3.2.2.2) relies on both learning performance as well as the similarity in user interaction data as opposed to only relying on learning performance. Thus, we argue that it can generate better performing user models since the user models can only rely on user interaction data when classifying users. This hypothesis was confirmed in our experimental evaluations presented in Chapter 8.

3.2.2.2 The hybrid approach

As mentioned earlier, following the basic clustering approach, without a clear (i.e., statistically significant) difference in average learning performance of different clusters, it is difficult to assign labels to the clusters found. This issue arose, for instance, when we experimented with the eye-gaze interaction data (see Chapter 8 for details). In the rest of this section, we present our solution for this issue.

In our framework, the only requirement for interpretability of the clusters is that there should be a significant difference between the average learning performances of members in different clusters, as measured by an appropriate statistical test. In other words, since we know the users in each cluster behave similarly, just knowing that the members of a cluster achieve significantly higher/lower average performance than the other clusters is enough to interpret salient behaviours observed in that cluster as effective/ineffective. Based on this requirement, we propose the hybrid approach. The hybrid approach finds the best cluster-set (in terms of the sum of within-cluster distances) with a significant difference in learning performance[15]. This is achieved by following the search for best cluster-set similar to GA *k*-means, and keeping track of

[15] The measure of learning performance used in this thesis is Proportional Learning Gain (PLG), i.e., the ratio of the difference between post-test and pre-test, over the maximum possible gain; described in percentage ratio (see Section 4.4 for details).

the best cluster-set visited, for which the learning performance difference is statistically significant. It should be noted that using the hybrid approach on datasets where basic clustering produces statistically significant clusters (e.g., interface actions data in our case) would result in finding the same cluster-set. This feature enables us to replace the basic clustering approach with the hybrid approach for all cases.[16]

3.2.3 Association rules mining to describe user behaviours

In our user modeling framework, association rules mining is used to identify the interaction behaviours that characterize each of the clusters found in the clustering phase. We use the Hotspot algorithm (Hall et al., 2009) to perform association rules mining on our clusters. Hotspot inspects the training data and generates the association rules corresponding to a class label (a specific cluster, in our case) in form of a tree. For instance, two sample generic rules derived from the same tree branching could be as follows:

If Action A frequency = High → Cluster X	If Action A frequency = High **and** Action B frequency = Low → Cluster X

The algorithm has three parameters that influence the type and number of rules generated: (*i*) the minimum level of support requested for a rule to be considered relevant (where support for rule X → Y is defined as the percentage of data-points satisfying both X and Y in the dataset); (*ii*) the tree branching factor, influencing how many new rules can be generated from an existing one by adding a new condition; (*iii*) the minimum improvement in confidence needed for creating a new tree branch (where confidence for rule X → Y is the probability that Y occurs when X does). Essentially, the goal is to find a few rules that characterize as many elements in

[16] More details on the hybrid approach is provided in Section 8.4.3

the cluster as possible and provide an easily understandable explanation of users' behaviours for each cluster.

Given the tree-like structure of the rules, in UMA framework we filter out rules such that, when there is a set of rules derived from the same tree branching, rules closer to the root and with low confidence are discarded. The rationale behind this choice is that rules with low confidence include interaction behaviours that are not representative of a specific cluster (i.e., these behaviours are observed in more than one cluster), and thus they tend to weaken the classification ability of the rule set as a whole (more detail on this point is provided in Section 3.3).

Class association rules mining algorithms generally work with both discrete and continuous values. The attributes that describe the user interaction behaviours in our user modeling tasks are continuous, but they need to be discretized, otherwise, they would produce a large number of very fine-grained rules that are unsuitable for classification. Choosing the appropriate number of bins for feature discretization involves a trade-off between information loss (having too few bins) and generating overly specific rules that are too detailed to capture meaningful patterns (having too many bins).

The UMA framework applies a grid search for parameter exploration to find the optimal parameter setting to be used for the Hotspot algorithm (i.e., minimum support, branching factor, and minimum improvement of confidence), as well as the optimal number of bins for the discretization. We used nested cross-validation for parameter optimization. The parameter values that produce the best results in the inner loop in most folds, as measured by the accuracy of the classifier user model (described in the next section), are selected.

Additionally, we are only interested in rules that apply exclusively to one class of users (representative rules), therefore only rules that have a confidence value greater than 50% are selected.

3.3 UMA framework: user classification

In the user classification phase, as new users interact with the system, they are classified in real-time into one of the clusters generated by the behaviour discovery phase, based on which association rules match their behaviours. The use of association rules to construct a classifier is called Associative Classification Mining or Associative Classification (Thabtah, 2007). Algorithms for Associative Classification usually generate a complete set of class association rules (CARs) from the training data and then prune this initial set to obtain a subset of rules that constitute the classifier. When a new unknown object (a new user in our case) is presented to the classifier, it is compared to a number of CARs and its class is predicted based on a measure that summarizes how well the user matches the CARs for each class. One simple scheme to use CARs for classification is to count the number of CARs satisfied for each cluster. This means that all rules are considered equally important for classification**,** failing to account for the fact that some rules with limited class support (i.e., applicable to a fewer number of members in that class compared to others) should be considered with caution when deciding the class label of a user. Given this limitation, we decided to apply a more sophisticated classification scheme that assigns a value to each rule, and calculates class membership scores based on the values of the satisfied rules that apply to a class (Han, 2003). We used a variant of this approach where instead of calculating membership scores based only on the satisfied rules, all of the CARs that represent a cluster are used. The rationale behind this choice is that in our user modeling task the rules that do not apply to a new user are also important for determining the final label. For instance, it is

important to penalize the score of class *c* when a major rule (which applies to most of the *c*'s members) is not satisfied for the new user, even if a less distinctive rule for *c* applies to her. Accordingly, the membership function we adopted returns a score S_A for a given class *A* as follows:

$$\mathbf{S_A} = \frac{\sum_{\mathbf{i=1}}^{\mathbf{m}} \mathbf{Test(r_i)} \times \mathbf{W_{r_i}}}{\sum_{\mathbf{i=1}}^{\mathbf{m}} \mathbf{W_{r_i}}}, \qquad \mathbf{Test(r_i)} = \begin{cases} \mathbf{1} & \mathbf{if\ r_i\ is\ satisfied} \\ \mathbf{0} & \mathbf{otherwise} \end{cases} \qquad \textbf{(3-1)}$$

where r_i's are the *m* rules selected as representatives for class *A*, W_{r_i} is the corresponding rule weight (based on a measure explained below), and $Test(r_i)$ is an indicator function that checks the applicability of a rule to a given instance. The class with the highest score would be assigned as the predicted label for the user. We tried different measures from the literature to define W_{r_i} (Geng & Hamilton, 2006) (including confidence, support, conviction, and leverage) and found confidence to be the measure that generates the best classification accuracy across our datasets.

3.4 UMA framework: adaptive support

In addition to classifying the user in one of the available clusters, the CSP's user model also returns the satisfied association rules causing that classification decision (Figure 3-1C). These rules represent the distinctive interaction behaviours of a specific user so far, from the set of previously identified rules (in the behaviour discovery phase) for each cluster. These clusters are associated with high (effective) or low (ineffective) learning performance. The goal of the adaptive support component of the framework is to design interventions that would specifically target these behaviours (i.e., reinforcing the effective behaviours and discouraging the ineffective ones).

3.4.1 Deriving interventions from association rules

The first step for building the intervention mechanism is to process the association rules discovered in the behaviour discovery phase and derive representative interventions from those rules. As mentioned before, our framework relies on finding clusters of users that achieve significantly different learning performances (higher learning gain or HLG vs. lower learning gain or LLG). The number of HLG or LLG clusters may be more than one, and there might be clusters with average learning performance which do not differ from the HLG or LLG clusters. However, following the hybrid approach, we will find at least one HLG and one LLG cluster if such clusters exist in the data. Based on this premise, we can use the association rules discovered for the HLG cluster(s) and the underlying interaction patterns to devise interventions that encourage users to interact more effectively; similarly, we can leverage the association rules discovered for the LLG cluster(s) and the underlying patterns to devise interventions that discourage users from interacting in an ineffective manner.

To demonstrate the process, let us assume the behaviour discovery phase for the CSP applet has produced 2 clusters, one cluster with higher learning gain (HLG) and another one with lower learning gain (LLG). Each cluster will have a set of representative association rules (i.e., rules that are exclusive to that cluster). Figure 3-3 shows a sample rule for the LLG cluster.

Auto AC frequency = High[17] **and** Direct Arc Click frequency = Low[17] → Cluster LLG

Figure 3-3 A sample class association rule

This rule indicates that members of the LLG cluster (i.e., low learners) show very high frequency of *Auto AC* actions, while rarely using the *Direct Arc Click* (DAC) action[18]. A

[17] As described in Section 3.2.3, values for the features are discretized for the association rules mining. "High" and "Low" here represent the highest and lowest bins for the respective feature.

possible explanation of why both these behaviours are associated with limited learning is that they identify users who are not very engaged in the exploration process because they prefer to (*i*) run the algorithm to completion instead of stepping through it; (*ii*) leave to the applet the selection of the next arc to work on, rather than being proactive in choosing it.

Thus, this rule identifies two possible interventions (*intervention items* from now on) to help students correct these suboptimal behaviours exhibited by low learners: (*i*) discourage excessive use of the Auto AC; (*ii*) encourage higher usage of DAC. At this point, the involvement of a human designer is needed to decide how these intervention items should be implemented. For example, if the interventions are hint messages, the text for the messages is decided by the human designer. In some cases, the human designer might even decide that an intervention is only relevant at a certain stage of interaction or when other environmental factors are present. This additional information can also be present in the rules if captured during feature extraction (see Sections 7.6 and 9.2).

While one can try to implement all intervention items that can be extracted from the discovered rules, we are interested in devising a strategy for ranking and selecting the rules (and by extension, the intervention items) based on a weighting scheme (described in the next section) that captures their prominence. This is useful when there are limited resources and the designer has to focus on a subset of intervention items.

3.4.1.1 Offline ranking process for intervention items

In this section, we present a ranking strategy for association rules and intervention items. This ranking is performed offline and only once for each ELE. It takes as input the association rules for

[18] See Section 2.4 for the detailed description of these actions.

the LLG and HLG clusters from the behaviour discovery phase. During this process, the weight for each association rule is calculated and patterns are filtered and presented to the human expert for generation of the intervention items.

First, we define the weight ω_r[19] for each rule r as:

$$\omega_r = Conf(r) \times \frac{ClassCov(r)}{|preconditions(r)|}$$

where $ClassCov(r)$ is the class coverage for rule r and is defined as the ratio of the total number of users in the class that rule r applies to over the total number of users in that class; $|preconditions(r)|$ is the number of preconditions for the rule r; and $Conf(r)$ is confidence of rule r as defined in Section 3.2.3.

Then, depending on the desired number of interventions (i.e., depending on implementation time that the designer is willing to invest) the lowest ranking rules may be discarded from the rest of the offline process.

Next, each precondition of each remaining rule (e.g., Direct Arc Click frequency = Low and Auto AC frequency = High in our sample rule in Figure 3-3), is listed. Preconditions may appear in rules of more than one cluster[20]. If a precondition appears in more than one cluster, it is discarded because it indicates a weak association of that pattern to any of these clusters. In other words, the pattern (a.k.a., precondition) applies indiscriminately to members of more than one cluster when taken alone, and is only useful when it is combined with other patterns[21]. Finally, an intervention item is designed by the human expert for each remaining precondition. The

[19] Not to be confused with W_{r_i} in equation (3-1)

[20] Note that rules for each cluster are guaranteed to be unique due to the requirement that their confidence must be above 50%. However, this does not guarantee that individual preconditions of rules are also unique.

[21] One possible way to use this kind of patterns is to generate interventions based on a combination of patterns; however that would make the interventions very complicated. Therefore, we decided to limit the interventions to only one pattern.

human expert is aided greatly by the fact that the preconditions describe only one pattern and are associated with high or low learning performance.

3.4.2 Intervention mechanism

The intervention mechanism in the UMA framework has two components: intervention controller and intervention presenter (Figure 3-4). Intervention controller selects the next intervention to be given and intervention presenter delivers the respective hint.

3.4.2.1 Intervention controller

The process of providing adaptive interventions starts by identifying which of the available intervention items are relevant at any given point of a user's interaction with an ELE based on the satisfied association rules at that time. There may be several rules that are active (i.e., all of their preconditions are satisfied) at a given time, which brings to bear an important problem with the basic approach of considering all the corresponding intervention items for providing adaptive support. The problem is that addressing all the relevant patterns by delivering the respective interventions at any given time might overwhelm the student and reduce the overall effectiveness of the intervention mechanism.

Our solution is to only address the most prominent behaviour at any given time (i.e., only deliver one intervention at a time). This solution requires devising a strategy for ranking the intervention items that are relevant at any given time based on their prominence.

Based on this approach, at each hinting opportunity, the intervention controller chooses the intervention item with the highest ranking among the relevant items following the online ranking process described next.

3.4.2.2 Online ranking process for intervention items

As mentioned above, this process is done online during the user's interaction with the ELE. It takes as input the rules that are satisfied at each given time for the user and calculates the score for each intervention item; subsequently based on these scores the most relevant intervention item is determined.

Each intervention item h is by definition mapped to a precondition which belongs to one or more rules in the class. Therefore, each intervention item can be triggered by one or more rules in the class. We show this set of rules by $rules(h)$. During a user's interaction with the ELE, for each intervention item h, its score is calculated as the sum of the weight of those rules in $rules(h)$ that are satisfied at the given time (namely the rules that caused the user's current classification in one of the available clusters).

Thus, intervention items are continuously re-ranked based on their score as the user interacts with the ELE. This score is calculated based on how the user model currently classifies the user and which rules trigger the classification. Note that at any time, only a subset of intervention items may be relevant depending on whether there is any satisfied rule that triggers that item. Here is an example of how an item becomes relevant: when a user is classified as LLG, and our sample rule in Figure 3-3 is activated, then the intervention items related to its preconditions become relevant. For the precondition "Direct Arc Click frequency = Low", the intervention entails prompting the user to perform more Direct Arc Click actions and for the precondition "Auto AC frequency = High", the intervention item involves prompting the user to try alternative actions instead of Auto AC.

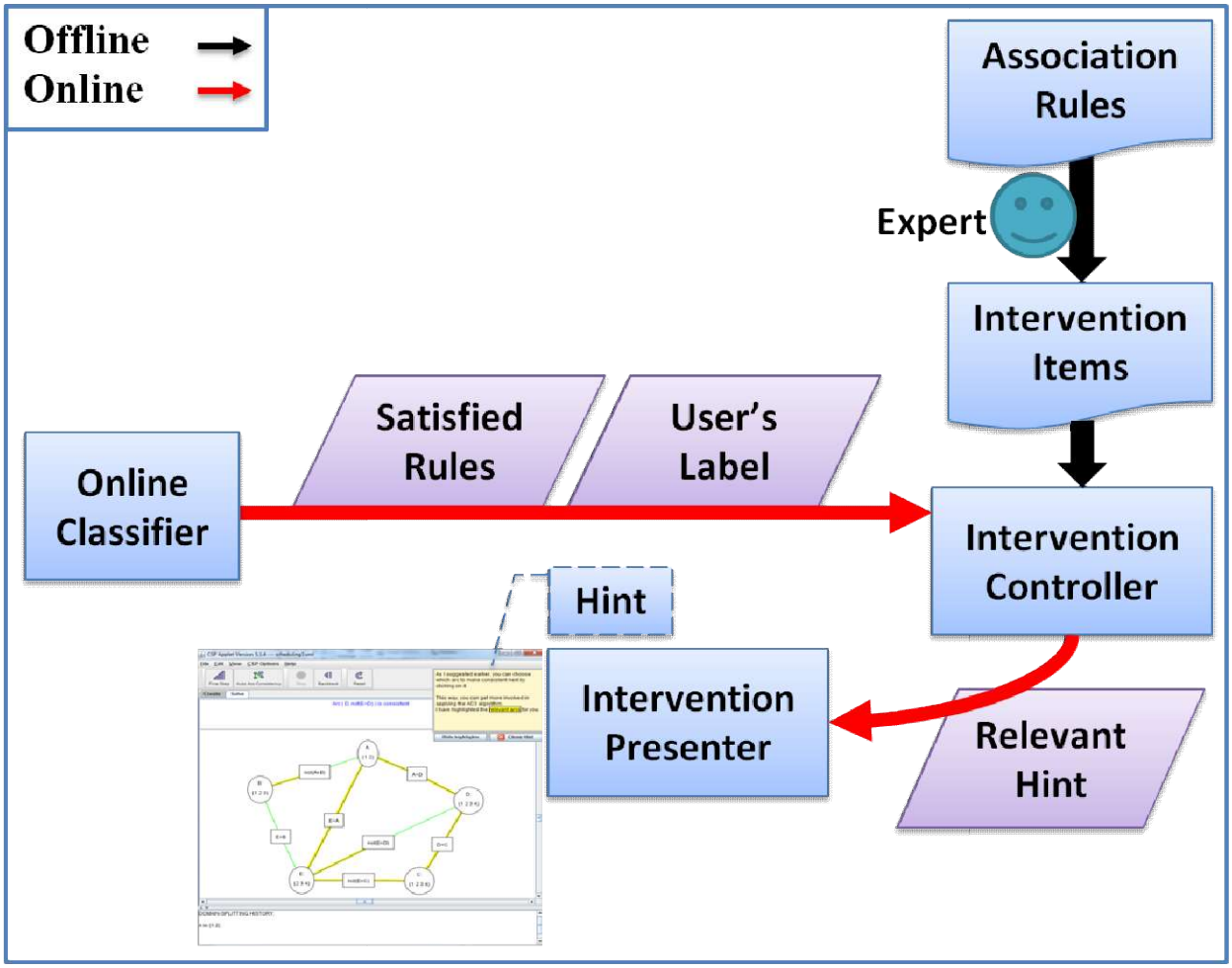


Figure 3-4 The adaptive support process

3.4.2.3 Intervention presenter

The second element of the intervention mechanism is the intervention presenter. The objective of the intervention presenter is to deliver interventions while trying to maximize effectiveness and minimize the intrusiveness of the process. In the rest of Section 3.4, we present the final version of the intervention presenter component. In Chapter 6 we explain the iterative process that led to this final design.

3.4.3 Delivering the adaptive interventions

Our first design principle is to deliver adaptive interventions incrementally, following well-established practices in the Intelligent Tutoring Systems literature as reviewed in (Maloy, Verock-O'Loughlin, Edwards, & Woolf, 2013; VanLehn, 2011). Thus, each selected intervention item is first delivered with a textual hint that prompts or discourages a target behaviour, followed when needed by a textual hint that reiterates the same advice, accompanied by a related interface adaptation that can help the user follow the advice (e.g., highlighting relevant interface items).

3.4.4 Intervention strategy

Delivering adaptive interventions also requires deciding whether the interventions should be subtle or forceful. Subtle interventions are in the form of suggestions that can be easily ignored by the user. Forceful interventions make the user follow the related advice by reducing or eliminating user's options for the next action. We decided to adopt the subtle approach for the adaptive-CSP applet because, although with this approach the user can easily decide to ignore the system's suggestions (even when they are appropriate), it has the very desirable advantage of being less intrusive than the forceful approach. Therefore, from a usability point of view, it makes sense to try and see whether subtle adaptive interventions can already improve the effectiveness of the CSP applet. The general mechanism to deliver subtle incremental adaptive interventions in the CSP applet works as follows:

(*i*) Each intervention item selected for delivery (*target item* in the rest of this section) is first presented as a textual hint message shown in a box at the upper right corner of the applet, as shown in Figure 3-5a (level-1 hint). This message is phrased as a suggestion for behaviours to be adopted or avoided. For instance, a level-1 textual hint for the DAC_fr intervention item, that we

used in our example above, is "*Do you know that you can tell AC-3 which arc to make consistent by clicking on that arc?*" which aims to promote the *Direct Arc Click* action. In order to draw user's attention to the new message, the hint box first appears in the center of the screen, and then quickly moves to the upper right corner of the applet.

(*ii*) After receiving a level-1 hint on the *target item*, the student is given some time to change her behaviour accordingly (a *reaction window* equal to 20 actions for the given hint). During this time, the user model will keep updating the feature vector describing the user interaction behaviour, its classification, and the ranked list of active intervention items, excluding the *target item*, thus other hints may be given during this time if relevant and needed. To avoid overwhelming the user with hints, a maximum frequency of one hint per ten actions is enforced (i.e., hinting opportunity). In other words, students may receive a hint every 10 actions, but they may receive a hint on the same behaviour only every 20 actions. At each hinting opportunity, either a new hint is displayed or, if there are no new hints, the hint box disappears. As shown in Figure 3-5, the user can also close the hint box using the "close" button at any time.

(*iii*) At the end of the reaction window, based on the updated feature vector, the user model determines whether the user has followed the hint for the *target item* or not. If at this point the preconditions for the adaptation rule that generated the level-1 hint for the *target item* are still satisfied, then the user has not followed the hint and the *target item* is selected for delivery again.

(a) (b)

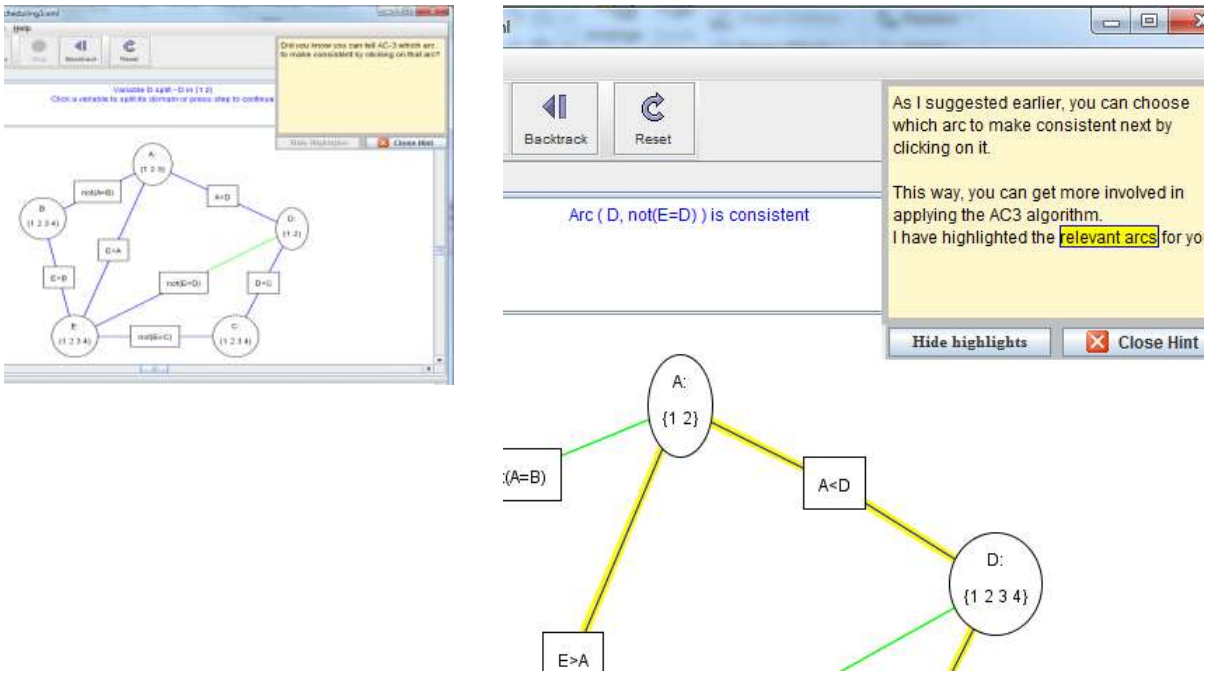


Figure 3-5 Sample Level-1 (a) and Level-2 (b) hints.
The text for the Level-1 hint on the left: "Did you know you can tell AC-3 which arc to make consistent by clicking on that arc?"

(*iv*) In this case, a level-2 hint is delivered, consisting of both a text message as well as the highlighting of relevant interface elements that can help the user follow the hint. Figure 3-5b shows an example of a level-2 hint for the DAC_fr intervention item. The text messages in level-2 hints are more detailed than for level-1 hints. The first part of a level-2 hint reiterates the level-1 hint with slightly different wording, while the second part provides a justification for the suggestion[22] and, if relevant, mentions the highlighted elements of the interface. For instance, in Figure 3-5b, the arcs of the displayed CSP graph that can be selected via Direct Arc Click are highlighted in yellow. The phrase “relevant arcs” in the hint message is highlighted similarly to

[22] Please note that, while the content of the first part (related to the behaviour) is derived from the pattern, the justification is based on the designer’s interpretation of why that behaviour is good or bad based on his/her pedagogical knowledge of the domain. Therefore, adding the justification is optional and depends on the designer’s knowledge of the domain and confidence in his/her interpretation of the behaviour pattern.

create a visual link between the text and the change in the graph. Different hints will cause different interface elements to be highlighted, such as nodes in the graph (Figure 3-6 right) or toolbar buttons (Figure 3-6 left).

Figure 3-6 In a level-2 hint, the relevant elements of the interface are highlighted

The highlight effect is automatically removed as soon as the user performs a relevant action (e.g., clicks on an arc for DAC_fr). The highlights can also be manually toggled using the "hide/show highlights" button at the bottom of the hint box (see Figure 3-5b).

(***iv***) If the user's behaviour during the reaction window following a level-2 hint shows that the user still did not follow the system's suggestion, the corresponding intervention item will be put back on the list of active items and, if selected, it will be delivered starting again from a level-1 hint. In the current version of the applet, there is no limit to the number of times a hint is delivered following the above process (alternating between level-1 and level-2)

3.5 Summary

In this chapter, we described different phases of the proposed User Modeling and Adaptation (UMA) framework: *Behaviour Discovery*, *User Classification*, and *Adaptive Support*. In the behaviour discovery phase, clusters of users who interact in the same way are found, and the most descriptive behaviours of each cluster are mined in form of class association rules. In the user classification phase, an online classifier user model is built (based on the discovered rules in the previous phase) that classifies each new user into the appropriate group of users based on his/her behaviour. Finally, the adaptive support phase consists of an intervention controller and an

intervention presenter. The intervention controller is the mechanism for selecting adaptive suggestions based on the output of the user model that identifies in real-time if a student is learning well from the CSP applet and, if not, why. The intervention presenter is the user-facing element of the framework and uses a two-level subtle method of delivering interventions (suggested by the intervention controller) using both text messages and interface highlights. More details on each phase of the framework and the user experiments used to arrive at the final version of the framework are provided in Chapters 5, 6, and 7. In the next chapter, we describe the data collection process and datasets used in our data mining efforts.

Chapter 4: User Studies for Data Collection

In this chapter, we describe the two user studies that we ran to collect user interaction data with the purpose of evaluating the proposed user modeling and adaptation framework. First, we describe the common study design used for both user studies, then we provide more information about each study, and finally, we describe the data extraction process from the user activity logs collected during the two studies.

4.1 Study design

The general process of the studies done throughout this thesis involves some common elements with minor changes depending on the design and the goals of the study. One common requirement is that participants should be learning the concepts related to Constraint Satisfaction Problems (CSPs) for the first time. Additionally, they should already have a basic understanding of graph theory which is required for learning how to represent the CSP problems and how to solve them using the AC-3 algorithm. Therefore, all the studies start by asking participants to study a textbook chapter on Constraint Satisfaction Problems and the AC-3 algorithm (Poole & Mackworth, 2010). This part was allotted 45 minutes and all the participants reported finishing the material within the given time in all of our studies. Another common component of all the main studies in this thesis is the measurement of learning gain by asking participants to write a pre-test and a post-test before and after the main study task, respectively. Both pre-test and post-test in all studies involved conceptual as well as problem-solving questions designed to evaluate their understanding of the CSP concepts that were covered in the chapter they had studied (sample tests are available in Appendix A-2). The tests were taken in pen-and-paper format and were marked following a marking scheme. Another requirement of the studies was that participants did not have prior experience using the CSP applet. Because of this requirement we

made sure every participant received the same information regarding how to use the applet before starting the main task. We provided a comprehensive video that described all the functionalities of the CSP applet to the participants. Participants could pause the video and ask questions about the information provided in the video or could ask about any of the actions in the applet after the video. Most of the participants found the information provided in the video adequate and were ready to use the CSP applet afterward. A few participants chose to ask questions from the experimenter or watch parts of the video again. The main task of all studies included finding all the solutions for a set of given CSP problems. The difficulty of the problems increased from the first to the last problem (as measured by the number of variables and constraints and the number and type of steps necessary to find the solutions, ranked by the author). A sheet containing all the necessary information about the task was given to the participants prior to the start of the task. Students were asked to work with the applet for a minimum of 10 minutes however there was no maximum time limit for this part of the study. As mentioned before, after finishing the main task participants were asked to write a post-test that had a similar structure to the pre-test. This process is shown in Figure 4-1.

