On Creating a Student Model to Assess Effective Exploratory Behaviour in an Open Learning Environment

by

Andrea Bunt

BSCH (Computing and Information Science)
Queen’s University, 1999

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in THE FACULTY OF GRADUATE STUDIES (Department of Computer Science)

We accept this thesis as conforming to the required standard

The University of British Columbia
August 2001
© Andrea Bunt, 2001
In presenting this thesis/essay in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying for this thesis for scholarly purposes may be granted by the Head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Computer Science
The University of British Columbia
2366 Main mall
Vancouver, BC
Canada V6T 1Z4

Aug. 23, 2001
Date
Abstract

Open learning environments give users a high degree of freedom and control to explore. This freedom and control is beneficial for some students, resulting in a deeper understanding of the material than they would gain through traditional means of instruction. For others, this type of environment is problematic, since for various reasons, these students are not able to explore effectively. One way to address this problem is to augment the environments with tailored support.

To provide feedback tailored to the student’s difficulties, the environment must have some way of monitoring and assessing her exploration. This thesis investigates the creation of a student model that assesses the effectiveness of the student’s exploratory behaviour. Monitoring user behaviour in an open learning environment is difficult since there is typically little information available to the model to make its assessment. The model can view with which items the student experiments, but does not have direct access to the effects of those experiments on her understanding of the domain. As a result, how to model effective exploratory behaviour has not been extensively researched.

The Student Model in this thesis has been implemented and evaluated in the context of the Adaptive Coach for Exploration (ACE). The model monitors the student’s exploration of ACE’s activities to generate an assessment of how effectively the learner is exploring. Using this assessment, ACE’s Coach provides tailored feedback to guide the student’s exploration process. To handle the large amount of uncertainty present in the modelling task, the Student Model is based on Bayesian Networks.

The features of ACE’s Student Model have been developed and refined using two evaluations of ACE with human subjects. The first evaluation was used to evaluate the effects of including tailored support in an open learning environment, and also provided insight into ways to improve the Student Model’s preliminary design. The second evaluation tested those improvements. Results of the evaluations found that both the frequency with which students accessed the tailored feedback and the number of activities that they explored effectively (as determined by the Student Model) were positively correlated with learning.
Contents

Abstract ii

Contents iii

List of Tables vii

List of Figures viii

Acknowledgements x

Dedication xi

1 Introduction 1
  1.1 Intelligent Tutoring Systems and Open Learning Environments 2
  1.2 Problems with Open Learning Environments 4
  1.3 Student Modelling 6
    1.3.1 Types of Assessment 7
    1.3.2 Bandwidth 7
  1.4 Thesis Approach: Modelling Effective Exploratory Behaviour 8
  1.5 Thesis Goals and Contributions 9
  1.6 Outline 11

2 Related Work 12
  2.1 Support in Open Learning Environments 12
## Contents

2.1.1 Cognitive Tools ............................................. 12
2.1.2 Tailored Support ............................................. 13

2.2 Student Modelling Using Bayesian Networks .................. 16
  2.2.1 Examples of Bayesian Student Models ..................... 16
  2.2.2 Drawbacks of Bayesian Networks ......................... 17

3 ACE ................................................................. 19
  3.1 The GUI ......................................................... 20
    3.1.1 The Machine Unit ........................................ 22
    3.1.2 The Arrow Unit ........................................... 23
    3.1.3 The Plot Unit ............................................. 24
    3.1.4 Tools ..................................................... 25
  3.2 The Student Model ............................................ 27
  3.3 The Coach ..................................................... 27
    3.3.1 Curriculum ............................................... 27
    3.3.2 Hints .................................................... 27
  3.4 The Knowledge Base .......................................... 29
    3.4.1 Function Attributes ...................................... 30
    3.4.2 Function I/O Concepts ................................... 30
    3.4.3 Relevant Exploration Cases ............................... 30

4 Student Model - Version I ........................................ 34
  4.1 Uncertainty in the Modelling Task ............................ 34
  4.2 Structure of the Student Model's Bayesian Networks ........... 36
    4.2.1 High Level Description of the Model's Bayesian Networks 36
  4.3 Network Design ............................................... 39
    4.3.1 Static vs. Dynamic ...................................... 39
    4.3.2 Machine/Arrow Units .................................... 39
    4.3.3 Plot Unit ................................................. 42
6.2.2 Study Participants ........................................... 68
6.2.3 Experiment Design .......................................... 68
6.2.4 Results ...................................................... 68

7 Conclusions and Future Work .................................. 71
7.1 Satisfaction of Thesis Goals ................................... 71
  7.1.1 Model for Exploration .................................... 71
  7.1.2 Evaluation of the Model and Intelligent Support ......... 72
7.2 Generalizability .................................................. 73
7.3 Limitations ..................................................... 74
7.4 Future Work ..................................................... 75
  7.4.1 Exploration of Relevant Exploration Cases .............. 75
  7.4.2 Extending the Model's Diagnostic Abilities ............. 76
  7.4.3 Formal Evaluation ......................................... 76
  7.4.4 Automated Testing ....................................... 77
7.5 Conclusion ...................................................... 77

Bibliography .......................................................... 78

Appendix A Programming Credits ................................ 84

Appendix B Curriculum File ........................................ 85

Appendix C Observer Sheet .......................................... 87

Appendix D Pre-Test ................................................ 88

Appendix E Post-Test ............................................... 97
List of Tables

3.1 Functions Types and their Attributes ........................................... 29
3.2 Function I/O Concepts ............................................................... 29
3.3 Symbols for the Exploration Cases .............................................. 30

4.1 CPT for an Example Function Node fn. ...................................... 47
4.2 CPT for the Ability to Correctly Generate the Output for a Simple Function. 47
4.3 CPT for Understanding the Slope Concept. ................................. 48
4.4 CPT for Understanding the Concepts in Arrow Unit Given Exploration and Direct Evidence ......................................................... 48
4.5 CPT for Understanding Positive Intercepts given Exploration and Direct Evidence ................................................................. 48

5.1 Experiment Schedule ................................................................. 52
5.2 Information Recorded in the Log Files ........................................ 53

6.1 An Example CPT for an Exploration Node in the Second Version of the Student Model ......................................................... 66
6.2 Unnecessary Navigation Hints .................................................... 69
6.3 Premature Passes ................................................................. 69
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>ACE's Modules</td>
<td>20</td>
</tr>
<tr>
<td>3.2</td>
<td>The ACE Interface</td>
<td>21</td>
</tr>
<tr>
<td>3.3</td>
<td>The Machine Unit</td>
<td>22</td>
</tr>
<tr>
<td>3.4</td>
<td>The Arrow Unit</td>
<td>23</td>
</tr>
<tr>
<td>3.5</td>
<td>The Plot Unit</td>
<td>24</td>
</tr>
<tr>
<td>3.6</td>
<td>The Tools Icons</td>
<td>25</td>
</tr>
<tr>
<td>3.7</td>
<td>The Lesson Browser</td>
<td>25</td>
</tr>
<tr>
<td>3.8</td>
<td>The Exploration Assistant</td>
<td>26</td>
</tr>
<tr>
<td>3.9</td>
<td>An Example Hint Window</td>
<td>28</td>
</tr>
<tr>
<td>3.10</td>
<td>A Navigation Hint</td>
<td>28</td>
</tr>
<tr>
<td>3.11</td>
<td>Rules for the Relevant Exploration Cases</td>
<td>31</td>
</tr>
<tr>
<td>3.12</td>
<td>The Knowledge Base's Representation of a Linear Function in the Arrow Unit</td>
<td>32</td>
</tr>
<tr>
<td>4.1</td>
<td>A High-Level Description of the Bayesian Networks</td>
<td>35</td>
</tr>
<tr>
<td>4.2</td>
<td>Two Different Possibilities for the Direction of the Arcs in a Network with</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>Dimensional Variables</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>The Static Portion of the Bayesian Network for the Machine and Arrow Units</td>
<td>40</td>
</tr>
<tr>
<td>4.4</td>
<td>Part of the Dynamic Portion of the Bayesian Network for the Arrow and</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>Machine Units that Contains Exploration Nodes</td>
<td></td>
</tr>
<tr>
<td>4.5</td>
<td>Part of the Dynamic Portion of the Bayesian Network for the Arrow and</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>Machine Units that Contains Knowledge Nodes</td>
<td></td>
</tr>
</tbody>
</table>
4.6 The Static Portion of the Bayesian Network for a Linear Function in the Plot Unit .......................................................... 44

4.7 An Example Dynamic Portion of the Bayesian Network for the Plot Unit .... 45

6.1 A High-Level Description of the Second Version of the Student Model ...... 62

6.2 A Simplified Example of the New Bayesian Network for the Machine/Arrow Units ................................................................. 63

B.1 A Sample Curriculum File ................................................................ 86
Acknowledgements

First, I would like to thank my supervisor, Cristina Conati, for all of her guidance. In addition, her financial generosity allowed me to attend a number of conferences, giving me the opportunity to share my work and ideas with others in the field. I also feel very fortunate to have as a surrogate supervisor, Gord McCalla, a professor at the University of Saskatchewan. Gord got me started in this field several years ago when he hired me to work in the AIRES lab for the summer. His continued active interest in my research career means a great deal to me. Collaborating with Kasia Muldner and Michael Huggett on the ACE environment was a great experience and I would not have been able to build something as substantial as ACE without them. Alan Mackworth graciously agreed to be my second reader, providing me with a number of very helpful comments. I also greatly appreciate the input I received from my proof-readers, Stephane Durocher and my dad, Rick Bunt. My dad provided me with extremely helpful pieces of advice on how to organize the thesis and highlight the contributions. Steph also deserves a big thanks being amazingly patient during my stress episodes and for being so encouraging and caring. Finally, I would like to thank my parents, Rick and Gail Bunt, who have both been incredibly loving and encouraging my whole life. Without their sound advice and calming words I might not stuck through a turbulent first year in grad school. They have been there for me always, without question or hesitation.

ANDREA BUNT

The University of British Columbia
August 2001
To my mom and dad
Chapter 1

Introduction

As software packages and the World Wide Web continue to grow, users are often faced with applications that are rich with features and information, but that provide little guidance on how to use the features or find the information that is relevant to them. Consider a user that is trying to find information on a topic of interest on the Internet. When the user opens a Web browser, there are no direct instructions on how to find this information. Rather the user must begin to experiment with various search queries and explore different web pages until she is able to find what she is looking for. This type of environment, which requires its users to explore and experiment without providing much explicit instruction, is known as an open environment.

The problem with open environments is that, for various reasons, users are not always able to explore effectively. A potential solution to this problem would be to augment the environments with a model of the user that could assess the effectiveness of the user's exploratory behaviour. This would allow the environments to sense when their users are not exploring effectively and adapt accordingly.

The thesis investigates issues involved in creating a user model that can assess a user's exploratory behaviour in open learning environments (open environments that are used for educational purposes). If properly designed, this model would allow the environment to sense when the student is experiencing difficulty with the exploration process.
and provide tailored assistance, thereby, improving the experience of learning through open environments for students with a wide range of learning styles.

1.1 Intelligent Tutoring Systems and Open Learning Environments

Intelligent Tutoring Systems (ITSs) are computer-based educational programs that present material in a flexible and personalized way. According to Shute [40], an ITS is a system that is able to:

"a) accurately diagnose a student's knowledge structures, skills, and/or style using principles, rather than pre-programmed responses, to decide what to do next and b) adapt instruction accordingly".

This kind of diagnosis and adaptation, which is usually accomplished using Artificial Intelligence techniques, is what distinguishes ITSs from Computer-Assisted Instruction.

The ultimate goal of any ITS is to achieve the same results as a human tutor, addressing what is known as the "2 sigma problem" [6], the name for the experimental result that students who were tutored on an individual basis achieved test scores that were two standard deviations better than students who were exposed only to the typical classroom experience. Since resource constraints rarely allow for students to benefit from individual human tutoring, ITS research has focused on ways to simulate this type of personalized instruction. Some ITSs have concentrated on replicating the guided instruction provided by one-to-one tutorial interactions, while others have focused on different educational paradigms, such as open learning environments.

Advocates of systems of the first type believe that students learn best when presented with a set of focused activities, on which they work under the supervision of a computerized tutor or coach [4][40][46]. Although systems of this type differ in the strategies they use to provide this supervision, such as the nature and timing of feedback they provide on
the students' performance, all follow the principle that the tutor should be controlling the interaction.

Advocates of open learning environments (also known as discovery worlds and exploratory learning environments) place less emphasis on learning through explicit instruction and more emphasis on providing the learner with the opportunity to explore an instructional environment, acquiring knowledge of concepts in the learning domain in the process. Students are allowed to explore these environments by experimenting with different items in the environment's interface. The experiments often involve a simulation where students are able to manipulate different aspects and parameters in order to observe the effects these changes have on outcomes of the simulation. The idea is that through performing a meaningful set of experiments, the students should come to understand relationships between concepts in the domain by generalizing the results of their experiments. Apart from the emphasis on experimentation, open learning environments differ significantly from tutor-controlled ITSs in that they place the onus on the student to initiate these experiments.

For instance, in Smithtown [39], whose target domain is microeconomics, students are provided with a set of variables and given the opportunity to hypothesize on the relationships between these variables, which the students can verify by manipulating the values of those variables. An example of a different approach would be SCI-WISE [49], which focuses less on domain-specific experimentation and more on helping students understand the stages of effective scientific inquiry as they undertake a number of different research projects.

Benefits derived from open learning environments are twofold. First, through active involvement in the learning process students can, in theory, acquire a deeper, more structured understanding of concepts in the domain [39] [43]. Second, these environments provide students with the opportunity to develop meta-cognitive skills associated with effective exploration [31]. These meta-cognitive skills (i.e., skills pertaining to how to learn that are independent of the underlying domain) include hypothesis formation, the ability to construct meaningful experiments and self-monitoring (the ability to monitor one's
progress and understanding). In addition, although not directly mentioned in the literature, self-explanation would seem to be another meta-cognitive skill relevant to learning in an open environment. Self-explanation [10] refers to a student’s tendency and ability to spontaneously generate explanations to themselves as they are exposed to material in a learning context. While the term “material” has typically been associated with written text, self-explanation also applies in an open learning environment since the students will be exposed to instructional material that is not accompanied by tutor-generated explanations. Thus, to successfully learn from the material, students must be able to generate their own explanations.