4.2 First user study

The goal of the first user study was to collect user interaction data so that we could investigate whether students interact differently with the CSP applet in terms of how they use the interface actions. We also wanted to find out if there were meaningful differences in learning performance of the students that could be associated to the differences in their interaction behaviours.

This user study followed the experimental protocol for the CSP applet described in the previous section. All participants were university students who had taken a set of courses ensuring that they were familiar with basic graph theory, thus having the prerequisites to study

Constraint Satisfaction Problems. As explained in Chapter 1, because of this specific background knowledge requirement, recruiting subjects and running the user experiment for the CSP applet was challenging and very time-consuming. In this study, the main task included using the CSP applet to find all the solutions for two CSP problems where the first problem did not have any solutions and the second problem had 3 solutions (problems named "Simple-Problem 2" and "Scheduling-Problem 2" in the CSP applet). The study was done during summer and fall semesters in the year 2010. During this study, all student actions were logged and later processed. The resulting dataset of interface actions contains action logs of 65 users (compared to 24 users in (Bernardini & Conati, 2010)), totaling 13,078 actions over 62,752 seconds of interaction.

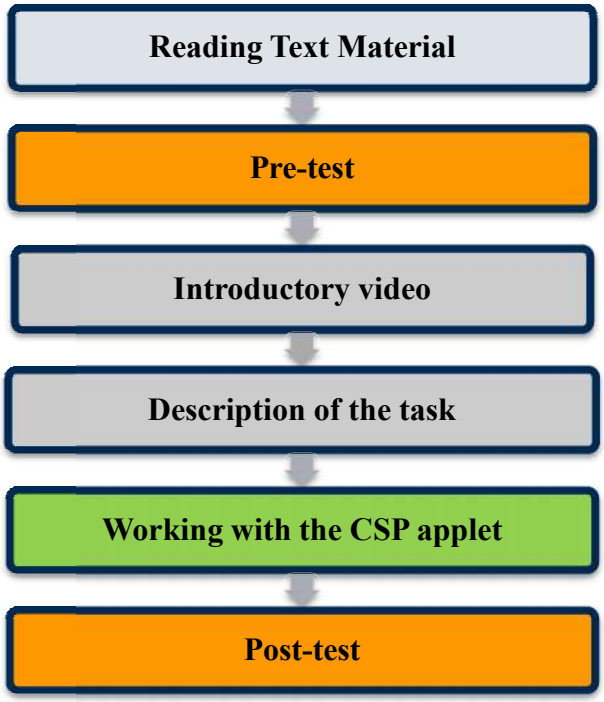


Figure 4-1 The Study process

4.3 Second user study

During 2011, we ran a second user study to collect user interaction patterns from two sources: (*i*) interface action logs similar to the first study, and (*ii*) eye-gaze patterns which were collected using an eye-tracker. The purpose of the study described in this section was to collect data in order to investigate whether a user's attention patterns can be indicators of effective vs. non-effective learning with the CSP applet.

Fifty computer science students participated in this user study. The data for 5 students was not usable due to technical issues, reducing the dataset to 45 users. The general structure of the study is similar to the study design described in Section 4.1, except for changes necessary to use the eye-tracker. Participants were run one at a time, and each experimental session was structured as follows. Similar to the previous study, participants were asked to study the same textbook chapter on Constraint Satisfaction Problems and the AC-3 algorithm and write a pre-test for evaluating their understanding of the material they had studied. Next, participants were shown a video that explained the functionalities of the CSP applet.

The main part of the experiment was run on a Pentium 4, 3.2GHz, with 2GB of RAM with a Tobii T120 eye-tracker as the main display. Tobii T120 is a remote eye-tracker embedded in a 17" display (Figure 4-1), providing unobtrusive eye-tracking (as opposed to what head-mounted devices do). In addition to the user's eye-gaze data, Tobii also records video data of the user's face. After undergoing a calibration phase for the eye-tracker, the participants started working with the applet to solve two CSP problems: first an easier problem involving 3 variables, 3 constraints and at most 2 domain splitting actions to find its unique answer; next, a more difficult problem involving 5 variables, 7 constraints and a minimum of 5 domain splitting actions to find its two solutions. Participants were instructed to find both of these solutions. All relevant

instructions for this phase were provided on a written instruction sheet. No time limit was given for this phase, which lasted on average 16.7 (SD = 9.0) minutes. The study ended with a post-test analogous to the pre-test.

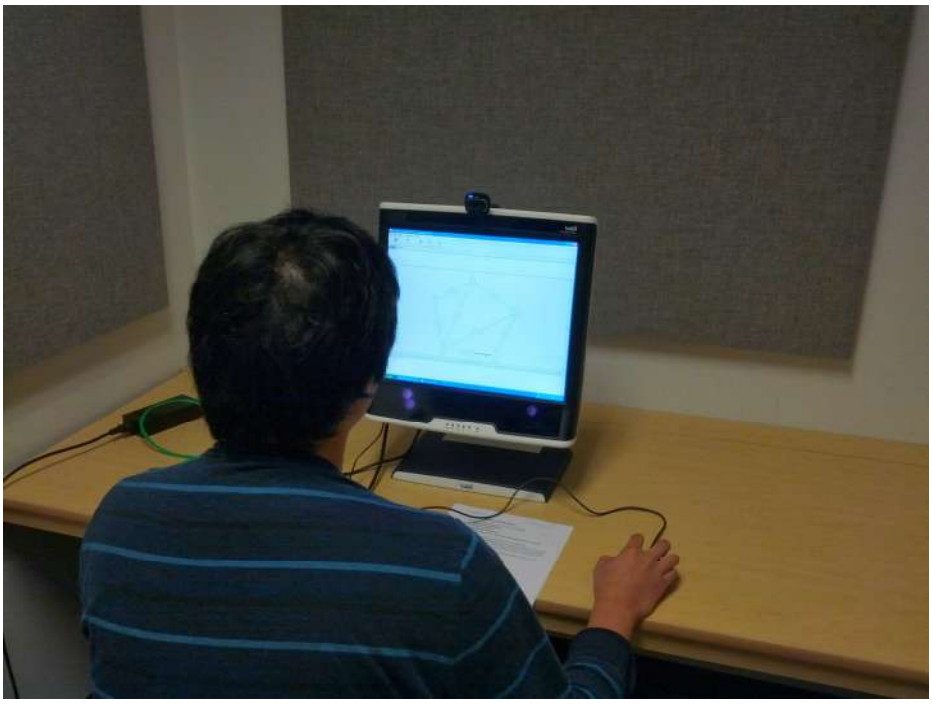


Figure 4-2 A participant working with the CSP applet on a display with integrated eye-tracker

4.4 Measure of learning performance

To quantify students' learning performance as a result of working with the CSP applet we need to define a measure based on their pre-test and post-test scores. Learning level was measured using a standard measure of learning performance (Proportional Learning Gain, or PLG), calculated as follows:

$$PLG = \begin{cases} \frac{PostT - PreT}{Max - PreT} \times 100 & if\ PostT - PreT > 0 \\ 0 & otherweise \end{cases} \quad (4\text{-}1)$$

where *PreT* and *PostT* are the student's test scores and *Max* is the maximum possible score.

The advantage of PLG compared to measuring the absolute difference between pre-test and post-test scores, is that it assigns more value to each point improvement of post-test score

compared to pre-test score when users start from a higher pre-test score. This is important because the level of understanding needed to improve user's score raises as the basis of user's knowledge (pre-test score) increases.

4.5 Datasets

Given the duration of data collection, we used different sub-sets of the overall data for our analyses depending on availability of data at the time. In the rest of this thesis we use three versions of the data collected during the above user studies:

1) CSP-Action-65: This dataset contains the interface-action data of 65 users collected during the first study. This was the basis for our initial analysis and evaluation of the user model performance.
2) Eye-and-Action-45: This dataset contains interface-action data and eye-tracking information for 45 users, who participated in the second study. This dataset was used in our eye-gaze and eye-movement analyses reported in Chapter 8.
3) CSP-Action-110: This is the final dataset of interface-action data for all 110 users that worked with the original version of the applet (i.e., non-adaptive). This dataset is the result of combining the interface action data from the above two datasets and is the most comprehensive one on interface-action data. The behaviour patterns found in this dataset are used for building the adaptive version of the applet.

The descriptive statistics for the three mentioned datasets are presented in Table 4-1. We compared the average learning performance of users in CSP-Action-65 and Eye-and-Action-45 datasets and found no significant difference, $t(108) = 0.47$, $p = 0.64$. This suggests that the additional information provided by the eye-tracker did not affect learning performance of

students, as we expected. Thus, given that every other aspect of the protocols and material of the 2 studies are similar, we can safely combine the two to create the CSP-Action-110 dataset.

Table 4-1 Descriptive Statistics for the 3 Datasets

	CSP-Action-65	Eye-and-Action-45	CSP-Action-110
Number of users	65	45	110
Total number of actions	13,078	12,308	25,386
Average length of the interaction with the applet (minute)	16.1	16.7	16.35
Average Pre-test score (out of 22)	13.39 (SD = 5.08)	11.36 (SD = 5.36)	12.56 (SD = 5.24)
Average Percentage Learning Gain (%)	52.75 (SD = 32.25)	44.46 (SD = 28.06)	49.36 (SD = 30.74)

In the next chapter, we report the results of the behaviour discovery and user classification phases on CSP-Action-65 and CSP-Action-110 datasets.

Chapter 5: Behaviour Discovery and User Classification on Interface Action Data

This chapter covers the results of the Behaviour Discovery phase of the User Modeling and Adaptation (UMA) framework (described in Chapter 3) on collected interface action data obtained by the user studies (described in Chapter 4). In this chapter we provide answers to the following research questions posed in the introduction:

Q1: *How can data mining techniques be used to identify from the ELE interaction data, groups of users with different performance levels and their distinguishing interaction patterns?*

Q2: *Is it possible to use the patterns detected in [Q1] to build a classifier user model that effectively classifies new users based on their behaviours during the interaction?*

5.1 Data extraction

For each user, we calculate one feature vector that summarizes his/her interaction behaviour during the session. To calculate this feature vector from the collected action logs, we calculated: (*i*) usage frequency of each interface action, and (*ii*) mean and standard deviation of latency between actions. Average latency is an indicator of the time spent reflecting after an action and planning for the next one, while the standard deviation of latency tells if the user was consistent or selective in the amount of pausing after each action. As described in Chapter 2, we have 7 interface actions (i.e., Fine Step, Direct Arc Click, Auto Arc Consistency, Stop, Domain Splitting, Backtrack, and Reset) thus the calculated feature vectors are 21-dimensional. Table 5-1 shows the feature names for the three features generated for action “Fine Step”.

Table 5-1 Feature names for the "Fine Step" action

Action	Frequency	Average latency	Standard deviation of latency
Fine Step	Fine Step frequency	Fine Step pause average	Fine Step pause standard deviation

5.2 Outcomes of behaviour discovery

In this section, the outcomes of behaviour discovery phase are presented. All results are for the CSP-Action-110 dataset.

5.2.1 Outcomes of user clustering

First, we determined the optimal number of clusters for the data (k = 2) following the method described in Section 3.2.2. Then, we checked if/how the discovered clusters relate to learning performance of students. From the calculated test scores, we computed the proportional learning gains for each student (in percentage) and then analyzed the clusters detected for K=2, to see if there is any significant difference with regard to learning gains. An independent samples t-test revealed a significant difference in the learning gain between the two clusters (p = .03 < .05**)** with a medium effect size (Cohen d = .47). We refer to these clusters as High (n = 18, M = 61.32, SD = 27.38) and Low (n = 47, M = 39.28, SD = 62.06) Learning Gains (HLG and LLG respectively). There is no significant difference between the average pre-test scores of LLG and HLG (p = .19**)**, indicating that behaviour patterns of the HLG group have an impact on their learning. This information is summarized in Table 5-2.

Table 5-2 Learning difference between the two clusters

	ALL		HLG		LLG		t-value	p-value	Cohen's d
	Mean	SD	Mean	SD	Mean	SD			
PLG	49.36	30.60	53.20	30.33	41.13	29.54	1.96	**0.0273**	0.40
Pre-test	12.56	5.24	12.20	5.30	13.33	5.03	-1.07	0.1438	-0.22

5.2.2 Outcomes of association rules mining

As explained in Section 3.2.3, there are 3 parameters for the association rules mining algorithm (i.e., minimum support, branching factor, and minimum improvement of confidence) and an additional parameter for the discretization algorithm (i.e., the optimal number of bins). Using nested cross-validation we found 50%, 3, and 5% to be the optimal values for minimum support, branching factor, and minimum improvement of confidence accordingly. We also found 7 to be the optimal upper-bound for the number of bins for the discretization algorithm.

Table 5-3 shows the representative rules (i.e., rules with a confidence value above 50%) for the HLG and LLG clusters in the CSP-Action-110 dataset, where we report the preconditions for each rule but leave out the consequence[23]. The table also shows, for each rule, its level of confidence (*conf*), and support within its cluster (*class cov*). In the rest of this section, we discuss a few examples of the rules generated by the framework and how they can be used to define adaptive interventions. The complete list of adaptive interventions is presented in Section 6.1.

Auto Arc Consistency frequency appears in Rule 1 for the HLG cluster, with its value in the lowest bin, while it appears in Rule 5 for LLG with its value falling in the highest bin, indicating that LLG members use *Auto Arc Consistency* much more than HLG members. These 2 rules

[23] The consequence is always the class label

reinforce the idea that students are prone to abusing this action without learning the details of the algorithm[24]. Thus it would be beneficial to trigger an intervention to advise them against this behaviour (i.e., high frequency of Auto Arc Consistency) for students who engage in it.

Table 5-3 Representative rules for HLG and LLG clusters

Rules for HLG cluster* (75/110)**:** Rule 1: Auto AC frequency = Lowest (Conf = 92.31%, Class Cov = 48/75) └ Rule 2: Auto AC frequency = Lowest **and** Fine Step frequency = lowest (Conf = 100% , Class Cov = 33/75) Rule 3: Direct Arc Click Pause STD = Highest (Conf = 92.11%, Class Cov = 35/75) Rule 4: Direct Arc Click Pause Avg = Highest (Conf = 91.43%, Class Cov = 32/75)
Rules for LLG cluster* (35/110)**:** Rule 1: Backtrack frequency = Highest (Conf = 85.19%, Class Cov = 23/27) └ Rule 2: Backtrack frequency = Highest **and** Auto AC frequency = Highest (Conf = 100%, Class Cov = 20/20) └ Rule 3: Backtrack frequency = Highest **and** Direct Arc Click Pause STD = Lowest (Conf = 100%, Class Cov = 19/19) └ Rule 4: Backtrack frequency = Highest **and** Direct Arc Click Pause Avg = Lowest (Conf = 100%, Class Cov = 16/16) Rule 5: Auto AC frequency = Highest (Conf = 82.35%, Class Cov = 28/34) └ Rule 6: Auto AC frequency = Highest **and** Reset frequency = Highest (Conf = 100%, Class Cov = 15/15)

[24] Recall that Auto Arc Consistency quickly runs AC-3 to completion making the whole graph arc-consistent

└ Rule 7: Auto AC frequency = Highest **and** Auto AC Pause STD = High (Conf = 93.33%, Class Cov = 14/15)

└ Rule 8: Auto AC frequency = Highest **and** Auto AC Pause STD = High **and** Direct Arc Click frequency = Lowest (Conf = 100%, Class Cov = 14/14)

└ Rule 9: Auto AC frequency = Highest **and** Reset Pause Avg = Low (Conf = 93.33%, Class Cov = 14/15)

Rule 10: Reset frequency = Highest (Conf = 78.26%, Class Cov = 18/23)

└ Rule 11: Reset frequency = Highest **and** Fine Step Pause Avg = Lowest (Conf = 87.5%, Class Cov = 14/16)

└ Rule 12: Reset frequency = Highest **and** Domain Split frequency = Highest (Conf = 85%, Class Cov = 17/20)

└ Rule 13: Reset frequency = Highest **and** Domain Split frequency = Highest **and** Auto AC frequency = Highest (Conf = 100%, Class Cov = 15/15)

└ Rule 14: Reset frequency = Highest **and** Domain Split frequency = Highest **and** Domain Split Pause STD = Lowest (Conf = 93.75%, Class Cov = 15/16)

└ Rule 15: Reset frequency = Highest **and** Domain Split Pause STD = Lowest (Conf = 84.21%, Class Cov = 16/19)

* Conf= Confidence; Avg= Average; Class Cov= Class Coverage, STD = Standard Deviation

Rule 8 for LLG indicates that low learners show very high frequency of *Auto AC* actions, while rarely using the *Direct Arc Click* (DAC) action. A possible explanation of why both of these behaviours are associated with limited learning is that they identify users who are not very engaged in the exploration process because they prefer to (*i*) run the algorithm to completion instead of stepping through it; (*ii*) leave to the applet the selection of the next arc to work on, rather than being proactive in choosing it.

Thus, this rule identifies two possible interventions (*intervention items* from now on) that address these suboptimal behaviours exhibited by low learners: (*i*) discourage excessive use of the Auto AC; (*ii*) encourage higher usage of DAC.

As our last example, we discuss an intervention regarding the time spent after an action (pause). Rule 4 for HLG indicates that high learners tend to spend more time after performing the DAC action (*Direct Arc Click Pause Avg* = Highest) which could indicate reflecting on the outcomes of the action. Conversely, Rule 4 for LLG indicates that low learners use the *Backtrack* action frequently and tend to spend less time after performing DAC. Again there are two possible intervention items for these rules: (*i*) encourage reflection after DAC actions; (*ii*) discourage excessive use of Backtrack action.

5.2.3 Summary

In summary, this section illustrates that the proposed behaviour discovery process can effectively identify groups of students who interacted similarly with the CSP applet as applied to the CPS-Action-110 dataset. Also, rules generated by our framework are informative and can be used for generating real-time adaptive interventions. These interventions, however, are appropriate only if the classifier user model can recognize which users need them. In the next section, we present the evaluation results for the classifier user model generated from these association rules.

5.3 Resulting rule-based classifier

One of the first evaluations of this work was finding out if the user model created from the association rules can achieve an acceptable accuracy in classifying new users while they are interacting with the system (Kardan & Conati, 2011). This section presents the performance of

the user model classifier built in the user classification phase for the CSP-Action-110 dataset. We present the over-time accuracy (described later in this section) of the classifier user model.

As mentioned in Section 5.2, behaviour discovery on the CSP-Action-110 dataset generated two clusters of users, namely High Learning Gain (HLG) and Low Learning Gain (LLG) groups, therefore creating a binary classification problem. We evaluated the average over-time accuracy (described later in this section) of the rule-based classifier using 10-fold cross-validation. In all of our evaluations of the classifier user model, the reference labels (i.e., ground truth) are the cluster labels assigned to each user, based on the user's complete interaction data at the end of the session. This makes the classification task especially challenging at the early stages of the interaction since less data is available to the classifier.

The over-time accuracy is calculated as users progressed through the session at different percentage points (e.g., first 10% of the session, first 25% of the session, etc.). Average over-time accuracy is calculated by averaging the 100 points derived from calculating the over-time accuracy at each percentage of the session. Our goal is to provide adaptive support during the session, not at the end of it, thus average over-time accuracy was selected over final accuracy (i.e., using 100% of session data) to favor classifiers that achieve better performance earlier in the session.

The over-time accuracy of the classifier is shown in Figure 5-1. We presented the overall accuracy as well as class accuracy for HLG and LLG classes. For comparison, we also included the overall accuracy of the most likely class (HLG in this case) classifier as the baseline. The rule-based classifier reaches a relatively high accuracy in early stages of the interaction which is essential when the goal is to provide adaptive interventions to improve the user experience with an ELE. The average over-time accuracy is 77.58% (SD = 5.3). More importantly, the classifier

achieved an overall accuracy of 78.2% after observing only the first 25 percent of the interactions, indicating that this user model can be reliably used to trigger adaptive interventions from the early stages of the interaction (Figure 5-1).

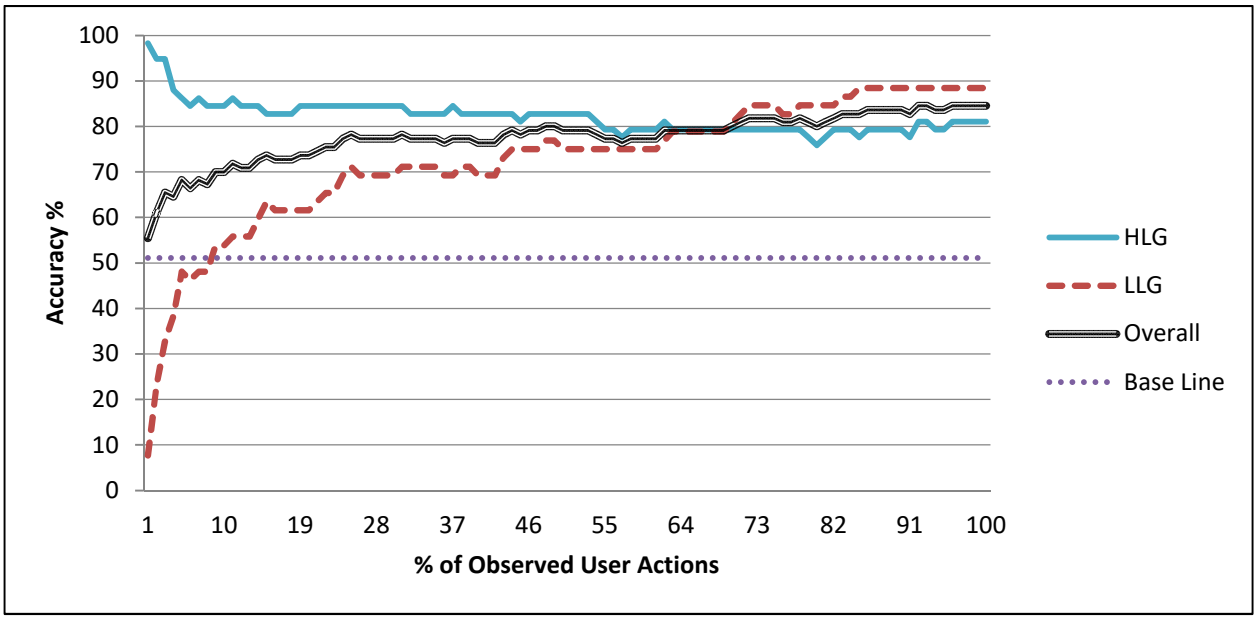


Figure 5-1 Performance of the rule-based classifier on CSP-Action-110 dataset

5.4 Discussion

It is essential to ensure that the results obtained for the accuracy of a user model can be repeated for new students (i.e., generalizability across the population). In addition to using proper accuracy evaluation schemes (such as nested cross-validation used here), one should also look into the size and diversity of the dataset used for the training and testing of the model. We used a relatively large dataset with 110 students, compared to 24 students used in the previous similar studies on the CSP applet (Amershi & Conati, 2009; Bernardini & Conati, 2010), and the data was collected over 2 years from 4 different cohorts of students. This made us confident in the generalizability of the user model across the population and repeatability of the results for new students. If fact, in Section 7.5, we provide the accuracy results for the user model on

another group of 38 students that participated in our final evaluation study (i.e., a new test-set). As expected, the accuracy of the model (78.95%) remained consistent with the results obtained here (77.58%).

One feature of our datasets which made the classification challenging was the fact that we used summary features for describing the user interactions, and these features by nature have a high variance at the early stages of the interaction. This characteristic of the features meant that in order to achieve high accuracy in the early stages of the interaction, we needed a model that is not highly sensitive to small variations in the feature values. We think the discretization of feature values into bins before training the model helped in this regard by reducing the fluctuations in the data over the interaction (i.e., the bin assigned to a "discretized feature" is much less likely to change over the interaction compared to the value of the underlying feature). As mentioned earlier, in our case, discretization was done in association rules mining process prior to building the rule-based classifier user model. In Chapter 8, we provide a comparison in terms of over-time accuracy between the rule-based classifier used in UMA framework and other common classifiers available in the Weka data mining toolkit.

5.5 Summary

In this chapter, we provided an empirical evaluation of the behaviour discovery and user classification phases of the UMA framework. The results on CSP-Action-110 dataset provide evidence that not only the UMA framework can cluster users into meaningful groups [Q1], but also it can classify new users accurately [Q2]. More importantly, the framework generates rules that provide a fine-grained description of common behaviours for users in different clusters. These rules appear to be suitable to guide adaptive interventions targeted at improving

interaction effectiveness which is relevant to [Q3]. The next chapter discusses the process of providing adaptive support based on the discovered association rules.

Chapter 6: Providing Adaptive Support based on Interface Action Patterns

In this chapter, we describe the process of providing *adaptive support* based on the discovered association rules in the behaviour discovery phase (described in Section 5.2). Providing adaptive support is the third and final phase of the UMA framework proposed in Chapter 3. First, we focus on the intervention controller and the intervention items extracted from the association rules presented in Table 5-3, that are the outcome of applying behaviour discovery to the CSP-Action-110 dataset. Then we describe the iterative design process for the intervention presenter component of the CSP applet which resulted in the final design that is presented in Chapter 3.

6.1 Adaptive interventions for the CSP applet

Based on the 19 rules extracted in the behaviour discovery phase, we derived 9 intervention items summarized in

Table 6-2. We followed the offline ranking process described in Section 3.4.1.1 and arrived at the weights and ranking presented in Table 6-1.

There were originally 15 patterns (i.e., preconditions) in the discovered rules. However, when designing the interventions items, we made two design decisions that reduced this number to 9. These two decisions are specific to the CSP applet and the features calculated for the interface actions in the CSP-Action-110 dataset. One issue was related to designing interventions for features pertaining to the standard deviation of the time spent reflecting after each action. It is rather easy to describe the behaviour that results in a low vs. high value for these features, i.e., users are consistent vs. selective/irregular in the amount of time they spend after the target action. However, it is confusing and impractical to tell users to be more consistent (for low standard deviation) or selective (for high standard deviation) in the amount of time they are spending after an action. After careful deliberation and multiple discussions, we could not find a

way to formulate the hint messages related to the standard deviation of pause-after-action features that were detected in the process (namely Direct Arc Click Pause STD[25], Auto AC Pause STD, and Domain Split Pause STD). Finally, we decided to discard the four patterns related to these features from the rest of the process[26].

The second issue was an observation that there were opposing patterns in the rules for opposing clusters. For example, the precondition Auto AC frequency = Lowest is present in rules 1 and 2 for HLG cluster while Auto AC frequency = Highest is present in rules 2, 5, 6, 7, 8, 9, and 13 for the LLG cluster. The intervention item for Auto AC frequency = Lowest in HLG entails encouraging users to use Auto AC less often (or use alternative actions) and the interventions item for Auto AC frequency = Highest in LLG entails discouraging users from using Auto AC too often (and use alternative actions). These two items are essentially the same, thus it is possible to design one intervention item for both patterns and map the association rules for both preconditions to trigger that same item.

This was also the case for Direct Arc Click Pause Avg = Highest for HLG and Direct Arc Click Pause Avg = Highest for LLG patterns. In this case, we are interested in giving a hint if a user is not spending enough time after performing Direct Arc Clicks. The condition of not spending enough time can be determined by HLG4 (if not satisfied) or LLG4 (if satisfied), both triggering "encourage the user to spend more time thinking about the outcomes of the Direct Arc Click action". As a result, the total number of intervention items was reduced to 9 (see Table 6-2).

[25] This feature appears in one pattern for HLG and one pattern for LLG.

[26] It noteworthy that the standard deviation of pause-after-action features are still useful as they contribute to the user's classification.

Table 6-1 Weight and ranking of association rules

Rule	Confidence (%)	Class Coverage	Weight	Rank in class
HLG1	92.31	0.64	59.0784	1
HLG2	100	0.44	22	4
HLG3	92.11	0.466667	42.98467	2
HLG4	91.43	0.426667	39.01013	3
LLG1	85.19	0.657143	55.982	2
LLG2	100	0.571429	28.57143	4
LLG3	100	0.542857	27.14286	5
LLG4	100	0.457143	22.85714	6
LLG5	82.35	0.8	65.88	1
LLG6	100	0.428571	21.42857	7
LLG7	93.33	0.4	18.666	10
LLG8	100	0.4	13.33333	15
LLG9	93.33	0.4	18.666	11
LLG10	78.26	0.514286	40.248	3
LLG11	87.5	0.4	17.5	12
LLG12	85	0.485714	20.64286	8
LLG13	100	0.428571	14.28571	13
LLG14	93.75	0.428571	13.39286	14
LLG15	84.21	0.457143	19.248	9

To implement the online ranking process described in Section 3.4.2.2, we saved the pre-calculated weights for the rules as well as the mapping of the rules to intervention items in a data structure as presented in

Table 6-2.

Additionally, we implemented a method that, given the *id* of the rules that were active (provided by the classifier user model), would produce a list of relevant intervention items sorted by their score (i.e., sum of the weights of the active rules mapped to that item, as explained in Section 3.4.2.2). Thus at every hinting opportunity, if a user is classified as LLG, the intervention controller produces the best intervention item to be delivered and triggers the intervention presenter to deliver that intervention item.

Table 6-2 List of intervention items, their descriptions, and corresponding association rules[27]

Intervention Code	Intervention Description	Corresponding association rules
DAC_fr	Using Direct Arc Click more often	LLG 8
DAC_PA	Spending more time for reflection, after performing Direct Arc Clicks	HLG 4[28], LLG 4
Reset_fr	Using Reset less frequently	LLG 6, 10, 11, 12, 13, 14, 15
AAC_fr	Using Auto Arc-Consistency less frequently	HLG 1, 2 LLG 2, 5, 6, 7, 8, 9, 13
DS_fr	Using Domain Splitting less frequently (only when appropriate)	LLG 12, 13, 14
FS_PA	Spending more time after performing Fine Steps	LLG 11
BT_fr	Using Backtrack less frequently (only when appropriate)	LLG 1, 2, 3, 4
FS_fr	Using Fine Step less frequently	HLG 2
Reset_PA	Spending more time after Resetting the graph, for planning	LLG 9

The next section illustrates the iterative design process behind the intervention presenter mechanism which is tasked with delivering the interventions while trying to maximize effectiveness and minimize the intrusiveness of the process.

6.2 Delivering the adaptive interventions

As mentioned in Chapter 3, the goal for the intervention presenter component is to deliver hints in the most effective way while avoiding intrusion as much as possible. The final hint

[27] The text messages for these hints are provided in Appendix A.3

[28] As mentioned earlier, in this case we are interested in giving a hint if a user is not spending enough time after performing Direct Arc Clicks. The condition of not spending enough time can be determined by HLG4 (if not satisfied) or LLG4 (if satisfied). The same applies for other interventions when HLG association rules are listed.

delivery mechanism described in Chapter 3 is the result of an iterative design and evaluation process based on three different pilot studies which we describe in the rest of this section. All these studies had a similar set up: participants first studied a textbook chapter on the AC-3 algorithm and then wrote a pre-test to establish their initial understanding of the concepts covered in the chapter. Next, they used the new CSP applet with interventions to solve two CSPs, while their gaze was tracked with a Tobii T120 eye-tracker. Afterward, they wrote a post-test analogous to the pre-test. At the end, a qualitative evaluation of the relevant aspects of the interaction was done using a post-hoc questionnaire and a follow-up interview.

6.2.1 First pilot study

The first pilot was a Wizard-of-Oz study during which an experimenter triggered the interventions following a set of simplified rules based on the adaptation rules described in Section 6.1. The goal of the study was to pilot test the general two-level hints approach in terms of visibility, intrusiveness and the follow-rate of the interventions. Six computer science students (all second or third-year students) were recruited for this pilot study.

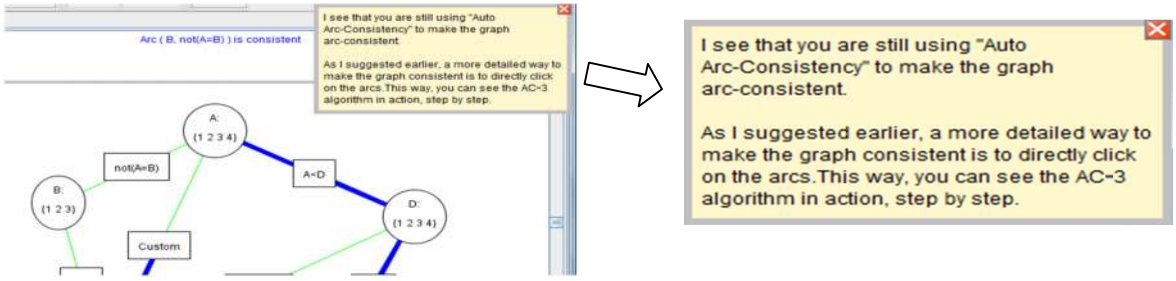


Figure 6-1 A hint suggesting the use of Direct Arc Click action with the interface highlights (left); and the content of the hint box (right).

The initial version of the two-level hints consisted of:

- A stationary message box at the top-right corner of the applet's window which was used to display the text messages, and
- An interface change for level-2 hints which was bolding the relevant interface element(s) (i.e., increasing the thickness of the border)

A level-2 hint as presented in this initial version of interventions is shown in Figure 6-1.

The results of the post-hoc questionnaire for textual hints were very encouraging as summarized in Figure 6-2. The participants did not find the hint messages intrusive or annoying. They found the messages easy to notice and useful in the process of interaction. Moreover, most of the participants reported following the instructions provided in the hints. The rest of this section will present quantitative results derived from action logs and eye-gaze data collected during the interaction.

Regarding visibility of the hints, out of 27 hints provided in total, 25 of them were attended to by the participants. One of two omitted hints was a level-1 hint given to participant 4 (P4), while she did not notice this hint, the subsequent level-2 of the same hint (with interface change) managed to get her attention. The second case was a level-2 hint given to P6, where he decided not to follow a level-1 hint prior to this hint and was given a level-2 hint. In this case, the interface change reminded him of the recommended action (Direct Arc Click) from the level-1 hint, thus he followed the hint without having to look at the hint box. These two cases underscore the importance of the two-level hinting strategy reinforced by interface changes.

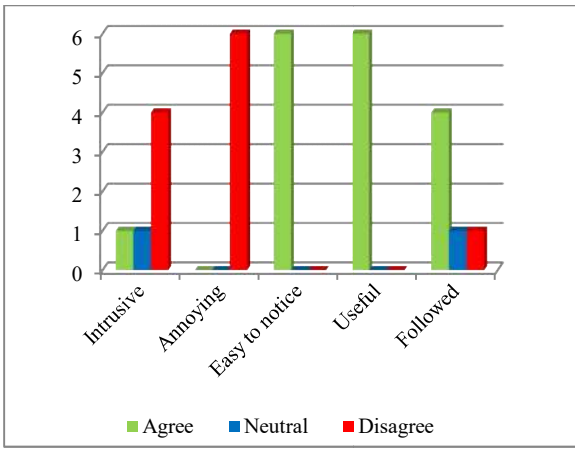


Figure 6-2 Reception of the text hints by participants

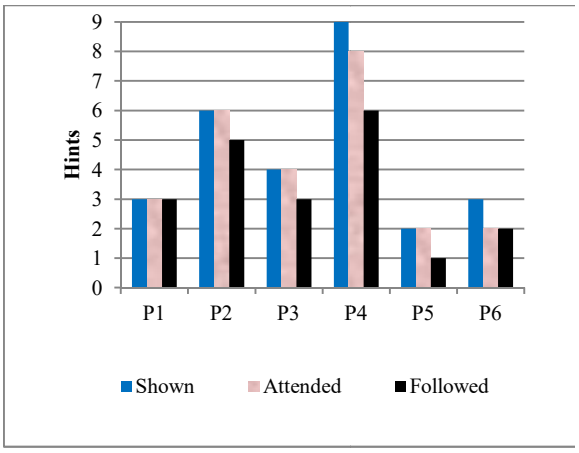


Figure 6-3 Number of hints shown, attended and followed for each participant

The number of hints shown, attended to and followed by each participant is illustrated in Figure 6-3. Out of 27 hints given, 20 were followed by the participants (74% follow-rate). Students, who show many detrimental behaviours, would get more hints. Such students are the target group that we want to help learn better from their interaction with the CSP applet. Therefore, P2 and P4 are of particular interest. Both of these participants reported finding the interventions relevant and useful. However, P4 did not follow every hint, and generally only followed the recommendations when repeated in the form of a level-2 hint. This is reflected in her self-assessment of how often she followed the hints as well (Table 6-3).

We also analyzed the average reading time of the hint messages for each participant, overall and for the hints they dismissed/followed (Table 6-3). We can observe that participants differ in terms of reading time for hint messages which –if necessary– can be further investigated as a guide for estimation of user-adaptive reaction time for hints (i.e., minimum time necessary for reading a hint before users can react to it). Another trend is that users who received more hints

also spent less time reading each hint (on average). This is expected as these users are the ones with sub-optimal interaction behaviours and this again shows the importance of the two-level progressive hinting strategy which gets more intrusive the second time a hint is provided.

Table 6-3 Hint rate, self-rated following of hints, and average reading time for each participant in the first pilot

	P1	P2	P3	P4	P5	P6
Followed Hints - Self-rated (1-5)	4	4	4	2	4	3
Avg. Reading Time (ms)	2814	1642	1547	925	2639.5	9460
Avg. Reading Time: Followed (ms)	2814	1530.6	1663	937.5	3464	8975
Avg. Reading Time: Dismissed (ms)	-	2199	1199	887.5	1815	9945
# Hints given	3	6	4	9	2	3

In summary, participants generally did not find the hint messages intrusive, and they rated them as useful. The interviews, however, uncovered various issues with visibility: for the textual hints, subjects reported that sometimes it was “hard to notice when a new message appeared”. For the highlights provided in the level-2 hints, some participants “did not notice them at all”; others saw them but failed to “make the connection with the textual hint”. These observations were confirmed by the eye-gaze data collected with the eye-tracker.

6.2.2 Design alternatives

To address the issues uncovered in the first pilot study related to interface changes, we came up with a set of design alternatives for how hints are presented. We added visual effects to the hint box in order to make sure students would easily notice when a new hint is presented (goal:

notice new hints). We also came up with alternatives for the interface changes (goal: notice interface changes) and tried to use visual effects to make it easy for students to create a connection between these changes and the text message presented in the hint box (goal: connect text and interface changes). We aimed to achieve these mentioned improvements while trying to keep the intrusiveness of the hints at an acceptable level.

To achieve the "notice new hints" goal we tried two alternative visual effects for the hint box inspired by (Gluck et al., 2007):

1- Moving Box (Figure 6-4): the hint box would originate from the center of the relevant area on the screen (i.e., center of the graph if the hint was related to the graph elements and the toolbar if the hint was related to an action on the toolbar) in minimized mode and grow and move to the top right corner of the screen reaching the normal size in its permanent location.

2- Sliding Box (Figure 6-5): When a new text message appears in the hint box, the box appears in the top right corner of the screen and slides down from the top incrementally displaying the text.

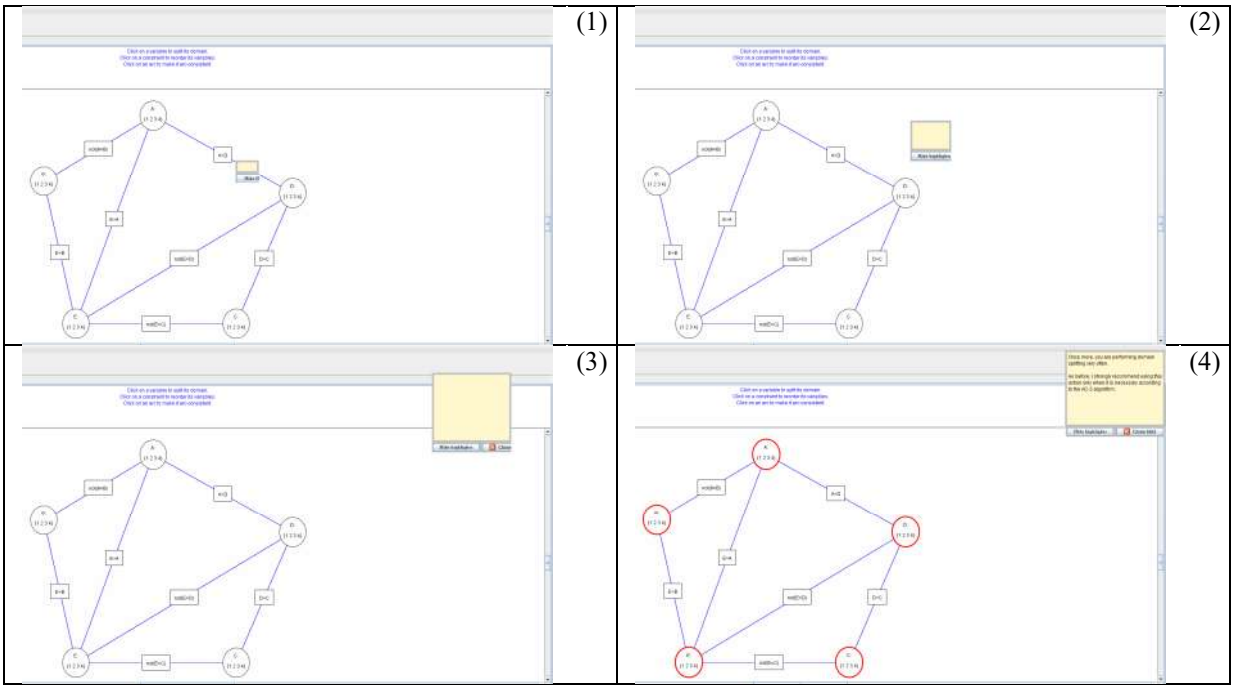


Figure 6-4 A Moving Box originated from the relevant point on screen

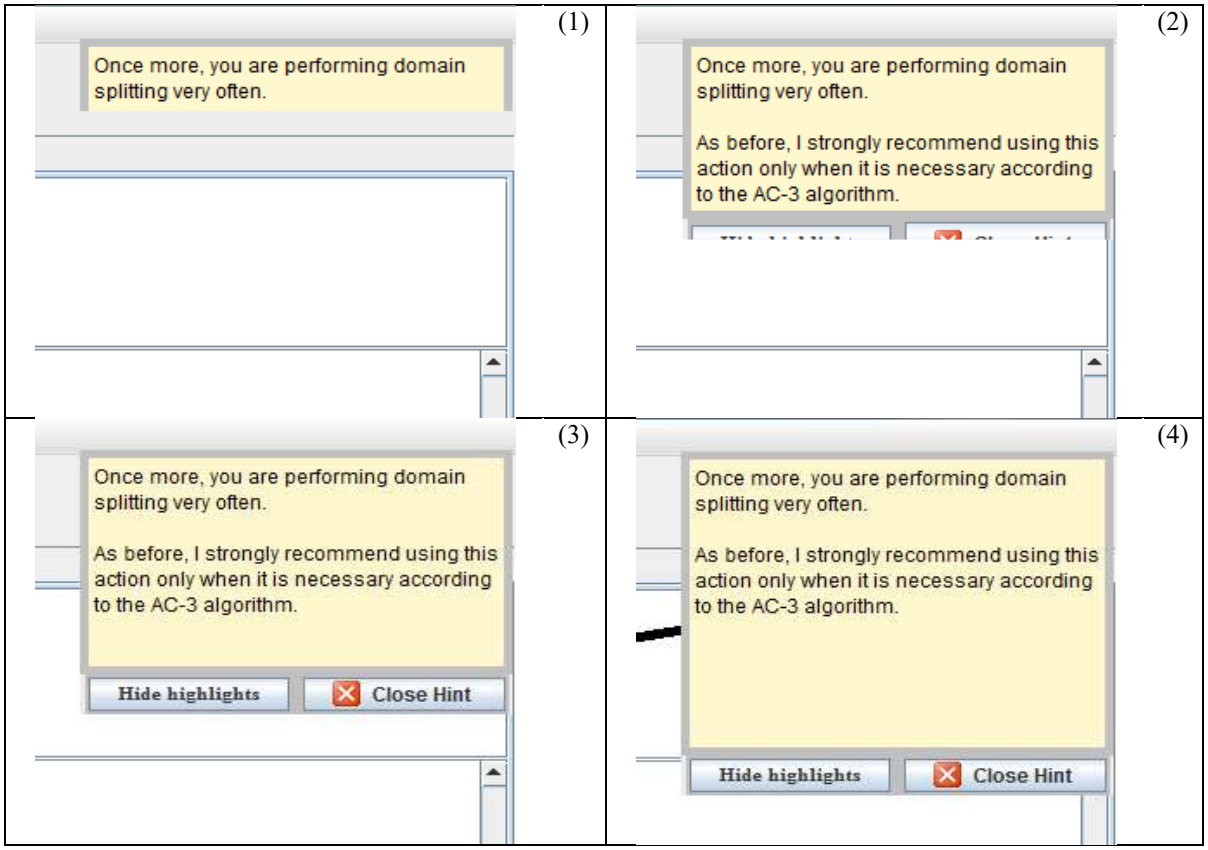


Figure 6-5 Sliding hint box

To achieve the “connect text and interface changes” goal for level-2 hints, we focused on creating a visual link, with these visual effects:

1- Arrow (Figure 6-6): after a new hint is displayed, an arrow is drawn from the hint box to the relevant area of the screen (the area is determined following the same strategy used for the moving box effect) and will disappear after a time interval.
2- Keyword highlights (Figure 6-7): the same highlighting effect used on interface element(s) is also used on relevant keywords in the text message.

We should note that in addition to creating a visual link between the text and the interface changes, the arrow effect also serves to draw user’s attention to the hint box thus helping to make sure the user realizes that a new hint is available for level-2 hints (“notice new hints” goal), however we considered this a secondary function for the arrow effect.

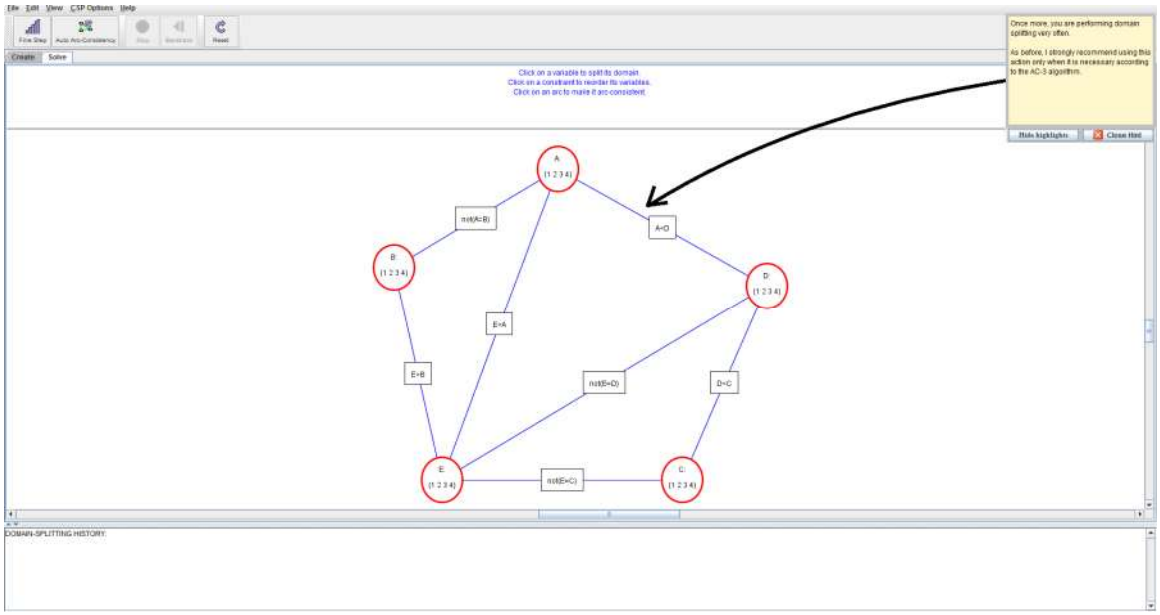


Figure 6-6 An arrow pointing to the relevant point on the screen

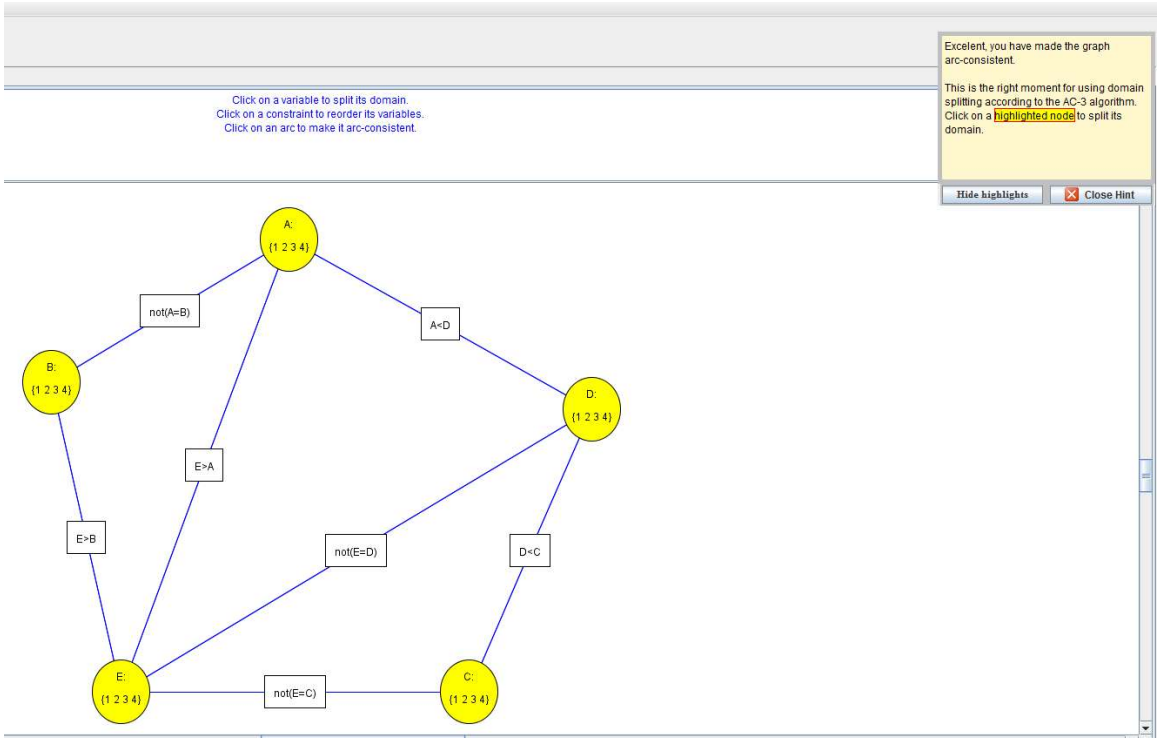


Figure 6-7 Keyword highlighting (with yellow highlight effect)

To achieve the "notice interface changes" goal for level-2 hints, we aimed to increase the attentional draw (Gluck et al., 2007) by using more prominent effects compared to the original version used in the first pilot study where we only increased the border size of the relevant items. Based on the list of available effects in increasing degree of the attentional draw (and intrusiveness) presented in (Gluck et al., 2007), we selected the following:

1- Yellow highlight (Figure 6-8): the relevant items are highlighted with yellow color.

2- Blinking (Figure 6-9): The blinking effect is generated by repeatedly increasing the border size of the relevant elements and resetting them back to normal.

In both cases, the effect disappears after the user performs any action in the relevant area of the screen (e.g., if the effect is related to any of the elements on the graph, then an action on the graph would make the effect disappear).

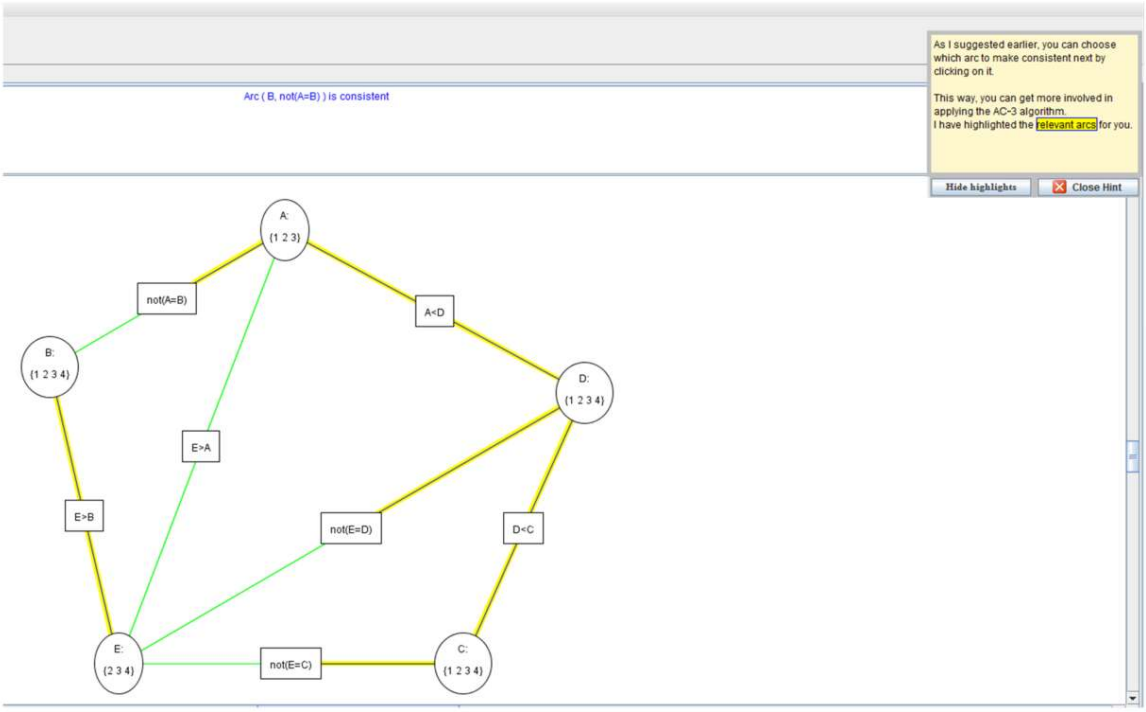


Figure 6-8 Yellow highlight effect for arcs, on a sample problem

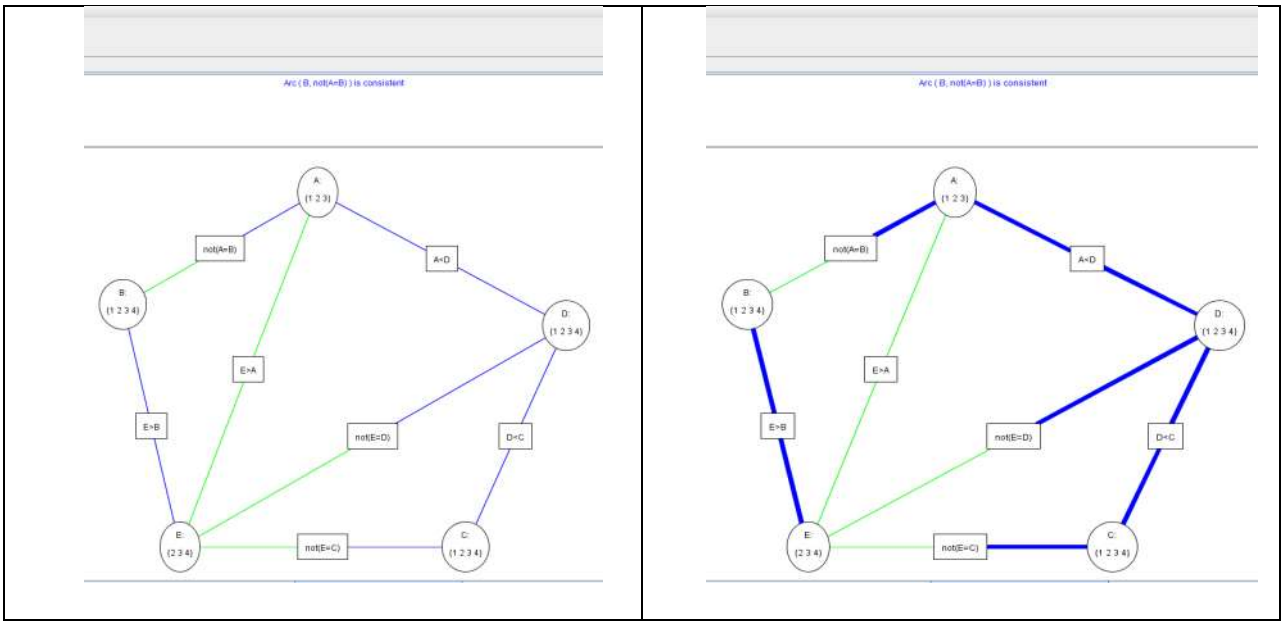


Figure 6-9 Blinking effect for arcs, on the same sample problem

Next, we describe the selection process we used for the design alternatives and updates to the final design.