1.2 Problems with Open Learning Environments

While there has been increasing evidence of the effectiveness of tutor-controlled environments (e.g., [2], [14], [13], [25], [26]), empirical evaluations of open learning environments have yielded mixed results. The most significant predictor of success in open learning environments appears to be the students’ activity levels, since learners who are more active explorers have been shown to benefit more from environments that provide less structure [31][39][38]. An important finding from the point of view of this research is that learning in these environments depends on a number of user-specific traits, such as meta-cognitive skills and knowledge level.

Whether or not students possess the necessary meta-cognitive skills influences how much they benefit from these environments [42][35][39]. Students interacting with open learning environments have been found to have problems with the scientific inquiry process, including difficulty formulating hypotheses, performing experiments and drawing conclusions based on the results of their experiments [18]. Other common problems uncovered by empirical evaluations include not knowing how to generalize results of their exploration, and not being able to self-monitor [42]. The inability that many students have to self-explain [36][10] would also result in students having difficulty generating the necessary explanations about the phenomena they observe to be able to draw conclusions and to generalize results.
Another user-dependent feature that influences students' ability to learn in open learning environments is their domain knowledge. Lower-achieving students tend to have more difficulty in open environments, while indications of the opposite have been found in more restricted environments [34], [35]. For instance, domain knowledge has been found to affect a student's ability to employ certain exploration strategies in a given context [23] [27]. Two common exploration strategies are top-down, where students start with a hypothesis and then perform experiments to verify that hypothesis, and bottom-up, where students perform a series of experiments and gather the data to draw conclusions. The top-down strategy does not work for students without enough domain knowledge to begin formulating a hypothesis, while the bottom-up strategy is ineffective for students who do not have a broad enough understanding of the domain to perform a range of different experiments [27].

Finally, complete coverage of the exploration space can also be problematic without any guidance. In environments with large exploration spaces, students often fail to uncover all the important concepts in the domain [35]. This could be related to a number of factors, including motivation, self-monitoring and domain knowledge.

Systems whose evaluations have yielded fairly positive results include Sherlock 2 [24], a simulation environment for trouble-shooting avionics, MACROSIM [20], a discovery environment targeted at economics, and SCI-WISE. Both Sherlock 2 and MACROSIM, however, are more restricted than pure open learning environments. Sherlock 2 provides scaffolding, restricting the user's interactions within the environment by providing a very structured set of activities. In a study of the MACROSIM environment, students were given an explicit list of tasks to complete. Furthermore, the MACROSIM evaluation tested learning differences between a group who received normal classroom instruction as well as the opportunity to use the system, and a group that was exposed only to the classroom instruction. Finally, a study of SCI-WISE found positive learning outcomes for all students, with a much larger gain for low-achieving students, but the evaluation did not have a control group [49]. Although these evaluations provide some hope that open learning environments can be successful, their results do not refute any of the findings discussed in this section.
Both Sherlock 2 and MACROSIM removed a key feature of open learning environments, which is the freedom students have to initiate the experiments. Also, since the SCI-WISE and MACROSIM evaluations did not include control groups, it is not possible to determine whether the positive learning outcomes should be attributed to the environments or to the additional time that the students were exposed to the educational material outside of the normal classroom instruction.

1.3 Student Modelling

The problems uncovered in empirical evaluations of open learning environments indicate that additional support is needed to make the environments beneficial for all users. One way of providing this support is to supply each student with feedback on the exploration process throughout the interaction. Since the sense of control and freedom that open learning environments provide to the user has the potential to be very beneficial for learning, it is important to interrupt to provide this feedback only when warranted. Augmenting these environments with a student model is fundamental to determining when and how sensible feedback should be provided.

User modelling is the process of building data structures and inference mechanisms that allow an application to assess certain properties of its user and tailor the interaction accordingly. Examples of relevant properties include i) behaviour patterns, ii) cognitive states such as knowledge, preferences and goals, iii) non-cognitive states such as emotions and personality traits. Equipped with this information, the application can adapt its behaviour to meet the needs of the user in an informed manner. Student modelling is sometimes thought of as a sub-problem of the user modelling problem [22], where the target application is an ITS. In a way, however, student modelling is a more difficult problem since no assumptions can be made concerning the user’s knowledge level, which is constantly changing. Student models are used in ITSs for a number of purposes, including to decide when and how to advance the student through the curriculum, offer advice (both solicited and unsolicited), generate problems and activities, and provide tailored explanations [45].
1.3.1 Types of Assessment

So far, the majority of student modelling work has focused on assessment during problem solving. One such assessment is of the student's knowledge of concepts in the domain (e.g., [11]). A second common form of assessment is of the quality of a student's solution steps (e.g., [5]) which can also be used to assess the student's mastery of the rules used to generate the solutions (e.g., [3]). There are many challenges involved with assessing this type of behaviour. One challenge is to determine if a mistake was due to a slip or to incomplete knowledge. On the other hand, when the student demonstrates some correct behaviour there is the possibility that she was just guessing [28]. Other challenges include recognizing the student's solution path (plan recognition) and the uncertainty involved in judging the quality of the solution.

Recently there have been new initiatives in student modelling that have gone beyond knowledge assessment. One of these initiatives has been to model meta-cognitive skills, such as the student model for effective example studying in [14]. Modelling meta-cognitive skills brings a new set of challenges since assessing this type of skill requires access to information that cannot be found in the student's final answer or the steps that lead to that final answer. In addition to knowledge, this type of assessment also requires information on factors such as the student's focus of attention and her reasoning process.

1.3.2 Bandwidth

As illustrated by the examples mentioned above, each type of student model assessment requires different kinds of information about the student. The less of this pertinent information that the model is able to obtain directly through the student's interaction with the system, the more uncertainty there is in the modelling process. The issue of the amount and quality of information about the student that is available to the model is referred to as the bandwidth issue [45]. When trying to assess a student's knowledge during problem solving, a high bandwidth situation would mean that the model would be able to view not only the steps of the student's solution, but also the mental processes that she used to generate these
steps. This has the lowest degree of uncertainty since the model would be able to directly use the correctness of the student's solutions along with the domain principles she used to generate her solutions to determine which set of concepts and rules she understands. In a low bandwidth situation, the model might only be able to view the student's final answer, requiring the model to infer the student's understanding of the concepts involved in the problem from very limited information.

High bandwidth is more difficult to come by when modelling a skill such as self-explanation. Rarely would the model be provided with direct information on factors such as the amount of time the student spends reasoning about different parts of the material and the quality of this reasoning process. Thus, the bandwidth is often increased by either designing a restricted interface or by asking the student enough questions. For instance, in Conati and VanLehn's [15] assessment of self-explanation, they provided a restricted interface to obtain information on what portion of the interface the student was focusing her attention. This data was obtained by masking regions of interest and forcing the students to click on the masks to read the underlying material.

The bandwidth issue creates special challenges for student modelling in open learning environments. As is the case with effective self-explanation, assessing effective exploratory behaviour requires modeling factors that are not easily observable. With open learning environments, however, designing a restricted interface or disrupting the user in any way would remove some of the freedom and control that has the potential to be very beneficial.

1.4 Thesis Approach: Modelling Effective Exploratory Behaviour

To address the concern that open learning environments are not beneficial to all students, this research concentrates on the development of a student model aimed at assessing the effectiveness of the student's exploratory behaviour. Having a model that can sense when the student is experiencing difficulty with the exploration process will allow the environment
to alter the interaction to target the causes of difficulty.

Modelling students’ exploration has not been extensively researched. Open learning environments are problematic from a student modelling point of view because of two factors. The first factor is the uncertainty involved in assessing the student’s behaviour in a low bandwidth situation through an unrestricted interface that gives the model little information about the student’s mental processes and intentions. The second challenge is dealing with the unpredictability of the student’s exploration path, since there is no pre-defined sequence of activities or tasks for the student to complete. To cope with the uncertainty, the model proposed is based on Bayesian Networks [33]. The unpredictability is handled by dynamically constructing portions of the Bayesian Networks depending on the particular exploration path that the student takes.

The Student Model for exploration has been implemented in the context of the Adaptive Coach for Exploration (ACE) [7], an intelligent exploratory learning environment for the domain of mathematical functions. ACE uses the assessment of the Student Model to provide tailored feedback in the form of hints targeted at guiding and improving the students’ exploration of the material provided throughout the course of the interaction.

1.5 Thesis Goals and Contributions

The primary objective of this thesis is to investigate how to incorporate student modelling into open learning environments as a mean of making the environments beneficial for all types of learners through the addition of intelligent support. To meet this objective, the thesis has the following goals:

1. To create a student model that is suitable for providing tailored feedback on the exploration process. This entails:

   (a) Investigating what features are needed in a student model to be able to assess the effectiveness of a student’s exploratory behaviour.

   (b) Creating a student model that can accurately assess these features during the
student's interaction with the environment.

2. To demonstrate the benefits of adding intelligent support to open learning environments using the assessment of a student model.

Most research on student modelling to date has focused on much more structured activities, such as problem-solving and question-answering tasks. Student modelling for these types of activities has the advantage that there is a better definition of correct behaviour, making it easier to recognize this behaviour and also to formalize it in a model. The same cannot be said for effective exploratory behaviour. In addition, student modelling in problem-solving domains, such as algebra and physics, has the advantage that solutions are often broken down into a number of distinct steps, which widens the bandwidth in a non-disruptive manner since it is natural to force students to articulate these steps. Forcing students to articulate the steps of their exploration clashes with the unrestricted nature of open learning environments.

The satisfaction of the goals mentioned above represents an interesting challenge. The low bandwidth, as a result of the type of skill being modelled and the inherent unrestricted nature of open learning environments, results in a great deal of uncertainty in the assessment task. The originality of this work lies in the fact that there has been very little work done on how to monitor exploration and how to provide tailored feedback in an open learning environment, despite the mounting evidence that such support is needed.

The thesis also contributes to research on the use of Bayesian Networks in student modelling. Although Bayesian Networks are well suited for diagnosis involving large degrees of uncertainty, they have yet to be employed in open learning environments. Another contribution is to extend work being done in student modelling to build Bayesian Networks dynamically at run-time.
1.6 Outline

Chapter 2 reviews previous work related to supporting exploration in open learning environments and student modelling using Bayesian Networks. Chapter 3 discusses the ACE environment, concentrating mainly on the Graphical User Interface and the Coach. Chapter 4 presents the Student Model that assesses the effectiveness of the user's exploratory behaviour within the ACE environment. Chapter 5 presents the results of a user study. Changes made to the student model based on findings from this study are discussed in chapter 6. Chapter 7 presents conclusions and plans for future extension.
Chapter 2

Related Work

This chapter reviews previous work related to creating a student model that assesses effective exploration in an open learning environment. Relevant work includes other approaches to supporting exploration in open learning environments as a contrast to the approach taken in this thesis. In addition, this chapter presents work related to student modelling using Bayesian Networks.

2.1 Support in Open Learning Environments

Section 1.2 described several difficulties that students have in open learning environments. Research aimed at helping students overcome these difficulties has followed two main approaches. The first approach has been to augment the environments with cognitive tools. The second approach has been to provide the students with more explicit, tailored support.

2.1.1 Cognitive Tools

Cognitive tools are designed to help scaffold the application of cognitive skills relevant to open learning environments. They include tools to help students with individual parts of the scientific inquiry process, such as hypothesis formation, and tools that help students apply other relevant meta-cognitive skills, such as self-monitoring and reflection. Example
cognitive tools include the Hypothesis Scratchpad in 4SEE [44], which is designed to help students structure their hypotheses, and the Fill-In Forms in [31], designed to help students monitor their progress as they explore. Other examples of cognitive tools are those that are designed to support reflection, including the graphical trace tool in [37] and the Reflection Assessment Phase in SCI-WISE [49] where peers (and sometimes teachers) evaluate each other's work.

Not all designers have formally evaluated the effectiveness of their cognitive tools. Those that have (e.g., [44] and [31]), however, have found that even carefully designed tools can sometimes interfere with the learning process. This is especially true when all learners are required to use the tools, even those who already possess and apply the targeted skills.

2.1.2 Tailored Support

This approach, which is the one taken in this thesis, supports students in open learning environments by providing explicit, tailored feedback. Because of the difficulty in monitoring student behaviour in such unrestricted environments, few other systems have followed this approach.

Veermans and Van Joolingen [48] claim that full learner modelling in open learning environments is impossible, and that the modelling should focus only on allowing the system to help the learner search the hypothesis and experiment spaces. Their approach focuses on evaluating and supporting the students' abilities to confirm or reject hypotheses by assessing the steps of their experimentation. Unlike ACE, this approach assumes that students will be active enough to generate hypotheses and perform the experimentation necessary to verify these hypotheses. In addition, since only the correctness of the students' decisions to confirm or reject is judged, the system cannot support students who are having difficulty covering the exploration space by making suggestions as to which hypotheses would give them a more complete understanding of the domain.

In [47], Veermans et al. evaluated the effectiveness of their intelligent help against a control group that received canned, untailored feedback. Although they did not find a
significant difference between the post-test scores for the two groups, they did find differences in the way the two groups used the environment. They found that students in the experimental group spent more time working on a task, performed more different experiments, and focused more of their experiments to the specific task that they were working on. They also found that the cognitive strengths the students brought to the interaction affected their ability to benefit from the tailored feedback. The pre-tests included questions designed to measure students' intuitive knowledge about the relationships between variables. Students in the experimental condition who scored higher on these questions tended to score higher on the post-tests. Since the same result was not obtained for the control group, this indicates that intuitive knowledge enabled students to better understand how to make use of the tailored support. Both the usage differences between the experimental and controls groups and the individual differences in being able to successfully learn from tailored support further indicate the potential benefits of having a model of exploration that could diagnose individual difficulties with the exploration process.

In Smithtown [39], where students are able to perform experiments involving variables in the domain of microeconomics, a distinction is made between the “exploration phase” and the “experiment phase”. In the exploration phase, the student is gathering information and making observations about variables in the particular example economy. In the experiment phase, the student forms hypotheses, manipulates variables and draws conclusions pertaining to the laws of supply and demand. In the latter phase, the system supports the students' discovery by guiding them through a fixed sequence of steps that involve generating hypotheses followed by a set of experiments to verify the correctness of those hypotheses. During this process, the system supplies intelligent feedback on the correctness of the students' steps. Smithtown is equipped with a large knowledge base of rules related to effective inquiry behaviour that help the system provide this intelligent feedback. Unlike ACE, however, Smithtown does not assist the student in the exploration phase and does not address the needs of students who were unable to perform experiments in the first place.
With Belvedere [32], students construct diagrams of their scientific inquiry behaviour, an activity that allows them to understand the relationships among the various components of the inquiry process. Components in the diagram represent different parts of the inquiry process, such as the hypothesis, the data, and the theory. Components can be connected by arcs with labels such as "supports", "explains" and "conflicts", that explain the relationships between the components. The environment's intelligent support provides advice on the syntactic correctness of the diagrams. For example, a student may have indicated in her diagram that some of the data explain hypotheses. In this case, this system would analyze the diagram and inform the student that data support hypotheses, but do not explain them. The system also provides advice related to the completeness of the diagrams, such as suggesting that the diagram include a hypothesis. Belvedere does not, however, attempt to monitor or understand the student's exploration process.

Finally, Hypadapter [21] has a user model to support exploratory learning in a hypertext system. Information in the model includes user characteristics such as curiosity level and material presentation preferences, which are obtained from a questionnaire, and knowledge level which is determined based on the links that the user has followed. The model is used by the system to restrict the amount of information and links available for the user to explore. The model does not, however, try to capture how effectively the user is exploring the material presented.

The work of this thesis differs significantly from what has been done previously in two ways. First, the model permits the system to target the needs of less active students. Second, once the model detects that a student is experiencing difficulty with the exploration process, the system can guide this process throughout the interaction, not just when the student submits work to be evaluated. The solution proposed here, however, solves only half of the problem. The model can detect when the student is not exploring effectively, but is not rich enough in features to diagnose the causes of poor exploration, such as lack of motivation or meta-cognitive skills.
2.2 Student Modelling Using Bayesian Networks

Bayesian Networks [33] are directed acyclic graphs whose nodes represent random variables and whose arcs represent direct probabilistic dependencies among variables. Each random variable has a set of values that it can take on, such as True or False for binary random variables. Each node in the network with predecessors has an associated Conditional Probability Table (CPT), while nodes without predecessors have prior probabilities. The arcs in the networks define the dependencies amongst variables in the network, rendering nodes either dependent or independent given evidence (see [9] for a good explanation of independence in Bayesian Networks). The network can be queried at any time to obtain the belief that a given node is a specified value. New evidence can be introduced into the network by setting the values of one or more of the variables in the network to a specific observed value.

In any low-bandwidth situation, interpreting user characteristics such as domain knowledge and meta-cognitive skills based on limited observations of student behaviour involves a great deal of uncertainty. Bayesian networks provide a sound way of modelling and processing this uncertainty, making them very suitable for student modelling. Apart from the ability to formalize the uncertainty in the modelling task, the inferences provided by Bayesian Networks are well suited for student model assessment since the value of every node can be computed given whatever evidence is available. This includes being able to predict effects given causes (i.e., predict students’ actions given information on the students’ features) and vice versa.

2.2.1 Examples of Bayesian Student Models

A number of other ITSs have used Bayesian student models, including [12], [14], [28], [29], [30] and [50]. Much of their use so far, however, has been for knowledge assessment, such as in [29], [28] and [30]. The student model in HYDRIVE [30], an environment in which students learn how to trouble-shoot an aircraft hydraulics system, assesses the students’ knowledge of aircraft components and the strategies they use to fix these components. Mayo and Mitrovic [29] use Bayesian Networks to assess students’ knowledge of SQL, information that
is used to tailor the tutor's curriculum. Finally, the student model in OLAE [28] assesses
the student's knowledge of the laws and rules of physics. The student model in Andes [12],
the successor to OLAE, also performs plan recognition since it can determine which solution
path the student was following.

Recently, there have been some interesting applications of Bayesian Networks that
have extended beyond knowledge assessment. These applications include extending the
student model in Andes to assess how effectively the student is self-explaining [14] and
the work done in the I-Help project, which employs inspectable Bayesian Networks in a
distributed setting [50]. The Student Model in ACE provides another form of innovative
assessment in that it assesses the effectiveness of the student's exploration.

2.2.2 Drawbacks of Bayesian Networks

One problem with using Bayesian Networks is that if the networks are large, their specifica-
tion is a time-consuming process. For this reason, student model research has also inves-
tigated ways to modify and construct Bayesian Networks at run-time. The student model
in Andes [12], constructs the Bayesian Networks directly from problem solution graphs gen-
erated by the problem solver. SModel [50], the distributed student modelling server for
I-Help environment, has a facility to construct Bayesian Networks for its student models
when provided with a XML description of the networks nodes, links and prior probabilities.
The Bayesian Networks in ACE are partially specified by the model designer and then ex-
tended during the interaction according to the curriculum and the student's exploration of
the environment.

Another concern with using Bayesian networks in real-time environments is that
they can be computationally expensive [22]. The fact that they have been used successfully
in several environments indicates that, as long as the networks are not too large, this com-
putational complexity is manageable. The Bayesian Networks in ACE's Student Model are
kept to a manageable size by using two techniques. The first is to divide the model into
two smaller networks and the second is to extend the networks dynamically according to
the specific material that the student is exploring. Both of these techniques are discussed in greater detail in chapter 4.

The final big concern with using Bayesian networks in a modelling application is the issue of how the model designers arrive at the values for the CPTs. One way to deal with this problem is to hand-design the CPTs using informed estimates and to refine these values through empirical evaluations. Another technique involves using machine learning to learn to CPTs from data. ACE follows the first approach.
The Adaptive Coach for Exploration (ACE) [7] is an intelligent open learning environment for the domain of mathematical functions. The primary goal behind the development of ACE was to create an environment to evaluate the benefits of providing tailored guidance to support exploration by adding intelligence (in the form of a user model and coach) to the system. A secondary and complementary goal was to develop an environment that allows students to explore the phenomena associated with mathematical functions in a highly graphical manner. The style and content of the material being presented in ACE is based loosely on Stewart's precalculus textbook [41].

Figure 3.1 illustrates ACE's four modules: the GUI, the Coach, the Student Model and the Knowledge Base. Arrows in the diagram represent the flow of information between the modules. The GUI dispenses information to the Coach and Student Model related to the learner's actions in the environment and receives directives from the Coach pertaining to what material to present. The communication between the Student Model and the Coach is unidirectional: the Student Model provides the Coach with information the Coach needs about the student so that the Coach can make decisions regarding the provision of tailored feedback. Currently, the Coach does not inform the Student Model of the kind of feedback it provides to the student and how the student responds to it. This is potentially a limitation to be addressed in future versions of the system. Since much of the Coach's feedback is
in the form of suggestions, some of which are quite vague, it is first necessary to observe how students tend to respond to this type of feedback before it can be incorporated into the Student Model. Finally, all modules access the Knowledge Base for function-related knowledge.

This chapter provides a brief introduction to ACE's components. The Student Model, the primary focus of this thesis, is discussed in detail in chapter 4. Since the development of ACE was joint work, appendix A gives the greatly deserved credits.

3.1 The GUI

The GUI is designed to allow students to explore numerous aspects of functions. These aspects include the relationship between the input and output of a function as well as the relationship between function graphs and their equations. Figure 3.2 is a screen shot of the complete interface for one of ACE's units. The top-left window is the central place of interaction, in which the learner works on the exercises that ACE provides to explore various function concepts. At the bottom-right of this panel, there is a "Next Exercise" button that
Figure 3.2: The ACE Interface
the student presses to indicate that she is finished exploring the current exercise. The right panel is a set of hypertext help pages that contain instructions on how to use ACE and function-related definitions. The bottom panel displays messages from the Coach. In addition, there is a series of icons in the top-left corner, whose functionalities are discussed in section 3.1.4.

The material that the GUI presents is divided into units and exercises. Units are collections of exercises whose material is presented with a common theme and mode of interaction. Currently, ACE has three units: the Machine Unit, the Arrow Unit and the Plot Unit. Exercises within the units differ in function type and equation.

3.1.1 The Machine Unit

The Machine Unit (fig. 3.3) provides the student with the opportunity to explore the relationship between an input and the output that a given function generates. The student
can explore this relationship by dragging any number of inputs displayed at the top of the screen to the tail of the function “machine” (the arrow shown in fig. 3.3). The machine computes the output and spits it out the other end of the arrow by encasing the output in an animated pink ball. If there are multiple steps involved in the computation (e.g., substitution and algebraic operations), the student must click the “step” button (found directly below the machine) to view the equation being resolved.

### 3.1.2 The Arrow Unit

The Arrow Unit (fig. 3.4) allows the student to explore how a range of inputs is mapped onto a range of outputs. This unit requires more active thought on the part of the students as they must both select which input to experiment with and connect that input to the correct output. This is the only activity within ACE that allows students to demonstrate their understanding directly.
Each input has a “dragball” which can be connected to any of the above outputs. As the student drags an input up to one of the outputs, a connecting line is formed. If the student succeeds in choosing the correct output, the line turns green, otherwise the line turns red. The student can reconnect an input to a different output at any time.

3.1.3 The Plot Unit

The Plot Unit is ACE's most interesting unit in terms of the range of exploration that it permits. The goal of this unit is to have the student gain an understanding of the relationship between the graph of a function and its equation. In doing so, the student should also become familiar with different properties of graphs, including slopes and intercepts.

The student can manipulate the graph by either dragging it around the screen (using the mouse) or by editing the equation box (shown in the bottom-left hand corner of fig. 3.5). Changes in the position of the graph are immediately visible in the function's equation (also
Figure 3.6: The Tools Icons

Figure 3.7: The Lesson Browser

shown in a non-editable form in the top-right hand corner of the screen) and changes in
the equation immediately update the graph. The student can zoom in and out using the
magnifying glass icons to view the graph from a variety of different perspectives.

3.1.4 Tools

At the top of the main interaction window, there is a series of icons, which are magnified in
figure 3.6. The first, second and fourth icons (from left to right) support navigation through
the curriculum, the third icon represents the Exploration Assistant and the last icon is a
calculator tool.
Curriculum Navigation

When the student clicks on the "Next Exercise" button, the system presents the curriculum in a sequential order. However, to remain consistent with the design principles of open learning environments and allow the student as much freedom as possible, there are three additional ways to navigate other than using the "Next Exercise" button. The right and left arrows in figure 3.6 allow the student to skip forward to the next exercise and return to the previous exercise, respectively. The fourth icon (a scroll) opens the Lesson Browser (see fig. 3.7), which lets the student jump directly to any exercise in the curriculum by clicking on it.

The Exploration Assistant

The Exploration Assistant is a cognitive tool that helps students monitor and organize their exploration. It displays and categorizes students' exploratory actions within an exercise in terms of relevant exploration cases (see sec. 3.4.3 for a description of relevant exploration cases). Figure 3.8 shows the tool open for an exercise in the Machine Unit after the student...
3.2 The Student Model

The Student Model monitors the student’s exploratory actions in the environment to produce a probabilistic assessment of the effectiveness of the student’s exploratory behaviour. The assessment technique, which is based on Bayesian Networks, is a primary focus of this thesis and will be discussed in detail in chapter 4.

3.3 The Coach

ACE’s coaching component has two main duties. The first is to build the curriculum and the second is to use the assessment of the Student Model to provide tailored feedback in the form of hints geared towards helping the student explore more effectively.

3.3.1 Curriculum

The Coach builds the curriculum at run-time from a text file that specifies the unit and function type for each exercise. The coefficients and exponents of the function equations can either be specified manually in the text file or generated randomly from within a specified range (also included in the text file). Appendix B shows a sample curriculum file. Currently the content of the curriculum remains fixed once it is built. A long-term goal is to be able to adapt the curriculum dynamically throughout the course of the interaction as the Student Model provides an assessment of the student’s weaknesses.

3.3.2 Hints

The Coach provides two types of hints: exploration hints and navigation hints. Exploration hints provide suggestions to the student on how to improve her exploration and can be obtained on demand as the student explores an exercise by clicking on the “Get Hint” button (located at the bottom of fig. 3.2). Hints are supplied to the student at increasing
levels of detail. The lowest detail level is a generic suggestion to explore the current exercise further. The most detailed hints provide suggestions on exactly what things to try. Figure 3.9 shows an example of three levels of hints for the Machine Unit. The Student Model determines the concept that is the focus of a hint.