6.2.3 Informal focus group

We combined the mentioned visual effects for "notice new hints" and "connect text and interface changes" goals to create 3 different alternatives (the keyword highlighting effect was combined with both sliding and moving box effects, however, the arrow was only combined with sliding box due to the visual noise caused by combined moving box and arrow animations that happen in opposite directions). Combining highlighting and blinking options for interface changes with these 3 alternatives on visual effects for hint box and the visual link would have resulted in 6 different conditions. Given the limited pool of participants for our studies (due to requirements for the participants described in Section 4.1), we decided to use an informal focus group setting to reduce the number of conditions by removing the alternatives where a consensus was reached by all participants.

The informal focus group consisted of 5 Computer Science graduate students who were all involved in HCI-related research. At the beginning of the 90-minute session, the goal of the hinting process was explained briefly and all 6 alternatives were shown to the participants. Then, each alternative was shown again and they were given a chance to experience it while working with the app in a typical use setting. Then the design alternatives were discussed in terms of pros and cons. Finally, participants were asked to pick one alternative for each aspect of the design (namely, visual effect for hint box, the blinking vs. highlighting effect for interface changes, and visual link for the hint message and interface changes). All participants preferred the moving hint box compared to sliding effect. They were mostly concerned that the sliding effect was too subtle, especially on larger display screens. There was no consensus regarding the blinking vs. highlighting effects for interface changes and highlighting was favored by 2 as the less intrusive alternative, others did not rule out blinking as a viable option. In terms of the "visual link", all

participants preferred the keyword highlight effect over the arrow as the more visually appealing option. Participants especially liked the fact that the text message explicitly mentioned that the relevant items were highlighted on the graph or toolbar.

Based on the findings of the informal focus group we decided to compare the blinking vs. highlighting effects in a within-subject design described next.

6.2.4 Second and third pilot studies

We ran two pilot studies to compare the blinking vs. highlighting effects for interface changes by collecting user preferences in a within-subject design with 2 conditions blinking and highlighting. Each user would start with one condition and switch to the other condition. The starting condition was balanced between the 2 conditions. We aimed to run these studies in a close to the real-world situation while making sure all users received different hints in both conditions. Therefore, we developed a version of the applet that used a random user model to trigger different hints with a uniform distribution. This study also provided an opportunity to test the intervention controller and intervention presenter components that were added to the original applet.

For this pilot study, we relaxed the requirements for participants and included the students who had already studied a course that covered the constraint satisfaction problems because we were not planning to measure the learning gain. As a result, at the beginning of the study, participants were given the option of studying the reading material on solving CSPs if they were not familiar with the concept or wanted to refresh their memory. Then, they watched an introductory video on the CSP applet. Afterward, the main task of solving two CSP problems using the CSP applet was explained to the participants (similar to the task described in Section 4.1). During the task, the participant would receive different hints selected randomly by the

random model. For the first problem, the interface changes were presented according to the starting condition, and the alternative condition was used for the second problem. Finally, a post-hoc questionnaire was given to the participant and an interview was conducted to collect user preferences between the 2 alternatives, comparing them from different aspects. The questionnaire asked participants to rate statements about each condition on a scale of 1-5 ("strongly disagree" to "strongly agree").

For pilot #2, we recruited a total of 5 participants after which we decided to conclude this pilot and make modifications to the software based on the feedback from participants to make the settings more realistic and improve the blinking effect. For pilot #2, the random model would trigger a hint at every hinting opportunity (every 10 actions by the user) which could be too high in a realistic scenario. Therefore we changed the model to trigger the hints based on the outcome of a binary event with equal chances, which reduced the average hinting frequency by 50%. One complaint about the blinking condition was that the blinking effect was too slow in such a way that it could be confused with other animations in the applet (e.g., when the user clicked on the arc to make it consistent). To rectify this issue, we decreased the blinking interval to half a second. We also added a button (the "hide/show highlights" button) to the hint box, to give users the ability to hide or display the interface changes manually in addition to automatic hiding mechanism originally implemented (Figure 6-10). After these changes, we conducted another pilot study (pilot #3) with 6 participants. Pilot #3 helped to test the modifications and confirm the findings of pilot #2.

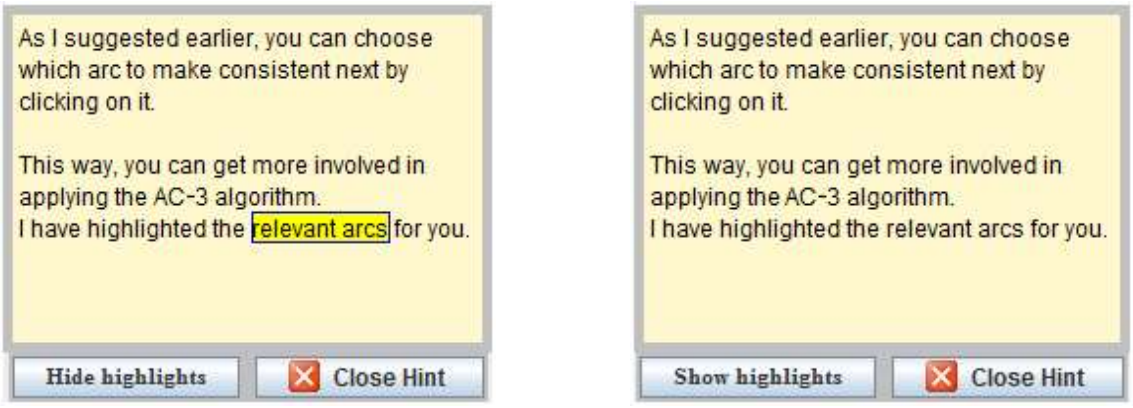


Figure 6-10 Multi-function button for showing/hiding the interface changes

6.2.5 Findings and Summary

In both pilots, highlighting the relevant items and the text keywords with yellow color was preferred overall (Figure 6-11). Participants found the blinking effect easier to notice and useful for detecting when a new hint was available, however, preferred the highlighting effect overall as they found blinking too intrusive and annoying. The highlighting with yellow color was also voted as the more effective way to create the visual link between text messages and the interface changes. The summary of user ratings aggregated across the two pilot studies is presented in Figure 6-11. Participants also rated the "hide/show highlights" button very positively.

To summarize, the final design of the intervention presenter component is the result of an iterative design process. After testing our original two-level hint design with a Wizard-of-Oz pilot study we modified the design to address 3 main issues raised by participants and confirmed by eye-tracking. More specifically:

- Visibility of new textual hints was increased by adding the motion and change in initial positioning of the hint box as described earlier.

- Visibility of the highlights for level-2 hints was increased by testing two alternative ways to emphasize the target elements, namely yellow highlighting vs. blinking. Yellow highlighting proved to be the preferred and more effective method based both on user's feedback as well as eye-gaze data.
- The connection between the level-2 hint messages and the related interface changes was emphasized by both mentioning in the hint text that the relevant elements are highlighted, and by using the same highlight effect on keywords in the text.

6.3 Summary

In this chapter, we presented the process of building the two main components for providing adaptive support in the CSP applet, namely the intervention controller and intervention presenter, based on the discovered association rules in the behaviour discovery phase. For the intervention controller component, we described how a rule triggers an intervention item and listed the intervention items extracted from the association rules. For the intervention presenter component, we described the iterative design process and pilot studies used to arrive at the final hint delivery process for the CSP applet.

Next chapter presents the results of the experimental evaluation of the new CSP applet which comes with adaptive support.

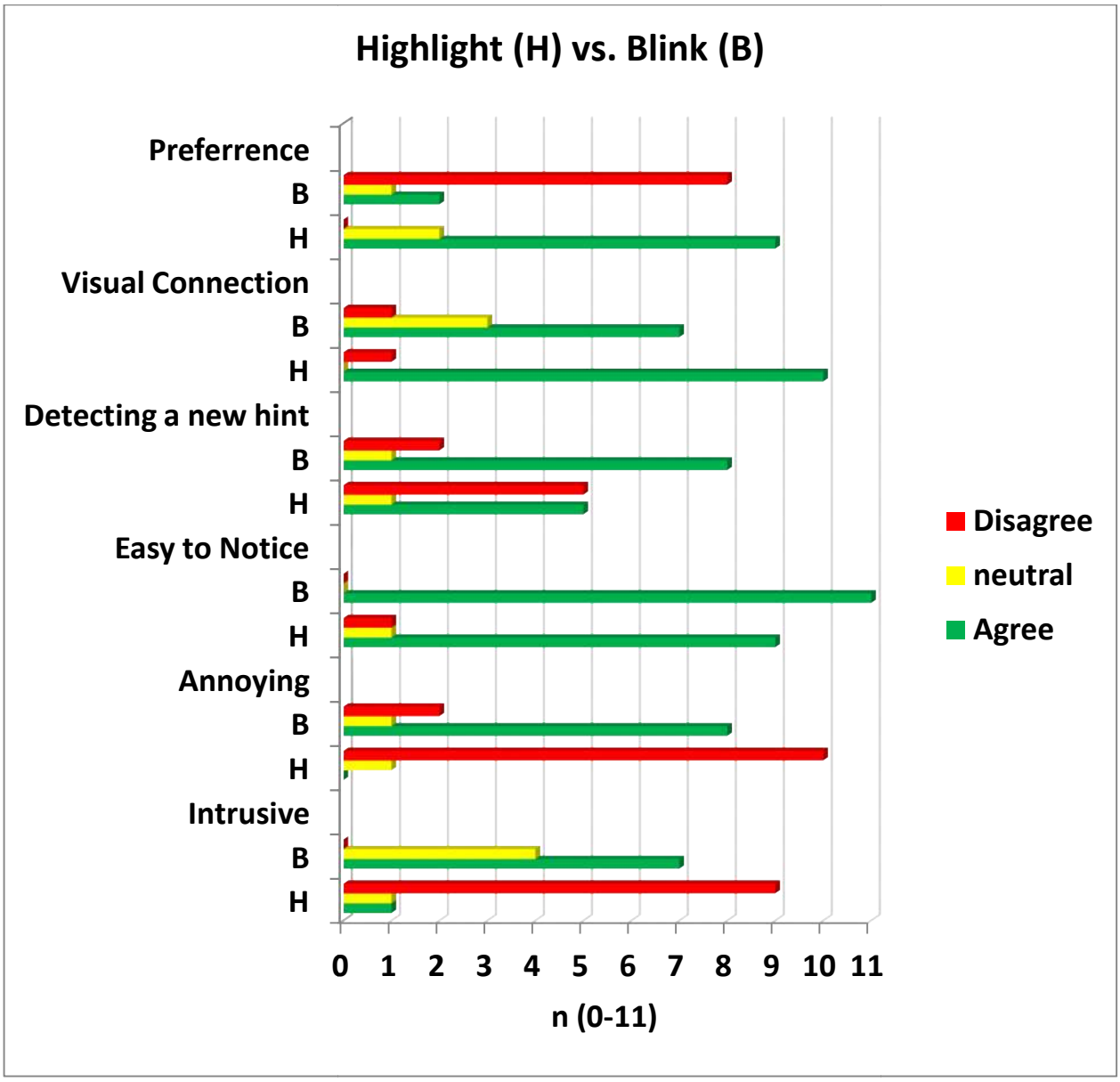


Figure 6-11 Summary of user preferences for pilot #2 and pilot #3

Chapter 7: Comprehensive Summative Evaluation

In this chapter, we describe the final evaluation of the UMA framework on the CSP applet. We evaluated the effect of the adaptive support provided by the intervention mechanism (described in Chapter 6) added to the CSP applet on the learning performance of users. In the rest of this chapter, we first provide details of the user study conducted to evaluate the new version of the CSP applet and then provide the results of the study in terms of both quantitative and qualitative measures. In this chapter we provide answers to the following questions posed in the Introduction chapter:

Q3: *Can the discovered behaviour patterns in [Q1] be used to derive adaptive interventions that are effective in improving user's learning performance?*

Q4: *Is the resulting adaptive version of the exploratory environment more effective than the non-adaptive version?*

7.1 User study

To investigate the effectiveness of the intervention mechanism, we conducted a user study that tested the following hypothesis:

H1: Under similar conditions, participants who work with the adaptive-CSP applet (i.e., receiving adaptive support, a.k.a. Adaptive condition) will have a higher learning performance compared to ones that work with the original CSP applet (Control condition).

7.1.1 Procedure

The experiment was a between-subjects design, fitting in a single session lasting at most 2 hours. There were two randomized experimental conditions: in the *Control* condition participants

worked with the original CSP applet; in the *Adaptive* condition participants used the adaptive-CSP applet[29].

During the study each participant: (1) studied a textbook chapter on the AC-3 algorithm; (2) wrote a pre-test[30] on the concepts covered in the chapter; (3) filled out a questionnaire about different aspects of attitude towards help seeking and completed an online test for reading comprehension speed; (4) watched an introductory video on how to use the main functionalities of the CSP applet; (5) used the CSP applet to solve three CSPs; (6) took a post-test[31] analogous to the pre-test; and (7) took a questionnaire on aspects relevant to both groups (described below). Participants in the Adaptive condition took an additional questionnaire on aspects specific to the adaptive-CSP applet. The study ended with a follow-up interview that solicited explanations for user ratings.

7.1.2 Study material

The study tests involved items selected from a standard bank of homework questions used in an introductory AI course at UBC. They required students to apply knowledge of different aspects of the AC-3 algorithm on selected CSP networks. The maximum possible score for both tests was 25 based on existing marking schemes.

[29] It should be noted that this design does not fully isolate the contribution of the hints from the adaptive delivery component. We did plan to have a third condition with hints triggered randomly, but we could not get enough users with the right background for a 3-way study. We chose not to use this random condition as control because we used it in our pilots and user feedback indicated irritation with it. Thus we felt that it would not be a good control due to its intrusiveness. This was later verified by a pilot study using the random user model (Section 7.4)

[30] Available in Appendix A.2

All study participants took a post-questionnaire[31] (*general questionnaire* from now on) in which they rated the following three statements (using a Likert scale from 1: strongly disagree to 5: strongly agree):

- I felt more confident answering the post-test questions compared to the pre-test questions (*improved confidence*)
- I think I did better in the post-test compared to the pre-test (*perceived learning*)
- I found the applet helpful in understanding the AC-3 algorithm (*helpfulness of the applet*)

The additional questionnaire administered to participants in the Adaptive condition (referred to as *intervention questionnaire*) included 18 items on different elements of the intervention mechanism, also ranked on a 5-point Likert scale. Pairs of similar statements were provided to evaluate hint messages and highlights in terms of *relevance*, *usefulness*, *noticeability*, *intrusiveness*, and *annoyance*:

- In general, the [hint messages | highlighting of the interface items] were appropriate given my behaviour (*relevance*).
- The [messages displayed in the hint box| highlighting of the interface items] were useful for me (*usefulness*).
- I easily noticed the [new messages displayed in the hint box| highlighting of the interface items] (*noticeability*).
- I found the [hint box| highlighting of the interface items] intrusive (*intrusiveness*).
- I found the [hint box| highlighting of the interface items] annoying (*annoyance*).

[31] Available in Appendix A.1

It should be noted that, in our first pilot, we did not have items for annoyance, because we thought intrusiveness would capture a user's negative perception of how the interventions interfere with their work. However, some participants who rated the hints as intrusive commented in follow-up interviews that they did not perceive these interruptions as negative; therefore a new set of items targeting annoyance was added.

The remaining 8 items focused on specific attributes of each element of the intervention mechanism. These items are:

- The messages displayed in the hint box were easy to understand.
- I usually followed the advice displayed in the hint box.
- Highlighting the interface items helped me notice that a new message has appeared in the hint box.
- Highlighting the interface items helped me find the right elements of the applet to work with.
- I could easily see the connection between the highlighted elements and the text shown in the hint box.
- I found the 'hide highlights' button helpful.
- I found the mechanism that automatically hid the highlights helpful.
- I found the 'show highlights' button helpful.

Screenshots of all the relevant intervention elements were provided in the questionnaire to facilitate recollection (general and intervention questionnaires are available in Appendix A-1).

7.1.3 Study tasks

The 3 CSP problems used in the study task were selected to provide incremental coverage of all relevant aspects of the AC-3 algorithm, as well as different outcomes. The first CSP had one

solution that could be found without domain splitting. The second CSP had no solution and required one domain splitting to reach this conclusion. The third problem had 3 solutions, which required multiple domain-splitting and backtracking actions to be found.

Participants were given an instruction sheet indicating that they needed to report the outcome of using AC-3 on each of the problems. They were given a minimum of 15 minutes for working on this activity but no maximum time limit was imposed.

7.1.4 Study participants

Thirty-eight university students ranging in age from 19 to 28 (11 female) participated in the experiment. We selected the number of participants by performing a power analysis (Lenth, 2006) a priori on the parameters of our experimental design, defined to detect a large effect size of $d = 1.0$ in terms of performance with 0.8 power. Considering the phenomenon observed by Bloom (Bloom, 1984), that the average student tutored on a one-to-one basis could perform two standard deviations better than students who learn in a conventional class setting, we decided to set an ambitious goal of improving the average learning performance of students by at least one standard deviation.

Participants were selected such that they did not have any knowledge on solving constraint satisfaction problems beforehand (a new concept); however, they had the computer science pre-requisite knowledge necessary to learn this concept (e.g., basic graph theory and algebra). We used flyers, email, and social media to recruit participants with these prerequisites at UBC.

7.2 Results for performance measures

We looked at the impact of study condition on three different measures of performance: *learning performance* from pre-test to post-test, as measured by the Percentage Learning Gains (PLG) defined in equation 4-1; *task performance,* which is a score out of 10 given based on the

number of correct solutions found for the 3 CSP problems using the applet (including identifying the CSP with no solution); *task time*, i.e., the time in minutes spent working on the 3 CSP problems using the applet; Summary statistics for the three performance measures and pre-test for each condition are shown in Table 7-1.

An independent samples t-test on the pre-test scores of the two conditions found no significant difference ($t(36) = .189$, $p = .851$, $\eta^2 = .001$), indicating that the random assignment of participants to conditions was successful in generating two groups with comparable initial knowledge. Also, the correlation between pre-test and PLG is not significant ($r = .178$, $p = .126$) indicating that other factors are behind the variation in the PLG.

In addition to studying the impact of experimental conditions on our performance measures, we are also interested in verifying whether the impact is moderated by a student's initial knowledge, as measured by pre-test. Thus, we ran a moderated multiple regression analysis (Cohen, West, Aiken, & Cohen, 2002) for each of our performance measures, with that measure as the dependent variable. Independent variables (IV) are *condition* (coded as -1 for Control and 1 for Adaptive), *pre-test*, and the *interaction* between condition and pre-test. This interaction is calculated by first centering pre-test (by subtracting the sample mean) and then multiplying pre-test and condition together. Alpha level is set to 0.05, and Pearson *r* is reported to show the effect size (small for .1, medium for .3, and large for .5 (Cohen, 1988)). Models are adjusted for family-wise error by applying the Bonferroni correction. Data screening did not suggest problems with assumptions of normality and linearity (e.g., excess kurtosis of -0.82 and skewness of 0.21 for the Control condition and excess kurtosis of -1.01 and skewness of -0.32 for the Adaptive condition are all well within the -2 and 2 range). The multicollinearity assumption was not violated by any of the models, based on the condition of Variance Inflation

Factors being smaller than ten[32] (O'Brien, 2007). For each independent variable, we report both standardized regression coefficient (β) and squared semi-partial correlation (sr^2).

Table 7-1 Descriptive Statistics for the performance measures

Measure	Pre-test (of 25)		Post-test (of 25)		PLG		Task Performance		Task Time	
Condition	Cont.	Adapt.	Cont.	Adapt.	Cont.	Adapt.	Cont.	Adapt.	Contr.	Adapt.
Mean	9.66	9.37	13.18	17.97	28.59	54.29	8.58	8.42	18.32	18.37
Std. Dev.	5.553	3.696	5.822	4.264	19.045	24.784	2.434	2.364	3.198	5.756

7.2.1 Results on learning performance

The overall regression model significantly predicted PLG, $F(3, 34) = 7.435$, $p = .001$, $R^2 = .396$, adjusted $R^2 = .343$. There was a significant PreTest×Condition interaction, $\beta = -.296$, $t(34) = -2.048$, $p = .048$, $sr^2 = .075$, with a medium effect size ($r = .331$). The interaction accounts for an additional 7.5% of the variance in PLG. There was also a significant main effect for condition $\beta = .517$, $t(34) = 3.878$, $p < .001$, $sr^2 = .267$, with a large effect size ($r = .554$), however the main effect for pre-test was not significant ($\beta = .128$, $t(34) = .886$, $p = .382$, $sr^2 = .014$).

The main effect for condition shows that the difference in learning performance of the two groups in the study is significant; with the Adaptive condition having greater learning performance compared to the Control condition (the average standardized difference is 1.034 which is slightly more than one standard deviation). This confirms our hypothesis H1.

The interaction effect indicates the effect of condition on PLG is moderated by pre-test. To better qualify this finding, we plotted the effect of condition on PLG at three different levels of

[32] VIF(PLG) = 1.66, VIF(Task Performance) = 1.15, and VIF (Task Time) = 1.05

standardized pre-test scores labeled (z-scores) as high (z=1), average (z=0), and low (z=−1); (Figure 7-1). The line equations are derived from the standardized β values. As shown in Figure 7-1 the effect of condition on PLG is highest for the low pre-test group and decreases as pre-test performance improves, i.e., the interventions are most effective for the students with lower initial knowledge. This is very encouraging for us because these are the students who need the most help.

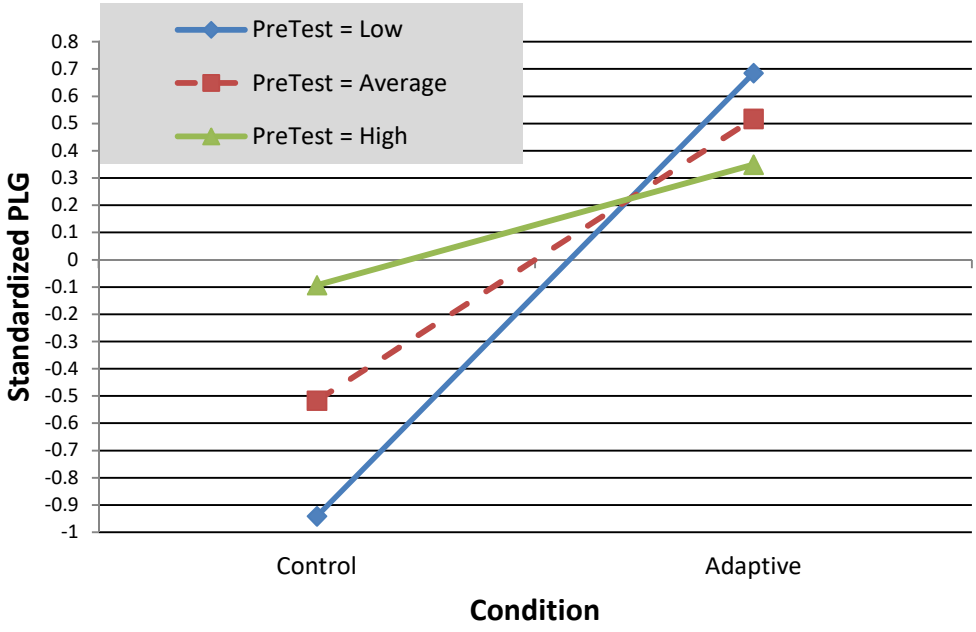


Figure 7-1 Interaction between PreTest and Condition

7.2.2 Results on task performance and completion time

The regression models with the aforementioned IVs did not significantly predict task performance, $F(3, 34) = 1.650$, $p = .196$, $R^2 = .127$, adjusted $R^2 = .050$ *or* task time, $F(3, 34) = .554$, $p = .649$, $R^2 = .047$, adjusted $R^2 = .038$. These results indicate that condition did not significantly explain the variance of task performance or task time, i.e., the two conditions are

not significantly different in terms of these two measures. The fact that participants in the Adaptive and Control conditions performed equally well in the task with the CSP applet, while achieving very different results for learning gain, confirms that unaided interaction with the CSP applet can lead students to find solutions without obtaining a clear understanding of the process. In other words, students can use or misuse the simulation to find the solution(s) to the provided problems, and achieve a high task performance level. As evidenced by the results here, high task performance level does not always translate to learning.

7.3 Results on interventions acceptance

The success of any adaptive support mechanism highly depends on the users' perception of its quality. To gauge this perception, in this section we will look at both objective and subjective indicators: respectively, the actual follow-rate for the hints and post-questionnaires ratings.

7.3.1 Hint follow-rate

For every hint received by a user in the Adaptive condition, we check the user model after the reaction window to see if the user is still showing the behaviour that triggered the hint. If the user was not showing that behaviour, the hint is marked as followed and vice versa. Figure 7-2 shows the descriptive statistics for the number of hints given in total, at level-1 and level-2, along with the follow-rate for each. One participant did not receive any hints, and thus it is not included in this analysis as well as in the analysis of subjective measures in the next section.

Table 7-2 Descriptive Statistics for Hints

	N	Min	Max	Mean	Std. Dev.
Number of Hints (Level-1 and Level-2)	18	1	15	7.06	4.65
Level-2 Hints	18	0	8	1.56	2.09
Follow-rate Level-1 (%)	18	16.67	100	62.96	32.95
Follow-rate Level-2 (%)	10	0.0	100	73.33	37.84
Follow-rate overall (%)	18	16.67	100	64.44	31.65

As shown in Table 7-2, the overall follow-rate is 64.4 percent, indicating a relatively good acceptance of the adaptive suggestions. Notably, the average follow-rate for hints increases from 63 percent for level-1 hints to 73.3 percent for level-2 hints. This validates the two-level design of our interventions.

To ascertain whether the number of hints received affects follow-rate, we ran a correlation analysis between these two measures. We found a very strong negative relationship ($r = -0.754$, $p < 0.001$) between follow-rate and number of hints given.

7.3.2 Qualitative evaluation

We first compare the user ratings on the general post questionnaire items, which were completed by both the Control and Adaptive conditions. Then, we discuss the ratings provided by participants in the Adaptive condition for a selected set of items from the interventions questionnaire.

7.3.2.1 User ratings for the general questionnaire

We used Mann-Whitney U tests to compare the ratings of the three items in the general post questionnaire. The distributions in the two groups did not differ significantly for *Improved*

Confidence (Mann–Whitney $U = 150.50$, $n_1 = n_2 = 19$, $P = .322$ two-tailed), *Perceived Learning* (Mann–Whitney $U = 142.50$, $n_1 = n_2 = 19$, $P = .223$ two-tailed), and *Helpfulness* of the applet (Mann–Whitney $U = 171.50$, $n_1 = n_2 = 19$, $P = .766$ two-tailed). The ratings were generally positive for all three items across conditions (see Figure 7-2). The positive ratings reflect the value of the CSP applet for visualizing the steps of the AC-3 algorithm compared to studying examples in the textbook as revealed by the follow-up interviews (e.g., It is "much easier to learn" from the applet; I could "see the outcomes in real-time" and "step-by-step"; I easily "learned from my mistakes").

Notably, however, there is a mismatch between the relatively high perceived learning for the Control condition and the actual learning performance of the participants in this condition. Possibly, perceived learning is positively influenced by their task performance, which as illustrated in the previous section was as good as that of the students in the Adaptive condition. The mismatch between perceived and actual learning is concerning because these students will not feel the need to explore more problems to improve their learning if left to their own devices.