If the student tries to move on to a new exercise (either by clicking on the "Next Exercise" button or by using one of the navigation tools) before the Coach feels that the student is ready, the Coach will generate a navigation hint. In these situations the student receives the message displayed in figure 3.10. This message contains a warning along with a suggestion to stay in the exercise and obtain an exploration hint. Since the system is designed to give the student as much control as possible, the student can either choose to follow the Coach's advice or to move on. As with exploration hints, the Coach makes the decision of whether or not to generate a navigation hint based on information obtained from the Student Model.
### Table 3.1: Functions Types and their Attributes

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Equation Form</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>( f(x) = a )</td>
<td>y-intercept</td>
</tr>
<tr>
<td>Linear</td>
<td>( f(x) = ax + b )</td>
<td>y-intercept, x-intercept, slope</td>
</tr>
<tr>
<td>Power</td>
<td>( f(x) = ax^c )</td>
<td>scaling factor, exponent</td>
</tr>
<tr>
<td>Polynomial</td>
<td>( f(x) = a_nx^n + a_{n-1}x^{n-1} + ... )</td>
<td>set of {coefficient, exponent}</td>
</tr>
</tbody>
</table>

### Table 3.2: Function I/O Concepts

<table>
<thead>
<tr>
<th>Function Type</th>
<th>Equation</th>
<th>Substitution</th>
<th>Arithmetic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>( f(x) = a )</td>
<td>simple</td>
<td>simple</td>
</tr>
<tr>
<td>Linear</td>
<td>( f(x) = ax + b )</td>
<td>simple</td>
<td>simple</td>
</tr>
<tr>
<td>Power</td>
<td>( f(x) = ax^c )</td>
<td>simple</td>
<td>complex</td>
</tr>
<tr>
<td>Polynomial</td>
<td>( f(x) = a_nx^n + a_{n-1}x^{n-1} + ... )</td>
<td>complex</td>
<td>complex</td>
</tr>
</tbody>
</table>

### 3.4 The Knowledge Base

For the purpose of this system, the function domain is limited enough that it does not require an extensive knowledge engineering effort. Other math domains, such as first-order calculus, require the student to possess extensive background knowledge already, including having mastered mathematical functions. This is problematic because not only would this background knowledge have to be modelled in the knowledge base, but students also often lack this background knowledge, making it difficult for the Student Model to determine whether the student is having difficulty understanding the concepts being presented or is missing the prerequisite knowledge. Since ACE was designed to test the effects of adding intelligent support, it is not desirable to have this confounding variable.

This section highlights some of the key features of the knowledge base. The system currently fully supports four different types of functions: constant functions, linear functions, power functions and polynomial functions. Each function type has an associated set of attributes, concepts and exploration cases.
3.4.1 Function Attributes

Function attributes, described briefly in table 3.1, are basic properties of a function's equation and graph. The definition of each function in the Knowledge Base contains methods to compute or access these attributes.

3.4.2 Function I/O Concepts

Function I/O concepts are basic operations that a student must be able to perform to generate the correct output for a particular function. These concepts include being able to perform arithmetic operations and substitution, which are further divided into simple and complex arithmetic and substitution, based on the complexity of the function. The I/O concepts for each function are summarized in table 3.2.

3.4.3 Relevant Exploration Cases

Relevant exploration cases are the salient concepts that should be explored in each exercise in order to gain a thorough understanding of the target material. In the Plot Unit with a constant function, for example, the student should explore how the graph looks with both positive and negative intercepts.

The rules given in figure 3.11 define which exploration cases are relevant for each function type within each unit. Table 3.3 describes the abbreviations found in the rules. For example, the first rule states that, in the Machine Unit, if the function is a constant

Table 3.3: Symbols for the Exploration Cases

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>S+</td>
<td>Small positive number</td>
</tr>
<tr>
<td>S-</td>
<td>Small negative number</td>
</tr>
<tr>
<td>Z</td>
<td>Zero</td>
</tr>
<tr>
<td>L+</td>
<td>Large positive number</td>
</tr>
<tr>
<td>L-</td>
<td>Large negative number</td>
</tr>
<tr>
<td>Pos_Neg</td>
<td>Positive and Negative version of the same number.</td>
</tr>
</tbody>
</table>
Machine Unit

ConstantFunction \rightarrow S+, S-, Z, L+, L-
LinearFunction \rightarrow S+, S-, Z, L+, L-
PowerFunction \rightarrow S+, S-, Z, Pos_Neg
PolynomialFunction \rightarrow S+, S-, Z, Pos_Neg

Arrow Unit

ConstantFunction \rightarrow S+, S-, Z, L+, L
LinearFunction \rightarrow S+, S-, Z, L+, L-
PowerFunction \rightarrow S+, S-, Z, Pos_Neg
PolynomialFunction \rightarrow S+, S-, Z, Pos_Neg

Plot Unit

ConstantFunction \rightarrow \text{positive intercept, negative intercept}
LinearFunction \rightarrow \text{positive intercept, negative intercept}
\hspace{1cm} \text{positive slope, negative slope, zero slope}
PowerFunction \rightarrow \text{even exponent, odd exponent, positive}
\hspace{1cm} \text{shift, negative shift, large positive scaling,}
\hspace{1cm} \text{small positive scaling, large negative}
\hspace{1cm} \text{scaling, small negative scaling, zero}
\hspace{1cm} \text{scaling}

Figure 3.11: Rules for the Relevant Exploration Cases
Figure 3.12: The Knowledge Base’s Representation of a Linear Function in the Arrow Unit

function then the student should explore how the function behaves with small positive numbers (S+), small negative numbers (S-), zero (Z), large positive numbers (L+) and large negative numbers (L-). The rule is the same for a linear function, while with power and polynomial functions the students should see how the function behaves with a positive and negative version of the same number (Pos.Neg in fig. 3.11), since a function equation with even exponents will generate the same output in both cases. Large numbers are not included as relevant exploration cases for either of these functions since they could cause some students to get fixated on complicated arithmetic unnecessarily. The Arrow Unit has the same set of rules since it also deals with the relationship between input and output.

In the plot unit, the rules specify which function manipulations the students should explore. For a constant function, they should view the graph with both positive and negative intercepts. The rule for linear functions builds on this, adding also positive and negative slopes along with the zero slope, which turns the function back to a constant one. The relevant exploration cases for a power function include shifting (which results in the student viewing the graph at a variety of intercepts), scaling (which changes the width and orien-
tation of the graph) and odd and even exponents (which change the shape of the graph). Polynomial functions were not implemented at the time this research was done, but are planned for future versions of ACE.

Figure 3.12 displays the representation of a linear function for the Arrow Unit in the Knowledge Base. Although it may not be ideal, this method of knowledge representation is sufficiently detailed to allow ACE's other modules (the Coach, GUI and Student Model) to carry out their responsibilities in an efficient manner. In particular, the representation of the relevant exploration cases is an important part of the Student Model's assessment of effective exploration behaviour, the focus of the next chapter.
Chapter 4

Student Model - Version I

The first version of ACE's Student Model was built keeping in mind two primary objectives. The first was for the Student Model to generate an assessment of a student's exploration of the ACE environment that would allow the Coach to provide tailored feedback aimed at guiding and improving this exploration when necessary. The second goal was to attempt to use the student's exploratory actions as a means of assessing the student's knowledge of concepts in the domain, since effective exploration should enable the student to gain better understanding of the explored concepts. This chapter examines the techniques used by the first version of the Student Model to generate its assessment. The second version of the Student Model is discussed in chapter 6.

4.1 Uncertainty in the Modelling Task

Modelling exploratory behaviour in an open environment is not an easy task. Maintaining a natural, unrestricted interaction style gives the student sufficient freedom to explore, but providing this type of freedom lowers the bandwidth of information available to the Student Model. The Student Model has access to low-level information such as mouse clicks and keystrokes, but not to higher-level information such as the student's intentions and cognitive states. Without such information, there is considerable uncertainty involved in the
assessment process. To manage this uncertainty, the Student Model uses Bayesian Networks. As described in section 2.2, Bayesian Networks provide a computational framework that allows the uncertainty involved in a modelling task to be formulated and processed in a principled way.

Modelling exploratory behaviour in a situation with limited bandwidth is particularly filled with uncertainty. The Student Model can view with which items in the interface the student is experimenting. This information, however, is not always sufficient to allow the Student Model to determine what the student was trying to accomplish with these interface manipulations. Furthermore, this information does not always give any indication of what the student understands, making it harder for the Student Model to determine directly if the student's actions are contributing to exploration that is effective enough to help them learn the material. The model will sometimes have access to more information than other times; using Bayesian Networks will allow the model to leverage dynamically any available information.
4.2 Structure of the Student Model’s Bayesian Networks

The Student Model is separated into two Bayesian Networks: one for the input/output units (Machine and Arrow) and another for the graph manipulation unit (Plot). Separating the model into two networks helped to both divide the initial design task into two more manageable components and address some performance concerns. Updating the probabilities in Bayesian Networks is an NP-hard problem [17], where the running time increases with the number of nodes and arcs. The advantages of having one large network are that all information about the student is available to the coaching component throughout the session and there is no overhead involved in switching between networks. Having multiple smaller networks, on the other hand, reduces the running time associated with updating (or querying) a node during the session because of the smaller subset of arcs and nodes involved in the computation. In ACE, the skills and behaviour being modelled in the Machine Unit and the Arrow Unit have virtually no overlap with those modelled in the Plot Unit. Thus, the Coach does not require access to both sets of information at one time. Dividing the model into these two separate networks allows the system’s performance to remain reasonable throughout the session, despite that the fact the networks grow with each new exercise (a feature that is discussed in subsequent sections). The two networks have a common high-level design, which this section describes.

4.2.1 High Level Description of the Model’s Bayesian Networks

Each network in ACE’s Student Model consists of two classes of nodes: exploration nodes and knowledge nodes. Exploration nodes represent the effectiveness of the student’s exploratory behaviour. These nodes are present in the network at several different levels of granularity: the exploration of individual exploration cases within an exercise, of individual exercises, of groups of related exercises (units), of groups of concepts that appear across multiple exercises and overall exploratory behaviour. Each exploration node can take on the
value of either True or False. A True value means that the student has sufficiently explored the item associated with the node (i.e., the function, concept, unit or relevant exploration case).

Figure 4.1 provides a high-level description of the interactions among the different classes of nodes. Assessment at different levels of granularity allows the coaching component to provide a wide range of tailored feedback, some of which is already in ACE. The assessment of individual exploration cases and groups of concepts can be used to select the content of hints within an exercise. The assessment of how well the student has explored groups of concepts and related exercises can be used to adapt the curriculum and to provide suggestions as to which exercises to explore. Finally, the overall assessment can be used to provide general, high-level feedback on the student's exploration strategies (or lack thereof). Currently, ACE's Coach does not fully support the two latter types of feedback but, because the relevant assessments are already provided by the Student Model, it could easily be extended to do so.

Knowledge nodes represent the student's level of understanding of the relevant domain concepts that she demonstrates during the interaction. Exploration nodes are used in conjunction with evidence of correct behaviour to produce an assessment of the student's understanding (see fig. 4.1). Like exploration nodes, knowledge nodes are also binary variables, where a True value represents the probability that the student understands that concept. Evidence of good exploratory behaviour carries less weight than does evidence of correct skill application, as is reflected in the Student Model's Conditional Probability Tables. In ACE's initial design, however, evidence of correct skill application currently comes only in the Arrow Unit. There was a plan to include additional units that would provide some explicit testing of students' knowledge of all concepts in the domain, but time did not permit all of the elements of the initial proposal to be implemented. Furthermore, any unit that involved a lot of testing would take away some of the feeling of freedom and control that ACE presently permits. Part of the future development of ACE will include adding more activities that test correct skill application while still maintaining the exploratory nature of
ACE's Bayesian Networks are mainly composed of dimensional variables, meaning the nodes form hierarchies, with skills represented at different levels of specificity. According to Jameson [22], there are two possibilities for the direction of the arcs when using variables of this type. The first option is to have the arcs point from the sub-skills to the more general skills (fig. 4.2A). With this option, the sub-skills are independent, unless direct evidence of the general skill can be gathered, and having evidence of one sub-skill does not increase the probability that the student possesses another. The other option has the arcs pointing from the more general skills to the sub-skills (fig. 4.2B). With this option, evidence of the student having one sub-skill influences the probabilities that the student also possesses the other sub-skills (unless the higher-level skill is observed, in which case the sub-skills become independent). HYDRIVE [30] is an example of a system that uses option A; OLAE [28] is a system that uses option B.

Option A was selected for ACE's Student Model; having explored one sub-concept well does not increase the probability of having effectively explored another sub-concept. Option B makes the most sense in situations where the sub-skills are related. Thus, within ACE, the sub-skills are not considered to be related; the model expects the students to explore all aspects thoroughly. The assumption of independence of sub-skills was also made for the knowledge nodes. Furthermore, when necessary, the effect of related sub-skills can still be achieved by adding arcs between them (with a slight performance cost).
4.3 Network Design

4.3.1 Static vs. Dynamic

Each Bayesian Network has a static portion and a dynamic portion. The static portion is completely specified ahead of time by the model designer, meaning it remains identical for every student and session. In addition to this static component, each network has a dynamic portion that is constructed throughout the interaction with the student.

The reason for having part of the network be dynamic is that there are aspects of the modelling process that vary from session to session. First, the curriculum is generated automatically at run-time from a text file and, thus, there could be a different curriculum for each session. Adding nodes dynamically to the network removes any curriculum-dependent information from the static portion, avoiding situations where the model designer has to build a separate network for every conceivable curriculum. In addition to the structure of the curriculum, the exact nature of each exercise is unknown ahead of time since the coefficients and exponents of the function equations may be randomly generated. Finally, using ACE's navigation tools, each student may take a different path through the curriculum, choosing to explore some of the exercises and ignore others. Dynamic additions to the network would be even more essential should curriculum itself become dynamic since virtually no details of a session other than the general concepts presented would be known a priori.

4.3.2 Machine/Arrow Units

Static Portion

Figure 4.3 shows the pre-defined portion of the network for the Machine and Arrow Units. This network contains nodes that represent the student's understanding of concepts that are involved with the ability to generate the input and output of a function such as substitution and arithmetic (see nodes “simpleArithmetic”, “simpleSubstitution”, “complexArithmetic” and “complexSubstitution” in fig. 4.3). In addition, the network contains nodes that represent higher-level effective exploratory behaviour such as the exploration of individual units.
Figure 4.3: The Static Portion of the Bayesian Network for the Machine and Arrow Units (e.g., “arrowExploration” in fig. 4.3) and overall exploratory behaviour (“overallExploration” in fig. 4.3). In all subsequent figures, the nodes that are labeled with “exploration” in conjunction with those labeled “f,” and “fCase,” refer to exploration nodes while all others refer to knowledge nodes.

Dynamic Portion

The portion of the network that is added dynamically (see fig. 4.4) contains nodes representing the exploration of individual exercises, the exploration of relevant exploration cases, and the exploration of categories of concepts. The effectiveness of a student’s exploration of an individual exercise is represented by the nodes labeled “f_i” (e.g., “f_1” in fig. 4.4). These nodes are added to the network every time a student visits a new exercise. In this example, the student has visited three functions: f_1, f_2 and f_3. As function nodes are added to the network, they are linked to their corresponding unit nodes. As illustrated by figure 4.4, “f_1” and “f_2” represent exercises in the Machine Unit and “f_3” is an exercise in the Arrow Unit.

The degree to which the student is considered to have explored an individual exercise is influenced by whether or not she has explored the salient concepts within that exercise.
Figure 4.4: Part of the Dynamic Portion of the Bayesian Network for the Arrow and Machine Units that Contains Exploration Nodes
in addition to how long she spent exploring the exercise. This relationship is accomplished in the network by adding nodes for each of the relevant exploration cases associated with the function presented in the exercise (information that can obtained from the knowledge base) and a node representing exploration time. The relevant exploration case nodes are labeled “$f_i$Case_i” (e.g., “$f_1$Case_1” in fig. 4.4) and the time node is labeled “$f_i$Time” (e.g., “$f_1$Time” in fig. 4.4). The time node represents a measure of how long the student spent exploring the exercise as a whole, not each individual exploration case. A measure of how long the student spent exploring each case would be more informative, but would also be more difficult to determine. How to measure time spent exploring a particular case will be investigated as part of future research.

Since the relevant exploration case nodes are instances of concepts that can appear across multiple exercises, nodes are added to the network representing the more general effectiveness of the student’s exploration of these concepts (e.g., “exploredSmallPos” in fig. 4.4). If an exploration concept is not in the network already, it is added dynamically at the same time as the corresponding relevant exploration case.

The final part of the Arrow/Machine network that is added dynamically (shown in fig. 4.5) deals with the Arrow Unit, the only unit that gives the student the opportunity to directly demonstrate her knowledge. In this unit, nodes representing proficiency in generating correct input and output of a function are added to the network (e.g., $f_3$IO in fig. 4.5). These “$f$IO” nodes are attached to nodes representing the type of substitution and arithmetic needed to be able to generate the correct output for the function in question. When a student connects an input to an output, a new node labeled “$f$IOCase_i” (e.g., “$f_3$IOCase_1” in fig. 4.5) is attached to the “$f$IO” node and set to True if the answer is correct and to False, otherwise.

### 4.3.3 Plot Unit

The structure of the network for the Plot Unit is similar to the Arrow and Machine Units. The difference is that there are more detailed knowledge and exploration hierarchies since
this unit involves a wider range of concepts and, accordingly, a greater potential for exploration.

**Static Portion**

The static portion of the network contains a detailed knowledge hierarchy and a mirrored exploration hierarchy for the concepts associated with the graph of a function. Feeding into each node of the knowledge hierarchy is its corresponding exploration node and a node representing some form of explicit evidence of this skill. The former nodes have the prefix label “test” (e.g., “testPosIntercept” in fig. 4.6). Unfortunately, this type of direct evidence cannot be gathered with ACE’s current set of activities, and so these nodes remained unobserved. Figure 4.6 shows a portion of the network for a linear function. Concepts involved with the graph of a linear function include intercepts (positive and negative) and slopes (positive, negative and zero). There are analogous portions of the network for each of the different types of functions: the more complicated the function, the more complicated the
Figure 4.6: The Static Portion of the Bayesian Network for a Linear Function in the Plot Unit
associated exploration hierarchy. For example, the hierarchy for a constant function consists only of intercept exploration while the hierarchy for a power function contains concepts such as shifting, scaling and exponents.

**Dynamic Portion**

As with the Machine/Arrow network, every time a student visits a new exercise, additional nodes are added dynamically. Figure 4.7 shows an example portion of the dynamic network for the Plot Unit, where the student has visited a constant function ("f1") and a linear function ("f2"). The first node to be added dynamically is the function node (labeled "f1") representing how well the student has explored that particular exercise (e.g., "f1" in fig. 4.7). A subtle difference present in this network is rather than having the unit exploration node influenced by the functions exploration nodes, it is instead influenced by the exploration nodes for each of the different types of functions (see fig. 4.6) since there will only be one exercise associated with each function type. As in the Arrow/Machine network the relevant exploration cases nodes (e.g., "f1 Case") in fig. 4.7) and a time node (e.g., "f1 Time" in fig. 4.7) feed into each function node. The exploration case nodes represent specific instances of concepts at the bottom level of the exploration hierarchy, such as the exploration of positive and negative intercepts ("exploredPosIntercept" and "exploredNegIntercept" in figs. 4.7 and 4.6).
4.4 Evidence

The nodes that are observed as evidence are the exploration case nodes (e.g., “fiCase1” in fig. 4.4), time nodes (e.g., “fiTime” in fig. 4.4) and, in the case of the Arrow Unit, nodes for the generation of input and output (e.g., “f3IOCase3” in fig. 4.5). Time nodes are observed as True when the time that the student has spent exploring an exercises passes over a pre-specified threshold. This threshold is unit-dependent and was determined by observing people using the system.

The exploration case nodes are all set to False when they are initially added to the network and are not set to True until the student performs interface actions that the system considers to be an indication of the student having explored the associated concepts. Each unit has a different interpretation of what interface actions provide such evidence. In the Machine Unit, dragging an input that is an instance of a relevant exploration case to the Machine is considered evidence of the student having explored that case. In the Arrow Unit, this evidence is introduced when the student drags a relevant input to an output. Finally, the student is considered to have explored a relevant exploration case in the Plot Unit when she either drags and drops the graph to an interesting position (as defined by the relevant exploration cases) or edits the function equation to change the graph in a desired way.

4.5 Conditional Probability Tables

The Conditional Probability Tables (CPTs) in the Bayesian Networks were constructed using estimates developed as a part of this research. As is the case with all human-designed CPTs, such estimates need to be refined further through empirical evaluations. Rather than fully describing the tedious details of the CPTs, this section will highlight some of the important features instead.

As illustrated in table 4.1, the CPTs for the function exploration nodes (e.g., “fi” in fig. 4.4) are set so that the more relevant exploration cases the student explores, the higher is the probability that she has effectively explored that function. The same technique is
Table 4.1: CPT for an Example Function Node fn.

<table>
<thead>
<tr>
<th>fn,Case1</th>
<th>fn,Case2</th>
<th>fn,Case3</th>
<th>fn,Time</th>
<th>fn</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: CPT for the Ability to Correctly Generate the Output for a Simple Function.

<table>
<thead>
<tr>
<th>simpleArithmetic</th>
<th>simpleSubstitution</th>
<th>simpleArrowOutput</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.45</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.45</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0.03</td>
</tr>
</tbody>
</table>

used to update the CPTs for the unit and concept exploration nodes.

For the nodes representing knowledge of concepts in the domain, the CPTs were constructed using estimates of the importance of mastering the associated sub-skills in order to fully understand the higher level skills. Table 4.2 shows an example for the ability to generate the correct output ("rightSimpleArrowOutput" in fig. 4.3) for functions involving simple arithmetic and simple substitution. In this case, both simple arithmetic and simple substitution are seen as having equal importance in mastering the higher level concept. Table 4.3 shows an alternative example for the slope concept ("slope" in fig. 4.6) where the zero slope is of lesser importance.

Finally, tables 4.4 and 4.5 show examples of two CPTs for nodes that depend on
Table 4.3: CPT for Understanding the Slope Concept.

<table>
<thead>
<tr>
<th>posSlope</th>
<th>negSlope</th>
<th>zeroSlope</th>
<th>slope</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.8</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.6</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>T</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.03</td>
</tr>
</tbody>
</table>

Table 4.4: CPT for Understanding the Concepts in Arrow Unit Given Exploration and Direct Evidence

<table>
<thead>
<tr>
<th>simpleArrowOutput</th>
<th>complexArrowOutput</th>
<th>arrowExploration</th>
<th>arrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.8</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.2</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>T</td>
<td>T</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>0.03</td>
</tr>
</tbody>
</table>

Table 4.5: CPT for Understanding Positive Intercepts given Exploration and Direct Evidence

<table>
<thead>
<tr>
<th>exploredPosIntercept</th>
<th>testPosIntercept</th>
<th>posIntercept</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>0.6</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>0.03</td>
</tr>
</tbody>
</table>
some explicit evidence of the student understanding the corresponding concept and the exploration of that concept. In both cases, if the student has only explored the concept well, the network is less confident that she understands that concept but the network still uses good exploration as some indication of understanding. An alternative would be to use any explicit negative evidence as an indication that the student definitely does not understand the concept. However, it is possible that she may be deliberately exhibiting incorrect behaviour as part of her experimentation.

4.6 Implementation

The Student Model is written in Java and the Bayesian Networks were built using the JavaBayes Tool [16]. The JavaBayes tool comes with a GUI that allows the user to create networks and to perform operations on nodes in the networks such as observing and querying. In addition, the tool can store networks created with the GUI to a text file. Minor modifications were made to the original JavaBayes code to permit a different application, like ACE, to load a network from one of these text files and use the inference engine at run-time, as well as to allow the application to alter the structure of the network.
This chapter presents the results of a user study conducted with ACE in October and November of 2000. The study, although limited in a number of ways, provides some initial support for the effectiveness of the ACE learning environment in promoting learning. In addition, the study provides some evidence of the benefits of this particular approach to supporting exploratory behaviour using tailored feedback with the help of the Student Model's assessment described in chapter 4.

5.1 Study Goal

The goal of the study was to evaluate the effectiveness of ACE, including the accuracy of the Student Model, to verify the hypothesis that intelligent support in an open learning environment can help to make the experience more beneficial for all types of learners. In addition, analyzing the students' behaviour as they used the system would give insight into ways to improve the interface, the Coach interventions and the assessment of the Student Model.
5.2 Participants

Initially the plan was to conduct the study with ACE's target population, which is high school students. Because of unforeseen difficulties in coordinating with a local high school, the study was run instead using university students taking a computer literacy course. Although not the target population, these subjects were still considered suitable since the teaching assistants for the course felt that their students had very limited math knowledge.

The study participants were volunteers from a first year computer literacy course offered by the Computer Science Department at the University of British Columbia. This course introduces students to computers and application programs and does not serve as an introduction to computer science. The participants were screened further so that those selected had not taken a university math course at the 200 level or above and had not taken any math within the past year. The total number of subjects was 14: 10 females and 4 males. All subjects signed a consent form and were paid a $20 stipend for their participation.

5.3 Experiment Design

Initially the plan was to conduct a 2-group experiment. One group would use the complete ACE environment; the other would act as a control group, using the ACE environment with the tailored support disabled. This type of design would be a convincing means of evaluating the effectiveness of adding intelligent support to an open learning environment since learning and usage differences could be compared with and without intelligent support. This type of design would also provide support for including a student model to assess exploratory behaviour, since the intelligent support is based on this assessment.

Ultimately, the decision was made to abandon the 2-group design. Several students who volunteered for the study did not show up for their sessions, leaving only 14 subjects. Dividing such a small subject pool into two groups was unlikely to give any reliable information on ACE's effectiveness. As a result, all the subjects were pooled into one group that used the full ACE environment. More useful information could likely be obtained by
Table 5.1: Experiment Schedule

<table>
<thead>
<tr>
<th>Activity</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-test phase</td>
<td>20 mins</td>
</tr>
<tr>
<td>ACE session</td>
<td>45 mins</td>
</tr>
<tr>
<td>Post-test phase</td>
<td>15 mins</td>
</tr>
</tbody>
</table>

analyzing how usage of ACE relates learning in one large group than could be obtained from
a two-group design with two small groups. This decision was also affected by the fact that
several subjects turned out to have better knowledge than expected, thus causing a ceiling
effect on the pre-test.

The study took place in a research lab in the Computer Science Department at the
University of British Columbia. At most three subjects took part in the study at one time.
Subjects were instructed not to communicate with each other and dividers were placed
between the computers so that they couldn’t see each other. There was always an observer
present in the room who remained in the background.

As illustrated in table 5.1, the duration of each session was at most 80 mins and
consisted of a pre-test phase, a session with ACE and a post-test phase. The pre-test
phase included a paper and pencil test designed to gauge students’ knowledge of the topics
targeted by ACE. The test (found in appendix D) consists of 39 questions, divided equally
into questions on function output recognition and generation, graph property recognition,
equation property recognition and graph-equation correspondence. The pre-test phase also
included 13 qualitative questions designed to gain information on the students’ tendency
to explore, their prior computer experience and previous math course. In addition, the
subjects wrote a “Need for Cognition” test [8]. This is a well-respected test of an individual’s
tendency to engage in and enjoy cognitive activities and was used in the study to see if there
was any relation between this measure and a student’s tendency to explore. The post-test
phase consisted of a ACE topic test constructed to be similar but different to the one given
to subjects in the pre-test phase and a nine item questionnaire targeted at the students’
subjective view of their ACE experience (see appendix E for a copy of the post-test).
Table 5.2: Information Recorded in the Log Files

<table>
<thead>
<tr>
<th>Interaction Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of exercises visited</td>
<td>The number of exercises the student chose to explore.</td>
</tr>
<tr>
<td>Number of exercises passed</td>
<td>An exercise is passed if ACE let the student leave an exercise without a navigation hint.</td>
</tr>
<tr>
<td>Number of stay events</td>
<td>A stay event occurs when a student follows a navigation hint and remains in the current exercise.</td>
</tr>
<tr>
<td>Number of leave events</td>
<td>A leave event occurs when the student does not follow a navigation hint and chooses to move on.</td>
</tr>
<tr>
<td>Number of exploration hints</td>
<td>The total number of exploration hints accessed by the student.</td>
</tr>
<tr>
<td>Average hint level</td>
<td>The average hint level accessed by the student. The hint levels range from 0-2 with level 0 being the most general hint and level 2 being the most specific hint.</td>
</tr>
<tr>
<td>Number of exploratory actions</td>
<td>Actions performed by the student that the Student Model uses as evidence of the student having explored a relevant exploration case.</td>
</tr>
<tr>
<td>Lesson Browser usage</td>
<td>The number of times the student jumped to an exercise using the Lesson Browser.</td>
</tr>
<tr>
<td>Exploration Assistant usage</td>
<td>The number of times the student opens the exploration assistant.</td>
</tr>
</tbody>
</table>

5.4 Data Collection Techniques

Each session was observed by one researcher who recorded informal observations on an observation sheet (see appendix C). Several sessions were also captured on videotape. In addition, ACE was instrumented to produce log files that capture the sessions at a finer level of detail. Table 5.2 summarizes the key interaction events captured in the logs.