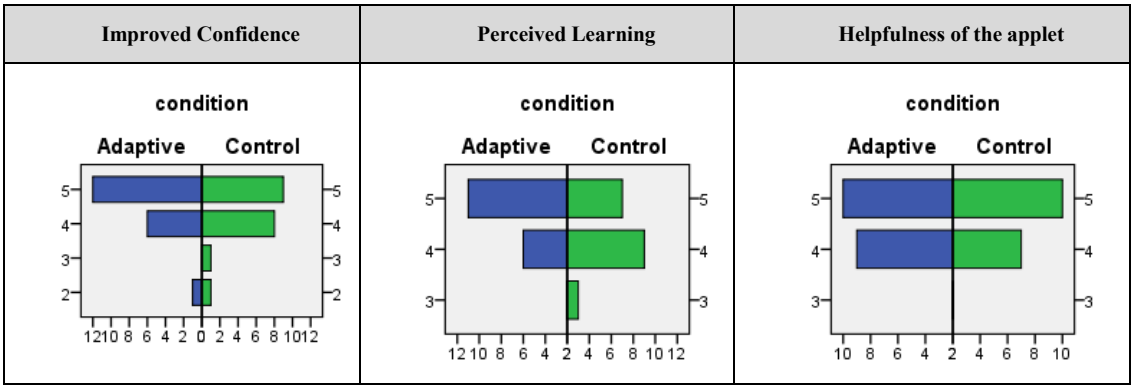


Figure 7-2 User ratings of the general items (5: most positive)

7.3.2.2 User ratings for the interventions questionnaire

In this section, we report user ratings for items on the intervention questionnaire that evaluated hint messages and highlights in terms of *relevance*, *usefulness*, *noticeability*, *intrusiveness*, and

annoyance (as described in Section 7.1.2). One participant in the Adaptive condition did not receive any hints; therefore, we only have 18 ratings for the hint messages. Also, due to the fact that highlights were only present for the level-2 hints, and only 8 of the 18 participants received a level-2 hint with highlights, we report ratings for hint messages and highlights separately.

The rating distributions for the items relating to hint messages are shown in Figure 7-3. To simplify the analysis in the following discussion we combined the "agree" and "strongly agree" ratings as *agree-all* and "disagree" and "strongly disagree" ratings as *disagree-all*. The majority of the users (over 67%) found the hints relevant to their behaviour, whereas 28% found them irrelevant. Usefulness received fewer negative ratings, with only 22% reporting that they did not find the hint useful, however, there were more neutral ratings, resulting in fewer users agreeing over usefulness (45%). Based on post interviews, one prominent reason given for negative rating of usefulness and relevance was the nature of the textual hints. Some users reported expecting the hints to give more explicit help on how to solve the problem at hand, rather than telling them how to use the applet, possibly because they focused on getting the job done (e.g., "I wanted to find the solution and the hints kept telling me to slow down") as opposed to trying to improve their understanding of the AC-3 algorithm, which is the task that is supported by the intervention mechanism.

Our design decision to increase the prominence of the appearance of the hint box (to ensure new hint messages are not missed) succeeded in improving noticeability of these hints, reaching 100% of agree-all rating, at the expense of only a small increase in ratings for intrusiveness (28% agree-all, which is 6% higher than the analogous ratings from the first pilot). Compared to results from our pilot studies, a higher number of users rated the hint box as annoying. In the follow-up interviews, the majority of those who reported annoyance indicated that their reason for such

rating was repeatedly receiving hints that they did not agree with (i.e., the alternating level-1 and level-2 hints given for a behaviour after the first level-2 hint was given for that behaviour). This happens when a user insists on not following a hint, and this hint stays ranked high in the list of active intervention items, indicating high relevance to learning performance. In addition to the negative ratings, we observe a trend in actual follow-rate where it declines for the repeated hints compared to the first two times a hint is delivered (i.e., first delivery of level-1 and level-2 hints). We cannot judge from the available data to what extent hints given repeatedly were actually justified (recall that the user model is not 100% accurate), but regardless, our results indicate that we should consider reducing the number of repeated hints in future versions of the applet[33]. In the post interview, none of the users attributed their annoyance to the animation used for the appearance of the hint box, which provides further positive feedback on the current design of the hint box.

Ratings for the corresponding questionnaire items for highlights were generally more positive than for hint boxes, with the exception of noticeability: *relevance* (88% agree-all, median = "agree"), *usefulness* (75% agree-all, median = "agree"), *noticeability* (88% agree-all, median = "agree"), *intrusiveness* and *annoyance* (0% agree-all, median = "disagree" for both items). Positive ratings for highlights show an effective balance between *noticeability* and *intrusiveness/annoyance*.

[33] Some ideas for this are presented in Section 9.2.1

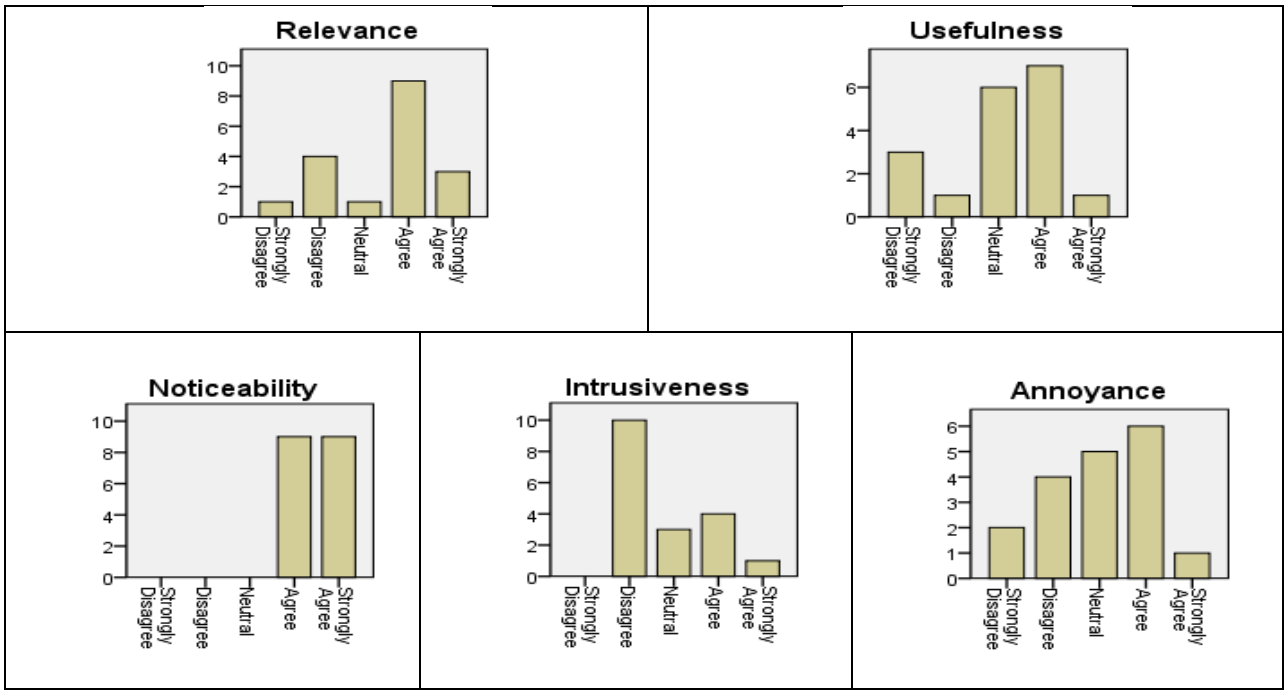


Figure 7-3 Ratings of selected items for the hint messages

The distribution of user ratings for the other 8 items in the intervention questionnaire is presented in Table 7-3. Other than self-reported hint follow-rate, none of the other items are rated negatively. The positive ratings presented in Table 7-3 show that the modifications for creating a visual link between hint messages and interface changes have been successful. Also, both the automatic and manual methods for hiding highlights are rated positively (last 3 items) although the lower number of ratings may imply that some users may not have noticed and used them. Considering the actual follow-rate of 64.4%, it is interesting that only 16.7% of users rated the item related to following the hints negatively.

Table 7-3 Distribution of ratings for specific items in the intervention questionnaire

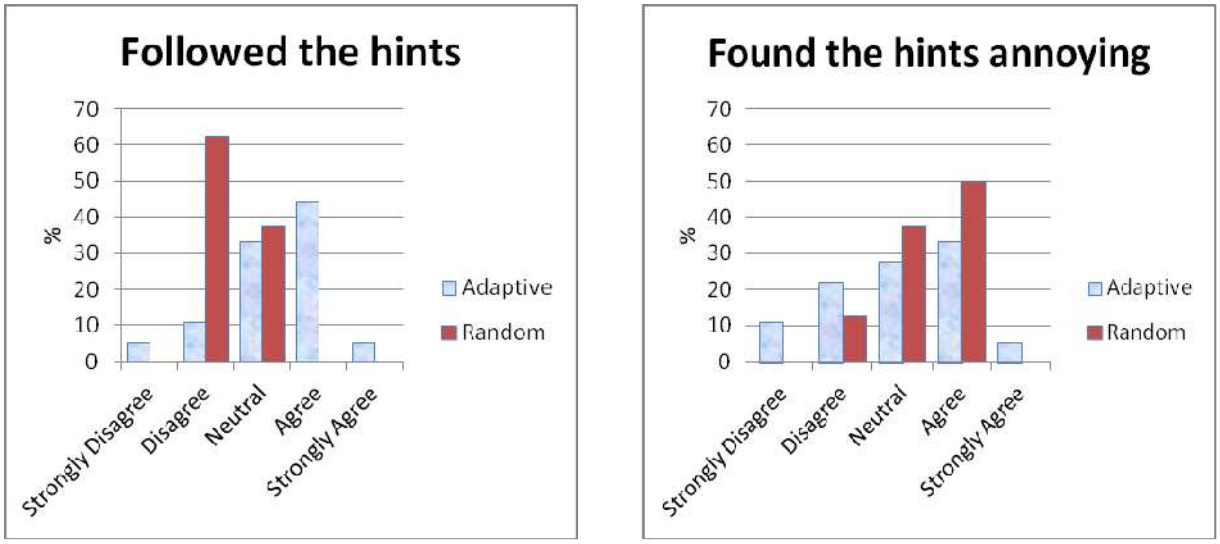
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The messages displayed in the hint box were easy to understand.	0	0	0	13	5
I usually followed the advice displayed in the hint box.	1	2	6	8	1
Highlighting the interface items helped me notice that a new message has appeared in the hint box.	0	0	3	2	3
Highlighting the interface items helped me find the right elements of the applet to work with.	0	0	1	4	3
I could easily see the connection between the highlighted elements and the text shown in the hint box	0	0	0	5	3
I found the 'hide highlights' button helpful	0	0	0	3	1
I found the mechanism that automatically hid the highlights helpful	0	0	3	1	4
I found the 'show highlights' button helpful	0	0	1	2	0

7.4 Pilot study with random user model

The initial design of the study included a third condition with hints triggered by the random user model that was used in pilot #3. However, we could not get enough users with the right background for a 3-way study and decided to run a 2-way study instead. We also decided not to use the Random condition as the control because user feedback in pilot studies indicated irritation with it. Thus we felt that it would not be a good control due to its intrusiveness.

We chose instead to conduct, at a later time, a further pilot study of the random user model with the goal to provide further insights on its acceptance. The study was structured similarly to the main study, with the only difference that instead of the classifier user model, a random user

model was employed. At each hinting opportunity (every 10 actions), this model would present a hint with 50% probability. If a hint was presented, it would be chosen at random among the set of available hints. For this pilot study, we managed to recruit 8 participants. As expected, the reaction to random hints was not positive. None of the participants agreed with the statement "I usually followed the advice displayed in the hint box" with 5 out of 8 disagreeing (62.5%) and the other 3 (37.5%), rating this statement as neutral. Compare this to 3 out of 18 disagreeing (16.7%) and 9 agreeing (50%) with the statement for the Adaptive condition. Only one participant (12.5%) did not find the hints annoying, while 3 (37.5%) rated this statement as neutral and 4 (50%) found the hints annoying (compared with a balanced rating of 6 agreements (33.3%) and 7 disagreements (38.9%) for Adaptive condition). Figure 7-4 shows the detailed distribution of ratings for the Random and Adaptive conditions for the above items in terms of percentage of participants.



(a) (b)

Figure 7-4 Distribution of ratings for (a) "Followed" and (b) "Annoying" in Random vs. Adaptive conditions

In terms of performance measures, while the sample size is too small to run statistical tests for comparison, we observed that in terms of PLG and Task Performance the Random condition is not performing better than the Control condition (Table 7-4). Perhaps the most indicative of the negative effects of random hints is the trend observed on Task Time measure which shows that participants in the Random condition are spending more time to finish the task while not performing any better in terms of learning.

Table 7-4 Performance statistics for the Random and Control conditions

Condition	n	Task Performance (out of 10)	Task Time (minutes)	PLG (percentage)
Control	19	8.58 (SD = 2.43)	18.32 (SD = 3.20)	28.59 (SD = 19.05)
Random	8	7.75 (SD = 2.71)	22.38 (SD = 7.41)	25.08 (SD = 31.89)

Given the above observations, we argue that selection of non-adaptive CSP for the Control condition was the better choice for a 2-way study. We should also point out that relevance of the hints (which is based on the user model in the Adaptive condition) is very important to convince users to follow the hints and potentially benefit from them. As one participant in the Random condition put it "the hints would be very helpful if they [were] shown at the right time".

7.5 Evaluation of the user model on the new data

As a final evaluation of the user model, we decided to use the interface-action data of the new 38 users that participated in the final study as a new test-set. We exported the final label assigned by the Adaptive-CSP's user model for each of the 19 users in the Adaptive condition. To obtain the labels for the 19 users in the Control condition, we fed their actions from the logs to a simulator that updated the user model and produced the final label predicted by the model. To

obtain the reference labels (i.e., ground truth) for these 38 users, we calculated their feature vectors and determined their cluster labels based on the distance of each vector from the centroids of the HLG and LLG clusters previously identified in the CPS-Action-110 dataset.

Recall that we are not using the conventional approach of splitting users based on learning performance into classes to label them (i.e., PLG-based median split). In fact, to demonstrate that labeling users solely based on median split on PLG is somewhat arbitrary, consider the following: for labeling the Control condition users by a median split, if the median of the users in the Control condition (29.03) is used, we get 9 users as HLG; while using the median of all users in the CSP-Action-110 dataset (45.83) would identify only 3 HLG users in the Control condition. A detailed comparison between the conventional PLG-based approach and our labeling approach is provided in Chapter 8.

The accuracy of the model on the new test-set is 78.95%, which is in line with the cross-validated accuracy reported in Section 5.3. It should be noted that while final accuracy of the user model on the CSP-Action-110 dataset is slightly higher than the accuracy achieved on the new testset, half of these users received hints which affect the time-based features (e.g., artificially increase the pause time after actions because user is reading the hint), that in turn affects the accuracy of the model. In fact, we are pleased with the robustness of the model considering the addition of hints.

We were also interested to see if users in the Control condition[34] who were labeled as HLG (n = 8) by the model performed significantly better than their counterparts that were labeled as LLG (n = 11). To test this hypothesis we compared the average PLG for the two groups using Welch's

[34] We excluded users in the Adaptive condition as their average learning performance was boosted by the hints as reported in Section 0

t-test. Among the users in the Control condition, there was a significant difference for average learning performance between the HLG (M = 40.85, SD = 15.28) and the LLG (M = 19.87, SD = 16.88) groups; T(16.08) = 3.17, $p = 0.006 < 0.05$, with a large effect size ($d = 1.30$). This finding is interesting and encouraging given the small sample size.

7.6 Application of the UMA framework on a different ELE

Following the successful evaluation of the UMA framework on the CSP applet, we started applying the framework to a more complex simulation on electric circuits called Circuit Construction Kit (CCK). CCK supports over 20 different action types, which can be applied to several different circuit components with a variety of outcomes, originating a large set of possible interface actions (over 120 actions) that allows users to build, modify and test different electric circuits. Given the complexity of the environment, a more complex representation of user interaction events was necessary (Kardan, Roll, & Conati, 2014).

The behaviour discovery and user classification phases of the framework have been successfully applied to the user action logs. Specifically, multiple representations for interactions have been compared and evaluated in terms of accuracy of user classification (Fratamico et al., 2017). The results of user classification are promising and consistent with those obtained with the CSP applet, with the strongest classifier user models beating the baseline as early as 30% into the interaction (baseline accuracy = 68.8%) and all models achieving over 80% accuracy at the end of the interaction. The average over-time accuracy, which measures the accuracy during the interaction at different intervals[35], for the best model was 76.5%. The usefulness of extracted rules from the behaviour discovery phase for informing adaptive support has been established in

[35] See Section 5.3 for more details on calculation of average over-time accuracy.

(Fratamico et al., 2017). The next step of this project is generating the personalized adaptations and evaluating the resulting adaptive support system. These findings are an exciting step towards establishing the generality of the proposed framework in the context of exploratory learning environments[36].

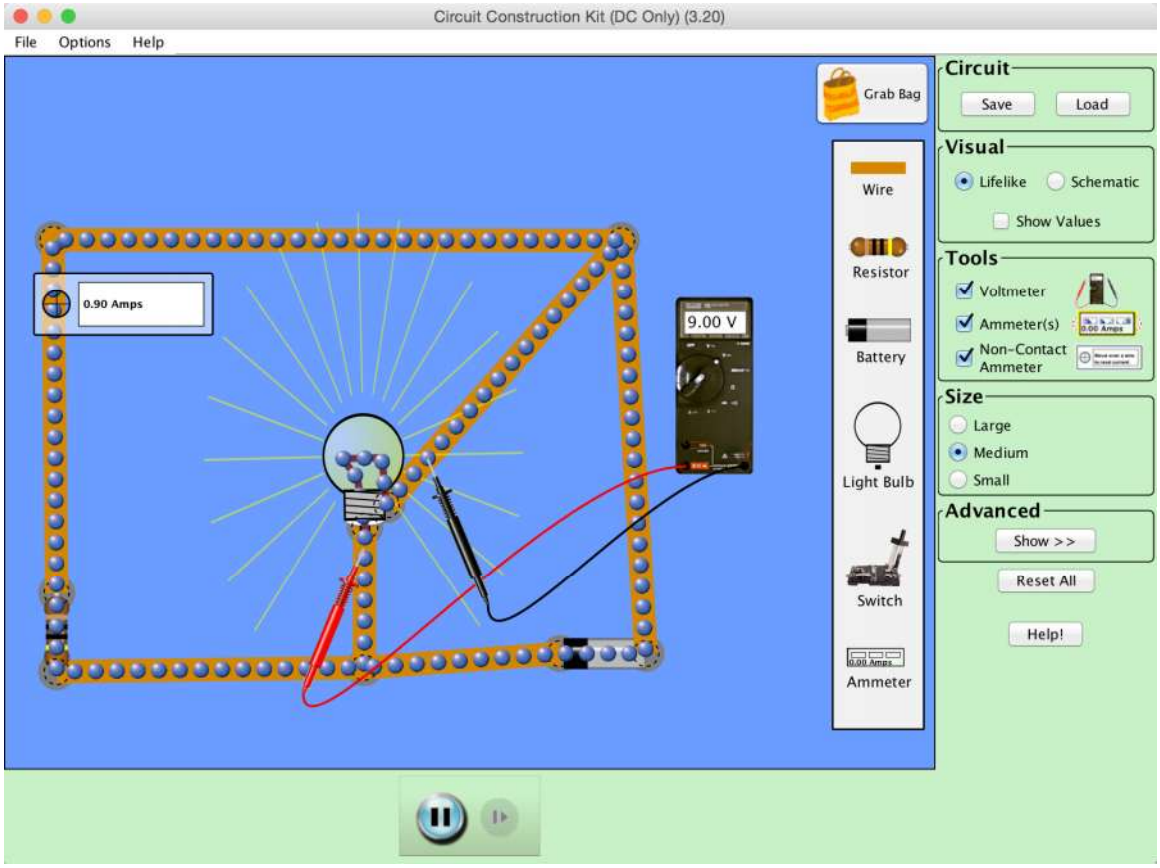


Figure 7-5 A screenshot of the CCK environment

7.7 Summary

In this chapter, we presented the formal evaluation of the final step of the UMA framework, namely providing user-adaptive interventions designed to guide students in learning at best through open-ended exploration, as tested on an exploratory learning environment called

[36] It should be noted that using the UMA framework with the CCK simulation required extending the approach with a process to add levels of knowledge engineering (albeit simple) to the creation of the feature sets necessary for user modeling.

AIspace CSP applet. There is extensive evidence that not all students can learn well from interactive simulations, due to the rather unstructured and open-ended nature of the interaction that they support. Our work is the first to show formal results on the effectiveness of providing adaptive support for open-ended exploration in an interactive simulation. More interestingly, our approach relies on the strategies that are automatically learned from previous learner interaction data.

A formal evaluation of these interventions against the original version of the CSP applet not only shows very encouraging results for both learning performance of users and their perception of the system, but also provides valuable insights for further improvements.

Students in the two experimental conditions performed similarly in terms of time on task and task performance[37], but students in the Adaptive condition learned significantly more than students in the Control condition based on pre- and post-tests. Furthermore, the effect of interventions on learning is even more pronounced for students with lower levels of initial knowledge. In terms of users' acceptance of the intervention mechanism, there was a positive attitude towards the applet in general and users followed 64.4% of the recommendations given to them. The users' ratings of the interventions were generally positive; however, they indicated that the number of repeated hints should be reduced since that was a source of annoyance for users.

We also reported the findings of a pilot study where we used a random user model to provide hints to students to isolate the effect of getting any hints from receiving them adaptively, and the acceptability of such setting by users. The qualitative results provided evidence that adaptive

[37] As mentioned earlier, it is possible to misuse the simulation to find the solution(s) to the provided problems, and achieve a high task performance level without learning the concepts.

delivery of hints plays an important role in their effectiveness and earning user's trust of the system.

Additionally, we provided a brief report on applying the UMA framework on a more complex ELE for physics called CCK (Fratamico et al., 2017). The positive results of that work in terms of user classification accuracy and usefulness of the discovered patterns for providing adaptive support, provides valuable evidence towards the generality of the UMA framework.

In conclusion, the findings reported in this chapter provide ample evidence for positive answers to questions [Q3] and [Q4] posed in the Introduction chapter of this thesis (see Section 1.2).

Chapter 8: Mining Eye-gaze Data for Building the User Model

This chapter presents an experimental evaluation of eye-gaze data as a source for modeling user's learning in ELEs. We compare the performance of classifier user models trained only on eye-gaze data vs. models trained only on interface actions vs. models trained on the combination of these two sources of user interaction data. Our findings show that including eye-gaze data as an additional source of information to the CSP applet's user model significantly improves model accuracy compared to using interface actions or eye-gaze data alone.

In this chapter we provide some insights into the following question posed in the Introduction chapter:

Q5: *What is the value (if any) of eye-gaze data as another source of data for the classifier user model?*

8.1 Data preparation and preprocessing

Eye-tracking data can be rather noisy when collected with eye-trackers that, like the Tobii T120 used here, do not constrain the user's head movements (Goldberg & Helfman, 2010). In this section, we briefly explain the process we used to deal with two sources of noise in our dataset. This validation process is crucial to ensure that the data reliably reflects the attention patterns that users generated while working with the CSP applet.

The first source of noise relates to the eye-tracker collecting invalid samples while the user is looking at the screen, due to issues with calibration, excessive user movements, or other user-related issues (e.g., eyeball shape). Thus, eye-gaze data for each user needs to be evaluated to ascertain whether there are enough valid samples to retain this user's data for analysis. The second source of noise relates to users looking away from the screen either for task-related reasons (e.g., looking at the instruction sheet) or due to getting distracted. During the looking-

away events, the eye-tracker reports invalid samples similar to when there is a tracking error on the user's gaze, even though there was no gaze to track. Thus, sequences of invalid samples due to looking-away events must be removed before starting the validation process of user's actual eye-gaze samples. One source of looking-away events was when participants were done with the first CSP problem and wanted to switch to the second problem[38]. They had to look at the instruction sheet to find the name of the second problem (Figure 8-1). Looking-away events were automatically detected when the user's gaze moved out of the screen gradually, by calculating the trajectory of fixations heading outside the screen. Automatic detection, however, is not possible when the user's gaze abruptly moves away from the screen. These events were manually identified by an investigator using videos of the user recorded during the study.

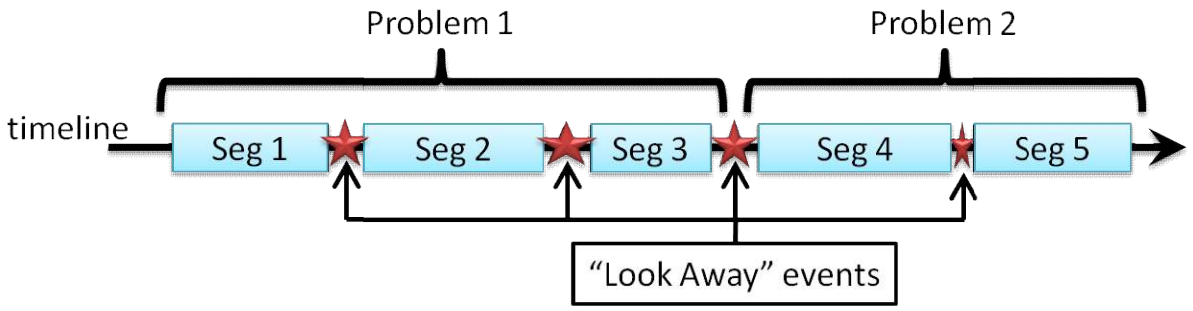


Figure 8-1 A sample timeline showing segments and "looking away" events

Detection of looking-away events resulted in the partitioning of the remaining eye-gaze samples into sequences occurring between two such events (*segments* from now on, see Figure 8-1). The next step was to analyze the validity of these gaze segments. In particular, we needed to set a "minimum data quality" threshold to discard segments and users that did not meet this threshold from the dataset. We use this threshold to determine, for each user in our dataset: (*i*) whether there are enough valid samples in the user's complete interaction, represented by the

[38] As described in Chapter 4, during the main task, participants solved 2 problems with increasing difficulty using the applet.

aggregation of her eye-gaze segments; if so, (*ii*) whether there are sufficient valid samples in each segment. This second step is to avoid accepting cases in which a large number of the invalid samples in an overall valid interaction are concentrated in a few segments, making the eye-gaze data in these segments unreliable.

We determined the threshold by plotting the percentage of segments that got discarded for different threshold values. The threshold value of 85% was selected because it is where the percentage of discarded segments starts to rise sharply (Figure 8-2). Figure 8-3 shows the percentage of samples left after discarding the invalid segments based on the 85% threshold. For all users except one, more than 90 percent of the samples were kept. The average duration of each user's interaction with the CSP applet only changed from 16.7 (SD = 9.0) to 16.3 (SD = 8.8) minutes. Next, we will explain the eye-gaze features calculated for each user.

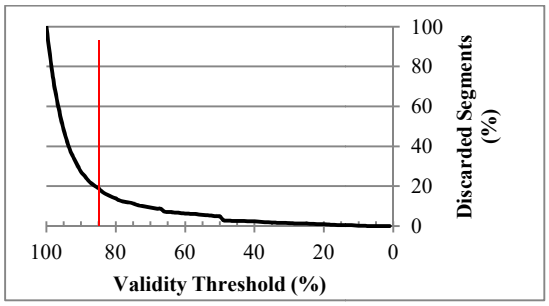


Figure 8-2 Percentage of segments discarded for different threshold values

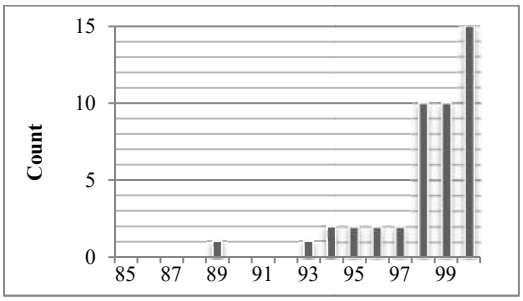


Figure 8-3 Histogram of users with different percentage of segments left after removing the invalid segments

8.2 Eye-gaze features

An eye-tracker provides eye-gaze information in terms of *fixations* (i.e., maintaining eye-gaze at one point on the screen) and *saccades* (i.e., a quick movement of gaze from one fixation point to another), which are analyzed to derive a viewer's attention patterns. As mentioned in the related work section, previous research on using eye-gaze information for assessing learning in

ELEs relied on tracking one specific attention pattern, predefined a priori (Amershi & Conati, 2009; Conati & Merten, 2007). In contrast, in our analysis, we use a large set of basic eye-tracking features, described by (Goldberg & Helfman, 2010) as the building blocks for comprehensive eye-data processing. These features are built by calculating a variety of statistics upon the basic eye-tracking measures described in Table 8-1.