5.5 Results and Discussion

Since it was not possible to run a 2-group study, linear regression analysis was used as a data analysis technique to verify if there is any correlation between different aspects of ACE usage and student learning. The following is a summary of the analysis performed on the
event counts summarized in table 5.2.

5.5.1 Effect of ACE on Learning

To verify that ACE triggers any learning at all, the pre-test and post-test means were compared and a significant difference was found between the two \( (p = 0.013) \). Regression analyses were also performed with a number of event counts. Each analysis involved the given event count and the pre-test score as independent variables and the post-test score as the dependent variable.

Using the students’ improvement scores as the independent variable and the event count as the dependent variable would not have been as reliable a method of evaluating how each event count affects learning. There are a number of problems associated with using this method in educational research, which are summarized in [19]. One of these problems relates to ceiling effects, which is particularly relevant to this analysis since 8 of the 14 subjects showed a near ceiling effect, scoring 90% or higher on the pre-test. Ceiling effects restrict the distribution of potential improvement scores for different levels of initial knowledge, since higher ability students will not be able to improve as much. Another problem is that not all improvement intervals have the same meaning. A small improvement for a student with high pre-test scores is often much more meaningful than the same improvement for a student with low pre-test scores since improvement in this interval can mean that the student had mastered the more difficult domain concepts. Instead, [19] suggests using pre-test scores and the experimental treatment as predictors in the regression model and post-test scores as the independent variable. Having pre-test scores as the first predictor in the model explains the portion of the variance in post-test scores that is due to prior ability, leaving the event count to explain what is left of that variance.

Because of the small sample size, no more than two independent variables could be added to the regression model while still obtaining results that were statistically significant. Pre-test score is always a significant predictor of post-test score.

The following event counts were found to be significant positive predictors of post-
test score (after controlling for pre-test score):

1. Total number of exploration hints accessed \[ p = 0.0406, R^2 = 84.6\% \]

2. Number of exercises passed \[ p = 0.0093, R^2 = 87.9\% \]

The first result provides an initial indication that ACE’s support of the exploration process, in terms of exploration and navigation hints generated using the Student Model’s assessment, improves learning. The second result provides initial evidence that the Student Model’s assessment reflects students’ learning in the environment, since the Coach determines that a student has passed an exercise by querying the Student Model for relevant probabilities.

The above results could also be attributed to additional factors that were not controlled for, including the student’s general academic ability and conscientiousness. The fact that there is not a significant correlation between the two event counts supports the supplied interpretation, but a study that controls for these factors would be required to rule out this possibility.

The total number of exploratory actions was not found to be a significant predictor of learning. This may be because of how the exploratory actions were recorded in the log. Every interface action that indicated exploration of a relevant exploration case was counted. The Student Model, however, uses only one such action to set a relevant exploration case node to True. Exploratory actions after this point are considered by the Student Model to be redundant. In fact, a few cases of students performing redundant explorations were observed. This inability to self-monitor is consistent with one of the problems of open learning environments [42]. Some students may not have the self-monitoring skills necessary to understand when they have explored a concept sufficiently and as a result begin to over-explore that concept rather than move on.

5.5.2 Effect of Student Characteristics on ACE Usage

Two sources of information were used to try to capture a student’s tendency to explore: the “Need for Cognition” (NFC) test and the qualitative portion of the pre-test. Results of
past evaluations of open learning environments suggest that students who are more active explorers benefit more from this style of learning (e.g., [39]). Whether or not students are active explorers might be influenced by certain personality traits, such as whether they are exploratory by nature. If so, the Student Model should be able to identify learners who are exploratory by nature to help the Coach adapt the instruction accordingly.

There weren't any significant results for the qualitative pre-test questions, which is hard to interpret since these questions were hand-crafted and may not be suitable for capturing the desired information. For the NFC, there was one significant result. The number of exploratory actions was found to be a negative predictor of the NFC score \( p = 0.0285, R^2 = 34.0\% \). The interpretation of this result is that the larger the number of exploratory actions a student performed, the lower was their “Need for Cognition”. This result seems somewhat surprising at first, but it may just be an indication of the over-exploration problem mentioned earlier and of the fact that students with higher NFC scores exhibit more controlled exploration. Or, the NFC might not be a good measure of exploratory tendency.

5.5.3 Subjects’ Perception of ACE

These statistics were obtained using the qualitative portion of the post-tests. Two statistically significant results came from this analysis:

1. How helpful the students found the hints to be, when controlling for pre-test scores, was a positive predictor of the post-test scores \( p = 0.0339, R^2 = 85\% \). This indicates that when students expressed that they liked the hints supplied by ACE this was not simply a product of a “desire to please”.

2. How helpful the student found the hints was also a positive predictor of the average level of hint they accessed \( p = 0.0267, R^2 = 34.7\% \). This is not surprising since the higher level hints contain more specific suggestions. However, even a generic hint, a suggestion to explore further, was enough to elicit further exploratory behaviour in many students.
5.6 Student Model Accuracy and Limitations

The log files captured several of the Bayesian Network probabilities after a student visited an exercise. These probabilities included those for function exploration, unit exploration, exploration of any relevant concepts, and for any pertinent knowledge nodes. None of these probabilities related significantly to the post-test scores, the NFC or the subjective questions on the pre-test. The only significant statistic concerning the Student Model, discussed in section 5.5.1, was that the number of exercises passed was positively correlated with post-test scores (controlling for pre-test scores). This result is encouraging since the Coach uses the Student Model's assessment of the effectiveness of the student's exploration of that exercise and its associated concepts to determine if a student has passed an exercise. It is disappointing, however, that none of the probabilities in the Bayesian Networks nor choosing to follow or disregard the Coach's navigation hints (stay or move on events) were significant predictors of post-test scores.

Observing students during the sessions and going through the log files manually afterward uncovered some limitations of the first version of the Student Model that could potentially explain some of the lack of results. The more significant limitations include the Student Model's treatment of knowledge levels and how it assesses whether or not a student has effectively explored a relevant exploration case.

5.6.1 Knowledge Levels

Although the study participants were not the target population, having several highly knowledgeable students use the system uncovered one significant flaw in the model. Students who already understand certain concepts should not be expected to explore these concepts as thoroughly as someone with lower knowledge levels. In fact, some of the subjects who appeared to be able to self-monitor would often explore only concepts they did not completely understand. The first version of the Student Model did not take this issue into account, the Machine and Plot units do not test a student's knowledge of relevant concepts directly. Instead the Student Model interpreted signs of inactivity in certain exercises as poor ex-
ploratory behaviour, causing the Coach to intervene when the students tried to leave. This would lead to unnecessary stay events for high ability students, making it hard to correlate this event count with the post-test scores.

5.6.2 Exploration of Relevant Exploration Cases

Currently, the Student Model looks only at interface actions to determine if the student has explored a particular exploration case. This limitation was especially apparent in the Plot Unit. There were a few students who performed several of the desired interface actions in this unit, but did not end up learning the underlying concepts, while others learned these concepts after minimal exploratory actions. For example, a couple of students experimented with several negative and positive slopes, but not did learn the difference between the two, while others caught on after experimenting with only one of each. These observations suggest that effective exploration is more than just performing the appropriate interface manipulations; the students must also pay careful attention to the results of those actions. For example, in the Plot Unit it is possible that students who did not learn from their actions were moving the function graph around, but did not look at the function equation. Lacking the skills or motivation to self-explain could potentially impede learning in these situations.

5.7 Proof-of-Concept

Part of the hypothesis of this thesis is that it is beneficial to include a student model that can assess exploratory behaviour in an open learning environment because of the individualized form of tailored support that it permits. There are several issues to be resolved in terms of how best to assess exploration and what form this tailored support should take. This study may not provide a concrete test of the hypothesis, but anecdotal evidence of students’ behaviour using ACE does highlight the importance of developing a model to assess exploration and provides evidence of what kinds of features that such a model should include. To illustrate this point, the following subsections summarize a few key examples of students
using the system. The first case shows the need to elicit exploratory behaviour from passive learners. The next two cases involve students failing to uncover all of the concepts available for exploration.

5.7.1 Case 1: Passive/Timid Learner

On the pre-test, this student exhibited a lack of understanding of constant functions. Her exploratory behaviour was adequate in the Machine Unit, but very poor in the Arrow Unit. In this unit, when she was presented with a constant function, she tried one input, got it wrong, stared at the screen for a length of time and then tried to leave. When the Coach issued a warning she remained in the exercise, stared at the screen for another length of time and then chose to move on despite the Coach's second suggestion to stay.

This example illustrates that some learners are passive and/or timid in an open learning environment, and as a result, fail to improve. Providing hints upon request may not be the solution for these learners (she did not access a hint despite the Coach's suggestions that she do so) since students do not always access intelligent support when needed [1]. Thus, it is even more important to have a student model that can identify timid learners and allow the system to provide more proactive, unsolicited guidance.

5.7.2 Case 2: Failing to Uncover Concepts

This learner was very active in the Plot Unit, but failed to improve on post-test questions targeted at the zero slope for linear functions. He experimented with all the relevant exploration cases in the Plot Unit except this one. The Coach generated navigation hints when he tried to leave the exercise. He chose to stay in the exercise, but did not request a hint from the Coach. Although he remained an active explorer, he did not end up experimenting with a zero slope. This example shows the potential benefit of having a model that can assess the exploration of individual concepts. This assessment influenced the Coach's decision to provide a navigation hint when this student tried to leave the exercise, and to provide hints at increasing levels of details targeted at the zero slope, had the student requested them.
User-initiated hints might not always be an adequate means of eliciting more thorough exploration. Furthermore, determining the appropriate hint level should be informed by the Student Model, since a generic suggestion to explore further may not be helpful to students who have explored all but the more subtle concepts. This is also consistent with the finding that students do not always benefit from high-level hints [1]. There were three other participants whose behaviour fits this description.

5.7.3 Case 3: Succeeding in Uncovering Concepts

This student scored poorly on the pre-test questions involving negative slopes. He did not experiment with negative slopes in the Plot Unit until he received a navigation hint from the Coach. At this point he asked for more hints and obtained one targeted at negative slopes. As a consequence, he performed the appropriate explorations and improved on the post-test. The benefit of tailored hints was also very noticeable in two other participants.

Despite the study’s limitations, it provided some insight into how to improve the Student Model to make it more accurate. These changes are discussed in the next chapter.
Chapter 6

Student Model - Version II

As discussed in section 5.6, ACE's pilot study uncovered some limitations of the first version of the Student Model. This chapter discusses the changes made to the model to address some of these problems; others are left as future work. Results of an evaluation of the updated Student Model conducted using a small group of subjects similar to the participants in the original study are also presented.

6.1 Changes to the Network Structure

Figure 6.1 presents a high-level description of the second version of the Student Model, while figure 6.2 shows a simplified example Bayesian Network that would be generated during an ACE session. In this simplified example, the student is working on exercises in the Machine and Arrow Units and has visited three exercises as indicated by the \( f_1, f_2, \) and \( f_3 \) nodes. To simplify the diagram, there are only two types of functions (constant and linear) and the details of the relevant exploration case nodes (e.g., “\( f_1 \text{Case}_1 \)” in fig. 4.4) and the specific exploration concept nodes (e.g., “\( \text{smallPosInput} \)” in fig. 4.4) have been omitted.

The major changes to the network structure (shown in fig. 6.1) include the incorporation of knowledge into the assessment of effective exploration and the inclusion of another category of exploration nodes that assess how effectively the student is exploring the more general
concepts targeted by ACE, such as how effectively the student is exploring the relationship between the input and output for each of the different types of functions (constant, linear, power and polynomial). In addition, as illustrated in the example Bayesian Network, the similarity of exercises is now taken into account and the time nodes have been removed.

6.1.1 Knowledge Nodes

As discussed in the previous chapter, the first version of the Student Model did not take the student's knowledge of the material covered by ACE into consideration, assuming that all students should explore the material to the same extent. Thus, even students who already had high knowledge often received warnings from the Coach as they navigated through the curriculum. These unnecessary warnings do not follow ACE's design principle that tailored feedback should be provided only to students who are experiencing difficulty.

The proposed solution to this problem uses the student's knowledge as part of the assessment of how well that student is exploring. If the student has high knowledge of a concept, she should not be expected to explore that concept as thoroughly. If the student chooses not to focus on a concept that she already understands, the Student Model should
Figure 6.2: A Simplified Example of the New Bayesian Network for the Machine/Arrow Units
still consider this to be effective exploration since it demonstrates the student's ability to reflect on her own understanding. On the other hand, the Student Model still expects students with low knowledge to explore thoroughly.

In the second version of the Student Model, components of the Bayesian Networks that represent exploration nodes are now influenced by the student's knowledge (see fig. 6.1). More specifically, as shown in figure 6.2, every exploration node is now influenced by one or more knowledge nodes. These knowledge nodes have a different meaning than those in the first Student Model. In the first version of the Student Model model, knowledge nodes were updated throughout the interaction using explicit evidence of the student's knowledge of concepts in addition to the student's exploration of those concepts. Because there is very little explicit evidence of the students' knowledge as they use ACE (only in the Arrow Unit), the majority of these nodes could essentially be used only as a means of determining how effectively the student was exploring. In the second version of the Student Model, the knowledge nodes represent either the students' knowledge of concepts prior to exploration, or knowledge assessed through correct behaviour in ACE's less exploratory activities. Explicit evidence on these nodes can currently be obtained only in the Arrow Unit.

Since exploration nodes are now influenced by more than just the student's interface actions, their meaning has also changed since the first version of the model. A True value for an exploration node represents the probability that the student has effectively explored that item, using a broader definition of what effective exploration means. The Student Model believes the student to have explored a concept effectively if she would not benefit from further exploration of that concept. For a student not to benefit from further exploration of a concept means that the combination of the student's exploration actions and her knowledge level has caused her to understand that concept. A low probability means that the student should explore that concept more thoroughly to increase her understanding.

In figure 6.2, the nodes "constantFuncIOEx" and "f_1" are two examples of exploration nodes influenced by knowledge nodes. The node "constantFuncIOEx" is influenced by the corresponding knowledge node, "constantFuncIO", while "f_1" is influenced by "sim-
pleArithmetic", "simpleSubstitution" and "constantFuncIO" because the exercise involves all of these concepts. Table 6.1 shows a typical CPT for an exploration node. The exploration node, $f_1$, has a high probability value (currently set to 0.8) if all of the associated knowledge nodes are True. If one or more of the associated knowledge nodes are False, the probability that the student would benefit from further exploration of that exercise is once again determined by the student’s activity level. The more exploration cases the student explores, the higher is the probability for the function exploration node. If all of the knowledge nodes are true and the student continues to explore, the probability increases slightly with the added exploration.

6.1.2 Additional General Exploration Concepts

The first version of the Student Model provided an assessment of students’ exploratory behaviour within particular exercises with the nodes in the network that represent the relevant exploration cases and the individual functions. The Student Model also assessed the student’s behaviour over a longer period of time with the nodes representing unit exploration, related exploration cases appearing in multiple exercises, and overall exploratory behaviour. The pilot study also uncovered the need to maintain a long term assessment of concepts that span a number of exercises and/or units but do not correspond to specific exploration cases. For example, a number of subjects thoroughly explored most of the exercises in the arrow and machine units with the exception of constant functions. This type of information, however, could not be obtained from the original network since constant function exploration was not explicitly represented as one of the relevant exploration cases.

The model now has additional nodes that represent the student’s exploration of these more general exploration concepts; these include the exploration of the input/output relationship for each type of function and how effectively the student is exploring the concepts of arithmetic and substitution. These exploration concepts are not associated with particular exploration cases, but with exercises or units. Depending on the generality of these concepts, the associated exploration nodes are influenced either by how well the student
Table 6.1: An Example CPT for an Exploration Node in the Second Version of the Student Model

<table>
<thead>
<tr>
<th>constant</th>
<th>simple Substitution</th>
<th>simple Arithmetic</th>
<th>$f_1$ Case$_1$</th>
<th>$f_1$ Case$_2$</th>
<th>$f_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>0.03</td>
<td></td>
</tr>
</tbody>
</table>
explores related exercises (e.g., “constantFuncIOEx” in fig. 6.2) or how well she explores related units (e.g., “substitutionEx” in fig. 6.2).

6.1.3 Similarity of Exercises

The first version of the Student Model did not take the similarity of exercises into account in its assessment of effective exercise exploration. If a student explores one exercise thoroughly and then chooses to explore a similar exercise, she should not be expected to explore this second exercise as thoroughly. To address this concern, arcs were added between functions nodes (see “f1”, “f2”, and “f3” in fig. 6.2). The strengths of the arcs are determined by similarity scores, which are computed based on the unit and function types.

6.1.4 Time Nodes

The final change consisted of removing the times nodes. How well a student explored an exercise did not seem to depend on time spent in the exercises but how long they spent reasoning about their actions and how long they spent exploring each individual exploration case. Thus, for the time being, the time nodes have been removed from the network and will be added again as part of the future research which incorporates a more sophisticated view of what it means to explore a relevant exploration case effectively.

6.2 Evaluation of Changes

6.2.1 Study Goal

The new Student Model was evaluated with a small group of subjects. A larger participant group was sought but was not obtained because of a lack of interested participants and time constraints. The goal of the study was to investigate how the changes in the model influenced the subjects’ interactions with the system. Ideally, the assessment of the new Student Model should result in the Coach intervening less frequently with the high ability students without causing the Coach to ignore the students who are experiencing difficulty.
Also, the model should be able to detect situations in which students are systematically failing to explore general concepts, such as the input and output of specific function types.

6.2.2 Study Participants

Since this study was designed to address some of the limitations uncovered by the pilot study, participants with similar levels of math ability were sought. No computer literacy course was being offered at the time of the study, but the recruited subjects were similar to the previous ones since they had also graduated from high school and had not taken any university math courses. A total of five subjects participated in the study.

6.2.3 Experiment Design

The second study used essentially the same experimental design as the pilot study (see sec. 5.3). The only difference was that the second version of the Student Model requires some initial indication of the student’s knowledge. This information was obtained from the pre-tests. After the students wrote the pre-test and before they used ACE, the pre-tests were marked and used to set the values of the knowledge nodes. If it was obvious that the student understood a particular concept, the corresponding knowledge node was set to True. If they obviously did not understand the concept the node was set to False. If there was insufficient evidence or conflicting evidence, the node remained unobserved, with the prior probabilities set to \( P(True) = 0.5 \). An alternative to setting the nodes to either True or False would be to adjust the prior probabilities. This approach will be investigated in the future.

6.2.4 Results

The log files of the students' interaction with ACE were analyzed to see how changes made to the model influenced the number of navigation hints generated by the Coach. The changes were intended to reduce unnecessary interruptions, without the model becoming too lenient. Thus, to evaluate these changes, the old and new logs were analyzed by hand for two event counts: the number of unnecessary navigation hints and the number of premature passes.
Unnecessary navigation hints were considered to occur when the Coach generated a navigation hint despite it being clear from the pre-test that the student understood the concepts associated with that exercise. A premature pass occurred when the Student Model determined that the student passed the exercise, but the student did not appear to understand the associated concepts on the post-test. The modifications to the model do not address the problem of the model overestimating the students' exercise exploration, so the count of premature passes was not expected to change significantly from the previous experiment.

Tables 6.2 and 6.3 show the results of the analysis. As desired, the new Student Model resulted in a dramatic reduction in the percentage of navigation hints that were deemed to be unnecessary (from 38% with the old model to 5% with the new model). This is an important reduction since, with the previous version, students were receiving an average of 4.4 unnecessary interruptions per session while they received an average of fewer than one with the new model. The number of premature passes did rise slightly (from 4% to 9%). This indicates that the second model still overestimates that students' exploratory
behaviour: when students perform a large number of exploratory interface actions, the model assesses good exploratory behaviour even though some of these students do not learn from their exploration. The results of this study provide further evidence that the model needs to place more weight on the amount of time the student spends reflecting on each of her exploratory actions and on the student’s ability to self-explain the phenomena that she observes in the environment to accurately assess effective exploration. The elimination of the time nodes and incorporating exercise similarity also could have caused part of the increase in premature passes since both changes resulted in the model becoming more lenient.

The study did not provide an evaluation of the general exploration concept nodes. None of the participants in this study chose to ignore any particular high-level concepts, as was the case in the first study. The coaching component also needs to be extended before this change can be fully evaluated. Currently, the Coach makes suggestions within an exercises and warns the student as she leaves an exercise. The Coach does not, however, supply exercise navigation hints that try to steer the student towards exploring exercises that target the higher-level concepts that she has not effectively explored.
Chapter 7

Conclusions and Future Work

This thesis presented research on creating a student model that can assess the effectiveness of a student's exploratory behaviour in an open learning environment. This work addresses a substantial limitation of open learning environments; previous empirical evaluations have found that students' ability to learn in these environments depends on certain user-specific features that influence their ability to explore effectively. The Student Model introduced here permits the provision of tailored feedback on a student's exploration process in an attempt to make the environment beneficial for all learning styles. It does so by monitoring the students' actions in the environment unobtrusively to maintain the unrestricted nature of open learning environments.

7.1 Satisfaction of Thesis Goals

7.1.1 Model for Exploration

The primary goal of the thesis was to create a model suitable for the provision of tailored feedback on the exploration process. This goal involved two parts: i) to determine what type of features the model should include, ii) to build a model that is capable of assessing these features during students' interaction with the ACE environment.

The model's features were determined through an iterative design process. The
model was first constructed based on an initial hypothesis of how to assess effective exploration. This hypothesis was tested using an evaluation with human subjects. The results of this evaluation were then used to design a revised model, which was once again evaluated.

The model was created using Bayesian Networks to handle the large amount of uncertainty involved in monitoring student behaviour in an open environment. The type of uncertainty present in the assessment process stems from the freedom the environment gives the student and the lack of information available concerning the reasons behind the students' actions and how these actions contribute to effective exploration. The model performs its assessment by monitoring the student's exploration of the relevant exploration cases present within each exercise. Nodes corresponding to these relevant exploration cases form the basis of the rest of the assessment, including the effectiveness of the student's exploration of exercises, of units, of concepts, and of the student's overall exploratory behaviour. Exploration nodes are influenced by a set of relevant knowledge nodes, incorporating the student's existing knowledge into the assessment of effective exploration.

The model also had to deal with the fact that ACE's curriculum is built at runtime and that each student can choose to take a different path through the curriculum. To address this problem, the model constructed portions of the Bayesian Networks dynamically over the course of the interaction. These portions involved the function nodes and nodes for each of their associated relevant exploration cases.

The Student Model is suitable for the provision of tailored feedback because it allows the environment to sense when and what the student is not exploring effectively. When this occurs, the Coach can guide the exploration process using the Student Model's assessment of which concepts and exercises have yet to be sufficiently explored.

7.1.2 Evaluation of the Model and Intelligent Support

The second goal of the thesis was to evaluate the effectiveness of providing intelligent feedback that supports the exploration process. Two user evaluations were performed. The first was designed to provide a sense of how ACE's support affects how students use and
learn from the environment. The second evaluated the changes made to the model after the first study to improve its accuracy. A number of key results and observations from these evaluations validate the Student Model’s design and provide indications that the type of feedback ACE supplies is beneficial to students. In particular:

- The more hints the students accessed, the more they improved.

- The number of exercises that the Student Model felt that the students had effectively explored was positively correlated with the students’ improvements.

- Some students are inactive despite low knowledge.

- Some students are unable to discover all important domain concepts without guidance.

The first result provides support for the addition of Student Model-tailored feedback since the content of the hints was tailored to concepts that the Student Model felt were insufficiently explored. The second result is an initial indication that the Student Model is able to accurately assess effective exploration. The final two points are observations from the studies that illustrate why it is important to have a student model that can identify students who are not actively exploring and that can generate a type of assessment that allows the Coach to give them specific suggestions targeted at improving their exploration. These studies also uncovered additional factors to include in future versions of the model to improve its assessment and diagnostic capabilities. These factors are described in section 7.3.

7.2 Generalizability

The Student Model was tailored specifically to the ACE environment, but its high-level design is generalizable to any open learning environment with a set of activities and exploration concepts of interest within those activities. Using the model’s framework would require defining the static portion of the network (both the structure and the CPTs) and building a knowledge base that could identify the exploration cases and higher-level concepts
that are relevant to each activity. In addition, the environment would have to decide what behaviour patterns signify effective exploration of the relevant exploration cases.

7.3 Limitations

Despite being subjected to an iterative design process, the proposed Student Model still has two main limitations. First, it has a limited view of what it considers to be effective exploration of a relevant exploration case. As demonstrated by the user evaluations, the current interface actions alone may not be sufficient to determine if the student has effectively explored a concept. Other factors that seem to be important include whether or not the student is attending to the effects of her exploratory actions, and whether or not the student is reflecting on and generating self-explanations about the material she is exploring.

The second main limitation in design of the Student Model is that, although it assesses the effectiveness of the student’s exploratory behaviour, it is not sufficiently rich in features to diagnose the causes of poor exploration. Results of many empirical evaluations of open learning environments show that effective exploration depends on a number of factors, including meta-cognitive skills, exploration strategies and, potentially, motivation. Currently the model can inform the Coach if the student is not exploring effectively and allows the Coach to provide specific suggestions as to what elements need further exploration. The model does not, however, allow the Coach’s feedback to target the underlying causes of the poor exploration.

Finally, the evaluations of ACE provide indications of the accuracy of the Student Model and the benefits of providing tailored feedback. Since the evaluations did not have control groups, however, it is not possible to determine exactly what features of ACE contributed to these positive learning outcomes. Untailored Coach instructions or the GUI by themselves could potentially be responsible for the obtained outcomes.
7.4 Future Work

7.4.1 Exploration of Relevant Exploration Cases

To fully assess effective exploration, the model needs to perform a more sophisticated assessment of what it means for a student to explore a relevant exploration case effectively. Currently, the model bases this assessment on only the student's interface actions. The findings of past evaluations of open learning environments and the work done in this thesis suggest, however, that the exploration process depends on a number of factors, including knowledge of exploration strategies, domain knowledge, motivation, and meta-cognitive skills, such as self-explanation, self-monitoring and reflection. In addition, factors such as the student's emotional state and personality traits could also be relevant, especially to modelling motivation. To assess effective exploration properly these additional factors should be considered by the Student Model.

Modeling each additional factor along with the probabilistic dependencies among factors, would allow the model to perform a richer assessment of effective exploration behaviour, as long as the model has evidence of some of these factors. ACE's current interface can provide very little information about these additional factors. Tracking self-explanation, which seems to be one of the most relevant factors, requires, among other things, information on the student's focus of attention.

Knowing where the user is focusing her attention would greatly increase the model's ability to interpret the user's actions as evidence of good exploratory behaviour. It may be possible that a student's interface actions indicate good exploration but that she is not attending to the results of that exploration. For example, in ACE's graph manipulation activity, the student may be actively manipulating the graph but not examining the changes this causes in the function equation. In this case, the student would fail to gain an understanding of the relationship between a function graph and its equation. Additional variables modelling the user's attention could be added to the networks and updated using eye-tracking.
Even with the additional information from the eye-tracker assessing self-explanation and other meta-cognitive skills relevant to effective exploration, requires more information on the student's thought processes than is currently provided by ACE's interface. The challenge will be to re-design the interface to provide more information without taking away the sense of freedom and control that ACE currently permits.