Table 8-1 Description of basic eye-tracking measures

Measure	Description
Fixation Rate	Rate of eye fixations per milliseconds
Number of Fixations	Number of eye fixations detected during an interval of interest
Fixation Duration	Time duration of an individual fixation
Saccade Length	Distance between the two fixations delimiting the saccade (*d* in Figure 8-4)
Relative Saccade Angles	The angle between the two consecutive saccades (e.g., angle *y* in Figure 8-4)
Absolute Saccade Angles	The angle between a saccade and the horizontal (e.g., angle *x* in Figure 8-4)

Of these measures, *Fixation Rate*, *Number of Fixations*, and *Fixation Duration* are widely used (e.g., (Canham & Hegarty, 2010; Hegarty et al., 1995; Jarodzka et al., 2010; Loboda & Brusilovsky, 2010)); we also included *Saccade Length* (e.g., distance *d* in Figure 8-4), *Relative Saccades Angle* (e.g., angle *y* in Figure 8-4) and *Absolute Saccade Angle* (e.g., angle *x* in Figure 8-4), as suggested in (Goldberg & Helfman, 2010), because these measures are useful to summarize trends in user attention patterns within a specific interaction window (e.g., if the user's gaze seems to follow a planned sequence as opposed to being scattered). Statistics such as sum, average, and standard deviation can be calculated over these measures with respect to: (*i*)

the full CSP applet window, to get a sense of a user's overall attention; (*ii*) specific areas of interest (AOI from now on) identifying parts of the interface that are of specific relevance for understanding a user's attention processes.

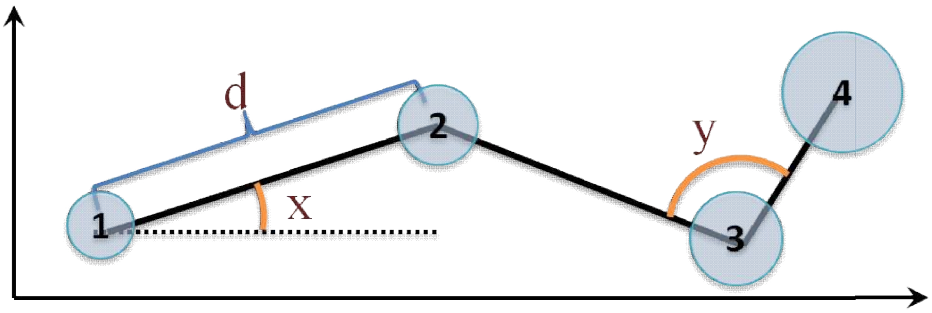


Figure 8-4 Saccade-based eye measures

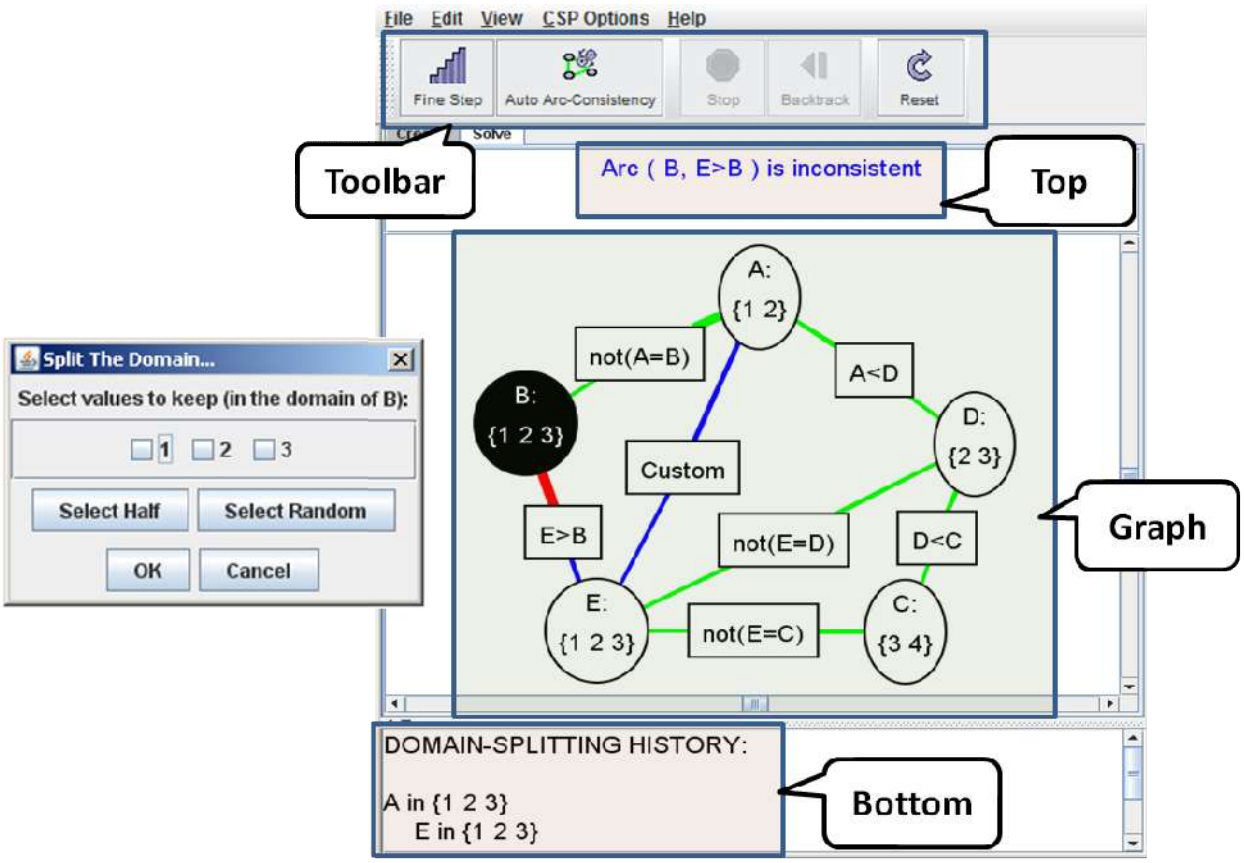


Figure 8-5 Areas of Interest for the CSP applet

We defined four AOIs for our analysis, corresponding to the areas that provide conceptually different functionalities in the CSP applet. Rectangles corresponding to these AOIs are shown in Figure 8-5. One AOI covers the region of the applet toolbar that includes action buttons (*toolbar* AOI); one covers the main graph panel (*graph* AOI); another covers the part of the top panel where the description of every step of the algorithm is displayed (*top* AOI); and the last covers the part of the bottom panel that displays domain splitting information (*bottom* AOI).

Table 8-2 shows the set of eye-gaze features calculated from the eye movement measures in Table 8-1 over the full CSP applet window. Table 8-3 shows the set of features calculated for each of the four AOIs. As the table shows, the two sets are different. For the AOIs, we added features that measure a user's relative attention to each AOI: *Proportion of Total Number of Fixations* and *Proportion of Total Fixation Duration* give the percentage of the overall number of fixations and fixation time, respectively, that were spent in each AOI. We also added features that quantify gaze transitions between different pairs of AOIs (Goldberg & Helfman, 2010) (including from an AOI to itself), as a way to capture the dynamics of a user's attention patterns. Transitions are represented both in terms of total number (*Number of Transitions between pairs of AOIs* in Table 8-3), as well as a proportion of all transitions (*Proportion of Transitions between pairs of AOIs*). Adding the aforementioned AOI-specific features substantially increases the overall number of features considered. In order to keep this number manageable, for the AOIs we did not compute saccade-based features, which are less commonly used than fixation-based features in eye-tracking research. In total, we included 67 features, 11 for the full CSP window, and 56 for AOI.

Table 8-2 Derived eye-tracking features for the full CSP applet window

Fixation Rate
Total Number of Fixations
Sum of Fixation Durations
Mean and Std. Dev. of Fixation Durations
Mean and Std. Dev. of Saccade Length
Mean and Std. Dev. of Relative Saccade Angles
Mean and Std. Dev. of Absolute Saccade Angle

Table 8-3 Derived eye-tracking features for each of the four AOIs

Fixation Rate
Total Number of Fixations
Proportion of Total Number of Fixations
Mean Fixation Durations
Proportion of Total of Fixation Durations
Highest Fixation Duration
Number of Transitions between pairs of AOIs
Proportion of Transitions between pairs of AOIs

8.3 Eye Movement Data Analysis Toolkit (EMDAT)

Based on the works described in Sections 8.1 and 8.2, we developed and published Eye Movement Data Analysis Toolkit (EMDAT), an open-source toolkit for gaze data processing. The main functionalities of EMDAT include:

1- computation of a large number of gaze features some of which are not typically provided by existing commercial and open source gaze-processing software, and

2- sophisticated data preprocessing and validation not available in other gaze processing software. In addition, EMDAT has been designed with generalizability in mind and can be reused for a variety of applications. Simple configuration files allow experimenters to quickly adapt the Python library to their needs.

Compared to the existing well-established gaze processing software, such as Tobii Studio, a commercial product, and OGAMA, an open-source GUI-driven package (available at:

www.ogama.net), EMDAT (as of version 2.1.1) added some important features (e.g., features related to saccade and transitions between AOIs) and its embedded preprocessing and validation module ensure reliability of the results. While EMDAT is customizable in various ways, the provided best-practice settings file enables researchers with different levels of experience with eye-gaze data to start using it with little effort.

Since the release of version 2.1.1, the development of EMDAT has been continued by other members of the Intelligent User Interfaces group in Laboratory for Computational Intelligence (LCI) at UBC.

8.4 Interface actions and eye-gaze data

Considering the additional cost of acquiring eye-gaze interaction data (i.e., the eye-tracker and related software) and extra effort needed for preparing the eye-gaze data (due to extra noise), we were interested in comparing this source of user interaction data with interface actions. Thus, in the rest of this chapter, we describe the setting we used to compare the interface actions vs. eye-gaze data vs. a combination of the two on the Eye-and-Action-45 dataset.

8.4.1 Comparison dimensions

The interaction data used as features by a classifier user model to perform on-line user classification can include a variety of sources. As we discussed at the beginning of this chapter, we want to compare using features based on interface actions vs. eye-gaze data vs. a combination of the two. We also want to evaluate the effectiveness of each of the two major components of our classifier user model: (1) using the hybrid approach (described in Section 3.2.2.1) to generate the training set for the classifiers (i.e., groups of users with labels that describe their learning performance) compared to a conventional approach; (2) using our proposed rule-based classifier for learning vs. other available classifiers (see Section 3.3). Thus, we have three dimensions in

our evaluation: the *feature set*, the approach for *training set generation,* and the *type of classifier*. In the rest of this section, we describe each of these three evaluation dimensions.

8.4.2 Different feature sets for classification

The first set of features consists of statistical measures that summarize a user's interface actions (ACTION dataset from now on[39]). We calculated usage frequency for each action, as well as mean and standard deviation of the time interval between actions (similar to other action datasets in Section 4.5) for a total of 12,308 actions. As described in Section 2.4, there are 7 actions available on the interface resulting in 21 features (none were highly correlated).

The second set of features captures user's attention patterns using eye-gaze information collected by the eye-tracker (EYE dataset from now on), namely fixations (i.e., maintaining eye-gaze at one point on the screen) and saccades (i.e., a quick movement of gaze from one fixation point to another). The features were derived by computing a variety of statistics (sum (total), average, standard deviation, and rate) as appropriate, for the measures shown in Table 8-2 and Table 8-3. Unlike the ACTION dataset, of the initial 67 features in the EYE dataset, we found and removed 16 features that were highly correlated ($r > 0.7$), reducing the final number of eye-related features to 51.

Finally, the third set of features (ACTION+EYE dataset) is obtained by combining the two feature sets described above. For each user, the ACTION and EYE feature vectors are concatenated to form a new vector with 72 features. This process generated a dataset with 45 data-points (participants) with 72 dimensions (features).[40]

Given these three datasets, we want to test the following hypothesis:

---

[39] This label is used to refer to the interface action portion of the Eye-and-Action-45 dataset. Not to be confused with the CSP-Action-65 and CSP-Action-110 datasets used in previous chapters.

[40] No additional highly correlated features were found in the ACTION+EYE dataset.

H1: Combining both eye-tracking and interface action data significantly enhances the performance of the resulting user model, as opposed to using either eye-tracking or interface actions data alone.

8.4.3 Different approaches for training set generation

As mentioned earlier, the first step in our approach for building a classifier user model is to identify groups of users that interact similarly with the learning environment and then label these groups based on the learning performance of their members, in order to provide the training set for the classifier. As pointed out in Section 3.2.2.1, we used the hybrid approach for clustering the EYE dataset[41]. The hybrid approach finds the best cluster-set (in terms of the sum of within-cluster distances) with a significant difference in learning performance.

When determining the optimal number of clusters with the hybrid approach using the three different feature sets described in Section 8.4.2 (ACTION, EYE, and ACTION+EYE), we found that two was always the optimal number of user groups (i.e., clusters), but with slightly different composition. We use Fleiss' kappa (a measure of agreement among more than two raters) for comparing the three different sets of user labels thus generated and found high agreement (kappa = 0.701). This kappa value shows that the two groups detected using each feature set share the same core of users (supporting the relevance of using clustering to detect these groups), with few users that are labeled differently when using different sources of data (showing that there is non-overlapping information captured by each source). For illustration, the size and performance measures associated with the two clusters generated by the hybrid approach applied to the

[41] As mentioned in Section 3.2.2.1, using the hybrid approach on datasets where basic clustering produces clusters with statistically significant difference in learning (e.g., interface actions data in our case) would result in finding the same cluster-set. This feature enables us to replace the basic clustering approach with the hybrid approach for all cases. Essentially, all results presented in the previous chapters using the basic clustering, would be repeated using the hybrid approach.

ACTION+EYE dataset is shown in Table 8-4, where LLG stands for Low Learning Gain and HLG stands for High Learning Gain. The difference in PLG is significant ($p = 0.017 < 0.05$) with a medium effect size ($d = 0.625$).

When the performance measure of interest for classification is available (in our case, PLG), the conventional method for creating a training set of labeled classes is to divide the performance spectrum into different ranges and assign users within each range into one group. Thus, in our evaluation, we want to compare our hybrid approach for generating the training set against the standard approach that relies solely on PLG. We generate what we call the PLG-based training set by dividing users into two groups based on the median of the PLG measure (45.83). Table 8-4 reports the size and PLG measures for the corresponding groups.

Table 8-4 Descriptive statistics of the training sets generated via different methods

		Hybrid on ACTION+EYE	PLG-based
HLG	Number of users	19	22
	Average (std. dev.)	53.29 (SD = 22.79)	68.27 (SD = 12.39)
LLG	Number of users	26	23
	Average (std. dev.)	32.45 (SD = 39.33)	15.40 (SD = 30.29)

When grouping users together, the hybrid approach relies on both PLG as well as the similarity in user interaction data as opposed to only relying on PLG. Thus, we argue that it can generate better performing user models since the user models can only rely on user interaction data when classifying users. This is the second hypothesis we will test in our evaluation:

H2: The hybrid approach for training set generation outperforms the conventional PLG-based approach in terms of user model performance.

8.4.4 Different types of classifiers

Our goal is to evaluate the rule-based classifier generated by our user modeling framework. Thus, we compare its performance with a battery of ten different classifiers available in the Weka toolkit on the EYE, ACTION, and ACTION+EYE datasets. These classifiers are C4.5, Support Vector Machine, Linear Ridge Regression, Binary Logistic Regression, Multilayer Perceptron, as well as Random Subspace and AdaBoost with different base classifiers. We tested the 10 Weka classifiers on each of the three datasets, and report the results for the classifier with the highest performance, which we will simply refer to as the Weka classifier. The third hypothesis tested in this study is the following:

H3: The rule-based classifier will have better performance compared to the best Weka classifier on each dataset.

8.5 Results and discussion

In this section, we present the evaluation results across each of the three dimensions described in the previous section. We compare the performance of the rule-based and Weka classifiers described in the previous section in terms of their average over-time accuracy in classifying new users as high or low learners. This means that the interaction features for a new user are calculated over incremental time intervals and the classifier is asked to provide a label for this user at the end of each interval. In Section 5.3, classifier accuracy was calculated after each user action, because only actions were used as data sources. Here, however, we have two different data sources, which provide information at different rates (typically length of a fixation is much shorter than the time between two interface actions). Thus, we compute current accuracy of the classifier at intervals of 30 seconds, i.e., long enough for observing at least one user action and a fair number of fixations. Then, to be able to combine accuracy data across users (with different

interaction durations), we retrieve current accuracy after every one percent of user interaction, calculating 100 accuracy points for each user.

We use 9-fold cross-validation for calculating the performance of the classifiers. Table 8-5 summarizes the over-time accuracy of the two classifiers on the three feature sets (ACTION, EYE, and ACTION+EYE) using both the hybrid and the PLG-based approach to generate the training set. We also report the average Cohen's kappa value for agreement between the actual labels and the labels predicted by the model. Cohen's kappa accounts for agreement by chance (Ben-David, 2008) and is useful here for comparing performance across different dimensions, because the size of the classes generated by the PLG-based approach and by the hybrid approach on each feature set are slightly different, changing the probability of agreement by chance in each case.

A 3 (feature set) by 2 (training set approach) by 2 (classifier type) ANOVA with kappa scores as dependent measure shows significant main effects for each factor ($F(1.43,198) = 294.27$ for feature set; $F(1,99) = 398.02$ for training set; $F(1,99) = 329.98$ for classifier type, with $p < 0.001$ for all factors).

Table 8-5 Average over-time performance results for different training sets, classifiers, and feature sets. The best performance in each column is indicated in bold.

Training Set	Classifier	Measure	Feature Set		
			ACTION	EYE	ACTION+EYE
PLG-based	Weka	Accuracy	51.18	57.62	58.18
		Kappa	0.027	0.144	0.157
	Rule-based	Accuracy	57.24	64.29	62.2
		Kappa	0.134	0.283	0.245
Hybrid	Weka	Accuracy	79.87	71.49	77.24
		Kappa	0.359	0.384	0.522
	Rule-based	Accuracy	**84.04**	**81.76**	**84.51**
		Kappa	**0.471**	**0.614**	**0.675**

For post-hoc analysis, we used pair-wise t-tests with Bonferroni adjustment using the estimated marginal means for each factor. Pair-wise comparisons over the *feature set* factor show that the models trained on the EYE+ACTION dataset outperform the models trained either on EYE or ACTION feature sets ($p < 0.001$), thus supporting H1. Pair-wise comparisons over the *training set* factor show that the hybrid approach outperforms the PLG-based approach ($p < 0.001$), thus supporting H2. Finally, pair-wise comparisons over the *classifier type* factor show that the rule-based classifier significantly outperforms the Weka classifier ($p < 0.001$), thus supporting H3. The findings show that we were able to extend our user modeling framework with an effective training set generation approach (H2), and the updated framework is able to build models that employ interface actions and eye-gaze data effectively (H3), reinforcing the validity of our findings regarding the added value of eye-gaze data (H1).

8.6 Ensemble model for combining EYE and ACTION features

The superior performance generated by the feature set that combines eye-gaze and action information indicates that there is an advantage in leveraging both data sources. Thus, we decided to investigate whether we could further this advantage by using a more sophisticated approach to combine eye-gaze and action information. In particular, for each combination of training set (hybrid and PLG-based) and classifier type (rule-based vs. Weka) we created an ensemble classifier (Baker, Pardos, Gowda, Nooraei, & Heffernan, 2011) that classifies a new user by using majority voting among the three following classifiers on the ACTION+EYE dataset: one trained using only the action-based features subset, one trained using the eye-based features subset, and one trained over the complete ACTION + EYE feature set. This ensemble model benefits from the added information captured by the eye-gaze data (if any) by being able to correctly classify the user in some of the cases where the classifier trained solely on the action-based features fails. Moreover, in some cases combining the features in the way that it is done in the previous section on the ACTION+EYE dataset introduces some noise in the dataset, thus diluting the information value gained. In such cases, classifiers trained on the eye-based subset and an action-based subset will not be affected and will be able to capture characteristics of each user as detected by each data source. Therefore, we hypothesize that:

H4: Each ensemble model outperforms the individual model equivalent to it (i.e., the model with the same classifier type and training set generation approach).

Table 8-6 shows the performance results for the ensemble models (measured by kappa scores). In order to evaluate the performance of the ensemble models vs. the individual models described in the previous section, we performed a 2 (model type) by 2 (training set approach) by 2 (classifier type) ANOVA with kappa scores for the ACTION+EYE dataset as the dependent

measure. Here, we are only interested in testing to see whether there is a main effect for the model type factor (i.e., individual vs. ensemble). The analysis shows a significant main effect for the model type factor ($F(1,99) = 165.420$, with $p < 0.001$). Post-hoc analysis using pair-wise t-tests with Bonferroni adjustment shows that the ensemble models significantly ($p < 0.001$) outperform their individual model counterparts thus supporting H4. Particularly, we are interested in the best performing individual model (rule-based model trained using hybrid training set) and its ensemble equivalent, where in addition to improved average over-time performance (86.56% vs. 84.51%), the ensemble model exhibits a more balanced performance across the HLG and LLG classes as well (85.33% and 87.52% for the ensemble vs. 79% and 88.54% for the individual model respectively).

Table 8-6 Average over-time performance results for different training sets and classifiers for the ensemble models, in terms of kappa scores

Training Set	PLG-based		Hybrid	
Classifier	Weka	Rule-based	Weka	Rule-based
Kappa	0.194	0.315	0.585	0.725

Considering the ultimate goal of providing adaptive interventions to the users during their interaction, we are also interested in having a user model that can achieve an acceptable accuracy in early stages of the interaction. Thus, we plotted the over-time accuracy of the rule-based ensemble model trained using hybrid training set in Figure 8-6. Performance of the majority class classifier is also plotted as the baseline. The model achieves 80% accuracy in both classes after observing 22 percent of the interaction (Figure 8-6), which shows that this model is highly reliable for providing adaptive interventions during the user interaction.

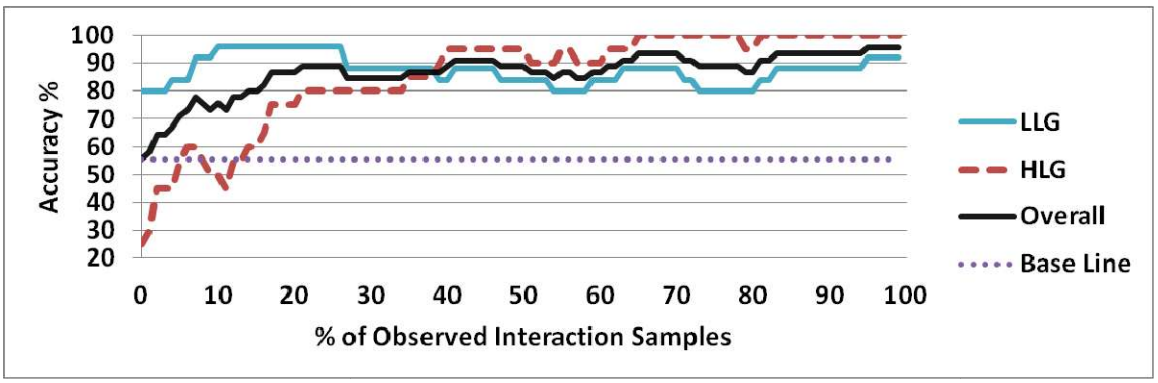


Figure 8-6 Over-time performance of the rule-based ensemble model

8.7 Summary

In this chapter, we presented an experimental evaluation of eye-gaze as an additional source of user data for modeling user's learning in an ELE, the CSP applet. The main finding is that eye-gaze data, when used as an additional source of user data in combination with the interface actions, significantly boosts the average over-time performance of the classifier user models trained to distinguish students who learned well from students who did not. These findings provide an answer to [Q5] posed in the Introduction chapter.

We also presented the details of our approach for generating training sets from the user data as applied to the eye-gaze data (i.e., the hybrid approach). The hybrid approach which extends the clustering approach used in the previous chapters[42] enabled us to effectively utilize eye-gaze data when building classifier user models. The evaluation results obtained here demonstrated that using the hybrid approach leads to models with significantly higher performance compared to a conventional alternative (PLG-based).

[42] As mentioned in Chapter 3, by design, the hybrid approach is completely compatible with the clustering approach used in the previous chapters and produces the same clusters when applied to the interface-action data.

Chapter 9: Conclusion

The main research goal of this thesis was to devise and evaluate a framework for adding adaptive support to Exploratory Learning Environments (ELEs). The main idea was that an ELE with user-adaptive interventions would be more effective compared to its non-adaptive counterpart. We thoroughly tested this idea by building and evaluating user-adaptive support for an existing ELE (i.e., the AIspace CSP applet).

9.1 Thesis contributions

The work described in this thesis helps advance the research in educational data mining, user modeling, and user adapted interactions particularly Intelligent Tutoring Systems (ITS) in different aspects. The main contributions of this thesis are the following:

9.1.1 First in-depth and complete evaluation of using data mining for user modeling and providing user-adaptive interventions to support open-ended interaction with ELEs

We proposed the User Modeling and Adaptation (UMA) framework that provides a data mining based solution for providing user-adaptive support to students in an exploratory learning environment. UMA framework applies our modified version of clustering called the hybrid approach to identify groups of students who interact in the same manner with our test-bed ELE (AIspace CSP applet) and detect the successful vs. non-successful groups (i.e., classes). Then, association rules mining is used to identify the prominent behaviour patterns of each class. These rules are used to build a classifier user model that is able to classify users into the discovered classes as they interact with the ELE. Finally, the CSP applet was furnished with an intervention mechanism that was built based on the discovered behaviour patterns following an iterative design process. The intervention mechanism, based on the output of the classifier user model,

provides adaptive support to the students to help them improve their learning from their interaction with the ELE.

We conducted both quantitative and qualitative evaluations of the components of the UMA framework as follows:

- For the hybrid approach, we compared the performance of the classifiers trained on feature-sets generated using the hybrid approach labels, with feature-sets generated by the common approach of labeling users based on their test scores in Chapter 8 and showed that using labels generated by the hybrid approach leads to significantly higher classification accuracy. **We also demonstrated that using the hybrid approach leads to models with significantly higher performance compared to a conventional alternative.**
- For UMA's rule-based classifier user models, we used average over-time accuracy[43] to measure the performance of the models. Our proposed classifier user model used by the CSP applet archives 78.2% cross-validated accuracy after only observing the first 25 percent of the interaction which provides an acceptable level of confidence for triggering adaptive interventions based on the user model. We also compared the accuracy of the rule-based classifier with other off-the-shelf classifiers available in a popular data mining toolkit in Chapter 8. The performance of the rule-based classifier on different datasets was superior to other classifiers when using the average over-time accuracy measure.
- To evaluate the effectiveness of the adaptive interventions provided to users based on the UMA-generated rules and user models, we measured the improvement in user's learning performance due to provided adaptive interventions compared to the non-adaptive version of

[43] This measure is more appropriate for the task of providing adaptive support because it favors classifiers that can achieve acceptable and consistent accuracy earlier in the interaction when less data is available as opposed to classifiers that can achieve very high accuracy but require much more data to get to an acceptable level of accuracy.

the CSP applet. Our experimental evaluation of the adaptive support mechanism added to the CSP applet shows more than one standard deviation improvement in learning performance for those students who used the new adaptive version of the applet vs. the students who used the original non-adaptive version.

There is extensive evidence that not all students can learn well from interactive simulations due to the rather unstructured and open-ended nature of the interaction that they support. Thus, several researchers have been working on how to address this problem by providing real-time, personalized user support. However, our work is the first to show formal results on the effectiveness of this approach. Moreover, as reported in Chapter 7, the UMA framework has been successfully applied to another ELE for physics that supports more complex interactions, providing evidence for the generality of the approach.

The methodology proposed in the UMA framework is easily applicable to any ELE provided that the interaction of students is captured in a meaningful way (e.g., for a more complex interface, more detailed representation of interaction events might be necessary). Furthermore, the built-in checks in the UMA framework (i.e., significance testing for clusters and calculation of cross-validated accuracy for the classifier user model with a built-in baseline) allows the researchers to get a quick quantitative feedback on the quality of the representation of the interaction data. Finally, the use of association rules, along with the ranking algorithm provided in the adaptive support phase, makes it easier for researchers to focus on developing interventions for the most relevant and impactful patterns.