7.4.2 Extending the Model's Diagnostic Abilities

Having the Bayesian Networks include variables on the additional factors discussed in section 7.4.1 would also extend the Student Model's ability to diagnose causes of poor exploration. As is the case with the current Student Model, querying the exploration nodes would allow the environment to sense when the student is experiencing difficulty with the exploration process. When the student isn't exploring effectively, the environment could query variables related to factors influencing this assessment, such as self-explanation and domain knowledge, to obtain a probabilistic estimate of the specific causes of the student's difficulty. This would permit the environment to tailor its feedback to directly address the most likely causes of difficulty.

7.4.3 Formal Evaluation

Additional formal evaluations are needed to evaluate both the effects of providing tailored feedback and the Student Model's role in this process. The first planned evaluation involves comparing learning and usage differences between an experimental group that would use the full ACE environment and a control group that would use only the ACE interface, with the Coach and Student Model disabled. To evaluate the role of the Student Model, a similar study could be conducted with both groups receiving the Coach's feedback. The experimental group, however, would receive feedback tailored to the assessment of the Student Model, while the control group could receive feedback either on demand or based on an untailored strategy.
7.4.4 Automated Testing

Eventually, this research aims to automate the administration of the pre-tests and post-tests. Currently these tests are used both to evaluate ACE and to allow the Student Model to obtain an estimate of the students' knowledge of function concepts prior to using the system. This process is not only time-consuming for the marker, but also requires that the students wait for the pre-tests to be marked and for the values of knowledge nodes to be set. Having ACE administer the tests would allow the Student Model to immediately use the results to set the values without the need for human assistance.

7.5 Conclusion

Apart from open learning environments, there are numerous other computer applications that require their users to explore autonomously. The fact that some users are able to explore these environments while others are not, creates a problem. If the environments are too open, some users will not be able to explore effectively. On the other hand, if they are too restricted, other users will get upset at the lack of freedom and control. The Student Model proposed in this thesis is a first step in the solution to this problem. It can help the environment distinguish between these two classes of users and helps to facilitate the provision of adapted support.
Bibliography


Appendix A

Programming Credits

ACE was developed jointly with Kasia Muldner and Michael Huggett. Kasia Muldner was primarily responsible for designing and implementing the Coach, while Michael Huggett built most of the GUI, with Kasia programming the Browser Tool and the Exploration Assistant. In addition, both Kasia and Michael were heavily involved in the first ACE user study.
Appendix B

Curriculum File

Figure B.1 is the curriculum that was used for both ACE user evaluations.
Figure B.1: A Sample Curriculum File
Appendix C

Observer Sheet

<table>
<thead>
<tr>
<th>Participant:</th>
<th>Date:</th>
<th>Time:</th>
</tr>
</thead>
</table>

Comments: (H=Help, M=Mood, E=Exploration, S=System, O=Other)

<table>
<thead>
<tr>
<th>Category</th>
<th>Time</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H/M/E/S/O</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix D

Pre-Test

Part 1: Questionnaire

1. A package gets dropped off at your doorstep, containing a brand new scooter. The only problem is that assembly is required. You
   (a) read the manual carefully, and then commence assembly
   (b) skim the manual quickly, and start to put the thing together
   (c) ask an expert to help you do it
   (d) trash the manual and start right away - you will figure it out as you go

2. When working with others, you
   (a) are very vocal about your ideas - you often have a good idea of how to do things
   (b) contribute some ideas, but don't like to be the main person in charge
   (c) are mainly quiet - you prefer for others to take charge

3. When learning how to use a new computer application for an assignment, you
   (a) only figure out use those things that are totally necessary to get the assignment done
   (b) figure out how to use it to get the assignment done, plus learn a few extra things that caught your attention
   (c) fully explore the application - you are very curious about the various things you may discover in there

4. When it comes time to decide on what and how to do a school project, you
   (a) prefer to have the teacher tell you
(b) prefer to make one up yourself
(c) a mix of a and b: the teacher initially helps, but you have the final say

5. When learning something, you like to
(a) get an overall picture of the thing you are learning, and then figure out the details
(b) figure out the details first, and then get an overall picture
(c) a mix of a and b
(d) other:

6. You are in a strange new city. You
(a) get a map right away and use it frequently to plan your routes
(b) get a map but use it only when absolutely necessary
(c) don't bother with a map - if lost, you'll ask someone

7. You play computer games
(a) almost every day
(b) once or twice a week
(c) less often that once a week

8. You use the Web to find information that you need (like a phone number, shopping, etc)
(a) almost every day
(b) once or twice a week
(c) less often that once a week

9. On a scale of 1-5, where 1 means very much, you like to surf the web
1) Very much 2) so-so 3) Neutral 4) not very much 5) not at all
Part 2: Recognizing the output of a function

For the following function equations specify the output of the function with the given inputs. Please show your work in the spaces provided.

1) \( f(x) = 3 \)

   What is the output of \( f(0) \)?
   a) 1           b) 0           c) 3           d) -3

   What is the output of \( f(4) \)?
   a) 4           b) 3           c) -4          d) 0

2) \( f(x) = 3x + 2 \)

   What is the output of \( f(4) \)?
   a) 4           b) 5           c) 2           d) 14

   What is the output of \( f(-4) \)?
   a) -4          b) 5           c) -10         d) 2
3) \( f(x) = 5x^2 \)

What is the output of \( f(-1) \)?

a) -1  

b) -5  

c) 5  

d) 1

What is the output of \( f(1) \)?

a) -1  

b) -5  

c) 5  

d) 1

4) \( f(x) = 3x^3 + 2x + 1 \)

What is the output of \( f(0) \)?

a) 0  

b) 1  

c) -1  

d) 6

What is the output of \( f(2) \)?

a) 2  

b) 29  

c) 23  

d) 6
### Part 3: Function Output

For each question, choose the three inputs and connect them to the correct outputs. Choose the three inputs will best help you understand function.

1) \( f(x) = -2 \)
   - Outputs: \(-5, -4, 14, -2, 2, 0, 14, 15, 29\)
   - Inputs: \(-45, -32, -8, -2, 0, 2, 6, 25, 34\)

2) \( f(x) = -2x - 5 \)
   - Outputs: \(-5, -45, 87, -65, -9, 9, -3, -15, 33\)
   - Inputs: \(-46, -19, -7, -1, 0, 2, 5, 20, 30\)

3) \( f(x) = -3x^2 \)
   - Outputs: \(0, -27, -12, -147, 12\)
   - Inputs: \(-3, -2, 0, 2, 7\)

4) \( f(x) = 2x^2 - x + 1 \)
   - Outputs: \(29, 1, 16, 79, 22\)
   - Inputs: \(-6, -3, 0, 3, 4\)
Part 4: Graph Properties

For each question, circle the appropriate response.

1) The y-intercept of the graph in this picture is: Positive/Negative/Zero

2) The y-intercept of the graph in this picture is: Positive/Negative/Zero

3) The slope of the graph in this picture is: Positive/Negative/Zero

4) The slope of the graph in this picture is: Positive/Negative/Zero
5) The slope of the graph in this picture is: Positive/Negative/Zero

6) The exponent in the function equation graph in this picture is: Even/Odd

7) The graph has been scaled by a: Positive Number / Negative Number

8) Which graph has been scaled by a larger number?: Graph A / Graph B
   Graph A
   Graph B
Part 5: Graph/Equation Properties

The function \( f(x) = 1x \) may be best described by the graph:

- a)
- b)
- c)
- d) None of these graphs

The function \( f(x) = -2x + 5 \) may be best described by the graph:

- a)
- b)
- c)
- d) None of these graphs

The function \( f(x) = 4x^2 - 5 \) may be best described by the graph:

- a)
- b)
- c)
- d) None of these graphs

The function \( f(x) = 1x^3 \) may be best described by the graph:

- a)
- b)
- c)
- d) None of these graphs
Part 6: Equation Properties

The function \( f(x) = 3 \) has
a) a positive slope    b) a negative slope    c) a zero slope

The function \( f(x) = 3x + 7 \) has
a) a positive slope    b) a negative slope    c) a zero slope

The function \( f(x) = -1x + 10 \) has
a) a positive slope    b) a negative slope    c) a zero slope

The function \( f(x) = -2x + 5 \) has
a) a positive y-intercept    b) a negative y-intercept    c) a zero y-intercept

The function \( f(x) = 3x - 6 \) has
a) a positive y-intercept    b) a negative y-intercept    c) a zero y-intercept

The function \( f(x) = 5 \) has
a) a positive y-intercept    b) a negative y-intercept    c) a zero y-intercept

The function \( f(x) = -7 \) has
a) a positive y-intercept    b) a negative y-intercept    c) a zero y-intercept
Appendix E

Post-Test

Part 1: Questionnaire

Comments can be added in the spaces provided.

I found the information in the hints helpful (please circle one)
   a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

I understood why the computer suggested I stay in an exercise (please circle one)
   a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

I found it helpful when the computer told me to explore an exercise more than I had (please circle one)
   a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

The amount of guidance provided by the computer was (please circle one)
   a) Not enough  b) OK  c) Just Right  d) A bit too much  e) Way too much
The computer gave me enough freedom to move around between activities (please circle one)
a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

I found the information in the help pages useful (please circle one)
a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

I prefer to learn this way over using a text book (please circle one)
a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

My favorite unit was:
Machine (unit 1)  Switchboard (unit 2)  Plot (unit 3)

I feel that I learned a lot using ACE (please circle one)
a) Strongly agree  b) agree  c) Neutral  d) disagree  e) Strongly disagree

The thing I liked the most about ACE:

The thing I liked the least about ACE:

What I would like to see added to ACE
Part 2: Recognizing the output of a function

For the following function equations specify the output of the function with the given inputs. Please show your work in the spaces provided.

1) \( f(x) = 8 \)

What is the output of \( f(0) \) ?
   a) 5  b) 0  c) 8  d) -8

What is the output of \( f(2) \) ?
   a) 2  b) 8  c) -2  d) 0

2) \( f(x) = 3x + 3 \)

What is the output of \( f(5) \) ?
   a) 18  b) 5  c) 6  d) 14

What is the output of \( f(-5) \) ?
   a) -5  b) 3  c) -12  d) -15
3) $f(x) = 2x^2$

What is the output of $f(-2)$?

a) -1  

b) -4  

c) 8  

d) -8

What is the output of $f(2)$?

a) 2  

b) 4  

c) 1  

d) 8

4) $f(x) = 2x^3 + 3x - 1$

What is the output of $f(0)$?

a) 0  

b) 1  

c) -1  

d) 4

What is the output of $f(1)$?

a) 1  

b) 4  

c) 5  

d) 6
Part 3: Function Output

For each question, choose the three inputs and connect them to the correct outputs. Choose the three inputs will best help you understand function.

1) \( f(x) = 5 \)
   Outputs: -5 -6 11 -2 2 0 5 15 22
   Inputs: -35 -27 -7 -3 0 3 5 23 39

2) \( f(x) = 2x - 4 \)
   Outputs: 4 -40 60 -88 -2 -16 -4 -8 38
   Inputs: -42 -18 -6 -2 0 1 4 21 32

3) \( f(x) = 4x^2 \)
   Outputs: 4 -0 -36 16 -4
   Inputs: -2 -1 0 1 3

4) \( f(x) = 2x^2 - 3x + 1 \)
   Outputs: 4 16 67 56 2
   Inputs: -5 -2 0 2 6
Part 4: Graph Properties

For each question, circle the appropriate response.

1) The y–intercept of the graph in this picture is: Positive/Negative/Zero

2) The y–intercept of the graph in this picture is: Positive/Negative/Zero

3) The slope of the graph in this picture is: Positive/Negative/Zero

4) The slope of the graph in this picture is: Positive/Negative/Zero
5) The slope of the graph in this picture is: Positive/Negative/Zero

6) The exponent in the function equation graph in this picture is: Even/Odd

7) The graph has been scaled by a: Positive Number / Negative Number

8) Which graph has been scaled by a larger number?: Graph A / Graph B

Graph A

Graph B
Part 5: Graph/Equation Properties

The function \( f(x) = 5x \) may be best described by the graph:

- a) 
- b) 
- c) 
- d) None of these graphs

The function \( f(x) = -2x - 6 \) may be best described by the graph:

- a) 
- b) 
- c) 
- d) None of these graphs

The function \( f(x) = 3x^2 + 5 \) may be best described by the graph:

- a) 
- b) 
- c) 
- d) None of these graphs

The function \( f(x) = -1x^3 \) may be best described by the graph:

- a) 
- b) 
- c) 
- d) None of these graphs
Part 6: Equation Properties

The function \( f(x) = -4 \) has
a) a positive slope  \hspace{1cm} b) a negative slope \hspace{1cm} c) a zero slope

The function \( f(x) = -5x + 2 \) has
a) a positive slope  \hspace{1cm} b) a negative slope \hspace{1cm} c) a zero slope

The function \( f(x) = 4x + 6 \) has
a) a positive slope  \hspace{1cm} b) a negative slope \hspace{1cm} c) a zero slope

The function \( f(x) = -4x + 5 \) has
a) a positive y-intercept  \hspace{1cm} b) a negative y-intercept \hspace{1cm} c) a zero y-intercept

The function \( f(x) = 3x - 4 \) has
a) a positive y-intercept  \hspace{1cm} b) a negative y-intercept \hspace{1cm} c) a zero y-intercept

The function \( f(x) = -7 \) has
a) a positive y-intercept  \hspace{1cm} b) a negative y-intercept \hspace{1cm} c) a zero y-intercept

The function \( f(x) = -3 \) has
a) a positive y-intercept  \hspace{1cm} b) a negative y-intercept \hspace{1cm} c) a zero y-intercept