9.1.2 Investigating the value of eye-gaze data for capturing student learning in ELEs

In the final part of the thesis, we focused on evaluating eye-gaze interaction data as an additional source of information for modeling and predicting learning performance of users. We

ran a user study to collect both user action and eye-gaze data along with learning performance information to conduct an experimental evaluation. The main finding was that eye-gaze data, when used as an additional source of user data in combination with the interface actions, significantly boosts the average over-time performance of the classifier user models trained to distinguish students who learned well from students who did not.

During the process of evaluating the predictive value of eye-gaze data in ELEs, we managed to propose improved methods for cleaning up and validating the noisy eye-gaze data.

9.1.3 Practical contributions

In addition to the main contributions discussed earlier, several practical contributions have also been made during this work. The most important ones are:

- **A new dataset containing the interface actions and eye-gaze data of 45 users interacting with an exploratory learning environment (AIspace CSP) made available to the EDM research community.**
- **A user modeling framework for ELE with the ability to generate and evaluate the performance of user models on different user interaction data.** In addition to the data mining functionalities provided in this framework, it is also able to utilize the data mining functionalities provided in the Weka data mining toolkit[44] (a publicly available collection of machine learning algorithms used for data mining), which expands its usability for mining and analyzing user data collected from different environments. This framework is currently used in other research projects at the Intelligent User Interfaces group in Laboratory for Computational Intelligence (LCI) at UBC (Fratamico et al., 2017).

[44] http://www.cs.waikato.ac.nz/ml/weka/

- **The initial versions of the Eye Movement Data Analysis Toolkit (EMDAT) for analyzing eye-tracking data.** EMDAT calculates different eye-gaze measures from the raw eye-gaze data exported from an eye-tracker. The main advantages of EMDAT include: (*i*) computation of a large number of gaze features some of which are not typically provided by existing commercial and open source gaze-processing software, and (*ii*) sophisticated data preprocessing and validation not available in other gaze processing software. This toolkit is currently used, maintained, and expanded by different members of the Intelligent User Interfaces group in LCI. It is publicly available and has been used by the research community (e.g., Intelligent Computer Tutoring Group at University of Canterbury, New Zealand).

9.2 Limitations and Future Directions

9.2.1 Further personalization of the hint delivery

While the overall perception of the interventions provided in the adaptive-CSP applet was generally positive, further personalization of the hint delivery could address the concerns raised by some users, for instance, reducing the number of repeated hints given to each user. To alleviate this, we can have multiple versions of text messages for each hint, and alternate between them each time a hint is delivered. However, ultimately, the applet should be able to tell when to stop delivering a certain intervention to a user, potentially by adopting a maximum for the number of times a hint is delivered in a session. Other areas for personalization include changing the frequency of hints delivered, the degree of prominence of the hint box movement animation, and even the length of the text message (if alternatives are available) based on classification confidence of the user model.

9.2.2 Enhancing results of students with higher level of prior knowledge

While on average, the interventions improved learning performance for all students, the students with the highest prior knowledge (i.e., pre-test score) benefited less from the interventions. One possible explanation is that the interventions were too shallow for these students. More analysis is needed to investigate the improvement of interventions with the goal of enhancing results for these students.

9.2.3 Considering combination of patterns for providing interventions

As mentioned in Chapter 6, there are specific features related to the standard deviation of time spent after actions for which we could not find any suitable interventions. It is rather easy to describe the behaviour that results in a low vs. high value for these features, i.e., users are consistent vs. selective/irregular in the amount of time they spend after the target action. However, it is confusing and impractical to tell users to be more consistent (for low standard deviation) or selective (for high standard deviation) in the amount of time they are spending after an action. We avoided using the combination of patterns in this thesis, however, one possible way to use these patterns is to look at the evolution of these features over-time, in combination with the corresponding average time spent after each action, and explore providing interventions that would indirectly prevent a user from drifting to a detrimental pattern.

9.2.4 Further application of the UMA framework

The UMA framework has already been successfully applied to two different ELEs (CSP applet and CCK simulation) and on two different sources of user interaction (interface actions and eye-gaze data) which strengthen the case for its generality to be used for providing user-adaptive support in ELEs. However, more ELEs need to be tested and explored to find the potential improvement points in the UMA framework.

The three-phase design of UMA framework makes it easy to make adaptations to the methods (if necessary) to accommodate other ELEs while adhering to the principles of each phase. In behaviour discovery, the underlying clustering method can be changed as long as the principle used in the hybrid approach, that clustering should be guided and validated using a performance measure, is followed. In user classification phase, given the gradual nature of the data received, it is essential that the underlying classifier should be able to make decisions with little data (with reasonable accuracy) and be tolerant of small variations in feature values. This characteristic is more important than the final accuracy achieved at the end of the interaction. Also, it is imperative that in addition to the class label, the classifier should provide the patterns that led to each classification decision (to be used in the adaptive support phase). For the adaptive support phase, based on our experience, we suggest starting from simpler interventions that are linked to single widely observed patterns (same as we did here) to maximize the effectiveness of interventions.

As an example of adapting the UMA framework, when applying it to the CCK simulation, given the high number of actions available and the complexity of the interactions, it was intuitive that a more detailed feature representation was necessary. However, no change to the underlying data mining methods (e.g., the clustering algorithm, the association rule mining method, or the rule-based classifier), was necessary.

9.2.5 Capturing and integrating more information about users' interactions

Application of the UMA framework to the CCK simulation showed that for more complex interactions, capturing the context information for the actions is essential. There is room for further integration of information such as the state of ELE (e.g., whether the problem graph is

arc-consistent at the time of user's interaction with the CSP applet) and other context information to be added to the clustering and association rules mining processes.

9.2.6 Estimating the probability of following each relevant intervention based on a user's previous reactions

The current version of the framework ranks relevant intervention items based on their expected impact as calculated from the association rules. However, as more data is collected with the adaptive version of the ELEs, the aggregated follow records of users can be used for selecting the next intervention to be delivered. In other words, the goal would be to deliver the item that is most likely to be followed by each user. This can be implemented by collaborative filtering or a similar scheme to calculate the likelihood that the current user would follow an intervention based on his/her prior follow record and similarity of his/her record to the existing users and their respective follow records.

Bibliography

Aleven, V., Roll, I., McLaren, B. M., & Koedinger, K. R. (2016). Help Helps, But Only So Much: Research on Help Seeking with Intelligent Tutoring Systems. *International Journal of Artificial Intelligence in Education*, *26*(1), 205–223. https://doi.org/10.1007/s40593-015-0089-1

Amershi, S., Carenini, G., Conati, C., Mackworth, A. K., & Poole, D. (2008). Pedagogy and usability in interactive algorithm visualizations: Designing and evaluating CIspace. *Interacting with Computers*, *20*(1), 64–96. https://doi.org/10.1016/j.intcom.2007.08.003

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

Anaya, A. R., & Boticario, J. G. (2011). Content-free collaborative learning modeling using data mining. *User Modeling and User-Adapted Interaction*, *21*(1–2), 181–216. https://doi.org/10.1007/s11257-010-9095-z

Baker, R. (2010). Mining Data for Student Models. In R. Nkambou, J. Bourdeau, & R. Mizoguchi (Eds.), *Advances in Intelligent Tutoring Systems* (Vol. 308, pp. 323–337). Springer Berlin / Heidelberg. Retrieved from http://www.springerlink.com/content/03365843w5w51514/abstract/

Baker, R., Corbett, A., Koedinger, K., & Roll, I. (2005). Detecting When Students Game the System, Across Tutor Subjects and Classroom Cohorts. In L. Ardissono, P. Brna, & A. Mitrovic (Eds.), *User Modeling 2005* (Vol. 3538, pp. 220–224). Springer Berlin / Heidelberg. Retrieved from http://www.springerlink.com/content/74uptxk8n7jc2gj8/abstract/

Baker, R., Corbett, A. T., & Koedinger, K. (2004). Detecting student misuse of intelligent tutoring systems. In *Intelligent tutoring systems* (pp. 54–76).

Baker, R., Pardos, Z., Gowda, S., Nooraei, B., & Heffernan, N. (2011). Ensembling predictions of student knowledge within intelligent tutoring systems. In *User Modeling, Adaption and Personalization* (pp. 13–24).

Baker, R., & Yacef, K. (2009). The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining*, *1*(1), 3–17.

Barnes, T., & Stamper, J. (2008). Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data. In *Proceedings of the 9th international conference on Intelligent Tutoring Systems* (pp. 373–382). Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-540-69132-7_41

Barnes, T., Stamper, J., Lehman, L., & Croy, M. (2008). A pilot study on logic proof tutoring using hints generated from historical student data. In *Educational Data Mining 2008: Proceedings of 1st International Conference on Educational Data Mining*.

Beck, J. E., Stern, M., & Haugsjaa, E. (1996). Applications of AI in education. *Crossroads*, *3*(1), 11–15. https://doi.org/10.1145/332148.332153

Beck, J. E., & Woolf, B. P. (2000). High-level student modeling with machine learning. In *Intelligent tutoring systems* (pp. 584–593).

Ben-David, A. (2008). About the relationship between ROC curves and Cohen's kappa. *Eng. Appl. Artif. Intell.*, *21*(6), 874–882. https://doi.org/10.1016/j.engappai.2007.09.009

Ben-Naim, D., Bain, M., & Marcus, N. (2009). A user-driven and data-driven approach for supporting teachers in reflection and adaptation of adaptive tutorials. In *Proceedings of*

the Second International Conference on Educational Data Mining-EDM09, Córdoba, Spain, Universidad de Córdoba,(21-30).

Bernardini, A., & Conati, C. (2010). Discovering and Recognizing Student Interaction Patterns in Exploratory Learning Environments. In V. Aleven, J. Kay, & J. Mostow (Eds.), *Intelligent Tutoring Systems* (Vol. 6094, pp. 125–134). Springer. Retrieved from http://www.springerlink.com/content/e022251q7h1741t3/

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning* (1st ed. 2006. Corr. 2nd printing). Springer.

Bloom, B. S. (1984). The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. *Educational Researcher*, *13*(6), 4–16. https://doi.org/10.3102/0013189X013006004

Borek, A., McLaren, B. M., Karabinos, M., & Yaron, D. (2009). How Much Assistance Is Helpful to Students in Discovery Learning? In U. Cress, V. Dimitrova, & M. Specht (Eds.), *Learning in the Synergy of Multiple Disciplines* (pp. 391–404). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-04636-0_38

Canham, M., & Hegarty, M. (2010). Effects of knowledge and display design on comprehension of complex graphics. *Learning and Instruction*, *20*(2), 155–166. https://doi.org/10.1016/j.learninstruc.2009.02.014

Changchien, S., & Lu, T. C. (2001). Mining association rules procedure to support on-line recommendation by customers and products fragmentation. *Expert Systems with Applications*, *20*(4), 325–335.

Cocea, M., Gutierrez-Santos, S., & Magoulas, G. D. (2008). Challenges for intelligent support in exploratory learning: the case of ShapeBuilder. In *Proceedings of the International Workshop on Intelligent Support for Exploratory Environments at ECTEL 2008*. Maastricht, The Netherlands.

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum.

Cohen, J., West, S. G., Aiken, L. S., & Cohen, P. (2002). *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences* (3rd Revised edition). Mahwah, N.J: Lawrence Erlbaum.

Conati, C., Aleven, V., & Mitrovic, A. (2013). –Eye-Tracking for Student Modelling in Intelligent Tutoring Systems. In *Design recommendations for intelligent tutoring systems* (pp. 227–236).

Conati, C., Carenini, G., Harati, M., Tocker, D., Fitzgerald, N., & Flagg, A. (2011). User-Adaptive Visualizations: Can Gaze Data Tell Us When a User Needs Them? In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Conati, C., & Merten, C. (2007). Eye-tracking for user modeling in exploratory learning environments: An empirical evaluation. *Knowledge-Based Systems*, *20*(6), 557–574. https://doi.org/10.1016/j.knosys.2007.04.010

Courtemanche, F., Aïmeur, E., Dufresne, A., Najjar, M., & Mpondo, F. (2011). Activity recognition using eye-gaze movements and traditional interactions. *Interacting with Computers*, *23*(3), 202–213. https://doi.org/10.1016/j.intcom.2011.02.008

de Jong, T. (2006). Technological Advances in Inquiry Learning. *Science*, *312*(5773), 532–533. https://doi.org/10.1126/science.1127750

Desmarais, M. C., & Baker, R. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, *22*(1–2), 9–38.

Dix, A. (2009). *Human-computer interaction*. Springer. Retrieved from http://link.springer.com/10.1007/978-0-387-39940-9_192

Eivazi, S., & Bednarik, R. (2011). Predicting Problem-Solving Behavior and Expertise Levels from Visual Attention Data. In *The 2nd workshop on the Eye Gaze in Intelligent Human Machine Interaction* (pp. 9–16). Palo Alto, California, USA.

Fournier-Viger, P., Nkambou, R., & Mephu Nguifo, E. (2009). Exploiting Partial Problem Spaces Learned from Users' Interactions to Provide Key Tutoring Services in Procedural and Ill-Defined Domains. In *Proceedings of the 2009 conference on Artificial Intelligence in Education* (pp. 383–390). Amsterdam, The Netherlands: IOS Press. Retrieved from http://dl.acm.org/citation.cfm?id=1659450.1659509

Fratamico, L., Conati, C., Roll, I., & Kardan, S. (2017). Applying a framework for student modeling in interactive simulations: comparing data representation granularity to handle environment complexity. *International Journal of Artificial Intelligence in Education (IJAIED)*.

Freyberger, J., Heffernan, N., & 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* (pp. 1–4).

Frezzo, D., Behrens, J., & Mislevy, R. (2010). Design Patterns for Learning and Assessment: Facilitating the Introduction of a Complex Simulation-Based Learning Environment into

a Community of Instructors. *Journal of Science Education and Technology*, *19*(2), 105–114. https://doi.org/10.1007/s10956-009-9192-0

Frias-martinez, E., Chen, S. Y., & Liu, X. (2006). Survey of Data Mining Approaches to User Modeling for Adaptive Hypermedia. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions On*, *36*(6), 734–749. https://doi.org/10.1109/TSMCC.2006.879391

García, E., Romero, C., Ventura, S., & Castro, C. (2009). An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *User Modeling and User-Adapted Interaction*, *19*(1), 99–132.

Geng, L., & Hamilton, H. J. (2006). Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, *38*. https://doi.org/10.1145/1132960.1132963

Gluck, J., Bunt, A., & McGrenere, J. (2007). Matching attentional draw with utility in interruption. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07* (p. 41). San Jose, California, USA. https://doi.org/10.1145/1240624.1240631

Gobert, J. D., Montalvo, O., Toto, E., Sao Pedro, M., & Baker, R. (2010). The Science Assistments Project: Scaffolding Scientific Inquiry Skills. In *Proceedings of the 10th International Conference on Intelligent Tutoring Systems - Volume Part II* (pp. 445–445). Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-642-13437-1_102

Goldberg, J. H., & Helfman, J. I. (2010). Comparing Information Graphics: A Critical Look at Eye Tracking. In *Proceedings of the 3rd BELIV'10 Workshop: BEyond time and errors: novel evaLuation methods for Information Visualization* (pp. 71–78). Atlanta, GA, USA: ACM. https://doi.org/10.1145/2110192.2110203

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, *11*, 10–18. https://doi.org/10.1145/1656274.1656278

Hamza-Lup, F. G., Goeser, P. T., Johnson, W., Thompson, T., Railean, E., Popovici, D. M., & Hamza-Lup, G. (2009). Interactive 3D Web-Based Environments for Online Learning: Case Studies, Technologies and Challenges. In *Mobile, Hybrid, and On-line Learning, 2009. ELML'09. International Conference on* (pp. 13–18).

Han, Y. (2003). CPAR: Classification based on predictive association rules. In *Proceedings of SIAM international conference on data mining*.

Hegarty, M., Mayer, R. E., & Monk, C. A. (1995). Comprehension of Arithmetic Word Problems: A Comparison of Successful and Unsuccessful Problem Solvers. *Journal of Educational Psychology*, *87*(1), 18–32.

Hunt, E., & Madhyastha, T. (2005). Data mining patterns of thought. In *AAAI Workshop on Educational Data Mining* (pp. 31–39).

Hussain, T. S., Roberts, B., Menaker, E. S., Coleman, S. L., Pounds, K., Bowers, C., … others. (2009). Designing and developing effective training games for the US Navy. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)* (Vol. 2009). NTSA. Retrieved from http://ntsa.metapress.com/index/95028595XL4525H3.pdf

Iqbal, S. T., Adamczyk, P. D., Zheng, X. S., & Bailey, B. P. (2005). Towards an index of opportunity: understanding changes in mental workload during task execution. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 311–320). New York, NY, USA: ACM. https://doi.org/10.1145/1054972.1055016

Jackson, G. T., Olney, A., Graesser, A. C., & Kim, H. J. (2006). AutoTutor 3-D Simulations: Analyzing user's actions and learning trends. In *Proceedings of the 28th annual meeting of the Cognitive Science Society* (pp. 1557–1562). Retrieved from http://csjarchive.cogsci.rpi.edu/Proceedings/2006/docs/p1557.pdf

Jameson, A. (2009). Adaptive interfaces and agents. In A. Sears & J. A. Jacko (Eds.), *Human-Computer Interaction: Design Issues, Solutions, and Applications* (pp. 105–130). CRC Press.

Jarodzka, H., Scheiter, K., Gerjets, P., & van Gog, T. (2010). In the eyes of the beholder: How experts and novices interpret dynamic stimuli. *Learning and Instruction*, *20*(2), 146–154. https://doi.org/10.1016/j.learninstruc.2009.02.019

Kardan, S., & Conati, C. (2011). A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. In M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero, & J. Stamper (Eds.), *Proceedings of the 4th International Conference on Educational Data Mining* (pp. 159–168). Eindhoven, the Netherlands.

Kardan, S., & Conati, C. (2013). Evaluation of a Data Mining Approach to Providing Adaptive Support in an Open-Ended Learning Environment: A Pilot Study. In *AIED 2013 Workshops Proceedings Volume 2 Scaffolding in Open-Ended Learning Environments (OELEs)* (pp. 41–48). Retrieved from http://ceur-ws.org/Vol-1009/aied2013ws_volume2.pdf#page=47

Kardan, S., & Conati, C. (2015). Providing Adaptive Support in an Interactive Simulation for Learning: An Experimental Evaluation. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (pp. 3671–3680). New York, NY, USA: ACM. https://doi.org/10.1145/2702123.2702424

Kardan, S., Roll, I., & Conati, C. (2014). The Usefulness of Log Based Clustering in a Complex Simulation Environment. In S. Trausan-Matu, K. E. Boyer, M. Crosby, & K. Panourgia (Eds.), *Intelligent Tutoring Systems* (pp. 168–177). Springer International Publishing. Retrieved from http://link.springer.com/chapter/10.1007/978-3-319-07221-0_21

Kiesmueller, U., Sossalla, S., Brinda, T., & Riedhammer, K. (2010). Online identification of learner problem solving strategies using pattern recognition methods. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education* (pp. 274–278). New York, NY, USA: ACM. https://doi.org/10.1145/1822090.1822167

Kim, K., & Ahn, H. (2008). A recommender system using GA K-means clustering in an online shopping market. *Expert Systems with Applications*, *34*(2), 1200–1209. https://doi.org/10.1016/j.eswa.2006.12.025

Knoepfle, D. T., Wang, J. T., & Camerer, C. F. (2009). Studying Learning in Games Using Eye-tracking. *Journal of the European Economic Association*, *7*(2-3), 388–398. https://doi.org/10.1162/JEEA.2009.7.2-3.388

Köck, M., & Paramythis, A. (2011). Activity sequence modelling and dynamic clustering for personalized e-learning. *User Modeling and User-Adapted Interaction*, *21*(1), 51–97. https://doi.org/10.1007/s11257-010-9087-z

Kotsiantis, S. (2012). Use of machine learning techniques for educational proposes: a decision support system for forecasting students' grades. *Artificial Intelligence Review*, *37*(4), 331–344. https://doi.org/10.1007/s10462-011-9234-x

Lenth, R. V. (2006). *Java Applets for Power and Sample Size*. Retrieved from http://www.stat.uiowa.edu/~rlenth/Power

Loboda, T. D., & Brusilovsky, P. (2010). User-adaptive explanatory program visualization: evaluation and insights from eye movements. *User Modeling and User-Adapted Interaction*, *20*(3), 191–226. https://doi.org/http://dx.doi.org/10.1007/s11257-010-9077-1

Loboda, T. D., Brusilovsky, P., & Brunstein, J. (2011). Inferring word relevance from eye-movements of readers. In *Proceedings of the 16th international conference on Intelligent UserIinterfaces* (pp. 175–184). New York, NY, USA: ACM. https://doi.org/10.1145/1943403.1943431

Maloy, R. W., Verock-O'Loughlin, R.-E. A., Edwards, S. A., & Woolf, B. P. (2013). *Transforming Learning with New Technologies* (2 edition). Boston: Pearson.

Mavrikis, M. (2010). Modelling Student Interactions in Intelligent Learning Environments: Constructing Bayesian Networks from Data. *International Journal on Artificial Intelligence Tools*, *19*(06), 733. https://doi.org/10.1142/S0218213010000406

Mavrikis, M., & Gutierrez-Santos, S. (2010). Not all wizards are from Oz: Iterative design of intelligent learning environments by communication capacity tapering. *Computers & Education*, *54*(3), 641–651. https://doi.org/10.1016/j.compedu.2009.08.033

Mavrikis, M., Gutierrez-Santos, S., Geraniou, E., & Noss, R. (2012). Design requirements, student perception indicators and validation metrics for intelligent exploratory learning environments. *Personal and Ubiquitous Computing*, 1–16. https://doi.org/10.1007/s00779-012-0524-3

Merceron, A., & Yacef, K. (2003). A web-based tutoring tool with mining facilities to Improve Teaching and Learning. In H. U. Hoppe, F. Verdejo, & J. Kay (Eds.), *Artificial Intelligence in Education* (pp. 201–208). Sydney, Australia: IOS Press.

Milligan, G. W., & Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, *50*(2), 159–179.

Muldner, K., Christopherson, R., Atkinson, R., & Burleson, W. (2009). Investigating the Utility of Eye-Tracking Information on Affect and Reasoning for User Modeling. In *Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization: formerly UM and AH* (pp. 138–149). Trento, Italy: Springer-Verlag. https://doi.org/http://dx.doi.org/10.1007/978-3-642-02247-0_15

Murray, T., & Woolf, B. P. (1992). Results of encoding knowledge with tutor construction tools. In *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 17–17). AAAI Press.

Njoo, M., & De Jong, T. (1993). Exploratory learning with a computer simulation for control theory: Learning processes and instructional support. *Journal of Research in Science Teaching*, *30*(8), 821–844. https://doi.org/10.1002/tea.3660300803

O'Brien, R. M. (2007). A Caution Regarding Rules of Thumb for Variance Inflation Factors. *Quality & Quantity*, *41*(5), 673–690. https://doi.org/10.1007/s11135-006-9018-6

Peña-Ayala, A. (2014). Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications*, *41*(4), 1432–1462.

Perera, D., Kay, J., Koprinska, I., Yacef, K., & Zaïane, O. R. (2009). Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Trans. on Knowl. and Data Eng.*, *21*(6), 759–772. https://doi.org/10.1109/TKDE.2008.138

Perera, D., Kay, J., Koprinska, I., Yacef, K., & Zaiane, O. R. (2009). Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(6), 759–772. https://doi.org/10.1109/TKDE.2008.138

Perkins, K., Adams, W., Dubson, M., Finkelstein, N., Reid, S., Wieman, C., & LeMaster, R. (2006). PhET: Interactive Simulations for Teaching and Learning Physics. *The Physics Teacher*, *44*(1), 18–23. https://doi.org/10.1119/1.2150754

Ploetzner, R., Lippitsch, S., Galmbacher, M., Heuer, D., & Scherrer, S. (2009). Students' difficulties in learning from dynamic visualisations and how they may be overcome. *Computers in Human Behavior*, *25*(1), 56–65. https://doi.org/10.1016/j.chb.2008.06.006

Poole, D. L., & Mackworth, A. K. (2010). *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press.

Rayner, K. (1995). Eye movements and cognitive processes in reading, visual search, and scene perception. In *Eye Movement Research Mechanisms, Processes, and Applications* (Vol. Volume 6, pp. 3–22). North-Holland. Retrieved from http://www.sciencedirect.com/science/article/pii/S0926907X05800030

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, *124*(3), 372–422. https://doi.org/10.1037/0033-2909.124.3.372

Romero, C., & Ventura, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, *33*(1), 135–146. https://doi.org/10.1016/j.eswa.2006.04.005

Romero, C., & 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. https://doi.org/10.1109/TSMCC.2010.2053532

Romero, C., Ventura, S., & García, E. (2008). Data mining in course management systems: Moodle case study and tutorial. *Computers & Education*, *51*(1), 368–384. https://doi.org/10.1016/j.compedu.2007.05.016

Romero, C., Ventura, S., Zafra, A., & Bra, P. (2009). Applying Web usage mining for personalizing hyperlinks in Web-based adaptive educational systems. *Computers & Education*, *53*(3), 828–840.

Rong-Fuh, D. (2010). Examining the validity of the Needleman–Wunsch algorithm in identifying decision strategy with eye-movement data. *Decision Support Systems*, *49*(4), 396–403. https://doi.org/10.1016/j.dss.2010.05.001

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

Salajan, F. D., Perschbacher, S., Cash, M., Talwar, R., El-Badrawy, W., & Mount, G. J. (2009). Learning with web-based interactive objects: An investigation into student perceptions of effectiveness. *Computers & Education*, *53*(3), 632–643. https://doi.org/10.1016/j.compedu.2009.04.006

Santos, O. C., & Boticario, J. G. (2015). Practical Guidelines for Designing and Evaluating Educationally Oriented Recommendations. *Comput. Educ.*, *81*(C), 354–374. https://doi.org/10.1016/j.compedu.2014.10.008

Shanabrook, D. H., Cooper, D. G., Woolf, B. P., & Arroyo, I. (2010). Identifying High-Level Student Behavior Using Sequence-based Motif Discovery. In *Proceedings of 3rd International Conference on Educational Data Mining* (pp. 191–200). Pittsburgh, PA, USA.

Shih, B., Koedinger, K. R., & Scheines, R. (2010). Unsupervised Discovery of Student Strategies. In *Proceedings of the 3rd International Conference on Educational Data Mining* (pp. 201–210).

Shute, V. J. (1993). A comparison of learning environments: All that glitters. In *Computers as cognitive tools.* (pp. 47–73). Hillsdale, NJ, England: Lawrence Erlbaum Associates, Inc.

Simola, J., Salojärvi, J., & Kojo, I. (2008). Using hidden Markov model to uncover processing states from eye movements in information search tasks. *Cognitive Systems Research*, *9*(4), 237–251.

Stamper, J., Eagle, M., Barnes, T., & Croy, M. (2011). Experimental Evaluation of Automatic Hint Generation for a Logic Tutor. In G. Biswas, S. Bull, J. Kay, & A. Mitrovic (Eds.), *Artificial Intelligence in Education* (Vol. 6738, pp. 345–352). Springer Berlin / Heidelberg. Retrieved from http://www.springerlink.com/content/f0w1t2642k4t6128/abstract/

Stamper, J., Eagle, M., Barnes, T., & Croy, M. (2013). Experimental Evaluation of Automatic Hint Generation for a Logic Tutor. *International Journal of Artificial Intelligence in Education*, *22*(1), 3–17. https://doi.org/10.3233/JAI-130029

Steichen, B., Wu, M. M. A., Toker, D., Conati, C., & Carenini, G. (2014). Te,Te,Hi,Hi: Eye Gaze Sequence Analysis for Informing User-Adaptive Information Visualizations. In V. Dimitrova, T. Kuflik, D. Chin, F. Ricci, P. Dolog, & G.-J. Houben (Eds.), *User Modeling, Adaptation, and Personalization* (pp. 183–194). Springer International Publishing. https://doi.org/10.1007/978-3-319-08786-3_16

Sweller, J. (1999). *Instructional design in technical areas*. Australian Council for Educational Research.

Tang, C., Yin, H., Li, T., Lau, R. W. H., Li, Q., & Kilis, D. (2000). Personalized Courseware Construction Based on Web Data Mining. In *Proceedings of the First International Conference on Web Information Systems Engineering (WISE'00)-Volume 2 - Volume 2*

(p. 2204–). Washington, DC, USA: IEEE Computer Society. Retrieved from http://dl.acm.org/citation.cfm?id=882512.885431

Tang, T., & McCalla, G. (2005). Smart Recommendation for an Evolving E-Learning System: Architecture and Experiment. *International Journal on E-Learning*, *4*(1), 105–129.

Tempelaar, D. T., Rienties, B., & Giesbers, B. (2015). In search for the most informative data for feedback generation: Learning analytics in a data-rich context. *Computers in Human Behavior*, *47*, 157–167. https://doi.org/10.1016/j.chb.2014.05.038

Thabtah, F. (2007). A Review of Associative Classification Mining. *The Knowledge Engineering Review*, *22*(01), 37–65. https://doi.org/10.1017/S0269888907001026

Ting, C. Y., Zadeh, M., & Chong, Y. K. (2006). A decision-theoretic approach to scientific inquiry exploratory learning environment. In *Intelligent Tutoring Systems* (pp. 85–94).

Trivedi, S., Pardos, Z. A., & Heffernan, N. T. (2011). Clustering students to generate an ensemble to improve standard test score predictions. In *Proceedings of the 15th international conference on Artificial Intelligence in Education* (pp. 377–384). Berlin, Heidelberg: Springer-Verlag. Retrieved from http://dl.acm.org/citation.cfm?id=2026506.2026557

Tsai, M.-J., Hou, H.-T., Lai, M.-L., Liu, W.-Y., & Yang, F.-Y. (2012). Visual attention for solving multiple-choice science problem: An eye-tracking analysis. *Computers & Education*, *58*(1), 375–385. https://doi.org/10.1016/j.compedu.2011.07.012

Van Joolingen, W. R., De Jong, T., & Dimitrakopoulou, A. (2007). Issues in computer supported inquiry learning in science. *Journal of Computer Assisted Learning*, *23*(2), 111–119. https://doi.org/10.1111/j.1365-2729.2006.00216.x

VanLehn, K. (1988). Student modeling. In M. Polson & J. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 55–78). Hillsdale, NJ: Erlbaum.

VanLehn, K. (2006). The Behavior of Tutoring Systems. *Int. J. Artif. Intell. Ed.*, *16*(3), 227–265.

VanLehn, K. (2011). The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist*, *46*(4), 197–221. https://doi.org/10.1080/00461520.2011.611369

Walker, E., Rummel, N., & Koedinger, K. R. (2014). Adaptive Intelligent Support to Improve Peer Tutoring in Algebra. *International Journal of Artificial Intelligence in Education*, *24*(1), 33–61. https://doi.org/10.1007/s40593-013-0001-9

Westerfield, G., Mitrovic, A., & Billinghurst, M. (2013). Intelligent Augmented Reality Training for Assembly Tasks. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), *Artificial Intelligence in Education* (pp. 542–551). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-39112-5_55

Zaïane, O. R. (2002). Building a recommender agent for e-learning systems. In *Computers in Education, 2002. Proceedings. International Conference on* (pp. 55–59 vol.1). https://doi.org/10.1109/CIE.2002.1185862

Zaïane, O. R., Xin, M., & Han, J. (1998). Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. In *Proceedings of the Advances in Digital Libraries Conference* (p. 19–). Washington, DC, USA: IEEE Computer Society. Retrieved from http://dl.acm.org/citation.cfm?id=582987.785951

Zhang, Y., & Jiao, J. (Roger). (2007). An associative classification-based recommendation system for personalization in B2C e-commerce applications. *Expert Systems with Applications*, *33*(2), 357–367. https://doi.org/10.1016/j.eswa.2006.05.005

Appendices

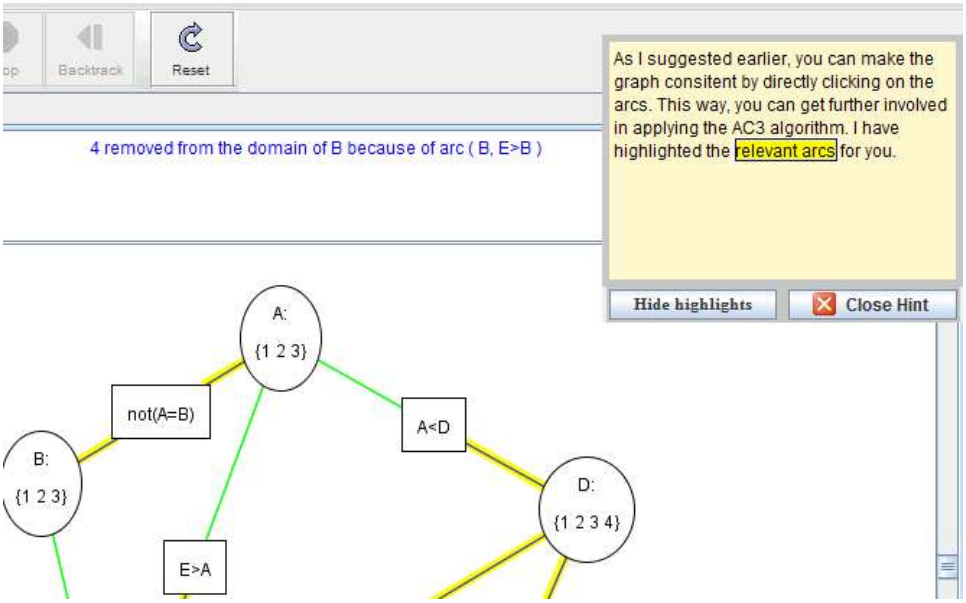
Appendix A Material Used in the Experiments

A.1 Post-study Questionnaires

General questionnaire

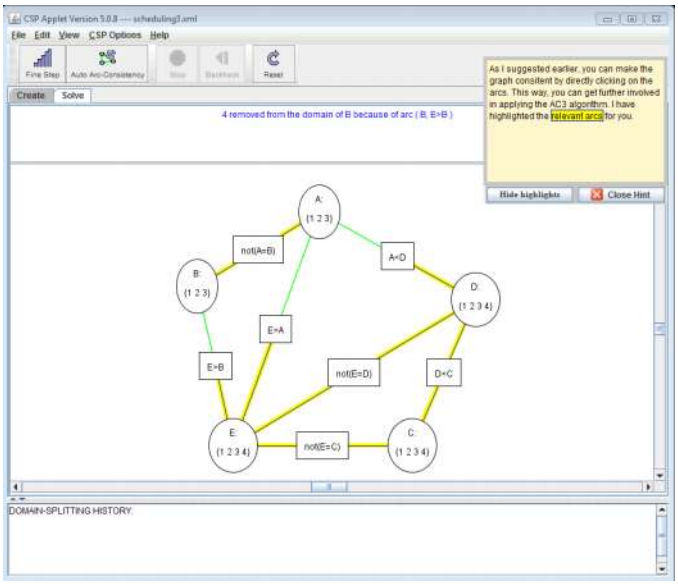
Please rate the following statements.						
		Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
19	I felt more **confident** answering the **post-test** questions compared to **pre-test** questions					
20	I found the applet **helpful** in **understanding** the AC-3 algorithm					
21	I think I did better in the **post-test** compared to **pre-test**					

Intervention Questionnaire



Please rate the following statements about the hint box that appeared at the right corner of the screen (shown above).

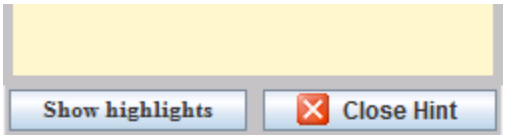
		Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
	In general, the **hints** were appropriate given my behaviour.					
	The **messages** displayed in the hint box were **easy to understand**.					
	The **messages** displayed in the hint box were **useful**.					
	I usually **followed** the advice displayed in the **hint box**.					
	I found the hint box **intrusive**.					
	I found the hint box **annoying**.					
	I easily **noticed** the new **messages** displayed in the Hint box.					

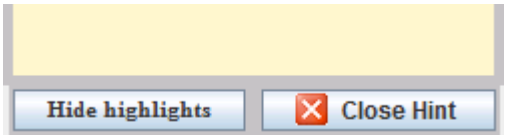


As shown above, with some hints, you saw that the corresponding elements of the applet (arcs, nodes, or buttons) were **highlighted** with **yellow color**. These changes are designed to draw your attention.

Please rate the following statements regarding the **highlighting** effect

		Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree	N/A
	In general, the **highlighting** of the interface items was **appropriate** given my behaviour.						
	Highlighting the interface items was **useful** for me.						
0	I found the **highlighting** of the interface items **intrusive**.						
1	I found the **highlighting** of the interface items **annoying**.						
2	I easily **noticed** the **highlighting** of the interface items.						
3	**Highlighting** the interface items helped me notice that a **new message** has appeared in the hint box.						
4	**Highlighting** the interface items helped me find the right **elements** of the applet to work with.						
5	I could easily see the connection between the **highlighted** elements and the text shown in the hint box						





There were two ways to hide the highlights in the applet
(1) Manually by clicking on the '**Hide highlights**' button (shown above).
(2) Let the applet determine when you have noticed the highlight and remove it automatically.
In both cases, you could use the '**Show highlights**' to see the highlights again (shown above).

Please rate the following statements for these methods

		Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree	Did not use
6	I found the '**hide highlights**' button **helpful**						
7	I found the mechanism that **automatically** hid the *highlights* **helpful**						
8	I found the '**show highlights**' button **helpful**						

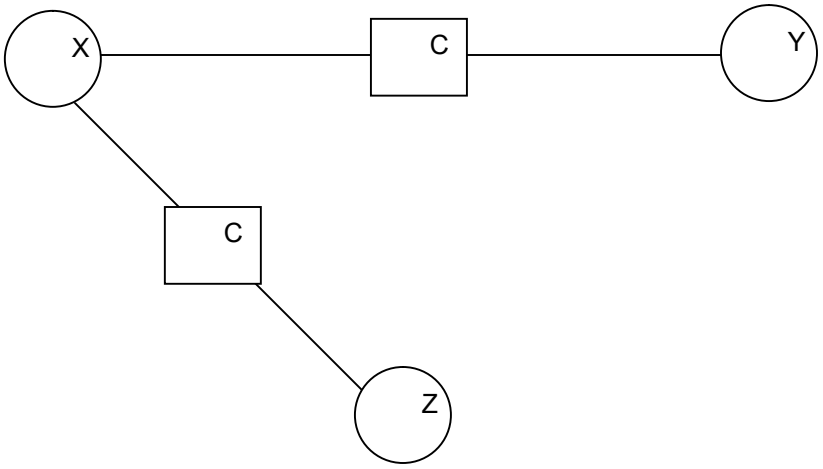
A.2 Pre-test and Post-test

CSP Pre-Test (15 Minutes)

*If you could not finish the question, please state why (*i.e.,* not enough time, didn't understand question, etc)*

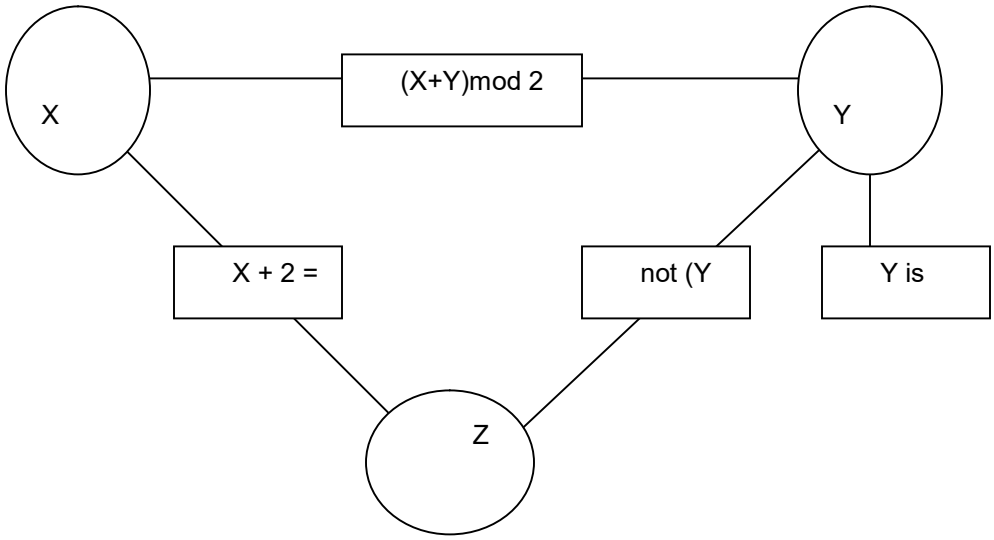
* If you need extra room write on the back of the test*

1. Consider the constraint graph shown below. Imagine that arc consistency has just reduced the domain of X as a result of considering the edge <X, C1>.



Do we have to add the edge <Y, C1> **back** to the list of "to-do arcs"? **Why** or **why not**?

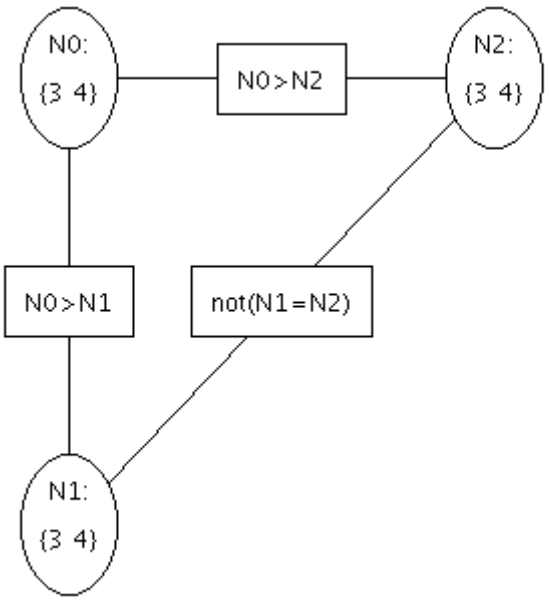
2. Consider the following constraint network: Note that *(X+Y) mod 2=1* means that *X+Y* is odd.



a) Is this constraint network arc consistent?

b) If it is, explain why the constraint network is arc consistent. If it isn't, explain **why** the constraint network is not arc consistent **AND** state which arcs are not arc consistent.

3. Consider the following constraint network.



a) Is this constraint network arc consistent?

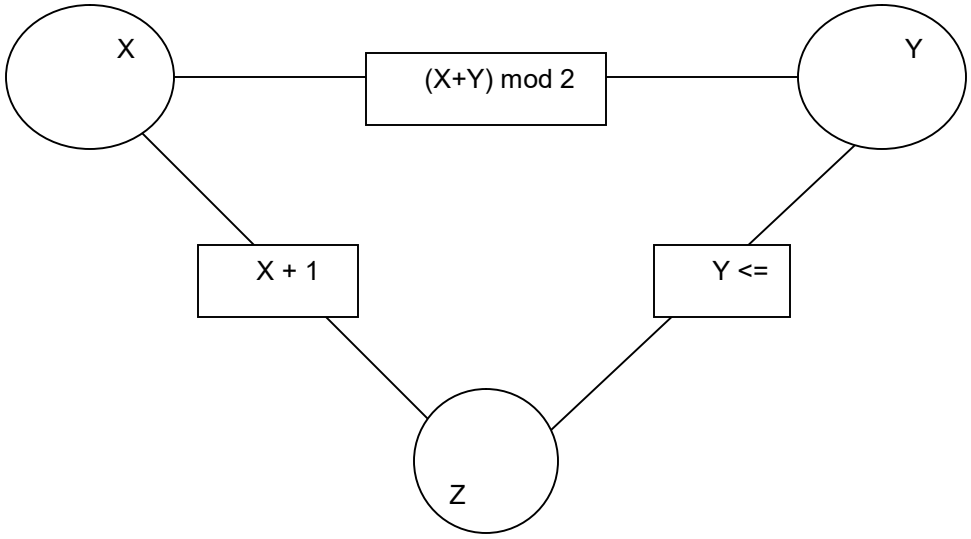
b) If it is, explain **why** the constraint network is arc consistent. If it isn't, make the network arc consistent and give **all** solutions.

4. Consider a scheduling problem with four activities labeled {A, B, C, D}. The domain of each activity is {2, 3, 4}. Suppose that the constraints on scheduling are as follows: $A \neq B$, $C<A$, $A< D$, $B=D$ and $C<D$.

a) Draw the initial constraint network, including domains of all nodes and arcs representing all binary relations.

b) Make the network arc consistent and give **all** solutions.

5. Consider the following constraint network. Note that *(X+Y) mod 2=1* means that *X+Y* is odd.



a) Make the constraint network arc consistent. For each domain element removed from a node, identify the constraint responsible for the removal, i.e., you have to write a sequence of steps of the form

"*val_1,…,val_K* removed from *VarX* because of *constraintC*".

b) Why is domain splitting necessary for this CSP?

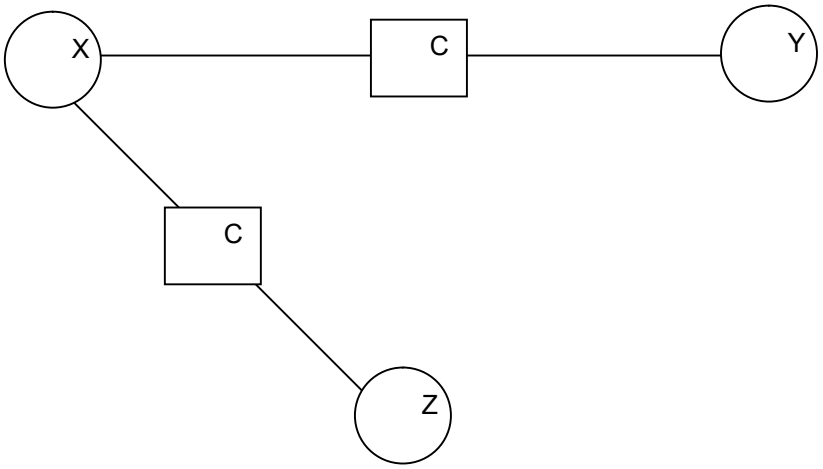
c) Show how domain splitting can solve this problem. Choose to split on X, and then find all solutions to the network. Write out the steps as you did for *part a* above, and include steps for domain splitting and backtracking where necessary.

CSP Post-Test (15 Minutes)

*If you could not finish the question, please state why (*i.e.,* **not enough time**, **didn't understand question**, etc)*

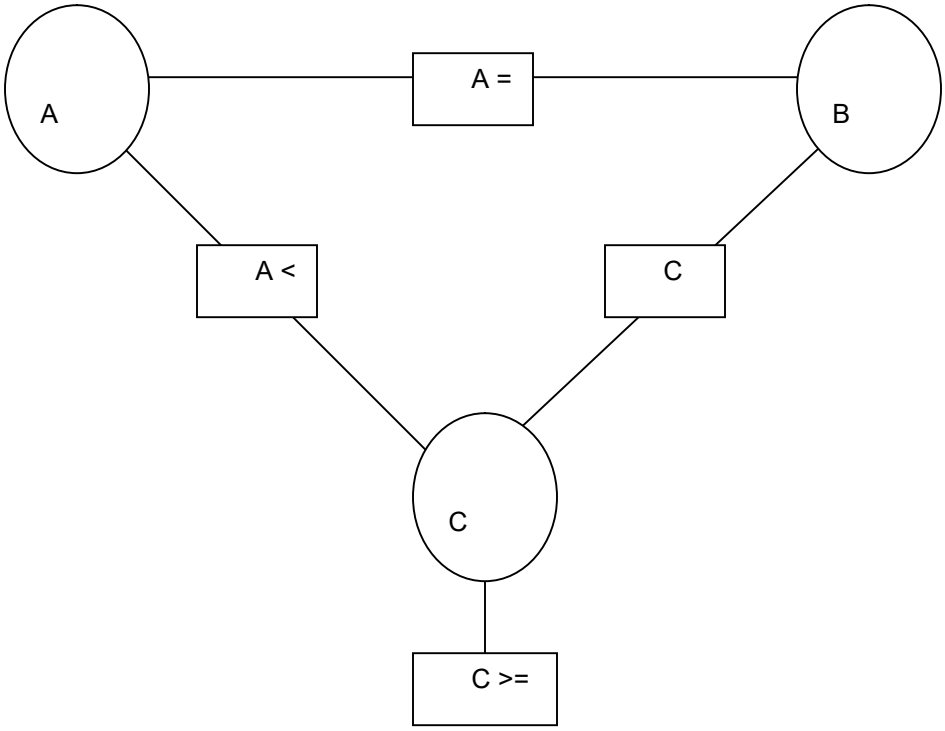
* If you need extra room write on the back of the test*

1. Consider the constraint graph shown below. Imagine that arc consistency has just reduced the domain of X as a result of considering the edge <X, C1>.



Do we have to add the edge <X, C2> **back** to the list of "to-do arcs"? **Why** or **why not**?

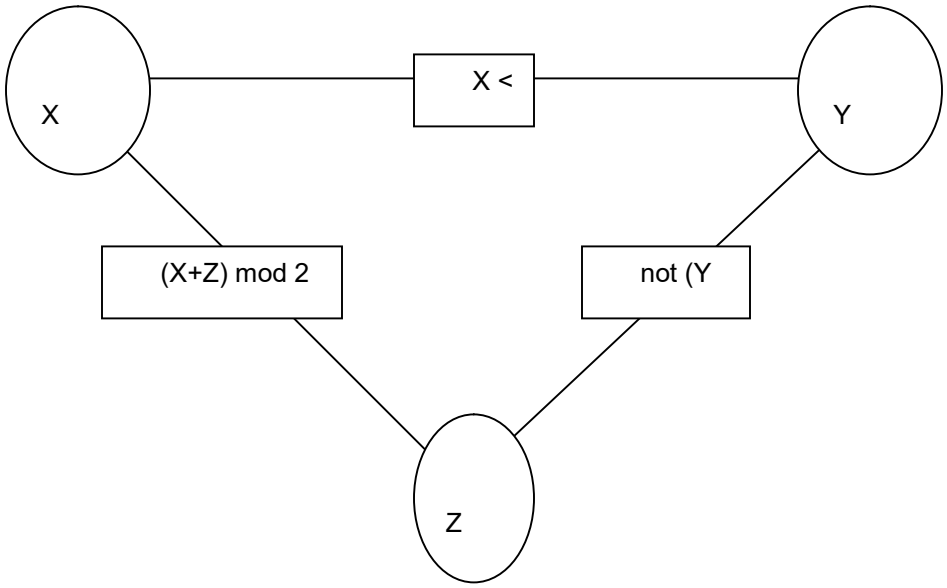
2. Consider the following constraint network.



a) Is this constraint network arc consistent?

b) If it is, explain **why** the constraint network is arc consistent. If it isn't, make the network arc consistent and give **all** solutions.

3. Consider the following constraint network: Note that *(X+Z) mod 2=1* means that *X+Z* is odd.



a) Is this constraint network arc consistent?

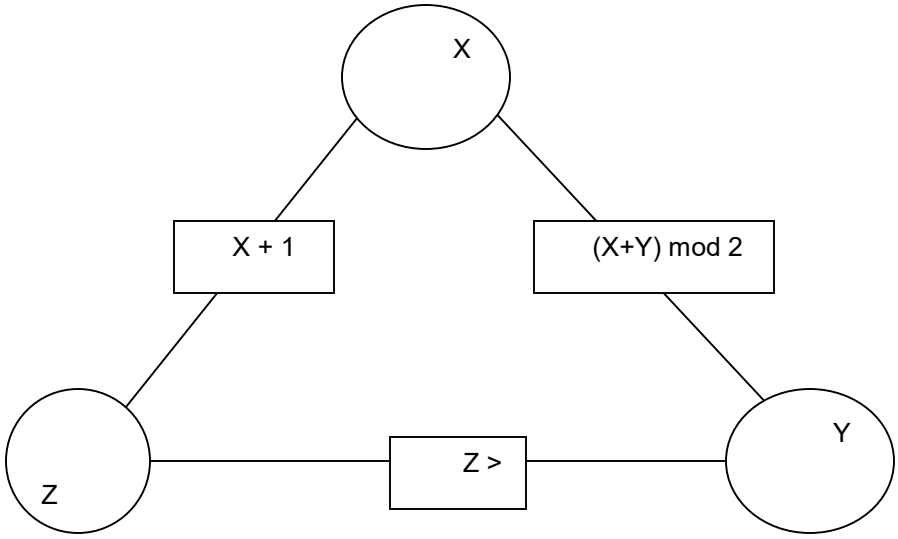
b) If it is, explain **why** the constraint network is arc consistent. If it isn't, explain **why** the constraint network is not arc consistent **AND** state which arcs are not arc consistent.

4. Consider the problem of scheduling four activities labeled {W, X, Y, Z}. The domain of each activity is {1, 2, 3}. Suppose that the constraints on scheduling are as follows: $W > X$, $W < Z$, $W = Y$, $Y = X$ and $Z \neq 2$.

a) Draw the initial constraint network, including domains of all nodes and arcs representing all binary relations.

b) Make the network arc consistent and give **all** solutions.

5. Consider the following constraint network. Note that *(X+Y) mod 2=1* means that *X+Y* is odd.



a) Make the constraint network arc consistent. For each domain element removed from a node, identify the constraint responsible for the removal, i.e., you have to write a sequence of steps of the form "*val1,…,valK* removed from *VarX* because of *constraintC*".

b) Why is domain splitting necessary for this CSP?

c) Show how domain splitting can solve this problem. Choose to split on X, and then find all solutions to the network. Write out the steps as you did for *part a* above, and include steps for domain splitting and backtracking where necessary.

A.3 Hint Messages

Intervention Code	Intervention Description	Level 1 message Level 2 message
DAC_fr	Using Direct Arc Click more often	"Did you know you can tell AC-3 which arc to make consistent by clicking on that arc?"
		As I suggested earlier, you can choose which arc to make consistent next by clicking on it. This way, you can get more involved in applying the AC-3 algorithm. I have highlighted the relevant arcs for you.
DAC_PA	Spending more time for reflection, after performing Direct Arc Clicks	Recently, you have performed a few direct arc clicks (clicking on the arcs) very fast. Please try to slow down.
		Once again, I see that you have performed a few direct arc clicks (clicking on the arcs) very fast. Please try to slow down, this may help you think more about the outcomes of each action.
Reset_fr	Using Reset less frequently	You have used the Reset button excessively. I recommend that you limit your usage of this action.
		You have reset the problem over and over again. Why don't you try using the other available actions instead of resetting the problem?
AAC_fr	Using Auto Arc-Consistency less frequently	You are often using "Auto Arc-Consistency" to make the graph arc-consistent. Please consider other options available in the applet.
		You are still using "Auto Arc-Consistency" to make the graph arc-consistent. As I suggested earlier, you can use other actions which can get you more involved in making the graph arc-consistent.
DS_fr	Using Domain Splitting less frequently (only when appropriate)	You have been using domain splitting very frequently. I suggest avoiding this action when it is not necessary according to the AC-3 algorithm.
		Once more, you are performing domain splitting very often. As before, I strongly recommend using this action only when it is necessary according to the AC-3 algorithm.

FS_PA	Spending more time after performing Fine Steps	You seem to be performing "Fine Step" actions rather quickly. Please try to slow down.
		Again, you seem to be performing "Fine Step" actions rather quickly. If you slow down, this may help you think more about the outcomes of each action.
BT_fr	Using Back Track less frequently (only when appropriate)	I see that you are using the "Back Track" button very often. I recommend using the Back Track button only when necessary according to the AC-3 algorithm.
		Again, you continue to use the "Back Track" button frequently. As I suggested earlier, you should only use the Back Track button when you are done applying the AC-3 algorithm on the CSP generated after the last domain splitting.
FS_fr	Using Fine Step less frequently	Using the "Fine Step" button has been your main choice for making the graph arc-consistent. Please consider other options available in the applet.
		You continue to rely on the "Fine Step" button to make the graph arc-consistent. As I suggested earlier, you can use other actions available for this purpose.
Reset_PA	Spending more time after performing after resetting for planning	After resetting the problem, you tend to perform your next action right away. I recommend that after resetting, you spend some time planning what to do next
		Once again, you are performing your next actions without any pause after resetting the problem. As I suggested earlier, after using the reset button, you should spend some time planning your strategy on how to apply the AC-3 algorithm effectively.